

ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

КАФЕДРА ЕЛЕКТРОННИХ ОБЧИСЛЮВАЛЬНИХ МАШИН

КВАЛІФІКАЦІЙНА РОБОТА

«Цифровий фільтр на FPGA з використанням VHDL»

Студент гр. КІУКІ-21-2

Бабков В.О.

Керівник

ас. Дяченко Д.О.

Харків 2025

Мета та завдання кваліфікаційної роботи

Мета кваліфікаційної роботи: розробка, моделювання та реалізація цифрового фільтра із заданими характеристиками на FPGA за допомогою мови опису апаратури VHDL.

Завдання:

- провести аналіз цифрових фільтрів та їх реалізації на FPGA;
- визначити оптимальну структуру фільтра;
- розробити математичну модель обраного фільтра;
- реалізувати архітектуру фільтра на VHDL та в MATLAB;
- провести моделювання у середовищі Active HDL.

Цифрові фільтри

ЦИФРОВІ ФІЛЬТРИ
 Класифікація за типом імпульсної характеристики

FIR ФІЛЬТРИ

(Finite Impulse Response)

ПЕРЕВАГИ

- Завжди стабільні
- Лінійна фазова характеристика
- Стійкі до квантування
- Прогнозована поведінка
- Відсутність зворотного зв'язку

НЕДОЛІКИ

- Велика кількість коефіцієнтів
- Висока ресурсомісткість
- Більша затримка
- Складність для вузьких смуг
- Більше обчислень

МАТЕМАТИЧНИЙ ОПИС

$$y(n) = \sum_{k=0}^{N-1} h(k) \cdot x(n-k)$$

k=0 до N-1 (скалярна сума)

IIR ФІЛЬТРИ

(Infinite Impulse Response)

ПЕРЕВАГИ

- Менше коефіцієнтів
- Висока ефективність
- Менша затримка
- Висока вибірковість
- Аналог аналогових фільтрів

НЕДОЛІКИ

- Ризик нестабільності
- Нелінійна фаза
- Чутливість до квантування
- Складний аналіз стабільності
- Зворотний зв'язок

МАТЕМАТИЧНИЙ ОПИС

$$y(n) = \sum b(k) \cdot x(n-k) - \sum a(k) \cdot y(n-k)$$

Прямий зв'язок + Зворотний зв'язок

ДОДАТКОВА КЛАСИФІКАЦІЯ ЦИФРОВИХ ФІЛЬТРІВ

ЗА ЧАСТОТНИМ ПРИЗНАЧЕННЯМ

- Низькочастотні (НЧФ)
- Високочастотні (ВЧФ)
- Смугові (СФ)
- Загоровидувальні (ЗФ)
- Воєнголосні

ЗА СПОСОБОМ РЕАЛІЗАЦІЇ

- Прямий форми ІІІ
- Каскадна форма
- Паралельна форма
- Решеткова структура
- Хвильові цифрові фільтри

ЗА МЕТОДОМ РОЗРОБКИ

- Віконні методи
- Частотна дискретизація
- Оптиміальні методи
- Більшій трансформаци
- Інваріантні імпульсна х-ка

КЛЮЧОВІ ПАРАМЕТРИ

- Сигнал пропускання
- Крутість фронту
- Затримка сигналу
- Стабільність
- Ресурсомісткість

3

Особливості реалізації цифрових фільтрів на ПЛІС

РЕАЛІЗАЦІЯ ЦИФРОВИХ ФІЛЬТРІВ НА FPGA

FPGA vs ПРОЦЕСОРИ ЗАГАЛЬНОГО ПРИЗНАЧЕННЯ

ПРОЦЕСОР (CPU)	FPGA (ПЛІС)
<ul style="list-style-type: none"> • Послідовне виконання інструкцій • Центральний процесор керує всім • Обмежена пропускова здатність • Недетермінованість виконання • Вища затримка обробки • Необхідність ОС та драйверів • Більше енергоспоживання • Складність real-time обробки • Обмеження архітектури процесора • Сильне використання ресурсів 	<ul style="list-style-type: none"> • Паралельні обчислення • Апаратно орієнтовані структури • Висока пропускова здатність • Детермінованість системи • Мінімальні затримки (latency) • Прямий доступ до апаратури • Нижче енергоспоживання • Гарантована real-time обробка • Гнучкість архітектури • Виділені ресурси для кожної задачі

