Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Модель рекурсивного цифрового цілочисельного фільтра
на мовах опису апаратури
(тема)

Виконав:
студент _2_ курсу, групи *СКСм-20-1*
Волков Є.І.
(прізвище, ініціали)

Спеціальність
123 Комп'ютерна інженерія
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Спеціалізовані комп'ютерні системи
(повна назва освітньої програми)

Керівник доц. Шкіль О.С.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри АПОТ _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проектування обчислювальної техніки

Рівень вищої освіти другий (магістерський)

Спеціальність 123 Комп'ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані комп'ютерні системи
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові Волкову Євгену Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Модель рекурсивного цифрового цілочисельного фильтра на мовах опису апаратури
затверджена наказом університету від «04» 11 2021 р. № 1635 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20 .12. 2021р.
3. Вихідні дані до роботи
аудиофайл
рекурсивний фільтр
співвідношення сигнал/шум 60 dB
4. Перелік питань, що потрібно опрацювати в роботі
Аналіз предметної області та постановка задачі.
Розробка моделі
Програмна реалізація
Аналіз отриманих даних.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) 21 слайд

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Приміт ка
1	Отримання завдання	02.09.2021-09.09.2021	
2	Аналіз літератури	10.09.2021-30.09.2021	
3	Розробка моделі	30.09.2021-03.11.2021	
4	Реалізація моделі	04.11.2021-30.11.2021	
5	Тестування отриманих данихі	01.12.2021-10.12.2021	
6	Оформлення пояснювальної записки	10.12.2021-23.12.2021	

Дата видачі завдання __02___ _____09_____2021р.

Студент _____
(підпис)

Керівник роботи _____ _____доц. Шкіль О.С._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить 61 сторінок, 18 рисунків, 24 джерела за переліком посилань.

ЦИФРОВА ОБРОБКА СИГНАЛІВ, ЦИФРОВІ ФІЛЬТРИ, MATLAB, ПЛІС

В ході виконання кваліфікаційної роботи проведено аналіз переваг та недоліків цифрових СІХ та НІХ фільтрів. Розглянуті етапи проектування цифрових фільтрів за допомогою MatLab і Simulink, FDATool.

Було виконане проектування оптимального цифрового фільтру з заданими частотними характеристиками при мінімальній кількості апаратних ресурсів. Було отримано VHDL опис фільтру. Проведено синтез та тестування спроектованого фільтру у середовищі пакету Xilinx.

ABSTRACT

Bachelor’s thesis contains 61 pages, 18 figures, 24 sources according to the list of links.

DIGITAL SIGNAL PROCESSING, DIGITAL FILTERS, MATLAB, FPGA

In the course of the qualification work the analysis of advantages and disadvantages of digital SIX and NIH filters was carried out. The stages of designing digital filters using MatLab and Simulink, FDATool are considered. The design of the optimal digital filter with the specified frequency characteristics with a minimum amount of hardware resources was performed. A VHDL filter description was obtained. Synthesis and testing of the designed filter in the Xilinx package environment was performed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

АЦП – аналогово-цифровий перетворювач;

ЦАП – цифроаналоговий перетворювач;

ЦОС – цифрова обробка сигналів

ЦСП (DSP) – цифровий сигнальний процесор;

АЧХ – амплітудно-частотна характеристика

НІХ – фільтр з нескінченною імпульсною характеристикою

СІХ – фільтр зі скінченною імпульсною характеристикою

ФЧХ – фазочастотна характеристика фільтра

ПЛІС – програмована логічна інтегральна схема

ФНЧ – фільтр нижніх частот

ФЧХ - фазо-частотна характеристика

ЦФ – цифровий фільтр

ВСТУП

Програмовані логічні інтегральні схеми (ПЛІС) отримують широке поширення в швидкодіючій цифровій обробці сигналів (ЦОС) завдяки паралельній архітектурі, що дозволяє значно прискорювати процеси обробки сигналів. Можливість переконфігуровувати ПЛІС та наявність паралельної архітектури дають велику обчислювальну потужність та дозволяють розробляти структури, що можуть бути налаштовані для найбільш ефективної реалізації алгоритмів цифрової обробки сигналів.

В операціях цифрової обробки сигналів особлива увага приділяється цифрової фільтрації, яка в середньому займає до половини всього обсягу обчислень. При цьому в таких системах, як системи розпізнавання мовлення чи вимірювальні системи, важливим є забезпечення лінійності фазових характеристик.

Проектування пристроїв цифрової обробки сигналів на ПЛІС є багаторівневим ієрархічним процесом. Досягти високої швидкості обчислень можна з допомогою методів паралельних розрахунків на ПЛІС. FPGA Xilinx останніх поколінь дозволяють реалізовувати більш ефективні пристрої порівняно з сигнальними процесорами.

В даний час ведеться активний пошук нових прийомів і підходів проектування цифрових фільтрів з метою зниження кількості використовуваних ресурсів і підвищення швидкодії при апаратній реалізації на програмованих логічних інтегральних схемах. Розробка пристроїв цифрової обробки сигналів на ПЛІС, у загальному випадку, представляє собою нову технологію проектування.

Найчастіше на практиці при побудові різних систем ЦОС широко застосовуються цифрові фільтри з постійними коефіцієнтами, які

використовують цілочисленну арифметику з фіксованою, а не з плаваючою точкою.

При проектуванні цифрових фільтрів для систем, що реалізуються на ПЛІС, метою проектування є отримання необхідних частотних характеристик при мінімальній кількості апаратних ресурсів, таких як логічні елементи. Неоптимальне вирішення цієї задачі призводить до нераціонального витрачання площі кристала, до невиправданого збільшення споживаної потужності, зниження швидкодії, перешкоджає розміщенню всієї системи, включаючи сам фільтр, на одному кристалі і, зрештою, підвищує вартість виробу. Таким чином, створення методів проектування рекурсивних цифрового фільтру з урахуванням основних факторів, що визначають їх апаратну реалізацію на ПЛІС, є актуальною науково-технічною проблемою.

Метою кваліфікаційної роботи є проектування оптимального цифрового фільтру з заданими частотними характеристиками при мінімальній кількості апаратних ресурсів на основі методів проектування рекурсивних цифрових фільтрів з урахуванням основних факторів, що визначають їх реалізацію.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Цифрові фільтри

До задач цифрової обробки сигналів можна віднести такі області як, цифрові методи вимірів, обробка сигналів у радіолокації, стиснення даних, аналіз спектру, прийом сигналів та аналіз вібрацій тощо. Одним з найважливіших методів, що використовуються у цифровій обробки сигналів, є цифрова фільтрація. Цифрові фільтри в порівнянні з аналоговими мають значні переваги, але мають і недоліки. Розглянемо їх більш детальніше.

До принципових переваг цифрових фільтрів в порівнянні з аналоговими можна віднести наступне [1-3]:

– можливість одночасного обслуговування одним арифметичним пристроєм кількох каналів, що забезпечується завдяки дискретності цифрових сигналів у часі;

– можливість одночасної реалізації різних АЧХ у різних каналах за допомогою одного арифметичного пристрою;

– можливість отримання скільки завгодно великого і навіть нескінченного згасання на певних частотах.

До переваг, які отримуються при реалізації цифрових фільтрів можна віднести наступне:

– високу стабільність характеристик, яка обмежена лише стабільністю опорної частоти;

– повторюваність характеристик, а саме частотні характеристики цифрового фільтру визначаються лише чисельними значеннями їх коефіцієнтів;

– простота перебудови параметрів;

– можливість реалізації лінійної ФЧХ;

– можливість реалізації точної скінченої імпульсної характеристики (СІХ);

– можливість отримання дуже вузьких смуг пропускання.

До недоліків можна віднести наступне:

– при деяких реалізаціях цифрових фільтрів, вони можуть бути нестійкими;

– наявність шумів обробки, що виникають через кінцеву розрядність цифрових схем, при цьому шуми обробки можуть бути зменшені за рахунок збільшення розрядності або за рахунок зменшення швидкодії цифрового фільтру;

– у рекурсивних фільтрів наявні нелінійні ефекти, та якщо не вжито спеціальних заходів, то можуть виникнути коливання малої амплітуди при постійному вхідному сигналі (малі граничні цикли) або коливання гранично великої амплітуди (великі граничні цикли);

– відповідно до швидкодії роботи цифрових пристроїв обмежений частотний діапазон.

– використання таких пристроїв як аналого-цифрових перетворювачів, що викликає обмеження чутливості, динамічного діапазону, лінійності сигналу, що обробляється.

При цьому в таких системах, як системи розпізнавання мовлення чи вимірювальні системи, важливим є забезпечення лінійності фазових характеристик. Ця вимога виконується при обробці сигналів цифровими фільтрами з кінцевою імпульсною характеристикою (СІХ-фільтрами або нерекурсивними фільтрами). Крім лінійності фазових характеристик СІХ-фільтри (англ. FIR – "finite impulse response") є принципово стійкими системами та процес обчислення їх коефіцієнтів не викликає труднощів.

Однак за підвищених вимог до вибірковості таких фільтрів – круті спади амплітудно-частотної характеристики (АЧХ) та характеристики загасання – необхідні значущі апаратні та обчислювальні витрати. Це накладає обмеження на практичну реалізацію СІХ-фільтрів на основі

традиційних компонентів цифрової електроніки, таких як мікропроцесори та цифрові сигнальні процесори (DSP).

На відміну від СІХ-фільтрів НІХ-фільтри (англ. IIR - "infinite impulse response"), або рекурсивні фільтри (фільтри з нескінченною імпульсною характеристикою) не мають строго лінійної фазової характеристики, за винятком окремого випадку, коли всі полюси знаходяться на одиничному колі. Однак вони більш ефективні, ніж СІХ-фільтри, та мають менший порядок за однакових параметрів. Рекурсивні фільтри відрізняються від СІХ-фільтрів наявністю зворотного зв'язку.

До недоліків НІХ-фільтрів відносяться ефект самозбудження при недостатній розрядності коефіцієнтів фільтрів або неправильному розрахунку передавальної характеристики.

Ураховуючи всі переваги та недоліки при реалізації системи цифрової обробки інформації вибір між СІХ та НІХ фільтрами проводиться за наступними критеріями:

– фазова характеристика БІХ фільтра – нелінійна, а СІХ фільтри можуть мати строго лінійну фазову характеристику, це означає, що такий фільтр не вносить спотворень у форму сигналу;

– СІХ фільтри – стійкі, т.к. реалізуються за нерекурсивною формою, однак НІХ фільтри можуть бути нестійкими;

– реалізація АЧХ складної форми або максимально прямокутної форми вимагатиме значної кількості коефіцієнтів КІХ фільтра, а НІХ фільтри із цим завданням справляються краще;

– НІХ фільтри, як правило, не мають еквівалентних аналогових фільтрів;

– СІХ фільтри дозволяють легко отримувати необхідні характеристики (рівень згасання, нерівномірності у смузі пропускання, частота зрізу тощо);

– НІХ фільтри суттєво економніші за кількістю операцій множення, складання та кількістю ліній затримки.

