МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кваліфікаційна наукова
праця на правах рукопису

ТКАЧУК ТАРАС ІГОРОВИЧ

УДК 004.31

ДИСЕРТАЦІЯ

ХАРАКТЕРИСТИКИ СКЛАДНОСТІ SH-МОДЕЛЕЙ СПЕЦІАЛЬНИХ ФУНКЦІЙ І ЇХ ЗАСТОСУВАННЯ ДЛЯ ОПТИМІЗАЦІЇ СПЕЦПРОЦЕСОРІВ

Спеціальність 05.13.05 – «Комп'ютерні системи та компоненти»

Галузь знань 05 – «Технічні науки»

Подається на здобуття наукового ступеня кандидата технічних наук

Дисертація містить результати власних досліджень.
Використання ідей, результатів і текстів інших авторів
мають посилання на відповідне джерело Т.І. Ткачук

Науковий керівник:
Дунець Роман Богданович
доктор технічних наук,
професор

Ідентичність всіх примірників дисертації
ЗАСВІДЧУЮ:
Вчений секретар спеціалізованої
вченої ради Д 35.052.08, д.т.н., проф. Луцик Я.Т.

Львів-2018

АННОТАЦІЯ

Ткачук Т.І. Характеристики складності SH-моделей спеціальних функцій і їх застосування для оптимізації спецпроцесорів. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук (доктора філософії) за спеціальністю 05.13.05 – «Комп'ютерні системи та компоненти». Інститут комп'ютерних технологій автоматики та метрології Національного університету "Львівська політехніка" Міністерства освіти і науки України, Львів, 2018.

У першому розділі досліджено особливості сучасних моделей реалізації спеціалізованих комп'ютерних систем, що реалізують спеціальні функції опрацювання сигналів з урахуванням оптимізації значень характеристик складності.

Проведено аналіз основних моделей формальних алгоритмічних систем й показано, що такі моделі не враховують апаратне забезпечення, що не дає змоги, використовуючи їх при проектуванні, оптимізувати спеціалізовані комп'ютерні системи з врахуванням всіх характеристик складності.

Проаналізовано моделі апаратних засобів спецпроцесорів і показано, що така модель дає змогу проектувати комп'ютерні засоби з врахуванням лише двох характеристик – продуктивність та обсяг обладнання, а інші характеристики складності не враховуються.

Проаналізовано моделі апаратно-програмних засобів спеціалізованих комп'ютерних систем М. Черкаського – SH-модель, H-модель, RH-модель. H-модель алгоритму дозволяє розв'язувати задачі лише деякого одного класу, які розрізняються тільки наборами вхідних даних. До моделей такого типу відносяться, наприклад, моделі комбінаційних пристроїв – суматори, матричні пристрої множення, дешифратори та інші. Показано, що такі моделі дають змогу проводити оптимізацію спеціалізованих комп'ютерних систем, що реалізують спеціальні функції, із врахуванням апаратної, часової, програмної та структурної характеристик складності.

Другий розділ стосується оптимізації SH-моделей операції множення, як основи спеціальних функцій опрацювання сигналів.

Удосконалено відомий метод обчислення структурної складності SH-моделей за рахунок виявлення та об’єднання у групи однотипних елементів схеми, що дало змогу отримувати матриці інциденцій значно меншого розміру без регулярно розташованих елементів а в результаті спростило обчислення й скоротило час на проектування системи.

Проведено аналіз та оптимізацію відомого матричного пристрою множення з горизонтальним розповсюдженням переносу, на основі характеристик складності його H-моделі, що дало змогу отримати пристрої множення з оптимізованими характеристиками складності.

У розділі також проведено аналіз характеристик складності відомого матричного пристрою множення з діагональним розповсюдженням переносу. Значення часової та апаратної характеристик складності було покращено, для цього вихідну матрицю було оптимізовано. Для досягнення необхідної продуктивності згідно вимог, модифіковану H-модель пристрою множення з діагональним переносом було оптимізовано з точки зору конвеєризації.

У процесі параметричної оптимізації матричного пристрою множення з діагональним розповсюдженням переносу було отримано двосходинковий конвеєрний пристрій множення, який задовольняє поставленим у завданні вимогам, а саме затримка сходинки конвеєра пристрою множення є меншою, ніж затримка на одному багаторозрядному суматорі, має оптимальні значення апаратної та структурної характеристик складності та кількість сходинок конвеєра не залежить від розрядності вхідних даних.

Також у розділі отримані характеристики складності SH-моделей двох типів схем керування спецпроцесорів й показано, що для підвищення ефективності обчислення спеціальних функцій необхідно реалізувати функціональні блоки пристрою таким чином, щоб програмна складність була мінімальною.

У третьому розділі автором проведено оптимізацію SH-моделей функцій згортки та швидкого перетворення Фур'є для спецпроцесора цифрової обробки сигналів.

Отримано характеристики складності SH-моделі згортки з неоднорідною структурою й показано, що таку структуру найкраще застосовувати для реалізації у спецпроцесорах, оскільки вона має високу швидкодію та низьку апаратну складність й найкраще підходить для суміщення з іншими алгоритмами. Розроблено конвеєрну оптимізовану H-модель пристрою згортки, із затримкою сходинки конвеєра не більшою, ніж затримка на одному багаторозрядному суматорі.

Також в роботі розроблено схему конвеєрного пристрою, що реалізує алгоритм швидкого перетворення Фур'є, оптимізованого за характеристиками складності H-моделі алгоритму. Проведено оптимізацію характеристики складності H-моделі метелика швидкого перетворення Фур'є на чотирьох помножувачах й отримано H-модель пристрою розділеного на чотири паралельні гілки, яка має у порівнянні з відомим пристроєм меншу структурну складність однієї гілки на 50%, що значно спрощує процес проектування, а також найкраще підходить для суміщення на одній структурі з алгоритмом згортки.

Також у розділі отримано RH-модель із суміщенням алгоритмів згортки та метелика ШПФ, яка має оптимальні значення часової та структурної характеристик складності. Такий пристрій виконуватиме обчислення алгоритму згортки, або ШПФ, в залежності від того, як будуть переключені мультиплексори.

Четвертий розділ присвячений реалізації VHDL-моделей оптимізованих за характеристиками складності пристроїв спецпроцесорів обробки сигналів для FPGA. Проведено порівняння SH- та VHDL-моделей комп'ютерних схем й показано, що VHDL-модель є доповненням до SH-моделі при оптимізації характеристик складності спеціальних функцій спецпроцесорів на схемотехнічному рівні. У дисертації реалізація пристроїв проводилась з

використанням програмного забезпечення Quartus II фірми розробника програмованих логічних інтегральних схем Altera.

Ключові слова: SH-модель, H-модель, характеристики складності, спецпроцесор опрацювання сигналів, структурний синтез, параметрична оптимізація, FPGA, VHDL-модель.

Список публікацій здобувача:

Наукові праці, в яких опубліковані основні наукові результати:

1. Ткачук Т.І. Характеристики складності пристроїв множення / Черкаський М.В., Ткачук Т.І. // Науково-технічний журнал «Радіоелектронні і комп'ютерні системи» Національного аерокосмічного університету ім. Жуковського, «Харківський авіаційний інститут». – 2012. – №5 (57). – С. 142-147.

2. Ткачук Т. Ефективний пристрій згортки / М. Черкаський, Т. Ткачук // Вісник «Комп'ютерні науки та інформаційні технології» Національного університету «Львівська політехніка». – 2012. – № 732. – С. 66-71.

3. Ткачук Т. Аналіз VHDL-Моделей оптимізованих матричних пристроїв множення // Науково-технічний журнал «Радіоелектронні і комп'ютерні системи» Національного аерокосмічного університету ім. Жуковського, «Харківський авіаційний інститут». – 2016. – № 6 (80). – С. 136-140.

4. Ткачук Т.І. Розробка управління автономного гоночного робота на базі PSoC фірми CYPRESS / Т.І. Ткачук, М.Ю. Шалений, Д.Р. Наумов // Вісник наукових досліджень. – 2010. – № 13. – С. 169-173.

5. Ткачук Т. Оптимізований метод вимірювання позитронних анігіляційних спектрів у наноматеріалах з розвиненою поруватістю для сенсорних застосувань / Клим Г., Костів Ю., Чалий Д., Івануса А., Ткачук Т. // Міжвузівський науково-технічний збірник "Вимірювальна техніка та метрологія" – 2016. – № 77. – С. 87-93.

6. Tkachuk T.I. Methodology and algorithm of multicomponent analysis of positron annihilation spectra for nanostructured functional materials / H.I. Klym, A.I. Ivanusa, Yu.M. Kostiv, D.O. Chalyy, T.I. Tkachuk, R.B. Dunets, I.I.

Vasylchyshyn // Journal of nano- and electronic physics. – 2017. – Vol. 9. – No 3. – P. 03037-1 – 03037-6. (Індексується у Scopus).

7. Ткачук Т.И. Параметрическая оптимизация устройства БПФ / Ткачук Т.И., Черкасский Н.В. // Вестник «Физика, Математика, Информатика» Брестского государственного технического университета БрГТУ. – 2013. – №5(83). – С. 22-25.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. Черкаський М.В. Порівняння двох швидкодіючих структур / Черкаський М.В., Ткачук Т.І. // Збірник матеріалів IV міжвузівської науково-технічної конференції науково-педагогічних працівників “Проблеми та перспективи розвитку економіки і підприємництва та комп’ютерних технологій в Україні”. –2009. – С. 230-231.

2. Черкаський М.В. Оцінка складності RH-моделей згортки і ШПФ / Черкаський М.В., Ткачук Т.І. // Матеріали 4-ї міжнародної науково-технічної конференції ACSN-2009. – 2009. – С. 255-258.

3. M. Cherkaskyy Structural synthesis of H-model of FFT / Mykola Cherkaskyy, Taras Tkachuk // Proceedings of the 6-th International Conference “Advanced Computer Systems and Networks: Design and Application” ACSN-2013. – 2013. – P. 122-124.

4. Tkachuk T. Two Methods to Determining the Time Complexity of Algorithm // Proceedings of the XIII-th. International Conference “Modern problems of radio engineering, telecommunications, and computer science” TCSET’2016. – 2016 – P. 452-454. (Індексується у Scopus).

5. Klym H. Investigation of changes in positronium trapping in pores under the water influence in nanostructured MgO-Al2O3 ceramics / Klym H., Kostiv Y., Ivanusa A., Tkachuk T. // Book of abstracts of the 15th Young Researchers’ Conference “Materials Science and Engineering” (15YRC), (Belgrade, Serbia, 7-9 December, 2016). – 2017. – P. 38–39.

SUMMARY

Tkachuk T.I. Complexity characteristics of specialized functions SH-models and it application for special processors optimization. – Qualification scientific work on the manuscripts rights.

A thesis submitted in fulfillment of the Candidate of Engineering Science (PhD) degree in technical sciences on specialty 05.13.05 – «Computer systems and components». – Institute of Computer Technologies, Automation and Metrology of Lviv Polytechnic National University of Ministry for Education and Science of Ukraine, Lviv, 2018.

In the first section the features of modern models of specialized computer systems realization, which implement special functions of processing signals, taking into account optimization of the values of complexity characteristics, are investigated.

The analysis of the main models of formal algorithmic systems has been carried out and it has been shown that such models do not take into account hardware that prevents them from using them during designing, optimizing specialized computer systems taking into account all the characteristics of complexity.

The models of hardware of special processors are analyzed and it is shown that such a model allows to design computer facilities taking into account only two characteristics - productivity and volume of equipment, and other characteristics of complexity are not taken into account.

Models of hardware and software of specialized computer systems M. Cherkasskyy are analyzed - SH-model, H-model, RH-model. The H-model of the algorithm allows solving tasks of only a certain class, which differ only in sets of incoming data. Models of this type include, for example, models of combinational devices - adder, matrix multiplication devices, decoders and others. It is shown that such models allow to optimize specialized computer systems that implement special functions, taking into account the hardware, time, software and structural characteristics of complexity.

The second section concerns the optimization of SH-models of the multiplication operation, as the basis of special functions for processing signals.

A well-known method for calculating the structural complexity of SH-models has been improved by detecting and integrating into a group of similar-type elements of the scheme, which made it possible to receive incident matrices of a much smaller size without regular elements and, as a result, simplified the calculation and reduced the time for designing the system.

The analysis and optimization of a known matrix multiplication device with horizontal transfer propagation, based on the characteristics of the complexity of its H-model, has been made, which made it possible to obtain multiplication devices with optimized complexity characteristics.

The section also analyzes the characteristics of the complexity of a well-known matrix multiplication device with a diagonal spread of the transfer. The value of the time and hardware characteristics of the complexity has been improved, for this the original matrix has been optimized. In order to achieve the required performance according to the requirements, the modified H model of the diagonal multiplication device has been optimized from the point of view of conveyance.

In the process of parametric optimization of the matrix multiplication device with the diagonal propagation of the transfer, a two-stage conveyor multiplication device was obtained that satisfies the requirements set in the task, namely, the delay of the stage of the conveyor of the multiplication device is less than the delay on one multi-bit adder, has the optimal values of the hardware and structural characteristics of the complexity and the number of steps in the conveyor does not depend on the amount of input data.

Also, in the section the characteristics of the complexity of the SH-models of two types of control systems of special processors are obtained and it is shown that in order to increase the efficiency of the calculation of special functions it is necessary to implement the functional blocks of the device in such a way that the software complexity was minimal.

In the third section author has optimized SH-models of convolution and fast Fourier transform functions for a special processor for digital signal processing.

The characteristics of the complexity of the SH-model of the convolution with the non-uniform structure are obtained and it is shown that this structure is best used for implementation in special processors, because it has high performance and low hardware complexity and is best suited for combining with other algorithms. The conveyor optimized H-model of the convolution device was developed, with the delay of the conveyor stage not greater than the delay on one multi-bit adder.

Also, the scheme of a conveyor device that implements a fast Fourier transform algorithm optimized for the characteristics of the complexity of the H-model of the algorithm is developed in the work. The optimization of the complexity of the H-model of the butterfly of the fast Fourier transform on four multipliers was made and the H-model of the device was divided into four parallel branches, which has, in comparison with the known device, a smaller structural complexity of one branch by 50%, which greatly simplifies the design process, and also best suited for combining on one structure with a convolution algorithm.

Also in the section the RH model is obtained combining the algorithms of convolution and the FFT butterfly, which has optimal values of time and structural characteristics of complexity. Such a device will compute the convolution algorithm, or FFT, depending on how the multiplexers will be switched.

The fourth section is devoted to the implementation of VHDL models optimized for the characteristics of the complexity of devices special processing signal processing for FPGA. Comparison of SH- and VHDL-models of computer circuits was carried out and it was shown that the VHDL model is an addition to the SH-model in optimizing the characteristics of the complexity of special functions of special processors at the circuit design level. In the dissertation implementation of devices was carried out using the Quartus II software firm of the developer of programmable logic integrated circuits Altera.

Keywords: SH-model, H-model, complexity characteristics, digital signal processor, structural synthesis, parametric optimization, FPGA, VHDL-model.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

H – hardware;

RH – reconfigured hardware;

SH– software hardware;

КС – комп’ютерні системи;

МТ – машина Тюрінга;

ПЗ – програмне забезпечення;

ПЛІС – програмовані логічні інтегральні схеми;

ПМ – пристрій множення;

СКС – спеціалізовані комп’ютерні системи;

ФАС – формальні алгоритмічні системи;

ЦОС – цифрова обробка сигналів;

ШПФ – швидке перетворення Фур’є;

ВСТУП

Актуальність теми. На сучасному етапі розвитку комп'ютерних наук і технологій, спеціалізовані комп'ютерні системи займають одне з провідних місць, оскільки у них система команд, особливості архітектури, набір структурних елементів та конструкторсько-технологічне виконання, у порівнянні із універсальними комп'ютерними системами, дають змогу суттєво підвищити ефективність розв'язання вузького кола спеціальних задач, потреба в яких є практично у всіх сферах діяльності людини від медицини, транспорту, космічних досліджень і до побуту, наприклад, розумний будинок [1, 2, 3, 4, 5, 6, 7, 8]. Попри широке розмаїття цих задач, більшість з них базуються на операціях «метелика» швидкого перетворення Фур'є, згортки, переміщення чи обертання на певний кут зображення, пошуку об'єктів у базах даних тощо, які в інформаційних технологіях прийнято об'єднувати під назвою спеціальні функції.

Якщо розвиток спеціалізованих комп'ютерних систем, чималий вклад у який зробили відомі українські та зарубіжні вчені В.С. Глухов, Р.Б. Дунець, А.О. Мельник, Я.М. Николайчук, В.М. Опанасенко, В.П. Тарасенко, В.С. Харченко, М.В. Черкаський, Mounir Arioua, Said Belkouch, Mohamed Agdad, Graham Jullien, Surin Kittitornkun, Pieter Hooijmans, Yu Hen Hu, Bill Krenik, K.L. Нео та інші, до недавнього часу проводився у напрямку апаратної реалізації спеціальних функцій, то з підвищенням тактової частоти роботи мікроелектронних елементів стало можливим будувати спеціалізовані комп'ютерні системи у поєднанні універсальних процесорів із спецпроцесорами. У цьому випадку поставлені задачі розв'язуються у комбінації програмного та апаратного забезпечення [11]. Слід зазначити, що рішення про те, яку частину задачі реалізувати апаратно, а яку програмно, як правило, приймається інтуїтивно з врахуванням досвіду розробника та результатів моделювання чи експериментів. Тут вся складність полягає в тому, що моделі апаратних засобів, наприклад архітектура комп'ютера, та програмних засобів, наприклад алгоритми, є різними за своєю природою [1, 9].

Також при проектуванні реконфігурованих спеціалізованих комп’ютерних систем, наприклад, із суміщенням на одній структурі обчислення декількох спеціальних функцій, ускладнюється структура такої системи, що у результаті збільшує затрати часу на саме проектування, а також у подальшому ускладнює розуміння роботи із системою для кінцевого користувача [10].

Спроби об’єднати абстрактну теорію алгоритмів та архітектуру комп’ютерів були зроблені професором Миколою Черкаським. Ним була створена SH-модель алгоритму, яка зберігає абстрактний характер алгоритмічних перетворень, які відбуваються в середовищі апаратних та апаратно-програмних засобів, формує необхідне до відповідного серидовища поняття «алгоритм», яке не суперечить його інтуїтивному тлумаченню, а також уточнює та розширює список його властивостей (дискретність, елементарність, детермінованість, масовість, та ієрархічність) та характеристик складності (часова складність, апаратна складність, програмна складність, структурна складність, ємнісна складність), які можна ефективно застосовувати для оптимізації спецпроцесорів.

Проте, дослідженню властивостей характеристик складності SH-моделей спеціальних функцій і подальшої оптимізації спеціалізованих комп’ютерних систем на їх основі було приділено недостатньо уваги. А тому актуальною є наукова задача створення ефективних структур спеціалізованих комп’ютерних систем, що реалізують спеціальні функції опрацювання сигналів, на основі побудови відповідних SH-моделей та оптимізації значень їх характеристик складності.

Зв'язок роботи з науковими програмами, планами, темами.

Тема дисертації відповідає науковому напряму кафедри спеціалізованих комп’ютерних систем Національного університету «Львівська політехніка» – «Теорія та практика побудови апаратного та програмного забезпечення спеціалізованих комп'ютерних пристроїв, систем та мереж на новітній елементній базі». Дисертація виконана в межах науково-дослідної роботи «Вдосконалення теорії проектування NoC з матричною топологією» (№

держреєстрації 0112U006717, 2013-2014рр., виконавець), а також дисертація виконана в межах науково-дослідної роботи ДБ/Наносенсор «Наноструктуровані скло-керамічні середовища для високонадійних оптоелектронних та сенсорних застосувань» (№ держреєстрації 0116U004411, 2016-2017рр., виконавець). Дисертація виконана в межах науково-дослідної роботи Державного фонду фундаментальних досліджень (№ Ф74/208-2017 від 06.11.2017) у рамках гранту Президента України “Модифіковані функціональні середовища на основі наноструктурованих стекол та кераміки для широких приладних застосувань” (№ держреєстрації 0117U007181, 2017р., виконавець).

Мета і задачі дослідження. Метою дисертаційної роботи є оптимізація спеціалізованих комп’ютерних систем (спецпроцесорів), що реалізують спеціальні функції опрацювання сигналів і мають оптимальне співвідношення значень характеристик складності – структурної, часової, апаратної, програмної.

Для досягнення поставленої мети у роботі потрібно вирішити такі основні **задачі**:

- провести оптимізацію характеристик складності арифметичних пристроїв та пристроїв спеціальних функцій спецпроцесора опрацювання сигналів;
- вдосконалити метод обчислення структурної характеристики складності;
- розробити структуру конвеєрного пристрою множення із затримкою сходинки конвеєра не більшою, ніж затримка на одному багаторозрядному суматорі, та відсутністю залежності від розмірності вхідних даних;
- розробити SH-моделі пристроїв згортки та метелика швидкого перетворення Фур’є, оптимізовані за характеристиками складності;
- розробити методи проведення оптимізації пристроїв спецпроцесора на основі аналізу характеристик складності SH-моделі алгоритму;

- розробити VHDL-моделі оптимізованих за характеристиками складності арифметичних пристроїв та пристроїв спеціальних функцій спецпроцесора опрацювання сигналів.

Об’єктом дослідження є спеціалізовані комп’ютерні системи.

Предметом дослідження є властивості і характеристики складності апаратно-програмних (SH) та апаратних (H) моделей спеціалізованих комп’ютерних засобів опрацювання сигналів.

Методи дослідження ґрунтуються на теорії проектування спеціалізованих комп’ютерних систем, характеристиках складності програмно-апаратної (SH) та апаратної (H) моделях алгоритмів М. Черкаського, алгоритмах Колмогорова, методах структурного синтезу та параметричної оптимізації спеціалізованих комп’ютерних систем.

Наукова новизна одержаних результатів. Завдяки проведеним дослідженням розв’язано науково-прикладну задачу створення ефективних структур спеціалізованих комп’ютерних систем, що реалізують спеціальні функції опрацювання сигналів, на основі побудови відповідних SH-моделей та оптимізації значень їх характеристик складності. За результатами дисертації:

- вперше оптимізовано SH-модель конвеєрного пристрою швидкого перетворення Фур’є, шляхом розділення пристрою на чотири паралельні гілки, що дало змогу мінімізувати значення структурної та часової характеристик складності;
- удосконалено структуру матричного пристрою множення з діагональним розповсюдженням переносу шляхом оптимізації його характеристик складності, що дало змогу отримати двоступеневий конвеєрний пристрій множення із затримкою сходинки конвеєра не більшою, ніж затримка на одному багато розрядному суматорі;
- набув подальшого розвитку метод обчислення структурної складності спеціалізованих комп’ютерних систем шляхом об’єднання однорідних частин схеми у блоки таким чином, щоб не виникало однорідності

матриці інциденцій, що дало змогу скоротити час проектування таких систем;

- удосконалено SH-модель функції згортки, в якій оптимізовано значення характеристик складності, що дало змогу отримати структуру реконфігурованого спецпроцесора, у якому апаратні засоби суміщають реалізацію функції згортки та швидкого перетворення Фур'є, при мінімальних значеннях часової й структурної характеристик складності та при оптимальних значеннях апаратної й програмної характеристик складності.

Практичне значення одержаних результатів полягає у тому, що вони можуть бути використані для проведення оптимізації пристроїв реалізації спеціальних функцій спецпроцесорів цифрової обробки сигналів, які мають покращені значеннями характеристик складності порівняно з існуючими аналогами. Розроблені пристрої доведені до інженерного рівня.

Результати проведених у роботі досліджень використовуються в навчальному процесі Національного університету «Львівська політехніка» на кафедрі «Спеціалізовані Комп'ютерні Системи» у лекційних курсах «Архітектура спеціалізованих комп'ютерних систем», «Технології проектування комп'ютерних систем» та «Дослідження і проектування контролерів периферійних пристроїв», при проведенні лабораторних, практичних і науково- дослідних робіт студентів.

Результати досліджень дисертації використано при розробці контрольно-перевірочної апаратури та нестандартного обладнання по темі «Січ-2-1», а також при аналізі варіантів реалізації спеціалізованих комп'ютерних пристроїв у дослідно-конструкторській роботі «Оновлення-24 МР» на державному науково-дослідному підприємстві «КОНЕКС».

Розроблений у дисертації метод оптимізації характеристик складності SH-моделей алгоритмів застосований при формуванні системи моніторингу за станом вод в басейні ріки Західний Буг, Західним центром українського

відділення "Міжнародного центру наукової культури – Всесвітня лабораторія" (ЗЦ УВВЛ).

Результати оптимізації структурної складності системи, одержані методом розбиття її на однорідні блоки, створюють наукове підґрунтя для розвитку альтернативних методів оптимізації структур пристроїв реалізації спеціальних функцій спецпроцесорів, що дасть змогу зменшити затрати часу на проектування.

Особистий внесок здобувача. Основні положення та результати досліджень дисертації одержані автором самостійно. В роботі теоретично обґрунтовані та розроблені методи структурного синтезу та параметричної оптимізації характеристик складності арифметичних пристроїв та пристроїв спеціальних функцій спецпроцесорів цифрової обробки сигналів; за результатами аналізу матричних пристроїв множення з горизонтальним та діагональним розповсюдженням переносу створено оптимізовані за характеристиками складності матричні пристрої множення, зокрема двоступеневий конвеєрний матричний пристрій множення з діагональним розповсюдженням переносу, із затримкою на сходинці конвеєра не більшою, ніж затримка на одному багато розрядному суматорі; запропоновано оптимізовану за структурною та часовою характеристикою складності H-модель пристрою ШПФ, розділену на 4 паралельні гілки, яка найкраще підходить для суміщення на одній структурі із пристроєм згортки; здійснено реалізацію VHDL-моделей оптимізованих арифметичних пристроїв та пристроїв спеціальних функцій спецпроцесора цифрової обробки сигналів, та проведено моделювання їх роботи на FPGA. У роботах, написаних у співавторстві, дисертантові належать: [104] аналіз характеристик складності пристроїв множення, опис методів оптимізації характеристик складності матричних пристроїв множення, а також інтерпретація одержаних результатів; [103] результати експериментальних досліджень можливості реалізації конвеєрного пристрою згортки з оптимізованими значеннями характеристик складності; [101] оптимізація алгоритмів управління системою робота та

інтерпретація одержаних результатів; [46] результати експериментальних вимірювань, дослідження шляхів оптимізації алгоритмів вимірювання позитронних анігіляційних спектрів у наноматеріалах; [86] отримання результатів експериментальних досліджень; [97] аналіз позитронних анігіляційних спектрів та отримання результатів експериментальних досліджень; [102] дослідження методів оптимізації пристроїв спеціальних функцій на прикладі алгоритму ШПФ, реалізація VHDL-моделі оптимізованого пристрою виконання алгоритму ШПФ та моделювання його роботи на FPGA, опрацювання отриманих експериментальних даних; [100] аналіз методів суміщення алгоритмів згортки та ШПФ на одній структурі; [25] аналіз характеристик складності RH-моделей згортки та ШПФ; [105] експериментальні дослідження варіантів структурного синтезу апаратної моделі конвеєрного пристрою ШПФ з оптимізованими значеннями характеристик складності.

Апробація результатів дисертації. Основні результати досліджень представлялися та обговорювалися на вітчизняних і міжнародних наукових конференціях, семінарах та школах, зокрема, особисто здобувачем у формі усних та стендових доповідей: IV-а міжвузівська науково-технічна конференція науково-педагогічних працівників “Проблеми та перспективи розвитку економіки і підприємництва та комп’ютерних технологій в Україні” 2009 (Львів, 2009); 4-а міжнародна науково-технічна конференція ACSN-2009, 2009 (Львів, 2009); 6-th International Conference Dependable Systems, Services and Technologies (DESSERT’12), 2012 (Sevastopol, 2012); 6-th International Conference ACSN-2013 “Advanced Computer Systems and Networks: Design and Applications”, 2013 (Lviv, 2013); XIII-th. International Conference “Modern problems of radio engineering, telecommunications, and computer science” TCSET’2016, (Slavsko, 2016); 15th Young Researchers’ Conference “Materials Science and Engineering” (15YRC), (Belgrade, Serbia, 2016).

Публікації. Основні результати дисертаційних досліджень висвітлено в 12 наукових працях, зокрема опубліковано 7 статтей у наукових фахових

виданнях (одна з них – одноосібна) та здійснено 5 публікації у збірниках матеріалів закордонних та всеукраїнських наукових конференцій міжнародного рівня. 2 наукові праці індексовано в наукометричній базі Scopus.

Структура і обсяг дисертації. Робота складається із вступу, чотирьох розділів, загальних висновків, списку використаних джерел із 119 назв та додатків. Загальний обсяг дисертації становить 165 сторінок, із них 139 сторінок основного тексту, 71 рисунок та 7 таблиць, а також список літератури на 13 сторінках.

РОЗДІЛ 1
АНАЛІЗ МОДЕЛЕЙ РЕАЛІЗАЦІЇ СПЕЦІАЛЬНИХ ФУНКЦІЙ ОПРАЦЮВАННЯ СИГНАЛІВ

Реалізація спеціальних функції займає одне з провідних місць у розробці спеціалізованих комп'ютерних систем опрацювання сигналів, оскільки у них система команд, особливості архітектури, набір структурних елементів та конструкторсько-технологічне виконання, у порівнянні із універсальними комп'ютерними системами, дає змогу суттєво підвищити ефективність розв'язання спеціальних задач. Попри широке їх різноманіття, більшість з них базуються на операціях «метелика» швидкого перетворення Фур'є, згортки, переміщення чи обертання на певний кут зображення, пошуку об'єктів у базах даних тощо.

При розробці спеціальних функцій вся складність полягає у виборі методу синтезу структури функції, яка розробляється. Розрізняють такі основні методи синтезу спеціальних функцій: апартно-програмна реалізація, яка базується на основі аналізу моделей алгоритму; апаратна реалізація, на базі аналізу моделей апаратних засобів спеціалізованих комп'ютерних систем; реалізація на основі аналізу моделей апаратно-програмних засобів проектування спецпроцесорів. Також при проектуванні реконфігурованих спеціалізованих комп'ютерних систем, наприклад із суміщенням на одній структурі обчислення декількох спеціальних функцій, ускладнюється структура такої системи, що у результаті збільшує затрати часу на саме проектування, а також в подальшому ускладнює розуміння роботи із системою для кінцевого користувача. Далі розглянемо використання кожної з моделей проектування спеціалізованих комп'ютерних систем детально.

1.1. Аналіз моделей алгоритмів

Термін "алгоритм" хоч і належить до фундаментальних математичних понять, проте не можливо дати точне його визначення, виходячи з інших фундаментальних понять. Сучасне інтуїтивне його тлумачення таке:

"Алгоритм - це точний припис, який задає обчислювальний процес, який починається з деякої системи початкових даних, і є спрямований на отримання результату, який повністю відповідає початковим даним" [13].

Таке формулювання зв’язує головні складові поняття алгоритму. У неявній формі тут присутні всі параметри алгоритму, але, разом з цим, таке формулювання не є математичним об'єктом. Воно не може бути основою для дослідження математичних і комп'ютерних проблем. Більшою мірою, воно пояснює зміст абстрактного алгоритму, тому що тут відсутні посилання на наявність технічних засобів виконання обчислювального процесу. Проте таке визначення алгоритму з’явилося не відразу. У ході еволюції терміну «алгоритм» були визначені умови, загальні для будь-яких математичних побудов [14, 15]:

1) кількість операцій, яка необхідна для розв’язання конкретної проблеми, має бути скінченною;

2) операції мають бути елементарними, щоб уникнути помилок на складному ланцюжку інтуїтивних переходів в процесі розв’язання задачі.

Термін “елементарність” математично не визначався і в даному контексті еквівалентний поняттю “простота і локальність”. А. Марков, крім того, надає цьому терміну зміст “загальна зрозумілість” [16]. Так як ці поняття не мали математичного змісту, це сприяло появі багатьох різних напрямків подальшого уточнення поняття “алгоритм” і побудови систем дослідження проблем антимоній і розв’язності з його допомогою [17].

Для подальшого уточнення змісту поняття алгоритму і прив'язки цього поняття до сучасних комп'ютерних апаратно-програмних засобів необхідні нові формальні засоби і моделі, а для дослідження проблем розв’язності – нові моделі алгоритму.

Побудова ефективних алгоритмів з використанням їхніх моделей є головним завданням теорії складності обчислень. Такі моделі поділяються на два класи, в залежності від того, дотримується алгоритм властивості елементарності кроку, чи ні. Формальні алгоритмічні системи цієї властивості

дотримуються, натомість у неформальних алгоритмах крок алгоритму інтегрований. Формальні алгоритмічні системи, у свою чергу, поділяються на алгоритми з використанням апаратно - програмного обчислювача (комп'ютерні алгоритми) і алгоритми з абстрактним обчислювачем (рис. 1.1).

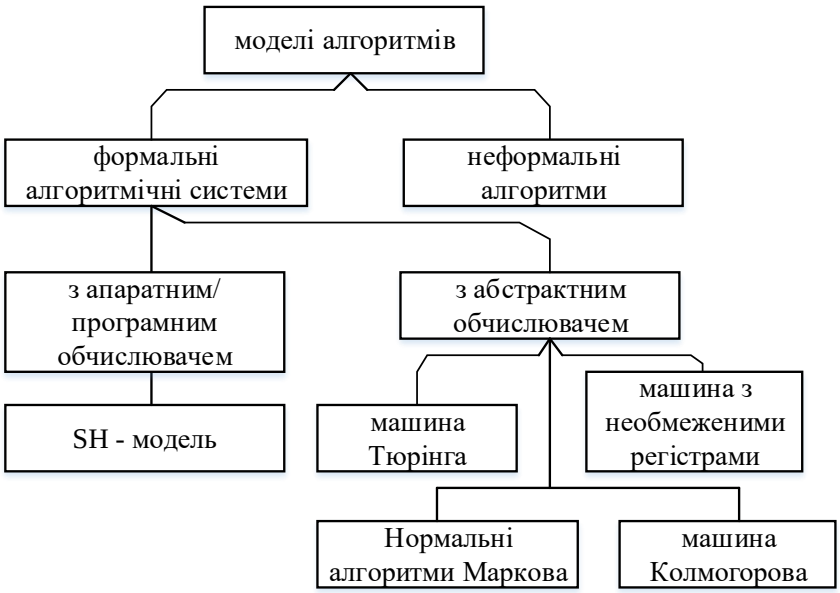


Рисунок 1.1. Класифікація моделей алгоритмів

Розглянемо моделі з абстрактним обчислювачем на прикладі машини Тюрінга і нормальних алгоритмів Маркова. Значення таких моделей велике, тому що вони найбільш точно відображають властивості і характеристики складності абстрактних алгоритмів.

До так званих абстрактних неформальних алгоритмів з інтегрованим кроком можна віднести: словесні алгоритми, дискретні математичні залежності, блок-схеми програм, програми на мові високого рівня. Моделі з інтегрованим кроком застосовуються в процесі розробки алгоритмів та підготовки їх для реалізації на комп'ютері (рис. 1.2).

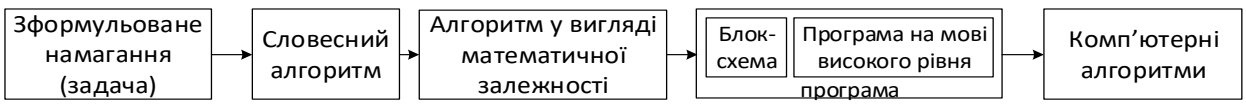


Рисунок 1.2. Реалізація комп’ютерного алгоритму

Розробка та дослідження алгоритмів, як і інших об'єктів розробки, проводиться у відповідності до схеми (рис. 1.3).

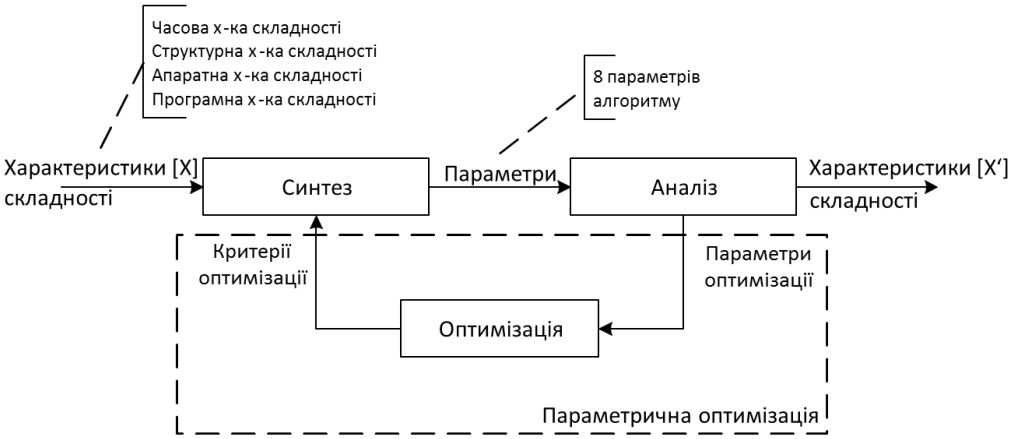


Рисунок 1.3. Розробка моделі алгоритму

Характеристики складності є вихідними даними для операції синтезу алгоритму, результатом синтезу є алгоритм з набором параметрів. Для визначення того, чи є об’єкт розробки самодостатнім з точки зору заданих характеристик, проводять його «аналіз». Якщо степінь розбіжності характеристик на вході блоку «синтез» і виході блоку «аналіз» не припустима, проводять оптимізацію об'єкта розробки, в нашому випадку алгоритму. Параметрами алгоритму є: правило початку, правило введення, правило виходу, правило безпосередньої переробки, правило закінчення, систему вхідних даних (потенційно нескінченну), систему проміжних результатів, систему кінцевих результатів (рис. 1.4) [18].

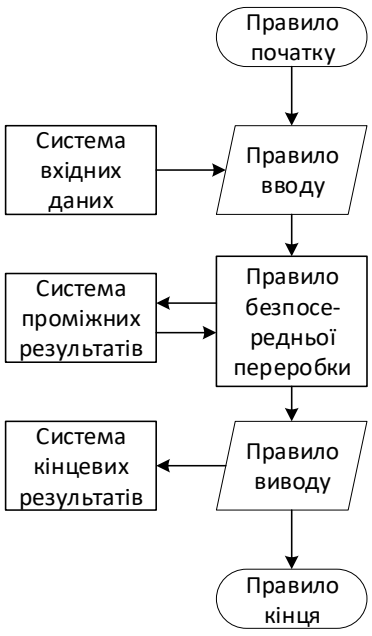


Рисунок 1.4. Структура алгоритму

У процесі розробки алгоритму зазвичай виконують параметричну оптимізацію, яка полягає в підборі необхідних параметрів. Параметри, при їх зміні, впливають на значення характеристик складності. Приклади такої оптимізації будуть наведені далі.

1.1.1. Машина Тюрінга

Серед відомих моделей формальних алгоритмічних систем (ФАС) найбільш популярною є машина Тюрінга. Вона уточнює інтуїтивне поняття алгоритму, володіє всіма властивостями алгоритму і має всі вісім параметрів. Команди машини Тюрінга моделюються простими операціями: читанням символу з рядка, знаходженням і читанням команди в програмі, реалізацією команд, записом нового або стиранням старого символу, пересуванням головки вправо, вліво або залишанням її на місці. Перевагою її над рекурсивними функціями є однотипність кроків для будь-яких алгоритмів. Фіксованість кроку машини Тюрінга дозволила порівнювати різні алгоритми, сформулювати початкові визначення теорії складності - дати поняття часової і ємнісної характеристик складності. Машина Тюрінга, з початку розроблення, була орієнтована на дослідження обчислювальних операцій, з точки зору доведення існування розв’язку для конкретної задачі. Складання програми для машини Тюрінга є простим і зрозумілим, тому в подальшому її почали застосовувати для аналізу задач і ефективних алгоритмів [19].

Машина Тюрінга - це ФАС з абстрактним обчислювачем. У складі абстрактного обчислювача не передбачено технічних засобів виконання операцій. Для ФАС з таким обчислювачем відсутня така технічна характеристика, як апаратна складність. Ця особливість не дозволяє повністю переносити результати досліджень математичних моделей на практику досліджень апаратно-програмних комп'ютерних засобів. Машина Тюрінга є корисною для побудови нових комп'ютерних моделей алгоритму.

Машина Тюрінга – це шістка [18]:

$$M: \langle A, Q, a_0, q_0, q_f, P \rangle$$

де A – кінцева множина символів зовнішнього алфавіту;

Q – кінцева множина символів внутрішнього алфавіту;

q_0 і q_f – початковий і кінцевий стани машини Тюрінга $[q_0, q_f \in Q]$;

a_0 – позначення пустої комірки стрічки $[a_0 \in A]$;

P – така програма, яка не може мати двох різних п'ятірок, в яких би збігалися два перших символи:

$$P: \{A\} \times \{Q\} \rightarrow \{A\} \times \{L, R, S\} \times \{Q\},$$

де L- задає дію зсуву головки вліво, R- вправо, S- залишити на місці.

Множини A,Q не пересікаються і не містять літер L,R,S;

Машина Тюрінга має одну і ту ж конфігурацію засобів реалізації алгоритму для розв’язання будь-якої задачі. В конфігурацію входять: нескінченна нерухома стрічка, що поділена на окремі комірки, в які можна розмістити не більше одного символу зовнішнього алфавіту; рухома головка, яка може стирати, записувати і зчитувати символи зовнішнього алфавіту в комірках стрічки; програма із кінцевою кількістю команд. Машина Тюрінга є абстрактною ідеалізованою моделлю алгоритму. На відміну від моделей software/hardware (SH), вона не враховує апаратні витрати, необхідні для реалізації алгоритму. Ця особливість абстрактних моделей не дозволяє у повній мірі використовувати досягнення теорії ФАС у проектуванні апаратно-програмних засобів, у деяких випадках цей недолік приводить до практично неприйнятних висновків.

Кроки, які виконуються машиною Тюрінга, дискретні, детерміновані і умовно елементарні, правило безпосередньої переробки має властивість «масовість». Фіксованість кроку машини Тюрінга дозволила порівнювати різні алгоритми за часовою і ємнісною складністю, сформулювати початкові визначення метричної теорії складності. Однак, таке розуміння кроку не вирішує проблеми точного математичного визначення поняття "елементарність".

При детальному аналізі прикладів роботи машини Тюрінга можна зробити наступні висновки:

• Кожен крок машини Тюрінга складається з максимум шести операцій, отже цей крок не є елементарним.

• Програма є частиною моделі алгоритму. Цю обставину підкреслено у зв'язку з поширеним судженням про те [19], що програма є алгоритмом, але це не зовсім вірно.

• Для вирішення однієї і тієї ж задачі може бути запропоновано деяка підмножина алгоритмів. Причому кожен алгоритм буде мати свою програму і свою кількість послідовних кроків (часову складність).

У формальному математичному визначенні машини Тюрінга дані про елементи її конструкції не задекларовані, тому її принципово неможливо побудувати. Уявний образ цієї машини (головка читання, стрічка, засоби управління) використовується лише для пояснення принципу її роботи. Крок машини Тюрінга теж є математичною абстракцією, його не можна ототожнити з якимось елементарним процесом, який відбувається в матеріальному середовищі. Він визначається зміною її станів без пояснень того, як це відбувається. Тому, термін «елементарність кроку» точно не визначається.

Характеристики складності машини Тюрінга.

Машина Тюрінга має дві характеристики складності, часову та ємнісну. Часова складність визначається кількістю кроків, необхідних для вирішення задачі. Ємнісна складність визначається кількістю використовуваних комірок стрічки. Іноді, для аналізу машини Тюрінга, застосовують складність програми, яку оцінюють кількістю команд програми. Зауважимо, що при аналізі моделей комп'ютерних алгоритмів, під терміном програмна складність буде розумітися інформаційна характеристика програми.

Основним завданням побудови ефективного алгоритму є зменшення його часової складності. Детальне дослідження машин Тюрінга, які розв’язують одну і ту ж задачу, показали, що мінімізації часової складності сприяє: мінімальній кількості зупинок і розворотів сліду, введення додаткових символів, вдалому вибору розташування на стрічці вихідних даних та вибору початкового і кінцевого положення головки. Перераховані способи мінімізації

часової складності, що застосовуються при побудові програми, є евристичними. Зазначимо, що ці способи частково застосовуються при розробці і комп'ютерних алгоритмів [18].

Одним з найбільш ефективних способів обробки даних на комп'ютерах є паралелізм. Початково Машина Тюрінга була послідовною машиною, тому всі, навіть найпростіші, операції вимагають великої часової складності. Для прискорення обчислювального процесу було запропоновано [20] багато стрічкову машину Тюрінга, яка зберігає всі переваги одно стрічкової, одночасно істотно зменшує часову складність і спрощує логіку обчислень.

Багато стрічкова машина Тюрінга.

Багато стрічкова машина Тюрінга (рис. 1.5.) складається з деякого скінченного числа стрічок, розбитих на комірки. В кожній з комірок може знаходитися один із символів зовнішнього алфавіту $A=\{a_0,a_1,.,a_n\}$. Комірки з символом a_0 вважаються пустими. Пристрій керування машини в кожен момент часу знаходиться в одному із скінченної множини станів $Q=\{q_0,q_1,.,q_m\}$.

Передбачається, що в кожен момент часу машина "проглядає" одну комірку кожної стрічки. Конфігурація машини вважається заданою, якщо відомий стан пристрою керування $\boldsymbol{q}$, стани всіх комірок, усіх стрічок і відмічені комірки, що сприймаються машиною. Для $\boldsymbol{к}$-стрічкової машини конфігурація її в $\boldsymbol{i}$-й момент часу описується системою $\boldsymbol{к}$ слів наступного вигляду:

$$\begin{matrix} a_{i11}\ldots a_{i1e}q_i a_{i1e+1}\ldots a_{i1t} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ a_{ik1}\ldots a_{ike}q_i a_{ike+1}\ldots a_{ikv} \end{matrix} \qquad (1.1)$$

По одному слову для кожної стрічки, перший індекс відповідає моменту часу, другий - номеру стрічки, третій - номеру комірки на стрічці, дивлячись зліва направо.

Залежно від внутрішнього стану пристрою керування q_i змісту комірок стрічок, які проглядаються в даний момент a_{i1e+1} … a_{ike+1} машина переходить у наступний стан за допомогою таких дій:

1. Зміст кожної комірки, яка проглядається, замінюється новим (може співпадати зі старим);

2. Головка кожної стрічки пересувається на одну клітинку вправо або вліво, або залишається нерухомою;

3. Внутрішній стан пристрою керування q_i переводиться в новий стан q, (який може збігатися зі старим).