СПЕЦІАЛІЗОВАНІ РЕСУРСИ FPGA

DSP SLICES (БЛОКИ DSP)	БЛОКИ ПАМ'ЯТІ
<ul style="list-style-type: none"> • MAC блоки (Multiply-Accumulate) • 25x18 біт множення (X18x25) • 48-бітні суматори/акумулятори • Pipeline глибина 3-4 такти 	<ul style="list-style-type: none"> • BRAM: 36 Кбіт блоки (X18x36) • Розподілена пам'ять, LUT як RAM • Звуши регістри (SRF) • Буферизація коефіцієнтів та даних

АРХІТЕКТУРИ РЕАЛІЗАЦІЇ FIR-ФІЛЬТРІВ

<h4 style="text-align: center;">ПОСЛІДОВНА АРХІТЕКТУРА</h4> <p style="font-size: x-small;">Структурна схема:</p>  <p style="font-size: x-small;">Характеристики:</p> <ul style="list-style-type: none"> • Один DSP slice • N тактів на вибірку (N - порядок фільтра) • Мінімальні ресурси • Частота семплення = fclk/N 	<h4 style="text-align: center;">ПОВНІСТЮ ПАРАЛЕЛЬНА АРХІТЕКТУРА</h4> <p style="font-size: x-small;">Структурна схема:</p>  <p style="font-size: x-small;">Характеристики:</p> <ul style="list-style-type: none"> • N DSP slices (N множення) • Один такт на вибірку 	<h4 style="text-align: center;">КОНВЕЕРНА АРХІТЕКТУРА</h4> <p style="font-size: x-small;">Pipeline стадії:</p>  <p style="font-size: x-small;">Характеристики:</p> <ul style="list-style-type: none"> • Частота семплення = fclk • Затримка = кількість стадій • Балансування ресурсів/швидкості • Вища тактова частота
--	---	---

ТЕХНІЧНІ ПАРАМЕТРИ ТА ОБМЕЖЕННЯ

ЧАСОВІ ХАРАКТЕРИСТИКИ	АРИФМЕТИЧНА ТОЧІСНІСТЬ	ВИКОРИСТАННЯ РЕСУРСІВ	ІНТЕРФЕЙСИ
<ul style="list-style-type: none"> • Clock period (Tclk) = 1/fclk • Setup time (Tsu) = 0.5-2 ns • Hold time (Th) = 0.1-0.5 ns • Propagation delay (Tpd) • Clock skew < 0.5 ns • Timing slack > 0 	<ul style="list-style-type: none"> • Фіксована точність (Qm, n формат) • Розрядність входу: 8-16 біт • Розрядність коефіцієнтів: 12-18 біт • Внутрішня розрядність: 48 біт • Двійковий додатковий код • Контроль переповнення 	<ul style="list-style-type: none"> • DSP48E slices: 1 на множення • BRAM: коефіцієнти та буфери • LUT: керує логіка • FF: pipeline регістри • I/O: AXI-Stream інтерфейси • Clock: глобальні мережі 	<ul style="list-style-type: none"> • AXI4-Stream • Avalon-ST • FIFO інтерфейси • Handshaking протоколи • Clock domain crossing • Backpressure підтримка

