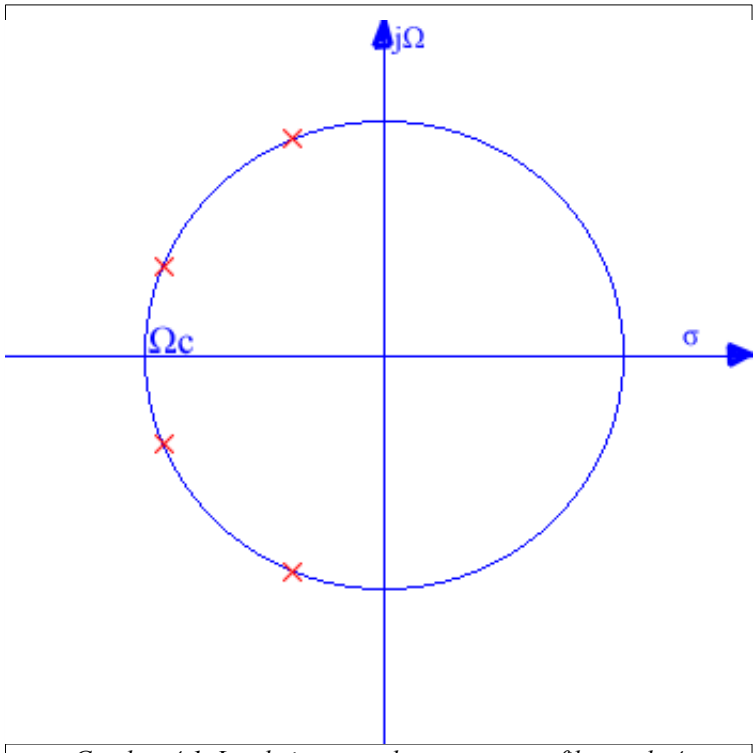
BAB IV
PERANCANGAN SISTEM

4.1 Desain Filter IIR

Desain filter IIR berguna untuk mencari koefisien frekuensi filter IIR. Koefisien tersebut kemudian digunakan dalam perhitungan proses filter. Frekuensi cut-off yang digunakan adalah 10.000Hz hingga 20.000Hz dengan selisih 1.000Hz.

4.1.1 Penentuan Tiang

Letak tiang dalam perancangan filter IIR berguna untuk menentukan fungsi alih sistem. Filter IIR orde 4 memilik 8 buah tiang yang tersebar pada lingkaran radius frekuensi *cutoff*. Filter stabil dan kausal akan didapatkan dengan memilih tiang di sebelah kiri sumbu imajiner. Gambar 4.1 menunjukan letak tiang untuk perancangan sistem ini.



Gambar 4.1. Letak tiang untuk perancangan filter orde 4

Dari gambar diatas, dapat disimpulkan bahwa sistem memiliki tiang dengan nilai:

$$p_1 = \Omega_c(\cos 112{,}5^o + j\sin 112.5^o)$$

$$p_2 = \Omega_c(\cos 157{,}5^o + j\sin 157{,}5^o)$$

$$p_3 = \Omega_c(\cos 202{,}5^o + j\sin 202{,}5^o)$$

$p_4 = \Omega_c(\cos 247{,}5^o + j\sin 247{,}5^o)$

Jika $\theta_1 = 112{,}5^o$ dan $\theta_2 = 157{,}5^o$, maka persamaan dapat disederhanakan menjadi.

$$p_1 = \Omega_c(\cos\theta_1 + j\sin\theta_1)$$
$$p_2 = \Omega_c(\cos\theta_2 + j\sin\theta_2)$$
$$p_3 = \Omega_c(\cos\theta_2 - j\sin\theta_2)$$
$$p_4 = \Omega_c(\cos\theta_1 - j\sin\theta_1)$$

4.1.2 Fungsi Alih

Menurut persamaan , diperoleh perhitungan polinomial penyebut sebagai berikut

untuk θ_1

$$(s-\Omega_c(\cos\theta_1 + j\sin\theta_1))(s-\Omega_c(\cos\theta_1 - j\sin\theta_1)) = s^2 - 2s\,\Omega_c\cos\theta_1 + \Omega_c^2(\cos^2\theta_1 + \sin^2\theta_1)$$
$$= s^2 - 2s\,\Omega_c\cos\theta_1 + \Omega_c^2$$