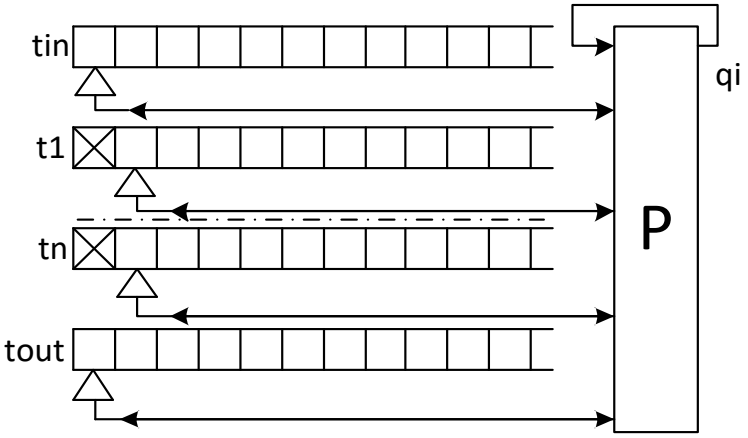


Рисунок 1.5. Графічне представлення багато стрічкової машини Тюрінга

Формальне задання багато стрічкової машини Тюрінга наступне [18]:

$$\langle A, I, Q, a_0, q_0, q_f, P \rangle,$$

де A - кінцева множина символів зовнішнього алфавіту;

Q - кінцева множина символів внутрішнього алфавіту;

I - вхідний алфавіт, $I \subset A$;

q_0, $q_f \in Q$ (початковий і кінцевий стани);

P – програма;

a_0 - порожня клітинка, $a_0 \in A$.

Як бачимо, на багато стрічковій машині Тюрінга можливе виконання декількох операцій на кожній із стрічок одночасно за однією командою. Використаний паралелізм суттєво збільшує продуктивність розв’язку задач.

Але це не узгоджується з вимогою дотримання елементарності кроку алгоритму [18].

Таким чином, недоліками машини Тюрінга з точки зору використання її для дослідження та проектування спеціалізованих комп'ютерних засобів є:

1. Неможливість оцінки апаратної складності, як характеристики моделі;

2. Відсутність математичної строгості у визначенні елементарності кроку алгоритму;

Процес розробки спеціалізованих комп’ютерних засобів, в тому числі для реалізації спеціальних функцій, потребує моделі, в яких апаратна складність та елементарність кроку алгоритму були би чітко визначені.

1.1.2. Алгоритмічна система Маркова

Алгоритмічна система Маркова [16] будується майже за тими ж принципами, що і система Тюрінга. Вона також уточнює поняття “алгоритм”, дає визначення кроку алгоритму через дві взаємозв’язані операції, що повторюються через кінцеву кількість циклів - розпізнавання і підстановки [18].

Нехай X - деякий кінцевий алфавіт, F(X) - напівгрупа слів у цьому алфавіті і [e ϵ F (X)], де e - порожнє слово. Якщо p, q ϵ F (X), то вирази p→q і p→.q називаються формулами підстановки в алфавіті X. При цьому вважається, що символи → не належать алфавіту X, а слова p і q можуть бути порожніми. Формула підстановки p→q називається простою, а формула підстановки p→.q заключною.

Нехай вираз [p→.q] означає будь-яку з формул підстановки p→q або p→.q. Кінцева послідовність R формул підстановки в алфавіті X:

$$\begin{cases} p_1 \to q_1 \\ p_2 \to q_2 \\ \ldots\ldots\ldots\ldots \\ p_l \to .q_l \end{cases} \qquad (1.2)$$

називається схемою або системою переписування. Будь-яка система переписування є функцією f: F(X)→F(X), значення якої обчислюються за такими правилами:

1. Якщо жодне з слів p_i (i = 1,2, ...,l) не є підсловом слова p, то p залишається без змін і є результатом переписування. Цей факт будемо записувати у вигляді R: p!

2. Якщо серед слів p_1, p_2, ..., p_l існують такі, що є підсловами слова p, то нехай m - таке найменше число, що $1 \leq m \leq l$ і p_m - підслово слова p. Слово p', отримане підстановкою слова q_m замість самого лівого входу (першого входу) слова p_m в слово p, будемо позначати R: p ├p'.

Робота за такими правилами може закінчитися з двох причин:

а) якщо формула підстановки $p_m \rightarrow [.]q_m$ є заключною;

б) якщо існує така послідовність $r_0, r_1, \ldots, r_k$ слів з F(X), що $p=r_0$, $q=r_k$ і R: r ├r_{i+l} для i=l, 2, ..., k-2 і або R: r_k!, або R: $r_k \rightarrow .r_k$.

Функція f: F(X)→F(X), визначена таким чином, називається нормальним алгоритмом Маркова в алфавіті X.

Робота алгоритму R може бути описана наступним чином. Нехай pєF(X). Знаходимо в R першу формулу підстановки $p_m \rightarrow [.]q_m$, таку, що p_m- підслово слова p. Виконуємо заміну першого входу слова p_m словом q_m, у слові p. Якщо p'- результат цієї підстановки, то коли $p_m \rightarrow [.]q_m$, заключна, то робота алгоритму закінчується, і результатом є слово p'. Якщо формула підстановки $p_m \rightarrow [.]q_m$ проста, то до слова p' застосовується той же пошук, що і до слова p, і т.д. Якщо, нарешті, отримано таке слово r_i, що R:r_j!, тобто жодне з слів p_i, i= 1,2, ..., l, не є підсловом слова r_i, то робота алгоритму закінчується і r_i буде його значенням.

Із сказаного випливає, що можлива ситуація, коли описаний процес ніколи не закінчиться. У цьому випадку будемо говорити, що алгоритм R не застосовується до слова p.

На відміну від машин Тюрінга, нормальні алгоритми Маркова не мають рухомих елементів реалізації алгоритму. Обчислення проводяться за

допомогою розпізнавачів та перетворювачів, технічні способи реалізації цих пристроїв не вказуються.

Характеристики складності нормальних алгоритмів Маркова.

Нормальні алгоритми Маркова також як і машина Тюрінга володіє трьома характеристиками складності: часовою складністю, ємнісною складністю і складністю програми. Кількісне значення часової і ємнісної складності визначається незалежно від довжини аналізованих слів і команд. Операції розпізнавання і перетворення проводяться методом перебору. У зв'язку з цим в літературних джерелах не розглядаються способи мінімізації часової складності нормальних алгоритмів Маркова. Точне значення задається для кожного алгоритму тільки для кількості команд програми.

Таким чином, нормальні алгоритми Маркова також як і машина Тюрінга не дають формально точного визначення властивості алгоритму "елементарність". Ця властивість, разом з іншими трьома (дискретність, детермінованість, масовість) є базовим для теорії алгоритмів. Наведений опис нормальних алгоритмів Маркова дозволяє зробити наступні висновки [18].

1. Кожен крок нормальних алгоритмів Маркова складається з двох операцій, які, строго кажучи, не є елементарними. Кожна операція розпізнавання і кожна операція перетворення можуть охоплювати різну кількість символів слова.

2. Програма є частиною моделі алгоритму і задається окремим списком послідовних команд. При цьому не вказується, де зберігаються ці команди.

1.1.3. Стан алгоритмічного процесу Колмогорова

Вивчаючи теорію абстрактних алгоритмів і базуючись на визначеннях алгоритму, які були описані до нього, А. М. Колмогоров дає більш загальне за формою визначення алгоритму, що задовольняє умовам обмеженої складності кожного кроку алгоритму.

Алгоритм Г задається за допомогою припису, який вказує спосіб переходу від “стану” S процесу обчислень до “наступного” стану S*, тобто за допомогою оператора $\Omega_г(S)=S^*$, який є оператором “безпосередньої

переробки". В класі $\mathfrak{S}(\Gamma) = \{S\}$ можливих станів виділяється клас $\mathfrak{A}(\Gamma)$ "вихідних" станів, який являє собою записи умов задач, і клас $\mathfrak{C}(\Gamma)$ "заключних" станів. При отриманні стану з класу $\mathfrak{C}(\Gamma)$ алгоритмічний процес закінчується, а з отриманого "заключного" стану витягується "розв'язок". Область $\mathfrak{D}(\Gamma) = \mathfrak{S}(\Gamma)$, на якій оператор Ω визначений, прийнято вважати такою що не пересікається з $\mathfrak{C}(\Gamma)$. Клас станів, який не входить ні у $\mathfrak{C}(\Gamma)$, ні у $\mathfrak{D}(\Gamma)$, позначено через $\mathfrak{N}(\Gamma)$ [21].

Алгоритмічний процес

$$S^0 = A \in \mathfrak{A}(\Gamma),$$
$$S^1 = \Omega_\Gamma(S^0),$$
$$S^2 = \Omega_\Gamma(S^1),$$
$$\ldots\ldots\ldots\ldots$$
$$S^{t+1} = \Omega_\Gamma(S^t),$$

може розвиватися одним з трьох способів:

1. При якомусь $t = \bar{t}$ процес переходить до заключного стану

$$S^{\bar{t}} = \mathfrak{C}(\Gamma),$$

після чого із цього заключного стану отримується розв'язок B.

2. При якомусь $t = \bar{t}$ процес закінчується безрезультатно:

$$S^t = \mathfrak{N}(\Gamma).$$

3. Процес продовжується безкінечно, не приводячи до результату.

Клас $\mathfrak{G}(\Gamma)$ тих $A \in \mathfrak{A}(\Gamma)$, які приводять до першого типу проходження процесів, називається «областю застосування» алгоритму [21].

Ще одним формулюванням Колмогорова є ідея локальності кожного окремого кроку («безпосереднє перетворення»). Локальна операція полягає у видаленні попередньо обмеженого «куска» об'єкта, який опрацьовується (стану), і заміні цього «куска» на інший, який визначається в залежності від першого.

Всі обчислювальні моделі з локальним перетворенням інформації легко можуть бути описані в колмогоровських термінах, назвемо їх «моделями

колмогорова». Моделі Поста і Тюрінга є прикладами таких моделей. З іншої сторони, моделі з нелокальними кроками, такі як нормальні алгорифми Маркова або машини з довільним доступом до пам’яті, потребують попереднього розщеплення кожного свого кроку на локальні кроки, і відповідно не є моделями колмогоровського типу [21].

А. Колмогоров [22] та А. Марков [16] давали таке тлумачення алгоритму, яке визначається переліком його властивостей:

а) алгоритм - це процес послідовної побудови величин, який проходить в дискретному часі так, що в початковий момент задається початкова скінченна система величин, а в кожний наступний момент система величин втримується за певним законом (програмою) із системи величин, які були в попередній момент часу (дискретність алгоритму);

б) система величин, які отримуються в якийсь (не початковий) момент часу, однозначно визначається системою величин, отриманих в попередні моменти часу (детермінованість алгоритму);

в) закон отримання наступної системи величин із попередньої повинен бути простим і локальним (елементарність кроків алгоритму);

г) якщо спосіб отримання наступної величини із якої-небудь заданої величини не дає результату, то повинно бути вказано, що потрібно вважати результатом алгоритму (спрямованість алгоритму);

д) початкова система величин може вибиратися із деякої потенційно нескінченної множини (масовість алгоритму).

1.1.4. Місце формальних алгоритмічних систем

ФАС абстрактних алгоритмів були розроблені ще до появи комп’ютерів. Моделі ФАС не використовували в своєму складі апаратних засобів. З точки зору сучасного стану комп'ютерних наук це є головним недоліком цих моделей, тому вони і не використовуються в практиці проектування апаратно-програмних засобів.

Разом з тим, моделі ФАС не втратили своє значення в теоретичному застосуванні. Ці моделі, особливо машина Тюрінга, дозволили сформулювати деякі положення теорії складності, яка зараз має наростаюче значення в практиці побудови ефективних алгоритмів. Теорія складності абстрактних алгоритмів дозволяє зв'язати модель задачі з ланцюжком моделей алгоритмів починаючи від словесного завдання, функціональної залежності, блок-схеми до програми на мові високого рівня. Для кожної з цих моделей теорія складності дозволяє сформулювати способи конструювання ефективних за часовою складністю алгоритмів. У даний час, будь-яка розробка апаратно-програмних і особливо програмних засобів, в тій чи іншій мірі, базується на теорії складності абстрактних алгоритмів.

Головне, на наш погляд, значення теорії абстрактних алгоритмів полягає в тому, що вона створила методологічну основу для подальших досліджень алгоритмів, в період її розвитку було зроблено декілька уточнень стосовно поняття алгоритм, створено низку математично точно описаних моделей, сформульовано вербальне тлумачення алгоритму, введено параметри, описано властивості, у тому числі “елементарність” та “спрямованість”, було введено часову та ємнісну характеристики складності.

1.2. Аналіз моделей апаратних засобів спецпроцесорів

Використання характеристик складності абстрактних алгоритмів є недостатнім для практичного проектування комп’ютерних систем, потрібен інший базис. Таким базисом у практиці дослідження і проектування комп’ютерних засобів на системному та функціональному етапах є архітектура комп’ютерів. В обчислювальну техніку термін “архітектура комп’ютерів” був введений фірмою IBM в кінці 50-х років двадцятого століття. Спочатку він означав можливість реалізовувати програми користувачів послідовністю моделей комп’ютерів деякого сімейства. Причому комп’ютери сімейства могли розрізняться за величинами продуктивності,

об’ємом обладнання та іншими характеристиками. Об’єднував моделі сімейства набір команд молодшої моделі [20].

Зараз під архітектурою комп'ютера розуміється як його загальна структура в цілому, так і організація його окремих елементів, необхідних для забезпечення його працездатності, і яка є доступною для користувача (програміста). Таким чином, даний термін охоплює і комп'ютер, і кристали, схеми та системні програми. Сюди також входять користувальницькі системи програмування, системи команд і система управління, організація пам'яті і адресації, інтерфейсу людина-комп'ютер і операцій введення-виведення і т.д. Реалізація конкретної архітектури на машинах даного сімейства може бути різною, але всі машини одного сімейства повинні бути здатні виконувати одну і ту ж програму. Вони можуть відрізнятися одна від одної на різних рівнях ієрархії апаратно-програмних засобів. Відмінності стосуються продуктивності та вартості [10].

Крім того, опис архітектури включає в себе роз'яснення принципу дії і можливостей будь-якого канального контролера. Частиною такого опису служить докладна структурна або принципова схема конкретної реальної, а не віртуальної машини.

Технічні деталі реалізації комп'ютерних засобів, невидимі для користувача, наприклад, елементна база, на якій будуються електронні пристрої комп'ютера, не входять в поняття архітектури. Це якраз та особливість комп'ютерних систем, яка за аналогією з будівництвом дає можливість використовувати для них термін "архітектура".

Спільність архітектури різних комп’ютерних систем забезпечує їх сумісність з точки зору користувача. У процесі розробки обчислювальної системи, її апаратно-програмних засобів, термін "архітектура" використовується для опису принципу функціонування, зміни структури і взаємозв'язків основних логічних вузлів.

Архітектура фон Неймана. Теоретичним базисом побудови сучасних комп'ютерів є принципи, сформульовані фон-Нейманом [10]. До них відносяться:

1. принцип програмного керування;
2. принцип однорідності пам’яті.

Вперше ці принципи були реалізовані в моделі обчислювача, запропонованої фон-Нейманом, який отримав назву машини з архітектурою фон-Неймана. Основною відмінністю цієї архітектури є послідовне виконання команд за правилом одна команда - одна дія. В даний час існує декілька стилів архітектури, які відрізняються використанням часового і просторового паралелізму, поділом пам'яті на банки і т.д. Але всі вони базуються на дотриманні принципів фон-Неймана [10].

В основу принципів фон Неймана покладено 5 положень:

- інформація кодується у двійковій формі і ділиться на слова;
- різнотипні слова інформації відрізняються лише способом використання, а не способом кодування;
- слова інформації розміщуються в комірках пам'яті та ідентифікуються номерами комірок, які називаються адресами слів;
- алгоритм роботи комп'ютера подається у формі послідовності керуючих слів, які визначають найменння операції і слова інформації, що беруть участь у цій операції;
- виконання обчислень згідно алгоритму зводиться до послідовності виконання команд у порядку, однозначно визначеному програмою.

Слід згадати відмінно розроблений теоретичний апарат мінімізації апаратної складності вентильних схем, який мав найважливіше практичне значення, коли схеми будувалися на дорогих і великих дискретних елементах. У зв'язку з революційними змінами в області технології мікроелектроніки ситуація змінилася, з'явилися інші пріоритети.

Характеристики складності архітектури комп’ютерів.

Основні характеристики комп'ютерів визначаються вимогами ринку, ці вимоги охоплюють всі рівні розробки, а саме: системний, функціонально-логічний, схемо-технічний, конструкторський. До архітектурного проектування відносяться системний і функціонально-логічний етапи проектування. Тут використовуються, головним чином, дві характеристики - продуктивність і обсяг обладнання, включаючи пам'ять. Продуктивність безпосередньо пов'язана з швидкодією, або часовою складністю. Обсяг обладнання еквівалентний апаратній складності. Наявність такої характеристики як апаратна складність принципово відрізняє архітектурну побудову від абстрактних алгоритмів. Ще одна відмінність - принцип ієрархічності - найважливіша властивість архітектурних побудов.

Засобами відображення і моделювання процесів є: структурні, функціональні, та логічні схеми; часові діаграми; графи станів; програми. Проте, моделей апаратних засобів для побудови ефективних спеціалізованих комп’ютерних систем опрацювання сигналів недостатньо, оскільки не враховується програмна складність. А саме вона досить часто впливає на кінцеву продуктивність комп’ютерних систем. Це пов’язано з тим, що для одного й того самого алгоритму може бути низка програм, кожна з яких забезпечує отримання тих самих результатів, але з різним часом роботи.

1.3. Аналіз моделей апаратно-програмних засобів реалізації спецпроцесорів

Розглянуті абстрактні моделі алгоритму містять ті чи інші апаратні засоби або припускають їх наявність. Наприклад, в машині Тюрінга припускається наявність рухомих головок, засобів читання і запису, дешифраторів при виборі тих чи інших символів зовнішнього та внутрішнього алфавітів. Але під час побудови ефективних алгоритмів вони не враховуються.

Нормальні алгоритми Маркова також містять підмножину елементарних операторів, розпізнавачів і перетворювачів, але в систему характеристик апаратні затрати не входять.

А.М. Колмогоров у своїх працях говорить про такі поняття як «безпосереднє перетворення», «алгоритмічний процес» та «стан алгоритмічного процесу» [22]. Але він не дає роз'яснення введених ним понять, він не розшифровує значення «алгоритмічного процесу». У Колмогорова існує система станів алгоритмічного процесу, але він не дає опису поняття «стан».

Також у працях Колмогорова згадується поняття «елемент» та «зв'язки між елементами», що значно наблизило його до об'єднання абстрактної теорії алгоритмів з архітектурою комп'ютерів. Опису терміну «елемент» Колмогоров не приводить, елемент ніяк не прив'язується до апаратної складової, що існує у комп'ютерах, а апаратура є невід'ємною частиною архітектури комп'ютерів.

Таким чином, між теорією абстрактних алгоритмів та теорією проектування спеціалізованих комп'ютерних систем є істотні відмінності, які не дозволяють одночасно використовувати переваги цих двох напрямків для проектування спеціальних функцій при розробці спецпроцесорів обробки сигналів. Для цього необхідні нові теоретичні побудови, які б пов'язали теорію абстрактних алгоритмів і теорію проектування спеціалізованих комп'ютерних систем (рис. 1.6).

Такою теорією може бути створена М.В. Черкаським теорія комп'ютерних алгоритмів, який описав модель алгоритму з прив'язкою до апаратної складової комп'ютерів [17]. Апаратно-програмна (software-hardware, SH) модель алгоритму у своїй основі використовує «елементарний перетворювач», як базу характеристик складності алгоритму. Також вона володіє розширеним набором властивостей, та характеристик складності комп'ютерного алгоритму.

Рисунок 1.6. Теорія комп'ютерних алгоритмів як зв'язна ланка між архітектурою комп'ютерів та теорією абстрактних алгоритмів [23]

Теоретична концепція побудови апаратно-програмної моделі алгоритму базується на тому, що:

1. основою побудови моделі повинна бути теорія алгоритмів;
2. структура моделі повинна включати об'єкти перетворення даних;
3. модель повинна враховувати інформаційні характеристики.

Апаратні засоби містять один або декілька елементарних перетворювачів. Елементарний *i*-тий перетворювач x_i є одиницею апаратної складності.

$$\forall i, x_i = 1 \tag{1.3}$$

1.3.1. SH-модель

SH-модель є фіксованою для деякого класу задач конфігурацією апаратно-програмних засобів перетворення, передавання і зберігання даних, що задає обчислювальний процес, який починається з деякої системи початкових даних і скерований на отримання результату, повністю визначеного цими початковими даними [12].

Це визначення відрізняється від вербального тлумачення алгоритму тим, що замість слів "точне приписання" використані слова комп'ютерної термінології – "…фіксована для деякого класу задач конфігурація апаратних засобів перетворення, передавання і зберігання даних…"

SH-модель це кортеж складових:

$$SH =< D, Q, q_0, q_f, G, P, M >, \tag{1.4}$$

де *D* – кінцева множина символів зовнішнього алфавіту;

Q – кінцева множина станів SH-моделі;

q_0, q_f – початковий і кінцевий стани, $q_0, q_f \in Q$;

G – конфігурація апаратних засобів моделі;

$$G = (X, U), \tag{1.5}$$

Де X – множина елементарних перетворювачів;

U – множина міжз'єднань;

P –програма;

M – зовнішня пам'ять.

Структура SH-моделі.

Перевагою машини Тюрінга є наявність незмінної структури (каркасу), яка зберігається для всіх алгоритмів, що моделюються. Це зумовило використання у визначенні моделі алгоритму слова "машина". Використання "каркасу" є зручним для побудови правила безпосереднього перероблення. Інші моделі алгоритмів не мають постійної структури. SH-моделі також не мають раз і назавжди встановленої структури. Однак, кожна конкретна модель алгоритму стосовно апаратної побудови має точно окреслену структуру, яка складається з двох множин: множини елементарних перетворювачів і множини з'єднань:

$$\begin{aligned} X &= \{x_1, x_2, \ldots, x_n\}; \\ U &= \{u_1, u_2, \ldots, u_m\}. \end{aligned} \tag{1.6}$$

Апаратні засоби містять один або декілька елементарних перетворень, зв'язаних між собою з'єднаннями. Кожний елементарний перетворювач виконує деяку операцію, ці операції можуть бути різними для різних перетворювачів. До множини елементарних перетворювачів належать також елементи тимчасової пам'яті, вони відображають реальні схемо технічні і системотехнічні пристрої - тригери, регістри, кеш та інші. Вибір ієрархічного рівня елементів тимчасової пам'яті відповідає ієрархічному рівню комп'ютерних схем, на якому реалізується алгоритм. Поняття "SH-модель" і "елементарний перетворювач" мають ієрархічний зміст, таким чином, для SH-

моделі справедлива властивість ієрархічності. Відзначимо, що абстрактні моделі обчислювачів такої властивості не мають.

Елементарний перетворювач x_i перетворює деяку сукупність вхідних даних d_i в сукупність вихідних результатів $d_i^{'}$:

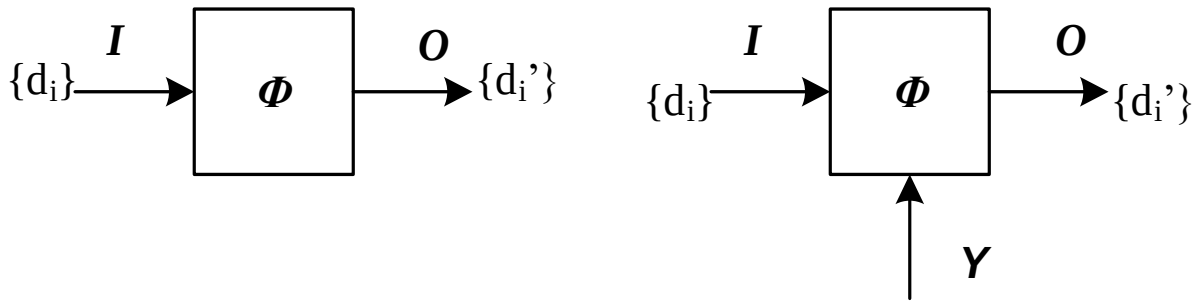
$$x_i : \{d_i\} \to \{d_i^{`}\} \tag{1.7}$$

Елементарний перетворювач – це модель неподільного елемента схеми апаратних засобів SH-моделі. Він заданий “чорною скринькою ” (рис. 1.7), який має входи, на які подаються вхідні дані, і виходи, з яких знімається результат перетворення даних. Операція виконувана “чорною скринькою” є елементарною. Перетворення даних виконується відповідно до функції, що задається “чорною скринькою”. Ця функція може бути змінена подачею сигналу “коду функції” на додатковий вхід Y. Розрізняємо два види “чорної скриньки ” [11]:

а) елементарний перетворювач без зовнішнього управління (рис. 1.7.а) - це трійка:

$$(I, \Phi, O), \tag{1.8}$$

де I - вхід, Φ - функція перетворення, O – вихід.



а) б)

Рисунок 1.7. Елементарний перетворювач, а) без входу керування; б) із входом керування.

б) елементарний перетворювач з зовнішнім керуванням (рис. 1.7.б) - це четвірка:

$$(I, \Phi, Y, O), \tag{1.9}$$

де I - вхід, Φ - функція перетворення , Y – вхід керуючого сигналу, O – вихід.

Використання елементарних перетворювачів в складі моделі алгоритму робить необхідним врахування системи зв'язків $U = \{u_i \mid i = 1, j\}$ між ними. Без цієї системи зв'язків дослідження апаратної реалізації алгоритму неможливе.

Конфігурація зв'язків впливає на інші характеристики спеціалізованих комп'ютерних засобів, зокрема, змінює значення апаратної і програмної складностей. Звідси наступна вимога: врахування конфігурації зв'язків або структурної складності - додаткової характеристики SH-моделі.

Властивості SH-моделі.

SH-модель має всі властивості, які випливають з вербального інтуїтивного тлумачення алгоритму.

I. Дискретність. Робота SH-моделі здійснюється множиною обмежених у часі кроків. Кожний крок може включати елементарні операції перетворення, передачу даних від одного елементарного перетворювача до іншого, а також операцію запису даних в елементи пам'яті.

II. Детермінованість. Кожний крок алгоритму повністю визначений функцією елементарного перетворювача і командою програми. Напрямок передачі даних від одного елементарного перетворювача до іншого точно визначений напрямком, що задають з'єднання або команди програми.

III. Елементарність. Операції перетворення, передачі і запису даних в елементи пам'яті SH-моделі є елементарними в просторі і часі.

IV. Масовість. Одна і та ж SH-модель може бути застосована для будь-якої кількості задач, які відрізняються лише набором вхідних даних при незмінному правилі безпосереднього перероблення.

V. Ієрархічність. Кожний елементарний перетворювач може бути представлений SH-моделлю нижчого ієрархічного рівня. З іншого боку, кожна SH-модель може бути використана як елементарний перетворювач вищого ієрархічного рівня.

Параметри SH-моделі.

До них відносяться вісім параметрів, сформованих ще А. Колмогоровим: правило початку; правило безпосереднього перетворення; правило закінчення; правило вводу; правило виводу; система вхідних даних; система вихідних даних; система проміжних даних.

Параметри алгоритму - правило початку та правило закінчення - задаються програмою. Системи проміжних і вихідних даних, правило безпосереднього перероблення - задаються апаратними або апаратно-програмними засобами. Система вхідних даних задається зовнішніми по відношенню до SH-моделі пристроями пам'яті.

Характеристики складності SH-моделі.

В процесах синтезу, аналізу і оптимізації SH-моделей пропонується використовувати чотири основних характеристики складності: апаратну, часову, програмну і структурну.

Апаратна складність – це кількість елементарних перетворювачів і елементів пам'яті деякого ієрархічного рівня апаратних засобів SH-моделі[24]:

$$A = |X|, \tag{1.10}$$

де X – множина елементарних перетворювачів та елементів пам'яті.

Визначення відображає ієрархічну побудову комп'ютерних засобів. Якщо під SH-моделлю розуміти операційний пристрій, то в якості елементарних перетворювачів розглядаються одно розрядні комірки або вентилі, для рівня регістрових передач елементами є операційні пристрої та регістри, на комп'ютерному рівні – вузли комп'ютера, на системному – комп'ютер.

При оцінці апаратної складності спеціалізованих комп'ютерних систем можливий інший підхід. Апаратну складність мікропроцесорів можна визначати кількістю транзисторів, розташованих на кристалі. Постійне збільшення апаратної складності (а зараз вона досягає 100 млн. транзисторів) створює сприятливі умови для розширення функціональних можливостей, для покращання практично всіх споживчих характеристик процесорів, зокрема

точності обчислень, збільшення продуктивності (без збільшення фізичних розмірів комп'ютерів, розширення функцій).

У роботі під апаратною складністю для спеціальних функцій буде розумітися кількість елементів структурної схеми пристрою (елементарних перетворювачів) деякого ієрархічного рівня, які використовують приблизно одну і ту саму кількість апаратних засобів кристалу спецпроцесора, та мають одинаковий час спрацювання.

Часова складність. Часова складність SH-моделі визначається кількістю елементів схеми, розташованих вздовж максимального критичного шляху розповсюдження сигналу [24]:

$$L = |maxX_i|, \tag{1.11}$$

де $maxX_i$- кортеж елементів SH-моделі, що належать до максимального критичного шляху розповсюдження сигналу, включаючи повторні проходження елементів в циклі.

Приймаючи, що елемент SH-моделі деякого ієрархічного рівня має затримку 1 умовну часову одиницю, то тоді, часову складність можна визначати як кількість елементів у критичному шляху проходження сигналу. Саме таким чином у роботі проводиться розрахунок часової складності.

Перехід від часової складності до визначення часу спрацювання схеми наведений у наступній формулі:

$$T = \sum_{\tau_{ei} \in \max |e_i|} \tau(e_i), \tag{1.12}$$

де $\tau(e_i)$ реальна затримка сигналу на елементі e_i.

Програмна складність. Проаналізуємо часову діаграму роботи деякого пристрою на рівні регістрових передач з точки зору інформації, що міститься в ній. Часова діаграма уявляє собою двомірну таблицю, на осі абсцис якої позначені тактові імпульси роботи спецпроцесора, на осі ординат - входи керування пристроями функціональних схем.

На рівні спецпроцесора кожній асемблерній команді відповідає власна мікропрограма (часова діаграма). Для кожної мікропрограми конфігурація сигналів керування фіксована. Програмна складність визначається логарифмічною мірою Р – ступенем нерегулярності (ентропії) часової діаграми [24]:

$$P = -F \log_2 \frac{F}{n_1 \cdot m_1}, \tag{1.13}$$

де $F = \sum_L f_l$;

n_l – кількість входів керування;

m_l – кількість тактових імпульсів часової діаграми;

f_l - кількість сигналів керування l-того фрагмента часової діаграми для обраного рівня ієрархії побудови апаратних засобів;

l – кількість фрагментів часової діаграми, конфігурації яких не повторюються.

Умова вибору фрагментів наступна:

$$\forall i, j; i \neq j \left\{ \left[\varphi_i \cap \varphi_j \neq \varphi_i \vee \varphi_j \right] \Rightarrow |f_l| = |\varphi_i| + |\varphi_j| \right\} \vee \left\{ \left[\varphi_i \cap \varphi_j = \varphi_i \vee \varphi_j \right] \Rightarrow |f_l| = |\varphi_i| \right\}, \tag{1.14}$$

де $\varphi_{i,} \varphi_j$ - фрагменти часової діаграми.

Оцінка програмної складності необхідна для того, щоб ефективно провести оцінку компілятора для розробки програмного забезпечення під спеціалізовану комп’ютерну систему, яка проектується. Для програмування спеціалізованих комп’ютерних систем використовують мови програмування низького рівня – асемблери. Для кожного окремого спецпроцесора використовується свій машинний код для програмування його роботи, наприклад виконання спеціальних функцій, тому значення програмної складності дає можливість провести ефективний аналіз затрат часу на розробку коду програми для роботи системи та оцінку складності його компілятора.

Структурна складність алгоритмічного пристрою є ентропія матриці інциденцій [23]:

$$S_u = \sum_{n\in N}\varphi_n \cdot \log_2 \frac{\sum_{n\in N}\varphi_n}{q\cdot r} - (q\cdot r - \sum_{n\in N}\Phi_n)\cdot \log_2 \frac{q\cdot r - \sum_{n\in N}\Phi_n}{q\cdot r} \quad (1.15)$$

де φ_n - кількість міжз'єднань n-го фрагмента матриці інциденції схеми;

$q \times r$ - розмір матриці.

У формулі (1.15) використовується лише множина з'єднань фрагментів матриці, що не повторюється. Вибір фрагментів проводиться з логічних умов:

$\forall$ i,j; i $\neq$ j $\{[\psi_i \cap \psi_j \neq \psi_i \vee \psi_j] \Rightarrow E=|\psi_i| + |\psi_j|\} \times \{[\psi_i \cap \psi_j = \psi_i \vee \psi_j] \Rightarrow \varphi=|\psi_i|\}$, i,j = $\overline{1,N}$,

де ψ_i, ψ_j - i-тий та j-тий фрагменти матриці інциденції.

Структурна складність відображає ступінь нерегулярності зв`язків схеми деякого рівня ієрархії побудови апаратних засобів і обчислюється за формулою [23]:

$$S = -E \cdot log_2 \frac{E}{q\cdot r} \quad (1.16)$$

де E - кількість елементів матриці інциденцій без регулярно розташованих елементів; q, r - розмір матриці.

Структурна складність SH-моделі визначається аналогічним способом, що і програмна. Відмінність лише в тому, що об'єктом розрахунків є матриця інциденцій. Одержання структурної складності проводиться в три етапи:

1. Схема SH-моделі перетворюється в орграф;
2. Орграф кодується у вигляді матриці інциденції;
3. Проводиться розрахунок значення нерівномірності матриці інциденції.

Аналіз структурної складності SH-моделей спеціальних функцій для спеціалізованих комп'ютерних систем дає змогу проводити ефективну параметричну оптимізацію структури таких функцій, що в результаті дає змогу реалізувати таку структуру системи, на синтез якої буде затрачено менше часу у порівнянні з існуючими структурами.

1.3.2. H-модель

Якщо говорити про спеціалізовані комп’ютерні засоби, то основною метою їх створення є досягнення оптимального рішення тої чи іншої задачі, більшої продуктивності системи, ніж це дає нам універсальна КС. Одним із основних аспектів є оптимізація часової складності. На універсальних комп’ютерних системах це досягається оптимізацією програмного забезпечення. Але час на вибірку команди з пам’яті є досить великий, тому наступним кроком оптимізації спеціалізованих комп’ютерних систем – є побудова алгоритмів арифметичних операцій, а головне спеціальних функцій, виключно апаратними засобами.

Використання апаратно-програмної моделі алгоритму (SH-моделі) дозволяє розширити її функції, що суттєво розширює поняття “масовість”[18]. Крім властивості масовості в традиційному розумінні, SH-модель дозволяє збільшити кількість класів розв’язувальних задач. Це робиться за рахунок зміни правила безпосереднього перероблення лише в апаратному середовищі, така властивість називається “багатофункціональність”[23].

При дослідженні властивостей масовості та багатофункціональності, розглядатимемо тільки один параметр алгоритму – правило безпосереднього перероблення. Саме цей параметр алгоритму дозволяє реалізувати перераховані властивості.

Структура правила безпосереднього перероблення SH-моделі принципово не відрізняється від структури абстрактних алгоритмів, подібно абстрактному алгоритму вона складається з функціонального блоку та блоку керування. Ця структура в загальному випадку подібна до структури моделі абстрактного алгоритму. Проте існує варіант структури, коли ця подібність порушується, цей варіант структури SH-моделі не містить програмного блоку.

Перетворення даних, що задається алгоритмом, здійснюється множиною елементарних перетворювачів, а напрямки передачі проміжних результатів визначаються множиною об'єднуючих їх зв'язків. Для моделей абстрактних алгоритмів такий варіант структури не існує. SH-модель, для якої правило

безпосереднього перероблення реалізується лише апаратними засобами, така модель має назву однофункціональна Н-модель (H-Hardware)[18].

Визначення Н-моделі алгоритму. Н-модель призначена для синтезу, аналізу та оптимізації комбінаційних обчислювальних засобів [18].

Формальни опис – Н-модель алгоритму це п’ятірка [18]:

$$H =< A, Q, q_0, q_f, G, >, \tag{1.17}$$

де A – кінцева множина символів зовнішнього алфавіту;

Q – кінцева множина станів Н-моделі;

q_0 і q_f – початковий і кінцевий стани $(q_0, q_f \epsilon Q)$;

G – конфігурація апаратних засобів моделі: $G = (X, U)$,

де X – множина елементарних перетворювачів;

U – множина з'єднань.

Така модель відрізняється від попередньої SH-моделі відсутністю програми в кортежі параметрів. Основна мета розробки апаратної моделі конкретного алгоритму – це оптимізація часової складності і ліквідація програмної складності. Побудова Н-моделі базується на першій аксіомі комп’ютерних алгоритмів: Алгоритм може бути виконаний апаратними засобами.

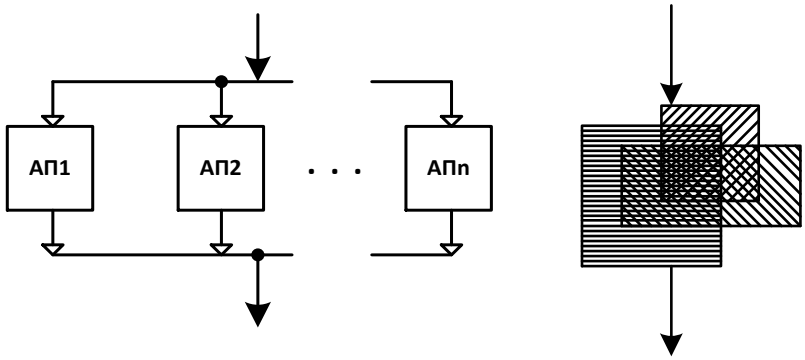
Н-модель алгоритму володіє властивістю масовості в класичному розумінні. Вона дозволяє розв’язувати задачі лише деякого одного класу, які розрізняються тільки наборами вхідних даних. До моделей такого типу відносяться, наприклад, моделі комбінаційних пристроїв – суматори, матричні перемножувачі, дешифратори та інші. Дана модель є суттєвою при побудові саме спеціалізованих комп’ютерних систем. В універсальних КС суттєва затримка при виконанні алгоритмів спеціальних функцій є при виборі коду команди із пам’яті. Саме для цього і існує Н-модель алгоритму – реалізація такої спеціалізованої комп’ютерної системи, в якій програмна складність обмежується синхронізацією подачі вхідних даних та зняття результату з

виходу системи, що дає змогу значно скоротити час на спрацювання функцій обробки даних.

1.3.3. RH-модель

У спеціалізованих комп'ютерних системах виникає потреба суміщення декількох операцій при реалізації спеціальних функцій. Створення багатофункціонального пристрою можливе двома шляхами: реалізація декількох паралельних схем виконання різних алгоритмів, повністю незалежних одна від одної, які розміщуються на одному кристалі (рис. 1.8.а); або, створення пристрою, в якому деякі алгоритми використовують ті ж самі апаратні засоби, але послідовно у часі (рис. 1.8.б).

Перший варіант є досить простим в реалізації. Він передбачає паралельне виконання алгоритмів. Тобто реалізовані алгоритмічні пристрої розміщуються паралельно відносно один одного і за допомогою подачі сигналу керування, з загальної шини відбувається подача вхідних даних на виконання необхідного алгоритму.



а) б)

Рисунок 1.8. Схеми багатофункціональних пристроїв з а) паралельним з'єднанням алгоритмічних пристроїв(АП); б) апаратним суміщенням функцій.

Другий варіант передбачає суміщення алгоритмів на одній структурі. Якщо алгоритми використовують схожі арифметичні операції, наприклад операція множення, або послідовність арифметичних операцій однакова у деякого набору алгоритмів, тоді такі алгоритми можна сумістити на одній

структурі, переходячи від одного алгоритму до іншого за допомогою мультиплексорів.

Правило безпосереднього перероблення Н-моделі може бути змінено за допомогою зовнішньої дії – сигналу, що задає нову конфігурацію потоків даних в середовищі елементарних перетворювачів. При цьому множина елементарних перетворювачів і конфігурація фізичних зв’язків залишається незмінною. Апаратна складність також залишається незмінною. Назвемо сукупність таких сигналів кодом реконфігурації. Наприклад, Н-модель арифметико-логічного пристрою змінює свої функції по коду реконфігурації: сумування може бути змінено на віднімання, арифметичні операції можуть бути змінені на логічні. В той же час в структурі цієї моделі відсутній програмний блок. Така Н-модель алгоритму із правилом безпосереднього перероблення, яке допускає зміну функцій перетворення даних за кодом реконфігурації, називається багатофункціональною Н-моделлю, або RH-моделлю (R-reconfigured). Таким чином, запропонований [25] ще один різновид Н-моделі, який має додатковий вхід коду реконфігурації. Ця модель володіє властивістю багатофункціональності, дозволяє реконфігурувати внутрішню систему зв’язків Н-моделі.

1.4. Задачі дослідження

Для побудови високоефективних спецпроцесорів, що реалізують спеціальні функції, зокрема для опрацювання сигналів, із врахуванням їх характеристик складності найкраще застосовувати SH-моделі. Для цього потрібно розв’язати такі задачі:

- провести аналіз й отримати значення характеристик складності арифметичних пристроїв спеціальних функцій спецпроцесора опрацювання сигналів;
- розробити структури SH-моделей пристроїв множення та пристроїв, що реалізують алгоритми згортки та швидкого перетворення Фур’є, оптимізовані за значеннями характеристик складності;

- розробити SH-модель конвеєрного пристрою множення, із затримкою сходинки конвеєра не більшою, ніж затримка на одному багаторозрядному суматорі, та відсутністю залежності від розмірності вхідних даних;
- вдосконалити метод обчислення структурної характеристики складності для SH-моделей на основі матричних побудов пристроїв спецпроцесора опрацювання сигналів;
- розробити VHDL-моделі оптимізованих за характеристиками складності арифметичних пристроїв та пристроїв спеціальних функцій спецпроцесора опрацювання сигналів.

Висновки до розділу

1. Проаналізовано основні моделі формальних алгоритмічних систем – машина Тюрінга, алгоритмічна система Маркова, алгоритмічні процеси Колмогорова й показано, що вони не враховують апаратне забезпечення, що не дає змоги оптимізувати спеціалізовані комп'ютерні системи з врахуванням всіх характеристик складності.
2. Проаналізовано моделі апаратних засобів спецпроцесорів – архітектуру фон Неймана і показано, що така модель дає змогу проектувати комп'ютерні засоби з врахуванням лише двох характеристик – продуктивність та обсяг обладнання, а інші характеристики складності, як програмна, алгоритмічна складність не враховуються.
3. Проаналізовано моделі апаратно-програмних засобів спеціалізованих комп'ютерних систем – SH-модель, H-модель, RH-модель і показано, що вони дають змогу оптимізувати спеціалізовані комп'ютерні системи, що реалізують спеціальні функції, із врахуванням апаратної, часової, програмної та структурної характеристик складності.
4. Сформульовано задачі дослідження характеристик складності SH-моделей спеціальних функцій опрацювання сигналів, та їх застосування для оптимізації спецпроцесора.

РОЗДІЛ 2
ОПТИМІЗАЦІЯ SH-МОДЕЛЕЙ ОПЕРАЦІЇ МНОЖЕННЯ, ЯК ОСНОВИ СПЕЦІАЛЬНИХ ФУНКЦІЙ ОПРАЦЮВАННЯ СИГНАЛІВ

Структурний синтез – це процедура перетворення символьного представлення алгоритму в комп’ютерну схему. Задачею структурного синтезу є пошук оптимальної структури технічного об'єкта, для реалізації заданих функцій в рамках обраного проекту. Результати структурного синтезу можуть бути представлені у вигляді таблиць порівняння декількох варіантів структурних схем. Структурна схема відображає арифметичні та логічні операції, які передбачені алгоритмом, та систему зв’язків між ними [26].

Існують варіанти представлення схем: Блок-схеми; UML-схеми (англ. Unified Modeling Language); Структурні схеми; Функціональні схеми; і т.д. UML-схеми базуються на уніфікованій мові моделювання, яка не зовсім вдало підходить для побудови комп’ютерної схеми. Частіше вона використовується для бізнес планів, та є невід’ємною частиною уніфікованого процесу розробки програмного забезпечення. Блок-схеми також не зовсім підходять для зображення чи опису структури пристрою. В пристроях обробки даних кращим буде наближене зображення схеми до проектованого алгоритму, так як така схема буде найбільш читабельною. Арифметичні операції можна зображати геометричними фігурами, наприклад прямокутниками, а зв’язки між ними - шинами, які з’єднують їх.

Звичайно структурна схема використовується для уявлення про складність пристрою, об’єм обладнання. Структурний синтез складається з декількох послідовних кроків:

- Розробка структурної схеми пристрою, методом перетворення символьного представлення алгоритму в структуру пристрою;
- Аналіз можливості апаратного виконання арифметичних залежностей;
- Розглядаються способи конвеєризації схеми:

1. Варіанти визначення набору ступенів конвеєра;
2. Задання оптимального такту конвеєра;
3. Оптимізація часу спрацьовування сходинок конвеєра.

- Оцінка можливості використання паралелізму виконання алгоритму;
- Остаточне узгодження часових співвідношень і кінцевої структури пристрою проводиться методами параметричної оптимізації.

Під оптимізацією розуміють процес вибору найкращого варіанту із всіх можливих. З точки зору інженерних розрахунків методи оптимізації дозволяють вибрати найкращий варіант конструкції, найкраще розділення ресурсів і т.п.

У процесі розв'язку задачі оптимізації зазвичай необхідно знайти оптимальне значення деяких параметрів, які визначають дану задачу. При розв'язку інженерних задач їх прийнято називати проектними параметрами, а в економічних задачах їх зазвичай називають параметрами плану. В якості проектних параметрів можуть бути значення лінійних розмірів об'єкта, маси, температури і т.п. Число n проектних параметрів x1, x2, …,xn характеризує розмірність (і степінь складності) задачі оптимізації [27].

Вибір оптимального рішення або порівняння двох альтернативних рішень проводиться за допомогою деякої залежної величини (функції), яка визначається проектними параметрами. Ця величина називається цільовою функцією (або критерієм якості). В процесі розв'язку задачі оптимізації мають бути знайдені такі значення проектних параметрів, при яких цільова функція приймає мінімум (або максимум). Таким чином, цільова функція – це глобальний критерій оптимальності в математичних моделях, за допомогою яких описуються інженерні або економічні задачі.

Цільову функцію можна записати у вигляді:

$$u = f(x1, x2, \ldots, xn) \qquad (2.1)$$

Прикладами цільової функції, які зустрічаються в інженерних і економічних розрахунках, являються міцність або маса конструкції,

потужність устаткування, об’єм випуску продукції, вартість перевезення вантажів, прибуток і т.п. (значення характеристик складності комп’ютерного алгоритму при проектуванні схем).

У випадку одного проектного параметра (n=1) цільова функція (2.1) являється функцією однієї змінної, а її графік – деяка крива на площині. При n=2 цільова функція являється функцією двох змінних, і її графіком є поверхня [28].

Варто відмітити, що цільова функція не завжди може бути представлена у вигляді формули. Іноді вона може приймати тільки деякі дискретні значення, задаватися у вигляді таблиці і т. п. У всіх випадках вона повинна бути однозначною функцією проектних параметрів.

