COMPUTER DESIGN SYSTEMS. THEORY AND PRACTICE

Vol. 7, No. 1, 2025

Олег Новосад[1], Сергій Щербовських[2]

[1] Кафедра систем автоматизованого проектування, Національний університет «Львівська політехніка», вул. С. Бандери, Львів, Україна, E-mail: oleh.v.novosad@lpnu.ua, ORCID 0009-0003-6590-457X
[2] Кафедра систем автоматизованого проектування, Національний університет «Львівська політехніка», вул. С. Бандери, Львів, Україна, E-mail: serhiy.v.shcherbovskykh@lpnu.ua, ORCID 0000-0001-8535-733X

ЗНИЖЕННЯ ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ ФІЛЬТРІВ З КІНЦЕВОЮ ІМПУЛЬСНОЮ ХАРАКТЕРИСТИКОЮ В ІНТЕГРОВАНИХ СИСТЕМАХ

Отримано: Лютий 20, 2025 / Переглянуто: Лютий 25, 2025 / Прийнято: Лютий 28, 2025

https://doi.org/

Анотація. Проблема оптимізації цифрових фільтрів з кінцевою імпульсною характеристикою (КІХ) є актуальною в умовах обмежених обчислювальних ресурсів вбудованих систем. Існуючі методи реалізації КІХ-фільтрів часто призводять до значних витрат енергії та часу обробки, що обмежує їхнє застосування в реальних умовах. Метою роботи є розробка методів оптимізації КІХ-фільтрів для зниження обчислювальної складності та забезпечення ефективної роботи на інтегрованих системах з обмеженими ресурсами. Методологія досліджень включає аналіз структурно оптимізованих схем фільтрів, зокрема багатоступеневих і блочних конфігурацій, а також використання алгоритмів скорочення розрядності коефіцієнтів. Результати дослідження показують, що запропоновані підходи зменшують обчислювальні витрати та енергоспоживання, не втрачаючи якості фільтрації. Новизна роботи полягає у поєднанні методів структурної оптимізації з алгоритмічними прийомами, що враховують особливості апаратної реалізації на вбудованих системах. Практичне значення роботи полягає в можливості застосування оптимізованих фільтрів для обробки сигналів у режимі реального часу на пристроях з обмеженими ресурсами. Напрями подальших досліджень передбачають дослідження ефективності методів оптимізації на різних платформах, таких як FPGA та мікроконтролери, а також адаптацію під різні типи сигналів.

Ключові слова: цифрова обробка сигналів, блочна реалізація фільтрів, оптимізація обчислювальної складності, енергоефективність, адаптивні фільтри

Вступ

В умовах стрімкого розвитку вбудованих систем, які часто використовуються для обробки сигналів у реальному часі, ефективність цифрових фільтрів з кінцевою імпульсною характеристикою набуває особливого значення. КІХ-фільтри знаходять широке застосування в системах обробки зображень, аудіо- та радіосигналів, а також у пристроях Інтернету речей (IoT) і медичних приладах. Проте використання стандартних методів фільтрації часто не є оптимальним для пристроїв з обмеженими обчислювальними ресурсами та енергетичними можливостями.

Актуальність дослідження полягає в необхідності розробки оптимізованих алгоритмів для фільтрів КІХ, що дозволяють значно зменшити обчислювальні витрати, зберігаючи при цьому високу якість фільтрації. Проблема полягає у тому, що традиційні методи обробки сигналів потребують значної кількості обчислень, що може призвести до перевантаження вбудованих систем та скорочення часу автономної роботи.

Метою цієї роботи є розробка методів оптимізації КІХ-фільтрів для інтегрованих систем, які забезпечать зниження обчислювальної складності та ефективне використання ресурсів. У роботі передбачається поєднання структурних підходів, таких як багатоступеневі та блочні фільтри, з алгоритмами скорочення розрядності коефіцієнтів. Результати досліджень можуть бути використані для підвищення продуктивності систем обробки сигналів у різних сферах, таких як портативні медичні пристрої, бездротові сенсори та аудіо обладнання.

Таким чином, оптимізація КІХ-фільтрів є важливим завданням, спрямованим на вдосконалення інтегрованих систем та підвищення їхньої продуктивності в умовах обмежених ресурсів.

Огляд сучасних джерел інформації за тематикою публікації

У [1] статті представлено ефективний підхід до проектування КІХ-фільтра першого типу з використанням алгоритму оптимізації Grasshopper (GOA) та його реалізацію на FPGA. GOA продемонстрував високу ефективність порівняно з іншими алгоритмами, такими як Parks McClellan і Sine Cosine, завдяки кращому загасання стоп-смуги та меншим пульсаціям у смузі пропускання. Також розглянуто апаратну реалізацію запропонованої концепції на FPGA платформі.

