Proceeding, Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2008)
Auditorium Universitas Gunadarma, Depok, 20-21 Agustus 2008 ISSN : 1411-6286

SIMULASI RANCANGAN FILTER BUTTERWORTH MENGGUNAKAN XILINX-ISE 8.1i DAN MODELSIM 6.1b

[1]Wahyu Kusuma Raharja , [2]Sunny Arief Sudiro

Jurusan Teknologi Informasi, Fakultas Teknologi Industri, Universitas Gunadarma
Jl. Margonda Raya 100, Depok, 16424
Email : [1]wahyukr@staff.gunadarma.ac.id
[2]sunny@staff.gunadarma.ac.id

ABSTRAK

Sebelum rancangan rangkaian elektonika dirakit, terlebih dahulu dilakukan simulasi rangkaian tersebut menggunakan perangkat lunak komputer. Hal ini bertujuan untuk memperkecil tingkat kegagalan pada saat perakitan rangkaian tersebut. Salah satu perangkat lunak yang digunakan adalah Xilinx-Ise 8.1i sebagai editor pemrograman VHDL dan perangkat lunak ModelSim 6.1 b digunakan untuk simulasi hasil program VHDL. Penelitian ini melakukan simulasi rangkaian filter Butterworth orde 2, jenis band pass filter dengan frekuensi pancung bawah sebesar 50 Hz, frekuensi pancung atas 3000 Hz, dan frekuensi pencuplikan 44100 Hz. Berdasarkan perhitungan dalam transformasi Z diperoleh koefisien numerator 0.034, 0, -0.067, 0, 0.034 dan koefisien denominator 1, 3.41, -4.37, 2.52, -0.55. Penerapan rangkaian filter memerlukan 8 komponen D Flip-Flop, 6 komponen multiplier, 6 komponen adder, dan 2 komponen divider.Komponen-komponen penyusun rangkaian filter diprogram menggunakan Xilinx-Ise 8.1i. Sinyal masukan dan keluaran rangkaian filter disimulasikan menggunakan perangkat lunak ModelSim 6.1b. Data hasil simulasi dilakukan perbandingan dengan hasil keluaran program Matlab. Berdasarkan hasil uji perbandingan diperoleh tingkat kesamaan pada pengujian bagian numerator dan denominator, dengan menggunakan koefisien data kecil dan integer. Sedangkan pada pengujian rangkaian lengkap, terjadi kesalahan karena penggunaan komponen Divider yang melakukan proses pembagian dengan hasil pembulatan.

Kata kunci : *simulasi, filter, Xilinx-Ise 8.1i, ModelSim 6.1b*

1. PENDAHULUAN

Rangkaian elektronika dibangun melalui beberapa tahap antara lain perancangan, perakitan, dan pengujian. Perancangan dilakukan dengan mengetahui kebutuhan dan kegunaan dari rangkaian. Simulasi pada saat perancangan diperlukan untuk dapat memperkecil tingkat kesalahan pada saat rangkaian dilakukan perakitan. Sehingga dapat menekan biaya kegagalan dari rangkaian elektronika yang dirancang dan dibangun. Dewasa ini perangkat lunak sebagai media simulasi dari rancangan rangkaian elektronika mengalami perkembangan yang beragam. Hal ini didukung pula dengan kemudahan implementasi perangkat lunak yang dapat diprogram, seperti modul *Field Programmable Gate Array* (FPGA). Pada penelitian ini sebagai dasar dalam mengimplementasikan algoritma yang telah dibangun ke modul FPGA. Algoritma yang dibangun dengan perangkat lunak MATLAB selanjutnya dibangun pula menggunakan perangkat lunak XILINX-ISE 8.1i.

Penelitian ini membahas simulasi rancangan filter tipe Butterworth orde 2 menggunakan program Xilinx dan Modelsim. Penelitian ini bertujuan untuk mengetahui tingkat keberhasilan dari rangkaian filter yang dirancang, sebelum diimplementasikan dalam rangkaian perangkat keras menggunakan FPGA.