Одже, основними перевагами НІХ-фільтірів є краща амплітудно-частотна характеристика в порівнянні з СІХ при меншій кількості коефіцієнтів, необхідних для реалізації, і меншій затримці проходження сигналу. Однак НІХ-фільтри мають нелінійності фазо-частотної характеристики (ФЧХ), можуть бути нестабільні та мають меншу швидкодію порівняно з СІХ фільтрами.

1.2 Рекурсивні фільтри

Розглянемо більш детальніше СІХ та НІХ фільтри. СІХ (англ. FIR – "finite impulse response") фільтр має імпульсну характеристика, яка обмежена у часі, тобто вона має скінчену кількість коефіцієнтів. З зазначеного моменту часу, вона дорівнює нулю. Якщо на вхід СІХ-фільтра подати одиничний імпульс, то на виході фільтра можна буде побачити кінцеву кількість відліків.

Як правило, ФЧХ СІХ-фільтра є лінійною. Такі фільтри мають назву нерекурсивні. СІХ-фільтри реалізуються без зворотних зв'язків, однак вони можуть мати велику кількість коефіцієнтів. Проте за допомогою математичних перетворень можна привести нерекурсивний фільтр до рекурсивної форми. Це дає змогу зменшити кількість коефіцієнтів, тобто порядок фільтру.

Математично СІХ-фільтр можна записати у вигляді:

, (1.1)

де y(n) – вихідний дискретний сигнал (сума зважених вхідних імпульсів), x(n) – вхідний дискретний сигнал (послідовність відліків), h(n) – коефіцієнти імпульсної характеристики фільтра, N – довжина (порядок) фільтра.

Передавальна характеристика СІХ-фільтра має вигляд:

, (1.2)

де операція – затримка послідовності на k-відліків.

НІХ фільтр (англ. IIR – "infinite impulse response") – це цифровий фільтр з нескінченною в часі імпульсною характеристикою, тобто якщо подати одиничний імпульс на вхід такого фільтру, то на виході фільтра можна буде побачити нескінчену кількість відліків.

НІХ фільтри також називають рекурсивними у зв'язку з тим, що при їх реалізації використовуються зворотні зв'язки (сигнал з виходу фільтра через елементи затримки надходить на фільтр та вносить зміни сам у себе).

Різницеве рівняння НІХ-фільтра має дрібно-раціональний вигляд.

, (1.3)

або

$$y(n) = b_0x(n) + b_1x(n-1) + b_Mx(n-M) + \ldots - a_1y(n-1) - a_2y(n-2) - \ldots - a_Ny(n-N).$$

Найбільш використовуються наступні НІХ фільтри:

– фільтр Чебишева,

– фільтр Баттерворта,

– фільтр Калмана,

– фільтр Бесселя тощо.

З різницевого рівняння можна побачити, що вихідний сигнал за рахунок зворотного зв'язку впливає сам на себе.

Передавальна характеристика НІХ-фільтра має наступний вигляд:

. (1.4)

На відміну від СІХ фільтрів, НІХ фільтри можуть бути не завжди стійкими. Для стійкості цифрового НІХ фільтра потрібно, щоб усі полюси передавальної характеристики по модулю були строго менше одиниці (тобто лежали всередині одиничного кола на z-площині).

Максимальна ступінь у виразу передавальної функції H(z) є порядок фільтра.

Передавальна функція може бути реалізована за допомогою різних структурних схем.

Найбільш поширені послідовні та паралельні структурні схеми. Якщо передавальну дробно-раціональну функцію рекурсивного фільтра надати у вигляді множини передавальних функцій першого і другого порядку, то структурна схема буде послідовною. Якщо передавальну функцію рекурсивного фільтра надати у вигляді суми передавальних функцій першого і другого порядку, то структурна схема буде параллельною.

Структура ЦФ відображає алгоритм обчислення реакції, що описується різницевим рівнянням. Крім загального вигляду (1.4), передавальна функція НІХ-фільтра може бути представлена в інших еквівалентних видах, серед яких практичний інтерес представляє два наступні:

– добуток множників другого порядку з речовими коефіцієнтами;

. (1.5)

– сума дробів другого порядку з речовими коефіцієнтами:

. (1.6)

Цім передавальним функціям відповідають свої еквівалентні види разностних рівнянь, тобто інші алгоритми обчислення реакції.

Три основних види передавальної функції НІХ-фільтрів:

– загальний (дрібно-раціональний) (1.4);

– добуток (1.5);

– сума (1.6).

Ці типи визначають три основні структури НІХ-фільтрів:

1) пряму, а саме Direct-Form I та її модифікації (на рис. 1.1 вони представлені для ланки 2-го порядку):

– пряму транспоновану структуру – Direct-Form I Transposed;

– пряму канонічну структуру- Direct-Form II;

– пряму канонічну транспоновану структуру – Direct-Form II Transposed;

2) каскадну з звена 2-го порядку із прямою структурою або її модифікацією;

3) паралельну з звена 2-го порядку із прямою структурою або її модифікацією.

На рис1.1 наведені різноманітні структурні схеми НІХ-фільтрів.

Послідовність синтезу НІХ-фільтру наступна:

– необхідно задати вимоги до амплітудно-частотної характеристики фільтра, з урахуванням того, що значення АЧХ в полосі пропуску не повинна перевищувати одиницю;

– необхідно обрати метод синтезу фільтра;

– далі робиться вибір апроксимуючих функцій відповідно до обраного типу аналогового прототипу;

– задати вимоги до фільтру відповідно технічного завдання, з урахуванням того, що максимально допустимі відхилення АЧХ сов падають з відповідними значеннями для НІХ фільтра; однак значення граничної частоти полоси пропускання та полоси загородження будуть залежати від обраного методу синтезу;

– зробити розрахунок передаточної функції;

– обрати структуру фільтру.

Рисунок 1.1 – Структура звена 2-го порядка: а) Direct_Form I; б) Direct_Form I Transposed; в) Direct_Form II; г) Direct_Form II Transposed

При проектуванні фільтру бажано використовувати структурно-функціональний підхід, що полягає в основі моделювання та синтезу цифрового фільтра. Цей підхід є системним підходом, що викладається в [3-5]. Відповідно до структурно-функціонального підходу цифровий фільтр може бути охарактеризовано з двох важливих сторін. Внутрішній його стан прийнято описувати на двох ієрархічних рівнях структурному та параметричному.

В роботі [24] наведено порівняння каскадних, паралельних, прямих та нерекурсивних структур побудови цифрових фільтрів і показано, що каскадна структура є найкращою для реалізації НІХ-фільтрів, оскільки:

– дозволяє реалізовувати ряд передавальних функцій цифрового фільтра невеликим набором основних функціональних елементів (ланок) низького порядку;

– чутливість характеристик фільтра до зміни параметрів (коефіцієнтів) найменша при послідовній структуру побудови фільтра;

– каскадна структура зручна у разі підстроювання після синтезу, оскільки кожна ланка фільтра ізольована один від одного.

Тому в побудова рекурсивних фільтрів у формі каскадного з'єднання (рис. 1.2) ланок другого порядку прямої або зверненої форми є оптимальнішою.

Рисунок 1.2 – Каскадна форма побудови фільтра

1.3 Використання ПЛІС для реалізації задач ЦОС

Для реалізації задач цифрової обробки сигналів у реальному масштабі часу можна використовувати таку апаратуру як мікроконтролери, цифрові сигнальні процесори або програмовані логічні інтегральні схеми. Кожна апаратна реалізація має як свої переваги, так і недоліки. Розглянемо їх більш детальніше.

Для цифрової обробки сигналів використовуються програмовані логічні матриці, які забезпечують більш високу швидкість обробки, порівняно з цифровими процесорами обробки сигналів (DSP).

Програмовані логічні інтегральні схеми (ПЛІС) набувають широкого поширення швидкодіючої цифрової обробки сигналів. Завдяки паралельній архітектурі та можливості реконфігурування ПЛІС мають велику

обчислювальну продуктивність і дозволяють розробляти структури які можна максимально налаштовувати для ідеальної реалізації алгоритмів ЦОС.

Однак дійсна арифметика не може бути безпосередньо застосована в ПЛІС, як наприклад у цифрових сигнальних процесорах (DSP), де є апаратний модуль операцій з плаваючою комою (FPU). Сигнальні процесори незамінні при реалізації складних алгоритмів обчислень, виконання яких на ПЛІС невигідне або неможливе. Незважаючи на цей недолік у ПЛІС є істотна перевага над цифровими сигнальними процесорами, це можливість високого ступеня паралелізму, що дозволяє проводити цифрову обробку на кілька порядків швидше.

Для роботи в реальному масштабі часу DSP-процесор повинен бути розрахований на виконання всіх кроків у програмі фільтрації в межах проміжку часу, що відповідає одному такту дискретизації, тобто 1/fs.

Високопродуктивний універсальний цифровий процесор з фіксованою точкою типу ADSP-2189M, що має швидкодію 75MIPS, здатний виконати операцію множення з накопиченням при реалізації одного каскаду фільтра за 13,3 нс.

Сигнальний процесор ADSP-2189M витрачає N+5 тактів при реалізації фільтра з кількістю каскадів N. Для 100-каскадного фільтра повний час обчислення становить приблизно 1,4 мкс. Це відповідає максимально можливій частоті дискретизації 714 кГц, обмежуючи, таким чином, ширину смуги частот оброблюваного сигналу декількома сотнями кілогерц [16].

При реалізації того ж фільтра на FPGA, можливе скорочення часу обчислення N каскадів за рахунок розпаралелювання виконання операцій. Коефіцієнт розпаралелювання (M) визначатиметься кількістю вбудованих у FPGA перемножувачів.

Так, наприклад, для FPGA "Spartan3E" залежно від типу кристала, скорочення часу складе від 0 до 800%.

Ще більшої ефективності можна досягти при використанні FPGA "VirtexXX", орієнтованих на цифрову обробку сигналів. Величина скорочення часу при цьому становитиме до 8000%.

Алгоритм фільтрації в DSP процесорі виконується наступним чином:

– спочатку отримання відліку від АЦП;

– переміщення відліку в циклічний буфер вхідного сигналу;

– потім оновлення вказівника циклічного буфера вхідного сигналу;

– обнулення аккумулятора;

– виконання одного циклу фільтрації, тобто цикл за всіма коефіцієнтами;

– вибір коефіцієнта з циклічного буфера коефіцієнтів;

– оновлення покажчика циклічного буфера коефіцієнтів;

– вибір відліку з циклічного буфера вхідного сигналу;

– оновлення вказівника циклічного буфера вхідного сигналу;

– збільшення коефіцієнта на відлік;

– додавання нового доданку до проміжного результату;

– видача відфільтрованого відліку на ЦАП.