У [2] статті представлено гібридну віконну технологію для безкінечною імпульсною характеристикою фільтрів, яка демонструє кращі характеристики, зокрема мінімальний пік бічної частоти і більше загасання, порівняно з традиційними вікнами, такими як Rectangular, Hanning, Hamming і Kaiser. Запропоноване вікно забезпечує значне зниження бічних частот до –72,7 дБ з піковою амплітудою -98 дБ при N=35. Крім того, використано це вікно для проектування низькочастотного фільтра, що підтверджує його ефективність та простоту.

У [3] статті розглянуто важливість цифрових фільтри, зокрема КІХ-фільтри, є важливим елементом у сучасній обробці сигналів і знаходять застосування в різних сферах, таких як фільтрація зображень і модуляція частоти. Для проектування оптимальних БІХ-фільтрів використовуються різноманітні методи оптимізації, що забезпечують покращені результати з точки зору коефіцієнтів фільтра і швидкої збіжності. У статті також розглядаються відомі алгоритми оптимізації, порівнюючи їх за характеристиками пульсацій і загасання в смузі зупинки.

У [4] статті проведено порівняльне дослідження двійкової та CSD форм представлення коефіцієнтів фільтра для проектування КІХ-фільтра без мультиплікаторів. Використано алгоритм пошуку зозулі (CSA), що мінімізує середньоквадратичну похибку між спроектованим фільтром та його ідеальною характеристикою. Дослідження показало, що CSD-представлення забезпечує мінімальну кількість суматорів і оптимальний знак показника степеня двох доданків (SPT).

У [5] розглядаються два методи оптимізації мурашиних колоній для проектування фільтрів з нескінченною імпульсною характеристикою (НІХ). Алгоритми мурашиних колоній використовуються для розв'язання складних оптимізаційних задач, зокрема для мінімізації помилок нелінійної і мультимодальної поверхні НІХ-фільтрів. Порівнюється продуктивність двох варіантів цих алгоритмів для оптимального проектування фільтрів ІКХ.

У [6] статті представлено новий підхід до адаптивного фільтра КІХ з використанням блочного методу найменших квадратів та двійкового кодування зі зсувом, що базується на розподіленій арифметиці. Запропонована схема використовує таблицю пошуку для зменшення складності та нову структуру для оновлення ваг, що призводить до зниження площі, енергоспоживання та збільшення пропускної здатності. Результати синтезу ASIC показують значне покращення в порівнянні з існуючими рішеннями.

У [7] статті змодельовано модуль фільтра швидкої найменшої квадратичної різниці (FLMS) для усунення акустичного шуму в голосовому зв'язку з використанням цифрової обробки сигналів у середовищі Simulink. Результати імітаційного моделювання підтверджують ефективність застосування адаптивного алгоритму FLMS для шумозаглушення, порівняно з іншими алгоритмами, такими як NLMS та RLS. Окрім цього, підкреслено переваги використання перетворення Фур'є для швидших обчислень у частотній області.

У [8] статті розглядається оптимізація фільтрів зі скінченною імпульсною характеристикою (КІХ) за допомогою рекурсивних алгоритмів, таких як RAG-N і BHM, з акцентом на квантування коефіцієнтів фільтрів. Досліджено вплив 8-, 10- та 12-розрядного квантування на ефективність фільтрів, а також побудовано імітаційну модель у Simulink. Результати показують, що хоча оптимізація зменшує використання апаратних ресурсів, час обчислень може збільшуватися з ростом кількості бітів квантування, що потребує подальших досліджень для поліпшення продуктивності фільтрів.

Постановка проблеми

Оптимізація цифрових фільтрів з кінцевою імпульсною характеристикою для вбудованих систем з обмеженими ресурсами є складним завданням через високу обчислювальну складність, зумовлену особливостями їхньої реалізації. На рис. 1 представлено схему цифрового КІХ фільтра.

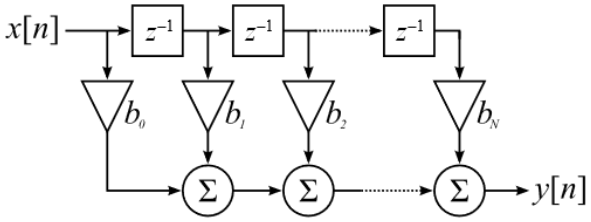

Рис. 1. Схема цифрового КІХ фільтру

При обробці сигналів за допомогою КІХ-фільтрів кожен вихідний зразок сигналу є результатом операції згортки вхідного сигналу з набором коефіцієнтів фільтра. Згортка в її стандартній формі описується наступним чином (1):

$$y[n] = \sum_{k=0}^{M-1} h[k] \cdot x[n-k], \tag{1}$$

де *y[n]* – вихідний сигнал, *x[n]* – вхідний сигнал, *h[k]* – коефіцієнти фільтра, а *M* – довжина фільтра. Ця формула передбачає виконання *M* множень і *M – 1* додавань для кожного нового значення *y[n]*, що створює високе навантаження на обчислювальні ресурси.

