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

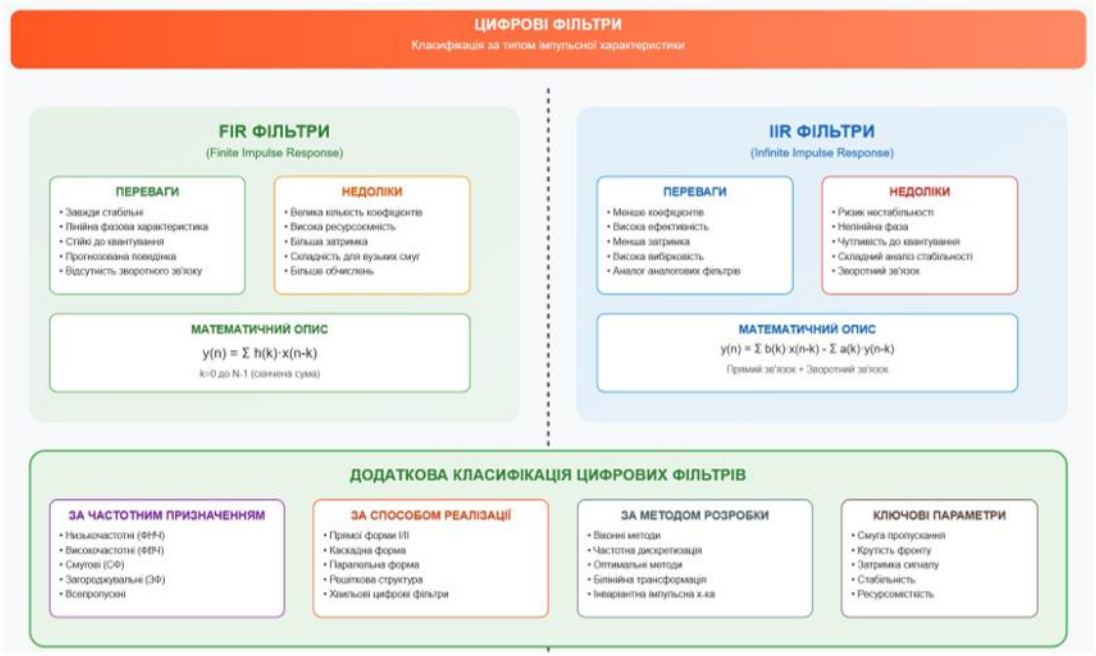
Мета та завдання кваліфікаційної роботи

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

Завдання:

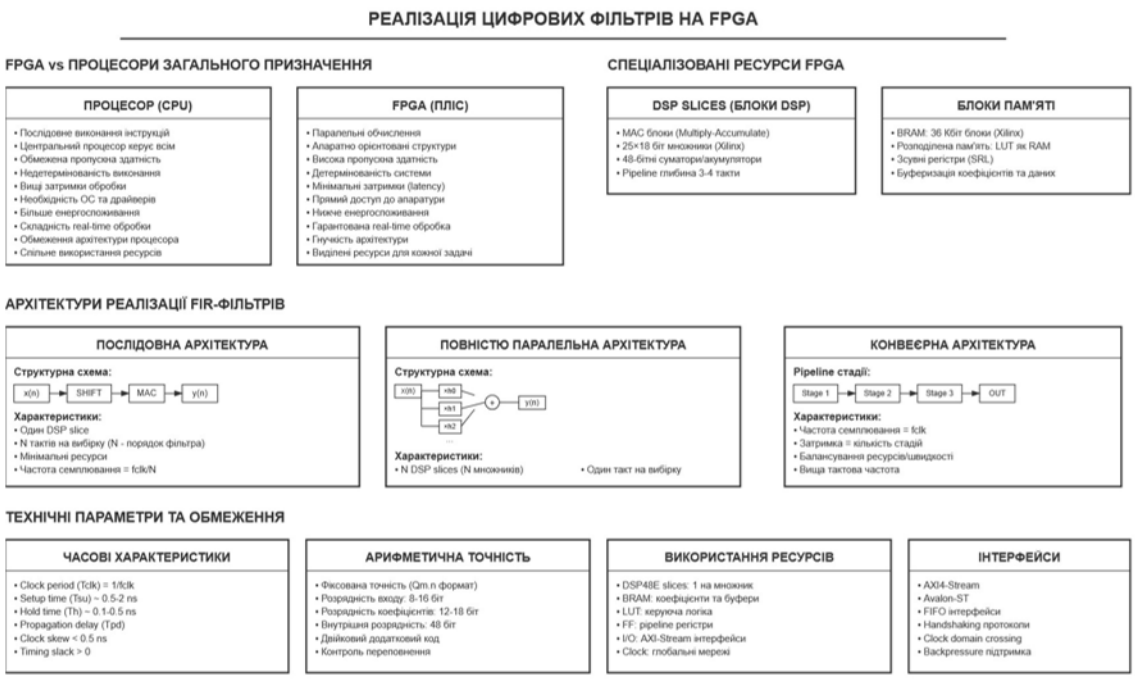
- провести аналіз цифрових фільтрів та їх реалізації на FPGA;
- визначити оптимальну структуру фільтра;
- розробити математичну модель обраного фільтра;
- реалізувати архітектуру фільтра на VHDL та в MATLAB;
- провести моделювання у середовищі Active HDL.