(4.1)

untuk θ_2

$$(s-\Omega_c(\cos\theta_2 + j\sin\theta_2))(s-\Omega_c(\cos\theta_2 - j\sin\theta_2)) = s^2 - 2s\,\Omega_c\cos\theta_2 + \Omega_c^2(\cos^2\theta_2 + \sin^2\theta_2)$$
$$= s^2 - 2s\,\Omega_c\cos\theta_2 + \Omega_c^2$$

(4.2)

sehingga didapatkan fungsi alih sebagai berikut

$$H_a(s) = \frac{\Omega_c^4}{(s^2 - 2s\,\Omega_c\cos\theta_1 + \Omega_c^2)(s^2 - 2s\,\Omega_c\cos\theta_2 + \Omega_c^2)} \qquad (4.3)$$

4.1.3 Transformasi Bilinear

Transformasi bilinear dilakukan dengan cara memasukan persamaan 2.4 ke dalam persamaan 4.3.

$$H(z) = H_a\left(\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}\right)$$

Apabila f_s frekuensi sampling maka

$$f_s = \frac{1}{T}$$

$$H(z)=\frac{\Omega_c^4}{((2f_s\frac{1-z^{-1}}{1+z^{-1}})^2-2(2f_s\frac{1-z^{-1}}{1+z^{-1}})\Omega_c\cos\theta_1+\Omega_c^2)((2f_s\frac{1-z^{-1}}{1+z^{-1}})^2-2(2f_s\frac{1-z^{-1}}{1+z^{-1}})\Omega_c\cos\theta_2+\Omega_c^2)}$$

$$H(z)=\frac{\frac{\Omega_c^4(1+4z^{-1}+6z^{-2}+4z^{-3}+z^{-4})}{(4f_s^2-4f_s\cos\theta_1\Omega_c+\Omega_c^2)(4f_s^2-4f_s\cos\theta_2\Omega_c+\Omega_c^2)}}{\begin{array}{l}(4f_s^2-4f_s\cos\theta_1\Omega_c+\Omega_c^2)(4f_s^2-4f_s\cos\theta_2\Omega_c+\Omega_c^2)+((4f_s^2-4f_s\cos\theta_1\Omega_c+\Omega_c^2)(-8f_s^2+2\Omega_c^2)+(4f_s^2-4f_s\cos\theta_2\Omega_c+\Omega_c^2)(-8f_s^2+2\Omega_c^2))z^{-1}+\\((4f_s^2+4f_s\cos\theta_1\Omega_c+\Omega_c^2)(4f_s^2-4f_s\cos\theta_2\Omega_c+\Omega_c^2)+(4f_s^2+4f_s\cos\theta_2\Omega_c+\Omega_c^2)(4f_s^2-4f_s\cos\theta_1\Omega_c+\Omega_c^2)+(-8f_s^2+2\Omega_c^2)(-8f_s^2+2\Omega_c^2))z^{-2}+\\((4f_s^2+4f_s\cos\theta_1\Omega_c+\Omega_c^2)(-8f_s^2+2\Omega_c^2)+(4f_s^2+4f_s\cos\theta_2\Omega_c+\Omega_c^2)(-8f_s^2+2\Omega_c^2))z^{-3}+(4f_s^2+4f_s\cos\theta_1\Omega_c+\Omega_c^2)(4f_s^2+4f_s\cos\theta_2\Omega_c+\Omega_c^2)z^{-4}\\ \hline (4f_s^2-4f_s\cos\theta_1\Omega_c+\Omega_c^2)(4f_s^2-4f_s\cos\theta_2\Omega_c+\Omega_c^2)\end{array}}$$

(4.4)