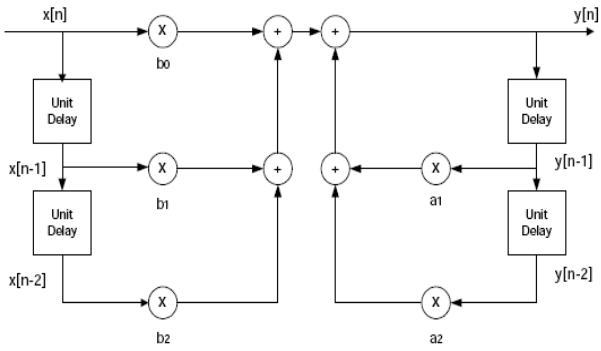
2. TINJAUAN PUSTAKA

Secara umum fungsi transfer dari filter digital IIR *(Infinite Impulse Response)* orde *L* [Smith, 1985] adalah sebagai berikut:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + ... + b_L z^{-L}}{1 + a_1 z^{-1} + ... + a_L z^{-L}} \quad (1)$$

Dalam kawasan waktu hubungan masukan-keluaran filter digital IIR dapat dirumuskan sebagai berikut:

$$y_k = \sum_{n=0}^{L} b_n x_{k-n} - \sum_{n=1}^{L} a_n y_{k-n} \quad (2)$$



Gambar 1. Diagram skematik filter IIR. [Rorabaugh and Britton, 1993]

Berdasarkan persamaan (2) diagram IIR di atas, dapat dijabarkan dengan persamaan :

$$y(n) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) + a_1 y(n-1) + a_2 y(n-2) \quad .....(3)$$

Berdasarkan teknik desainnya filter IIR dapat dikategorikan dalam beberapa teknik desain. Sedangkan filter FIR (*Finite Impulse Response*) pada dasarnya mempunyai 2 metode desain yaitu dengan jendela (*window*) dan pitajamak dengan pita transisi (*multiband with transition bands*).

Teknik desain filter digital IIR didasarkan pada transformasi bilinear dari prototipe fungsi transfer analog. Fungsi transfer analog biasanya salah satu dari tipe fungsi transfer: Butterworth, Chebyshev Tipe 1, Chebyshev Tipe 2 dan Elliptic (atau Cauer). Perbedaan antara tipe-tipe filter tersebut dapat dijelaskan dengan melihat filter lolos bawah (*lowpass*) analog seperti dibawah ini [Antoniou, 1993].

a. Fungsi transfer lolos bawah Butterworth mempunyai tanggap magnitude mendatar maksimum dan tanggap magnitude berkurang secara mendatar dengan frekuensi bertambah.
b. Fungsi transfer lolos bawah Chebyshev Tipe 1 mempunyai tanggap magnitude dengan riak setimbang pada pita lolos dan tanggap magnitude berkurang secara mendatar dengan frekuensi bertambah di sebelah luar pita lolos.
c. Fungsi transfer lolos bawah Chebyshev Tipe 2 mempunyai tanggap magnitude berkurang secara mendatar pada pita lolos dengan frekuensi bertambah dan tanggap magnitude riak setimbang pada pita cegah.
d. Fungsi transfer lolos bawah Elliptic mempunyai tanggap magnitude riak setimbang pada kedua pita, pita lolos maupun pita cegah.

3. METODE PENELITIAN

Langkah simulasi rancangan rangkaian filter sebagai berikut :

1. Menyusun program menggunakan Matlab.

Program Matlab yang disusun, didasarkan pada penetapan rancangan rangkaian filter menggunakan butterworth orde 2 berjenis band pass filter. Filter ditetapkan dapat melewatkan rentang frekuensi pada frekuensi *cutoff* bawah sebesar 50 Hz dan frekuensi *cutoff* atas sebesar 3000 Hz. Frekuensi sampling yang digunakan pada penelitian ini sebesar 44100 Hz. Berikut ini program Matlab dari fungsi pemfilteran :

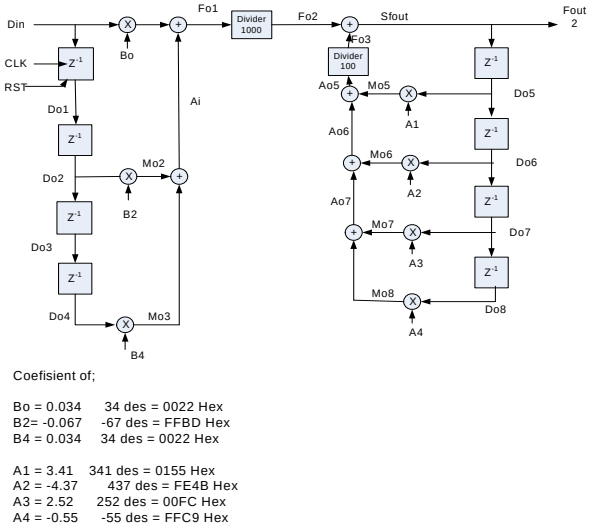
```
%Pemfilteran : Band Pass Filter
w1 = 50*2/fs;  w2 = 3000*2/fs;
[b,a]=butter(2,[f1 f2]);
s = [filter(b,a,y)];
```

Hasil dari program tersebut diperoleh nilai koefisien filter yaitu *Numerator* (b) and *Denominator* (a) yang besarnya sebagai berikut :

b = 0.034 0 -0.067
0 0.034
a = 1 -3.41 4.31
-2.52 0.55

2. Membuat skematik rangkaian filter butterworth.

Berdasarkan hasil keluaran koefisien pada program Matlab, selanjutnya dirancang skematik rangkaian filter seperti diperlihatkan gambar 2.