Основною проблемою є те, що із зростанням точності фільтра необхідно збільшувати *M*, що призводить до ще більшої обчислювальної складності, особливо на вбудованих платформах, таких як мікроконтролери та системи на кристалі (SoC), які часто мають обмежену кількість обчислювальних блоків і обмежений доступ до пам'яті. Для вирішення цієї проблеми необхідно розробити методи оптимізації структури КІХ-фільтрів, які знижують кількість обчислень без втрати якості обробки сигналу.

Один із підходів для зменшення кількості операцій — це багатоступенева структура фільтра, яка розбиває згортку на кілька етапів. Наприклад, при використанні багатоступеневого КІХ-фільтра загальна кількість обчислень може бути зменшена до (2):

$$y[n] = \sum_{i=0}^{L-1} \left(\sum_{j=1}^{M_i-1} h_i[j] \cdot x[n-1] \right), \tag{2}$$

де *L* – кількість рівнів багатоступеневого фільтра, а M_i – кількість коефіцієнтів на кожному рівні. Це дозволяє зменшити обчислювальні витрати за рахунок використання меншої кількості коефіцієнтів на кожному рівні.

Таким чином, проблема полягає у створенні методів для зниження обчислювальної складності реалізації КІХ-фільтрів, зберігаючи при цьому якість фільтрації на достатньому рівні для застосування у вбудованих системах.

Виклад основного матеріалу

Обчислювальна складність цифрових фільтрів з кінцевою імпульсною характеристикою є основним обмеженням для використання таких фільтрів у вбудованих системах з обмеженими ресурсами. КІХ-фільтри вимагають значної кількості обчислень, що зростає зі збільшенням точності фільтрації та кількості коефіцієнтів. Для більш детального розуміння цієї проблеми розглянемо базову структуру КІХ-фільтра та її обчислювальну складність. Посилаючись до формули (1), цифровий КІХ-фільтр функціонує шляхом згортки вхідного сигналу $x[n]$ з набором фіксованих коефіцієнтів $h[k]$, які визначають його частотні характеристики, де M – довжина фільтра, що дорівнює числу коефіцієнтів $h[k]$. Для кожного нового значення $y[n]$ потрібне виконання M множень і $M-1$ додавань. Таким чином, обчислювальна складність КІХ-фільтра для одного зразка вихідного сигналу пропорційна $O(M)$. У випадках, коли фільтр має високу роздільну здатність і потребує великого M, кількість операцій значно зростає.

Збільшення довжини фільтра M дозволяє досягти кращої селективності фільтра та забезпечити точніше придушення небажаних частотних компонентів. Проте для кожного збільшення M кількість обчислювальних операцій також зростає, що створює проблеми для вбудованих систем, де обмежені обчислювальні ресурси та енергоспоживання. Зокрема, в реальних додатках таких як портативні медичні прилади, пристрої Інтернету речей (IoT), бездротові сенсорні мережі, енергоспоживання та обмеження на використання процесорного часу є критичними параметрами.

У вбудованих системах зазвичай застосовуються процесори з низьким рівнем обчислювальної потужності, такі як мікроконтролери або спеціалізовані процесори для цифрової обробки сигналів (DSP). Ці пристрої мають обмежену кількість операцій з плаваючою комою або взагалі використовують лише цілочисельну арифметику. Також доступ до оперативної пам'яті у таких системах є обмеженим, що ще більше знижує можливості зберігання та обробки великих обсягів даних.

У результаті, якщо КІХ-фільтр має велику кількість коефіцієнтів M, це може спричинити такі проблеми:

- *Затримка обробки сигналу:* Збільшення часу обробки кожного зразка призводить до затримок, що є небажаним для додатків реального часу.
- *Збільшене енергоспоживання:* Для кожного нового значення вихідного сигналу виконуються M множень і $M-1$ додавань, що вимагає додаткової енергії, знижуючи час автономної роботи пристрою.
- *Перевантаження обчислювальних ресурсів:* Виконання великої кількості обчислень може перевищити обчислювальні можливості процесора, що призводить до падіння продуктивності та навіть до можливих збоїв.