Цільових функцій може бути декілька. Наприклад, при проектуванні виробів машино будівництва одночасно потрібно забезпечити максимальну надійність, мінімальну матеріалоємність, максимальний корисний об’єм (або вантажопідйомність). Деякі цільові функції можуть виявитися несумістимими, в таких випадках необхідно вводити пріоритет тої чи іншої цільової функції.

Можна виділити два типи задач оптимізації – безумовні і умовні. Безумовна задача оптимізації полягає в знаходженні максимуму або мінімуму діючої (дійсної) цільової функції від n діючих змінних і визначенні відповідних значень аргументів на деякій множині Б n-розмірного простору. Зазвичай розглядаються задачі мінімізації, до них легко зводяться і задачі на пошук максимуму, шляхом зміни знаку цільової функції на протилежний.

Умовні задачі оптимізації, або задачі з обмеженнями – це такі задачі, при формулюванні яких задаються деякі умови (обмеження) на множині Б. Ці обмеження задаються сукупністю деяких функції, які задовольняють рівнянням або нерівностям.

Обмеження-рівності виражають залежність між проектними параметрами, яка повинна враховуватись при знаходженні рішення (розв’язку). Ці обмеження відображають закони природи, наявність ресурсів, фінансові вимоги і т. п.

В результаті обмежень область проектування, яка визначається всіма проектними параметрами, може бути суттєво зменшена в відповідності з фізичною сутністю задачі.

Теорія і методи розв’язку задач оптимізації при наявності обмежень складають предмет дослідження одного із важливих розділів прикладної математики – математичного програмування. В даній дисертації розглядаються саме задачі оптимізації з обмеженнями.

Операція множення є базовою операцією більшості спеціальних функцій опрацювання сигналів. Реалізація пристроїв множення можлива трьома шляхами: на базі багаторозрядного сматора, на базі сукупності одноразрядних суматорів та матричні пристрої множення на базі сукупності комірок Гілда. Кожен із шляхів реалізації має свою SH-модель з власними значеннями характеристик складності на основі яких і приймається рішення про вибір того чи іншого шляху оптимізації при заданих параметрах.

Проведемо детальний аналіз пристроїв множення за характеристиками складності. Основним завданням є проведення структурного синтезу, та параметричної оптимізації конвеєрного пристрою множення, в якому затримка на сходинці конвеєра буде меншою, або рівною затримці на одному багато розрядному суматорі. Ще одним аспектом є те, що кількість сходинок конвеєра не повинна залежати від розрядності проектованого пристрою.

2.1. Обчислення структурної складності SH-моделей спеціальних функцій

Оптимізація SH-моделі проводиться одночасно за всіма характеристиками складності, при чому основним є оптимізація часовоє складності. Найбільш складним є обчислення структурної складності відомим методом [29, 31, 32, 33], оскільки пов’язано з отриманням матриці інциденцій, пошуку у ній регулярно розташованих елементів з подальшим їх вилученням, і лише після цього отримується значення структурної складності за формулою (1.16).

Наприклад, маємо наступну матричну структуру (рис. 2.1). Дана структура представляє собою Н-модель якогось пристрою, в якому присутні два типи елементарних перетворювачів: елементарний перетворювач типу 1 (ЕП1), та елементарний перетворювач типу 2 (ЕП2). Елемент (RG), який розташований внизу структури – це пам’ять, для збереження результату опрацювання даних. Для формування матриці інциденцій даної схеми, усі елементарні перетворювачі на ній номеруються порядковими номерами (x0 – x16), та усі зв’язки між елементарними перетворювачами також нумеруються (0 – 27).

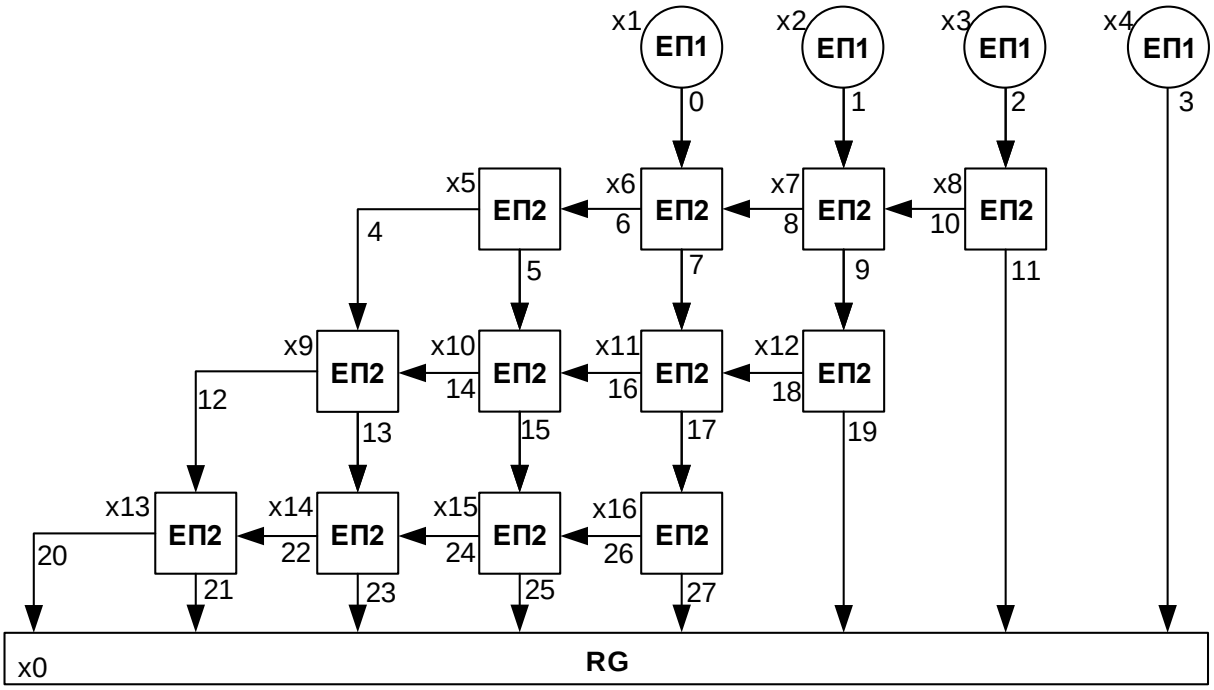


Рисунок 2.1. Н-модель пристрою X, що реалізує алгоритм Y

На основі такої структури пристрою (рис. 2.1) формуємо матрицю інциденцій (2.2), де по вертикалі розташовуємо нумерацію елементарних перетворювачів, а по горизонталі – зв’язків між ними. У матриці, на місцях де зв’язки перетинаються з елементарним перетворювачем ставимо одиниці зі знаком плюс, якщо сигнал виходить із елеметнтарного перетворювача, мінус, якщо сигнал входить у елементарний перетворювач.

I=		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
	x0				1								-1								-1	-1	-1		-1		-1		-1
	x1	1																											
	x2		1																										
	x3			1																									
	x4				1																								
	x5					1	1	-1																					
	x6	-1						1	1	-1																			
	x7		-1							1	1	-1																	
	x8			-1								1	1																
	x9					-1								1	1	-1													
	x10						-1									1	1	-1											
	x11							-1										1	1	-1									
	x12									-1										1	1								
	x13													-1								1	1	-1					
	x14														-1									1	1	-1			
	x15																-1									1	1	-1	
	x16																		-1									1	1

(2.2)

Наступним кроком у обчисленні структурної складності є пошук у матриці інциденцій регулярно розташованих елементів. Регулярно розташовані елементи – це такі елементи, які повторюються у матриці з певним кроком, для кожної конкретної матриці цей крок індивідуальний. У матриці (2.2) регулярно розташовані елементи виділені однаковими кольорами у матриці. Таким чином у матриці присутні 6 різнотипних елементів і вони ділять матрицю на розміри 3x3. Отже, використовуючи формулу (1.16) обраховуємо структурну складність пристрою $S = -6\log_2 \frac{6}{9} \approx 3.5$.

Пропонується вдосконалити метод обчислення структурної складності спеціальних функцій, об’єднуючи однотипні елементарні перетворювачі, або набори елементарних перетворювачів, які повторюються у структурі пристрою у блоки, що дасть змогу уникнути регулярно розташованих елементів матриці інциденцій, та зменшити її розміри. Алгоритм обчислення структурної складності буде наступним:

- пошук груп однотипних елементів у структурі пристрою;
- створення граф-схеми пристрою з об’єднаними у блоки елементами;
- формування матриці інциденцій граф-схеми;
- отримання значення структурної складності.

Розглянемо роботу вдосконаленого методу на прикладі. У структурі (рис. 2.1) проведемо пошук однотипних елементів які об’єднуємо у блоки x1 та x2 (рис. 2.2).

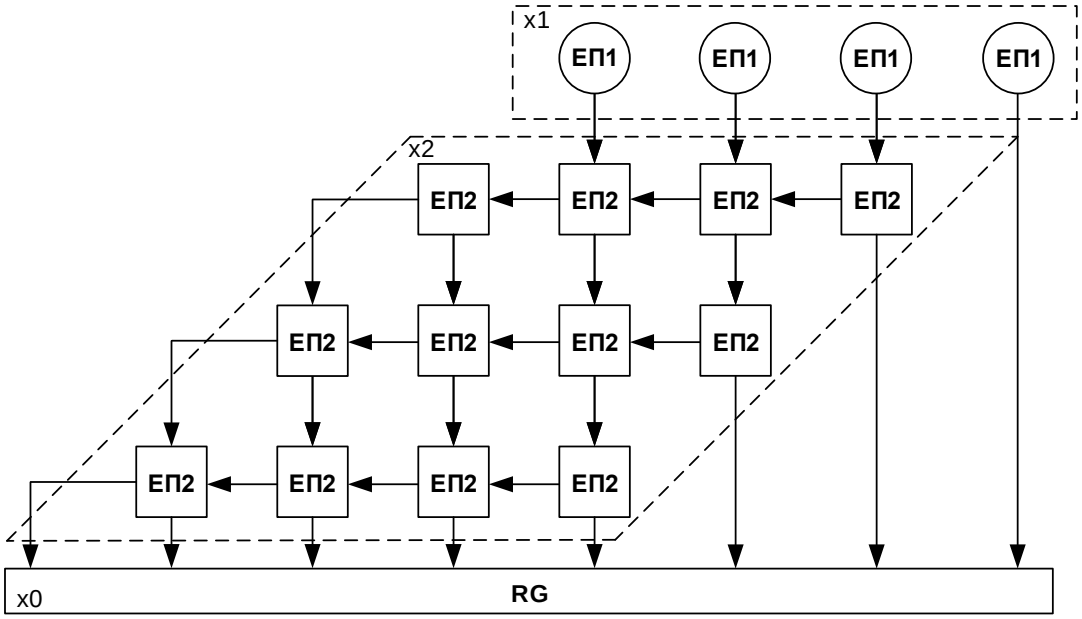


Рисунок 2.2. Визначення однотипних елементів Н-моделі та об’єднання їх у блоки

Зв’язки між блоками однотипних елементів об’єднуємо у шини, і в результаті отримуємо граф-схему пристрою (рис. 2.3), та створюємо матрицю інциденцій такої граф-схеми (2.3). Використовуючи формулу (1.16) проводимо обрахунок структурної складності $S = -6\log_2 \frac{6}{9} \approx 3.5$, як і в попередньому випадку.

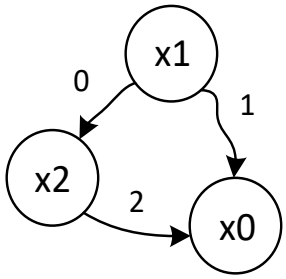


Рисунок 2.3. Граф-схема структури (рис. 2.2)

I=

	0	1	2
x0		-1	-1
x1	1	1	
x2	-1		1

(2.3)

Як видно з матриці інциденцій (2.3), об’єднання однотипних елементів схеми у блоки зменшує розміри матриці інциденцій, та значно спрощує процес обчислення структурної складності спеціальних функцій.

2.2. Пристрій множення на багаторозрядному суматорі

Цей пристрій містить у своєму складі лише один багаторозрядний суматор (рис. 2.4). Він був популярним в часи, коли апаратна складність у порівнянні з іншими характеристиками мала найбільшу вартісну вагу [35]. Тепер його конфігурація зберіглась у контролерах, вбудованих процесорах, деяких видах спеціалізованих систем. Але в згаданих схемах замість суматора використовується АЛП – вузол, який реалізує декілька функцій. Потенційна багатофункціональність, у порівнянні з іншими схемами, є перевагою пристрою множення на багаторозрядному суматорі. Суттєвим недоліком його є те, що він є лише частиною універсальної SH-моделі. Остання містить дві SH-моделі: SH-модель функціонального вузла і SH-модель пристрою керування. Характеристики складності пристрою керування залежать від обрання одного з декількох варіантів його побудови. Тому оцінка характеристик складності пристрою керування має проводитись для кожного конкретного випадку окремо.

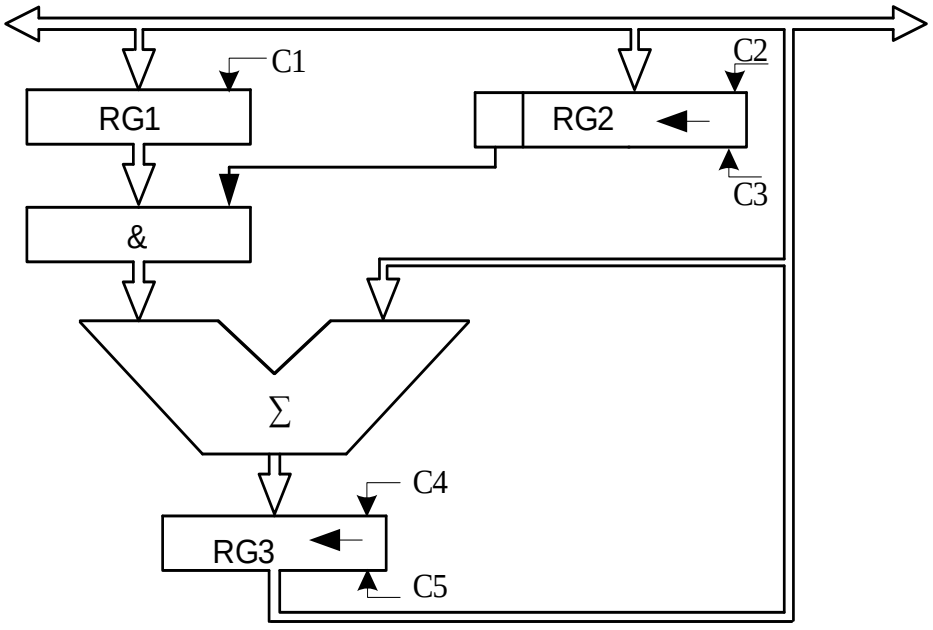


Рисунок 2.4. SH-модель пристрою множення на багаторозрядному суматорі

Принцип роботи пристрою множення пояснюється його функціональною схемою (рис.2.4) та часовою діаграмою (рис.2.5 б). Додатково зауважимо, регістри RG2 і RG3 є регістрами зсуву, сигнали управління: C1, C2, C3 є сигналами запису даних в регістри, а C3, C5 є сигналами зсуву даних в регістрах в бік старших розрядів.

Для послідовнісної схеми часова складність визначається підрахунком кількості тактів існування процесу відпрацьовування алгоритму, та відповідно до часової діаграми пристрою (рис.2.5. б) дорівнює [23]:

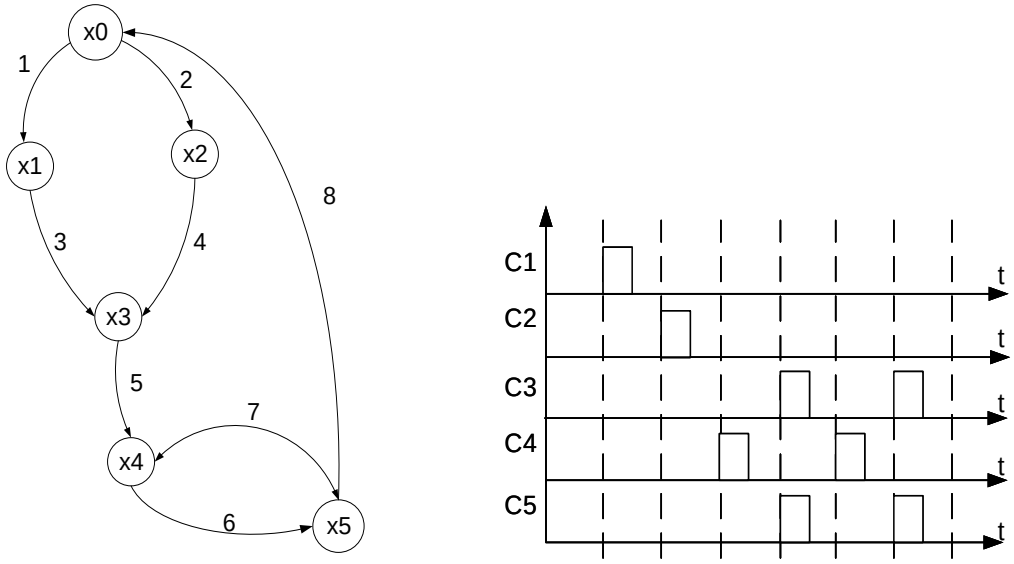
$$L= 2k (1 + n),$$

де k – одиниця складності часової діаграми;

n – кількість розрядів множника.

Програмна складність визначається за формулою (1.13) і рівна [23] $P = -5log_2 \frac{5}{5 \cdot 4} = 10.$

Основою для обчислення апаратної та структурної характеристик складності є граф схеми пристрою (рис.2.5 а). Кожна з вершин графу (Xi) відповідає одному елементарному перетворювачу схеми. За кількістю вершин апаратна складність у нашому прикладі дорівнює A=6.



а) б)

Рисунок 2.5. а) граф схема, б) часова діаграма, пристрою множення на багаторозрядному суматорі

Структурна складність визначається за допомогою матриці інциденцій (2.4) за виразом (1.16) $S = -16\log_2 \frac{16}{6 \cdot 8} = 25.6$.

I=

	1	2	3	4	5	6	7	8
x0	1	1						-1
x1	-1		1					
x2		-1		1				
x3			-1	-1	1			
x4					-1	1	-1	
x5						-1	1	1

(2.4)

Отже, даний пристрій не підходить для реалізації спеціальних функцій у спецпроцесорах опрацювання сигналів, оскільки він має високу часову, програмну та структурну характеристики складності.

2.3. Конвеєрний пристрій множення

Принципова різниця наступної схеми (рис. 2.6) від попередньої полягає в тому, що шина циклу, яка з'єднує вихід регістра Рг3 і правий вхід багато розрядного суматора відсутня, розірвана. На рисунку (рис.2.6) зображена Н-модель і-го ступеня конвеєрного пристрою множення: де Ri – множник; Bi - множене; Ai сума часткових проекцій; С – синхросигнали; Рг1, Рг2, Рг3 – регістри, які використовуються для зберігання розрядів проміжного результату А, множеного В і множника R; Σ – одно розрядний суматор; схема І-АБО – елемент мультиплексора МХ. Пристрій виконує наступну операцію: A = B x R + A0.

Кількість сходинок сумування матриці, рівна кількості розрядів множника *n*. Асимптотична продуктивність помножувача складає $1/(n\tau_e + \tau_r)$, де τ_e – час затримки переносу в одно розрядному суматорі; τ_r - час спрацювання регістра. Мультиплексор МХ служить для того, щоб при множенні на наступний розряд множника r_j пропускати частковий добуток через суматор СМ1 або обминати його, якщо $r_j = 0$.

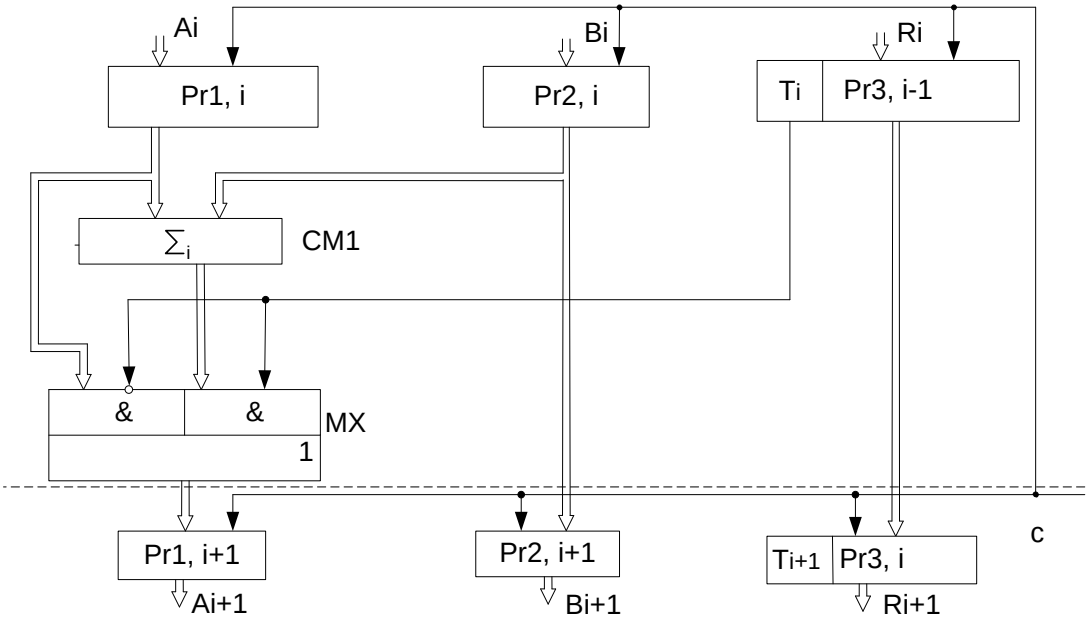
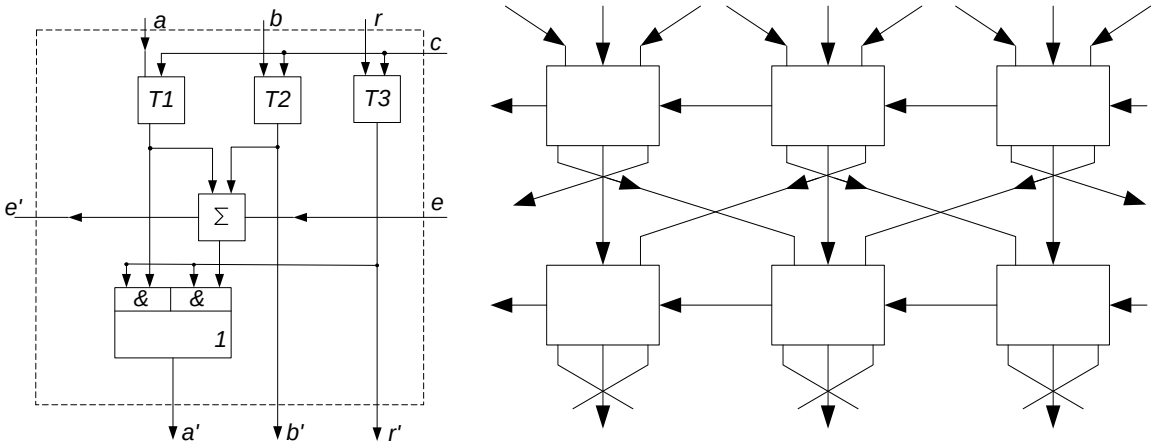


Рисунок 2.6. Н-модель конвеєрного пристрою множення

Аналізуємо наведений пристрій множення: часова складність не перевищує час спрацювання одного багато розрядного суматора, допускається суміщення з іншими алгоритмами, програмна складність рівна нулю, так як керування пристроєм обмежується тільки подачею синхроімпульсів, структура системи забезпечує зріст продуктивності. Але, в даній схемі є один вагомий недолік – кількість сходинок конвеєра тут напряму залежить від розрядності вхідних даних, а апаратна складність рівна n^2 (рис. 2.7), де n – розрядність вхідних даних.



а) б)

Рисунок 2.7. а) схема комірки систолічної матриці; б) фрагмента систолічної матриці, конвеєрного пристрою множення

Отже, даний пристрій не підходить для реалізації спеціальних функцій спецпроцесорів опрацювання сигналів, так як кількість сходинок конвеєра тут залежить від розрядності вхідних даних, що суперечить поставленим вимогам.

2.4. Матричні пристрої множення

Для синтезу схем точного множення двійкових чисел користуються матрицею, яку називають нульовим шаром схеми множення (рис. 2.8).

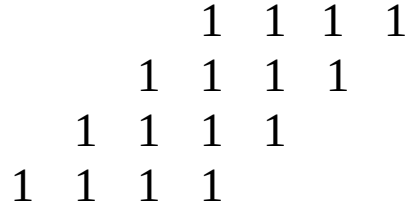


Рисунок 2.8. Нульовий шар матриці множення

Ця матриця є результатом порозрядної кон’юнкції двох двійкових множників, всі розряди яких дорівнюють одиницям. Цей нульовий шар є основою для апаратного синтезу різноманітних структур, які мають власні характеристики складності [104]. Для спрощення функціонального проектування пристроїв множення доцільно користуватися максимальними для заданої розрядної сітки двійковими множниками. Використання матриці заповненої тільки одиницями сприяє мінімізації помилок в наступному процесі згортання до значення добутку. Зауважимо, що структура комірок (схем кон’юнкцій), та система з’єднань нульового шару не змінюється залежно від алгоритму згортання. Серед багатьох існуючих алгоритмів згортання матриці оберемо два: з горизонтальним та діагональним розповсюдженням переносу.

2.4.1. Матричний пристрій множення з горизонтальним розповсюдженням переносу

Пристрій виконано у вигляді матриці з’єднаних між собою комірок. На рисунку (рис. 2.9) зображено таку матрицю з розрядністю вхідних даних A = B = 4 біти.

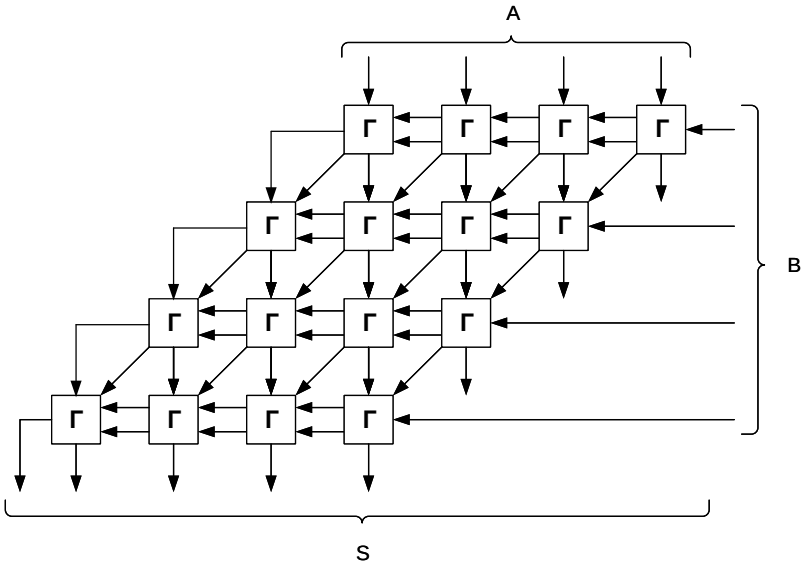


Рисунок 2.9. SH-модель пристрою множення з горизонтальним розповсюдженням переносу

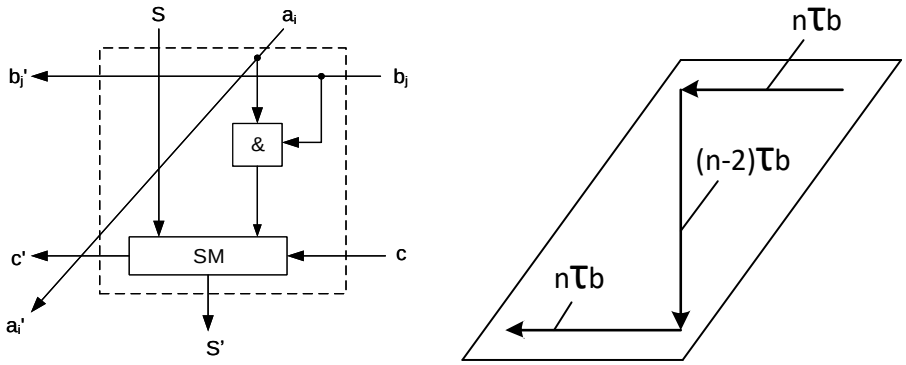
Тут символом Г позначено комірку матриці пристрою (Г – комірка Гілда). Це комбінаційна схема, управління її обмежене сигналом подачі на вхід двох n-розрядних співмножників. Структурна складність такої схеми рівна нулю (S=0), за рахунок того, що в структурі з'єднань спостерігається однорідність. H-модель комірки матриці приведена на рисунку (рис.2.10 а), де a_i – розряд множника A, b_j – розряд множника B; S – вхід часткової суми від попередньої комірки, S'– вихід часткової суми від суматора, c – перенос на вході одно розрядного суматора комірки, c' – перенос на виході комірки.

Схема кон'юнкції нульового шару розміщена в межах комірки. На рисунку (рис.2.10 б), наведені контури матриці, та максимальний критичний шлях розповсюдження сигналу. Він складається з двох горизонтальних та одної вертикальної ліній, з поміткою кількості елементарних перетворювачів, що належать відрізкам шляху. В сумі вони дорівнюють значенню часової складності: L=3n - 2.

Апаратна складність матриці рівна: $A=n^2$.

H-модель (рис.2.9) складна для сприйняття, її можна спростити не змінюючи внутрішнього змісту. На схемі разом із зв'язками переносів та часткових сум показані лінії нульового шару, останні не вносять змін у підрахунок структурної складності схеми, лише перевантажують її вигляд.

Тому, для спрощення аналізу і подальшої модифікації схеми, доцільно не показувати на рисунку лінії нульового шару.



а) б)

Рисунок 2.10. а) Н-модель комірки матриці (Г), б) максимальний критичний шлях розповсюдження сигналу, пристрою множення з горизонтальним розповсюдженням переносу

На рисунку (рис. 2.13) показана Н-модель без вхідних ліній нульового шару, але матриця кон’юнкцій збережена у комірках Гілда. Внутрішня структура комірки та максимальний критичний шлях сигналу такі самі, як показано на рисунку (рис.2.12 а, б).

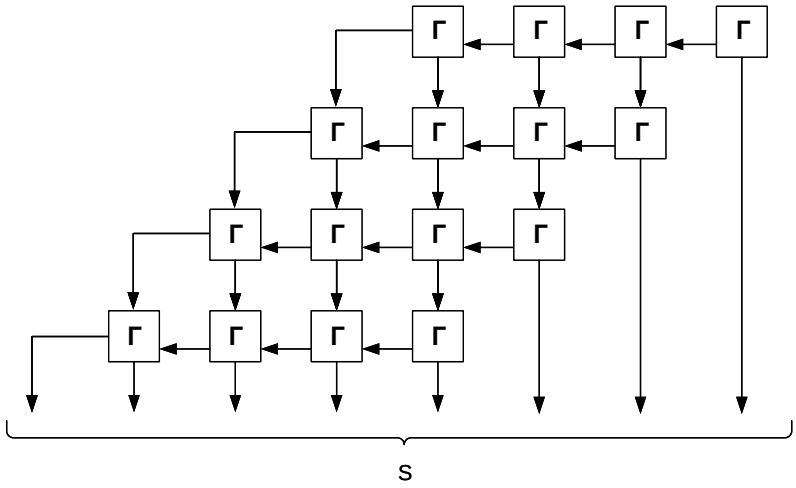
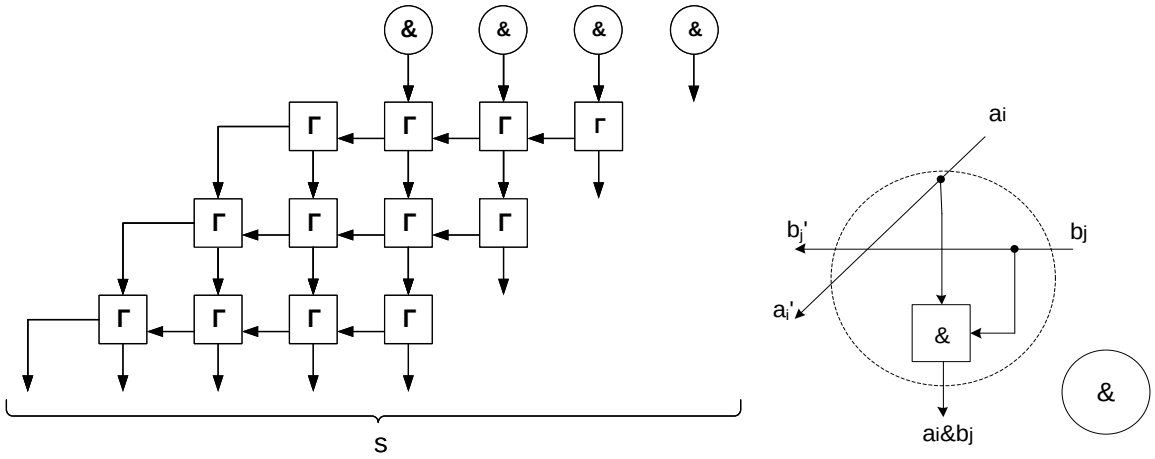


Рисунок 2.11. Н-модель пристрою множення з горизонтальним розповсюдженням переносу без вхідних ліній нульового шару

З метою зменшення часової складності вхідну матрицю можна модернізувати. По-перше, вхідні комірки першого горизонтального ряду можна замінити схемами кон’юнкції, як показано на рисунку 2.12 а. В

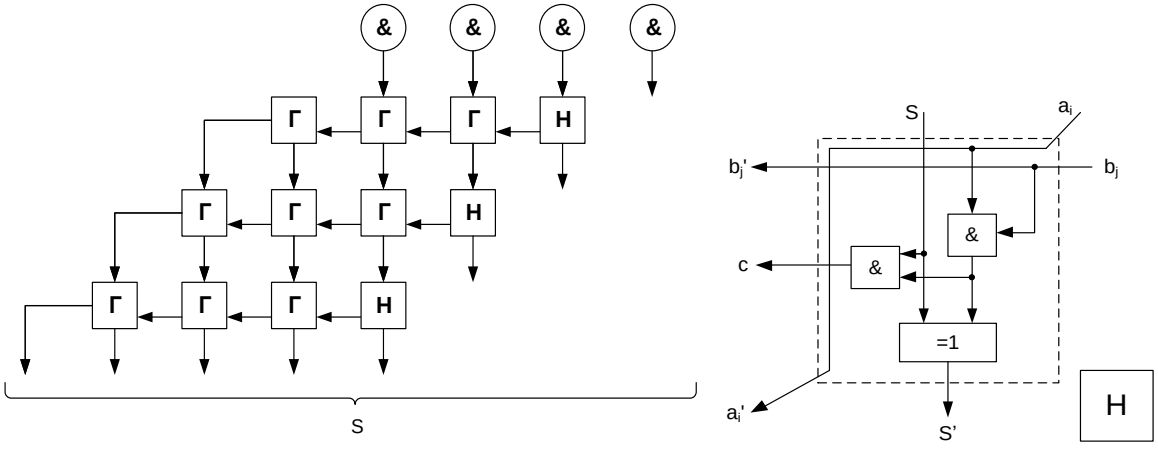
результаті зменшується часова та апаратна характеристики складності, але збільшується структурна.

По-друге, правий діагональний ряд комірок, можна замінити напівсуматорами, у зв'язку з тим, що на входи цих комірок не подаються сигнали переносу з попередніх комірок, так як її не існує. Модернізована схема, та внутрішня структура комірок напівсуматора (Н) зображені на рисунку 2.13 а та б. Внутрішня структура комірки Гілда та комірки кон'юнкції такі самі, як і в попередньому випадку (рис. 2.10 а, рис. 2.12 б).



а) б)

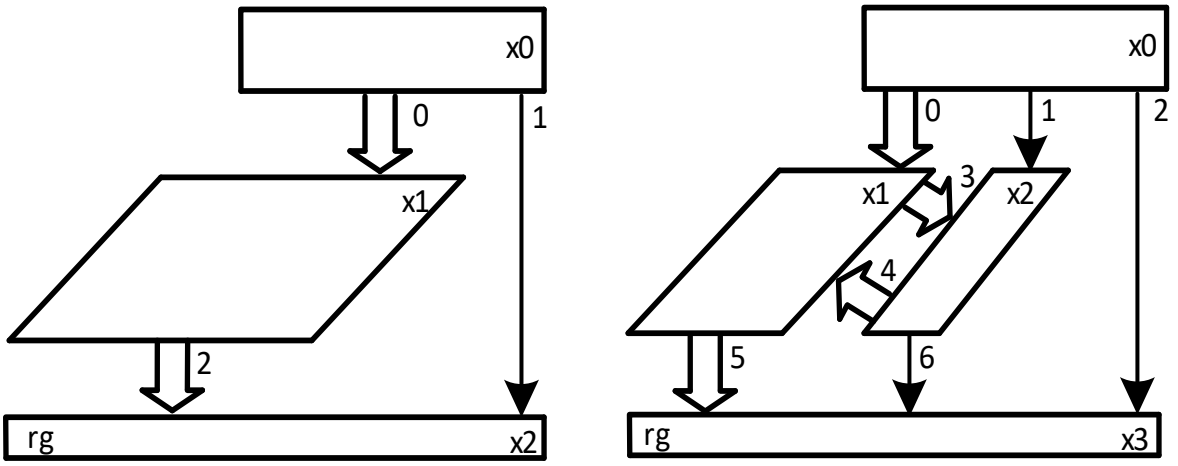
Рисунок 2.12. а) Н-модель пристрою множення із заміною першого ряду комірок матриці схемами кон'юнкцій, б) Н-модель комірки кон'юнкції



а) б)

Рисунок 2.13. а) Н-модель пристрою множення із заміною правого діагонального ряду схемою напівсуматора, б) Н-модель комірки напівсуматора

Підрахунки структурної складності полегшуються, якщо розбити структуру на однорідні блоки. Варіанти такого розбиття зображені на рисунку (рис.2.14 а, б), відповідно до схем (рис. 2.12 а, рис. 2.13 а). На блок-схемах додатково зображений регістр (rg), на який передається результат. У процесі розбиття виконується умова – кожний блок повинен мати всередині однорідну структуру зв'язків, тому їх структурна складність кожного блока дорівнює нулю. Таке розбиття означає перехід на вищий ієрархічний рівень представлення схеми, зменшується кількість вузлів та з'єднань, у схемі з'являється неоднорідність, що збільшує значення структурної складності.



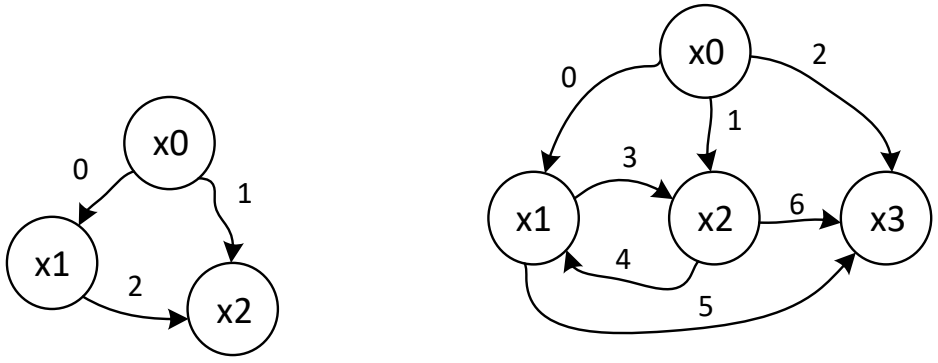
а) б)

Рисунок 2.14. Блок-схема пристрою множення а) із комірками кон'юнкції, б) із комірками напівсуматорами

Блок [x0] (рис.2.14 а) відповідає ряду комірок кон'юнкцій схеми (рис. 2.12 а). Блок [x1] відображає матрицю комірок [Г], розміром [(n-1)*n], схеми помножувача (рис. 2.12 а). Блок [x2] на рисунку (рис.2.14а) відображає регістр [rg], який використовується для збереження результату операції множення.

Схожа ситуація із описом блок-схеми (рис.2. 14 б), блок [x0] відповідає ряду комірок кон'юнкцій схеми (рис. 2.13 а), блок [x1] відображає матрицю комірок [Г], розміром $[(n-1)^2]$, блок [x2] відповідає ряду комірок [Н] схеми (рис. 2.13а), ну а блок [x3] позначає регістр (rg), який використовується для збереження результату операції множення.

Наступним кроком, у підрахунку структурної складності, є побудова графів кожної із розглянутих структур і аналіз матриць інциденцій. На рисунку (рис. 2.15 а, б) зображені графи, побудовані згідно з блок-схемами (рис. 2.14 а, б).



а) б)

Рисунок 2.15. Граф-схеми оптимізованих матричних пристроїв множення з горизонтальним переносом, зображених на рисунках 2.12 а та 2.13 а відповідно

Відповідно до граф-схем оптимізованих пристроїв множення з діагональним переносом було сформовано матриці інциденцій (2.5) для граф-схеми на рисунку 2.15 а, та (2.6) для граф-семи на рисунку 2.15 б.

I=

	0	1	2
x0	1	1	
x1	-1		1
x2		-1	-1

(2.5)

I=

	0	1	2	3	4	5	6
x0	1	1	1				
x1	-1			1	-1	1	
x2		-1		-1	1		1
x3			-1			-1	-1

(2.6)

Структурну складність обчислюємо за допомогою матриці інциденцій за виразом (1.16). Для пристрою множення, що на рисунку 2.12 а, значення структурної складності $S = -6\log_2 \frac{6}{9} \approx 3.5$. Для пристрою множення, який

зображений на рисунку 2.13 а, значення структурної складності $S = -14\log_2\frac{14}{4\cdot 7} \approx 14$.

Отже, провівши всі необхідні обчислення, ми можемо сформувати таблицю характеристик складності для розглянутих пристроїв множення з горизонтальним розповсюдженням переносу (табл. 2.1).

Таблиця 2.1

Характеристики складності Н-моделей ПМ з горизонтальним розповсюдженням переносу [104]

схема	L	A	S	P
рис. 2.11	3n-2	n^2	0	0
рис. 2.12 а	3n-4	n^2-n	3.5	0
рис. 2.13 а	3n-6	$(n-1)^2$	14	0

Розглянуті Н-моделі пристрою множення з горизонтальним розповсюдженням переносу розрізняються за характеристиками складності. Вибір однієї з них повністю залежить від напрямку використання. Для реалізації у крупно серійних виробах, у яких пристрій множення використовується як самостійний, слабо зв’язаний з іншими вузлами системи, можуть бути рекомендовані варіанти, наведені на рисунках (рис.2.12 а, рис.2.13 а), ці варіанти мають достатньо високу швидкодію. Їх структурна складність відносно варіанту (рис. 2.11) зросла, але збільшилась швидкодія. Варіант (рис. 2.11) привабливий для спрощених, мало розрядних процесорів з мінімізованою структурною складністю.

На діаграмі (рис. 2.16) приведено графічне представлення залежності кількості елементарних перетворювачів (вісь Y) необхідних на реалізацію кожного з розглянутих матричних пристроїв множення з горизонтальним переносом, в залежності від розрядності вхідних даних (вісь X). Аналізуючи дані з діаграми можемо побачити, що проведена параметрична оптимізація пристрою множення (рис. 2.11) дала змогу отримати SH-моделі з покращеними значеннями апаратної характеристики складності (рис. 2.12 а,

рис. 2.13 а). Виходячи з отриманих значень апаратної складності проаналізованих пристроїв отримано наступні покращення: оптимізований пристрій множення (рис. 2.12 а) має покращення значення апаратної складності 12% при розрядності вхідних даних у 8 біт, але при рості розрядності вхідних даних відсоток покращення зменшується і становить 1,5% при вхідних даних 64біти. Оптимізований пристрій множення (рис. 2.13 а) має покращення значення апаратної складності 23% при розрядності вхідних даних у 8 біт, але при рості розрядності вхідних даних відсоток покращення зменшується і становить 3,1% при вхідних даних 64біти.

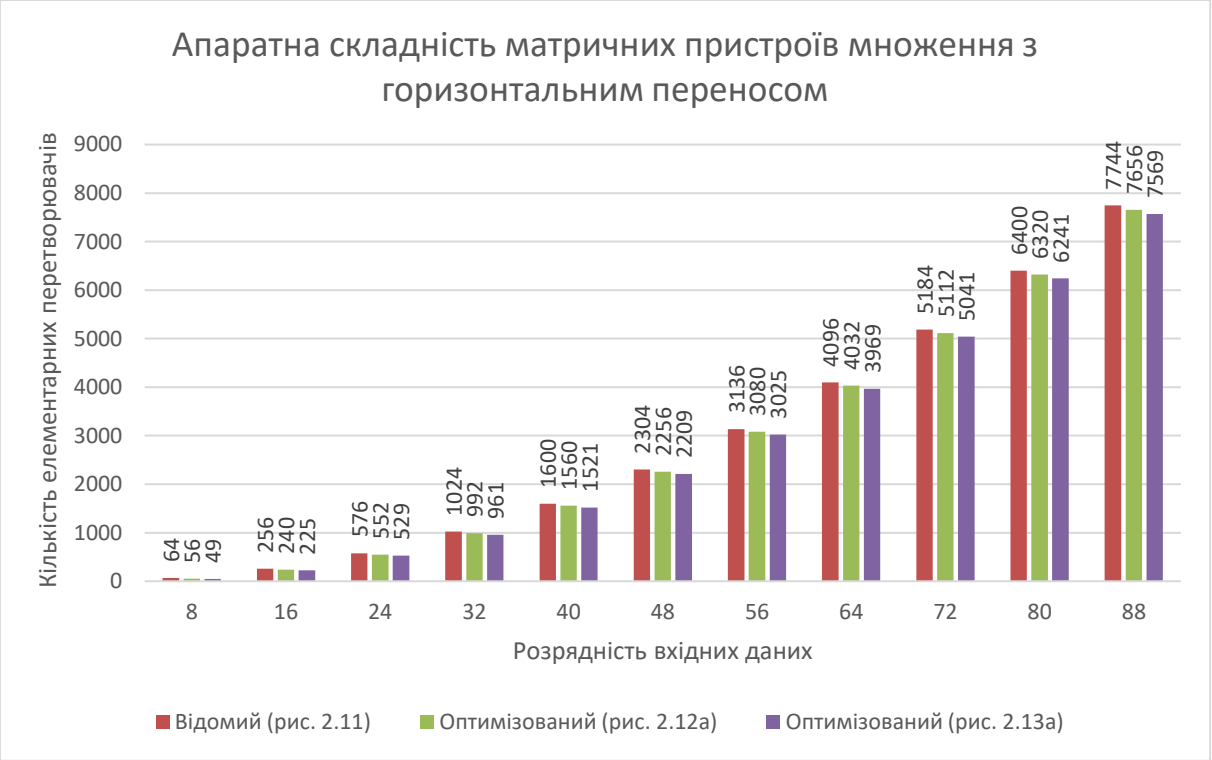


Рисунок 2.16. Порівняльна діаграма значень апаратної складності Н-моделей матричних пристроїв множення з горизонтальним переносом

На діаграмі (рис. 2.17) приведено графічне представлення залежності кількості елементарних перетворювачів (вісь Y) які належать максимальному критичному шляху розповсюдження сигналу у кожному з розглянутих матричних пристроїв множення з горизонтальним переносом, в залежності від розрядності вхідних даних (вісь X). Аналізуючи дані з діаграми можемо побачити, що проведена параметрична оптимізація пристрою множення (рис.