Для сигнальних процесорів компанії Analog Devices усі операції, що виконуються за один цикл фільтра, виробляються за один командний цикл процесора, завдяки чому суттєво збільшується ефективність обчислень. Ця перевага відома як реалізація циклів без додаткових операцій. Проблема полягає в майже повній відсутності оптимізації в трансляторах та необхідності програмувати на асемблері, що у свою чергу вимагає від програміста знання всіх тонкощів архітектури DSP.

Для реалізації систем цифрової фільтрації на FPGA програмування замінюється на побудову структури, реалізує виконання алгоритму. Ефективність роботи структури буде повністю визначатися закладеними у принципами. Проектування цифрових фільтрів для систем, що реалізуються на ПЛІС, пов'язане з вирішенням задачі отримання необхідних характеристик при мінімальній кількості логічних елементів, що використовуються.

Неоптимальне вирішення цієї задачі призводить до безглуздої витрати логічних і трасувальних ресурсів мікросхеми, до невиправданого збільшення споживаної потужності, зниження швидкодії та підвищення вартості, що перешкоджає бажанню розробників розміщувати свої системи на одному або малому числі кристалів.

Основна складність цифрових фільтрів, що складаються з помножувачів, суматорів, регістрів та інших допоміжних пристроїв визначається головним чином помножувачами [13].

Максимальна частота дискретизації досягається застосуванням паралельної обробки без мультиплексування. У практиці побудови високошвидкісних систем цифрової обробки сигналів широке застосування знайшли паралельні цифрові фільтри з постійними коефіцієнтами.

Помножувач на постійний коефіцієнт може бути реалізований ПЛІС декількома способами: як повний двовходовий помножувач і як одновходовий помножувач на константу. Як повні помножувачі найчастіше використовуються апаратні помножувачі, що з'явилися в новітніх сімействах ПЛІС, проте їх невелика кількість обмежує їх використання. Тому поширеним рішенням є використання помножувачів на константу, побудованих на логічних елементах ПЛІС.

Найбільш поширеним підходом до побудови помножувача на константу у ПЛІС є використання методу часткових добутків. При цьому кількість використовуваних логічних ресурсів і тактова частота визначаються як розрядність вхідних даних, і значенням константи. Одиницею виміру логічних ресурсів ПЛІС прийнято вважати кількість логічних вентелей, що необхідні для реалізації блоку. Наприклад, у ПЛІС фірми Xilinx елементарним логічним вентелем вважається slice, що складається з двох лут таблиць (LUT - look-up table) і тригерів (Flip Flops). На рис. 1.3 наведено приклад внутрішньої структури для slice ПЛІС XC3S700AN фірми Xilinx.

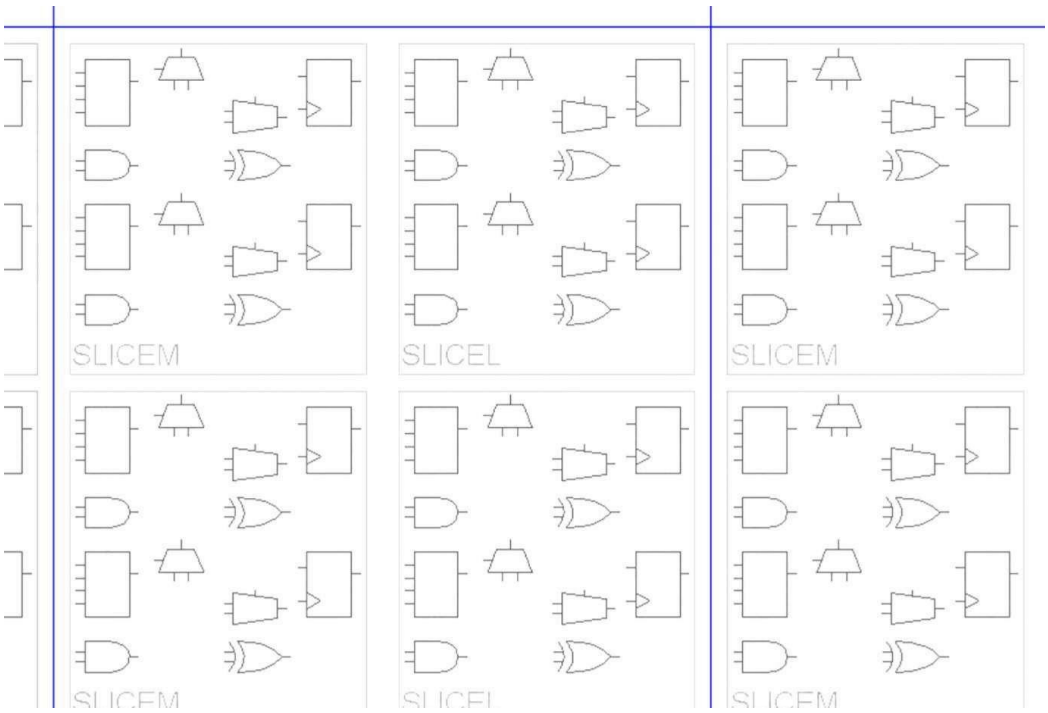


Рисунок 1.3 – Внутрішня структура ПЛІС XC3S700AN фірми Xilinx

Для помножувача на константу на основі методу часткових помножувачів головним чином використовуються LUT, які переважно визначають кількість займаних slice. Саме тому кількість необхідних LUT є основним критерієм ефективності реалізації помножувача на константу ПЛІС. Кількість LUT визначається після логічного синтезу.

Виходячи з переліченого вище в даний час спостерігається перехід до використання саме ПЛІС, що обумовлено наступними перевагами [16]:

– високою тактовою частотою, тобто частота роботи ПЛІС близько кількох ГГц;

– апаратною реалізацією помножувачів, що дозволяє множити за один такт;

– наявністю апаратних інтерфейсів проти DSP;

– гнучкістю під час розведення плати, що дозволяє подавати дані довільної розрядності.

Крім того, вартість ПЛІС нижча, ніж вартість сигнальних процесорів, а умови експлуатації їх однакові, тому при проектуванні пристроїв, коли це можливо, бажано використовувати саме ПЛІС.

Функціональність ПЛІС залежить від кількості розміщених на ній пристроїв обробки. Кожен пристрій, що розміщується на ПЛІС, використовує певну кількість її ресурсів. Одними з найбільш критичних елементів ПЛІС є помножувачі. До складу кожної ПЛІС входить фіксована кількість помножувачів, які можуть бути використані для побудови певної кількості систем. Тому пропонується для оптимального використання ресурсів ПЛІС мінімізувати число ненульових коефіцієнтів фільтра, що дозволить зменшити кількість помножувачів, що використовуються, а значить звільнити місце для інших пристроїв.

2 РЕАЛІЗАЦІЯ ФІЛЬТРІВ НА ПЛИС

2.1 Етапи проектування цифрових фільтрів на ПЛІС

Розробка пристроїв цифрової техніки на ПЛІС, у загальному випадку, є новою технологією проектування, включаючи їх виготовлення та супровід.

Проектування цифрових фільтрів на ПЛІС складається з наступних етапів:

1) розробка моделі ЦФ на системному рівні;

2) проектування лише на рівні регістрових передач;

3) логічний синтез, розміщення та трасування в ПЛІС.

Розглянемо етапи проектування цифрового фільтру з використанням середовища MatLab. На першому етапі розробляється модель ЦФ на системному рівні, для якого характерне вирішення таких завдань, як вибір типу фільтра, розрахунок порядку та коефіцієнтів передавальної функції, а також квантування коефіцієнтів [16]. Проектування можна виконувати у середовищі FDATool системи MatLab.

Оптимальна розрядність квантованих коефіцієнтів фільтра отримується завдяки розрахункам в пакету MatLab. Мета аналізу помилок квантування полягає у виборі такої довжини цифрового слова (коефіцієнтів ЦФ), щоб цифрова система досить точно реалізовувала бажану лінійну систему при мінімумі апаратних та програмних засобів. Збільшення довжини цифрового слова однією біт знижує значення помилок квантування вдвічі.

На другому етапі – рівні регістрових передач – формується опис ЦФ мовою опису апаратури (VHDL), що використовується для розміщення та трасування ЦФ ПЛІС.

На третьому етапі проектування ЦФ при логічному синтезі здійснюється перетворення HDL опису ЦФ до списку ланок, а також оптимізація списку ланок для конкретного кристала ПЛІС. На даному етапі

проектування проводять розміщення та трасування цифрового фільтра в ПЛІС за допомогою САПР ISE в автоматичному режимі.

Інтеграція математичних пакетів System Generator for DSP, MatLab і Simulink є ефективним способом вирішення задач у галузі цифрової обробки сигналів та дозволяє не вникати особливості архітектури ПЛІС, проводити проектування в основному в середовищі MatLab, отримуючи потім робочу конфігурацію ПЛІС [16]. Застосовуючи пакет SystemGenerator, розроблений спільно фірмами Xilinx і MathWorks, можна синтезувати потрібну системну функцію фільтра, провести моделювання його роботи, згенерувати код для програмування ПЛІС безпосередньо із середовища Simulink.

2.2 Середовище FDATool системи MatLab

Середовище FDATool – це графічний інтерфейс для розрахунку фільтрів та перегляду їх характеристик. Зовнішній вигляд наведено на рис. 2.1.