4

Методи синтезу цифрових фільтрів

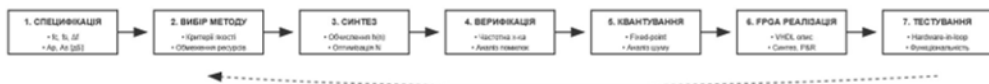
МЕТОДИ СИНТЕЗУ FIR-ФІЛЬТРІВ

<p>МЕТОД ПРЯМОГО ОБРАЗАННЯ Frequency Domain Translation</p> <p>Математичний опис: $H(z) \rightarrow FFT \rightarrow H(\omega) \rightarrow \text{truncate} \rightarrow h_n$ $N(z) = \sum_{n=0}^{N-1} h_n z^{-n}$</p> <p>Характеристика: • Ефект Гіббса на краях • Основа для вікнових методів • Нормована $K \rightarrow$ облімана</p>	<p>МЕТОД ВІКОН Windowing Method</p> <p>Типи вікон: • Rectangular: $w(n) = 1$ • Hamming: $w(n) = 0.54 - 0.46 \cos(2\pi n/N)$ • Hanning: $w(n) = 0.5(1 - \cos(2\pi n/N))$ • Blackman: 3-х лоп. cosine series</p> <p>Попереджені параметри: Rectangular: $\Delta\omega = 0.91\pi, A_s = 21\text{dB}$ Hamming: $\Delta\omega = 3.30\pi, A_s = 53\text{dB}$ Blackman: $\Delta\omega = 5.30\pi, A_s = 74\text{dB}$</p>	<p>ОПТИМАЛЬНИЙ СИНТЕЗ Remez-McClellan / Chebyshev</p> <p>Алгоритм Річеза: • Мінімальна оптимізація • Планові коефіцієнти • Максимальна селективність • Еквіріпні характеристики</p> <p>Функція колекції: $E(z) = W(z)H(z) - H_d(z)$ or ϵ bands</p>
<p>МЕТОД НАЙМЕНШИХ КВАДРАТІВ Least Squares Method</p> <p>Цільова функція: $J = \sum_{n=0}^{N-1} W(z)H(z) - H_d(z) ^2$</p> <p>Переваги: • Гладка характеристика • Невеликі пульсації • Стабільність сигналу • Малий рівень фронт</p>	<p>ЛІНІЙНЕ ПРОГРАМУВАННЯ Linear Programming Method</p> <p>Формулювання задачі: Minimize: σ^2 Subject to: $Ax \leq b$ де x - коефіцієнти фільтра</p> <p>Особливості: • Універсальність • Гнучке обмеження • Висока складність • Рідко використовується</p>	<p>СПЕКТРАЛЬНА ФАКТОРИЗАЦІЯ Spectral Factorization Method</p> <p>Принцип: $E(z) = W(z)H(z)$ $S(z) = H(z)H^*(z)$</p> <p>Застосування: • Адаптивні фільтри • Спеціальні процеси • Задачі PSD • Noise shaping</p>

ПОРІВНЯЛЬНА ТАБЛИЦЯ МЕТОДІВ

МЕТОД	СКЛАДНОСТЬ	СЕЛЕКТИВНІСТЬ	ПУЛЬСАЦІЇ	ФРОНТ ЗРІЗУ	СТАБІЛЬНІСТЬ	ГНУЧІСТЬ	ЗАСТОСУВАННЯ	FPGA
Вікновий	Низька	Середня	Залежить від вікна	Широкий	Висока	Середня	Загальне	Відносно
Річеза-МакКлеллана	Висока	Максимальна	Еквіріпні	Належний	Висока	Середня	Критичні системи	Дуже
Найменші квадрати	Середня	Середня	Низька	Широкий	Дуже висока	Висока	Аудіо, зображення	Дуже
Лінійне програмування	Дуже висока	Намагнутість	Контрольовані	Намагнутість	Середня	Максимальна	Спеціальні задачі	Складно
Спектр факторизація	Висока	Спеціальна	Залежить від PSD	Спеціальний	Середня	Низька	Адаптивні, PSD	Середня
Пряме образання	Найвища	Низька	Ефект Гіббса	Широкий	Висока	Низька	Широке застосування	Відносно