2.11) дала змогу отримати SH-моделі з покращеними значеннями часової характеристики складності (рис. 2.12 а, рис. 2.13 а). Виходячи з отриманих значень часової складності проаналізованих пристроїв отримано наступні покращення: оптимізований пристрій множення (рис. 2.12 а) має покращення значення часової складності 9% при розрядності вхідних даних у 8 біт, але при рості розрядності вхідних даних відсоток покращення зменшується і становить 1% при розрядності вхідних даних 64 біта. Оптимізований пристрій множення (рис. 2.13 а) має покращення значення часової складності 18% при розрядності вхідних даних у 8 біт, але при рості розрядності вхідних даних відсоток покращення зменшується і становить 2% при розрядності вхідних даних 64 біта.

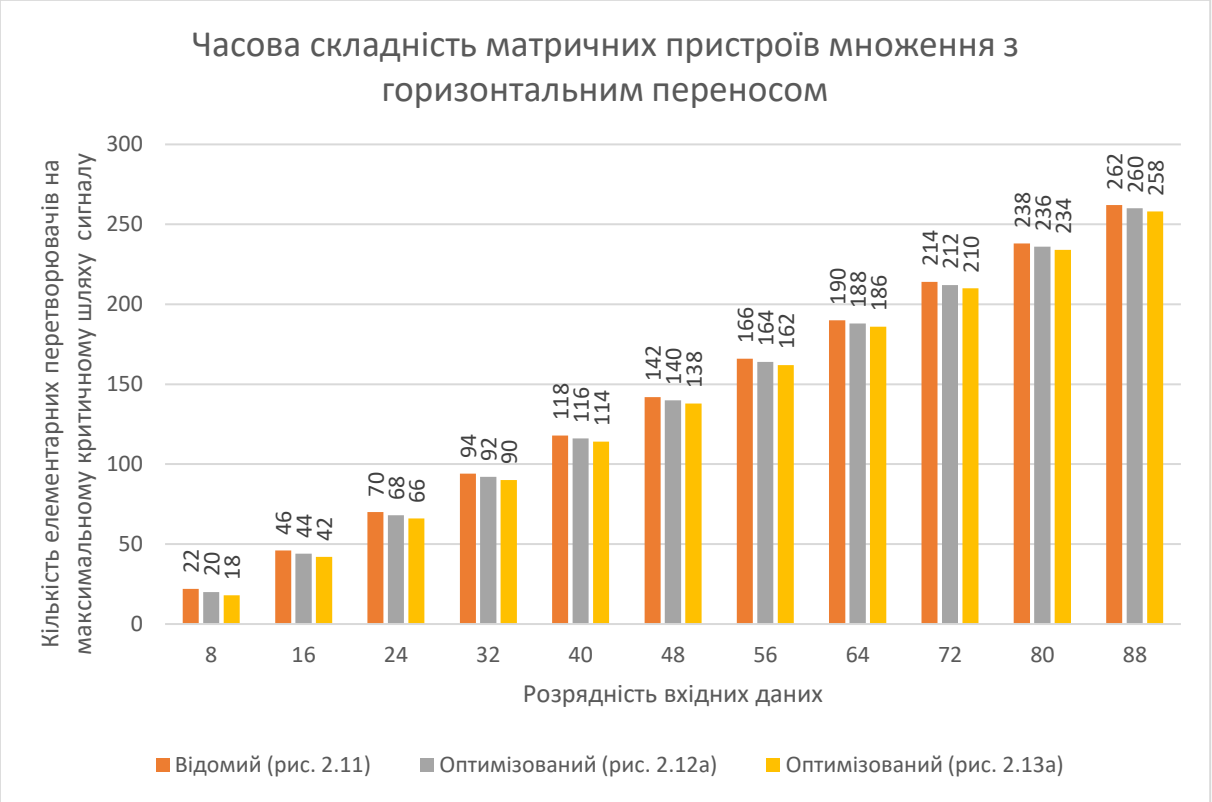


Рисунок 2.17. Порівняльна діаграма значень часової складності H-моделей матричних пристроїв множення з горизонтальним переносом

Нажаль, така параметрична оптимізація H-моделі пристрою множення з горизонтальним розповсюдженням переносу не дала змоги отримати такий пристрій, який б відповідав поставленій у завданні вимозі – затримка на

пристрої має бути меньшою або рівною затримці на багаторозрядному суматорі.

2.4.2. Матричний пристрій множення з діагональним розповсюдженням переносу

На відміну від попереднього, Н-модель пристрою множення з діагональним розповсюдженням переносу складається із двох частин: матриці комірок Г (Гілда), перенос в яких реалізований по діагоналі, та n розрядного суматора, з горизонтальним переносом (рис. 2.18). Помножувач з діагональним переносом має дві різнотипних комірки, на відміну від помножувача із горизонтальним розповсюдженням переносу (рис. 2.11), таким чином значення структурної характеристики складності цих пристроїв відрізняється.

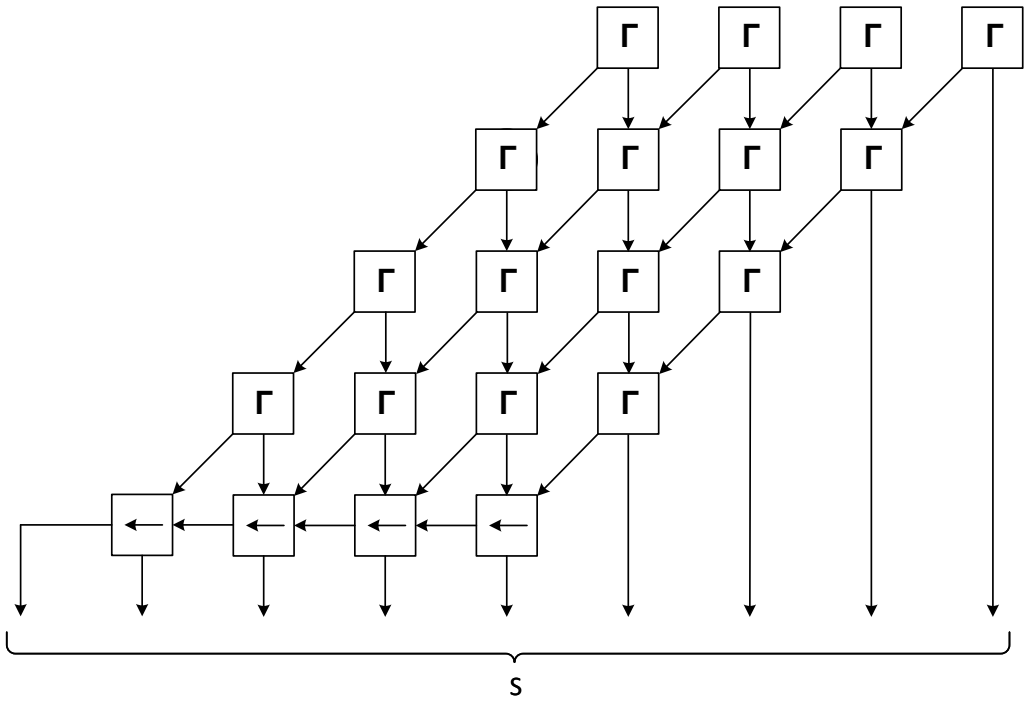
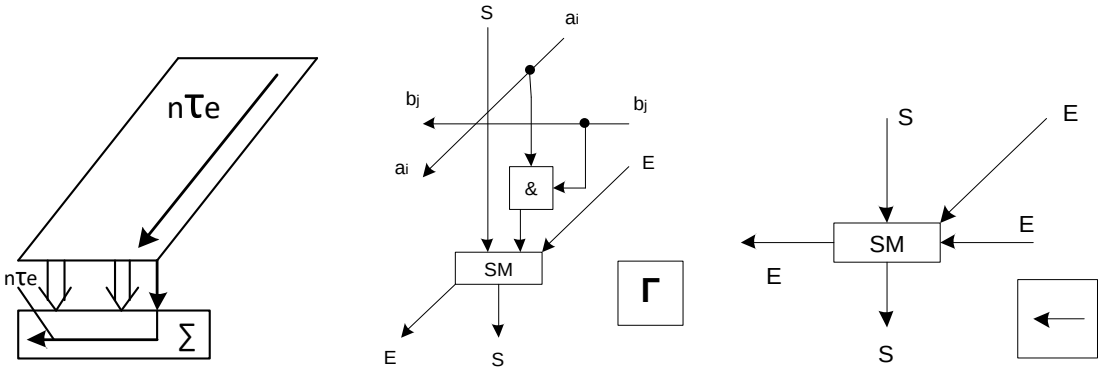


Рисунок 2.18. Н-модель пристрою множення з діагональним розповсюдженням переносу

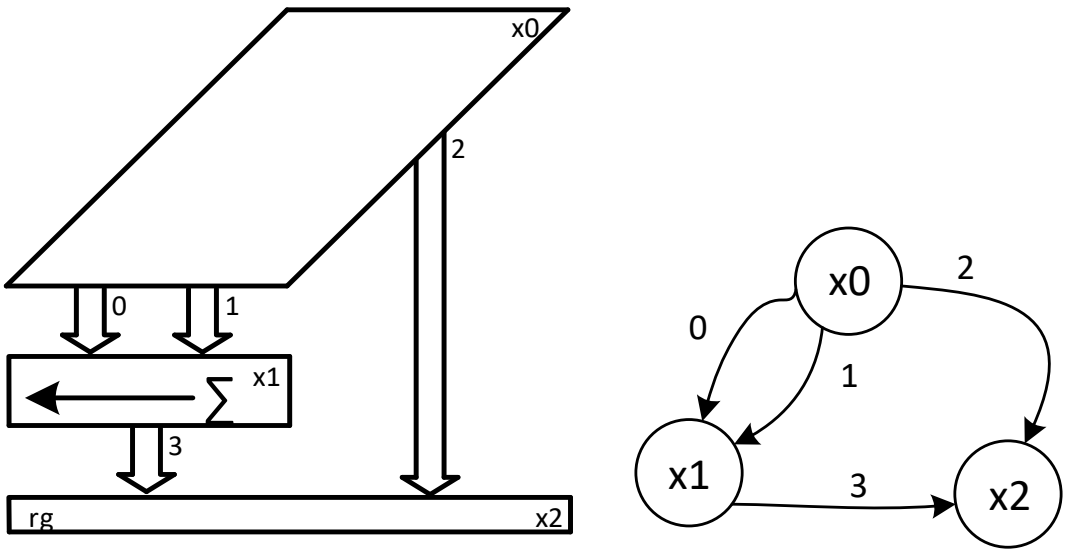
Значення часової складності пристрою рівне кількості комірок Г одної діагоналі, та кількості суматорів останнього рядка матриці , як зображено на рисунку (рис. 2.19 а) L=2n. Апаратна складність $A=n^2+n$ [104].



а) б) в)

Рисунок 2.19. а) максимальний шлях розповсюдження сигналу б,в) Н-моделі комірок пристрою множення з діагональним переносом

Для розрахунків структурної складності будуємо блок-схему Н-моделі пристрою (рис. 2.18), та граф-схему відповідно. На блок-схемі (рис. 2.20 а) блок [x0] відповідає матриці розміром [n^2] комірок [Г], блок [x1] відповідає останньому ряду суматорів з горизонтальним розповсюдженням переносу, блок [x2] позначає регістр для збереження результату множення на пристрої.



а) б)

Рисунок 2.20. а) Блок-схема; б) граф схема пристрою множення з діагональним розповсюдженням переносу

Обчислення структурної складності робимо описаним раніше методом. Використовуючи граф-схему пристрою (рис. 2.20 б) та матрицю інциденцій

(2.7), згідно з формулою (1.16) структурна складність пристрою $S = -8\log_2 \frac{8}{3 \cdot 4} \approx 4.7$ [104].

I=

	0	1	2	3
x0	1	1	1	
x1	-1	-1		1
x2			-1	-1

(2.7)

Значення часової та апаратної характеристик складності можна покращити, для цього вихідну матрицю потрібно оптимізувати. Вхідні комірки першого горизонтального ряду, та лівий вертикальний ряд можна замінити схемами кон'юнкції, такими ж як на рисунку (рис. 2.12 б), як показано на рисунку (рис. 2.21).

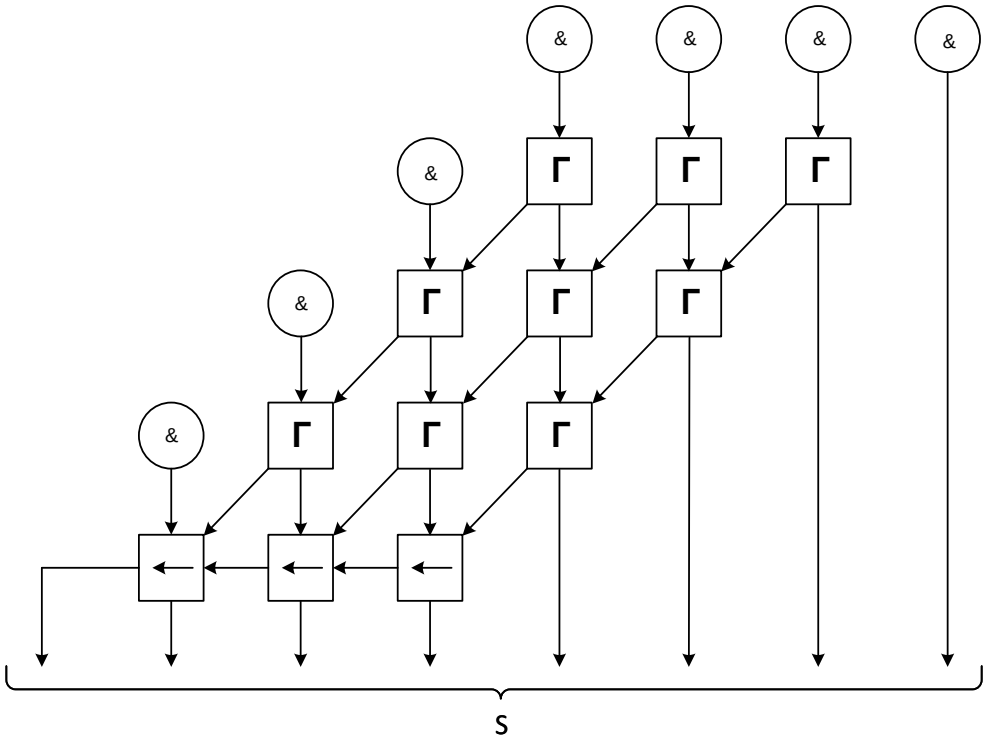


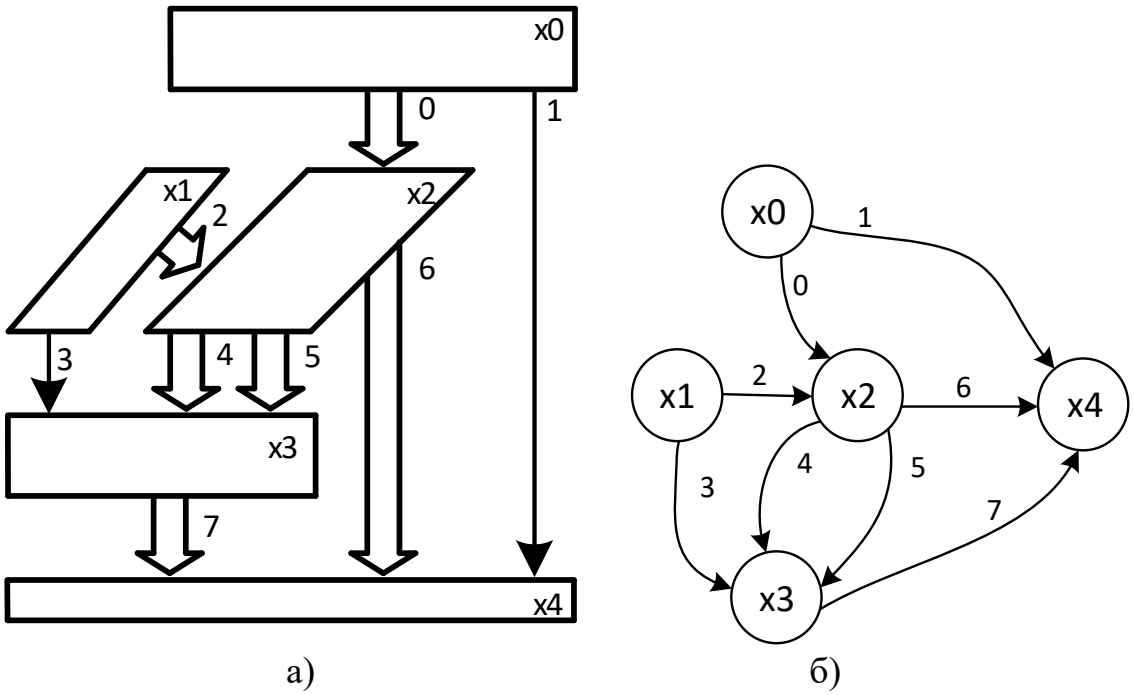
Рисунок 2.21. Оптимізований пристрій множення з діагональним переносом

В результаті параметричної оптимізації було покращено значення характеристик складності, у порівнянні з вхідною схемою (рис 2.18), часова складність (L=2n-2), та апаратна складність ($A=n^2-n$), але значення структурної складності зросло.

Для обчислення структурної складності будуємо блок-схему пристрою множення (рис. 2.21), яка матиме наступний вигляд (рис. 2.22 а).

У блок-схемі (рис. 2.22 а) присутні наступні блоки:

- [x0] – блок, який відображає верхню горизонтальну лінію елементів кон'юнкції, кількість яких рівна кількості розрядів вхідних даних [n];
- [x1] – блок елементів кон'юнкції лівої частини схеми (рис. 2.21), кількість який рівна [n-1];
- [x2] – матриця комірок Гілда [Г] пристрою множення, кількість яких рівна $[(n-1)^2]$
- [x3] – блок елементів (рис. 2.19 в) фінального формування результату множення, кількість яких рівна [n-1];
- [x4] – блок, який позначає регістр, куди передається і зберігається результат роботи пристрою.



а) б)

Рисунок 2.22. а) блок-схема б) граф схема, оптимізованого пристрою множення з діагональним розповсюдженням переносу

Відповідно до матриці інциденцій графу (2.8), та згідно формулі (1.16), структурна складність оптимізованого матричного пристрою множення з діагональним розповсюдженням переносу рівна $S = -16\log_2 \frac{16}{5 \cdot 8} \approx 21.1$.

I=

	0	1	2	3	4	5	6	7
x0	1	1						
x1			1	1				
x2	-1		-1		1	1	1	
x3				-1	-1	-1		1
x4		-1					-1	-1

(2.8)

Проаналізувавши наведені схеми та провівши необхідні обчислення характеристик складності можемо сформувати порівняльну таблицю для розглянутих пристроїв множення з діагональним переносом (табл. 2.2).

Таблиця 2.2

Характеристики складності ПМ з діагональним розповсюдженням переносу

Схема \ Х-ки складності	L	A	S	P
рис. 2.18	2n	n^2+n	4.7	0
рис. 2.21	2n-2	n^2-n	21.1	0

Розглянуті схеми пристрою множення з діагональним розповсюдженням переносу розрізняються за характеристиками складності. Вибір однієї із них залежить від напрямку застосування. Для реалізації у виробах, в яких вимагається висока швидкодія, де пристрій множення використовується як слабо зв’язаний з іншими вузлами системи, може бути рекомендований варіант наведений на схемі (рис.2.21), цей варіант має підвищену швидкодію, але збільшену структурну складність. Варіант пристрою (рис. 2.18) привабливий для спрощених, мало розрядних процесорів з малою структурною складністю.

Для досягнення необхідної продуктивності згідно вимог, реалізовану Н-модель пристрою множення (рис. 2.21) доцільно оптимізувати з точки зору конвеєризації, що продемонстровано у наступному пункті.

2.4.3. Конвеєрний пристрій множення з діагональним розповсюдженням переносу

Оптимізований матричний пристрій множення з діагональним переносом (рис. 2.21) можна покращити, створивши його двосходинковий конвеєрний варіант (рис. 2.23).

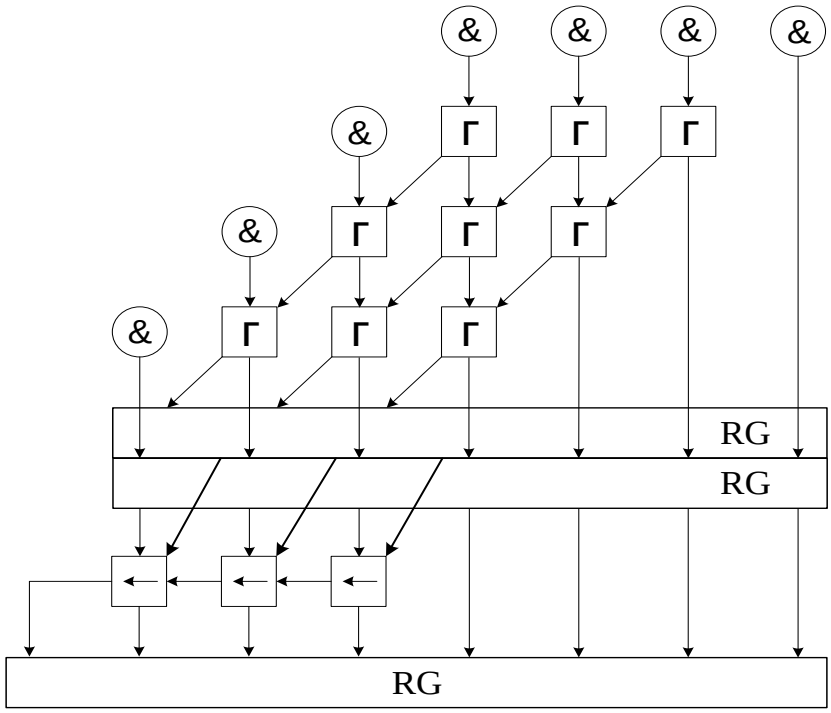
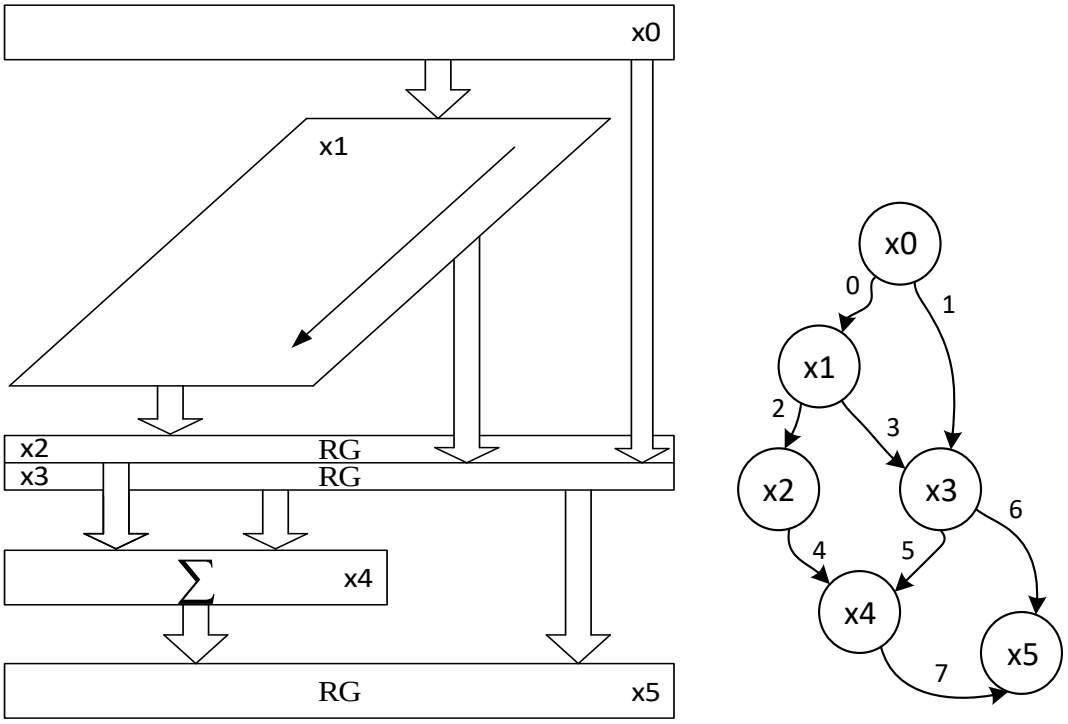


Рисунок 2.23 Н-модель конвеєрного матричного пристрою множення з діагональним розповсюдженням переносу

Розділення схеми проведено двома регістрами по горизонталі між матрицею комірок Гілда та суматором. На схемі зображено саме два регістри через те, що у верхній частині схеми формується два типи сигналів, часткові суми та переноси, тому для кожного типу сигналу доцільно враховувати окремі регістри. Для кожної частини розділеної схеми значення часової характеристики складності однакова, та рівна L=n-1. Апаратна складність такого пристрою множення, у порівнянні з не конвеєрним варіантом зросла, за рахунок доданих конвеєрних регістрів, таким чином $A=n^2+4n-2$ [104].

Для обрахунку структурної складності помножувача, розбиваємо схему на однорідні блоки і формуємо блок-схему (рис.2.24 а).



а) б)

Рисунок 2.24. а) –блок-схема, б) – граф блок-схеми, конвеєрного матричного пристрою множення з діагональним переносом

На блок-схемі (рис. 2.24) до пристрою множення (рис.2.23) присутні наступні блоки:

- [x0] – блок, який відображає верхню горизонтальну лінію елементів кон’юнкції, кількість яких рівна [2n-1] (n – кількість розрядів вхідних даних);
- [x1] – матриця комірок Гілда пристрою множення, кількість яких рівна $[(n-1)^2]$;
- [x2] – конвеєрний регістр, на який передаються переноси від часкових сумувань з першої сходинки. Розрядність такого регістра рівна [n-1];
- [x3] – конвеєрний регістр, на який передаються часткові суми з першої сходинки. Розрядність такого регістра рівна [2n-1];
- [x4] – блок елементів (рис. 2.19 в) фінального формування результату множення, кількість яких рівна [n-1];

- [x5] – блок, який позначає регістр, куди передається і зберігається фінальний результат роботи пристрою.

Отже, використовуючи вираз (1.16) та матрицю інциденцій схеми пристрою (2.9) визначаємо значення структурної складності конвеєрного пристрою множення з діагональним розповсюдженням переносу $S = -16\log_2 \frac{16}{6 \cdot 8} \approx 25.3$. За рахунок збільшення структурної, та апаратної характеристик складності, було досягнуто зменшення часової складності до наступного рівня: затримка сходинки конвеєра пристрою множення (рис. 2.23), буде меншою затримки на одному багато розрядному суматорі. Зауважимо, що часова складність матричного конвеєрного пристрою множення з діагональним переносом не зв'язана з довжиною конвеєрних регістрів, вона рівна L=n-1 [104].

I=

	0	1	2	3	4	5	6	7
x0	1	1						
x1	-1		1	1				
x2			-1		1			
x3		-1		-1		1	1	
x4					-1	-1		1
x5							-1	-1

(2.9)

На діаграмі (рис. 2.25) приведено графічне представлення залежності кількості елементарних перетворювачів (вісь Y) необхідних на реалізацію кожного з розглянутих матричних пристроїв множення з діагональним переносом, в залежності від розрядності вхідних даних (вісь X). Аналізуючи дані з діаграми можемо побачити, що проведена параметрична оптимізація пристрою множення (рис. 2.18) дала змогу отримати H-модель з покращеними значеннями апаратної характеристики складності (рис. 2.21) та H-модель конвеєрного пристрою множення (рис. 2.23). Виходячи з отриманих значень апаратної складності проаналізованих пристроїв отримано наступні результати: оптимізований пристрій множення (рис. 2.21) має покращення значення апаратної складності 22% при розрядності вхідних даних у 8 біт, але

при рості розрядності вхідних даних відсоток покращення зменшується, і при розрядності 64біти покращення становить 3%. Натомість конвеєрний пристрій множення (рис. 2.23) має збільшення значення апаратної складності на 30% при розрядності вхідних даних у 8 біт, це зумовлено тим, що було додано конвеєрні регістри у модель схеми пристрою. Але при рості розрядності вхідних даних відсоток різниці значення апаратної складності зменшується і при вхідних даних 64біти рівний 4,5% - це вказує на те, що такий пристрій множення буде мати значне покращення при реалізації його з великою розрядністю вхідних даних.

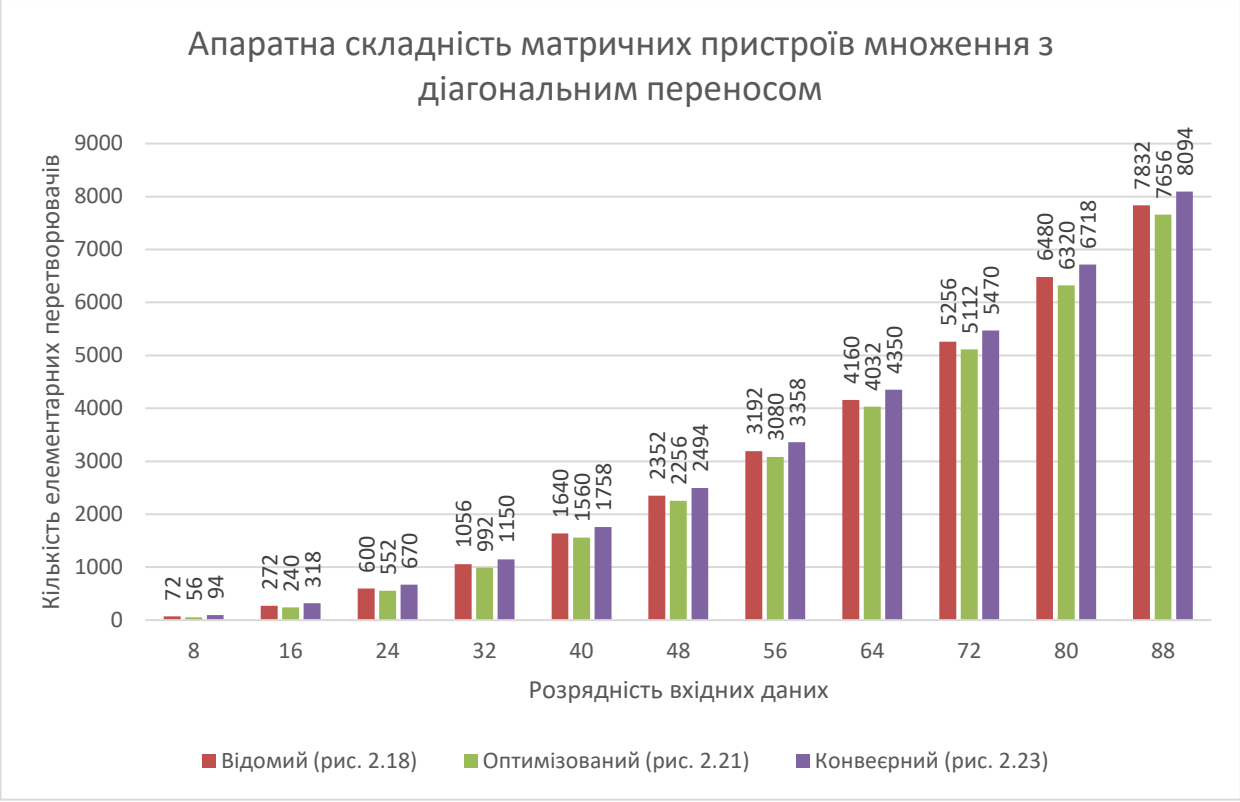


Рисунок 2.25. Порівняльна діаграма значень апаратної складності Н-моделей матричних пристроїв множення з діагональним розповсюдженням переносу

На діаграмі (рис. 2.26) приведено графічне представлення залежності кількості елементарних перетворювачів (вісь Y) які належать максимальному критичному шляху розповсюдження сигналу у кожному з розглянутих матричних пристроїв множення з діагональним переносом, в залежності від розрядності вхідних даних (вісь X). Аналізуючи дані з діаграми можемо побачити, що проведена параметрична оптимізація пристрою множення (рис.

2.18) дала змогу отримати SH-моделі з покращеними значеннями часової характеристики складності (рис. 2.21, рис. 2.23). Виходячи з отриманих значень часової складності проаналізованих пристроїв отримано наступні покращення: оптимізований пристрій множення (рис. 2.21) має покращення значення часової складності на 12% при розрядності вхідних даних у 8 біт, але при рості розрядності вхідних даних до 64 біти відсоток покращення становить 1,5%.

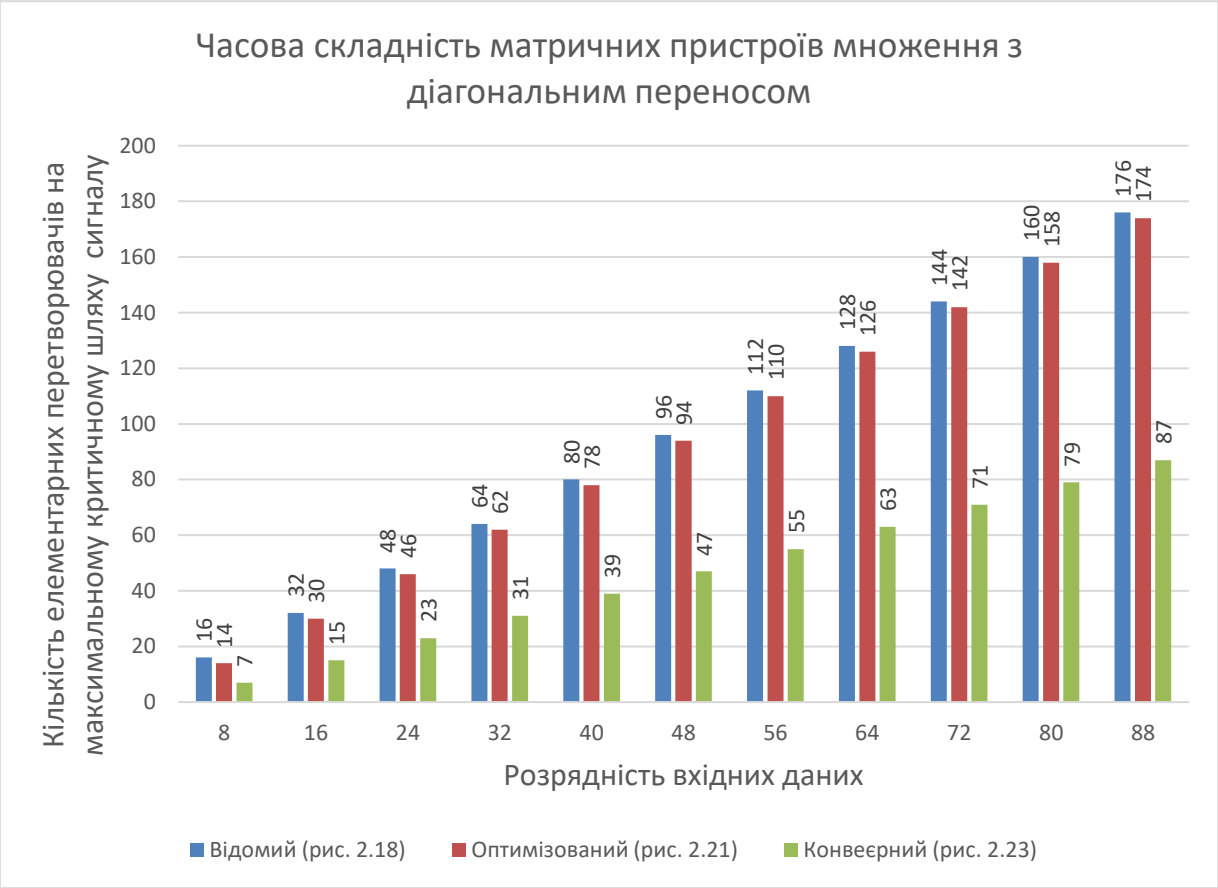


Рисунок 2.26. Порівняльна діаграма значень часової складності H-моделей матричних пристроїв множення з діагональним розповсюдженням переносу

Оптимізований конвеєрний пристрій множення (рис. 2.23) має покращення значення часової складності на 56% при розрядності вхідних даних у 8біт, та при рості розрядності вхідних даних відсоток не значно зменшується, так при розрядності вхідних даних 64біти відсоток покращення трохи більший 50%.

При проведенні аналізу порівняльних діаграм матричних пристроїв множення з діагональним переносом можна зробити висновок, що у процесі параметричної оптимізації матричного пристрою множення з діагональним розповсюдженням переносу було отримано двосходинковий конвеєрний пристрій множення, який задовольняє поставленим у завданні вимогам, а саме затримка сходинки конвеєра пристрою множення є меншою, ніж затримка на одному багаторозрядному суматорі, та має оптимальні значення апаратної та структурної характеристик складності.

2.5. Вибір оптимізованих пристроїв множення для спецпроцесорів

Вибір елементів спецпроцесора (пристрої керування, реалізація арифметичних операцій, обчислення спеціальних функцій) повністю залежить від задачі й поставлених вимог до конкретної спеціалізованої системи у якій даний спецпроцесор буде використовуватися. Те ж саме відноситься до пристроїв множення, вибір варіанту їх реалізації – апаратним чи апаратно-програмним способом, конвеєрна реалізація чи ні, залежить від вимог до системи, для якої цей пристрій множення розробляється.

У нашому випадку, розроблені оптимізовані моделі пристроїв множення (рис. 2.13 а, рис. 2.21, рис. 2.23) відрізняються за значеннями часової, апаратної та структурної характеристик складності. Значення програмної складності усіх оптимізованих пристроїв залишилось рівним 0, так як програмне керування у них обмежується подачею синхросигналів для роботи системи.

Як видно із таблиці 2.3, значення структурної характеристики складності оптимізованих матричних пристроїв множення не значно відрізняються. Можна зробити висновок, що Н-модель пристрою множення з горизонтальним переносом (рис. 2.13 а) є простішою для розуміння розробника, і на її проектування піде найменше затрат часу. Такий пристрій (рис 2.13 а) найкраще підходитме для спеціалізованих комп’ютерних систем, вимогою для

яких є найменші витрати на апаратуру, але не стоїть вимога мінімальної часової складності.

Таблиця 2.3

Характеристики складності Н-моделей оптимізованих матричних ПМ

схема	L	A	S	P
рис. 2.13 а	3n-6	$(n-1)^2$	14	0
рис. 2.21	2n-2	n^2-n	21.1	0
Рис. 2.23	n-1	n^2+4n-2	25.3	0

З діаграми (рис. 2.27) можна побачити, що значення апаратної складності оптимізованих матричних пристроїв множення зростають у перемножувачі з діагональним переносом, та двосходинковому конвеєрному перемножувачі.

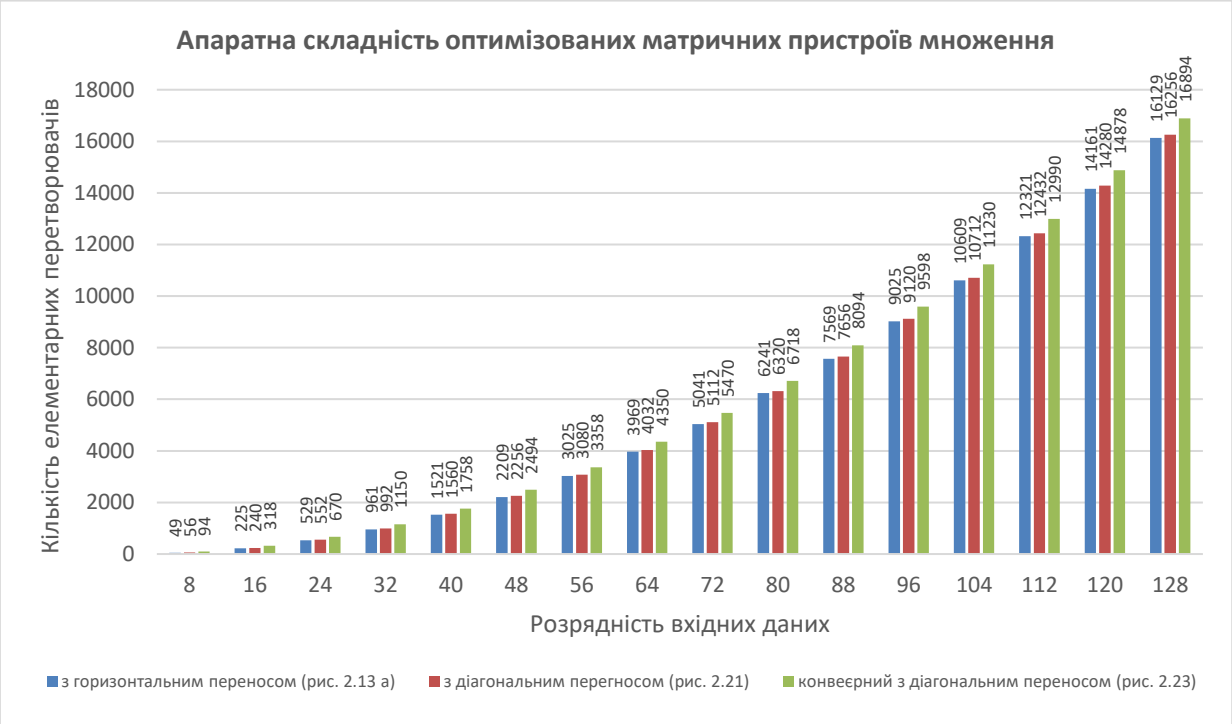


Рисунок 2.27. Порівняльна діаграма значень апаратної складності Н-моделей оптимізованих матричних пристроїв множення

Пристрій множення з діагональним розповсюдженням переносу (рис. 2.21) має значення апаратної складності трохи більше, у порівнянні із оптимізованим пристроєм множення з горизонтальним переносом, проте значення часової характеристики складності у нього є меншим (рис. 2.27), що

є досить важливим показником для спецпроцесорів, де вимагається покращене значення затримки на виконання операції множення.

Також, аналізуючи діаграми порівняння апаратної та часової характеристик складності оптимізованих матричних пристроїв множення (рис. 2.27, рис. 2.28), та таблицю 2.3, можна зробити висновок, що двосходинковий конвеєрний пристрій множення із діагональним розповсюдженням переносу (рис. 2.23) має покращені значення часової складності, що було досягнуто за рахунок збільшення апаратної та структурної характеристик складності. Такий пристрій множення (рис. 2.23) найкраще підходить для спецпроцесорів, в яких основна вимога – це швидкодія роботи системи.

Рисунок 2.28. Порівняльна діаграма значень часової складності Н-моделей оптимізованих матричних пристроїв множення

Таким чином, у ході аналізу пристроїв множення, було проведено параметричну оптимізацію характеристик складності Н-моделей цих

пристроїв, та отримано двосходинковий конвеєрний матричний пристрій множення, затримка на сходинці конвеєра якого є меншою, ніж затримка на n-розрядному суматорі, при одинаковій їх розрядності.

2.6. Характеристики складності керуючих вузлів спецпроцесора

Зазвичай спецпроцесор складається з двох частин: функціональної та пристрою керування. Функціональна частина призначена для реалізації обчислювального процесу перетворення, передачі і збереження даних та адрес. Вона власними апаратними засобами реалізує обчислювальний процес за допомогою сигналів управління. Пристрій керування дозволяє організувати обчислювальний процес, який відбувається в функціональній частині, цей пристрій генерує мікропрограми для виконання асемблерних команд, а також генерує сигнали управління для налагодження взаємодії процесора з іншими вузлами комп'ютера. Обидві частини об'єднують два потоки сигналів: від пристрою керування потік мікрокоманд, а від функціональної частини потік асемблерних команд і сигналів про особливі ситуації, що виникають при виконанні обчислювального процесу [23].

Використовують два способи побудови пристрою керування: мікропрограмний (гнучка логіка) і апаратний (жорстка логіка). На рисунку 2.29 разом з прикладом часової діаграми показані спрощені варіанти двох схем пристроїв керування: на рисунку 2.29 б мікропрограмного управління, на рисунку 2.29 в – апаратного. На рисунку 2.29 а наведений приклад мікропрограми у вигляді часової діаграми [23].

Основою для побудови блоку генерування мікропрограм є часова діаграма. У пристрої з мікропрограмним управлінням, сигнали управління часової діаграми записуються в постійний запам'ятовуючий пристрій (ПЗП). У пристрої з жорсткою логікою часова діаграма перетворюється на сукупність логічних рівнянь, які реалізуються вентильними схемами.

На однакових операційних пристроях за всіма характеристиками складності ці схеми (рис.2.29 б і в) майже рівноцінні. Істотна розбіжність у

схемах існує на більш низькому ієрархічному рівні, і спостерігається лише в побудові блоку генерування мікропрограм.

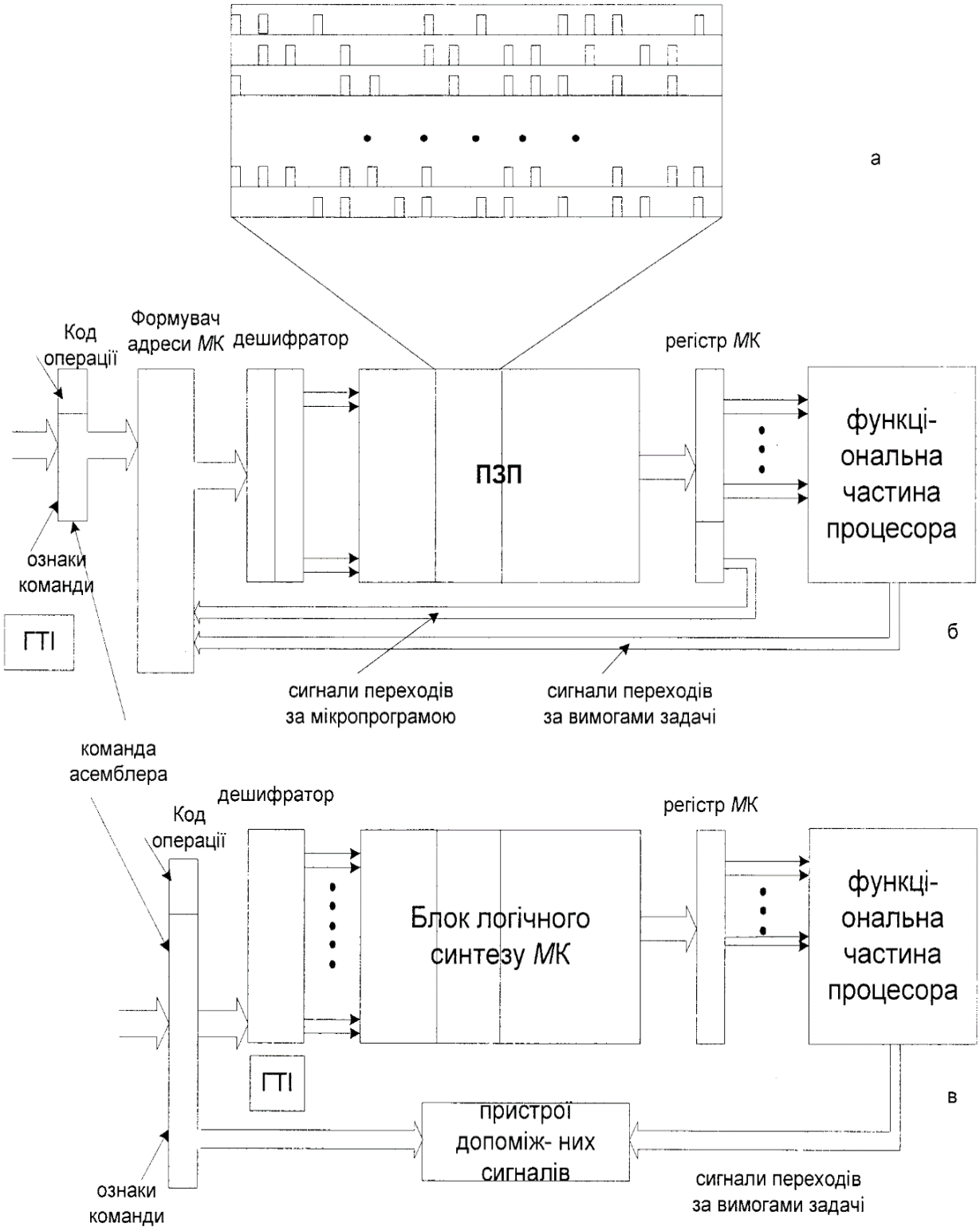


Рисунок 2.29. Схеми пристрою керування

Проведемо якісну оцінку характеристик складності наведених схем.