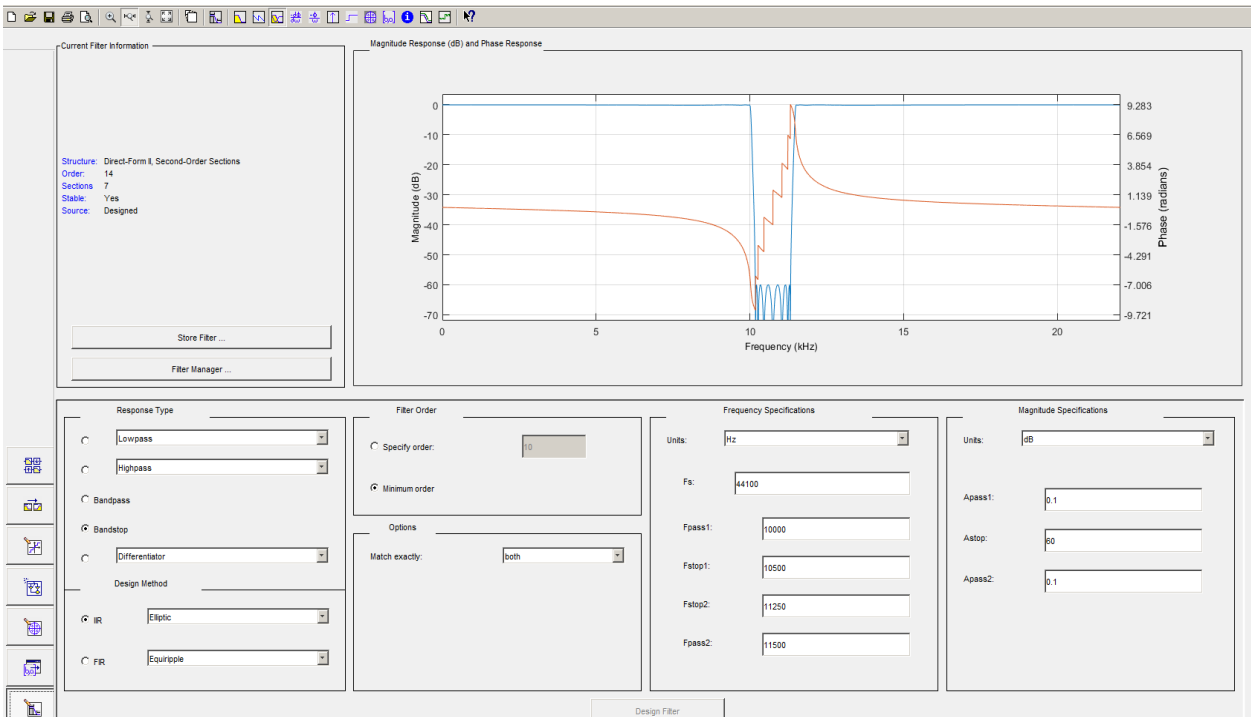


Рисунок 2.1 – Середовище FDATool

Основними етапами проектування в середовищі є вибір специфікації фільтра, розрахунок коефіцієнтів, архітектурна реалізація, аналіз помилок (шум округлення при квантуванні результатів арифметичних операцій, помилки квантування коефіцієнтів та вплив довжини вхідного слова – ефекти кінцевої розрядності). В середовищі також можна оцінити апаратні затрати проектує мого фільтру, отримати коефіцієнти фільтру. Також виконується генерація коду на мові C++ або VHDL та тестбенчу для здійснення тестування розробленого фільтру.

Застосування пакету Signal Processing (середовище FDATool) системи MATLAB/Simulink спільно з ModelSim SE дозволяє проводити проектування цифрових фільтрів у базисі універсальних та спеціалізованих процесорів цифрової обробки сигналів з використанням арифметики з фіксованою та плаваючою комою.

Для отримання результатів синтезу фільтру, який розробляється необхідно задати відповідні параметри, а потім здійснити синтез, після чого провести аналіз результатів фільтру для отримання оптимального до технічного завдання результату.

Відповідні параметри задаються на вкладці Design Filter. Тут необхідно задати тип вибірковості фільтру, який синтезується, а саме фільтр нижніх частот, верхніх частот, режекторний, або полосовий, обрати тип фільтра - нерекурсивний (FIR) або рекурсивний (IIR), метод синтезу, наприклад, метод вікон (синтез з використанням вагових функцій).

Середа FDATool підтримує декілька методів синтезу. При проектуванні фільтра в середовищі FDATool застосовуються наступні методи: Equiriple- синтез фільтрів з рівномірними пульсаціями АЧХ методом Ремеза; Least-Squares – мінімізація середньоквадратичного відхилення АЧХ від заданої та метод вікон (Window).

В розділі Filter Order необхідно задати порядок фільтра. Якщо обрати опцію «мінімальний», що порядок проектуємого фільтра оцінюється автоматично. Слід брати увагу на досить високий порядок не рекурсивного

фільтра, що виявляється зі зростанням відносіння FS/FC, що вважається одним з недоліків його використання.

Розрахунок фільтра здійснюється після натискання кнопки Design Filter.

Після завершення синтезу фільтру автоматично отримується графік характеристики згасання та поточна інформація про фільтр: структуру, порядок, стійкість та коефіцієнти фільтра. Це дозволяє проаналізувати отриманий результат.

Особливе місце на системному рівні проектування займає квантування коефіцієнтів фільтра, так як квантування вносить помилку в поведінку ЦФ внаслідок ефектів усічення/округлення, які виникають через обмежену розрядність подання коефіцієнтів проектованого ЦФ.

3 ПРОЕКТУВАННЯ ФІЛЬТРА

3.1 Розробка моделі

У роботі було запропоновано розробити цифрову систему фільтрації для заданого сигналу. Вхідний сигнал являє собою аудіофайл до якого додано невідому заваду.

Для того, щоб розробити систему фільтрації та здійснити виділення корисного сигналу від завади, заданий сигнал було проаналізовано у пакеті MATLAB.

Для цього аудіо файл було імпортовано за допомогою Impart Wizard до робочої області MATLAB. В результаті було отримано чисельний вектор аудіофайлу, який наведено на рисунку 3.1. Його вигляд у часовій області наведено на рисунку 3.2. Також файл було прослухано та наявність завади було чутко чути.

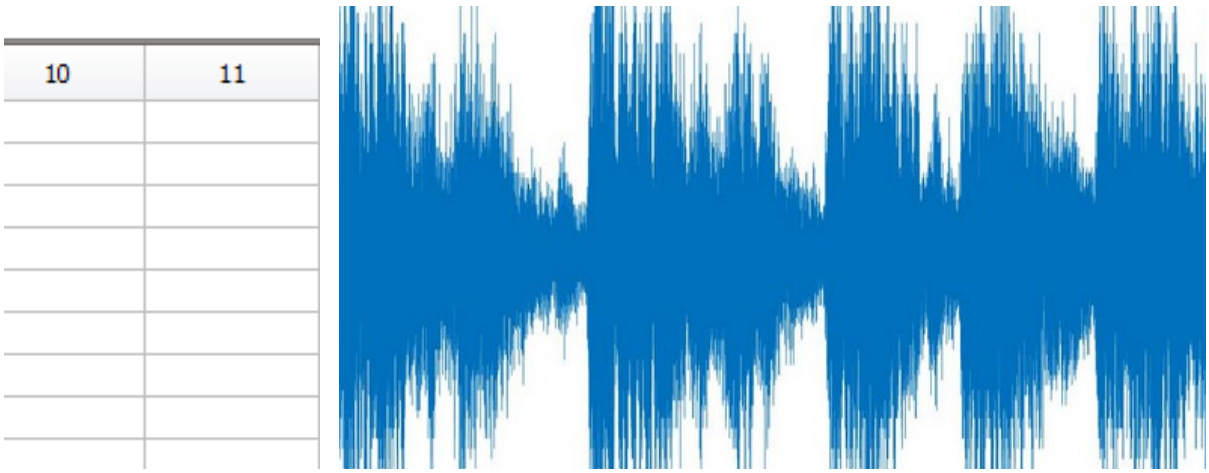


Рисунок 3.1 – Аудіофайл

Для більш детальнішого аналізу заданого аудіо файлу було використано додаток Signal analyzer. В додатку було проведено спектральний аналіз аудіофайлу и було знайдено заваду, яка знаходиться на частотах 10,7 – 11,2 кГц.

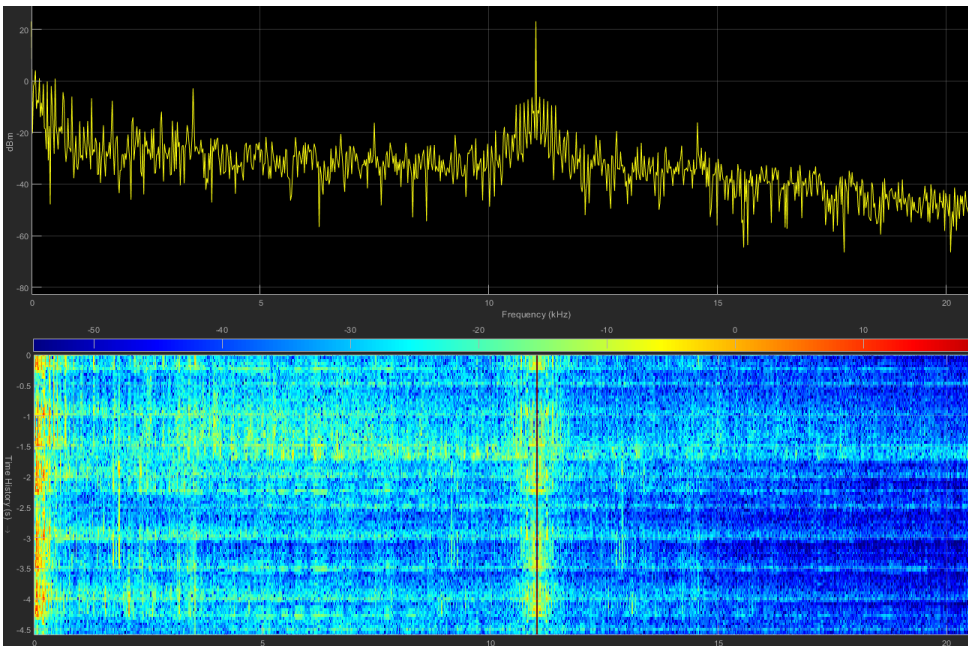


Рисунок 3.2 – Аналіз аудіофайлу

Використовуючи готові вбудовані фільтри було проведено аналіз засобів фільтрації, який необхідно використати для того, щоб видалити заваду з аудіофайлу. Найякіснішим у даному випадку є полосно-загороджуючий (режекторний) фільтр. Результат фільтрації наведено на рисунку 3.3.