Цифрові фільтри



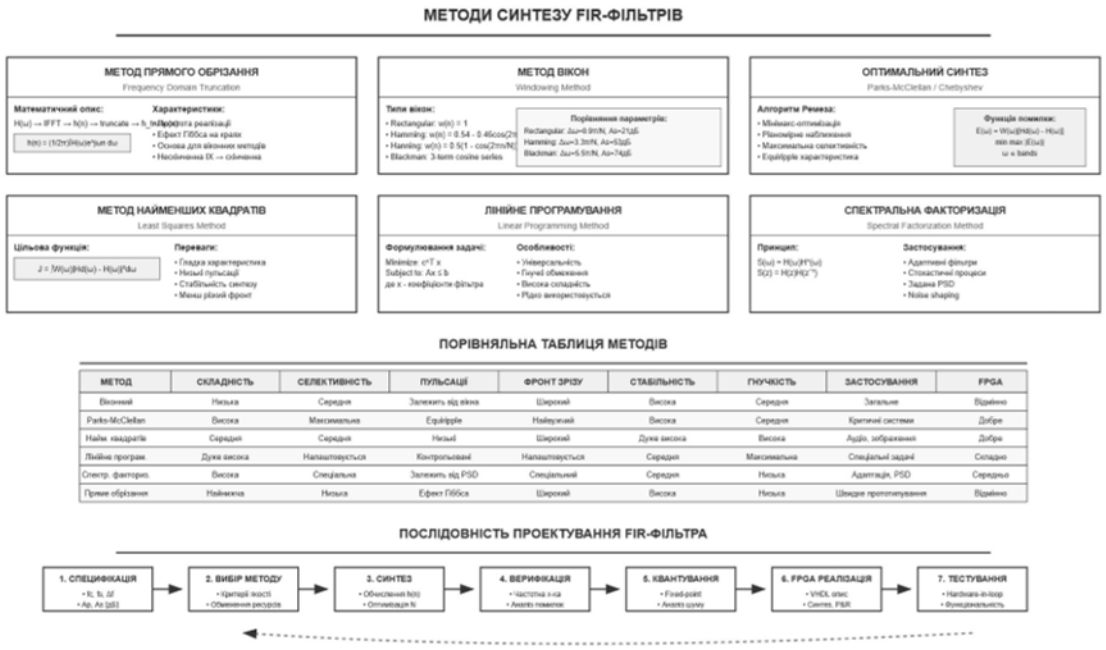
3

Особливості реалізації цифрових фільтрів на ПЛІС



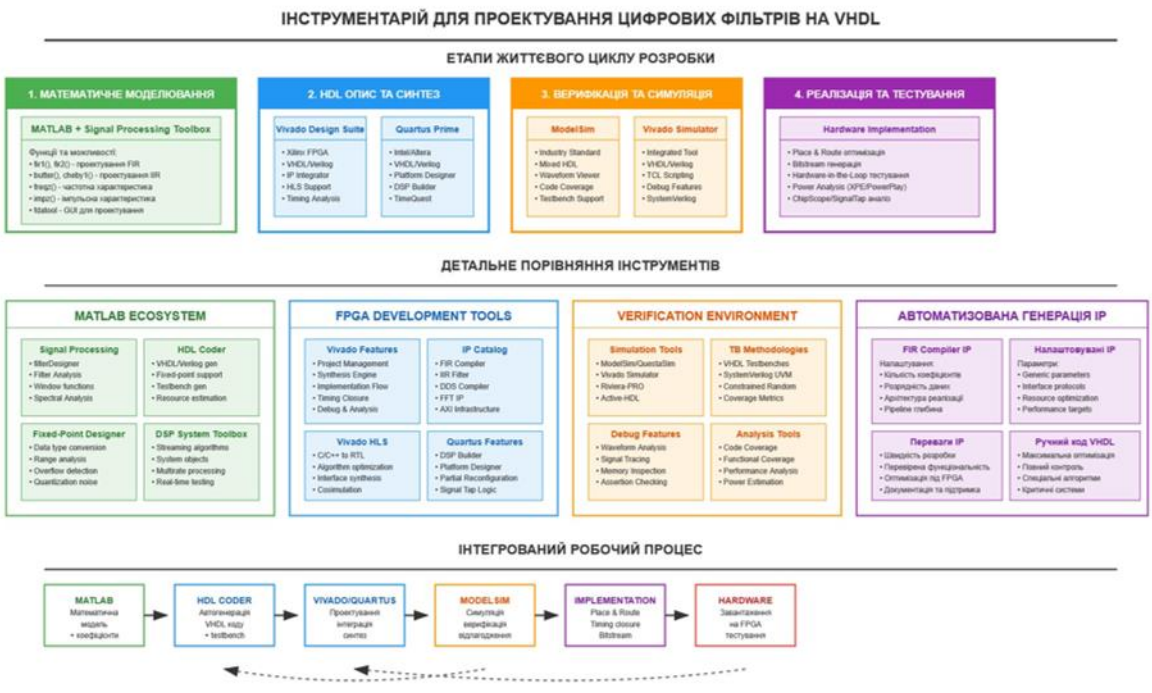
4

Методи синтезу цифрових фільтрів



5

Огляд сучасних інструментів для проєктування фільтрів на VHDL



6

Діаграма процесу розрахунку коефіцієнтів FIR-фільтра

ВИХІДНІ ПАРАМЕТРИ ПРОЕКТУВАННЯ

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

КРОК 1: ПРОЕКТУВАННЯ В MATLAB

MATLAB КОД

```
fs = 48000; % Частота дискретизації
fc = 8000; % Частота зрізу
N = 20; % Порядок фільтра
Wn = fc/(fs/2); % Нормалізація
h = fir1(N, Wn, 'low', hamming(N+1));
```

ХАРАКТЕРИСТИКИ ФІЛЬТРА

- Тип: Низькочастотний FIR
- Симетрична імпульсна характеристика
- Лінійна фазова характеристика
- Геммінг: побічні пелюстки -43дБ
- Ширина переходу: ~0.064π

КРОК 2: ПЕРЕТВОРЕННЯ В ФІКСОВАНУ 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: Q0.15 (signed)
- Діапазон: -32768 до +32767
- SNR квантування: ~96 дБ
- Помилка округлення: ±0.5 LSB
- Two's complement кодування

РОЗРАХОВАНІ КОЕФІЦІЄНТИ ФІЛЬТРА

INDEX	FLOAT VALUE	SCALED INT16	HEX	BINARY	VHDL CONSTANT	SYMMETRY
0	-0.0087	-285	FEE3	1111111011100011	x"FEE3"	h(20)
1	0.0000	0	0000	0000000000000000	x"0000"	h(19)
2	0.0223	730	02DA	0000001011011010	x"02DA"	h(18)
10	0.1746	5718	1656	0001011001010110	x"1656"	CENTER
18	0.0223	730	02DA	0000001011011010	x"02DA"	h(2)
20	-0.0087	-285	FEE3	1111111011100011	x"FEE3"	h(0)