Часова складність схеми з мікропрограмним управлінням L1 більше ніж схеми з апаратним управлінням L2. Операції в першій схемі: формування адреси мікрокоманди, його пошук, читання, складають три кроки; в другій схемі мікрокоманда генерується за один крок.

Апаратна складність відрізняється складністю організації ПЗП першої схеми, та складністю блоку логічного синтезу мікропрограми. У першому наближенні апаратні складності обох схем рівні [23].

Програмна складність обох схем дорівнює нулю. Крім генератора тактових імпульсів (ГТІ), пристрій управління не має внутрішнього пристрою управління.

Всі необхідні сигнали – початку процесу виконання команди, його закінчення, зміни адреси мікрокоманди в ПЗП, або зміни режиму генерації мікрокоманд в блоці жорсткої логіки – задають генератор тактових імпульсів, коди поля ознак з асемблерної команди, сигнали від схеми формування переходів, яка знаходиться у функціональній частини процесора.

Побудова ПЗП в загальному випадку регулярна, тому значення його структурної складності (S1) наближається до нуля. Інша ситуація в блоці жорсткої логіки. У зв’язку з тим, що структуру апаратного управління задає мікропрограма, величина структурної складності залежить від кількості команд асемблера, від ступеня нерегулярності часової діаграми. Таким чином значення структурної складності такої схеми виступає у функції двох змінних:

$$S_2 = f(w, P_0), \qquad (2.10)$$

де w - кількість команд асемблера;

P_0 - програмна складність часової діаграми функціональної частини.

Вираз для розрахунку значення структурної складності такої схеми наступний:

$$S_2 = -Q \log_2 \frac{Q}{YT}, \qquad (2.11)$$

де Q - кількість термів логічних рівнянь в блоці жорсткої логіки;

Y - кількість входів управління функціональної частини;

T - кількість дискретів часу часової діаграми.

Ємнісна складність мікропрограмного управління М1 дорівнює обсягу ПЗП, а ємнісна складність пристрою керування з жорсткою логікою дорівнює нулю.

Підсумуємо отримані оцінки характеристик складності двох схем. Програмна складність у них однакова, і вона дорівнює нулю. Зауважимо, що одиниця ємнісної складності, в даному випадку складності ПЗП, дорівнює одиниці апаратної складності вентильної схеми. З урахуванням цих зауважень остаточна оцінка наступна:

$$\begin{aligned} &M_1 = kA_2 \\ &S_1 = 0; S_2 = f(w, P_0); \\ &L_1 > L_2 \end{aligned} \qquad (2.12)$$

Таким чином, перевагою способу мікропрограмного управління є нульова структурна складність. Велика структурна складність є єдиним, але істотним недоліком апаратного управління.

Перевагою способу управління з жорсткою логікою є менші значення часової та апаратної складності. Ці переваги дозволяють підвищити продуктивність процесора і зменшити площу кристалу, яку займає пристрій керування. Останній фактор впливає на можливість удосконалення вузлів функціональної частини процесора. Апаратні засоби пристрою керування не беруть безпосередньої участі у вирішенні задач процесора. Якщо збільшення апаратної складності функціональних пристроїв є потужним способом зменшення часової та програмної складності, то збільшення апаратної складності пристрою управління має лише негативні наслідки. Головний недолік обумовлений тим, що при постійній загальній площі кристала збільшення апаратної складності пристрою управління зменшує площу під розміщення функціональних вузлів. Крім цього, кожна громіздка система уповільнює процеси, які відбуваються в ній.

Висновки до розділу

1. Вдосконалено відомий метод обчислення структурної складності SH-моделей за рахунок виявлення та об’єднання у групи однотипних елементів схеми, що дало змогу отримувати матриці інциденцій значно меншого розміру без регулярно розташованих елементів а в результаті спростило обчислення й скоротило час на проектування системи.

2. Отримано характеристики складності SH-моделі пристрою множення на багаторозрядному суматорі й показано, що його не доцільно застосовувати для реалізації спеціальних функцій, оскільки він має високу часову, програмну та структурну складність.

3. Отримано характеристики складності SH-моделі конвеєрного пристрою множення й показано, що він не підходить для реалізації спеціальних функцій, оскільки у ньому кількість сходинок конвеєра залежить від розрядності вхідних даних.

4. Проведено оптимізацію матричного пристрою множення з горизонтальним розповсюдженням переносу й отримано H-модель пристрою їз заміною першого ряду комірок матриці схемами кон’юнкції, яка має у порівнянні з відомим пристроєм меншу апаратну складність на 12% та часову складність на 9% при 8-и розрядних вхідних даних, та відповідно 1,5% та 1% при 64-х розрядних даних.

5. Проведено оптимізацію матричного пристрою множення з горизонтальним розповсюдженням переносу й отримано H-модель пристрою із заміною першого ряду комірок матриці схемами кон’юнкції та правого діагонального ряду комірок схемами напівсуматорів, яка має у порівнянні з відомим пристроєм меншу апаратну складність на 23% та часову складність на 9% при 8-и розрядних вхідних даних, та відповідно 3,1% та 1% при 64-х розрядних даних.

6. Проведено оптимізацію матричного пристрою множення з діагональним розповсюдженням переносу й отримано H-модель пристрою із заміною першого горизонтального ряду та лівого вертикального ряду комірок

матриці схемами кон’юнкції, яка має у порівнянні з відомим пристроєм меншу апаратну складність на 22% та часову складність на 12% при 8-и розрядних вхідних даних, та відповідно 3% та 1,5% при 64-х розрядних даних.

7. Проведено оптимізацію матричного пристрою множення з діагональним розповсюдженням переносу й отримано H-модель двосходнинкового конвеєрного пристрою множення, яка має у порівнянні з оптимізованим пристроєм більшу апаратну складність на 30% при 8-и розрядних вхідних даних, та відповідно 4,5% при 64-х розрядних даних, що зумовлено додаванням у схему конвеєрних регістрів. Проте значення часової складності конвеєрного пристрою множення, у порівнянні з оптимізованим, є меншим на 56% при 8-ми розрядних вхідних даних, та відповідно на 50% при 64-х розрядних даних.

8. Отримані характеристики складності SH-моделей двох типів схем керування спецпроцесорів й показано, що для підвищення ефективності обчислення спеціальних функцій необхідно реалізувати функціональні блоки пристрою таким чином, щоб програмна складність була мінімальною.

РОЗДІЛ 3
ОПТИМІЗАЦІЯ SH-МОДЕЛЕЙ ФУНКЦІЙ ЗГОРТКИ ТА ШВИДКОГО ПЕРЕТВОРЕННЯ ФУР'Є

Для реалізації спеціальних функцій на універсальних комп'ютерних системах не враховується повною мірою такий потужний спосіб, як апаратна реалізація алгоритму. Універсальні комп'ютери використовують асемблер, який не містить команд елементарних та спеціальних функцій.

Натомість, набір команд SH-, та H-моделей спецпроцесора, не прив'язані до асемблера. Тут перетворення даних за алгоритмом проводиться безпосередньо апаратними засобами. Алгоритм задається кодом функцій. Кількість коду мікропрограми обмежена конкретним напрямком спеціалізації комп'ютерної системи. В спецпроцесорах напрямок спеціалізації охоплює невелику кількість функцій та операцій.

Спеціалізація потрібна там, де її використання залишається поза конкуренцією з універсальними комп'ютерними системами. Прикладами таких систем можуть бути: бортові системи, мікроконтролери, потужні алгоритмічні процесори, спецпроцесори. Але на відміну від універсальних КС, спеціалізовані є обмежені в застосуванні.

У спеціалізованих комп'ютерних системах апаратні засоби проектуються для заданої алгоритмічної бази. В таких системах апаратні засоби організовані в синхронний конвеєр даних. Зауважимо, що в універсальних комп'ютерних системах, використовується головним чином конвеєр команд. Команди асемблера, в універсальних процесорах, мають різний час виконання, тому організувати для них ефективний конвеєр даних не завжди можливо. Для спеціалізованих комп'ютерних систем такої проблеми практично не існує.

3.1. Структурний синтез пристрою згортки

Пристрій згортки, що реалізує цифрову фільтрацію, є обов'язковим елементом в системах цифрової обробки сигналів (ЦОС). В процесі

проектування спеціалізованих комп'ютерних систем цього напрямку основну увагу приділяють отриманню високої продуктивності. Наше завдання пов'язується з параметричною оптимізацією характеристик складності SH-моделі згортки. Мета – отримання високої продуктивності обробки сигналів разом з оптимізацією витрат на проектування пристрою.

Основними архітектурними способами отримання високої продуктивності оброблення даних на обмеженому списку заданих алгоритмів є:

- апаратне виконання функціональних залежностей;
- конвеєризація процесу оброблення;
- використання паралелізму процесів на всіх ієрархічних рівнях системи.

Для розробки ефективного конвеєрного пристрою визначимо наступні умови:

1. Часова складність сходинки конвеєра не повинна перевищувати часову складність спрацювання одного багато розрядного суматора;
2. Програмна складність повинна наближатися до нуля;
3. Кількість ступенів конвеєра не повинна залежати від розрядності чисел;
4. Пристрій повинен допускати ефективне суміщення з іншими алгоритмічними пристроями реконфігурованої системи, що синтезується.

Структурний синтез здійснюється за формулою згортки:

$$y_к = \sum_{i=0}^{N-1} x_{k+i} b_i;\ \ i,k = \overline{0 \div N-1} \qquad (3.1)$$

де $i,k = 0 – (N-1), \ldots$,

$y_к$ – к-й відлік результату згортки,

$x_{к-i}$ – (к-і) відлік реалізації випадкового процесу,

b_i – і-й відлік вагової функції фільтру,

Індекс і замінений для зручності, з від'ємного на додатній. Така заміна допустима, якщо змінити напрямок читання масиву {x}.

В розгорнутому вигляді алгоритм згортки (для 4-ох точок) має вигляд:

$$y_0 = x_0b_0 + x_1b_1 + x_2b_2 + x_3b_3$$
$$y_1 = x_1b_0 + x_2b_1 + x_3b_2 + x_4b_3$$
$$y_2 = x_2b_0 + x_3b_1 + x_4b_2 + x_5b_3 \quad (3.2)$$
$$y_3 = x_3b_0 + x_4b_1 + x_5b_2 + x_6b_3$$

3.1.1. Систолічний пристрій згортки

Структура пристрою, схема АЛП комірки, та часова діаграма станів продемонстровані на рисунку 3.1. Процес цифрової фільтрації починається з послідовного завантаження конвеєрного ланцюга комірок систолічного пристрою вибірками масиву *{x}*. Часова діаграма завантаження вибірок *{x}* показана під структурою пристрою (рис. 3.1). Часова складність цієї мікрооперації дорівнює: $l_1=n$, де n – кількість комірок АЛП структури пристрою. Кожна комірка в процесі обчислення виконує подвійну операцію множення-сумування. Комірка АЛП складається з: помножувача, суматора та трьох конвеєрних регістрів. Часова складність фільтрації $l_2=N$, де N – розмір входу алгоритму згортки [103].

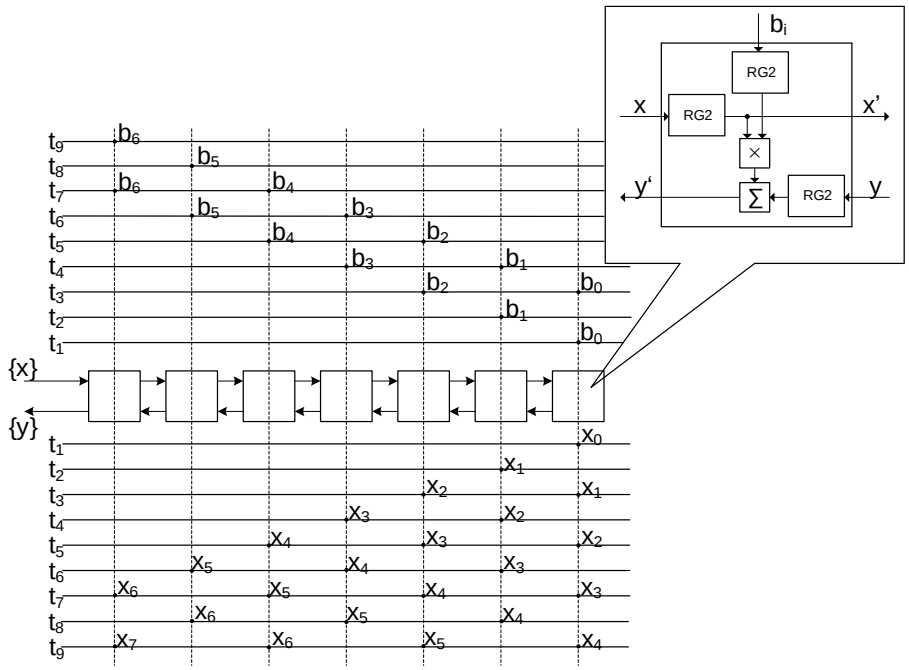


Рисунок 3.1. Н-модель реалізації алгоритму згортки на систолічній структурі

Часова діаграма розташована над структурою пристрою відображає мікрокоманди подачі вибірок масиву {b} на комірки АЛП. На кожному такті роботи конвеєра значення вибірок не змінюється [103].

Загальна часова складність пристрою рівна: L = n + N.

Апаратна складність: A = n.

Програмна складність: P = 0, пристрій не потребує мікропрограмного керування.

Структурна складність: S = 0. Пристрій не містить вузлів з різною внутрішньою структурою, внутрішня структура зв’язків однорідна.

Переваги такого пристрою: висока продуктивність виконання обчислень та висока продуктивність проектування. Якщо іде мова про проектування за допомогою мов опису апаратних засобів, то достатньо створити опис однієї такої комірки систолічної структури і правильно організувати з’єднання між комірками.

Недоліки: жорстка спеціалізація, реконфігурація практично неможлива. В такому пристрої кількість сходинок конвеєра напряму залежить від розрядності вхідних даних. Так як на такій структурі неможливе суміщення виконання декількох алгоритмів, систолічний пристрій згортки стає недоцільним у використанні його в спец процесорах.

3.1.2. Пристрій згортки з неоднорідною структурою

Для вирішення проблеми систолічних матриць, потрібно шукати рішення таких варіантів розв’язання, які піддаються реконфігурації та суміщенню операцій. Тому для розв’язання вищерозглянутого алгоритму згортки можна використати наведену структуру (рис. 3.2).

Процес починається з завантаження 4-х конвеєрних регістрів Рг вибірками масиву {x}. Загальна часова складність пристрою дорівнює L=n*N+1, де n – кількість операційних пристроїв одного блоку. Блок складається з пристрою множення, суматора та двох регістрів [103].

N = {x};

Апаратна складність: A = 4n + 2;

Програмна складність: P = 0.

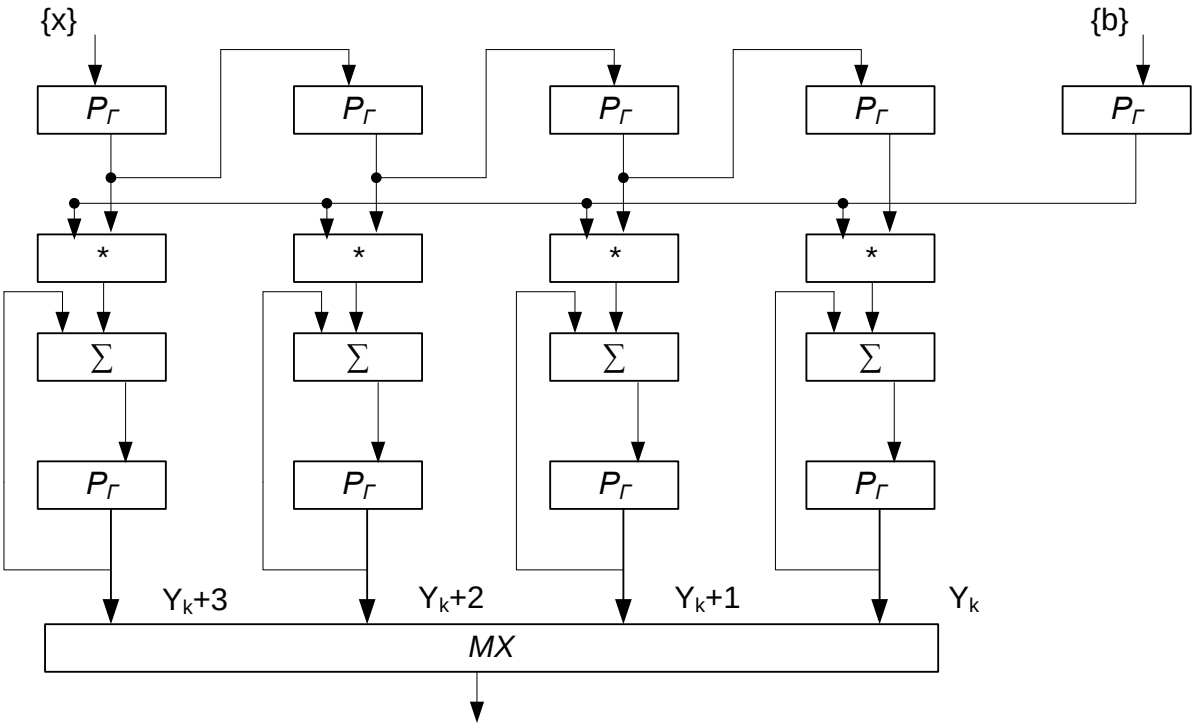
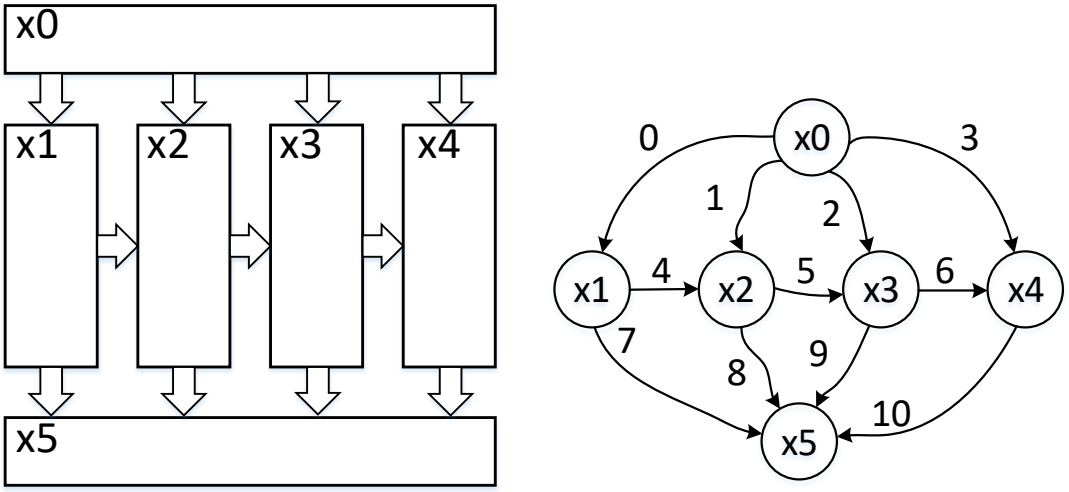


Рисунок 3.2. Реалізація SH-моделі алгоритму згортки по 4-х точках

Структурна складність такого пристрою не рівна нулю, як в попередньому варіанті на систолічній структурі. Тут присутня неоднорідність розміщення елементів схеми (рис. 3.3), що збільшує значення структурної складності.



а) б)

Рисунок 3.3. а –блок-схема, б – граф блок-схеми, пристрою згортки по 4-х точках

Відповідно до граф-схеми пристрою згортки (рис 3.3 б) зформовано матрицю інциденцій (3.3).

I=

	0	1	2	3	4	5	6	7	8	9	10
x0	1	1	1	1							
x1	-1				1			1			
x2		-1			-1	1			1		
x3			-1			-1	1			1	
x4				-1			-1				1
x5								-1	-1	-1	-1

(3.3)

Значення структурної складності, згідно з формулою (1.16), $S = -22\log_2 \frac{22}{6 \cdot 11} \approx 34.9$.

Переваги та недоліки такого пристрою: За апаратною складністю така реалізація є в два рази кращою у порівнянні з систолічною матрицею. Часова складність цієї схеми також краща в ~8 разів. Звичайно структурна складність такої схеми більша, але вагомим і вирішальним плюсом є продуктивність – вона вища в ~8 разів (табл. 3.1).

Таблиця 3.1.

Порівняння характеристик складності пристроїв згортки

	L	**A**	**S**	**P**
Систолічна структура	n+N	n	0	0
Структура по 4-х точках	n*N+1	4n+2	34.9	0

Схема пристрою згортки на чотирьох пристроях множення дає змогу зменшити значення часової складності. Можливий варіант такої схеми, побудований на одному помножувачі, пристрій реалізації одного тракту алгоритму згортки зображений на рисунку (рис. 3.4). Апаратна складність такого пристрою буде значно меншою, ніж у пристрою з 4 трактами, структурна складність тут також є меншою. Для досягнення затримки конвеєра до рівня одного багато розрядного суматора дану структуру було

оптимізовано, розділивши кожен крок виконання алгоритму конвеєрними регістрами.

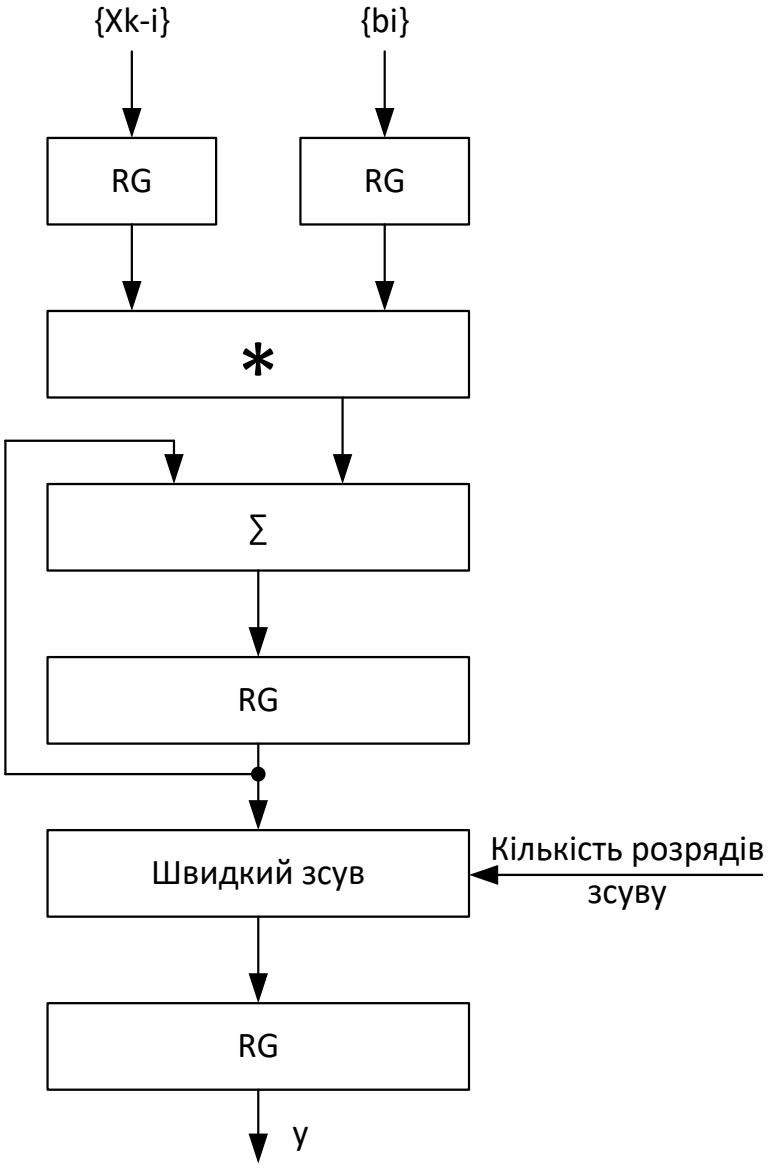


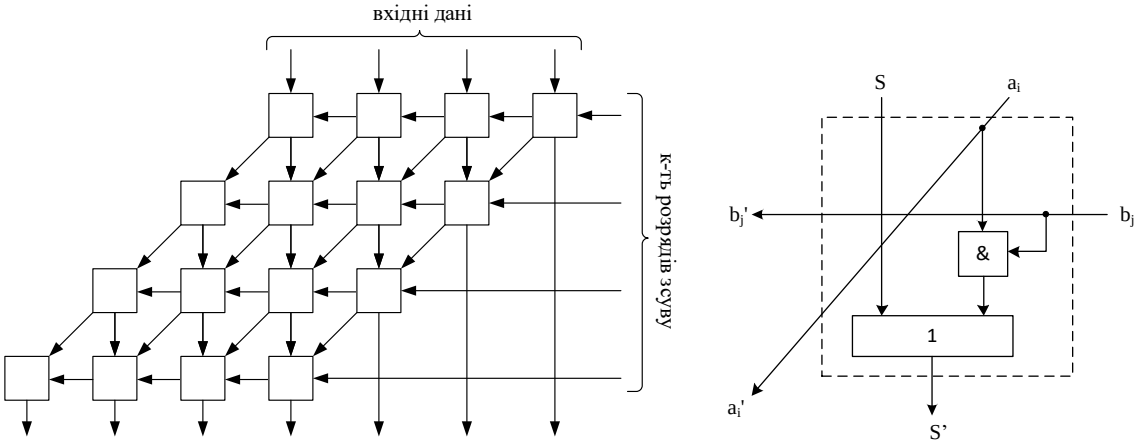
Рисунок 3.4. SH-модель одного тракту пристрою, що реалізує алгоритм згортки.

Для досягнення відповідності часової складність кожної сходинки конвеєра до часової складності одного багато розрядного суматора необхідна оптимізація схеми. Головним об’єктом дослідження можливості оптимізації тут є пристрій множення. Відомо багато варіантів схеми множення, які розрізняються за характеристиками складності, у тому числі часовою. Всі ці пристрої були розглянуті раніше.

Матричні пристрої множення в найбільшій мірі задовольняють нашим умовам. Найкращим варіантом буде використання конвеєрного матричного пристрою множення з діагональним розповсюдженням переносу (рис. 2.23).

Схема зсуву

Кінцевим блоком пристрою згортки є схема швидкого зсуву. Вона виконує роль запобіжника від перевантаження розрядної сітки. Схема дозволяє зсувати результат накопичення в бік молодших або старших розрядів. Діапазон зсування залежить від виконуваних задач та дорівнює один або декілька розрядів операндів. Один з варіантів реалізації схеми – матриця множення (рис. 3.5). В ній множене це число, яке потрібно зсунути, множник – число із нульовими розрядами крім одного. Розряд, що дорівнює одиниці, задає величину зсуву відносно заданого.



а) б)

Рисунок 3.5. а) – фрагмент матриці схеми зсуву, б) – внутрішня схема комірки.

Комірка матриці (рис. 3.5 б) реалізована на двох зв’язаних логічних вентилях: і та або. Часова складність одної комірки дорівнює часовій складності одного логічного елемента або ($\ell_{або}$). Вона менше часової складності одного елемента багато розрядного суматора: $\ell_{або} \approx \ell_{\Sigma}$. Фрагмент матриці показаний на рисунку (рис. 4.5 а). Не важко показати, що при діапазоні зсуву $\pm n/2$ розрядів, характеристики складності мають такі значення: $L_З = n/2$; $A = n^2$; $P = 0$; $S = 0$. Схема зсуву задовольняє вимозі за часовою складністю, так як $L_{1/2М} \approx n-1$ а $L_{\Sigma} = n$ тому $L_З < L_{\Sigma}$.

3.2. Синтез та оптимізація пристрою реалізації алгоритму швидкого перетворення Фур'є

Розглянемо приклад проведення синтезу пристрою виконання алгоритму Швидкого Перетворення Фур'є (ШПФ). Для структурного синтезу Н-моделі, початковою точкою є алгоритм. Розглянемо алгоритм ШПФ, одним із варіантів якого, є розбиття масиву вхідних даних на під масиви. Обчислення для парних і непарних відліків має наступний вигляд [102]:

$$X(k) = X_0(k) + W_N^k X_1(k) \tag{3.4}$$

$$X\left(k + \frac{N}{2}\right) = X_0(k) + W_N^k X_1(k) \tag{3.5}$$

де $W_N = e^{-j\frac{2\pi}{N}}$.

Графічне уявлення про алгоритм дає метелик ШПФ (рис. 3.6).

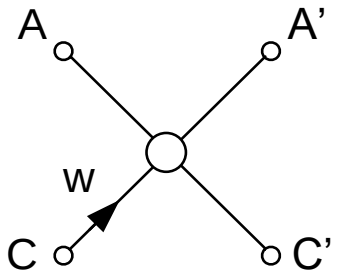


Рисунок 3.6. «Метелик» алгоритму ШПФ

Введемо наступні позначення змінних формули ШПФ. Отже виходячи з (3.4) і (3.5) метелик ШПФ виконує наступні обчислення:

$$\left.\begin{aligned} A' &= A + WC, \\ C' &= A - WC. \end{aligned}\right\} \tag{3.6}$$

Відповідно, у комплексній формі [102]:

$A = a + jb;\ C = c + jd;\ W = w + jv.$

Отже: $A' = (a + cw - dv) + j(b + vc + dw);\ C' = (a - cw + dv) + j(b - (vc + dw));$

Проведемо структурний синтез пристрою ШПФ за наведеними формулами. Пиведемо операнди, які потрібно обчислити для визначення метелика: $a' = a+cw-dv;\ \ b' = b+vc+dw;\ \ c' = a-(cw-dv);\ \ d' = b-(vc+dw)$.

Досвід свідчить, що найкраще співвідношення значень характеристик складності досягається використанням для структурного синтезу канонічної сукупності рівнянь алгоритму.

Обчислення метелика ШПФ передбачає виконання двох основних операцій – множення та сумування. Їм у відповідність ставляться два пристрої, пристрій множення та пристрій сумування. Кожен вхідний операнд використовується при обчисленнях двічі. На сьогодні створено багато схемо-технічних рішень реалізації даного алгоритму.

В процесі структурного синтезу будь якої обчислювальної системи, ми повинні на початку передбачити способи підвищення продуктивності системи, яку будемо проектувати.

Далі структурний синтез передбачає реалізацію комп'ютерної схеми. Схема пристрою ШПФ із чотирма пристроями множення (рис. 3.7), яка за своєю структурою відповідає канонічній формі алгоритму ШПФ. Як зображено на схемі, на суматорах потрібно виконувати операції як додавання так і віднімання, це реалізовується за допомогою сигналу керування суматором. Після реалізації схеми потрібно провести аналіз характеристик складності, які повинні задовольняти проект.

Аналізуючи схему можна відмітити, що організація завантаження вхідних даних в регістри не спричиняє ускладнення програмної складності. Обчислення метелика ШПФ на даному пристрої виконується за одне паралельне множення [102].

Може бути використано декілька варіантів вибору характеристик складності проекту, наприклад: задачею проекту може бути досягнення високої продуктивності, в такому випадку всі характеристики складності повинні бути підпорядковані цій задачі. Апаратна складність не є критичною, тому може бути її збільшення для розв'язку такої задачі.

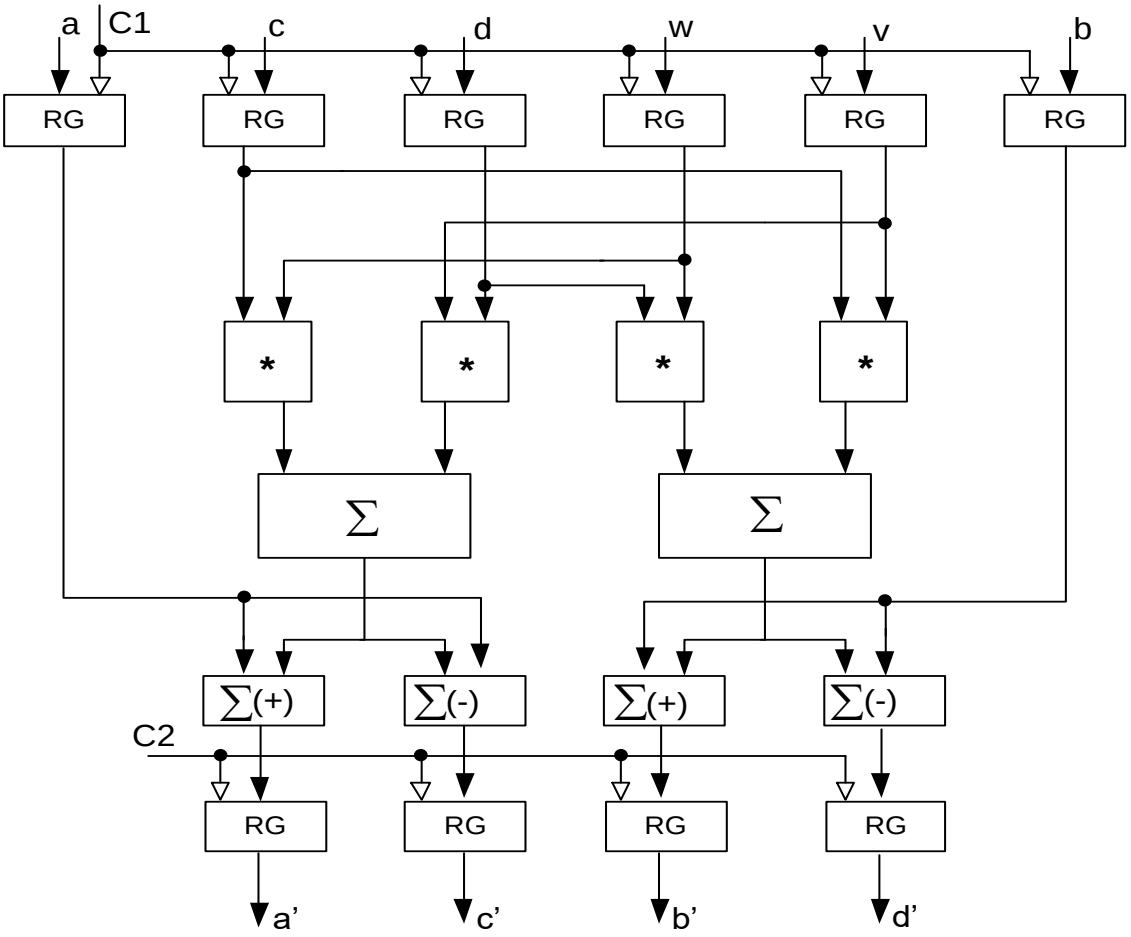


Рисунок 3.7. SH-модель пристрою, яка реалізує метелик ШПФ

Іншим варіантом поставленого завдання може бути розроблення пристрою ШПФ з мінімізованою апаратною складністю, в такому випадку допускається зростання часової складності кінцевого пристрою, завдяки чому зменшується продуктивність [105].

Отже проведемо поглиблений аналіз характеристик складності наведеної схеми (рис. 3.7) і визначимо чи можлива якась її оптимізація. На рисунку (рис.3.8) зображена часова діаграма розглянутого пристрою ШПФ.

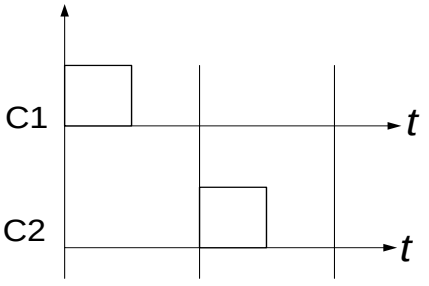


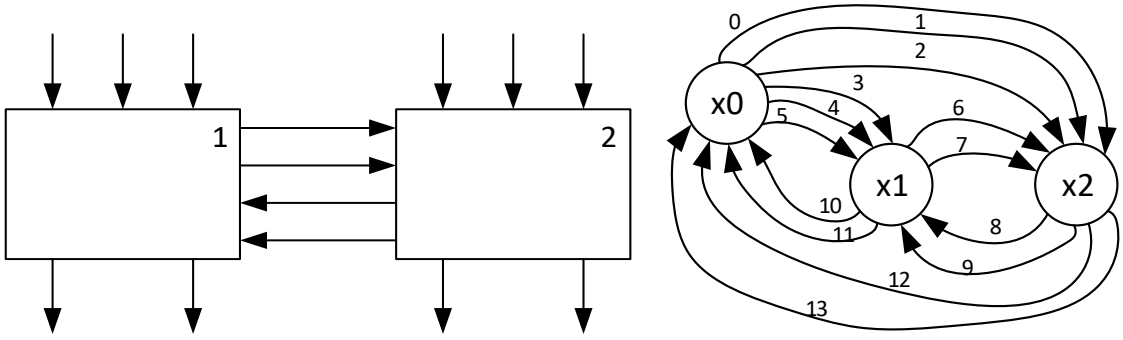
Рисунок 3.8. Часова діаграма роботи пристрою ШПФ.

Аналізуючи часову діаграму пристрою, бачимо, що програмна складність тут обумовлена тільки синхронізацією подачі вхідних даних та зняттям результату з регістрів. Таким чином програмна складність в конвеєрному режимі наближається до нуля.

Також в даній схемі присутній паралелізм, візуально можна побачити два паралельні блоки виконання обчислень, кожен з яких складається з двох помножувачів, трьох суматорів та регістрів, для отримання вхідних даних та збереження результатів обчислень.

Структурна складність пристрою ШПФ.

Для проведення аналізу структурної складності побудуємо блок-схему (рис. 3.9 а) розглянутого пристрою ШПФ (рис. 3.7).



а) б)

Рисунок 3.9. а) блок схема; б) граф - пристрою ШПФ з чотирма пристроями множення.

Блок-схема пристрою будується методом об’єднання однорідних частин схеми у блоки. Якщо розглянути структуру пристрою ШПФ (рис. 3.7), можна відмітити, що вона складається із двох однорідних частин, які містять у своєму складі по 5 регістрів, 2 помножувачі та 3 суматори кожна. Таким чином було сформовано блок-схему такого пристрою, яка зображена на рисунку (рис. 3.9 а). Блок схема містить два умовні симетричні блоки 1 та 2, а також лінії які позначають вхідні та вихідні сигнали, які потрапляють на пристрій.

Відповідно до блок-схеми пристрою будуємо граф роботи пристрою (рис. 3.9 б), де блоки схеми відповідають вузлам графу, в даному випадку x1 та x2, а шини зв’язків схеми – зв’язками між вузлами графу. В наведеному графі

додатково зображений вузол x0, який відповідає пам’яті пристрою, тобто елемент де зберігаються вхідні дані і передається результат роботи пристрою. Для обчислення структурної складності побудувано матрицю інциденцій графу (3.6).

I=

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
x0	1	1	1	1	1	1					-1	-1	-1	-1
x1				-1	-1	-1	1	1	-1	-1	1	1		
x2	-1	-1	-1				-1	-1	1	1			1	1

(3.6)

Отже, використовуючи формулу (1.16) обчислюємо структурну складність для розглянутого пристрою ШПФ $S = -28\log_2 \frac{28}{3 \cdot 14} \approx 17$. Таке значення структурної складності є достатньо низьким, але фактично, при проектуванні такого пристрою ШПФ, розробнику необхідно буде створити проект одного блоку, і потім об’єднати їх в робочу систему. Тому, для проведення повного аналізу структурної складності, додатково було обраховано значення структурної характеристики складності всередині одного блоку $S_{бл}$ = 137.8 [102].

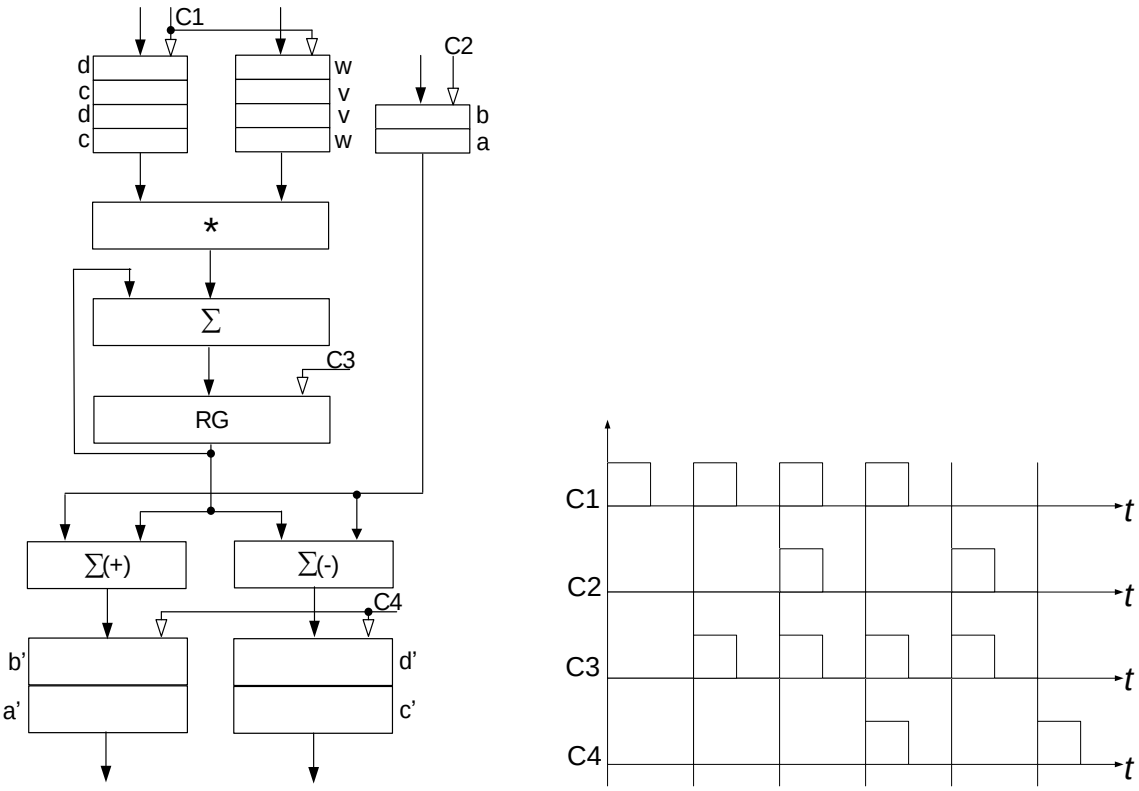
Після проведення аналізу характеристик складності пристрою швидкого перетворення Фур’є на чотирьох притроях множення можна зробити висновок, що така структура (рис. 3.7) підходить для реалізації у спецпроцесорах опрацювання сигналів, так як має низьке значення програмної складності та апаратної складності. Звичайно потрібно провести параметричну оптимізацію даного пристрою для покращення значення структурної складності всередині блоку, так як воно є досить високим. Для повної картини, додатково проведемо аналіз відомих схем пристрою швидкого перетворення Фур’є на одному пристрої множення та на двох пристроях множення.

3.2.1. H-модель пристрою ШПФ з одним помножувачем

Розглянутий варіант (рис. 3.7) є оптимальним для задачі, де основною метою є досягнення високої продуктивності. Його можливо покращити за рахунок повної конвеєризації обчислювального процесу. Якщо ж потрібний

варіант синтезу, який дозволяє отримати пристрій ШПФ з мінімізованим значенням апаратної складності, структура такого пристрою буде інша (рис. 3.10 а).

Потрібно зазначити, що організація завантаження вхідних даних та видача результатів в регістри для даного варіанту є значно складніша, ніж у попередньому варіанті схеми, що спричинило зростання програмної складності. На рисунку 3.10б наведена часова діаграма роботи пристрою швидкого перетворення Фур'є з одним помножувачем. Для повного обчислення метелика швидкого перетворення Фур'є виконуються чотири послідовні операції множення, так як тут присутній 1 пристрій множення. Отже використовуючи часову діаграму (рис. 3.10 б) та формулу (1.13) визначено програмну складність пристрою $\mathrm{P} = -12\log_2 \frac{12}{6 \cdot 4} \approx 12.$



а) б)

Рисунок 3.10. а) модифікована SH-модель пристрою ШПФ з одним пристроєм множення; б) часова діаграма роботи пристрою.

Структура такого пристрою значно відрізняється від структури попереднього, вона є неоднорідною. Для обчислення структурної складності

ми не можемо розділити схему на однорідні блоки, тому при обчисленні структурної складності пристрою кожен елемент системи виступає окремим елементом у матриці інциденцій. Структурна складність такого пристрою S = 48 [2].

Можна зробити наступний висновок, що дана структура (рис. 3.10 а) є кращою за значенням апаратної складності. Вона підходить для тих випадків, коли програмна складність не відіграє домінантного значення, а апаратура є обмежена. Тобто маємо обмежене місце на кристалі, обмежені апаратні засоби для реалізації поставленого завдання. Отже, отримуємо виграш на апаратних засобах, але програємо у швидкодії пристрою.

3.2.2. H-модель пристрою ШПФ з двома пристроями множення

Проміжним рішенням для апаратної реалізації метелика ШПФ може бути наступна схема (Рис. 3.11).

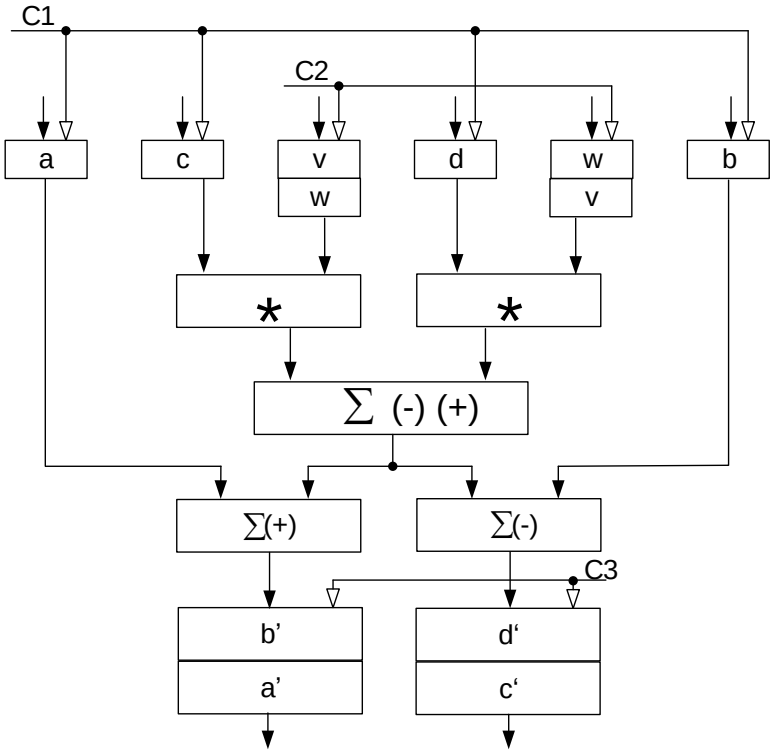


Рисунок 3.11. Модифікована SH-модель пристрою ШПФ з двома пристроями множення.