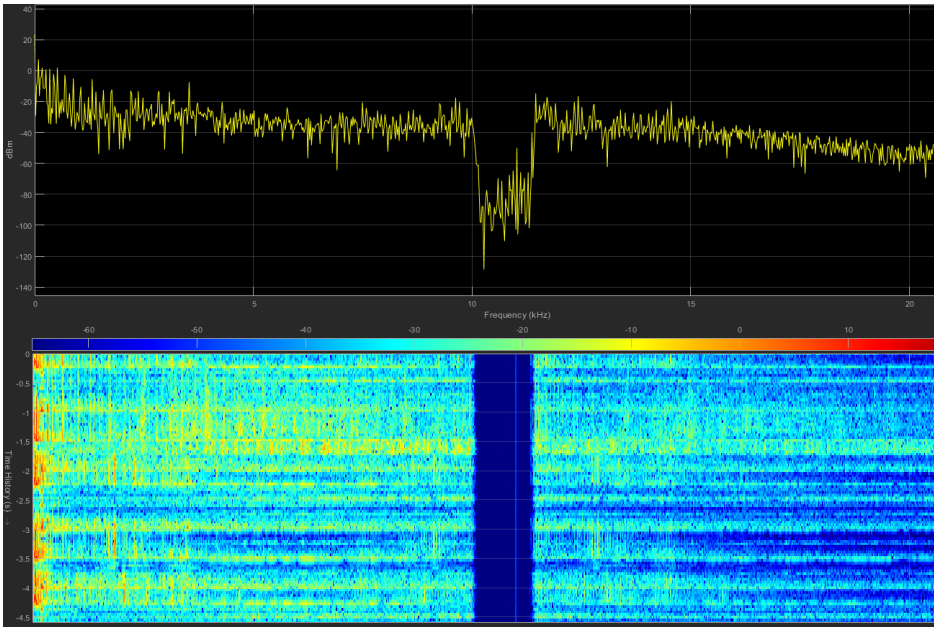


Рисунок 3.3 – Результат фільтрації

Для реалізації фільтру та проведення моделювання результатів побудови заданого фільтру у роботі було розроблено модель, яка наведена на рисунку 3.4. Модель містить наступні компоненти: джерело сигналу (програвач аудіо файлу з завадою), буфер приймального сигналу, фільтр, моделювання якого виконується у роботі, буфер вихідного сигналу, аналайзер, для побудови спектрограми, гучномовець для прослуховування результату обробки сигналу.

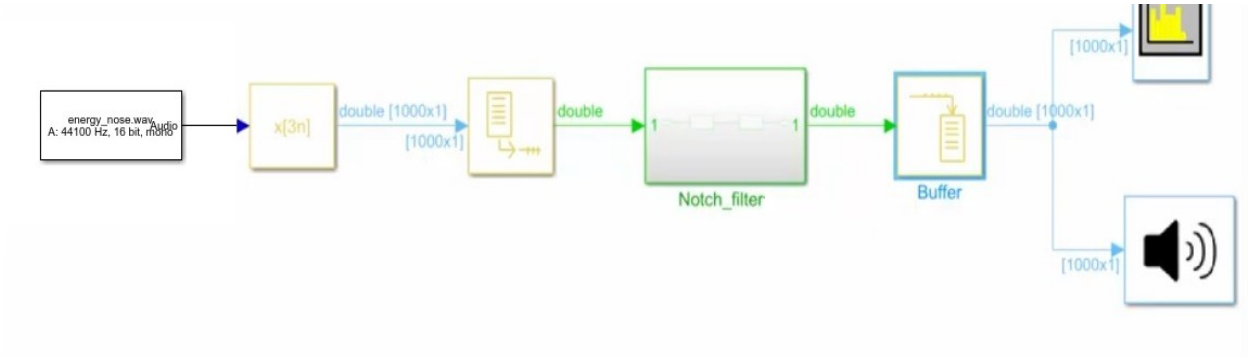


Рисунок 3.4 – Модель

3.2 Розробка фільтру

Для проектування фільтру був використаний додаток FDATool (Filter Design & Analysis Tool). За допомогою пакету можна здійснювати розрахунок передавальної функції фільтру, що розробляється, використовуючи різноманітні методи синтезу. Після розрахунку можна переглянути характеристики фільтра. Також можна аналізувати зміну характеристик фільтра відповідно до зміни його параметрів.

Також додаток надає інформацію про структуру фільтра, його порядок, стійкість та інше.

Відповідно до проведеного аналізу у розділах Filter Specifications і Magnitude Specifications були задані безпосередньо параметри специфікації

фільтра, а саме був побудований режекторний фільтр Bandstop. Був обраний тип фільтру рекурсивний (IIR – Infinite Impulse Response). Також було встановлено обрати найменший можливий порядок. Були задані безпосередньо наступні параметри специфікації фільтра: частоту дискретизації Fs 44 100Гц, граничні частоти смуги пропускання відповідно Fpass1 – 10 900 Гц та Fpass2 – 11 060 Гц, та смуг затримування Fstop1 – 10 950Гц та Fstop2 – 11 020Гц, припустимі загасання були задані в смузі пропускання Apass 1 і Apass 2 – 0,1dB та у смузі затримування Astop – 60 dB.

Після завдання типу фільтра та обраних його параметрів при натисканні кнопки Design Filter (Конструювання фільтра) було отримане програмне конструювання фільтра за заданими параметрами.

Було обрано спочатку фільтр-прототип: Elliptic (еліптичний) використовувався метод білінійного z-перетворення. Амплітудно-частотна та фазо-частотна характеристики фільтру наведені на рис. 3.5.

Рисунок 3.5 – Амплітудно-частотна та фазо-частотна характеристики фільтру Elliptic

З розрахунків можна побачити, що порядок фільтру – 12, кількість біполярних ланок дорівнює 6. Фільтр є стійким. Також з параметрів аналізу,

що наведені на рисунку для реалізації запропонованого фільтру необхідно 24 мультиплексора, 24 суматора.

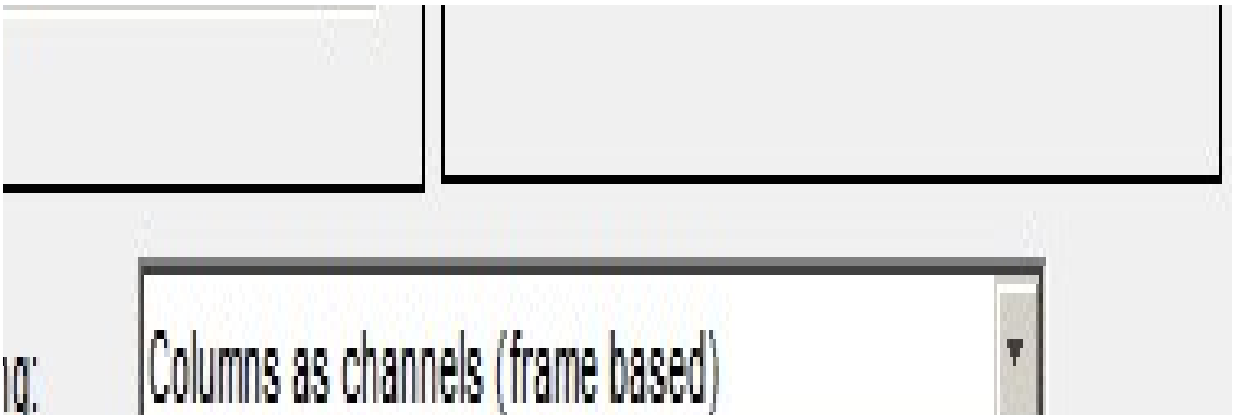


Рисунок 3.6 – Параметри фільтру Elliptic

Наступним кроком було обрано прототип фільтру Batterworth (Баттерворта), використовувався метод білінійного z-перетворення. Амплітудно-частотна та фазо-частотна характеристики фільтру Batterworth наведені на рис. 3.7.

Рисунок 3.7 – Амплітудно-частотна та
фазо-частотна характеристики фільтру Batterworth

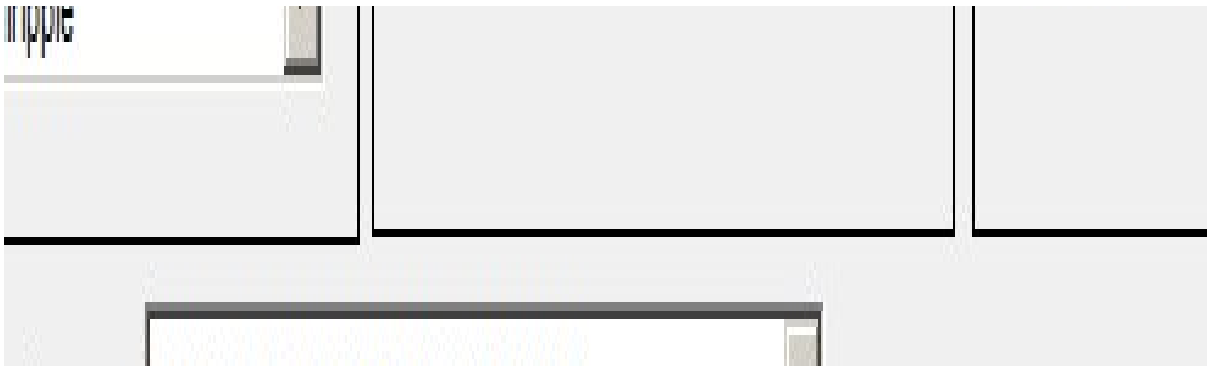


Рисунок 3.8 – Параметри фільтру Batterworth

З розрахунків можна побачити, що порядок фільтру є більший – 16, кількість біполярних ланок дорівнює 8. Фільтр є стійким. Для реалізації запропонованого фільтру Batterworth необхідно 32 мультиплексора, 32 суматори.

Порівняльний аналіз показав, що у еліптичного фільтра ступінчаста ФЧХ у полосі режекції, ніж у фільтра Batterworth. Однак це не є у даному випадку недоліком, тому що ця полоса частот знаходиться в зоні режекції.

Протилежно еліптичний фільтр при однакових параметрах потребує менше апаратних затрат, що є дуже важливим при реалізації. Тому був вибраний еліптичний фільтр, який при однакових заданих параметрах реалізації фільтра є більш оптимальнішим.

Внутрішню структуру фільтру (ланку) наведено на рисунку. На рисунку наведено роботу моделі з запропонованим фільтром. Можна побачити, що запропонована модель фільтру відповідає заданим параметрам та здійснює послаблення завади на 80 dB у полосі 100 Гц. Полоса режекції узгоджена з полосою завади.

На рисунку 3.10 наведено перша ланка розробленого фільтру. Фільтр складається з чотирьох ланок.