СИМЕТРІЯ КОЕФІЦІЄНТІВ

КРОК 3: РЕАЛІЗАЦІЯ В VHDL

VHDL КОНСТАНТИ

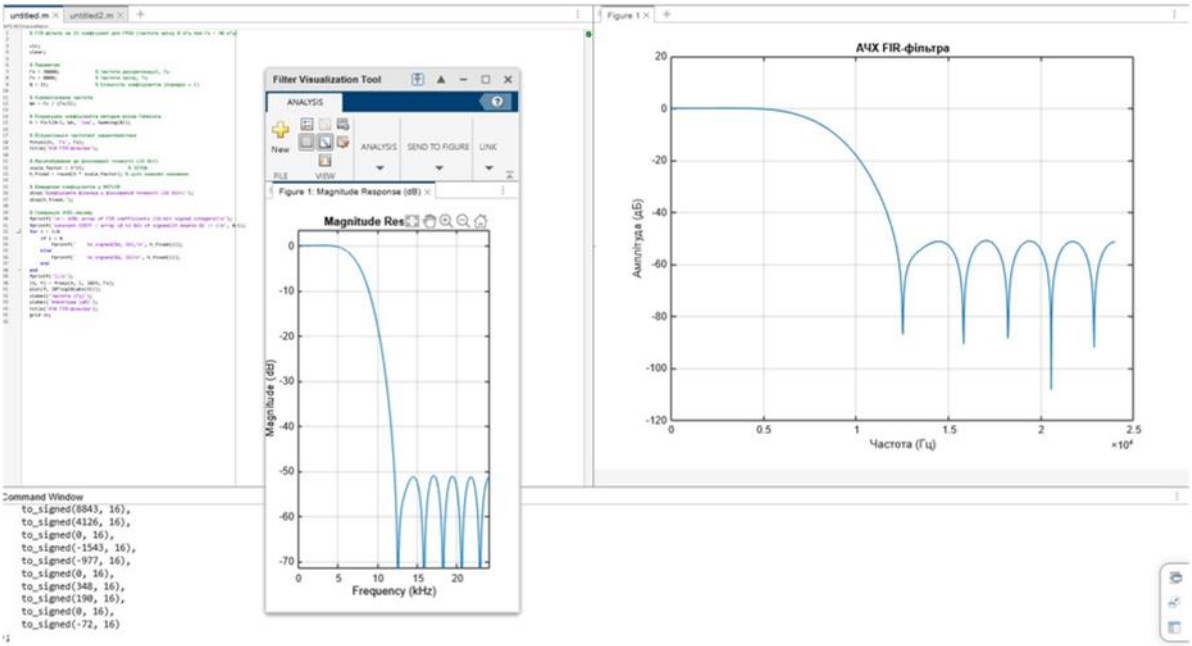
```
type coeff_array is array(0 to 20) of signed(15 downto 0);
constant FIR_COEFFS : coeff_array := (
x"FEE3", x"0000", x"02DA", x"0000", x"FD26",
x"0000", x"0676", x"0000", x"F32E", x"0000",
x"1656", -- center coefficient
-- symmetric part follows...
```

ПЕРЕВАГИ РЕАЛІЗАЦІЇ

- Використання стандартної бібліотеки numeric_std
- Відсутність операцій з плаваючою комою
- Ефективне використання DSP блоків FPGA
- Симетричність коефіцієнтів → економія пам'яті
- Константи в ROM або як генерики
- Простота масштабування та параметризації