Даний варіант є чимось середнім між двома раніше розглянутими. Що до програмної складності, то вона тут є меншою, ніж у пристрої ШПФ з одним помножувачем, але більшою, ніж з чотирма [105].

Порівнюючи три розглянуті схеми можна провести підсумки. По продуктивності схеми з одним, двома та чотирма пристроями множення мають наступне відношення: 1:2:4. Для апаратної складності таке співвідношення порушується через кількість регістрів, яких в пристрої з одним пристроєм множення більше на 2, ніж в пристрої з двома пристроями множення, і на 4, ніж з чотирма. За питомою продуктивністю кращою являється схема з чотирма пристроями множення, яка за своєю структурою відповідає канонічній формі алгоритму ШПФ. Такий ж висновок напрошується при аналізі програмної складності.

Провівши аналіз характеристик складності пристроїв, можна вивести значення характеристик у наступну таблицю.

Таблиця 3.2

Характеристики складності пристроїв ШПФ

Пристрій ШПФ	на 1 перемножувачі	на 2-х перемножувачах	на 4-х перемножувачах
S	48	44.8	16.8
A	2n	3n-2	4n
L	6t	3t	t
P	12	4.8	0

Отже, порівняння 3-х схем реалізації пристрою ШПФ, з одним, двома та чотирма пристроями множення показало, що за питомою продуктивністю, програмною та часовою складністю найкращою є схема з чотирма пристроями множення, яку ми розглядаємо для прикладу проведення параметричної оптимізації, з вимогою на конвеєризацію процесу обчислень, та затримку сходинки конвеєра не більшою, ніж затримка одного багато розрядного суматора.

3.2.3. Параметрична оптимізація пристрою ШПФ

Для отримання швидкодії конвеєра такої, щоб затримка сходинки була не більшою ніж затримка одного багато розрядного суматора, в схемі ШПФ потрібна оптимізація одного з блоків, а саме пристрою множення. Такий пристрій множення нам вже відомий, тут ми можемо скористатися конвеєрним пристроєм множення з діагональним розповсюдженням переносу (рис. 2.23). У такому пристрої множення, методом розділення відомої матричної схеми множення з діагональним розповсюдженням переносу на дві частини, та додавання конвеєрних регістрів, було отримано швидкодію роботи конвеєра рівносильну швидкодії роботи багато розрядного суматора.

На рисунку (рис. 3.12) зображений варіант чотирисходинкового конвеєрного пристрою, який виконує алгоритм метелик швидкого перетворення Фур'є.

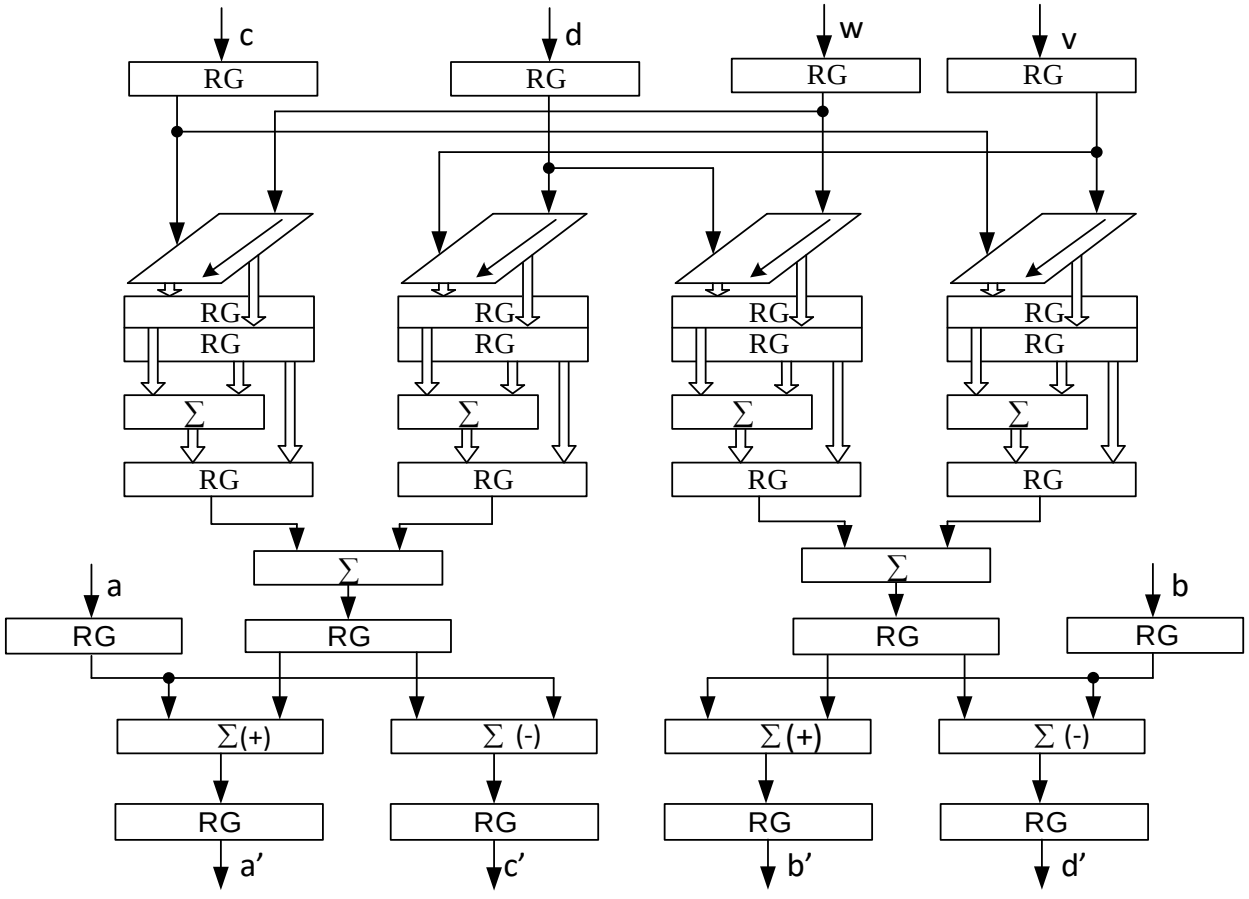


Рисунок 3.12. Конвеєрне виконання алгоритму «метелик ШПФ» з чотирма пристроями множення

Як вже зазначалось, керування операціями на суматорах, додавання чи віднімання, виконується за допомогою сигналів керування які подаються на кожен суматор.

Зовнішня структурна складність такого конвеєрного пристрою буде така сама, як і у пристрою зображеного на рисунку (рис. 3.7) S = 16.8. Це зумовлено тим, що структуру схеми можна розділити на два незалежні, паралельні блоки виконання. Але внутрішня структура блоків змінилась, їх структурна складність збільшилась за рахунок додавання регістрів при конвеєризації схеми, після проведення необхідних обрахунків, структурна складність одного блоку $S_{бл}$ = 202 [102]. Апаратна складність також збільшилась при додаванні регістрів.

Таку схему, для зручності проектування, можна модифікувати методом розділення першого суматора в кожному блоці. Таким чином схему буде розділено на 4 паралельні вітки виконання алгоритму (рис. 3.13). Такий варіант буде значно зручнішим в процесі проектування пристрою, наприклад, при створенні прошивки для FPGA. Потрібно створити проект одного такого блоку і просто продублювати його, правильно організувавши з'єднання між блоками.

Методом спрощення розробки пристрою, було порушено його структуру. В новому вигляді пристрій «метелик ШПФ» складається з чотирьох паралельних блоків, тому структурна складність пристрою збільшилась.

Таким чином, використовуючи граф схеми пристрою (рис. 3.14 б), та побудувавши на його основі матрицю інциденцій (3.7), можна визначити, що структурна складність пристрою $S = -40log_2\frac{40}{5 \cdot 20} \approx 52{,}8$. Як бачимо вона є більшою, ніж у нерозділеному пристрої з двома паралельними ланками. Але, взявши до уваги те, що ми розділили пристрій на 4 гілки, кожна з 4-х гілок (рис. 3.15) має значно меншу структурну складність, ніж 1 гілка у попередньому варіанті.

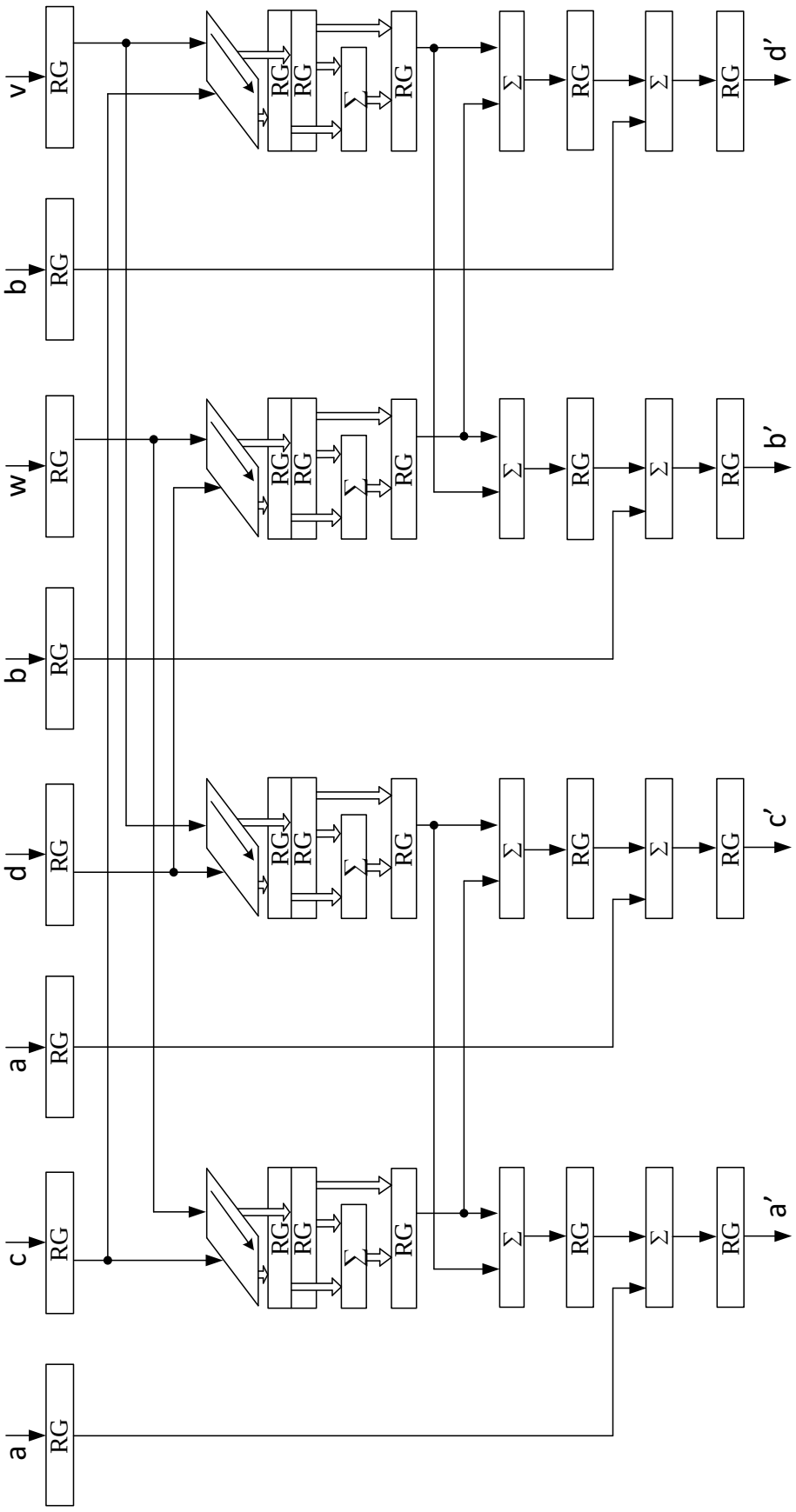
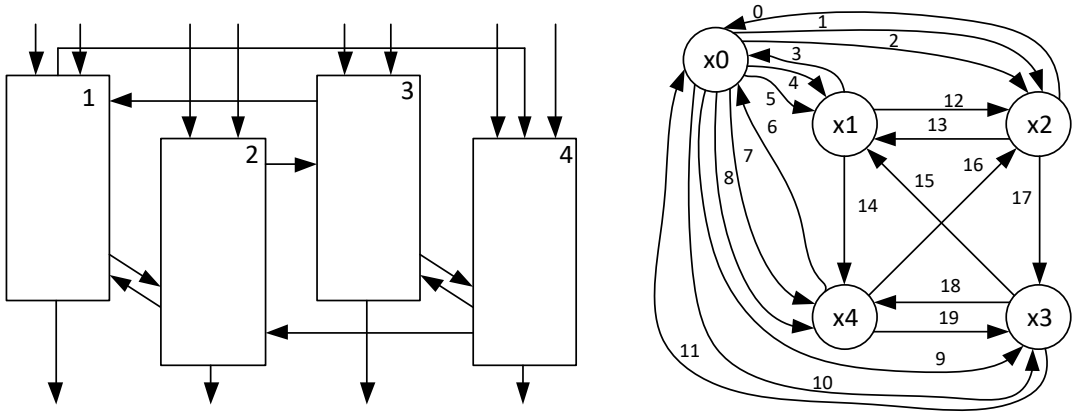


Рисунок 3.13. Конвеєрний пристрій ШПФ з розділенням на 4 паралельні гілки



а) б)

Рисунок 3.14. а) блок-схема; б) граф-схема пристрою швидкого перетворення Фур’є розділеного на 4 блоки

I=

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
x0	-1	1	1	-1	1	1	-1	1	1	1	1	-1								
x1				1	-1	-1							1	-1	1	-1				
x2	1	-1	-1										-1	1			-1	1		
x3										-1	-1	1				1		-1	1	-1
x4							1	-1	-1						-1		1		-1	1

(3.7)

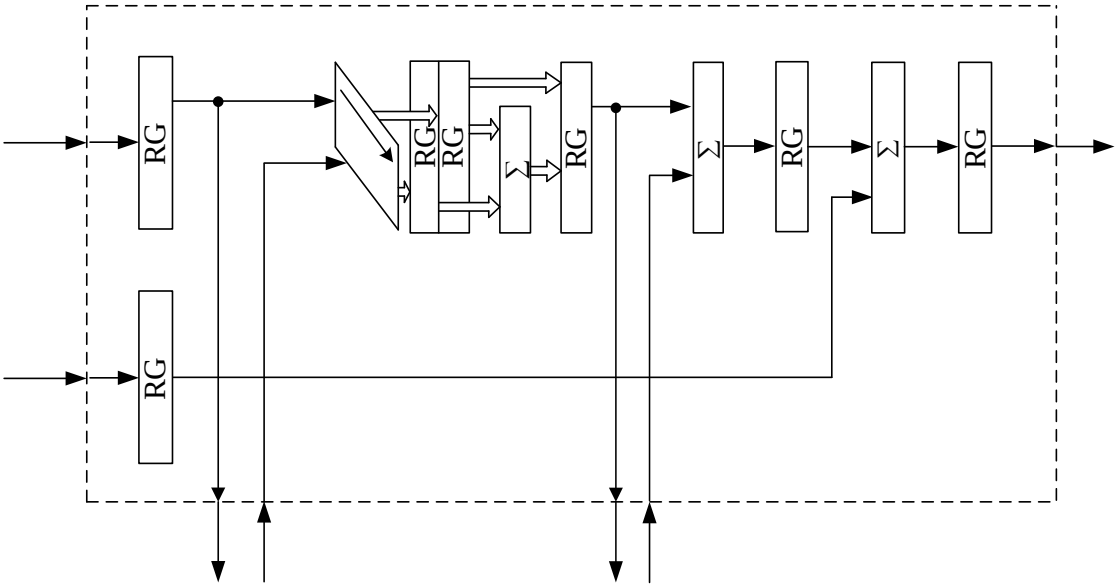


Рисунок 3.15. SH-модель одного блоку конвеєрного пристрою ШПФ

Для обчислення структурної складності одного блоку конвеєрного пристрою швидкого перетворення Фур’є (рис. 3.15) потрібно побудувати граф

схеми пристрою, та матрицю інциденцій. Кожна з вершин графу x1-x10 (рис. 3.16) відповідає блокам схеми пристрою (рис. 3.15) – регістри, елементи помножувача, суматори. Вершина x0 – це зовнішня пам’ять, з якої на пристрій поступають вхідні дані та куди передаються результати обчислень. Зв’язки графу u0-u18 відповідають зв’язкам на схемі (рис.3.15).

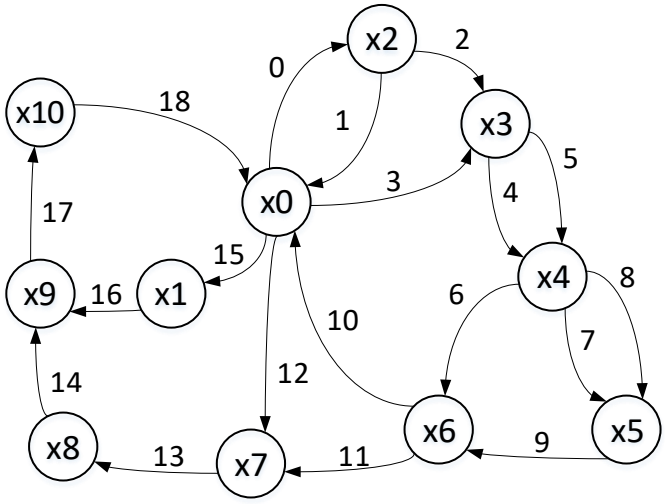


Рисунок 3.16. граф-схема одного блоку пристрою швидкого перетворення Фур’є

На основі побудованого графу схеми однієї гілки обчислення метелика швидкого перетворення Фур’є (рис. 3.16) формуємо матрицю інциденцій (3.8).

I=

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
x0	1	-1		1							-1		1			1			-1
x1																-1	1		
x2	-1	1	1																
x3			-1	-1	1	1													
x4					-1	-1	1	1	1										
x5								-1	-1	1									
x6							-1			-1	1	1							
x7												-1	-1	1					
x8														-1	1				
x9															-1		-1	1	
x10																		-1	1

(3.8)

На основі даних із матриці інциденцій (3.8) проводимо розрахунок значення структурної складності одного блоку конвеєрного пристрою ШПФ з

чотирма помножувачами. Використовуючи формулу (1.16) отримуємо значення структурної складності $S = -38log_2 \frac{38}{11 \cdot 19} \approx 93{,}5$.

Таблиця 3.3

Характеристики складності конвеєрних пристроїв ШПФ з чотирма ПМ

Схема \ х-ка	L	A	S	$S_{бл}$	P
Конв. ШПФ, 2 гілки (рис. 3.12)	n	38	17	202	0
Конв. ШПФ, 4 гілки (рис. 3.13)	n	44	53	93.5	0

Проаналізувавши значення характеристик складності конвеєрних структур реалізації метелика ШПФ (рис. 3.13 та рис. 3.14) можна зробити наступний висновок: методом збільшення значення апаратної характеристики складності пристрою досягнуто зменшення структурної складності блоку пристрою ШПФ, що значно спрощує проектування такої системи при розробці. Значення структурної характеристики складності зменшилось на 50% [102].

3.3. RH-модель пристрою суміщення алгоритмів згортки та швидкого перетворення Фур'є

Одним із варіантів реалізації високоефективних спецпроцесорів є суміщення операцій на одній структурі. Такий спосіб реалізації спеціальних функцій у спецпроцесорах обробки сигналів дає можливість мінімізувати значення апаратної характеристики складності, при реалізації декількох спеціальних функцій у одній структурі, з можливістю переключення між функціями за допомогою мультиплексорів. При чому така реалізація не суттєво впливає на значення часової складності пристрою.

На рисунку 3.17 наведена RH-модель пристрою, в якому суміщені операції реалізації спеціальних функцій згортки (рис. 3.2) та швидкого перетворення Фур'є (рис. 3.14). Такий пристрій (рис. 3.17) виконуватиме обчислення алгоритму згортки, або швидкого перетворення Фур'є, в залежності від того, як будуть переключені мультиплексори (*MX*) на схемі. На схемі (рис. 3.17) червоними лініями умовно позначені сигнали, які використовуються для передачі даних при обчисленні операції метелика швидкого перетворення Фур'є, а чорними – для передачі даних при обчисленні алгоритму згортки. У якості пристрою множення на схемі доцільно використати двосходинковий конвеєрний пристрій множення (рис.2.29).

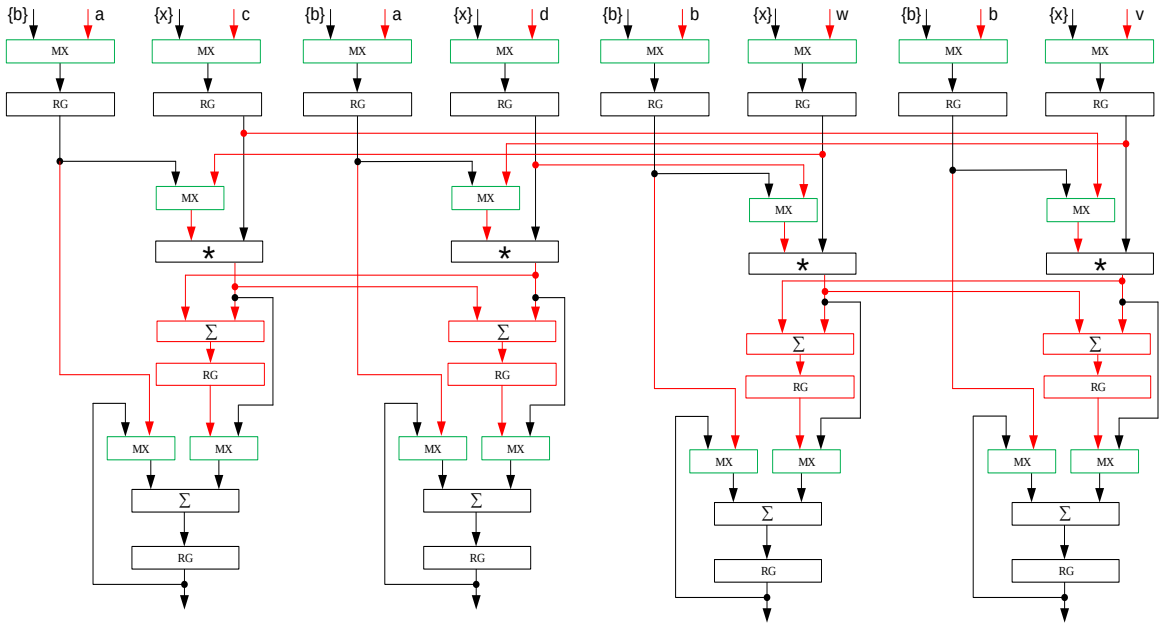
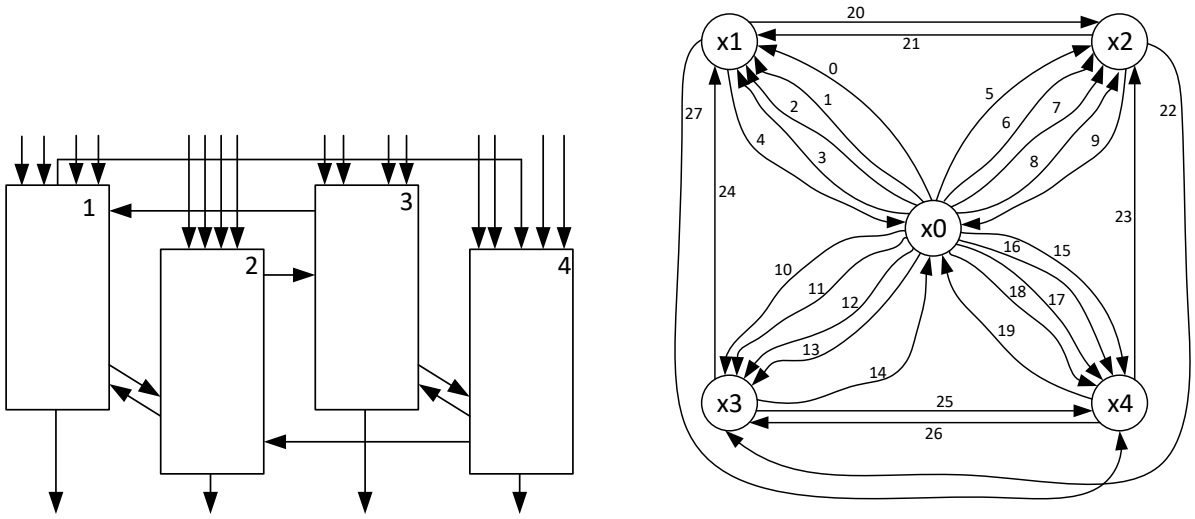


Рисунок 3.17. RH-модель із суміщенням спеціальних функцій згортки та швидкого перетворення Фур'є

При аналізі реконфігурованого пристрою бачимо, що він розділений на чотири симетричні гілки (рис. 3.18 а), як і у випадку пристрою швидкого перетворення Фур'є, їх відмінність полягає лише у кількості вхідних сигналів. Таким чином, при побудові матриці інциденцій (3.9) для граф-схеми (рис. 3.18 б) реконфігурованого пристрою згортки та ШПФ і проведенні необхідних обчислень можемо визначити, що значення структурної складності такого пристрою $S = -56 \cdot log_2 \frac{56}{140} \approx 72.8$.



а) б)

Рисунок 3.18. а) блок-схема; б) граф-схема пристрою із суміщенням спеціальних функцій згортки та швидкого перетворення Фур'є

I

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
x0	1	1	1	1	-1	1	1	1	1	-1	1	1	1	1	-1	1	1	1	1	-1								
x1	-1	-1	-1	-1	1																1	-1			-1			1
x2						-1	-1	-1	-1	1											-1	1	1	-1				
x3											-1	-1	-1	-1	1								-1		1	1	-1	
x4																-1	-1	-1	-1	1				1		-1	1	-1

(3.9)

Натомість, значення структурної складності однієї гілки схеми пристрою (рис. 3.19) значно змінилась, у порівнянні із пристроєм ШПФ (рис. 3.16). Це зумовлено додаванням мультиплексорів, які перемикають виконання операцій, та відповідно збільшенням зв'язків у схемі.

Після побудови граф-схеми гілки реконфігурованого пристрою згортки та ШПФ (рис. 3.20), та на її основі – матриці інциденцій (3.10), можна обрахувати значення структурної складності пристрою, яке буде рівним $S = -58log_2\frac{58}{29 \cdot 16} = 174$.

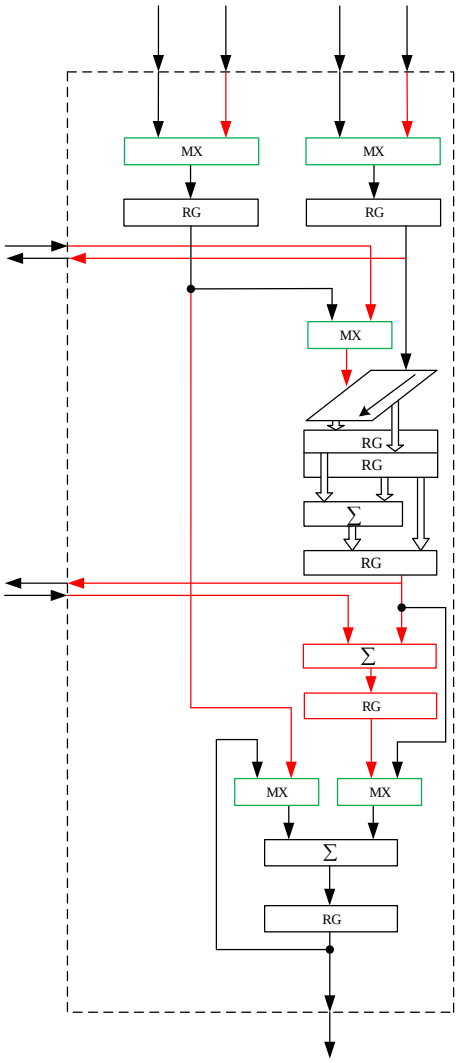


Рисунок 3.19. RH-модель одного блоку реконфігурованого пристрою

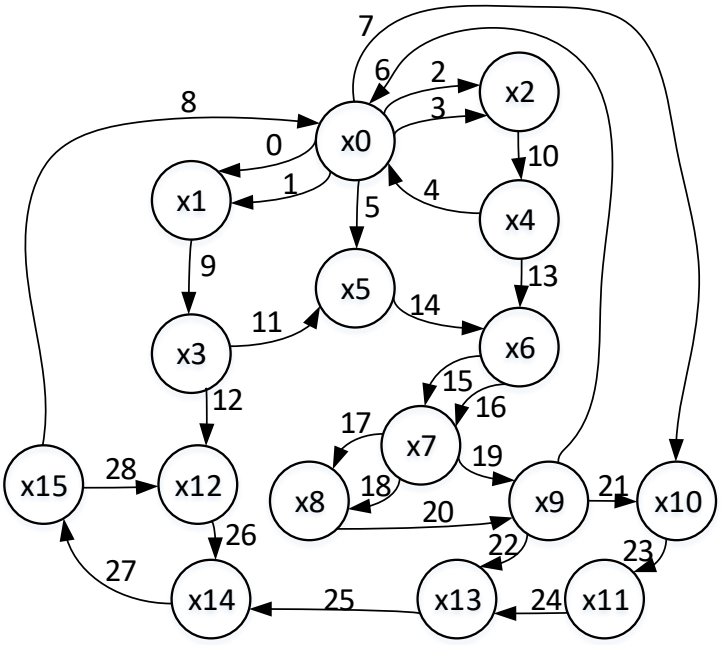


Рисунок 3.20. граф-схема одного блоку реконфігурованого пристрою

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
x0	1	1	1	1	-1	1	-1	1	-1																				
x1	-1	-1								1																			
x2			-1	-1							1																		
x3										-1		1	1																
x4					1						-1			1															
x5						-1						-1			1														
x6														-1	-1	1	1												
x7																-1	-1	1	1	1									
x8																		-1	-1		1								
x9							1													-1	-1	1	1						
x10								-1														-1		1					
x11																								-1	1				
x12													-1														1		-1
x13																							-1		-1	1			
x14																										-1	-1	1	
x15									1																			-1	1

I (3.10)

Якщо порівняти значення структурної складності однієї гілки пристрою обчислення метелика швидкого перетворення Фур’є (рис. 3.16), та гілки реконфігурованого пристрою (рис. 3.20), можемо побачити, що структура пристрою зросла на 46%, проте такий пристрій виконує дві спеціальні функції послідовно і є значно оптимальнішим по апаратній складності у порівнянні з двома паралельно розташованими спеціальними функціями на кристалі спецпроцесора. Апаратна складність реконфігурованого пристрою зросла на 20 мультиплексорів у порівнянні із пристроєм ШПФ (рис. 2.16). Якщо реалізовувати ці два пристрої (згортка та ШПФ) окремими елементами на кристалі спецпроцесора, необхідно використовувати 12 суматорів та 8 перемножувачів, а у випадку реконфігурованої структури – 8 суматорів та 4 перемножувачі.

В результаті суміщення пристроїв згортки та ШПФ, отримано реконфігурований 4-х сходинковий конвеєрний пристрій, затримка на сходинці конвеєра у порівнянні із пристроєм ШПФ зросла тільки на час спрацювання мультиплексора.

Висновки до розділу

1. Отримано характеристики складності SH-моделі згортки, реалізованої на базі систолічної структури й показано, що її не доцільно

застосовувати для реалізації у спецпроцесорах, оскільки такий пристрій має високу часову та апаратну складність.

2. Отримано характеристики складності SH-моделі згортки з неоднорідною структурою й показано, що таку структуру найкраще застосовувати для реалізації у спецпроцесорах, оскільки вона має високу швидкодію та низьку апаратну складність й найкраще підходить для суміщення з іншими алгоритмами.

3. Отримано характеристики складності SH-моделі метелика швидкого перетворення Фур'є на одному помножувачі й показано, що її не доцільно застосовувати для реалізації у спецпроцесорах, оскільки такий пристрій має високу часову та програмну складність.

4. Отримано характеристики складності SH-моделі метелика швидкого перетворення Фур'є на двох помножувачах й показано, що її не доцільно застосовувати для реалізації у спецпроцесорах, оскільки такий пристрій має підвищені значення часової та програмної характеристик складності.

5. Проведено оптимізацію характеристики складності SH-моделі метелика швидкого перетворення Фур'є на чотирьох помножувачах й отримано SH-модель пристрою розділеного на чотири паралельні гілки, яка має у порівнянні з відомим пристроєм меншу структурну складність однієї гілки на 50%, що значно спрощує процес проектування, а також найкраще підходить для суміщення на одній структурі з алгоритмом згортки.

6. Отримано RH-модель із суміщенням алгоритмів згортки та метелика швидкого перетворення Фур'є, яка має оптимальні значення часової та структурної характеристик складності.

РОЗДІЛ 4
РЕАЛІЗАЦІЯ ОПТИМІЗОВАНИХ ПРИСТРОЇВ СПЕЦПРОЦЕСОРІВ

Елементною базою для реалізації оптимізованих пристроїв спецпроцесорів опрацювання сигналів вибрано FPGA (англ. field-programmable gate array), які складаються із множини вентилів та зв'язків між ними, що програмуються відповідно до розробленого пристрою. Поширенню FPGA сприяє те, що низка фірм Xilinx, Altera, Microsemi, Atmel та інші пропонують на ринку велике розмаїття кристалів FPGA з різними характеристиками (тактова частота роботи, кількість реконфігурованих блоків, кількість арифметичних блоків і т.д.) і тим самим розширюють можливості їх застосування [34].

Інструментальним засобом розробки на FPGA при проектуванні комп'ютерних засобів як на структурному рівні, так і на поведінковому рівні, є мова VHDL (англ. VHSIC (Very high speed integrated circuits) Hardware Description Language), що є однією з найпоширеніших мов опису апаратних засобів. У результаті, отримується VHDL-моделі, які в конкретно вибраній FPGA встановлюють зв'язки між вентилями інтегральної схеми.

Проведемо порівняння SH- та VHDL- моделей. Об'єктами VHDL-моделі є рівні апаратних засобів та множина зв'язків між пристроями різних ієрархічних рівнів від логічних схем, вентилів та закінчуючи процесорами.

До об'єктів VHDL- моделі встановлені такі критерії оптимальності [34]:

- мінімальна сумарна довжина усіх зв'язків;
- максимально можлива довжина одного зв'язку;
- кількість переходів з одного шару на інший;
- загальна кількість зв'язків.

Для VHDL-моделі, подібно до SH-моделі, розглядаються такі властивості як модульність та ієрархічність.

Модульність – як об'єднання за структурними та функціональними критеріями кількох елементів певного ієрархічного рівня у єдиний модуль

(об’єкт). На схемі прийнято вентиль та модуль розглядаються як самостійні одиниці, кожна з яких виконує свою функцію, тому для аналізу моделей потрібно розглядати обидві властивості – дискретність та модульність.

Властивість ієрархічність для VHDL-моделі відмінна від ієрархічності у SH-моделі. Оскільки ієрархічність закладена в модель елемента SH-моделі – елементарного перетворювача, тому вона дає можливість підсумовувати значення структурної складності на різних рівнях ієрархії. А у VHDL-моделі такої можливості немає і сукупна структурна складність не обчислюється [34].

До характеристик складності об’єктів VHDL-моделі також можна зарахувати такі характеристики складності як апаратна, часова.

Різниця між SH- та VHDL-моделями полягає у об’єктах розроблення і, відповідно до цього, у методах та результатах проектування. Для SH-моделі – це розроблення уявних структурних, функціональних та принципових схем системи, для VHDL-моделі – розроблення топології реальних схем системи в цілому.

При об’єднанні SH- та VHDL-моделей – SH-модель стає на службі у VHDL-моделі. Підчас розробки функціональної схеми, мінімізується часова, структурна та програмна характеристики складності відповідно, головним чином за рахунок апаратної складності. Потім проектується топологія системи та створюється VHDL опис системи з часовими діаграмами її функціонування.

Множина зв’язків в SH-моделі задає структурну складність. У VHDL-моделі зв’язки є елементами схеми та мають просторовий сенс, натомість у SH-моделі –зв’язки не оцінюються довжиною. У VHDL вона присутня, проте у неявній формі і математично не визначена.

Розробляючи зв’язки, між об’єктами VHDL-моделі, деколи вводяться додаткові вершини (вершини Штейнера). Вершини Штейнера можуть бути прийняті за окремі об’єкти якщо вони мають розгалуження [34].

Схема пристрою, що проектується, описується SH-моделлю, а у VHDL не описується, оскільки там, як і в програмуванні, – процес програмування

відокремлений від процесу проектування апаратури. У SH-моделі такого відокремлення немає.

Для VHDL-моделей властивість дискретність – це дискретність сітки робочого поля, визначена структурою робочого поля ПЛІС. Елементарність у VHDL-моделі відсутня і це поняття потрібно ввести. Регістрова пам’ять рахується як апаратна складність. Ємнісна складність – ПЛІС з вбудованою пам’яттю великого об’єму.

У таблиці 4.1 наведено порівняння SH- та VHDL-моделей за трьома параметрами [34].

Таблиця 4.1

Порівняння SH- та VHDL-моделей

Параметр для порівняння	**SH-модель**	**VHDL-модель**
Об’єкт (система вхідних даних)	• елементарний перетворювач; • множина зв’язків між елементарними перетворювачами.	• елементи логіки; • множина зв’язків.
Властивості об’єктів	• дискретність; • детермінованість; • елементарність; • масовість; • ієрархічність (ієрархія функцій, елементарних перетворювачів).	• модульність; • ієрархічність елементів.
Характеристики складності об’єктів	• апаратна; • часова; • структурна; • програмна.	• апаратна; • часова.

Отже, VHDL-модель є доповненням SH-моделі на схемотехнічному рівні.

У дисертації реалізація пристроїв проводилась з використанням програмного забезпечення Quartus II фірми розробника програмованих логічних інтегральних схем Altera. Коди описів всіх наведених пристроїв мовою опису апаратних засобів наведені у додатках.

4.1. VHDL-моделі оптимізованих пристроїв множення

Першим пристроєм, в якому було проведено оптимізацію характеристик складності, та який використаний для реалізації спеціальних функцій, був конвеєрний матричний пристрій множення з діагональним розповсюдженням переносу (рис. 2.23). Для спрощення процесу розробки VHDL-моделі пристрою було прийнято рішення не заміняти верхній ряд комірок Гілда на елементи кон'юнкції, так як це не суттєво впливає на час затримки сходинки конвеєра, проте ускладнює процес проектування [99].

Першочергово потрібно розробити VHDL-модель комірки Гілда (рис. 2.10а), в її складі є один логічний елемент **&**, та один одно бітовий суматор. Структура такої комірки зображена на рисурку 4.1.

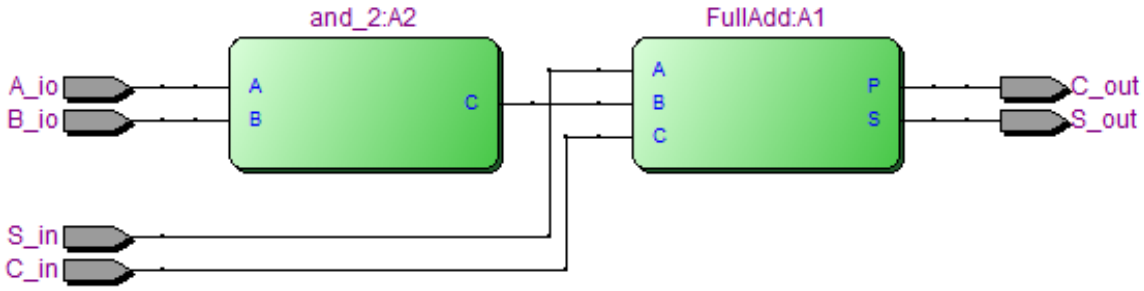


Рисунок 4.1 Структурна схема комірки Гілда у Quartus

На рисунку 4.1 наведено структуру комірки Гідна, змодельовану програмою Quartus II. Як видно з рисунка, всередині комірки присутні складові частини: Елемент and_2:A2 – це елемент виконання операції логічного &, та елемент FullAdd:A1 – елемент суматор. Внутрішнє представлення суматора можна побачити на рисунку (рис. 4.2). Структура суматора – це комбінаційна схема, яка включає в себе тільки логічні елементи [& - s1..s4, p1..p3, та OR - S, P]. Такий варіант 3-х входового суматора є досить поширеним та його варіант реалізації є доступним у літературі [2].

Після реалізації комірки Гілда для схеми пристрою множення з діагональним розповсюдженням переносу, необхідно створити матрицю цих комірок, а її розмір напряму залежить від розрядності вхідних даних. У

дисертації, для тестування було обрано розрядність вхідних даних 9 біт, а отже матриця комірок Гілда матиме розміри 9*9 комірок (рис. 4.3).

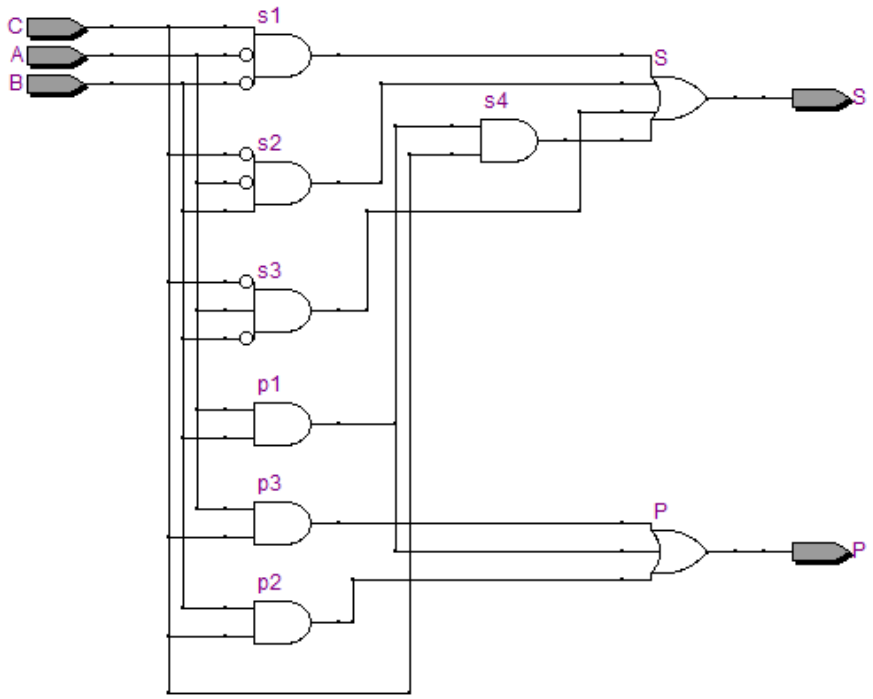


Рисунок 4.2 Внутрішня структурна схема суматора

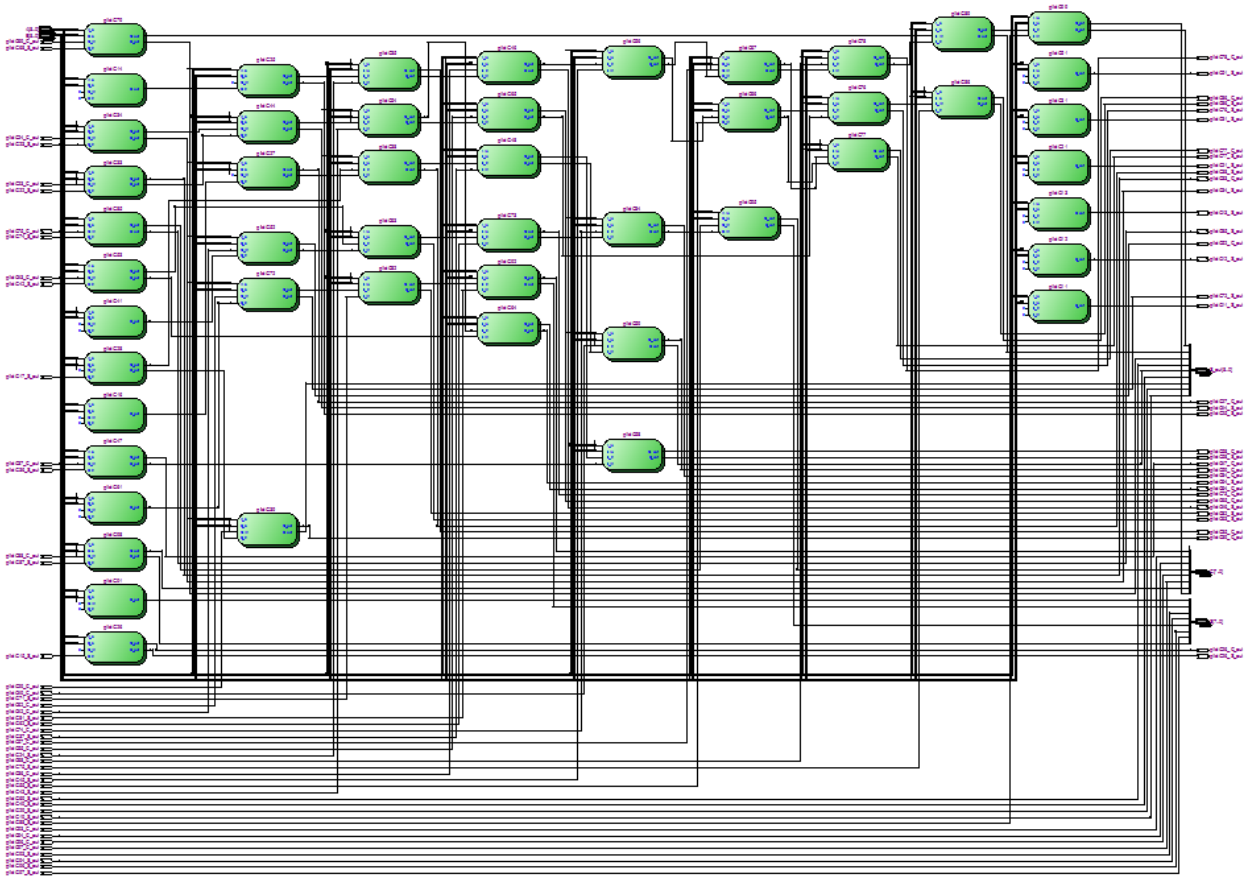


Рисунок 4.3 Структура матриці комірок Гілда розмірністю 9x9 з розповсюдженням переносу по діагоналі реалізована у Quartus

В помножувачі з діагональним розповсюдженням переносу додатковим елементом є результуючий суматор, структура елементарних перетворювачів якого зображена на рисунку 2.23 в. Реалізація одного такого елемента на FPGA у Quartus матиме такий самий вигляд як на рисунку 4.2. Приклад реалізації фінального ряду з 8-ми таких однорозрядних суматорів на FPGA у ПЗ Quartus зображений на рисунку 4.4.