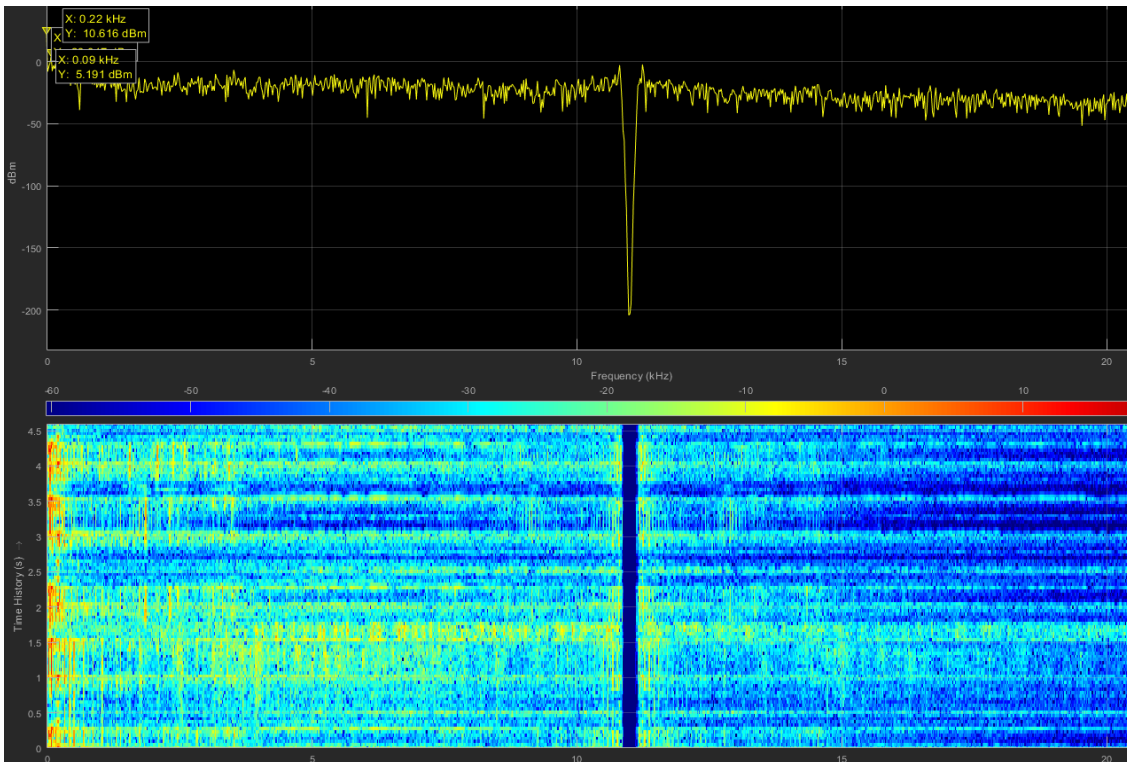


Рисунок 3.9 – Результати фільтрації

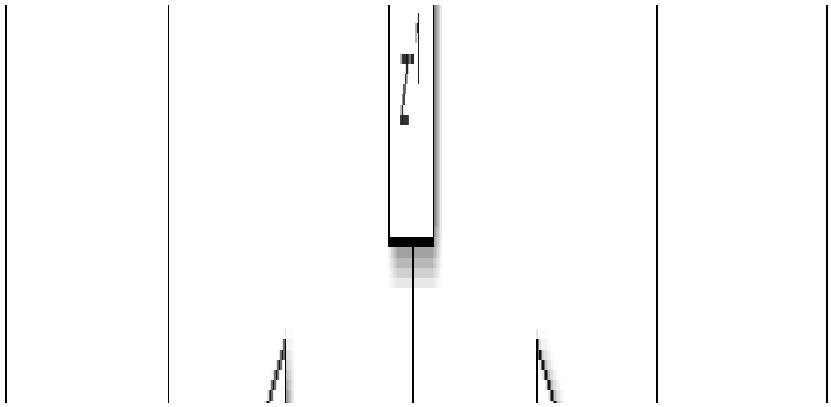


Рисунок 3.10 – Структура першої ланки фільтру

3.3 VHDL-опис фільтру

Для реалізації цифрового фільтрав на ПЛІС був обраний кристал фірми Xilinx. А саме реалізація цифрового фільтру здійснювалася на ПЛІС Spartan-3E сімейства FPGA, яка завдяки особливостям своєї архітектури надає дві ключові переваги для цифрової обробки сигналів. По-перше, архітектура

FPGA через високу продуктивність дуже добре підходить для паралельного виконання функцій цифрової обробки сигналів і, по-друге, можливість програмування ПЛІС дозволяє знайти компроміс між задіяними ресурсами ПЛІС та продуктивністю проектованого пристрою за допомогою вибору відповідного рівня паралелізму.

Було сгенеровано VHDL-опис розробленого фільтру для Spartan-3E та відповідно тестбенч для здійснення моделювання та перевірки результатів синтезу. Було виконано синтез, імплементацію, та тестування розробленого фільтру.

Блок Entity наведено в лістингу 3.1. Вхідними даними для фільтру є тактовий сигнал Clk, сигнал початку фільтрації clk_enable, сигнал скидання reset, та сигнал вхідних даних, що фільтуються filter_in. Вихідні дані отримуються на виході filter_out.

Для розрахунку було обрано формат `double`. Коефіцієнти фільтра представляються за допомогою вектора, що відповідає формату 1.16 (MN, де M — загальна кількість двійкових розрядів, що використовується для представлення чисел; N — число розрядів дробової частини, Numerator), що часто застосовується для представлення чисел у цифрових 16-бітових сигнальних процесорів.

Листинг 3.1

```
ENTITY filter IS
   PORT( clk:   IN    std_logic;
         clk_enable:   IN    std_logic;
         reset  :   IN    std_logic;
         filter_in  :  IN  std_logic_vector(15 DOWNTO 0);
         filter_out: OUT   std_logic_vector(32 DOWNTO 0)
         );

END filter;
```

Коефіцієнти фільтру наведено у листингу 3.2

Листинг 3.2

```
   CONSTANT scaleconst1 : real := 9.9725972625269421E-01; --
double
  CONSTANT coeff_b1_section1 : real := 1.0000000000000000E+00;
  CONSTANT coeff_b2_section1 : real := -7.1239585659800507E-03;
  CONSTANT coeff_b3_section1 : real := 1.0000000000000000E+00;
  CONSTANT coeff_a2_section1 : real := -2.0231454432960927E-02;
  CONSTANT coeff_a3_section1 : real := 9.9456277873445742E-01;
   CONSTANT scaleconst2 : real := 9.9725972625269421E-01; --
double
  CONSTANT coeff_b1_section2 : real := 1.0000000000000000E+00;
  CONSTANT coeff_b2_section2 : real := -7.1239585659800507E-03;
  CONSTANT coeff_b3_section2 : real := 1.0000000000000000E+00;
  CONSTANT coeff_a2_section2 : real := 6.0228865629273209E-03;
  CONSTANT coeff_a3_section2 : real := 9.9456252448784699E-01;
   CONSTANT scaleconst3 : real := 9.9345775330382491E-01; --
double
  CONSTANT coeff_b1_section3 : real := 1.0000000000000000E+00;
  CONSTANT coeff_b2_section3 : real := -7.1239585659800507E-03;
  CONSTANT coeff_b3_section3   : real := 1.0000000000000000E+00;
  CONSTANT coeff_a2_section3 : real := -1.2494145576771906E-02;
  CONSTANT coeff_a3_section3 : real := 9.8692301663716597E-01;
   CONSTANT scaleconst4   : real := 9.9345775330382491E-01; --
double
  CONSTANT coeff_b1_section4 : real := 1.0000000000000000E+00;
  CONSTANT coeff_b2_section4 : real := -7.1239585659800507E-03;
  CONSTANT coeff_b3_section4 : real := 1.0000000000000000E+00;
  CONSTANT coeff_a2_section4 : real := -1.6605062521070080E-03;
  CONSTANT coeff_a3_section4 : real := 9.8692276431639925E-01;
```

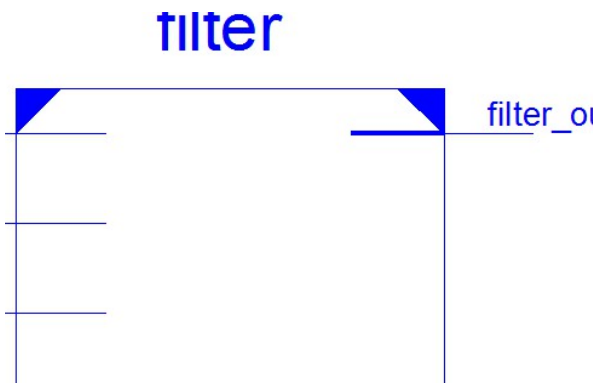
У листингу 3.3 наведено опис однієї секції фільтру.

Листинг 3.3

```
--   ------------------ Section 1 ------------------

  delay_process_section1 : PROCESS (clk, reset)
  BEGIN
    IF reset = '1' THEN
      delay_section1(0) <= 0.0000000000000000E+00;
      delay_section1(1) <= 0.0000000000000000E+00;
    ELSIF clk'event AND clk = '1' THEN
      IF clk_enable = '1' THEN
        delay_section1(1) <= delay_section1(0);
        delay_section1(0) <= a1sum1;
      END IF;
    END IF;
  END PROCESS delay_process_section1;

  inputconv1 <= scaletypeconvert1;


  a2mul1 <= delay_section1(0) * coeff_a2_section1;

  a3mul1 <= delay_section1(1) * coeff_a3_section1;

  b1mul1 <= a1sum1;


  b2mul1 <= delay_section1(0) * coeff_b2_section1;

  b3mul1 <= delay_section1(1);


  a2sum1 <= inputconv1 - a2mul1;

  a1sum1 <= a2sum1 - a3mul1;

  b2sum1 <= b1mul1 + b2mul1;

  b1sum1 <= b2sum1 + b3mul1;

  scale2 <= b1sum1 * scaleconst2;

  scaletypeconvert2 <= scale2;
```

Результати синтезу наведено на рис. 3.11.

Розміщення та розведення топологічних елементів обраної ПЛІС наведено на рис. 3.12.