Розглянемо приклад фільтра високого порядку для аудіо сигналів з довжиною фільтра $M=1024$. Для кожного нового вихідного зразка $y[n]$ потрібно виконати 1024 множення і 1023 додавання. Якщо частота дискретизації сигналу становить 44,1 кГц, як в аудіосистемах, то кожної секунди потрібно виконати (3):

$$44100\times(1023 \text{ множень}+1023 \text{ додавань}) = 90278700 \text{ операцій за секунду.} \quad (3)$$

Таке навантаження є значним для вбудованих систем з обмеженими обчислювальними можливостями, і його зменшення є важливим завданням. Для вирішення проблеми обчислювальної складності КІХ-фільтрів використовуються різні підходи, включаючи структурні та алгоритмічні оптимізації. Серед них основними є:

1. *Багатоступеневі структури фільтрів*: Каскадне розбиття фільтра на декілька рівнів дозволяє зменшити обсяг операцій за рахунок використання меншої кількості коефіцієнтів на кожному етапі.

2. *Блочна реалізація фільтрів:* Замість обробки кожного окремого зразка сигналу, блочний підхід дозволяє обробляти сигнал блоками, що зменшує загальну кількість операцій.
3. *Швидка згортка на основі швидкого перетворення Фур'є (FFT):* Замість прямої згортки використовується FFT, яка знижує складність операції з $O(M^2)$ до $O(M \log M)$.
4. *Скорочення розрядності коефіцієнтів:* Зменшення кількості бітів для представлення коефіцієнтів фільтра також зменшує кількість необхідних операцій та енергоспоживання, зберігаючи при цьому достатню якість фільтрації.

Багатоступеневі структури КІХ-фільтрів – це підхід, що дозволяє зменшити обчислювальні витрати шляхом поділу фільтра на кілька каскадних етапів. Цей метод особливо корисний для вбудованих систем, де важливо мінімізувати обчислювальні ресурси і зберігати високу якість обробки сигналу. Розглянемо основи багатоступеневих фільтрів, їх переваги та математичний апарат, що описує цей підхід. Класичний КІХ-фільтр застосовує всі коефіцієнти одночасно для розрахунку вихідного сигналу $y[n]$, що вимагає значної кількості обчислювальних операцій. У багатоступеневій структурі цей процес поділяється на кілька етапів або каскадів, кожен із яких виконує часткову фільтрацію сигналу. Кінцевий результат фільтрації досягається шляхом послідовного застосування цих етапів. Формально, багатоступеневий КІХ-фільтр можна описати як комбінацію кількох коротших фільтрів. Наприклад, для сигналу $x[n]$, вихідний сигнал $y[n]$ багатоступеневого фільтра з L етапів можна представити як (4):

$$y[n] = y_L[n] = f_L(f_{L-1}(\dots f_1(x[n])\dots)), \tag{4}$$

де f_i - це оператор фільтрації для iii-го етапу. Кожен каскад обробляє сигнал за своїм набором коефіцієнтів, що забезпечує необхідні частотні характеристики в результаті.

Розглянемо багатоступеневий КІХ-фільтр із двома каскадами як найпростішу модель. Нехай фільтр складається з двох етапів, на кожному з яких виконується згортка з меншим числом коефіцієнтів M_1 і M_2 відповідно. Тоді вихідний сигнал $y[n]$ можна представити як (5):

$$y[n] = \sum_{k=0}^{M_2-1} h_2[k] \cdot \left(\sum_{j=0}^{M_1-1} h_1[j] \cdot x[n-j-k] \right), \tag{5}$$

де $h_1[j]$ та $h_2[k]$ – коефіцієнти фільтрів першого та другого етапів відповідно.

Кожен каскад обробляє сигнал із меншою кількістю коефіцієнтів, що значно зменшує обчислювальні витрати в порівнянні з одно етапним фільтром такої ж загальної довжини. $M = M_1 * M_2$. Це дозволяє досягти майже тих самих частотних характеристик, але з меншими обчислювальними ресурсами. Переваги багатоступеневих структур:

1. *Зменшення обчислювальної складності:* Поділ фільтра на кілька каскадів дозволяє використовувати менше коефіцієнтів на кожному етапі, тим самим скорочуючи кількість множень і додавань, необхідних для розрахунку вихідного сигналу.
2. *Гнучкість у виборі частотних характеристик:* Багатоступеневі структури дозволяють налаштовувати кожен етап на різні частотні діапазони, що може покращити селективність фільтра.
3. *Зменшення енергоспоживання:* Завдяки меншій кількості обчислень, багатоступеневі фільтри є менш енергоємними, що особливо важливо для портативних пристроїв та пристроїв IoT.
4. *Можливість апаратної реалізації:* Завдяки каскадній структурі, кожен етап фільтра можна реалізувати окремо на апаратному рівні, що забезпечує можливість паралельної обробки сигналу, особливо на FPGA або DSP.