ПОСЛІДОВНІСТЬ ПРОЕКТУВАННЯ FIR-ФІЛЬТРА

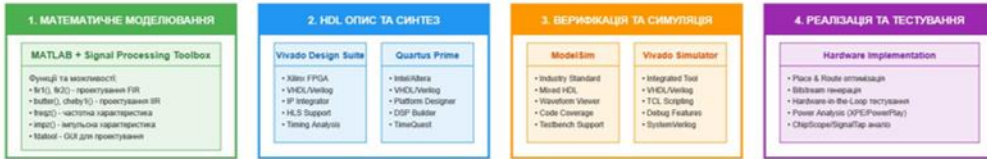


5

Огляд сучасних інструментів для проектування фільтрів на VHDL

ІНСТРУМЕНТАРІЙ ДЛЯ ПРОЕКТУВАННЯ ЦИФРОВИХ ФІЛЬТРІВ НА VHDL

ЕТАПИ ЖИТТЄВОГО ЦИКЛУ РОЗРОБКИ



ДЕТАЛЬНЕ ПОРІВНЯЛЬНЕ ІНСТРУМЕНТІВ



ІНТЕГРОВАННИЙ РОБОЧИЙ ПРОЦЕС



6

Діаграма процесу розрахунку коефіцієнтів FIR-фільтра

ВИХІДНІ ПАРАМЕТРИ ПРОЕКТУВАННЯ

Частота дискретизації (fs): 48 кГц Частота зразка (fc): 8 кГц Нормалізована частота: fo/fn = 8/24 = 0.333
 Порядок фільтра: N = 20 (21 коефіцієнт) Тип вікна: Геммінга (Hamming) Розрядність: 16 біт (signed)

КРОК 1: ПРОЕКТУВАННЯ В МАТЛАБ

МАТЛАБ КОД

```
fs = 48000; % Частота дискретизації
fc = 8000; % Частота зразка
N = 20; % Порядок фільтра
fn = fc/(fs/2); % Нормалізація
h = fir1(N, fn, 'low', hamming(N));
```

ХАРАКТЕРИСТИКИ ФІЛЬТРА

- Тип: Низькочастотний FIR
- Симетрична імпульсна характеристика
- Плавна фазова характеристика
- Глибина нульові лобів -43дБ
- Ширина пропуску -0.064т

КРОК 2: ПЕРЕТВОРЕННЯ В ФІКОВАНУ ТОЧКУ

МАСШТАБУВАННЯ

```
scale_factor = 2^15 - 1; % 32767
max_coeff = max(abs(h));
h_scaled = h * scale_factor / max_coeff;
h_int16 = round(h_scaled);
h_binary = dec2bin(h_int16, 16);
```

АНАЛІЗ КВАНТУВАННЯ

- Формат: Q8 15 (signed)
- Діапазон: -32768 до +32767
- SNR квантування: -96 дБ
- Плавка округлення: до 5 LSB
- Two's complement кодування

РОЗРАХОВАНІ КОЕФІЦІЄНТИ ФІЛЬТРА

INDEX	FLOAT VALUE	SCALED INT16	HEX	BINARY	VHDL CONSTANT	SYMMETRY
0	-0.0087	-285	FEE3	11111101100011	"FEE3"	N(20)
1	0.0000	0	0000	00000000000000	"0000"	N(19)
2	0.0223	730	92DA	000001011011010	"92DA"	N(18)
10	0.5746	5758	1656	000011000000010	"1656"	CENTER
18	0.0223	730	92DA	000001011011010	"92DA"	N(2)
20	-0.0087	-285	FEE3	11111101100011	"FEE3"	N(0)

СИМЕТРИЯ КОЕФІЦІЄНТІВ

КРОК 3: РЕАЛІЗАЦІЯ В VHDL

VHDL КОНСТАНТИ

```
type coeff_array is array(0 to 20) of signed(15 downto 0);
constant FIR_COEFFS : coeff_array := (
    "FEE3", "0000", "92DA", "0000", "F7D5",
    "0000", "8676", "0000", "F732", "0000",
    "1656", -- center coefficient
    -- symmetric part follows...
```

ПЕРЕВАГИ РЕАЛІЗАЦІЇ

- Використання стандартної бібліотеки numeric_std
- Відсутність операцій з плаваючою комою
- Ефективне використання DSP blocks FPGA
- Симетричність коефіцієнтів -- економія пайплет
- Константи в ROM або ж генератор
- Проста масштабування та параметризації