а)

б)

Рисунок 3.11 – Результати синтезу

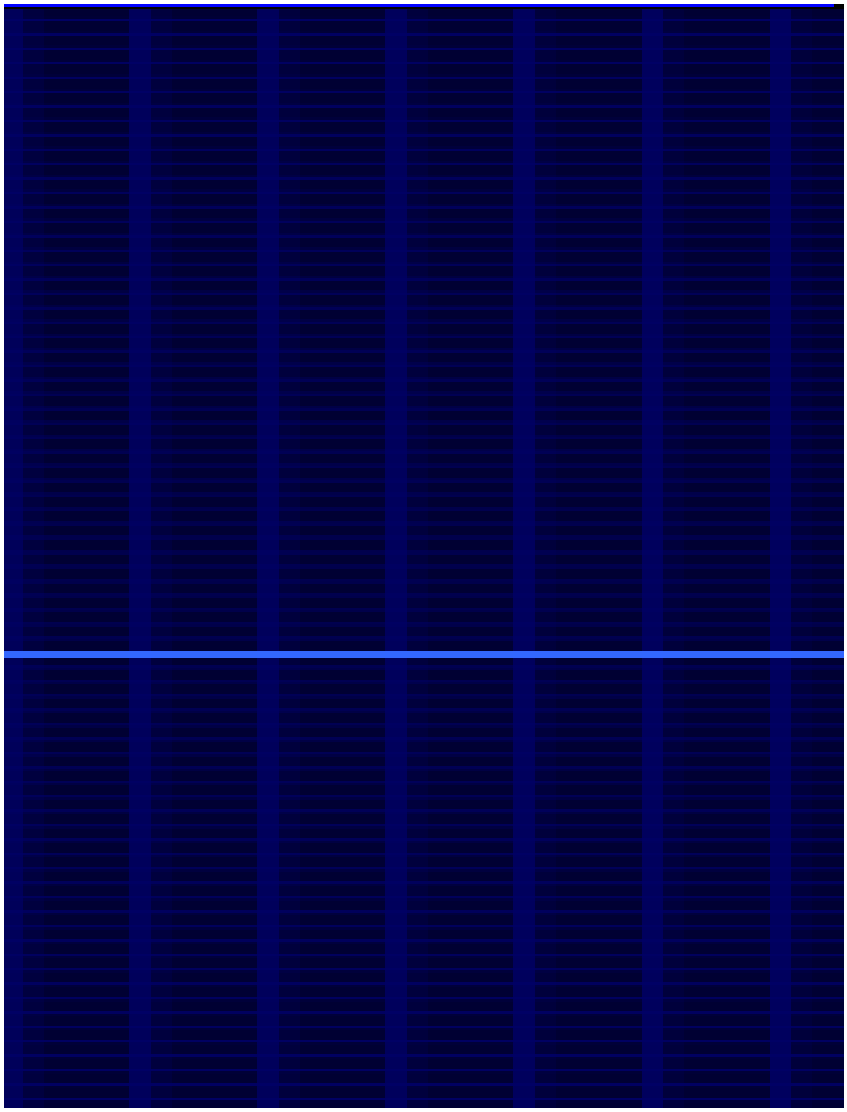


Рисунок 3.12 – Розведення топологічних елементів в ПЛІС

3.4 Тестування

Для верифікації моделі проекту використовувався випробувальний стенд на тестовій мікропрограмі, який частково наведений у лістингу 3.4.

Листинг 3.4

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.all;
USE IEEE.numeric_std.ALL;


PACKAGE filter_tb_pkg IS

  -- Type Definitions
  TYPE filter_in_data_log_type IS ARRAY (0 TO 16575) OF real;

  -- Functions
  FUNCTION to_integer( x : IN std_logic) RETURN integer;
  FUNCTION to_hex( x : IN std_logic) RETURN string;
  FUNCTION to_hex( x : IN std_logic_vector) RETURN string;
  FUNCTION to_hex( x : IN signed ) RETURN string;
  FUNCTION to_hex( x : IN unsigned ) RETURN string;
  FUNCTION to_hex( x : IN real ) RETURN string;
   FUNCTION SLICE( x : IN bit_vector; slice : In Integer) RETURN
std_logic_vector;
   FUNCTION SLICE( x : IN bit_vector; slice : In Integer) RETURN
signed;
   FUNCTION SLICE( x : IN bit_vector; slice : In Integer) RETURN
unsigned;

  -- Procedures
  PROCEDURE filter_in_data_log_procedure
    (SIGNAL clk      : IN    std_logic;
     SIGNAL reset    : IN    std_logic;
     SIGNAL rdenb    : IN    std_logic;
     SIGNAL addr     : INOUT unsigned(14 DOWNTO 0);
     SIGNAL done     : OUT   std_logic);

  PROCEDURE filter_out_procedure
    (SIGNAL clk      : IN    std_logic;
     SIGNAL reset    : IN    std_logic;
     SIGNAL rdenb    : IN    std_logic;
     SIGNAL addr     : INOUT unsigned(14 DOWNTO 0);
     SIGNAL done     : OUT   std_logic);

END filter_tb_pkg;

PACKAGE BODY filter_tb_pkg IS
  FUNCTION to_integer( x : IN std_logic) RETURN integer IS
    VARIABLE int: integer;
  BEGIN
    IF x = '0' THEN
      int := 0;
    ELSE
      int := 1;
    END IF;
    RETURN int;
  END;
```

```
  FUNCTION to_hex( x : IN std_logic_vector) RETURN string IS
    VARIABLE result  : STRING(1 TO 256); -- 1024 bits max
    VARIABLE i       : INTEGER;
    VARIABLE imod    : INTEGER;
    VARIABLE j       : INTEGER;
    VARIABLE jinc    : INTEGER;
    VARIABLE newx    : std_logic_vector(1023 DOWNTO 0);
  BEGIN
    newx := (OTHERS => '0');
    IF x'LEFT > x'RIGHT THEN
      j := x'LENGTH-1;
      jinc := -1;
    ELSE
      j := 0;
      jinc := 1;
    END IF;

    FOR i IN x'RANGE LOOP
      newx(j) := x(i);
      j := j+jinc;
    END LOOP;  -- i
    i := x'LENGTH-1;
    imod := x'LENGTH MOD 4;
    IF    imod = 1 THEN i := i+3;
    ELSIF imod = 2 THEN i := i+2;
    ELSIF imod = 3 THEN i := i+1;
    END IF;
    j := 1;
    WHILE i >= 3 LOOP
      IF    newx(i DOWNTO (i-3)) = "0000" THEN result(j) := '0';
      ELSIF newx(i DOWNTO (i-3)) = "0001" THEN result(j) := '1';
      ELSIF newx(i DOWNTO (i-3)) = "0010" THEN result(j) := '2';
      ELSIF newx(i DOWNTO (i-3)) = "0011" THEN result(j) := '3';
      ELSIF newx(i DOWNTO (i-3)) = "0100" THEN result(j) := '4';
…….
      ELSE result(j) := 'X';
      END IF;
      i := i-4;
      j := j+1;
    END LOOP;
    RETURN result(1 TO j-1);
  END;

  FUNCTION to_hex( x : IN std_logic ) RETURN string IS
  BEGIN
    RETURN std_logic'image(x);
  END;
…….



  PROCEDURE filter_in_data_log_procedure
    (SIGNAL clk      : IN    std_logic;
```

```
     SIGNAL reset    : IN    std_logic;
     SIGNAL rdenb    : IN    std_logic;
     SIGNAL addr     : INOUT unsigned(14 DOWNTO 0);
     SIGNAL done     : OUT   std_logic) IS

  BEGIN
-- Counter to generate Addr.
    IF reset = '1' THEN
      addr     <= TO_UNSIGNED(0,15);
    ELSIF clk'event and clk = '1' THEN
      IF rdenb = '1' THEN
        IF (addr = TO_UNSIGNED(16575, 15 )) THEN
          addr     <= addr;
        ELSE
          addr     <= addr + TO_UNSIGNED(1,15);
        END IF;
      ELSE
        addr <= addr;
      END IF;
    END IF;

-- Done Signal generation.
    IF reset = '1' THEN
      done <= '0';
    ELSIF (addr = TO_UNSIGNED(16575, 15 )) THEN
      done <= '1';
    ELSE
      done <= '0';
    END IF;
  END filter_in_data_log_procedure;

  PROCEDURE filter_out_procedure
    (SIGNAL clk      : IN    std_logic;
     SIGNAL reset    : IN    std_logic;
     SIGNAL rdenb    : IN    std_logic;
     SIGNAL addr     : INOUT unsigned(14 DOWNTO 0);
     SIGNAL done     : OUT   std_logic) IS

  BEGIN
-- Counter to generate Addr.
    IF reset = '1' THEN
      addr     <= TO_UNSIGNED(0,15);
    ELSIF clk'event and clk = '1' THEN
      IF rdenb = '1' THEN
        IF (addr = TO_UNSIGNED(16575, 15 )) THEN
          addr     <= addr;
        ELSE
          addr     <= addr + TO_UNSIGNED(1,15);
        END IF;
      ELSE
        addr <= addr;
      END IF;
    END IF;
```

```
-- Done Signal generation.
    IF reset = '1' THEN
      done <= '0';
    ELSIF (addr = TO_UNSIGNED(16575, 15 )) THEN
      done <= '1';
    ELSE
      done <= '0';
    END IF;
  END filter_out_procedure;