Припустимо, що необхідно розробити низькочастотний КІХ-фільтр з роздільною здатністю, що вимагає $M = 64$ коефіцієнтів. Якщо застосувати одноступеневий фільтр, потрібно виконати 64 множення і 63 додавання для кожного нового зразка. Однак за допомогою двоступеневої структури можна розділити фільтр на два каскади з меншою кількістю коефіцієнтів, наприклад, $M_1 = 8$ і $M_2 = 8$. Тепер обчислення кожного каскаду вимагатиме (6):

8 множень і 7 додавань на кожному етапі, всього 2×8 = 16 множень і 14 додавань. (6)

Таким чином, багатоступенева структура дозволяє зменшити кількість множень і додавань з 64 до 16 і 63 до 14 відповідно, що є суттєвим зменшенням обчислювальних витрат.

У вбудованих системах, таких як системи керування, аудіо обробка та портативні медичні пристрої, багатоступеневі фільтри дозволяють досягти необхідної якості обробки сигналу з мінімальним навантаженням на процесор. Наприклад, у системах моніторингу здоров'я багатоступеневі фільтри можуть бути використані для придушення шумів, що накладаються на сигнал серцевого ритму, забезпечуючи при цьому швидку обробку даних із сенсорів без надмірного навантаження на обчислювальні ресурси.

Блочна реалізація фільтрів - це підхід до обробки сигналів, який дозволяє зменшити кількість обчислювальних операцій, групуючи обробку даних у блоки. Така реалізація особливо ефективна для вбудованих систем, де обмежені обчислювальні ресурси, пам'ять та енергоспоживання. Основний принцип блочної реалізації полягає в обробці декількох зразків сигналу одночасно, що дозволяє використовувати оптимізовані алгоритми для більшої обчислювальної ефективності. У класичному КІХ-фільтрі обробка сигналу виконується покроково - для кожного нового зразка сигналу *x[n]* обчислюється відповідне значення вихідного сигналу *y[n]*. В блочній реалізації обробка сигналу виконується блоками, коли замість одного значення *y[n]* обчислюється одразу кілька вихідних значень для блоку зразків. Якщо позначити розмір блоку як *B*, то блочний КІХ-фільтр працює за формулою (1). Де *n* - індекс зразка у блоці, *M* - довжина фільтра, а *h[k]* - коефіцієнти фільтра. У блочній реалізації обчислюються значення вихідного сигналу для *B* зразків, що зменшує потребу в обчисленнях за рахунок повторного використання проміжних результатів.

Розглянемо фільтр, що обробляє сигнал у блоках розміром BBB. Припустимо, що вхідний сигнал розбито на блоки $x_0, x_1, \ldots x_{B-1}$. Для кожного блоку виконується згортка з коефіцієнтами фільтра. Вихідний сигнал для кожного блоку y_b можна описати як згортку вхідного блоку з коефіцієнтами фільтра (7):

$$y_b[n] = \sum_{k=0}^{M-1} h[k] \cdot x_{b-n}[k], \tag{7}$$

де b – номер блоку, n – індекс всередині блоку, а x_{b-n} – зразки вхідного сигналу, що належать поточному блоку. Переваги блочної реалізації фільтрів:

1. *Ефективність для обробки великих обсягів даних:* Блочний підхід дозволяє швидко обробляти великі масиви даних, що важливо для вбудованих систем, які працюють у режимі реального часу.
2. *Зменшення енергоспоживання:* Завдяки зниженню обчислювальної складності, блочна реалізація зменшує енергоспоживання, що особливо важливо для портативних і автономних пристроїв.
3. *Сумісність із паралельними обчисленнями:* Блочний підхід природно підтримує паралельну обробку, оскільки кожен блок можна обробляти незалежно. Це дозволяє оптимізувати роботу на процесорах із багатьма ядрами, GPU або FPGA.

Недоліки блочної реалізації фільтрів:

1. *Затримка обробки:* Потреба в накопиченні блоку даних перед обробкою створює затримку, що може бути критичним для систем реального часу.
2. *Проміжні спотворення:* При обробці великих блоків можуть виникати спотворення на межах блоків, що потребує спеціальних технік для уникнення цього, таких як накладення блоків або застосування віконних функцій.

Уявімо систему, де сигнал надходить із частотою дискретизації 44,1 кГц (типова частота для аудіо сигналів). Для обробки кожного зразка окремо потрібна висока обчислювальна потужність, але якщо використовувати блочну обробку з розміром блоку *B=1024*, можна обробляти сигнал із суттєвим зниженням обчислювальних витрат, застосовуючи FFT для блочної згортки. Таким чином, для кожного блоку виконуються операції швидкого перетворення Фур'є, що дозволяє зменшити

кількість операцій порівняно з послідовною обробкою кожного зразка окремо. Цей метод використовується в аудіо обробці, де необхідна висока швидкість обробки та низьке енергоспоживання.

Блочна реалізація фільтрів є потужним методом для зниження обчислювальних витрат у системах із великим обсягом оброблюваних даних. Цей підхід дозволяє досягти високої обчислювальної ефективності, забезпечуючи при цьому мінімальне енергоспоживання та ефективне використання пам'яті, що робить його ідеальним для вбудованих систем та застосувань у реальному часі.