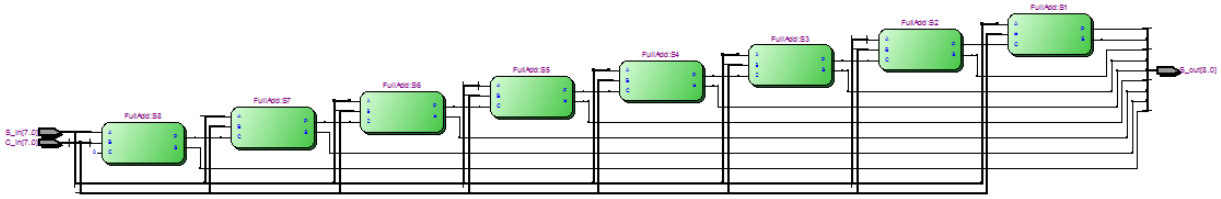


Рисунок 4.4 Структура ряду 8-ми суматорів для ПМ з діагональним переносом розрядністю вхідних даних 9 біт реалізована у Quartus

Як можемо побачити із графічного представлення, реалізація всіх елементів для створення даного пристрою множення є простою і не створить труднощів для розробника. Також в структурі конвеєрного матричного пристрою множення із горизонтальним переносом присутні регістри, які необхідні для реалізації конвеєрної передачі даних. Структура конвеєрного ПМ з діагональним переносом зображена на рисунку 4.5.

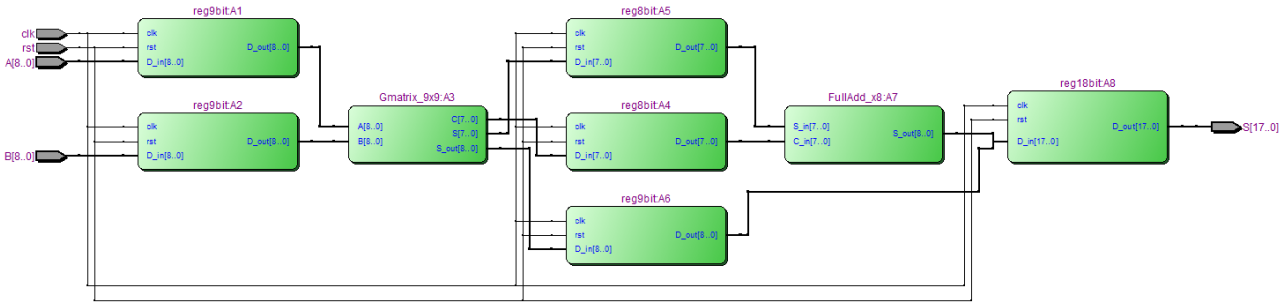


Рисунок 4.5 Структура конвеєрного матричного ПМ з діагональним розповсюдженням переносу реалізована у Quartus

Як видно з рисунку (рис. 4.5), блоки підписані як «reg» являються регістрами для конвеєрної передачі даних по структурі, яка складається із двох сходинок. Першою сходинкою є блок «Gmatrix», у якому розміщена матриця комірок Гілда (рис. 4.3). Другою сходинкою конвеєра операції множення є блок «FullAdd», у якому розміщений ряд суматорів (рис. 4.4), для фінального

обчислення операції множення. Як бачимо, робота такого пристрою відбувається наступним чином: процес множення починається із завантаження конвеєрних регістрів A1 та A2 вхідними даними, після чого значення передаються в обробку на матрицю комірок Гілда A3; наступним кроком слідує проміжне збереження результату конвеєрними регістрами A4, A5 та A6; останнім кроком є обробка проміжних даних рядом суматорів A7 і передача результату на останню сходинку конвеєра A8. На рисунку 4.6 наведена діаграма роботи двосходинкового конвеєрного матричного пристрою множення з діагональним розповсюдженням переносу.

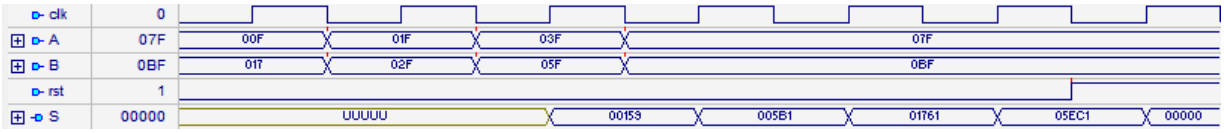


Рисунок 4.6 Діаграма просування даних по конвеєру пристрою множення

При моделюванні структури конвеєрного матричного ПМ з діагональним переносом у Quartus, на FPGA серії Cyclone IV GX, розрядністю вхідних даних 9 біт, програмне забезпечення дало наступні результати: кількість елементів логіки, необхідних для реалізації пристрою рівна 197; кількість однобітових регістрів, необхідних для реалізації конвеєра рівна 61; час затримки на сходинці конвеєра рівний 8,1нс (рис. 4.7).

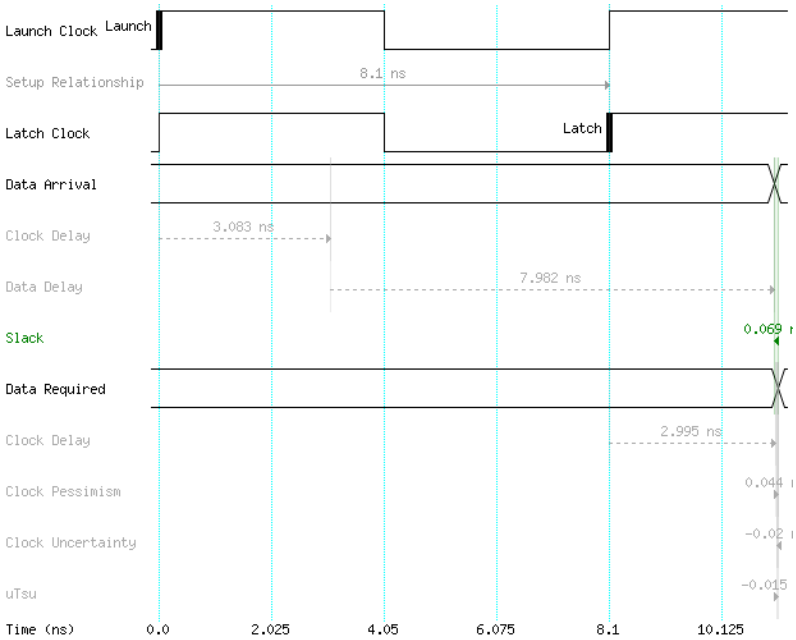


Рисунок 4.7 Діаграма відображення затримки сходинки конвеєра матричного ПМ (9x9) з діагональним розповсюдженням переносу на FPGA у Quartus

Якщо порівняти ці дані з неоптимізованим пристроєм множення, з такою ж розрядністю вхідних даних, модельованим на такому ж кристалі, то отримаємо наступні результати: кількість елементів логіки рівна 187 елементів; кількість однобітових регістрів рівна 36; час затримки на обрахунок одного результату множення рівний 14,6нс. У таблиці 4.2 приведені дані порівняння оптимізованого пристрою множення з відомим.

Таблиця 4.2

Результат моделювання конвеєрного пристрою множення з діагональним розповсюдженням переносу

показник \ ПМ	Конвеєрний оптимізований двосходинковий	Не оптимізований односходинковий
Елементи логіки (од)	197	187
Регістри (од)	61	36
Затримка (нс)	8,1	14,6

З отриманих у ході моделювання пристрою множення результатів, можна зробити наступні висновки:

1. Для реалізації конвеєрного ПМ з діагональним розповсюдженням переносу, з розрядністю вхідних даних 9біт, необхідно 197 елементів логіки, що на 5% (10 ел.) більше, у порівнянні із необхідною кількістю елементів для реалізації неоптимізованого ПМ.
2. Для реалізації двосходинкового конвеєрного ПМ з діагональним розповсюдженням переносу, з розрядністю вхідних даних 9 біт, необхідно 61 однобітовий регістр, що на 41% (25 рег.) більше, у порівнянні із необхідною кількістю регістрів для реалізації неоптимізованого односходинкового ПМ.
3. Перший результат операції множення на двосходинковому конвеєрному матричному ПМ з’явиться з затримкою 16,2нс, а на неоптимізованому ПМ за 14,6нс. Проте всі наступні результати множення на двосходинковому конвеєрному ПМ будуть з’являтися із затримкою 8,1нс, а для неоптимізованого ПМ нічого не зміниться.

Отже, підчас оптимізації характеристик складності пристрою множення з діагональним розповсюдженням переносу було реалізовано структуру двосходинкового конвеєрного ПМ, із затримкою сходинки конвеєра 8,1нс, що на 6,5нс краще (44%), ніж у випадку із затримкою на неоптимізованому пристрої множення при великій кількості послідовних множень.

4.2. VHDL-моделі оптимізованих пристроїв спеціальних функцій спецпроцесора обробки сигналів

У роботі проведено оптимізацію характеристик складності таких пристроїв спеціальних функцій цифрової обробки сигналів як алгоритм «згортка» та алгоритм «метелик» швидкого перетворення Фур'є. Дані пристрої були оптимізовані за всіма характеристиками складності згідно поставлених вимог. Так як у розглянутих алгоритмах присутня операція множення, тому підчас їх проектування було прийнято рішення використовувати у їх складі оптимізованого конвеєрного матричного пристрою множення з діагональним розповсюдженням переносу, VHDL-модель якого зображена на рисунку (рис. 4.5).

4.2.1. VHDL-модель неоднорідної структури реалізації алгоритму згортки

Розробка VHDL-моделі пристрою реалізації алгоритму цифрової фільтрації виконувалась відповідно до структури, зображеної на рисунку 3.4. Основу такого пристрою виконання алгоритму згортки складають операції множення та сумування. Модель алгоритму згортки описаного мовою VHDL та реалізованою програмним забезпеченням Quartus зображено на рисунку 4.8.

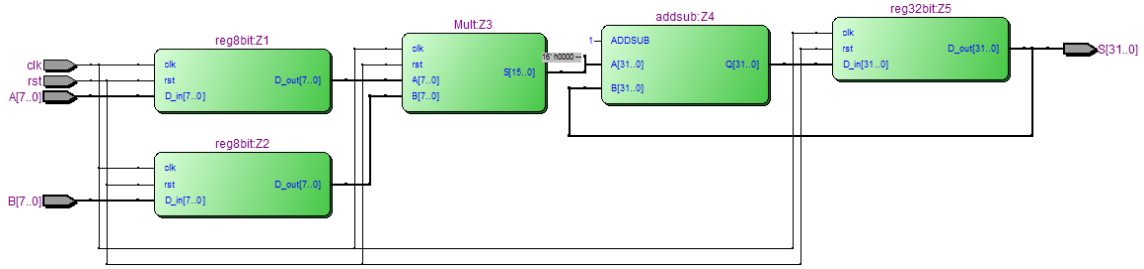


Рисунок 4.8 Структура алгоритму згортки реалізована у Quartus

Пристрій згортки, структура якого зображена на рисунку 4.8, був реалізованим із розрядністю вхідних даних 1 байт. Елементами такої системи є конвеєрні регістри (reg8bit), пристрій множення (Mult) та сумування/віднімання з накопиченням результату (addsub + reg32bit). Внутрішнє представлення елемента (addsub) зображено на рисунку 4.9.

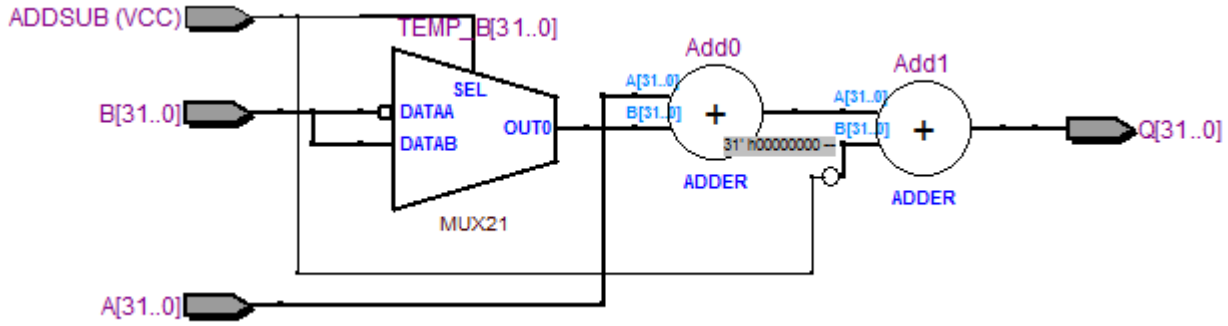


Рисунок 4.9 Структура елемента сумування/віднімання реалізована у Quartus

Даний елемент (рис. 4.9) складається з мультиплексора (MUX21) та двох суматорів (ADDER). В залежності від значення вхідного сигналу на вході (ADDSUB) відбувається операція додавання або віднімання вхідних сигналів (A) та (B).

У якості елемента множення у структурі згортки (рис. 4.8) використано двосходинковий конвеєрний пристрій множення із діагональним розповсюдженням переносу (рис. 4.5). В результаті отримано конвеєрну структуру, яка складається із 3-х сходинок, що реалізовує алгоритм згортки.

При моделюванні структури згортки у програмі Quartus, на FPGA серії Cyclone IV GX, розрядністю вхідних даних 8 біт, програмне забезпечення дало наступні результати: кількість елементів логіки, необхідних для реалізації пристрою рівна 207; кількість однобітових регістрів, необхідних для реалізації конвеєра рівна 86; час затримки на сходинці конвеєра рівний 8,1нс.

4.2.2. VHDL-модель структури реалізації алгоритму «метелика» швидкого перетворення Фур'є

Розробка VHDL-моделі оптимізованого пристрою реалізації алгоритму ШПФ, розділеного на 4 блоки, виконувалась відповідно до структури,

зображеної на рисунку 3.14. Пристрій складається із 4-х паралельних гілок, тому структура його при моделюванні на ПЛІС за допомогою програми Quartus матиме наступний вигляд (рис. 4.10).

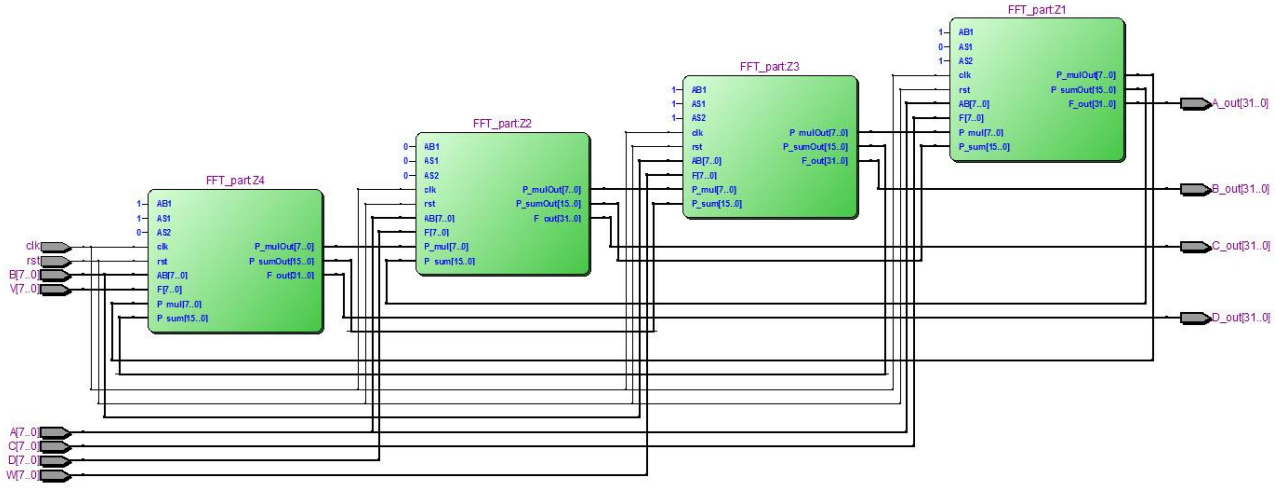


Рисунок 4.10 Структура алгоритму ШПФ, розділеного на 4 паралельні гілки, реалізована у Quartus

Всередині кожен із блоків пристрою ШПФ складається із конвеєрних регістрів, пристрою множення та двох суматорів, відповідно до рисунку 3.16. Конвеєр обчислення операції метелика швидкого перетворення Фур'є складається із 4-х сходинок. Представлення одного такого блоку на FPGA зображено на рисунку 4.11.

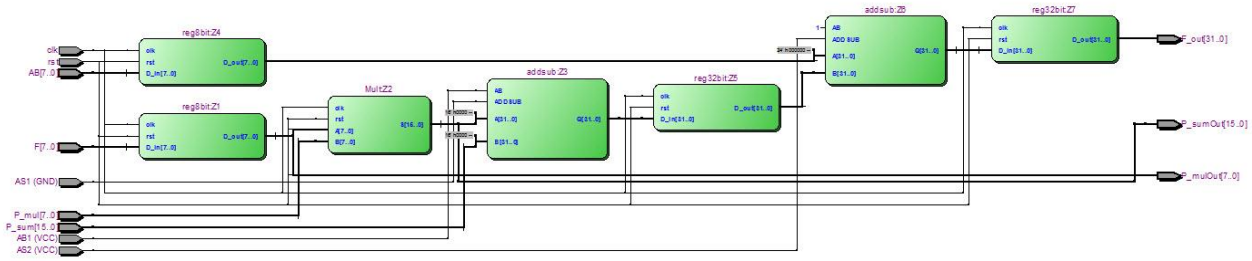


Рисунок 4.11 Структура однієї гілки алгоритму ШПФ реалізована у Quartus

У якості елемента множення (Mult) у структурі ШПФ використано двосходинковий конвеєрний пристрій множення із діагональним розповсюдженням переносу (рис. 4.5). Внутрішня структура елементів сумування / віднімання (addsub) така сама, як і на рисунку 4.9.

Підчас моделювання пристрою ШПФ (рис. 4.10) програмою Quartus II на FPGA моделі Cyclone IV GX, були отримані наступні результати: кількість елементів логіки, необхідних на реалізацію структури рівна 929; кількість

одно бітових регістрів, необхідних на реалізацію – 350; затримка сходинки конвеєра – 8,1нс.

Також було окремо проведено моделювання одного із паралельних блоків, для аналізу результатів. При моделюванні одного блоку (рис. 4.11), отримано наступні результати: кількість елементів логіки – 286; кількість одно бітових регістрів, необхідних на реалізацію – 104; затримка сходинки конвеєра – 8,1нс.

Для аналізу якості оптимізації пристрою ШПФ, було також розроблено VHDL-модель неоптимізованого пристрою ШПФ, реалізованого на 4х помножувачах (рис. 3.13). Така структура, на відміну від розділеної на 4 паралельні гілки є значно складнішою для розуміння. Результатом опису її мовою VHDL та моделювання у Quartus є наступна структура (рис. 4.12). Результатом моделювання даної VHDL-моделі (рис. 4.12) у Quartus II на ПЛІС моделі Cyclone IV GX, є наступні результати: кількість елементів логіки – 849; кількість одно бітових конвеєрних регістрів – 350; час затримки конвеєра 8,1нс.

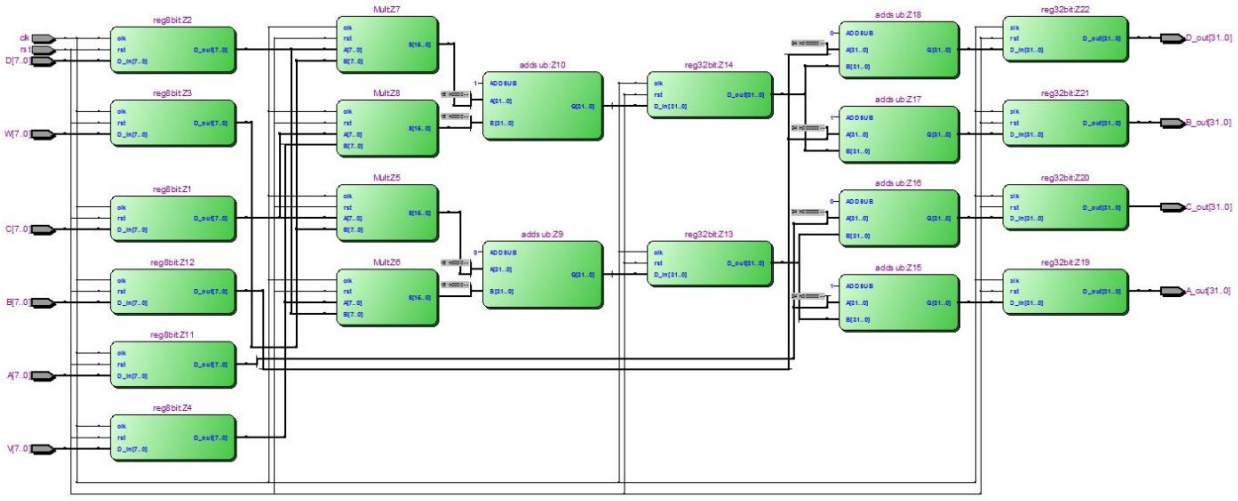


Рисунок 4.12 Структура відомого алгоритму ШПФ на чотирьох помножувачах, реалізована у Quartus

Отже, із вигляду промодельованих структур відомого пристрою ШПФ (рис. 4.12) та оптимізованого, розділеного на 4 паралельні гілки (рис. 4.10), одна гілка якого має значно простіший вигляд для розуміння та проектування, можна підвести підсумки. В ході параметричної оптимізації характеристик

складності 4-х сходинкового конвеєрного пристрою ШПФ, за рахунок збільшення значення апаратної характеристики складності (80 елементів логіки на FPGA), було отримано структуру пристрою ШПФ, методом розбиття її на 4 паралельні гілки, в результаті чого, на її проектування іде менше затрат часу. Значення структурної складності однієї гілки пристрою ШПФ (рис. 4.11) є покращеним на 50% у порівнянні із гілкою відомого пристрою ШПФ (рис. 4.12).

На рисунку 4.13 приведена часова діаграма роботи пристрою швидкого перетворення Фур'є, розділеного на 4 паралельні гілки.

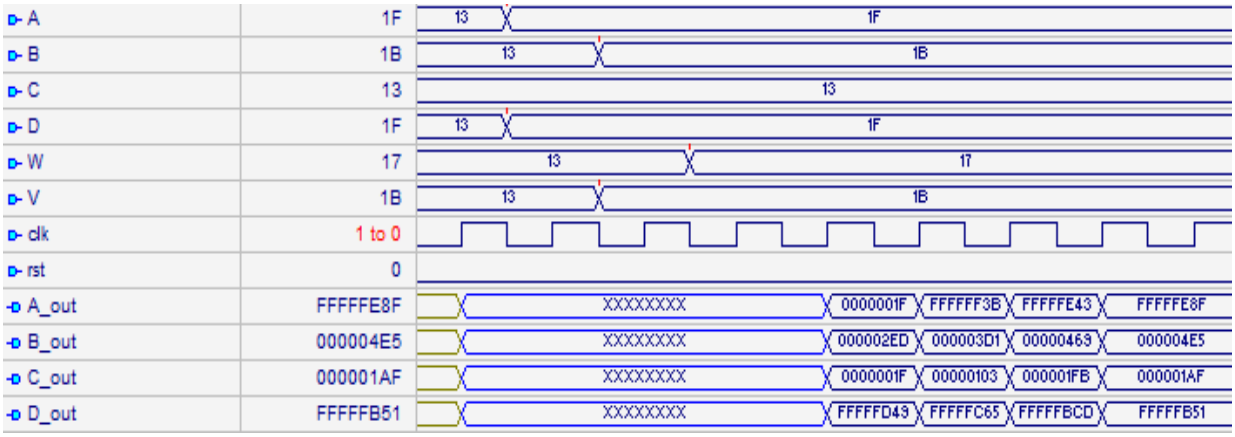


Рисунок 4.13 Діаграма роботи VHDL-моделі конвеєрного пристрою ШПФ, розділеного на 4 паралельні гілки

4.3. VHDL-модель пристроїю з суміщенням спеціальних функцій згортки та швидкого перетворення Фур'є

У роботі реалізовано RH-модель із суміщенням спеціальних функцій згортки та метелика швидкого перетворення Фур'є (рис. 3.19). Додатковими елементами, реалізація яких ще не розглядалася на рівні VHDL-моделі, є мультиплексори. Мовою VHDL для FPGA реалізація мультиплексора є досить простою, та варіанти такої реалізації доступні в відкритих джерелах. На рисунку 4.14 наведено часову діаграму прикладу роботи реалізованого автором мультиплексора, який застосований у реконфігурованому пристрої із суміщенням спеціальних функцій згортки та метелика швидкого перетворення

Фур’є. Функціонування такого пристрою не вносить суттєвих змін у затримку сходинки конвеєера, так як затримка на перемикання ліній зв’язку є короткою.

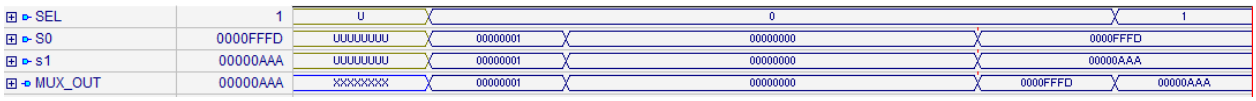


Рисунок 4.13 Діаграма роботи VHDL-моделі мультиплексора

При реалізації алгоритмів згортки та швидкого перетворення Фур’є окремими елементами на кристалі FPGA моделі Cyclone IV GX, ці спеціальні функції будуть використовувати 1757 елементів логіки кристалу. Також необхідно використовувати велику кількість ніжок спецпроцесора, на якому будуть параелльно реалізовані ці функції.

При реалізації на тому ж кристалі RH-моделі із суміщенням спеціальних функцій згортки та метелика швидкого перетворення Фур’є, які виконуються послідовно, при переключенні сигналів що проходять крізь мультиплексори, така структура (рис. 3.19) буде використовувати 1160 елементів логіки, що на 34% менше, ніж у випадку паралельної реалізації спеціальних функцій.

Розроблені у дисертації методи та пристрої знайшли застосування у роботі державного науково-дослідного підприємства "КОНЕКС", Західного центру українського відділення "Міжнародного центру наукової культури – Всесвітня лабораторія", при виконанні науково-дослідних робіт та у навчальному процесі Національного університету “Львівська політехніка”.

Результати проведених у роботі досліджень використано при розробці контрольно-перевірочної апаратури та нестандартного обладнання по темі "Січ-2-1", а також при аналізі варіантів реалізації спеціалізованих комп’ютерних пристроїв у дослідницько-конструкторській роботі "Оновлення-24 МР".

Результати дослідження характеристик складності алгоритмів були застосовані в роботі спеціалізованих програмних систем, зокрема використаний метод оптимізації характеристик складності SH-моделей алгоритмів. Цей метод дає змогу оптимізувати наявні алгоритми пошуку найближчого сервера обробки даних спеціалізованих програмних систем.

Застосування даного методу у формуванні системи моніторингу за станом вод в басейні ріки Західний Буг, стане підставою для використання його в рамках тристоронньої співпраці басейнових рад України, Польщі та Білорусії.

Висновки до розділу

1. Проведено порівняння SH- та VHDL-моделей комп'ютерних схем й показано, що VHDL-модель є доповненням до SH-моделі при оптимізації характеристик складності спеціальних функцій спецпроцесорів на схемотехнічному рівні.

2. Реалізовано VHDL-модель оптимізованого двосходинкового конвеєрного матричного пристрою множення й показано, що оптимізація його характеристик складності дала змогу покращити значення затримки на обробку даних у порівнянні з відомим неоптимізованим пристроєм на 6,5нс.

3. Реалізовано VHDL-модель оптимізованого пристрою згортки. Проведено синтез VHDL-моделей схем пристроїв ШПФ з чотирма помножувачами, відомої та розділеної на 4 блоки, й показано, що оптимізація характеристик складності спрощує та прикорює процес синтезу VHDL-моделі такого пристрою.

4. Розроблено VHDL-модель реконфігурованого пристрою, що реалізує алгоритми згортки та метелика швидкого перетворення Фур'є й показано, що суміщення спеціальних функцій для спецпроцесорів обробки сигналів скорочує об'єм обладнання на 34% у порівнянні з паралельною реалізацією спеціальних функцій.

5. Розроблені у дисертації методи та пристрої знайшли застосування у роботі державного науково-дослідного підприємства "КОНЕКС", Західного центру українського відділення "Міжнародного центру наукової культури – Всесвітня лабораторія", при виконанні науково-дослідних робіт та у навчальному процесі Національного університету "Львівська політехніка".

ОСНОВНІ РЕЗУЛЬТАТИ ТА ВИСНОВКИ

Сукупність одержаних у роботі результатів розв’язує актуальну науково-прикладну задачу створення ефективних структур спеціалізованих комп’ютерних систем, що реалізують спеціальні функції опрацювання сигналів, на основі побудови відповідних SH-моделей та оптимізації значень їх характеристик складності.

Основні наукові результати дослідження відповідають меті роботи та дають підстави для таких висновків:

1. Здійснено порівняльний аналіз основних моделей формальних алгоритмічних систем, моделей апаратних засобів спецпроцесорів та моделей апаратно-програмних засобів спеціалізованих комп’ютерних систем і показано, що моделі апаратно-програмних засобів дають змогу найкраще оптимізувати спеціалізовані комп’ютерні системи, що реалізують спеціальні функції, із врахуванням апаратної, часової, програмної та структурної характеристик складності.

2. Вдосконалено відомий метод обчислення структурної складності SH-моделей за рахунок виявлення та об’єднання у групи однотипних елементів схеми, що дало змогу отримувати матриці інциденцій значно меншого розміру без регулярно розташованих елементів а в результаті спростило обчислення й скоротило час на проектування системи.

3. Отримано характеристики складності SH-моделей пристроїв множення на багаторозрядному суматорі та конвеєрного пристрою множення й показано, що їх не доцільно застосовувати для реалізації спеціальних функцій, оскільки вони мають не оптимальні значення характеристик складності.

4. Проведено оптимізацію матричного пристрою множення з горизонтальним розповсюдженням переносу й отримано H-модель пристрою із заміною першого ряду комірок матриці схемами кон’юнкції та правого діагонального ряду комірок схемами напівсуматорів, яка має у порівнянні з відомим пристроєм меншу апаратну складність на 23% та часову складність

на 9% при 8-и розрядних вхідних даних, та відповідно 3,1% та 1% при 64-х розрядних даних.

5. Проведено оптимізацію матричного пристрою множення з діагональним розповсюдженням переносу й отримано H-модель двосходнинкового конвеєрного пристрою множення, яка має у порівнянні з оптимізованим пристроєм більшу апаратну складність на 30% при 8-и розрядних вхідних даних, та відповідно 4,5% при 64-х розрядних даних, що зумовлено додаванням у схему конвеєрних регістрів. Проте значення часової складності конвеєрного пристрою множення, у порівнянні з оптимізованим, є меншим на 56% при 8-ми розрядних вхідних даних, та відповідно на 50% при 64-х розрядних даних.

6. Отримано характеристики складності SH-моделей алгоритму згортки, реалізованого на базі систолічної структури та з неоднорідною структурою й показано, що пристрій з неоднорідною структурою найкраще застосовувати для реалізації у спецпроцесорах, оскільки вона має високу швидкодію та низьку апаратну складність й найкраще підходить для суміщення з іншими алгоритмами.

7. Отримано характеристики складності SH-моделей метелика швидкого перетворення Фур’є на одному помножувачі та на двох помножувачах й показано, що їх не доцільно застосовувати для реалізації у спецпроцесорах, оскільки такі пристрої мають високі значення часової та програмної характеристик складності.

8. Проведено оптимізацію характеристики складності SH-моделі метелика швидкого перетворення Фур’є на чотирьох помножувачах й отримано SH-модель пристрою розділеного на чотири паралельні гілки, яка має у порівнянні з відомим пристроєм меншу структурну складність однієї гілки на 50%, що значно спрощує процес проектування, а також найкраще підходить для суміщення на одній структурі з алгоритмом згортки.

9. Отримано RH-модель із суміщенням алгоритмів згортки та метелика швидкого перетворення Фур’є, яка має оптимальні значення часової

та структурної характеристик складності. На реалізацію двох пристроїв (згортки по чотирьох точках та ШПФ на чотирьох помножувачах) окремими паралельними елементами на кристалі спецпроцесора необхідно використовувати 12 суматорів та 8 помножувачів, а у випадку оптимізованої RH-моделі – 8 суматорів та 4 помножувачі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ахо А., Дж. Хопкрофт, Жд. Ульман. Структуры данных и алгоритмы. – М.: Вильямс, 2000. – 384 с.
2. Вишенчук И.М., Черкаский Н.В. Алгоритмические устройства и супер ЭВМ. – К.: Техніка, 1991. – 197 с.
3. Дунець Р.Б. Аналіз та синтез топологій комп'ютерних видавничо-поліграфічних систем: Монографія – Львів: НВФ "Українські технології", 2003. – 192 с.
4. Каган Б. М. Электронные вычислительные машины и системы: учеб. Пособие для вузов. – 3-е изд., - Москва: Энергоатомиздат, 1991. – 591с.
5. Клим Г.І. Інтелектуальна система моніторингу довкілля з використанням плівкових сенсорів / Г.І. Клим // Міжвузівський збірник "Комп'ютерно-інтегровані технології: освіта, наука, виробництво". – 2011. – № 5. - С. 120-125.
6. Мельник А.О. "Спеціалізовані комп'ютерні системи реального часу", - ДУ "Львівська політехніка", Львів, 1996. – 53 с.
7. Мельник А.О. Програмовані процесори обробки сигналів. – Львів: Видавництво Національного університету "Львівська політехніка", 2000. – 55 с.
8. Тарасенко В.П., Черкасский Н.В., Каневский Ю.С., Кривуля Г.Ф., Алипов Н.В., Завадский В.А. Арифметика, принципы организации, диагностика и формализованное проектирование вычислительных структур и устройств. — К.: Вища школа, 1989 — 343 с.
9. Коротєєва Т.О. Алгоритми та структури даних. Навчальний посібник. Львів: Видавництво Львівської політехніки, 2014. – 280с.
10. Мельник А.О. Архітектура комп'ютера. Наукове видання. – Луцьк: Волинська обласна друкарня, 2008. – 470 с.
11. Cherkaskyy Mykola. Theoretical Fundamentals Software/Hardware Algorithms // Proceedings of the International Conference TCSET"2004. February 24-28, Lviv-Slavsko, Ukraine, 2004, p. 9-13.

12. Черкаський М.В. SH-модель алгоритму. // Вісник “Комп’ютерна інженерія та інформаційні технології”. – Львів: Національний університет “Львівська політехніка”, 2001. – № 433. – С. 127–134.
13. Марков А. А. Теория алгорифмов. Труды математического института им. В. А. Стеклова.– Издательство Академии наук СССР, 1954. – Т.42. – 375с.
14. Donald Knuth The Art of Computer Programming.— Addison-Wesley Professional, 2015. — Т. Volume 4, Fascicle 6: Satisfiability. — xiii+310 pp. — ISBN 978-0-13-439760-3.
15. Проблеми Гільберта, Збірник за редакцією П.С. Александрова, М., Наука, 1969 р., 240 с.
16. Марков А.А. Нагорный Н.М. Теория алгорифмов, М., Наука, 1984 р., 432с.
17. Черкаський М.В. Еволюція тлумачення поняття “алгоритм” // Вісник Нац. ун-ту “Львівська політехніка”. – 2003. – № 492. – С.142 – 146. 29.
18. Мурад Хуссейн Халил Н-модель алгоритма и универсальная SH-модель вычислителя и их использование для исследования компьютерных средств / дис. канд. техн. наук: 05.13.13 // Национальный ун-т "Львовская политехника". — Л., 2007. — 122 с.
19. Хопкрофт Дж., Мотвани Р., Ульман Дж. Введение в теорию автоматов, языков и вычислений. — М. : Вильямс, 2007. — 528 с.
20. A.S. Tanenbaum, A.S. Woodhull. Operating Systems Design and Implement., Third Edition. – Amherst, Massachusetts. Prentice Hall. 2006. – 1080 p.
21. Колмогоров А. Н. Теория информции и теория алгоритмов. – Москва, НАУКА, 1987. – 304 с.
22. А. Н. Колмогоров, И. Г. Журбенко, А. В. Прохоров. Введение в теорию вероятностей. — М., 1982. — 160 с.
23. Мурад Х.Х. Н-модель алгоритма и универсальная SH-модель вычислителя и их использование для исследования компьютерных средств: дис. к. т. наук: 05.13.13 / М. Хуссейн Халил. — Л., 2007. — 122 с.
24. Черкаський М.В. Універсальна SH-модель // Вісник Нац. ун-ту “Львівська політехніка”. – 2004. – № 523. – С. 150–154.

25. Черкаський М.В. Оцінка складності RH-моделей згортки і ШПФ / Черкаський М.В., Ткачук Т.І. // Матеріали 4-ї міжнародної науково-технічної конференції ACSN-2009. – Львів: НВФ «Українські технології», 2009. – с. 255-258.
26. Бибило П.Н. Синтез логических схем с использованием языка VHDL. – М:. “Солон -Р”, 2000. – 384 с.
27. Глушков В.М. Синтез цифровых автоматов. – М: Наука, 1962. – 476 с.
28. Грига В. М. Оцінка варіантів синтезу матричних та багатошарових перемножувачів двійкових чисел // Науковий журнал “Комп’ютерно-інтегровані технології: освіта, наука, виробництво”. – Луцьк: Луцький національний технічний університет, 2011. - №5. – С. 120-125.
29. Черкаський М.В., Абдалла Саїд Садек, “Псевдо SH-модель” Комп’ютерні системи та мережі // Вісник Нац. ун-ту“Львівська політехніка”. – 2004. – № 523. – С. 145–150.
30. Глухов В. С., Глухова О. В. Результати оцінки структурної складності помножувачів елементів полів Галуа // Вісник Нац. ун-ту “Львівська політехніка”. – 2013. – № 773.
31. Черкаський М.В. Історичний аспект складності алгоритму/ Черкаський М.В., Мітьков В.С. // Вісник НУ “Львівська політехніка”. - 2002. - №463. - С.111-118.
32. Черкаський М.В. Складність апаратно-програмних комп’ютерних засобів. // Матеріали міжнародної науково-технічної конференції “Сучасні проблеми в комп’ютерних науках в Україні” (CCU’2000). Славське, 2000. С. 58-67.
33. Черкаський М.В. Складність програм та апаратної реалізації алгоритмів. К. Техніка, 1993. - 125 с.
34. Черкаський М.В. Співвідношення об’єктів SH- та VHDL-моделей / Черкаський М.В., Бережанський Ю.І. // Вісник НУ “Львів. політехніка”. – 2011. – № 717. – С. 199–203.

35. Грига В. Особливості побудови багатотактових операційних пристроїв / В. Грига // Матеріали 5-ї міжнародної конференції “Сучасні комп'ютерні системи та мережі: Розробка та використання” – Львів: Національний університет “Львівська політехніка”, 2011. - С. 243-244.
36. Грига В. Спеціалізований перемножувач на декілька констант / В. Грига // Матеріали 4-ї міжнародної конференції молодих вчених “Комп’ютерні науки та інженерія”. – Львів: Національний університет ”Львівська політехніка”, 2010. – С. 180-181.
37. Грушвицкий Р.И., Мурсаев А.Х., Угрюмов Е.П. Проектирование систем на микросхемах с программируемой структурой. – 2-е изд., перераб. и доп. – СПб:. БХВ-Петербург, 2007. - 736 с.
38. Палагин А.В. Проектирование реконфигурируемых цифровых систем: монография / А.В. Палагин, А.А. Баркалов, В.Н. Опанасенко, Л.А. Титаренко. – Луганск: изд-во ВНУ им. В. Даля, 2011. – 432 с.
39. Палагин А.В., Опанасенко В. Н. Реконфигурируемые вычислительные системы: Основы и приложения. – К.: Просвіта, 2006. – 280 с.
40. Palagin A.V. Design and application of the PLD-based reconfigurable devices / A.V. Palagin, V.N. Opanasenko // Design of Digital Systems and Devices. – Springer, Verlag, Berlin, Heidelberg. – 2011, Vol. 79. – PP. 59–91.
41. Palagin A. The structure of FPGA based cyclic-code converters / Alexander Palagin, Vladimir Opanasenko, Sergey Krivoi // Optical Memory & Neural Networks (Information Optics). Springer. – 2013, Vol. 22, N.4. – PP. 207–216.
42. Опанасенко В.Н. Высокопроизводительные реконфигурируемые компьютеры на базе FPGA / В.Н. Опанасенко // Проблеми інформатизації та управління, 2009. – 3(27). – С. 114 - 118.
43. Зотов В. Ю. Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы Xilinx. – М.: Горячая линия – Телеком. 2007. – 520 с.
44. Каневский Ю.С. Формализованое проектирование параллельных вычислительных структур. Арифметика, принципы организации,

диагностика и формализованое проектирование вычислительных структур и устройств, - К.: Выща шк., 1989.- С. 115-170.

45. Келим Ю.М. Вычислительная техника. Учебное пособие для студ. М: Издательский центр “Академия”, 2005. – 384 с.
46. Клим Г. Оптимізований метод вимірювання позитронних анігіляційних спектрів у наноматеріалах з розвиненою поруватістю для сенсорних застосувань / Клим Г., Костів Ю., Чалий Д., Івануса А., Ткачук Т. // Міжвузівський науково-технічний збірник "Вимірювальна техніка та метрологія", 2016, № 77, с.87-93
47. Клим Г.І. Комп’ютерна система для дослідження нанопустот в матеріалах методом анігіляційної спектроскопії / Г.І. Клим // Міжвузівський збірник Комп’ютерно-інтегровані технології: освіта, наука, виробництво. – 2013. – № 11. – С. 40-45.
48. Кочан Р.В. Комбінований метод корекції нелінійності дводіапазонних АЦП / Р.В. Кочан, О.В. Кочан, Г.І. Клим, Н.Є. Гоц // Вісник національного університету «Львівська Політехніка», сер. Автоматика, вимірювання та керування. –2014. – № 802. – С. 50-54.
49. Кун С. Матричные процессоры на СБИС: Пер. с англ. – М.: Мир, 1991. – 672 с.
50. Мельник А. О. Порівняльний аналіз способів матричного подання алгоритму / А. О. Мельник, І. Д. Яковлєва// Вісник “Комп’ютерні системи та мережі”. – Львів: Національний університет “Львівська політехніка”, 2009. – № 658. – С. 31–39.
51. Мельник А.О. Принципи побудови буферної сортувальної пам’яті. Вісник Державного університету “Львівська політехніка” “Комп’ютерна інженерія та інформаційні технології”, №307, 1996, С. 65-71.
52. Мельник А.О. Розробка двоканальних конвеєрних пристроїв для реалізації матриці корегування в алгоритмі швидкого косинусного перетворення / Мельник А.О., Ерметов Ю.О. // Вісник ДУ”ЛП”

“Комп‘ютерна інженерія та інформаційні технології”, №370, 1999, с.28-34.

53. Мельник А.О., Мельник В.А. Персональні суперкомп’ютери: архітектура, проектування, застосування. Монографія. Львів: Видавництво Львівської політехніки, 2013. – 516 с.
54. Миллер Р. Последовательные и параллельные алгоритмы / Р. Миллер, Л. Боксер – М.:БИНОМ. Лаборатория знаний, 2007. – 960 с.
55. Наконечний А.Й., Наконечний Р.А., Павлиш В.А. Цифрова обробка сигналів. Навчальний посібник. Львів: Видавництво Львівської політехніки, 2010. – 368с.
56. Николайчук Я.М., Возна Н.Я., Пітух Р.І. Проектування спеціалізованих комп’ютерних систем: навч. посіб. – Тернопіль: Терно-граф, 2010. - 392с.
57. Пелед Ф., Лиу Б. Цифровая обработка сигналов: Теория, проектирование, реализация: Пер, с англ. - Киев. : Вища школа. Головное изд-во, 1979. – 264 с.
58. Питерсон Дж. Теория сетей Петри и моделирование систем. – М.: Мир, 1984. – 264с.
59. Поспелов Д.А. Введение в вычислительные системы. Москва: “Советское радио”, 1972.- 323 с.
60. Поспелов Д.А. Логические методы анализа и синтеза схем. М: “Энергия”, 1974.- 368 с.
61. Рабинер Л., Гоулд Б. Теория и применение цифровой обработки сигналов / Пер. с англ. А.Л. Зайцева, Э.Г. Назаренко, Н.Н. Тетекина; Под ред. Ю.Н. Александрова.- М.: Мир, 1978. - 848 с.
62. Рабинович З.Л. Основы теории элементных структур ЭВМ. – М:. Радиосвязь, 1982. – 279 с.
63. Рабинович З.Л., Раманаускас В.А. Типовые операции в вычислительных машинах. – К.: Техника, 1980. – 264 с.

64. Рак Ю.П., Дунець Р.Б. Проектування технологічних ліній оперативної поліграфії: системний підхід. – Дрогобич: НВЦ ”Каменяр” ДДПУ, 2002. – 112с.
65. Савельев А.Я. Прикладная теория цифровых автоматов. – М.: Высш. шк., 1987. – 272 с.
66. Самофалов К.Г., Корнейчук В.И., Тарасенко В.П. Цифровые ЭВМ: Теория и проектирование / Под общ. ред. К.Г. Самофалова – К.: Выща шк., 1989. – 424 с.
67. Самофалов К.Г., Луцкий Г.М. Основы теории многоуровневых конвейерных вычислительных систем. – М.: Радио и связь, 1989. – 272 с.
68. Теслюк В.М., Денисюк П.Ю. Автоматизація проектування мікро електромеханічних систем на компонентному рівні. Монографія. Львів: Видавництво Львівської політехніки, 2011. – 192с.
69. Угрюмов Е.П. Цифровая схемотехника. – СПб: ”БХВ-Петербург”, 2001. – 528 с.
70. Фрумкин М.А. Систолические вычисления. – М:. Наука, 1990. – 191 с.
71. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: Учебник для вузов. – СПб.: Питер, 2007. – 667 с.
72. Цмоць І.Г. Інформаційні технології та спеціалізовані засоби обробки сигналів і зображень у реальному часі. – Львів, 2005. – 227с.
73. Шпаковский Г.И. Архитектура параллельных ЭВМ/ Г.И Шпаковский. — Мн.: Университетское, 1989. — 192 с.
74. Яковлєва І. Д Синтез та оцінка різних варіантів паралельних перемножувачів двійкових чисел / І. Д. Яковлєва // Науковий вісник Чернівецького ун-ту. Фізика. Електроніка. Вип. 426. – Чернівці: Чернівецький національний університет імені Юрія Федьковича, 2008. – С. 33 - 38.
75. Яковлєва І. Д. Оцінка варіантів синтезу паралельних обчислювальних пристроїв сортування / І. Д. Яковлєва // Вісник “Комп’ютерні системи та

мережі". – Львів: Національний університет "Львівська політехніка", 2008. - №630. – С. 124-130.