Gambar 2. Skematik rangkaian *band pass filter* Butterworth orde 2.

3. Membuat program menggunakan Xilinx.

Berdasarkan skematik yang telah dibangun, selanjutnya membuat program menggunakan Xilinx-Ise 8.1i. Komponen-komponen yang diperlukan untuk menyusun rangkaian seperti terlihat gambar 2 terdiri atas :

a. Komponen D Filp Flop (DFF) sebanyak 8 unit.

Setiap komponen DFF ditulis dengan program :

```
-- Delay 16 BIT Unit Use D FlipFlop(2)--DFF2
---------------------------------------

library ieee;
use ieee.std_logic_1164.all;

ENTITY dff2 IS
PORT( D                              : IN
STD_LOGIC_VECTOR(15 downto 0);
                Clk, Res : IN STD_LOGIC;
                Q        : OUT
STD_LOGIC_VECTOR(15 downto 0));

END dff2;

ARCHITECTURE behavioral OF dff2 IS
BEGIN
PROCESS(Clk, Res) --We only care about Clk
BEGIN
IF Res = '1' THEn Q <= X"0000"; -- determine Q=
0 Hexa
        Else
        IF (Clk'event) AND (Clk='1') THEN --
Positive Edge
                                Q <= D;
        END IF;
END IF;
END PROCESS;
END behavioral;
```

b. Komponen Multiplier sebanyak 7 unit.

Program Xilinx untuk membuat komponen Multiplier adalah :

```
-- MULTIPLIER 32 BIT
--------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_signed.ALL;
USE ieee.std_logic_unsigned.ALL;

ENTITY signed_mult IS
        PORT (
                a:              IN
STD_LOGIC_VECTOR (15 DOWNTO 0);
                b:              IN
STD_LOGIC_VECTOR (15 DOWNTO 0);
                result:         OUT
STD_LOGIC_VECTOR (31 DOWNTO 0)
        );
END signed_mult;

ARCHITECTURE rtl OF signed_mult IS
        SIGNAL a_int, b_int:
        SIGNED (15 downto 0);
        SIGNAL pdt_int:
        SIGNED (31 downto 0);

BEGIN
        a_int <= SIGNED (a);
        b_int <= SIGNED (b);
        pdt_int <= a_int * b_int;
        result <=
STD_LOGIC_VECTOR(pdt_int);
END rtl;
```

c. Komponen Adder sebanyak 6 unit.

Program Xilinx untuk membuat komponen Adder adalah :

```
-- Adder 32 bit
-----------------------------------------
LIBRARY IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity add32i is
        port( a, b                      : in
        STD_LOGIC_VECTOR(31 downto 0);
            sum             : out
        STD_LOGIC_VECTOR(31 downto 0));
           -- cout                     : out
        STD_LOGIC                   );
end add32i;

architecture STRUCTURE of add32i is
component add16
        port( a, b                      : in
        STD_LOGIC_VECTOR(15 downto 0);
                        cin                 : in
STD_LOGIC;
            sum             : out
        STD_LOGIC_VECTOR(15 downto 0);
                        cout                : out
STD_LOGIC );
end component;

signal cout1, cout2, scin : STD_LOGIC;
begin

scin <= '0';
add16i1: add16 port map (a(15 downto 0), b(15
downto 0), scin, sum(15 downto 0), cout1);
add16i2: add16 port map (a(31 downto 16), b(31
downto 16), cout1, sum(31 downto 16), cout2);

end structure;
```

d. Komponen Divider sebanyak 2 unit.