Швидка згортка на основі швидкого перетворення Фур'є (FFT) є методом обчислення згортки сигналів, що значно знижує обчислювальну складність, використовуючи властивості спектрального перетворення. Завдяки застосуванню FFT, процес згортки, який вимагає великої кількості множень та додавань у часовій області, перетворюється на простішу операцію множення у частотній області. Цей підхід є особливо корисним для обробки сигналів у вбудованих системах, де важливо знизити обчислювальні витрати. Згортка двох сигналів у часовій області $x[n]$ і $h[n]$, що представляє вхідний сигнал і імпульсну характеристику фільтра відповідно, математично описується як (8):

$$y[n] = (x \cdot h)[n] = \sum_{k=0}^{N-1} x[k] \cdot h[n-k], \tag{8}$$

де N - довжина оброблюваного сигналу, а $y[n]$ - результат згортки. Пряме обчислення згортки для великих N є обчислювально витратним, оскільки потребує $O(N^2)$ операцій. Однак, згідно з теоремою згортки, згортку в часовій області можна виконати шляхом множення спектрів у частотній області (9):

$$y[n] = \mathcal{F}^{-1}\{\mathcal{F}(x[n]) \cdot \mathcal{F}(h[n])\}, \tag{9}$$

де F та F^{-1} позначають пряме та обернене перетворення Фур'є. У такому вигляді згортка зводиться до трьох основних етапів:

1. Пряме перетворення Фур'є $x[n]$ і $h[n]$.
2. Множення отриманих спектрів.
3. Обернене перетворення Фур'є для отримання результату в часовій області.

Використання FFT для обчислення прямих та обернених перетворень значно зменшує обчислювальну складність до $O(N \log N)$, що є суттєвим покращенням порівняно з $O(N^2)$ для класичної згортки.

Використання FFT у згортці особливо ефективне для довгих сигналів та фільтрів, оскільки зменшує кількість операцій до $O(N \log N)$. Це значно покращує продуктивність системи і зменшує час обробки.

Прикладом покращення є розрахунок згортки для сигналу з $N=1024$ зразків. Пряма згортка потребувала б близько 1024^2=1,048,576 операцій, тоді як згортка за допомогою FFT потребує лише близько $1024 \log_2 1024 = 10{,}240$ операцій, що суттєво знижує обчислювальні витрати.

Метод Overlap-Add (перекриття-накладання) використовується, коли сигнал настільки довгий, що одночасна обробка всього сигналу за допомогою FFT є неефективною. Суть методу:

1. Сигнал розбивається на блоки з перекриттям. Наприклад, при згортці сигналу з фільтром довжини M блоки обираються так, щоб їхня довжина була не менше ніж M.
2. Кожен блок обробляється за допомогою швидкої згортки окремо, і результат для кожного блоку додається, перекриваючись із сусідніми блоками.
3. Завдяки такому перекриттю уникнути накладання спотворень на межах блоків, що дозволяє отримати точну згортку для всього сигналу.

Переваги швидкої згортки на основі FFT

1. *Значне зменшення обчислювальних витрат:* Зниження складності з $O(N^2)$ до $O(N \log N)$ дозволяє ефективно обробляти довгі сигнали та великі фільтри.

2. *Придатність для обробки в реальному часі:* Методи на основі FFT підходять для режиму реального часу завдяки можливості обробки даних блоками, що дозволяє знижувати затримки.

3. *Оптимізація апаратної реалізації:* Алгоритми FFT добре підходять для реалізації на спеціалізованих апаратних платформах, таких як DSP або FPGA, що дозволяє суттєво знизити енергоспоживання вбудованих систем.

4. *Гнучкість у виборі алгоритму:* Залежно від типу сигналу та системних вимог можна використовувати різні варіанти швидкої згортки, такі як прямолінійна згортка, циклічна згортка, Overlap-Add або Overlap-Save.

Швидка згортка на основі швидкого перетворення Фур'є є потужним інструментом для зниження обчислювальної складності в задачах обробки сигналів, зокрема для реалізації довгих КІХ-фільтрів. Цей підхід забезпечує високу обчислювальну ефективність, низьке енергоспоживання та дозволяє реалізувати складні фільтри на обмежених апаратних ресурсах.