END filter_tb_pkg;
```

Достовірність отриманих результатів перевірялася моделюванням за допомогою інструментальних засобів проектування Xilinx (САПР Integrated Software Environment – ISE, пакету Xilinx System Generator for DSP та системи Matlab/Simulink).

Результати тестування наведені на рис. 3.13. З часової діаграми можна побачити, що результати фільтрації співпадають с тестовою послідовністю, що означає, що фільтр працює правильно.

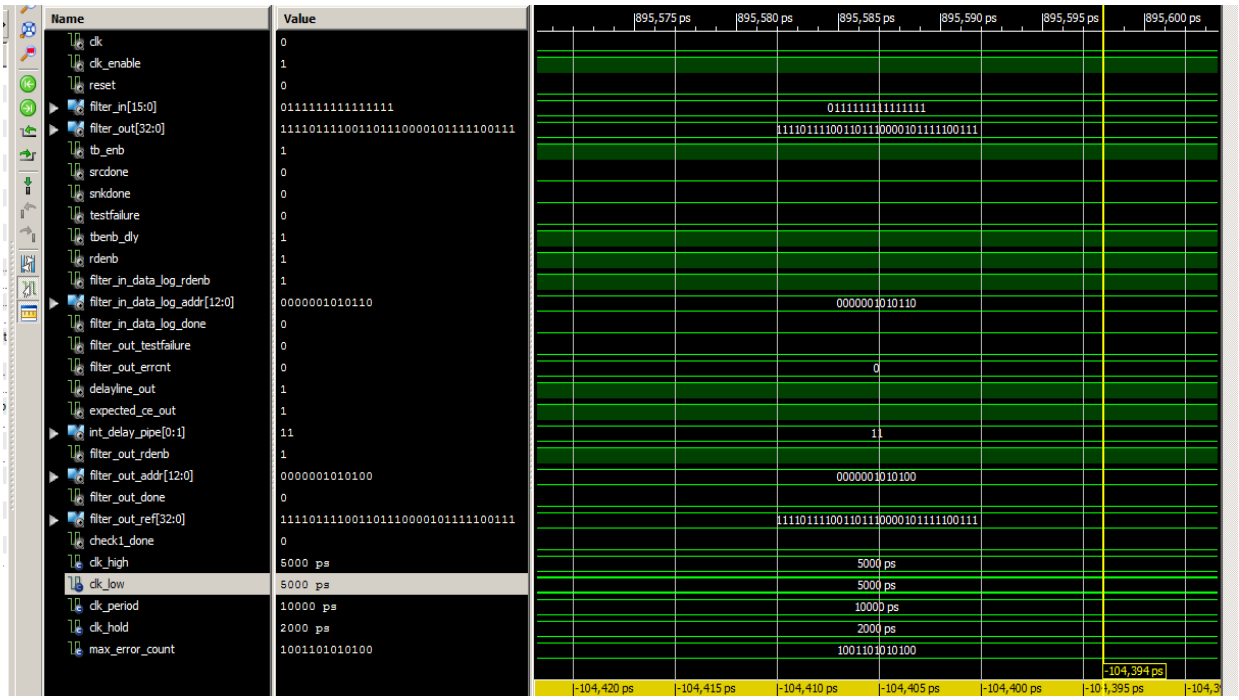


Рисунок 3.13 – Результати тестування

Також на рисунку 3.14 наведено апаратні затрати, які були отримані в середовищі Matlab та в результаті синтезу в Xilinx.

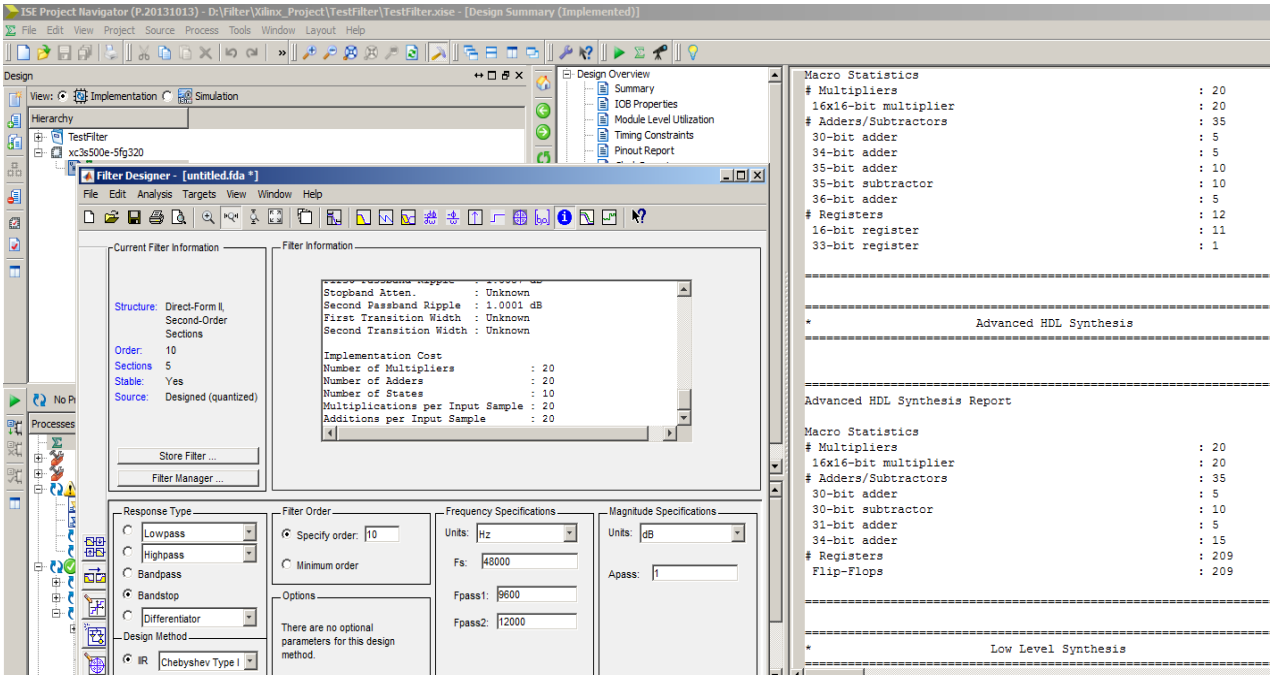


Рисунок 3.14 – Апаратні затрати

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було виконане проектування оптимального цифрового фільтру з заданими частотними характеристиками при мінімальній кількості апаратних ресурсів на основі методів проектування рекурсивних цифрових фільтрів з урахуванням основних факторів, що визначають їх реалізацію.

Для цього було проведено аналіз переваг та недоліків цифрових СІХ та НІХ фільтрів. Було сформульовані основні критерії за якими здійснюється вибіт типу цифрового фільтру.

Були проаналізовані апаратні реалізації цифрових фільтрів в DSP та ПЛІС. Було показано, що реалізація цифрового фільтру на ПЛІС обумовлено наступними перевагами: високою тактовою частотою, тобто частота роботи ПЛІС близько кількох ГГц, апаратною реалізацією помножувачів, що дозволяє множити за один такт, наявністю апаратних інтерфейсів проти DSP, гнучкістю ПЛІС під час розведення плати, що дозволяє подавати дані довільної розрядності.

Були розглянуті етапи проектування цифрових фільтрів за допомогою System Generator for DSP, MatLab і Simulink, Середовище FDATool. Це є ефективним способом вирішення задач у галузі цифрової обробки сигналів та дозволяє не вникати особливості архітектури ПЛІС, проводити проектування в основному в середовищі MatLab, отримуючи потім робочу конфігурацію ПЛІС.

Застосовуючи пакет SystemGenerator, розроблений спільно фірмами Xilinx і MathWorks, можна синтезувати потрібну системну функцію фільтра, провести моделювання його роботи, згенерувати код для програмування ПЛІС безпосередньо із середовища Simulink. Розробка фільтра за допомогою середовища Середовище FDATool дозволяє розраховувати фільтр лише задаючи його параметри, та отримуючи VHDL опис фільтру. У результаті

синтезу фільтру отримується графік характеристики згасання та поточна інформація про фільтр: структуру, порядок, стійкість та коефіцієнти фільтра. Це дозволяє проаналізувати отриманий результат та коректувати його до тих пір, доки не буде розроблено фільтр з необхідними параметрами.

Було проаналізовано аудіо файл, виявлено заваду, яку необхідно було от фільтрувати. Була розроблена модель системи, яка дозволяла тестувати систему у реальному часі, та бачити результат фільтрації на аналайзері.

Було обрано тип фільтру, метод синтезу фільтру, структуру фільтру. Було проаналізовано дві реалізації фільтру Batterworth та еліптичного, проведено порівняльний аналіз та обрано еліптичний фільтр, який при однакових заданих параметрах реалізації фільтра є більш оптимальнішим.

Було отримано VHDL опис фільтру. Проведено синтез та тестування спроектованого фільтру у середовищі пакету Xilinx.

Результати тестування показали, що результати фільтрації співпадають с тестовою послідовністю, що означає, що фільтр працює правильно.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Гоноровский, И. С. Радиотехнические цепи и сигналы [Текст]: учеб. пособие / И. С. Гоноровский, М. П. Демин. М.: Радио и связь, 1994.
2. Глинченко, А. С. Цифровая обработка сигналов [Текст]: учеб. пособие / А. С. Глинченко. 2-е изд., перераб. и доп. Красноярск: ИПЦ КГТУ, 2005.
3. Глинченко, А. С. Цифровая обработка сигналов [Текст]: учеб. пособие: в 2 ч. / А. С. Глинченко. Красноярск: ИПЦ КГТУ, 2001. Ч. 1.
4. Айфичер, Э. С. Цифровая обработка сигналов: практический подход [Текст]: пер. с англ. / Э. С. Айфичер, Б. У. Джервис. 2-е изд. М.: Издат. дом «Вильямс», 2004.
5. Смит С. Цифровая обработка сигналов. Практическое руководство для инженеров и научных работников / Стивен Смит — М. : Додэка_XXI, 2011. — 720 с.
6. Сергиенко, А. Б. Цифровая обработка сигналов [Текст]: учеб. для вузов / А. Б. Сергиенко. СПб.: Питер, 2002.
7. Сергиенко, А. Б. Цифровая обработка сигналов [Текст]: учеб. пособие / А. Б. Сергиенко. 2-е изд. СПб.: Питер, 2006.
8. Оппенгейм, А. Цифровая обработка сигналов [Текст]: пер с англ. / А. Оппенгейм, Р. Шафер. М.: Техносфера, 2006.
9. Гадзиковский В.И. Цифровая обработка сигналов [Электронный ресурс] / В.И. Гадзиковский. — Электрон. текстовые данные. — М.: СОЛОН-ПРЕСС, 2013. — 766 c. — 978-5-91359-117-3. — Режим доступа: http://www.iprbookshop.ru/26929.html
10. Давыдов А.В. Цифровая обработка сигналов: Тематические лекции. – Екатеринбург: УГГУ, ИГиГ, ГИН, Фонд электронных документов, 2005.

11. dsplib. Расчет КИХ фильтра с линейной фазочастотной характеристикой методом частотной выборки [электронный ресурс] (дата обращения: 14.10.2021).
12. User manual Discovery kit with STM32F407VG MCU. URL:ttps://www.st.com/content/ccc/resource/technical/document/user_manual/70/fe/4a/3f/e7/e1/4f/7d/DM00039084.pdf/files/DM00039084.pdf/jcr:content/translations/en.DM00039084.pdf (дата обращения: 14.04.2020)
13. Мистюков В. Г., Капитанов В. Д. Реализация высокопроизводительных сверхкомпактных КИХ-фильтров на ПЛИС // Scan Engineering Telecom, 1999.
14. Стешенко В. Б. ПЛИС фирмы Altera: элементная база, система проектирования и языки описания аппаратуры. М.: Додэка-XXI, 2002.
15. High Speed Polyphase CIC Decimation Filters [Электронный ресурс]. Режим доступа: http://www.kapik.com/publications/rcpapers/ HKYang96.pdf.
16. Анохин В., MatLab для DSP. Расчет цифровых фильтров с учетом эффектов квантования/ В. Анохин, А. Ланнэ [Електронний ресурс]. - URL: http://chipinfo.ru/literature/chipnews/200109/1.html (дата звернення 11.10.2021).
17. Проектирование цифровых систем на VHDL: Суворова Е. А., Шейнин Ю. Е. – СПб.: БХВ-Петербург, 2003. – 576 с.: ил.
18. Kang S., Lebelevici Y. CMOS Digital Integrated Circuits. Analysis and Design. Boston, McGrow-Hill, 1999.
19. Proceeding of the 9th International Conference – Mixed Design in Integrated Cirquits and Systems – Wroclaw, Poland. – 20-23 June 2002. 722p.
20. Грушвицкий Р. И., Мурсаев А. Х., Угрюмов Е.П. Проектирование систем на микросхемах программируемой логики. – СПб.: БХВ-Петербург, 2002. – 608 с.: ил.
21. Dr. Jack Horgan. FPGA Direction. August 2 - 6, 2004. From: EDACafe Monday August 9, 2004.

22. Проектирование цифровых схем с использованием языка VHDL: Учебное пособие/ В.В. Семенец, И.В. Хаханова, В.И. Хаханов. – ХНУРЭ, 2003. 492 с.
23. Novak A, Gramatova E., Ubar R. Handbook of testing of electronic systems Czech Technical University Publishing House, 2005 393 p.
24. Солонина А.И. Цифровая обработка сигналов в зеркале MatLab - СПб., БХВ-Петербург,2018, 560 с.