Program Xilinx untuk membuat komponen Divider adalah :

```
--Component Divider1000
--================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity divider1000 is
   Port ( ain : in  STD_LOGIC_VECTOR (31
downto 0);
          dvdout : out  STD_LOGIC_VECTOR (31
downto 0));
end divider1000;

architecture Behavioral of divider1000 is
--  signal a ,b : bit_vector (31 downto 0);
begin
--   a <= to_bitvector(ain);
--          b <= a sra 10;
           dvdout <=
to_stdlogicvector(to_bitvector(ain) sra 10);

end Behavioral;
```

4. Melakukan simulasi menggunakan Modelsim dari program Xilinx.

Simulasi digunakan untuk mengetahui kebenaran dari hasil program rangkaian filter yang telah disusun dengan program Xilinx. Perangkat lunak yang digunakan untuk simulasi tersebut adalah Modelsim. Pada program Modelsim dilakukan pemberian sinyal masukan sebagai berikut :

```
restart -f
force RST 0 10
force Din X"0A" 0, X"05" 100, X"A0" 200,
X"11" 300, X"2B" 400, X"45" 500, X"32" 600,
                X"62" 700, X"B2" 800, X"AA"
900, X"02" 1000
force CLOCK 1 50,0 100 -repeat 100
run 1000
```

5. Membandingkan hasil Matlab dengan Modelsim.

Langkah selanjutnya yaitu membandingkan hasil simulasi dari keluaran program Modelsim dengan keluaran program Matlab. Kedua program diberikan sinyal input yang sama.

4. HASIL DAN PEMBAHASAN

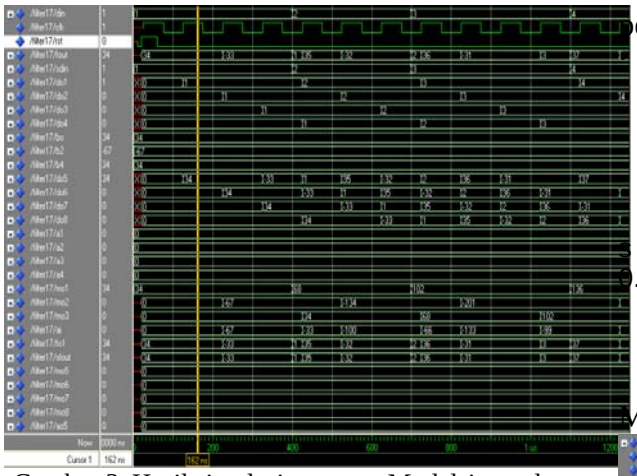
Pengujian dilakukan pada masing-masing sisi numerator (koefisien b), denominator (koefisien a), dan gabungan keduanya (rangkaian lengkap).

Pengujian 1 disimulasi pada bagian numerator menggunakan Bo =34, B2 = -67, and B4 = 34.

Hasil pengujian dalam program Matlab diperoleh :

```
>> b=[34 0 -67 0 34];
>> a=[1 0 0 0 0];
>> y=[1 1 1 2 2 2 3 3 3 4];
>> s = [filter(b,a,y)];
>> s
s =   34   34  -33    1   35  -32    2   36  -
31    3
```

Hasil pengujian dengan program Modelsim diperoleh :

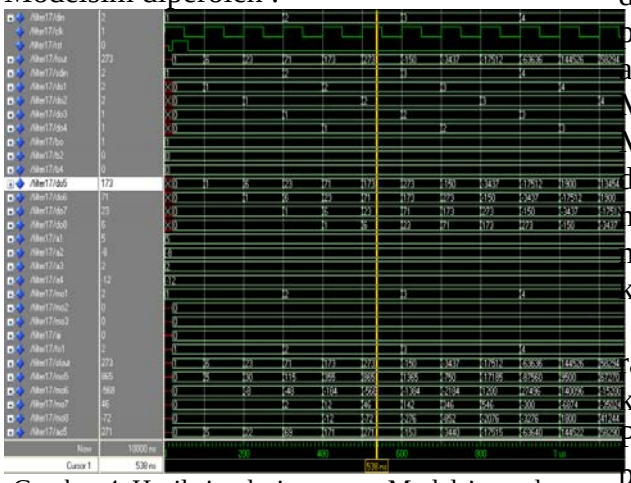


Gambar 3. Hasil simulasi program Modelsim pada bagian numerator

Pengujian 2 disimulasi pada bagian denominator menggunakan A1 = -5, A2 = 8, A3 = -2, and A4 = 12. Hasil pengujian dalam program Matlab diperoleh :

```
>> b=[1 0 0 0 0];
>> a=[1 -5 8 -2 12];
>> y=[1 1 1 2 2 2 3 3 3 4];
>> s = [filter(b,a,y)];
>> s
s =     1     6    23    71   173
273    -150    -3437
       -17512    -63636
```

Hasil pengujian dengan program Modelsim diperoleh :