Моделювання в Matlab

untitled.m | untitled2.m | +

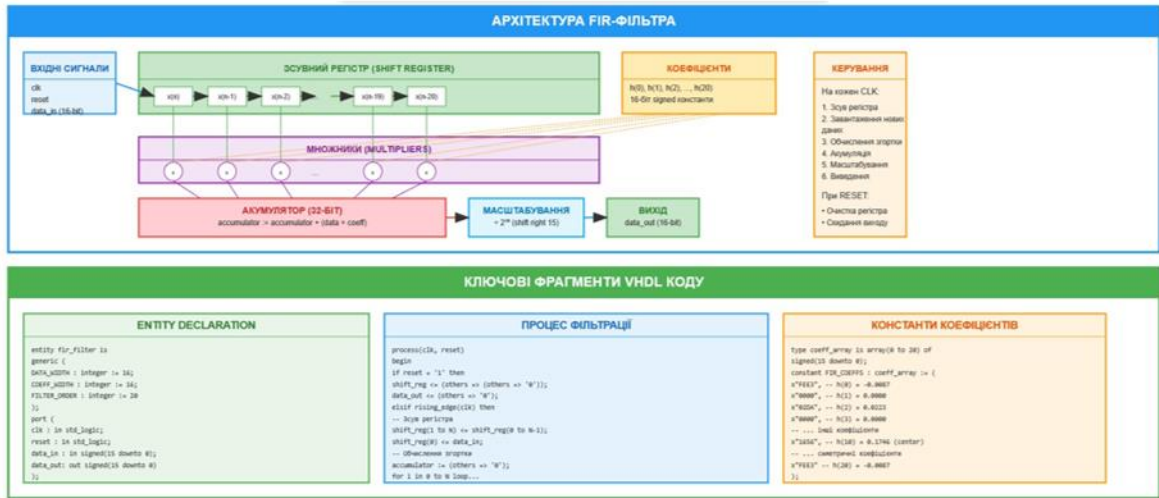
```
1 % FIR-фільтр на 20 коефіцієнтах при 48кГц зразковій частоті
2 fs = 48000; % Частота дискретизації
3 fc = 8000; % Частота зразка
4 N = 20; % Порядок фільтра
5 fn = fc/(fs/2); % Нормалізація
6 h = fir1(N, fn, 'low', hamming(N));
7 % Коэффициенты фильтра
8 % Коэффициенты фильтра после нормализации
9 % Коэффициенты фильтра после масштабирования
10 % Коэффициенты фильтра после округления
11 % Коэффициенты фильтра после квантования
12 % Коэффициенты фильтра после кодирования
13 % Коэффициенты фильтра после кодирования
14 % Коэффициенты фильтра после кодирования
15 % Коэффициенты фильтра после кодирования
16 % Коэффициенты фильтра после кодирования
17 % Коэффициенты фильтра после кодирования
18 % Коэффициенты фильтра после кодирования
19 % Коэффициенты фильтра после кодирования
20 % Коэффициенты фильтра после кодирования
21 % Коэффициенты фильтра после кодирования
22 % Коэффициенты фильтра после кодирования
23 % Коэффициенты фильтра после кодирования
24 % Коэффициенты фильтра после кодирования
25 % Коэффициенты фильтра после кодирования
26 % Коэффициенты фильтра после кодирования
27 % Коэффициенты фильтра после кодирования
28 % Коэффициенты фильтра после кодирования
29 % Коэффициенты фильтра после кодирования
30 % Коэффициенты фильтра после кодирования
31 % Коэффициенты фильтра после кодирования
32 % Коэффициенты фильтра после кодирования
33 % Коэффициенты фильтра после кодирования
34 % Коэффициенты фильтра после кодирования
35 % Коэффициенты фильтра после кодирования
36 % Коэффициенты фильтра после кодирования
37 % Коэффициенты фильтра после кодирования
38 % Коэффициенты фильтра после кодирования
39 % Коэффициенты фильтра после кодирования
40 % Коэффициенты фильтра после кодирования
41 % Коэффициенты фильтра после кодирования
42 % Коэффициенты фильтра после кодирования
43 % Коэффициенты фильтра после кодирования
44 % Коэффициенты фильтра после кодирования
45 % Коэффициенты фильтра после кодирования
46 % Коэффициенты фильтра после кодирования
47 % Коэффициенты фильтра после кодирования
48 % Коэффициенты фильтра после кодирования
49 % Коэффициенты фильтра после кодирования
50 % Коэффициенты фильтра после кодирования
51 % Коэффициенты фильтра после кодирования
52 % Коэффициенты фильтра после кодирования
53 % Коэффициенты фильтра после кодирования
54 % Коэффициенты фильтра после кодирования
55 % Коэффициенты фильтра после кодирования
56 % Коэффициенты фильтра после кодирования
57 % Коэффициенты фильтра после кодирования
58 % Коэффициенты фильтра после кодирования
59 % Коэффициенты фильтра после кодирования
60 % Коэффициенты фильтра после кодирования
61 % Коэффициенты фильтра после кодирования
62 % Коэффициенты фильтра после кодирования
63 % Коэффициенты фильтра после кодирования
64 % Коэффициенты фильтра после кодирования
65 % Коэффициенты фильтра после кодирования
66 % Коэффициенты фильтра после кодирования
67 % Коэффициенты фильтра после кодирования
68 % Коэффициенты фильтра после кодирования
69 % Коэффициенты фильтра после кодирования
70 % Коэффициенты фильтра после кодирования
71 % Коэффициенты фильтра после кодирования
72 % Коэффициенты фильтра после кодирования
73 % Коэффициенты фильтра после кодирования
74 % Коэффициенты фильтра после кодирования
75 % Коэффициенты фильтра после кодирования
76 % Коэффициенты фильтра после кодирования
77 % Коэффициенты фильтра после кодирования
78 % Коэффициенты фильтра после кодирования
79 % Коэффициенты фильтра после кодирования
80 % Коэффициенты фильтра после кодирования
81 % Коэффициенты фильтра после кодирования
82 % Коэффициенты фильтра после кодирования
83 % Коэффициенты фильтра после кодирования
84 % Коэффициенты фильтра после кодирования
85 % Коэффициенты фильтра после кодирования
86 % Коэффициенты фильтра после кодирования
87 % Коэффициенты фильтра после кодирования
88 % Коэффициенты фильтра после кодирования
89 % Коэффициенты фильтра после кодирования
90 % Коэффициенты фильтра после кодирования
91 % Коэффициенты фильтра после кодирования
92 % Коэффициенты фильтра после кодирования
93 % Коэффициенты фильтра после кодирования
94 % Коэффициенты фильтра после кодирования
95 % Коэффициенты фильтра после кодирования
96 % Коэффициенты фильтра после кодирования
97 % Коэффициенты фильтра после кодирования
98 % Коэффициенты фильтра после кодирования
99 % Коэффициенты фильтра после кодирования
100 % Коэффициенты фильтра после кодирования
```