4.1.4 Koefisien

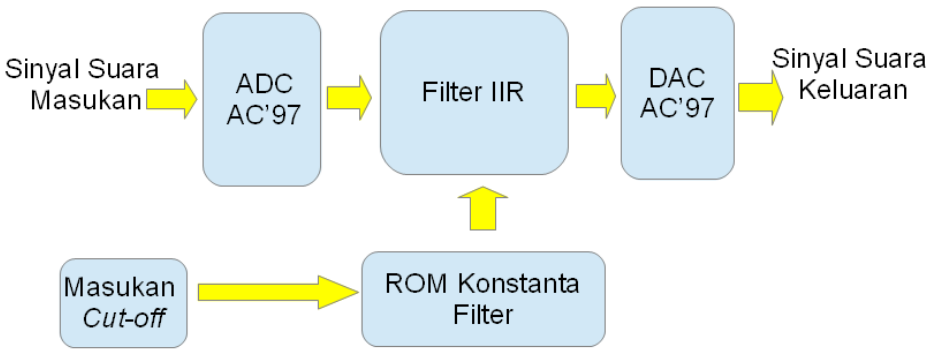
Koefisien filter a1 hingga a4 didapatkan dari penyebut persamaan 4.4, sedangkan b0 hingga b4 merupakan pembilang. Koefisien yang didapat dari perhitungan diperlihatkan pada tabel 4.1.

Tabel 4.1. Koefisien filter

fc	a1	a2	a3	a4	b0	b1	b2	b3	b4
10000.	-3.4580196	4.5161394	-2.6373437	0.5806595	0.0000897	0.0003589	0.0005383	0.0003589	0.0000897
11000.	-3.4043737	4.3841576	-2.5277784	0.5500401	0.0001278	0.0005114	0.0007671	0.0005114	0.0001278
12000.	-3.3508826	4.2551207	-2.4224811	0.5210627	0.0001762	0.0007049	0.0010574	0.0007049	0.0001762
13000.	-3.297559	4.1290113	-2.3213121	0.49364	0.0002363	0.0009450	0.0014176	0.0009450	0.0002363
14000.	-3.2444154	4.0058096	-2.2241343	0.4676893	0.0003093	0.0012373	0.0018559	0.0012373	0.0003093
15000.	-3.1914639	3.8854938	-2.1308134	0.4431319	0.0003968	0.0015871	0.0023807	0.0015871	0.0003968
16000.	-3.1387162	3.7680401	-2.0412174	0.4198935	0.0005000	0.0020000	0.0030000	0.0020000	0.0005000
17000.	-3.0861838	3.6534229	-1.9552176	0.3979033	0.0006203	0.0024812	0.0037218	0.0024812	0.0006203
18000.	-3.0338778	3.5416148	-1.8726878	0.3770944	0.0007590	0.0030359	0.0045539	0.0030359	0.0007590
19000.	-2.9818087	3.4325868	-1.7935047	0.3574033	0.0009173	0.0036692	0.0055038	0.0036692	0.0009173
20000.	-2.9299869	3.3263084	-1.7175482	0.3387698	0.0010964	0.0043858	0.0065787	0.0043858	0.0010964

4.2 Blok Diagram

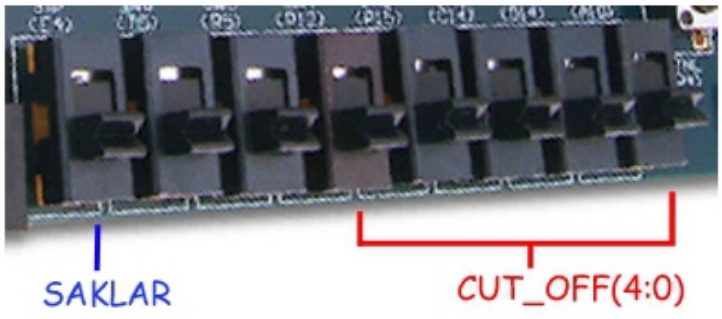
Sistem terdiri atas beberapa blok sub-sistem, yaitu AC'97 hardware driver, blok pemproses filter, blok konstanta filter, blok latch, dan blok switch. Blok pemproses filter, blok konstanta filter untuk menyimpan konstanta filter, blok *latch* untuk melewatkan filter tanpa melakukan proses, dan blok multiplexer untuk memilih antara proses filter dan *latch*.