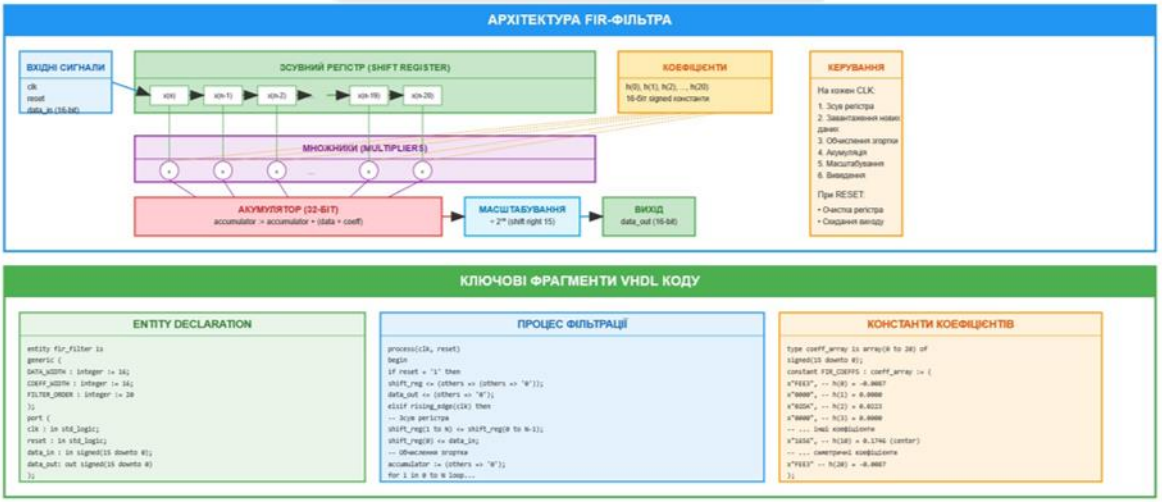
7

Моделювання в Matlab



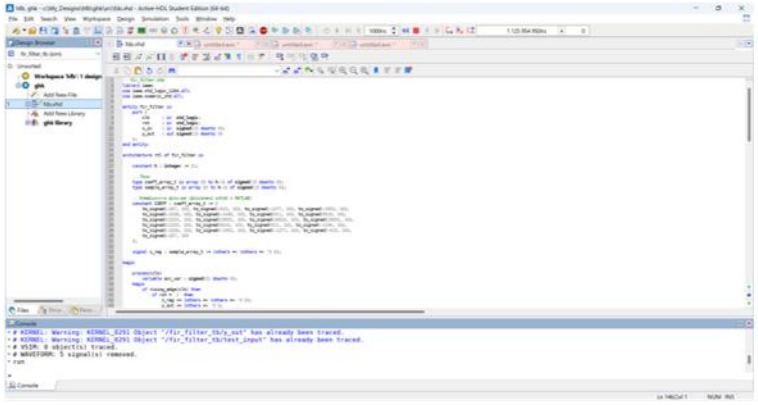
8

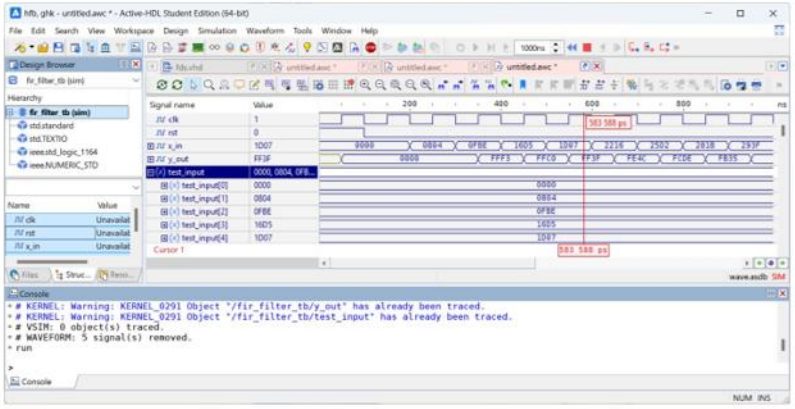
Архітектура розробленого цифрового фільтра



9

Реалізація фільтру 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 програмованій 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. 11

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
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;

    -- Òèïè
    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);

    -- Êîåô³ö³ºíòè ô³ëüòðà (ô³êñîâàí³ int16 ç 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'));
```

```
                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);

    -- Òåñòîâèé ñèãíàë: ñèíóñîïàä ç ÷àñòîòîþ ~1/20 ïåð³îä³â
    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

    -- ²íñòàíö³þâàííÿ ô³ëüòðà
    DUT: entity work.fir_filter
        port map (
            clk   => clk,
            rst   => rst,
            x_in  => x_in,
            y_out => y_out
        );

    -- Ãåíåðàö³ÿ òàêòîâîãî ñèãíàëó
    clk <= not clk after 50 ns;

    -- Òåñòîâèé ïðîöåñ
    stimulus: process
    begin
        wait for 100 ns;  -- òðèìàòè 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;
```