76. Baugh C.R., Wooley B.A. "A Two's Complement Parallel Array Multiplication Algorithm", IEEE Transactions on Computers, C-22, Dec. 1973, pp. 1045-1047.
77. Biand G., Jones E. V. "A Pipelined FFT Processor for Word-Sequential Data", IEEE Transactions on Acoustics, Speech and Signal Processing. vol.37, n.12, pp.1982-5, 1989.
78. Dadda L. Some Schemes for Parallel Multipliers // Alta Frequenza. – 1965. – v.34. – P.349-356.
79. Dunets R. Multi-functional nanostructured sensors and their adaptation into cyber-physical systems / R. Dunets, H. Klym, R. Kochan // Proceedings of the International Conference on Computer Science and Information Technologies (CSIT 2015), 14-17 September 2015, Lviv, Ukraine. – P. 154-157.
80. Dunets R. Research of the matrix method tasks flow graph algorithm / R. Dunets, V. Gryga // Advanced Computer Systems and Networks: Design and Application. Proceedings of the 6-th International Conference. ACSN-2013. – Lviv, Ukraine, 2013. – P. 118-121.
81. Dunets R. Spatio-temporal synthesis of transformation matrix of reverse fast cosine transformation / R. Dunets, V. Gryga // The Experience of Designing and Application of CAD Systems in Microelectronics. Proceedings of XIIIth International Conference. CADSM'2015. – Lviv-Poljana, Ukraine, 2015. – P. 45-49.
82. Gryga V. Construction of time-space graphs for algorithms of parallel multiplication / V. Gryga // Proceedings of 2nd International Conference of Young Scientists "Computer science and engineering". – Lviv: Lviv Polytechnic National University, 2007. – P. 83-85.
83. Gryga V. Transformation in space-time the parallel algorithm binary numbers multiplication / V. Gryga // Proceedings of 3rd International Conference

"Advanced Computer Systems and Networks: Design and Application". – Lviv: Lviv Polytechnic National University, 2007. – P. 75-77.

84. Jantsch A. Modeling Embedded Systems and SoCs – Concurrency and Time in Models of Computation. Systems on Silicon. Morgan Kaufmann Publisher, June 2003.
85. Klym H. „Cold" crystallization in nanostructurized $80GeSe_2$-$20Ga_2Se_3$ glass / H. Klym, A. Ingram, O. Shpotyuk, L. Calvez, E. Petracovschi, B. Kulyk, R. Serkiz, R. Szatanik // Nanoscale Research Letters. – 2015. – V. 10, No 49. – P. 1-8.
86. Klym H. Investigation of changes in positronium trapping in pores under the water influence in nanostructured MgO-Al2O3 ceramics / Klym H., Kostiv Y., Ivanusa A., Tkachuk T. // Book of abstracts of the 15th Young Researchers' Conference "Materials Science and Engineering" (15YRC), (Belgrade, Serbia, 7-9 December, 2016). – 2017. – P. 38–39.
87. Klym H. Multilayer thick-film structures based on spinel ceramics / H. Klym, I. Hadzaman, A. Ingram, O. Shpotyuk // Canadian Journal of Physics. – 2014. – V. 92(7/8). – P. 822–826.
88. Klym H. Multi-state PAL models for inner free-volume study of sensor ceramics for solid-state electronics / H. Klym, O. Shpotyuk, I. Karbovnyk, R. Kochan // Electronics and information technologies. – 2015.
89. Klym H. Novel temperature/humidity-sensitive multilayer thick-film structures for integrated sensors application / H. Klym, I. Hadzaman, O. Shpotyuk // Proceedings of 10th International Conference on Laser and fiber-optical networks modelling (LFNM'2010), 12-14 September, 2010, Sevastopol, Ukraine. – P. 152-154.
90. Klym H. Temperature and humidity sensitive ceramic materials in thick-film performance for multifunctional sensor application / H. Klym, I. Hadzaman, O/ Shpotyuk, M. Brunner // Proceedings of the 14th International Conference on Sensors, Technologies, Electronic and Applications (SENSOR-TEST 2009), Nurnberg, Germany, 26-28 May, 2009. – V. II. – P. 307-310.

91. Li J.-R. Integral nonlinearity correction of ADC using multi-resistor voltage divider / J.-R. Li, R. Kochan, O. Kochan, H. Klym // Proceedings of the 8-th International Conference on Intelligent Data Acquisition and Advanced Computer Systems: Technology and Applications (IDAACS), 24-26 September 2015, Warsaw, Poland. – P. 767-772.
92. Murthy N.R., Swamy N.M. “On the Real-Time Computation of DFT and DCT through Systolic Architectures”, IEEE Transactions on Signal Pricessing, vol.42, n.4, pp. 988-91, 1994.
93. Pezaris S. D. “A 40-ns 17b by 17b Array Multiplier”, IEEE Transactions on Computers, C-20, Apr. 1971, pp. 442-447.
94. Shpotyuk O. T/RH-sensitive thick-film structures for ecological control and environment monitoring / O. Shpotyuk, I. Hadzaman, H. Klym, M. Brunner // Proceedings of the 27th International Conference on Microelectronics, (MIEL 2010), 16-19 May, 2010, Nis, Serbia. – P. 235-238.
95. Sorokin N. Implementation of high-speed fixed-point dividers on FPGA.// Journal of Computer Science and Technology. – 2006. – 6 . – P. 8-11.
96. Wallace C. S. A Suggestion for Parallel Multipliers // IEEE Trans. Electron Computers. – 1964. – v.13. – P.14-17.
97. Klym H.I. Methodology and algorithm of multicomponent analysis of positron annihilation spectra for nanostructured functional materials / H.I. Klym, A.I. Ivanusa, Yu.M. Kostiv, D.O. Chalyy, T.I. Tkachuk, R.B. Dunets, I.I. Vasylchyshyn // JOURNAL OF NANO- AND ELECTRONIC PHYSICS, v. 9, No 3, 2017
98. Tkachuk T. Two Methods to Determining the Time Complexity of Algorithm // Proceedings of the XIII-th. International Conference “Modern problems of radio engineering, telecommunications, and computer science” TCSET’2016, Slavsko, Ukraine, Feb, 2016, p. 452-454.
99. Ткачук Т.І. Аналіз VHDL-Моделей оптимізованих матричних пристроїв множення // Науково-технічний журнал «Радіоелектронні і комп’ютерні системи» Національного аерокосмічного університету ім. М.Є.

Жуковського, «Харківський авіаційний інститут». – 2016. – №6(80). – С. 136-140.

100. Ткачук Т.І. Порівняння двох швидкодіючих структур / Ткачук Т.І., Черкаський М.В. // Збірник матеріалів IV міжвузівської науково-технічної конференції науково-педагогічних працівників “Проблеми та перспективи розвитку економіки і підприємництва та комп’ютерних технологій в Україні”. – Львів: Видавничий відділ Інституту підприємництва та перспективних технологій, 2009. – с. 230-231.
101. Ткачук Т.І. Розробка управління автономного гоночного робота на базі PSoC фірми CYPRESS / Т.І. Ткачук, М.Ю. Шалений, Д.Р. Наумов // Вісник наукових досліджень: Науковий журнал, №13. - Тернопіль: Редакційно – видавничий відділ Галицького інституту імені В’ячеслава Чорновола, 2010. – с. 169-173.
102. Черкасский Н.В. Параметрическая оптимизация устройства БПФ / Черкасский Н.В., Ткачук Т.И. // Вестник «Физика, Математика, Информатика» Брестского государственного технического университета БрГТУ. – 2013. – №5(83). – С. 22-25.
103. Черкаський М. Ефективний пристрій згортки / М. Черкаський, Т. Ткачук // Вісник «Комп’ютерні науки та інформаційні технології» Національного університету «Львівська політехніка». – 2012. – №732. – С. 66-71.
104. Черкаський М.В. Характеристики складності пристроїв множення / Черкаський М.В., Ткачук Т.І. // Науково-технічний журнал «Радіоелектронні і комп’ютерні системи» Національного аерокосмічного університету ім. М.Є. Жуковського, «Харківський авіаційний інститут». – 2012. – №5(57). – С. 142-147.
105. Cherkaskyy M. Structural synthesis of H-model of FFT / Mykola Cherkaskyy, Taras Tkachuk // Proceedings of the 6-th International Conference “Advanced Computer Systems and Networks: Design and Application” ACSN-2013, Lviv, Ukraine, Sep, 2013, p. 122-124.

106. Giambiasi N., Paillet J. – L., Chane F. From Timed Automata to DEVS Models. Proceedings of the 2003 Winter Simulation Conference, 2003.
107. Kang J. –W., Wey Ch. –L., Fisher P.D. Application of bipartite graphs for achieving race-free state assigments // IEEE Trans. Computers. – 1994. – Vol. 44. - №8. – P.1002-1011.
108. Nikmehr H. Architectures for floating-point division, Ph.D. dissertation, Univ. Adelaide, Adelaide, Australia. – 2005. – 258 p.
109. Ramanathan P., Chalasani S. Resource placement with multiple adjacency constraints in k-ary n-cubes // IEEE Trans. on Parallel and Distributed Systems.- 1995. – Vol. 6. – №5. – P. 511-519.
110. Zadiraka V., Yaroslav Nykolaichuk Y. Methods of effective protection of information flows.- Ternopil: Terno-graf, 2014. -308 p.
111. Image encryption algorithm based on one-dimensional and two-dimensional maps / M.Ya. Kushnir, G.V. Kosovan, O.V. Krulikovskyi // Proceeding of the II International Scientific- Practical Conferences “PREDT -2012”. – Chernivtsi, October 25-27, 2012. – p. 90-91.
112. Hasler M., Maistrenko Yu.L. An introduction to the synchronization of chaotic systems: Coupled skew tent maps // IEEE Trans, on CAS-1, Vol. 44, No. 10, 1997.p.856-866.
113. Z. Ji, Z. Sun and X. Liu, "The Design of Radix-4 FFT by FPGA," 2008 International Symposium on Intelligent Information Technology Application Workshops(IITAW), vol. 00, 2008, pp. 765-768
114. Young Jin Kim, Hyon Soo Lee, "The implementation of 2D FFT using multiple topology on 4 × 4 Torus", Communications and Information Technology 2009. 9th International Symposium, Sept. 2009, pp. 615-620.
115. Liang Gu, Xiaoming Li, Jakob Siegel, "An empirically tuned 2D and 3D FFT library on CUDA GPU", Proceedings of the 24th ACM International Conference on Supercomputing (ICS'10). ACM, 2010, pp. 305-314.

116. W. Wendi, D. Bo, Z. Chunming, Z. Peiheng, S. Ninghui, "Accelerating 2D FFT with non-power-of-two problem size on FPGA", The 2010 International Conference on Reconfigurable Computing and FPGAs, Dec. 2010, pp. 13-15.
117. Brahim Betkaoui, David B. Thomas, Wayne Luk, "Comparing Performance and Energy Efficiency of FPGAs and GPUs for High Productivity Computing", The 2010 International Conference on Field-Programmable Technology (FPT'10*)*, Dec. 2010, pp. 8-11.
118. K.S. Hemmert, K.D. Underwood, "An analysis of the double-precision floating-point FFT on FPGAs", Field-Programmable Custom Computing Machines 2005. FCCM 2005. 13th Annual IEEE Symposium, April 2005, pp. 171-180.
119. F. Franchetti, M. Puschel, Y. Voronenko, S. Chellappa, J.M.F. Moura, "Discrete fourier transform on multicore", Signal Processing Magazine IEEE, vol. 26, no. 6, November 2009, pp. 90-102.

ДОДАТОК 1

(VHDL-код оптимізованих пристроїв спецпроцесора ЦОС)

Конвеєрний матричний пристрій множення з діагональним розповсюдженням переносу:

***Файл Mult.vhdl**:*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity Mult is
	port( clk : in STD_LOGIC; A   : in STD_LOGIC_VECTOR(7 downto 0); B   : in STD_LOGIC_VECT(7 downto 0); rst : in STD_LOGIC; S   : out STD_LOGIC_VECTOR(17 downto 0)               );
end Mult;
architecture Mult of Mult is
signal a_t, b_t : STD_LOGIC_VECTOR (8 downto 0); signal c12, s12 : STD_LOGIC_VECTOR (7 downto 0); signal sr12, fins : STD_LOGIC_VECTOR (8 downto 0); signal tmp : STD_LOGIC_VECTOR (17 downto 0);
component FullAdd_x8 is
	port( S_in  : in STD_LOGIC_VECTOR(7 downto 0); C_in  : in STD_LOGIC_VECTOR(7 downto 0); S_out : out STD_LOGIC_VECTOR(8 downto 0)    );
end component FullAdd_x8;
component Gmatrix_9x9 is
	port( A    : in STD_LOGIC_VECTOR(8 downto 0); B    : in STD_LOGIC_VECTOR(8 downto 0); C    : out STD_LOGIC_VECTOR(7 downto 0); S    : out STD_LOGIC_VECTOR(7 downto 0); S_out : out STD_LOGIC_VECTOR(8 downto 0)    );
end component Gmatrix_9x9;
component reg18bit is
	port( clk   : in STD_LOGIC; rst   : in STD_LOGIC; D_in  : in STD_LOGIC_VECTOR(17 downto 0); D_out : out STD_LOGIC_VECTOR(17 downto 0)    );
end component reg18bit;
component reg9bit is
	port( clk   : in STD_LOGIC; rst   : in STD_LOGIC; D_in  : in STD_LOGIC_VECTOR(8 downto 0); D_out : out STD_LOGIC_VECTOR(8 downto 0)    );
end component reg9bit;
begin
	A1 : reg9bit
	port map( clk => clk, rst =>          rst, D_in => A, D_out => a_t );
	A2 : reg9bit
	port map( clk => clk, rst =>          rst, D_in => B, D_out => b_t );
	A3 : Gmatrix_9x9
	port map( A => a_t, B => b_t, C => c12, S => s12, S_out => sr12 );
	A7 : FullAdd_x8
	port map( S_in => s12, C_in => c12, S_out => fins );
	tmp <= fins & sr12;
	A8 : reg18bit
	port map( clk => clk, rst =>          rst, D_in => tmp, D_out => S );
end Mult;
```

***Файл Gmatrix.vhdl**:*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity Gmatrix_9x9 is
	port(A    : in STD_LOGIC_VECTOR(8 downto 0); B    : in STD_LOGIC_VECTOR(8 downto 0); C    : out STD_LOGIC_VECTOR(7 downto 0); S    : out STD_LOGIC_VECTOR(7 downto 0); S_out : out STD_LOGIC_VECTOR(8 downto 0)    );
end Gmatrix_9x9;
architecture Gmatrix_9x9 of Gmatrix_9x9 is
signal s11, s12, s13, s14, s15, s16, s17, s18 : STD_LOGIC; signal s21, s22, s23, s24, s25, s26, s27, s28 : STD_LOGIC; signal c22, c23, c24, c25, c26, c27, c28, c29 : STD_LOGIC; signal s31, s32, s33, s34, s35, s36, s37, s38 : STD_LOGIC; signal c32, c33, c34, c35, c36, c37, c38, c39 : STD_LOGIC;  signal s41, s42, s43, s44, s45, s46, s47, s48 : STD_LOGIC; signal c42, c43, c44, c45, c46, c47, c48, c49 : STD_LOGIC; signal s51, s52, s53, s54, s55, s56, s57, s58 : STD_LOGIC; signal c52, c53, c54, c55, c56, c57, c58, c59 : STD_LOGIC; signal s61, s62, s63, s64, s65, s66, s67, s68 : STD_LOGIC; signal c62, c63, c64, c65, c66, c67, c68, c69 : STD_LOGIC; signal s71, s72, s73, s74, s75, s76, s77, s78 : STD_LOGIC; signal c72, c73, c74, c75, c76, c77, c78, c79 : STD_LOGIC; signal s81, s82, s83, s84, s85, s86, s87, s88 : STD_LOGIC; signal c82, c83, c84, c85, c86, c87, c88, c89 : STD_LOGIC;c
component gild is
```

```
        port( S_in : in STD_LOGIC; C_in : in STD_LOGIC; S_out : out STD_LOGIC; C_out : out STD_LOGIC;
A_io : in STD_LOGIC; B_io : in STD_LOGIC    );
end component gild;
begin
        G11 : gild
        port map( S_in => '0', C_in => '0', S_out => s11, A_io => A(8), B_io => B(0));
        G12 : gild
        port map( S_in => '0', C_in => '0', S_out => s12, A_io => A(7), B_io => B(0));
        G13 : gild
        port map( S_in => '0', C_in => '0', S_out => s13, A_io => A(6), B_io => B(0));
        G14 : gild
        port map( S_in => '0', C_in => '0', S_out => s14, A_io => A(5), B_io => B(0));
        G15 : gild
        port map( S_in => '0', C_in => '0', S_out => s15, A_io => A(4), B_io => B(0));
        G16 : gild
        port map( S_in => '0', C_in => '0', S_out => s16, A_io => A(3), B_io => B(0));
        G17 : gild
        port map( S_in => '0', C_in => '0', S_out => s17, A_io => A(2), B_io => B(0));
        G18 : gild
        port map( S_in => '0', C_in => '0', S_out => s18, A_io => A(1), B_io => B(0));
        G19 : gild
        port map( S_in => '0', C_in => '0', S_out => S_out(0), A_io => A(0), B_io => B(0) );
        G21 : gild
        port map( S_in => '0', C_in => '0', S_out => s21, A_io => A(8), B_io => B(1));
        G22 : gild
        port map( S_in => s11, C_in => '0', S_out => s22, C_out => c22, A_io => A(7), B_io => B(1)  );
        G23 : gild
        port map( S_in => s12, C_in => '0', S_out => s23, C_out => c23, A_io => A(6), B_io => B(1)  );
        G24 : gild
        port map( S_in => s13, C_in => '0', S_out => s24, C_out => c24, A_io => A(5), B_io => B(1) );
        G25 : gild
        port map( S_in => s14, C_in => '0', S_out => s25, C_out => c25, A_io => A(4), B_io => B(1)  );
        G26 : gild
        port map( S_in => s15, C_in => '0', S_out => s26, C_out => c26, A_io => A(3), B_io => B(1)    );
        G87 : gild
        port map( S_in => s76, C_in => c77, S_out => s87, C_out => c87, A_io => A(2), B_io => B(7)  );
        G88 : gild
        port map( S_in => s77, C_in => c78, S_out => s88, C_out => c88, A_io => A(1), B_io => B(7)   );
        G89 : gild
        port map( S_in => s78, C_in => c79, S_out => S_out(7), C_out => c89, A_io => A(0), B_io => B(7)  );

        G91 : gild
        port map( S_in => '0', C_in => '0', S_out => S(7), A_io => A(8), B_io => B(8));
        G92 : gild
        port map( S_in => s81, C_in => c82, S_out => S(6), C_out => C(7), A_io => A(7), B_io => B(8) );
        G93 : gild
        port map( S_in => s82, C_in => c83, S_out => S(5), C_out => C(6), A_io => A(6), B_io => B(8));
        G94 : gild
        port map( S_in => s83, C_in => c84, S_out => S(4), C_out => C(5), A_io => A(5), B_io => B(8) );
        G95 : gild
        port map( S_in => s84, C_in => c85, S_out => S(3), C_out => C(4), A_io => A(4), B_io => B(8)   );
        G96 : gild
        port map( S_in => s85, C_in => c86, S_out => S(2), C_out => C(3), A_io => A(3), B_io => B(8)   );
        G97 : gild
        port map( S_in => s86, C_in => c87, S_out => S(1), C_out => C(2), A_io => A(2), B_io => B(8)  );
        G98 : gild
        port map( S_in => s87, C_in => c88, S_out => S(0), C_out => C(1), A_io => A(1), B_io => B(8)   );
        G99 : gild
        port map( S_in => s88, C_in => c89, S_out => S_out(8), C_out => C(0), A_io => A(0), B_io => B(8) );
end Gmatrix_9x9;
```

Конвеєрний пристрій Згортки:

***Файл Convolution.vhdl**:*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity Convolution is
	port( clk : in STD_LOGIC; A   : in STD_LOGIC_VECTOR(7 downto 0); B   : in STD_LOGIC_VECTOR(7 downto 0); rst : in STD_LOGIC; S   : out STD_LOGIC_VECTOR(31 downto 0)   );
end Convolution;
architecture Convolution of Convolution is
--first line
signal ar, br : STD_LOGIC_VECTOR(7 downto 0);
--second line
signal m : STD_LOGIC_VECTOR(15 downto 0);
--third line
signal sum : STD_LOGIC_VECTOR(31 downto 0);
--forth line
signal f : STD_LOGIC_VECTOR(31 downto 0);
--components
component reg8bit is
	port( clk   : in STD_LOGIC; rst   : in STD_LOGIC; D_in  : in STD_LOGIC_VECTOR(7 downto 0); D_out : out STD_LOGIC_VECTOR(7 downto 0)    );
end component reg8bit;
component mult is
	port( clk : in STD_LOGIC; A   : in STD_LOGIC_VECTOR(7 downto 0); B   : in STD_LOGIC_VECTOR(7 downto 0); rst : in STD_LOGIC;    S   : out STD_LOGIC_VECTOR(15 downto 0)    );
end component mult;
component addsub is
	port( ADDSUB : in std_logic; A, B : in std_logic_vector(31 downto 0); Q : out std_logic_vector(31 downto 0) );
end component addsub;
component reg32bit is
	port( clk   : in STD_LOGIC; rst   : in STD_LOGIC; D_in  : in STD_LOGIC_VECTOR(31 downto 0); D_out : out STD_LOGIC_VECTOR(31 downto 0)    );
end component reg32bit;
begin
	Z1 : reg8bit port map( clk =>clk, rst =>rst, D_in => A, D_out => ar );
	Z2 : reg8bit port map( clk =>clk, rst =>rst, D_in => B, D_out => br );
	Z3 : mult port map( clk =>clk, rst =>rst, A => ar, B => br, S => m );
	Z4 : addsub port map( ADDSUB => '1', A(0) => m(0), A(1) => m(1), A(2) => m(2), A(3) => m(3), A(4) => m(4), A(5) => m(5), A(6) => m(6), A(7) => m(7), A(8) => m(8), A(9) => m(9), A(10) => m(10), A(11) => m(11), A(12) => m(12), A(13) => m(13), A(14) => m(14), A(15) => m(15), A(16) => '0', A(17) => '0', A(18) => '0', A(19) => '0', A(20) => '0', A(21) => '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25) => '0', A(26) => '0', A(27) => '0', A(28) => '0', A(29) => '0', A(30) => '0', A(31) => '0',  B => f, Q => sum   );
	Z5 : reg32bit port map( clk =>clk, rst =>rst, D_in => sum,  D_out => f  );S <= f;
end Convolution;
```

***Файл FullAdd_x7cell.vhdl**:*

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity FullAdd_x7cell is
	port( S_in : in STD_LOGIC_VECTOR(6 downto 0); C_in : in STD_LOGIC_VECTOR(6 downto 0); S_out : out STD_LOGIC_VECTOR(7 downto 0)  );
end FullAdd_x7cell;
--}} End of automatically maintained section
architecture FullAdd_x7cell of FullAdd_x7cell is
signal c1_2, c2_3, c3_4, c4_5, c5_6, c6_7: STD_LOGIC;
component FullAdd is
	port( A : in STD_LOGIC; B : in STD_LOGIC; C : in STD_LOGIC; S : out STD_LOGIC; P : out STD_LOGIC );
end component FullAdd;
begin
```

```
        A1 : FullAdd
        port map( A => S_in(0), B => C_in(0), C => '0', S => S_out(0), P => c1_2  );
        A2 : FullAdd
        port map( A => S_in(1), B => C_in(1), C => c1_2, S => S_out(1), P => c2_3    );
        A3 : FullAdd
        port map( A => S_in(2), B => C_in(2), C => c2_3, S => S_out(2), P => c3_4    );
        A4 : FullAdd
        port map( A => S_in(3), B => C_in(3), C => c3_4, S => S_out(3), P => c4_5    );
        A5 : FullAdd
        port map( A => S_in(4), B => C_in(4), C => c4_5, S => S_out(4), P => c5_6    );
        A6 : FullAdd
        port map( A => S_in(5), B => C_in(5), C => c5_6, S => S_out(5), P => c6_7     );
        A7 : FullAdd
        port map( A => S_in(6), B => C_in(6), C => c6_7, S => S_out(6), P => S_out(7)    );
         -- enter your statements here --
end FullAdd_x7cell;
```

Файл and_2_x15cell.vhdl:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity and_2_x15cell is port( A : in STD_LOGIC_VECTOR(7 downto 0); B : in STD_LOGIC_VECTOR(7 downto
0); S : out STD_LOGIC_VECTOR(14 downto 0)     );
end and_2_x15cell;
--}} End of automatically maintained section
architecture and_2_x15cell of and_2_x15cell is
component and_2 is port( A : in STD_LOGIC; B : in STD_LOGIC; C : out STD_LOGIC    );
end component and_2;
begin
        A1 : and_2
        port map( A => A(0), B => B(0), C => S(0)    );
        A2 : and_2
        port map( A => A(1), B => B(0), C => S(1)    );
        A3 : and_2
        port map( A => A(2), B => B(0), C => S(2)    );
        A4 : and_2
        port map( A => A(3), B => B(0), C => S(3)    );
        A5 : and_2
        port map( A => A(4), B => B(0), C => S(4)    );
        A6 : and_2
        port map( A => A(5), B => B(0), C => S(5)    );
        A7 : and_2
        port map( A => A(6), B => B(0), C => S(6)    );
        A8 : and_2
        port map( A => A(7), B => B(0), C => S(7)    );
        A9 : and_2
        port map( A => A(7), B => B(1), C => S(8)    );
        A10 : and_2
        port map( A => A(7), B => B(2), C => S(9)    );
        A11 : and_2
        port map( A => A(7), B => B(3), C => S(10)    );
        A12 : and_2
        port map( A => A(7), B => B(4), C => S(11)    );
        A13 : and_2
        port map( A => A(7), B => B(5), C => S(12)    );
        A14 : and_2
        port map( A => A(7), B => B(6), C => S(13)    );
        A15 : and_2
        port map( A => A(7), B => B(7), C => S(14)    );
         -- enter your statements here --
end and_2_x15cell;
```

Конвеєрний пристрій ШПФ:

Файл FFT.vhdl:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity FFT is
        port( clk : in STD_LOGIC; A,B,C,D,W,V   : in STD_LOGIC_VECTOR(7 downto 0); rst : in STD_LOGIC;
A_out, C_out, B_out, D_out   : out STD_LOGIC_VECTOR(31 downto 0)  );
end FFT;
architecture FFT of FFT is
--first line
signal rm1, rm2, rm3, rm4 : STD_LOGIC_VECTOR(7 downto 0);
--second line
signal ms1, ms2, ms3, ms4 : STD_LOGIC_VECTOR(15 downto 0);
--third line
signal sr1, sr2 : STD_LOGIC_VECTOR(31 downto 0);
--forth line
signal rs1, rs2 : STD_LOGIC_VECTOR(7 downto 0);
signal rs3, rs4 : STD_LOGIC_VECTOR(31 downto 0);
--fifth line
signal fsr1, fsr2, fsr3, fsr4 : STD_LOGIC_VECTOR(31 downto 0);
--components
component reg8bit is
        port( clk   : in STD_LOGIC; rst   : in STD_LOGIC; D_in  : in STD_LOGIC_VECTOR(7 downto 0); D_out
: out STD_LOGIC_VECTOR(7 downto 0)    );
end component reg8bit;
component mult is
        port( clk : in STD_LOGIC; A   : in STD_LOGIC_VECTOR(7 downto 0); B   : in STD_LOGIC_VECTOR(7
downto 0); rst : in STD_LOGIC; S   : out STD_LOGIC_VECTOR(15 downto 0)  );
end component mult;
component addsub is
        port( ADDSUB : in std_logic; A, B : in std_logic_vector(31 downto 0); Q : out std_logic_vector(31 downto
0)   );
end component addsub;
component reg32bit is
        port( clk   : in STD_LOGIC; rst   : in STD_LOGIC; D_in  : in STD_LOGIC_VECTOR(31 downto 0);
D_out : out STD_LOGIC_VECTOR(31 downto 0)    );
end component reg32bit;
begin
        Z1 : reg8bit
        port map(  clk => clk, rst =>rst, D_in => C, D_out => rm1  );
        Z2 : reg8bit
        port map( clk => clk, rst =>rst, D_in => D, D_out => rm2   );
        Z3 : reg8bit
        port map( clk => clk, rst =>rst, D_in => W, D_out => rm3    );
        Z4 : reg8bit
        port map( clk => clk, rst =>rst, D_in => V, D_out => rm4    );
        Z5 : mult
        port map( clk => clk, rst =>rst, A => rm1, B => rm3, S => ms1   );
        Z6 : mult
        port map( clk =>clk, rst =>rst, A => rm4, B => rm2, S => ms2    );
        Z7 : mult
        port map( clk =>clk, rst =>rst, A => rm2, B => rm3, S => ms3  );
        Z8 : mult
        port map( clk =>clk, rst =>rst, A => rm1, B => rm4,S => ms4    );
        Z9 : addsub
        port map( ADDSUB => '0', A(0) => ms1(0), A(1) => ms1(1), A(2) => ms1(2), A(3) => ms1(3), A(4) =>
ms1(4), A(5) => ms1(5), A(6) => ms1(6), A(7) => ms1(7), A(8) => ms1(8), A(9) => ms1(9), A(10) => ms1(10),
A(11) => ms1(11), A(12) => ms1(12), A(13) => ms1(13), A(14) => ms1(14), A(15) => ms1(15), A(16) => '0',
A(17) => '0',A(18) => '0', A(19) => '0', A(20) => '0', A(21) => '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25)
=> '0', A(26) => '0', A(27) => '0', A(28) => '0', A(29) => '0', A(30) => '0', A(31) => '0',  B(0) => ms2(0), B(1)
```

```
=> ms2(1), B(2) => ms2(2), B(3) => ms2(3), B(4) => ms2(4), B(5) => ms2(5), B(6) => ms2(6), B(7) => ms2(7),
B(8) => ms2(8), B(9) => ms2(9), B(10) => ms2(10), B(11) => ms2(11), B(12) => ms2(12), B(13) => ms2(13),
B(14) => ms2(14), B(15) => ms2(15), B(16) => '0', B(17) => '0', B(18) => '0', B(19) => '0', B(20) => '0', B(21)
=> '0', B(22) => '0', B(23) => '0', B(24) => '0', B(25) => '0', B(26) => '0', B(27) => '0', B(28) => '0', B(29) =>
'0', B(30) => '0', B(31) => '0', Q => sr1    );
        Z10 : addsub
        port map( ADDSUB => '1', A(0) => ms3(0), A(1) => ms3(1), A(2) => ms3(2), A(3) => ms3(3), A(4) =>
ms3(4), A(5) => ms3(5), A(6) => ms3(6), A(7) => ms3(7), A(8) => ms3(8), A(9) => ms3(9), A(10) => ms3(10),
A(11) => ms3(11), A(12) => ms3(12), A(13) => ms3(13), A(14) => ms3(14), A(15) => ms3(15), A(16) => '0',
A(17) => '0', A(19) => '0' A(20) => '0', A(21) => '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25) => '0', A(26)
=> '0', A(27) => '0', A(28) => '0', A(29) => '0', A(30) => '0', A(31) => '0', B(0) => ms4(0), B(1) => ms4(1), B(2)
=> ms4(2), B(3) => ms4(3), B(4) => ms4(4), B(5) => ms4(5), B(6) => ms4(6), B(7) => ms4(7), B(8) => ms4(8),
B(9) => ms4(9), B(10) => ms4(10), B(11) => ms4(11), B(12) => ms4(12), B(13) => ms4(13), B(14) => ms4(14),
B(15) => ms4(15), B(16) => '0', B(17) => '0', B(18) => '0', B(19) => '0', B(20) => '0', B(21) => '0', B(22) => '0',
B(23) => '0', B(24) => '0', B(25) => '0', B(26) => '0', B(27) => '0', B(28) => '0', B(29) => '0', B(30) => '0', B(31)
=> '0', Q => sr2    );
        Z11 : reg8bit port map( clk =>      clk, rst =>rst, D_in => A, D_out => rs1    );
        Z12 : reg8bit
        port map( clk => clk, rst =>rst, D_in => B, D_out => rs2    );
        Z13 : reg32bit
        port map( clk => clk, rst =>rst, D_in => sr1, D_out => rs3    );
        Z14 : reg32bit
        port map( clk => clk, rst =>rst, D_in => sr2, D_out => rs4    );
        Z15 : addsub
        port map( ADDSUB => '1', A(0) => rs1(0), A(1) => rs1(1), A(2) => rs1(2), A(3) => rs1(3), A(4) =>
rs1(4), A(5) => rs1(5), A(6) => rs1(6), A(7) => rs1(7), A(8) =>  '0', A(9) =>  '0', A(10) => '0', A(11) = '0', A(12)
=> '0', A(13) => '0', A(14) => '0', A(15) => '0', A(16) => '0', A(17) => '0', A(18) => '0', A(19) => '0', A(20) =>
'0', A(21) => '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25) => '0', A(26) => '0', A(27) => '0', A(28) => '0',
A(29) => '0', A(30) => '0', A(31) => '0', B => rs3, Q => fsr1   );
        Z16 : addsub
        port map( ADDSUB => '0',A(0) => rs1(0),A(1) => rs1(1), A(2) => rs1(2), A(3) => rs1(3),A(4) =>
rs1(4), A(5) => rs1(5), A(6) => rs1(6), A(7) => rs1(7), A(8) =>  '0', A(9) =>  '0', A(10) => '0',A(11) => '0', A(12)
=> '0', A(13) => '0', A(14) => '0', A(15) => '0', A(16) => '0', A(17) => '0', A(18) => '0', A(19) => '0', A(20) =>
'0', A(21) => '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25) => '0', A(26) => '0', A(27) => '0', A(28) => '0',
A(29) => '0', A(30) => '0',A(31) => '0', B => rs3, Q => fsr2    );
        Z17 : addsub
        port map( ADDSUB => '1', A(0) => rs2(0), A(1) => rs2(1), A(2) => rs2(2), A(3) => rs2(3), A(4) =>
rs2(4), A(5) => rs2(5), A(6) => rs2(6), A(7) => rs2(7), A(8) =>  '0', A(9) =>  '0', A(10) => '0', A(11) => '0', A(12)
=> '0', A(13) => '0', A(14) => '0', A(15) => '0', A(16) => '0', A(17) => '0', A(18) => '0', A(19) => '0', A(20) =>
'0', A(21) => '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25) => '0', A(26) => '0', A(27) => '0', A(28) => '0',
A(29) => '0', A(30) => '0',A(31) => '0', B => rs4, Q => fsr3    );
        Z18 : addsub
        port map( ADDSUB => '0', A(0) => rs2(0), A(1) => rs2(1), A(2) => rs2(2), A(3) => rs2(3), A(4) => rs2(4),
A(5) => rs2(5), A(6) => rs2(6), A(7) => rs2(7), A(8) =>  '0', A(9) =>  '0', A(10) => '0', A(11) => '0', A(12) => '0',
A(13) => '0', A(14) => '0', A(15) => '0', A(16) => '0', A(17) => '0', A(18) => '0', A(19) => '0', A(20) => '0', A(21)
=> '0', A(22) => '0', A(23) => '0', A(24) => '0', A(25) => '0',A(26) => '0', A(27) => '0', A(28) => '0', A(29) => '0',
A(30) => '0', A(31) => '0',B => rs4, Q => fsr4   );
        Z19 : reg32bit
        port map( clk => clk, rst =>rst, D_in => fsr1, D_out => A_out    );
        Z21 : reg32bit
        port map( clk => clk, rst =>rst, D_in => fsr3,  D_out => B_out    );
        Z22 : reg32bit
        port map( clk => clk, rst =>rst, D_in => fsr4,  D_out => D_out    );
end FFT;
```

Оптимізований конвеєрний пристрій ШПФ розділений на 4 блоки:

Файл FFT.vhdl:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity FFT is
```

```
        port( clk : in STD_LOGIC; A,B,C,D,W,V   : in STD_LOGIC_VECTOR(7 downto 0); rst : in STD_LOGIC;
A_out, B_out, C_out, D_out   : out STD_LOGIC_VECTOR(31 downto 0)    );
end FFT;
architecture FFT of FFT is
--first line
signal m14, m31, m42, m23 : STD_LOGIC_VECTOR(7 downto 0);
--second line
signal s21, s12, s43, s34 : STD_LOGIC_VECTOR(15 downto 0);
--components
component FFT_part is
        port( clk, AB1 : in STD_LOGIC; AB, F, P_mul   : in STD_LOGIC_VECTOR(7 downto 0); P_sum    : in
STD_LOGIC_VECTOR(15 downto 0); rst : in STD_LOGIC;   AS1, AS2 : in std_logic; P_mulOut     : out
LOGIC_VECTOR(7 downto 0); P_sumOut     : out STD_LOGIC_VECTOR(15 downto 0); F_out     : out
STD_LOGIC_VECTOR(31 downto 0)   );
end component FFT_part;
begin
        Z1 : FFT_part
        port map( clk => clk, rst =>rst, AB1 => '1', AS1 => '0', AS2 => '1', AB => A, F => C, P_mul => m31,
P_sum => s21, P_mulOut => m14, P_sumOut => s12, F_out => A_out   );
        Z2 : FFT_part
        port map( clk => clk, rst = rst, AB1 => '0', AS1 => '0', AS2 => '0', AB => A, F => D, P_mul => m42,
P_sum => s12, P_mulOut => m23, P_sumOut => s21, F_out => C_out    );
        Z3 : FFT_part
        port map( clk => clk, rst =>rst, AB1 => '1', AS1 => '1', AS2 => '1', AB => B, F => W, P_mul => m23,
P_sum => s43, P_mulOut => m31, P_sumOut => s34, F_out => B_out    );
        Z4 : FFT_part
        port map( clk => clk, rst =>rst, AB1 => '1', AS1 => '1', AS2 => '0', AB => B, F => V, P_mul => m14,
P_sum => s34, P_mulOut => m42, P_sumOut => s43, F_out => D_out    );
end FFT;
```

ДОДАТОК 2

(акти впровадження результатів дисертації)

ЗАТВЕРДЖУЮ
Директор державного науково-дослідного
підприємства «КОНЕКС», к.т.н., с.н.с.
_____Варіченко Л.В.
«__23__» _____06_____ 2017р.

АКТ

про впровадження результатів дисертаційної роботи Ткачука Тараса Ігоровича, представленої на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.05 – *комп'ютерні системи та компоненти*

Даним актом підтверджую, що результати дисертаційної роботи Ткачука Тараса Ігоровича використано при розробці контрольно-перевірочної апаратури та нестандартного обладнання по темі «Січ-2-1», а також при аналізі варіантів реалізації спеціалізованих комп'ютерних пристроїв у дослідно-конструкторській роботі «Оновлення-24 МР».

Особливо слід відзначити ефективність використання досліджуваних автором методів оптимізації SH-моделей спеціальних функцій, що дають змогу значно скоротити час проектування спеціалізованих комп'ютерних систем на ПЛІС.

Використання запропонованих дисертантом наукових і технічних рішень, при створенні VHDL-моделей алгоритмів, дає змогу підвищити їх швидкодію та надійність.

Начальник відділення _____ В.В. Кондаков

ЗАТВЕРДЖУЮ
Заст. директора ЗЦ УВВЛ
_____ Попадик Д.І.
«___» _____ 2017р.

АКТ

про впровадження результатів дисертації Ткачука Тараса Ігоровича, представленої на здобуття наукового ступеня кандидата технічних наук

Даним актом підтверджую, що основні результати наукових досліджень, отримані при виконанні дисертації Ткачука Тараса Ігоровича, впроваджено Західним центром українського відділення "Міжнародного центру наукової культури – Всесвітня лабораторія" (ЗЦ УВВЛ).

Результати дослідження характеристик складності алгоритмів були застосовані в роботі спеціалізованих програмних систем, зокрема використаний метод оптимізації характеристик складності SH-моделей алгоритмів. Цей метод дає змогу оптимізувати наявні алгоритми пошуку найближчого сервера обробки даних спеціалізованих програмних систем.

Застосування даного методу у формуванні системи моніторингу за станом вод в басейні ріки Західний Буг, стане підставою для використання його в рамках тристоронньої співпраці басейнових рад України, Польщі та Білорусії.

Даний акт не є підставою для проведення фінансових взаєморозрахунків.

Керівник екологічних програм ЗЦ УВВЛ
к.т.н., доцент П. М. Грицишин

"ЗАТВЕРДЖУЮ"
Проректор з наукової роботи
Національного університету
"Львівська політехніка"
професор _____ Чухрай Н.І.
"___" _____ 2017 р.

АКТ

про використання результатів дисертації Ткачука Тараса Ігоровича «Характеристики складності Н-моделей спеціальних функцій і їх використання для оптимізації пристроїв спецпроцесора», представленої на здобуття наукового ступеня кандидата технічних наук при виконанні держбюджетної науково-дослідної роботи ДБ/Наносенсор кафедри спеціалізованих комп'ютерних систем Національного університету "Львівська політехніка"

Комісія у складі – начальника НДЧ, к.т.н., доцента Жук Л.В., завідувача відділу науково-організаційного супроводу наукових досліджень, к.т.н. Лазько Г.В., завідувача кафедри спеціалізованих комп'ютерних систем, д.т.н., професора Дунця Р.Б. та заступника начальника планово-фінансового відділу Чулой Т.М., цим актом підтверджують, що результати дисертації Ткачука Т.І. за темою «Характеристики складності Н-моделей спеціальних функцій і їх використання для оптимізації пристроїв спецпроцесора» використано при виконанні держбюджетної науково-дослідної роботи ДБ/Наносенсор «Наноструктуровані скло-керамічні середовища для високонадійних оптоелектронних та сенсорних застосувань», № держреєстрації 0116U004411.

У результаті теоретичних і практичних досліджень, виконаних Ткачуком Т.І. розроблено теоретичні основи, алгоритми, методи та рекомендації, спрямовані на створення оптимізованих комп'ютерних засобів для імплементації сенсорних структур в складові інтелектуальних інформаційно-вимірювальних систем та цифрової обробки сигналів, зокрема:

- розроблено двоступеневий конвеєрний пристрій множення, з використанням матричного пристрою множення з діагональним розповсюдженням переносу, із затримкою на сходинці конвеєра 3.358нс;
- отримані результати дали змогу розробити алгоритми та науково-обґрунтовані рекомендації для прийняття рішень щодо оптимізації характеристик складності спеціалізованих комп'ютерних систем.

Голова комісії

Начальник НДЧ,
к.т.н., доцент _____ Жук Л.В.

Завідувач відділу наково-організаційного
супроводу наукових досліджень, к.т.н. _____ Лазько Г.В.

Завідувач кафедри спеціалізованих
комп'ютерних систем, д.т.н., проф. _____ Дунець Р.Б.

Заст. начальника планово-фінансового відділу _____ Чулой Т.М.

“ЗАТВЕРДЖУЮ”

Проректор з наукової роботи
Національного університету
“Львівська політехніка”
професор
_____ Чухрай Н.І.
“26” ____12_____ 2017 р.

АКТ

про використання результатів дисертації Ткачука Тараса Ігоровича «Характеристики складності Н-моделей спеціальних функцій і їх використання для оптимізації пристроїв спецпроцесора», представленої на здобуття наукового ступеня кандидата технічних наук при виконанні науково-дослідної роботи в рамках гранту Президента України “Модифіковані функціональні середовища на основі наноструктурованих стекол та кераміки для широких приладних застосувань” кафедри спеціалізованих комп’ютерних систем Національного університету “Львівська політехніка”

Комісія у складі – начальника НДЧ, к.т.н., доцента Жук Л.В., завідувача відділу науково-організаційного супроводу наукових досліджень, к.т.н. Лазько Г.В., завідувача кафедри спеціалізованих комп’ютерних систем, д.т.н., професора Дунця Р.Б. та заступника начальника планово-фінансового відділу Чулой Т.М., цим актом підтверджують, що результати дисертації Ткачука Т.І. за темою «Характеристики складності Н-моделей спеціальних функцій і їх використання для оптимізації пристроїв спецпроцесора» використано при виконанні науково-дослідної роботи в рамках гранту Президента України “Модифіковані функціональні середовища на основі наноструктурованих стекол та кераміки для широких приладних застосувань”, № держреєстрації 0117U007181.

У результаті дослідження характеристик складності для пристроїв спецпроцесора, виконаних Ткачуком Т.І. розроблено теоретичні основи, алгоритми, методи та рекомендації, спрямовані на модифікацію та оптимізацію інформаційно-вимірювальних засобів для дослідження структурних особливостей сенсорних скло-керамічних матеріалів, зокрема:

- розроблено оптимізовану Н-модель конвеєрного пристрою обчислення алгоритму «Швидкого Перетворення Фур’є», шляхом розділення пристрою на 4 паралельні вітки, що дозволило значно покращити значення його характеристик складності;
- отримані результати дали змогу розробити алгоритми та науково-обґрунтовані рекомендації для прийняття рішень щодо оптимізації характеристик складності спеціалізованих інформаційно-вимірювальних систем.

Голова комісії

Начальник НДЧ,
к.т.н., доцент _____ Жук Л.В.

Завідувач відділу наково-організаційного
супроводу наукових досліджень, к.т.н. _____ Лазько Г.В.

Завідувач кафедри спеціалізованих
комп’ютерних систем, д.т.н., проф. _____ Дунець Р.Б.

Заст. начальника планово-фінансового відділу _____ Чулой Т.М.

“ЗАТВЕРДЖУЮ”
Проректор
з науково-педагогічної роботи
Національного університету
“Львівська політехніка”
_____ доц. Давидчак О.Р.
“___” _____ 2017 р.

АКТ
про впровадження у навчальний процес
у Національному університеті “Львівська політехніка”
результатів дисертаційної роботи на здобуття наукового ступеня
кандидата технічних наук Ткачука Тараса Ігоровича

Комісія у складі зав. кафедри спеціалізованих комп’ютерних систем д.т.н., проф. Дунця Р.Б., д.т.н., проф. Кочана Р.В. та д.т.н., доц. Клим Г.І. розглянула питання щодо впровадження результатів дисертаційної роботи на здобуття наукового ступеня кандидата технічних наук Ткачука Т.І. у навчальний процес Національного університету “Львівська політехніка”.

Даний акт складений про те, що в навчальному процесі на кафедрі спеціалізованих комп’ютерних систем впроваджені результати дисертаційної роботи Ткачука Т.І. щодо дослідження характеристик складності SH-моделей алгоритмів спеціальних функцій, а також проектування VHDL-моделей оптимізованих пристроїв виконання арифметичних операцій та пристроїв спеціальних функцій спецпроцесора.

Результати роботи використовуються в лекційних та лабораторних заняттях спеціальностей 7.05010203 та 8.05010203 “Спеціалізовані комп’ютерні системи” для дисциплін “Архітектура спеціалізованих комп’ютерних систем”, “Технології проектування комп’ютерних систем” та “Дослідження і проектування контролерів периферійних пристроїв”.

Голова комісії
Завідувач кафедри СКС
д.т.н., проф. Дунець Р.Б.

Члени комісії:

д.т.н., проф. Кочан Р.В.

д.т.н., доц. Клим Г.І.