Command Window

```
to_signed(6843, 16),
to_signed(4126, 16),
to_signed(0, 16),
to_signed(-1543, 16),
to_signed(-977, 16),
to_signed(0, 16),
to_signed(348, 16),
to_signed(190, 16),
to_signed(0, 16),
to_signed(-72, 16)
```

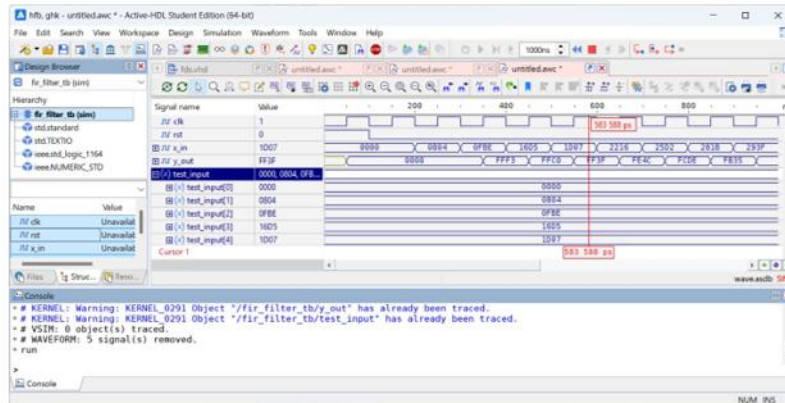
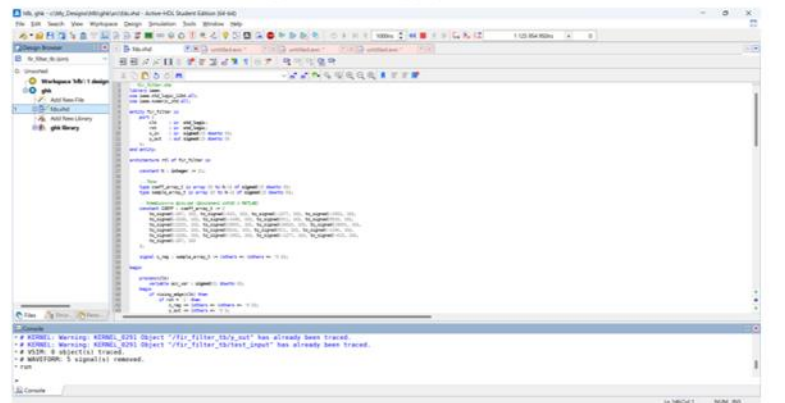
Figure 1.x