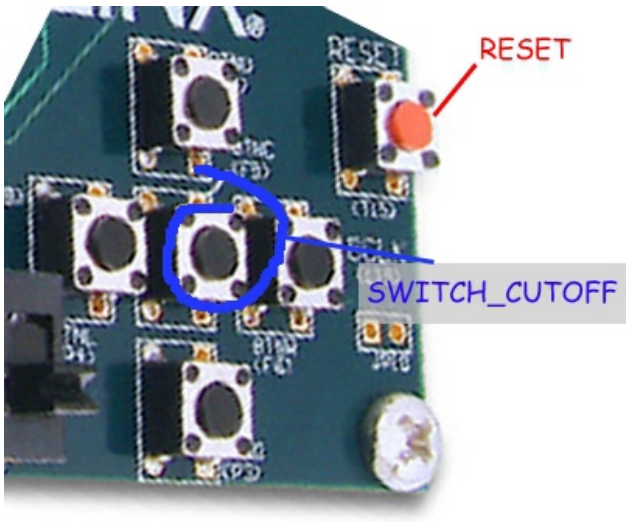
Gambar 4.2. Blok diagram sistem

4.3 Pengaturan Input Output Dengan Pin FPGA

Masukan untuk mengendalikan proses sistem berupa switch dan tombol pada board Atlyst. Switch yang digunakan adalah sw0, sw1, sw2, sw3, sw4, sw7. Sw0 hingga sw4 berfungsi untuk memilih frekuensi cutoff sedangkan sw7 untuk memilih apakah sinyal masukan difilter atau hanya dilewatkan. Tombol btnc berfungsi untuk men-set frekuensi cutoff. Tombol RESET berguna untuk menghapus isi ROM penyimpanan kode fpga dan me-*reset* register AC'97. Gambar 4.3 dan 4.4 menunjukan letak dan kegunaan input.



Gambar 4.3. Switch Board Atlys



Gambar 4.4. Button Board Atlys

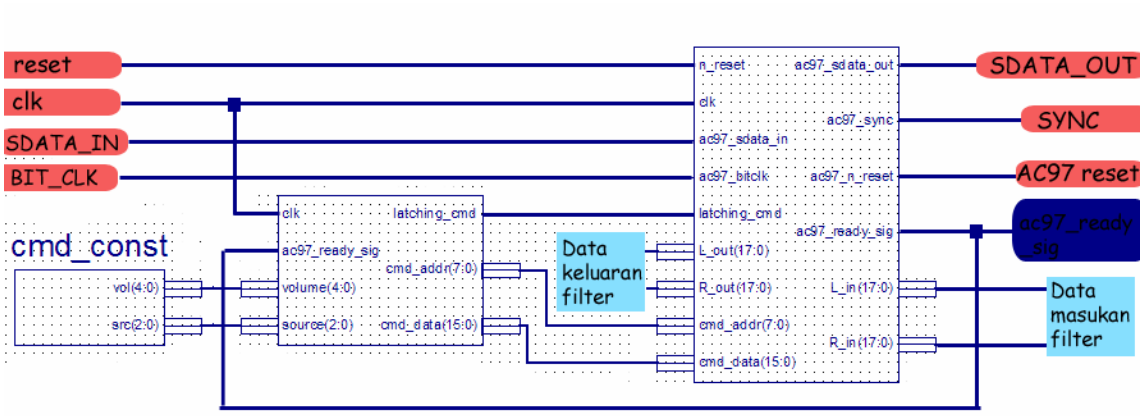
Agar FPGA mampu berhubungan dengan I/O sistem maka diperlukan file ucf

untuk pengaturan. Bentuk teks untuk relasi antara I/O sistem dengan konfigurasi pin FPGA ditunjukkan sebagai berikut:

```
NET "AC97_N_RESET" LOC = "T17";
NET "BIT_CLK" LOC = "L13";
NET "SDATA_IN" LOC = "T18";
NET "AC97_SDATA_OUT" LOC = "N16";
NET "SAKLAR" LOC = "E4";
NET "AC97_SYNC" LOC = "U17";
NET "CUT_OFF<0>" LOC = "A10";
NET "CUT_OFF<1>" LOC = "D14";
NET "CUT_OFF<2>" LOC = "C14";
NET "CUT_OFF<3>" LOC = "P15";
NET "CUT_OFF<4>" LOC = "P12";
NET "clk" LOC = "L15";
NET "n_reset" LOC = "T15";
NET "SWITCH_CUTOFF" LOC = "F5";  #BTNC
```

4.4 Codec AC'97 Hardware Driver

AC'97 hardware driver dihubungkan dengan blok komponen lainnya. Gambar 4.5 menunjukan konfigurasi hardware driver dengan komponen lainnya.