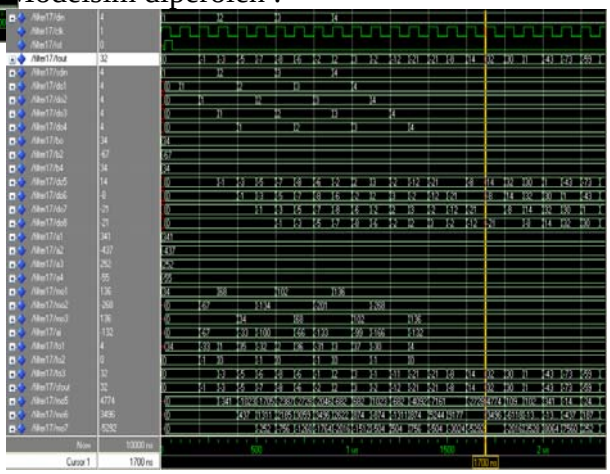


Gambar 4. Hasil simulasi program Modelsim pada bagian denominator

Pengujian 3 disimulasi pada rangkaian filter dengan memberikan nilai koefisien pada numertator dan denominator. Nilai koefisien diperoleh dari hasil perhitungan program Matlab. Hasil pengujian dalam program Matlab diperoleh

```
>> b=[0.034 0 -0.067 0 0.034];
>> a=[1 -3.41 4.37 -2.52 0.55];
>> y=[1 1 1 2 2 2 3 3 3 4];
>> s = [filter(b,a,y)];
>> s
s =   0.0340   0.1499   0.3297   0.5558
0.8485   1.1810   1.5405   1.9608
         2.4326  2.9621
```

Hasil pengujian dengan program Modelsim diperoleh :



Gambar 5. Hasil simulasi program Modelsim dengan rangkaian lengkap.

Berdasarkan tiga pengujian di atas dapat dijelaskan bahwa pengujian 1 dan pengujian 2 diperoleh hasil yang sama antara simulasi menggunakan program Matlab dengan simulasi program Modelsim. Kedua pengujian tersebut dilakukan pada masing-masing koefisien numerator dan denominator, yang menunjukkan hasil yang baik dengan kebenaran 100%.

Pengujian 3 dilakukan pada rangkaian lengkap filter yang melibatkan koefisien numerator dan denominator. Pada pengujian terjadi kesalahan atau perbedaan hasil antara program Matlab dengan Modelsim. Kesalahan yang terjadi diakibatkan adanya keterbatasan tipe data pada Modelsim, dengan pembulatan pada bilangan pecahan. Komponen divider yang dipakai menghasilkan seperti diperlihatkan tabel 1.

Tabel 1.
Data masukan (Fo1) dan data keluaran (Fo2) pada komponen Divider1000

Fo1	34	-33	1	35	-32	2	36	-31	3	37	-30	4
Fo2	0	-1	0	0	-1	0	0	-1	0	0	-1	0

Terlihat dari tabel 1 bahwa data-data masukan (Fo1) dibagi dengan konstanta 1024 dengan komponen Divider1000 diperoleh data keluaran (Fo2). Dari data tersebut, fungsi Divider1000 memberikan nilai hasil 0 jika data masukan positif dan nilai -1 jika data masukan negatif. Hal ini menyebabkan hasil proses terjadi

5. KESIMPULAN DAN SARAN

Rancangan rangkaian filter butterworth orde 2 telah berhasil dengan baik untuk masing-masing bagian numerator dan denominator. Pada pengujian rangkaian lengkap masih terdapat perbedaan antara hasil simulasi program Matlab dengan program Modelsim. Hal ini dipengaruhi oleh penggunaan komponen Divider yang melakukan proses pembagian dengan hasil pembulatan. Salah satu permasalahan pada proses simulasi dengan program Modelsim adalah penggunaan data integer, dimana data pecahan harus dilakukan pembulatan terlebih dahulu. Perlu dikembangkan penggunaan bilangan pecahan pada simulasi program Modelsim.

DAFTAR PUSTAKA

[1] Antoniou, A., 1993, *Digital Filters: Analisys, Design, and Applications*, McGraw-Hill, New York

[2] Rabiner, L. R. and Gold, 1975, *Theory and Application of Digital Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, Inc., New Jersey

[3] Rorabaugh and Britton, C., 1993, *Digital Filter Designer's Handbook*, Tab Books/McGraw-Hill, New York

[4] Smith, J.O., 1995, *Introduction to Digital Filter Theory*, in *Digital Audio Signal Processing: An Anthology* (J. Strawn, ed.), William Kaufmann, Inc., California

[5] Smith , J.O., Mar. 2007, *Spectral Audio Signal Processing*, online book.
http://ccrma.stanford.edu/~jos/sasp/