Архітектура розробленого цифрового фільтра



9

Реалізація фільтра на VHDL



10

Висновки

В результаті виконання кваліфікаційної роботи було досліджено теоретичні та практичні аспекти цифрової фільтрації сигналів з використанням фільтрів з кінцевою імпульсною характеристикою, а також здійснено розробку, моделювання та тестування такого фільтра в середовищі мовою VHDL. Особливу увагу було приділено процесу перенесення математичної моделі FIR-фільтра, реалізованої в середовищі MATLAB, до формату, придатного для синтезу на програмуванні логічній інтегральній схемі. Вибрані коефіцієнти фільтра було масштабовано у формат фіксованої точності, що дозволило ефективно реалізувати операції згортки без застосування чисел з плаваючою комою.

Моделювання архітектури фільтра в середовищі VHDL підтвердило коректність алгоритму обробки сигналу, зокрема завдяки використанню зсувного регістру для збереження історії входів і змінної акумуляції результатів згортки з фільтрувальними коефіцієнтами. Для перевірки працездатності було створено тестове середовище, яке забезпечило генерацію тактового сигналу, ініціалізацію та подачу контрольованого вхідного сигналу у вигляді квантизованої синусоїди. Отримані результати симуляції свідчать про правильну роботу схеми та відповідність її амплітудно-частотної характеристики попередньо розрахованій у MATLAB.

ДОДАТОК Б

Програмний код

Б.1 Лістинг коду MATLAB

```

% FIR-фільтр на 21 коефіцієнт для FPGA (частота зрізу 8 кГц при
Fs = 48 кГц)

clc;
clear;

% Параметри
Fs = 48000;           % Частота дискретизації, Гц
Fc = 8000;           % Частота зрізу, Гц
N = 21;              % Кількість коефіцієнтів (порядок + 1)

% Нормалізована частота
Wn = Fc / (Fs/2);

% Розрахунок коефіцієнтів методом вікна Геммінга
h = fir1(N-1, Wn, 'low', hamming(N));

% Візуалізація частотної характеристики
fvtool(h, 'Fs', Fs);
title('АЧХ FIR-фільтра');

% Масштабування до фіксованої точності (16 біт)
scale_factor = 2^15;           % 32768
h_fixed = round(h * scale_factor); % цілі знакові значення

% Виведення коефіцієнтів у MATLAB
disp('Коефіцієнти фільтра у фіксованій точності (16 біт):');
disp(h_fixed.);

% Генерація VHDL-масиву
fprintf('\n-- VHDL array of FIR coefficients (16-bit signed
integers)\n');
fprintf('constant COEFF : array (0 to %d) of signed(15 downto 0)
:= (\n', N-1);
for i = 1:N
    if i < N
        fprintf('        to_signed(%d, 16),\n', h_fixed(i));
    else
        fprintf('        to_signed(%d, 16)\n', h_fixed(i));
    end
end
end
fprintf(');\n');