У даній роботі було прийнято рішення використовувати блочну реалізацію фільтрів з огляду на її численні переваги, які можуть істотно покращити продуктивність вбудованих систем. Такий підхід базується на обробці вхідних даних блоками, що дозволяє зменшити кількість обчислень, необхідних для зсуву буферів, і оптимізувати управління пам’яттю. Крім того, блочна реалізація є порівняно нескладною в реалізації, що робить її особливо привабливою для інтеграції у вбудовані системи з обмеженими обчислювальними ресурсами. Однією з ключових переваг блочного підходу є можливість паралельної обробки даних, що забезпечує ефективніше використання апаратних ресурсів сучасних мікропроцесорів та цифрових сигнальних процесорів. Крім того, оптимізація з використанням блочної структури дозволяє мінімізувати кількість операцій копіювання та зменшити затримки, спричинені доступом до пам’яті.

```
1. void applyHighPassFilter(int16_t *input, int16_t *output, int length)
2. {
3.     int i, j;
4.     int blockSize = 32; // Define block size for processing
5.     int numBlocks = (length + blockSize - 1) / blockSize;
6.
7.     for (int block = 0; block < numBlocks; block++)
8.     {
9.         int start = block * blockSize;
10.         int end = start + blockSize;
11.         if (end > length)
12.             end = length;
13.
14.         for (i = start; i < end; i++)
15.         {
16.             insamp[i + MAX_FLT_LEN - 1] = input[i];
17.         }
18.
19.         for (i = start; i < end; i++)
20.         {
21.             int32_t acc = 0;
22.             for (j = 0; j < FILTER_LEN; j++)
23.             {
24.                 acc += (int32_t)coeffs[j] * (int32_t)insamp[i + MAX_FLT_LEN - 1 - j];
25.             }
26.             output[i] = (int16_t)(acc >> 15);
27.         }
28.
29.         memmove(insamp, &insamp[length], (MAX_FLT_LEN - 1) * sizeof(int16_t));
30.     }
31. }
32.
```

У результаті проведеного дослідження отримано наступні результати (рис. 2):

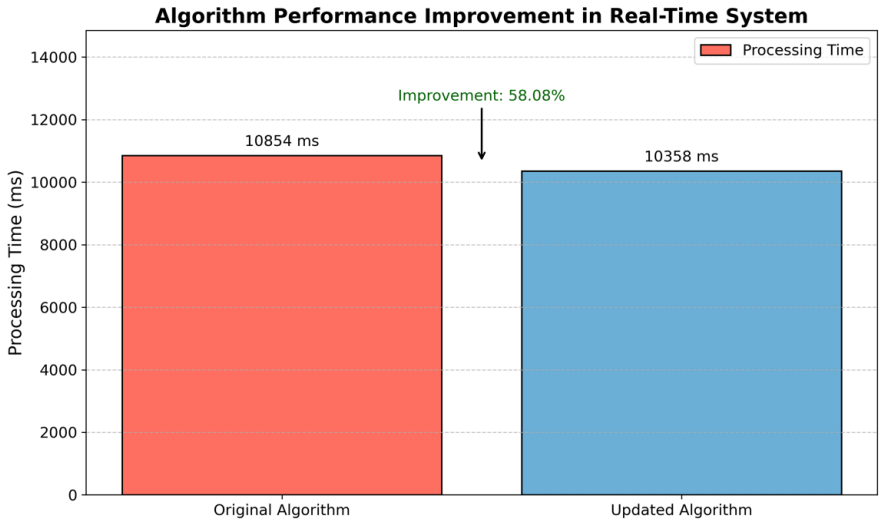

Рис. 2. Графічне відображення покращення алгоритму

Запропонована блочна реалізація фільтрів дозволила досягти суттєвого підвищення швидкості виконання алгоритму. Завдяки ефективній організації обробки даних та зменшенню кількості операцій з пам’яттю, вдалося збільшити продуктивність у 58.08% порівняно з базовою реалізацією. Такий результат демонструє доцільність використання блочного підходу в системах цифрової обробки сигналів, зокрема у вбудованих системах, де оптимізація обчислювальних витрат і ресурсів є критично важливою. Це підтверджує потенціал методу для впровадження в реальні застосунки з високими вимогами до швидкодії.

Висновки

У статті досліджено питання оптимізації цифрової обробки сигналів із використанням блочної реалізації фільтрів із фіксованими коефіцієнтами. Основна увага приділялася скороченню обчислювальної складності та ефективнішому використанню апаратних ресурсів. Запропонований підхід дозволив досягти значного покращення продуктивності, підвищивши швидкодію алгоритму у 58.08% порівняно з традиційною реалізацією. Окрім блочного підходу, у статті було проведено аналіз альтернативних методів оптимізації, зокрема використання багатостадійних фільтрів, розпаралелювання обчислень, а також оптимізації на рівні коду. Порівняння цих методів показало, що блочна реалізація є особливо ефективною для вбудованих систем, де обмеженість ресурсів і вимоги до енергоефективності є вирішальними факторами.