```

Б.2 Лістинг коду VHDL

```

-- fir_filter.vhd
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity fir_filter is
    port (
        clk      : in  std_logic;
        rst      : in  std_logic;
        x_in     : in  signed(15 downto 0);
        y_out    : out signed(15 downto 0)
    );
end entity;

architecture rtl of fir_filter is

    constant N : integer := 21;

    -- Coefficients
    type coeff_array_t is array (0 to N-1) of signed(15 downto 0);
    type sample_array_t is array (0 to N-1) of signed(15 downto 0);

    -- Coefficients from MATLAB
    constant COEFF : coeff_array_t := (
        to_signed(-207, 16), to_signed(-615, 16), to_signed(-1277, 16),
        to_signed(-1952, 16), to_signed(-2226, 16), to_signed(-1436, 16),
        to_signed(911, 16), to_signed(5616, 16),
        to_signed(12233, 16), to_signed(19691, 16),
        to_signed(24810, 16), to_signed(19691, 16),
        to_signed(12233, 16), to_signed(5616, 16),
        to_signed(911, 16), to_signed(-1436, 16),
        to_signed(-2226, 16), to_signed(-1952, 16), to_signed(-1277, 16),
        to_signed(-615, 16),
        to_signed(-207, 16)
    );

    signal x_reg : sample_array_t := (others => (others => '0'));

begin

    process(clk)
        variable acc_var : signed(31 downto 0);
    begin
        if rising_edge(clk) then
            if rst = '1' then
                x_reg <= (others => (others => '0'));
            end if;
        end if;
    end process;
end architecture;

```

```

        y_out <= (others => '0');
    else
        for i in N-1 downto 1 loop
            x_reg(i) <= x_reg(i - 1);
        end loop;
        x_reg(0) <= x_in;

        acc_var := (others => '0');
        for i in 0 to N-1 loop
            acc_var := acc_var + resize(x_reg(i) *
COEFF(i), 32);
        end loop;

        y_out <= resize(acc_var(30 downto 15), 16);
    end if;
end if;
end process;

end architecture;

-- fir_filter_tb.vhd
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity fir_filter_tb is
end entity;

architecture sim of fir_filter_tb is

    signal clk    : std_logic := '0';
    signal rst    : std_logic := '1';
    signal x_in   : signed(15 downto 0) := (others => '0');
    signal y_out  : signed(15 downto 0);

    -- Oãñòîâèè ñèãîàè: ñèíóñî;äà ç ÷ãñòîðîþ ~1/20 îãð³îä³â
    type test_array_t is array(0 to 99) of signed(15 downto 0);
    constant test_input : test_array_t := (
        0 => to_signed(0, 16),
        1 => to_signed(2052, 16),
        2 => to_signed(4030, 16),
        3 => to_signed(5845, 16),
        4 => to_signed(7431, 16),
        5 => to_signed(8726, 16),
        6 => to_signed(9682, 16),
        7 => to_signed(10267, 16),
        8 => to_signed(10559, 16),
        9 => to_signed(10559, 16),
        10 => to_signed(10267, 16),
        11 => to_signed(9682, 16),
        12 => to_signed(8726, 16),
        13 => to_signed(7431, 16),
        14 => to_signed(5845, 16),

```

```

15 => to_signed(4030, 16),
16 => to_signed(2052, 16),
17 => to_signed(0, 16),
18 => to_signed(-2052, 16),
19 => to_signed(-4030, 16),
20 => to_signed(-5845, 16),
21 => to_signed(-7431, 16),
22 => to_signed(-8726, 16),
23 => to_signed(-9682, 16),
24 => to_signed(-10267, 16),
25 => to_signed(-10559, 16),
26 => to_signed(-10559, 16),
27 => to_signed(-10267, 16),
28 => to_signed(-9682, 16),
29 => to_signed(-8726, 16),
30 => to_signed(-7431, 16),
31 => to_signed(-5845, 16),
32 => to_signed(-4030, 16),
33 => to_signed(-2052, 16),
others => to_signed(0, 16)
);

begin

-- ²íñòàíö³pâàííý ô³ëüòðà
DUT: entity work.fir_filter
    port map (
        clk    => clk,
        rst    => rst,
        x_in   => x_in,
        y_out  => y_out
    );

-- Ñãíáððàö³ý òàèèòîâîâîî ñèãíàèö
clk <= not clk after 50 ns;

-- Òãñòîââéé îðîòãñ
stimulus: process
begin
    wait for 100 ns; -- òðèèàðèè reset
    rst <= '0';

    for i in 0 to 99 loop
        x_in <= test_input(i);
        wait for 100 ns;
    end loop;

    wait;
end process;

end architecture;
```