Результати роботи мають практичну значущість для застосувань у телекомунікаціях, обробці аудіо- та відеосигналів, а також у пристроях Інтернету речей. Перспективи подальших досліджень включають інтеграцію запропонованого підходу з адаптивними фільтрами та аналіз його ефективності для нелінійних систем. Це відкриває нові можливості для оптимізації цифрової обробки сигналів у різних прикладних задачах.

Перелік використаних джерел

[1] T. Dutta, R. M. Aich, S. Dhabal and P. Venkateswaran, "Finite Impulse Response Filter Design using Grasshopper Optimization Algorithm and Implementation on FPGA," 2020 IEEE Applied Signal Processing Conference (ASPCON), Kolkata, India, 2020, pp. 313-317, https://doi.org/10.1109/ASPCON49795.2020.9276711.

[2] D. Kalaiyarasi and T. K. Reddy, "A hybrid window function to design finite impulse response low pass filter with an improved frequency response," 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), Chennai, India, 2017, pp. 138-143, https://doi.org/10.1109/ICECDS.2017.8389684.

[3] N. Singh and A. Potnis, "A review of different optimization algorithms for a linear phase FIR filter," 2017 International Conference on Recent Innovations in Signal processing and Embedded Systems (RISE), Bhopal, India, 2017, pp. 44-48, https://doi.org/10.1109/RISE.2017.8378122.

[4] N. Sajwan, A. Kumar, I. Sharma and L. K. Balyan, "Performance of Multiplierless FIR Filter based on CSD and Binary: A Comparative Study," 2019 International Conference on Signal Processing and Communication (ICSC), NOIDA, India, 2019, pp. 217-222, https://doi.org/10.1109/ICSC45622.2019.8938282.

[5] K. Loubna, B. Bachir and Z. Izeddine, "Optimal digital IIR filter design using ant colony optimization," 2018 4th International Conference on Optimization and Applications (ICOA), Mohammedia, Morocco, 2018, pp. 1-5, https://doi.org/10.1109/ICOA.2018.8370500.

[6] M. T. Khan and R. A. Shaik, "Analysis and Implementation of Block Least Mean Square Adaptive Filter using Offset Binary Coding," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 2018, pp. 1-5, https://doi.org/10.1109/ISCAS.2018.8350946.

[7] A. Wahbi, A. Roukhe and L. Hlou, "Enhancing the quality of voice communications by acoustic noise cancellation (ANC) using a low cost adaptive algorithm based Fast Fourier Transform (FFT) and circular convolution," *2014 9th International Conference on Intelligent Systems: Theories and Applications (SITA-14)*, Rabat, Morocco, 2014, pp. 1-7, https://doi.org/10.1109/SITA.2014.6847310.

[8] Z. Shang, "Recursive Algorithm FIR Filter Quantization and Low-Cost Structure Optimization Design Based on MATLAB and Simulink," *2023 IEEE 16th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*, Singapore, 2023, pp. 530-535, https://doi.org/10.1109/MCSoC60832.2023.00084.

Oleh Novosad[1], Serhiy Shcherbovskykh[2]

[1] Computer Design Systems Department, Lviv Polytechnic National University, Ukraine, Lviv, S. Bandery street 12, E-mail: oleh.v.novosad@lpnu.ua, ORCID 0009-0003-6590-457X

[2] Computer Design Systems Department, Lviv Polytechnic National University, Ukraine, Lviv, S. Bandery street 12, E-mail: serhiy.v.shcherbovskykh@lpnu.ua, ORCID 0000-0001-8535-733X

REDUCING THE COMPUTATIONAL COMPLEXITY OF FINITE IMPULSE RESPONSE FILTERS IN INTEGRATED SYSTEMS

Recieved: February 20, 2025 / Revised: February 25, 2025 / Accepted: February 28, 2025

Abstract. The problem of optimizing finite impulse response (FIR) digital filters is relevant in the context of limited computational resources of embedded systems. Existing methods for implementing FIR filters often lead to significant energy consumption and processing time, which limits their application in real-world conditions. The objective of this work is to develop methods for optimizing FIR filters to reduce computational complexity and ensure efficient operation on embedded systems with limited resources. The research methodology includes the analysis of structurally optimized filter circuits, in particular multi-stage and block configurations, as well as the use of algorithms to reduce the bit depth of coefficients. The results of the study show that the proposed approaches can significantly reduce computational cost and power consumption without sacrificing filter quality. The novelty of the work lies in the combination of structural optimization methods with algorithmic techniques that consider the peculiarities of hardware implementation on embedded systems. The practical significance of the work lies in the possibility of using optimized filters for real-time signal processing on devices with limited resources. Directions for further research include studying the effectiveness of the optimization methods on different platforms, such as FPGAs and microcontrollers, as well as adaptation to different types of signals.

Keywords: digital signal processing, block filter implementation, optimization of computational complexity, energy efficiency, adaptive filters