Gambar 4.5. Pengaturan AC-97 Hardware Driver

4.5 Konstanta

4.5.1 Konversi Koefisien

Blok Konstanta berfungsi untuk menyimpan nilai koefisien pengali filter. Blok ini berisi LUT(Look-Up Table). Koefisien pengali berbentuk pecahan desimal. Agar dapat disimpan dalam FPGA, koefisien pengali harus diubah menjadi format *fixed point two's complement* 32 bit. Cara konversi tersebut sebagai berikut.

1. Dikalikan dengan 2^{28} untuk menggeser koefisien sebesar 28 bit ke kanan.

2. Pembulatan untuk menghilangkan pecahan.
3. Konversi biner 32 bit.
4. Konversi bentuk two's complement.

4.5.2 Port

Blok konstanta memiliki beberapa port untuk antarmuka dengan blok lainnya. Port tersebut ditunjukan sebagai berikut.

```
cut_off           : in  STD_LOGIC_VECTOR (4 downto 0);
switch_cutoff     : in std_logic;
clk               : in std_logic;
x0                : out  STD_LOGIC_VECTOR (31 downto 0);
x1                : out  STD_LOGIC_VECTOR (31 downto 0);
x2                : out  STD_LOGIC_VECTOR (31 downto 0);
x3                : out  STD_LOGIC_VECTOR (31 downto 0);
x4                : out  STD_LOGIC_VECTOR (31 downto 0);
y1                : out  STD_LOGIC_VECTOR (31 downto 0);
y2                : out  STD_LOGIC_VECTOR (31 downto 0);
y3                : out  STD_LOGIC_VECTOR (31 downto 0);
y4                : out  STD_LOGIC_VECTOR (31 downto 0)
```

4.5.3 Cara Kerja Blok Konstanta

Saklar CUT_OFF akan menetukan nilai konstanta cutoff. Keluaran Konstanta akan berubah saat saklar CUT_OFF dan tombol SWITCH_CUTOFF ditekan.

```
process (clk,switch_cutoff)
begin
      if clk'event and clk='1' then
            if switch_cutoff='1' then
            x0<=sig_x0;
            x1<=sig_x1;
            x2<=sig_x2;
            x3<=sig_x3;
            x4<=sig_x4;

            y1<=sig_y1;
            y2<=sig_y2;
            y3<=sig_y3;
            y4<=sig_y4;
            end if;
      end if;
end process;
```

4.6 Proses Filter

Blok Proses Filter berfungsi untuk mem-filter sinyal masukan. Saat Hardware Driver siap mengirim data, Blok Proses Filter akan aktif memproses data. Port yang terdapat pada blok proses filter adalah sebagai berikut.

4.6.1 Port

Blok proses filter memiliki beberapa port untuk antarmuka dengan blok lainnya. Port tersebut ditunjukan sebagai berikut.

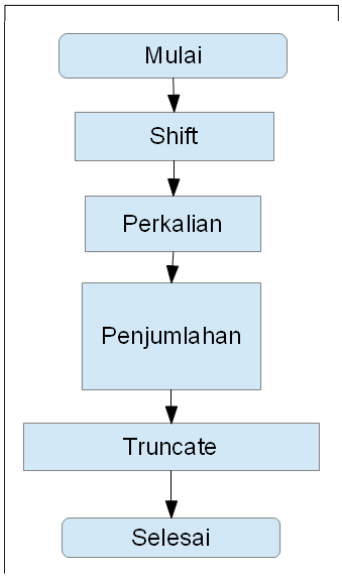
```
clk          : in std_logic;
n_reset      : in std_logic;
ready        : in std_logic;
bus_in       : in std_logic_vector(17 downto 0);
bus_out      : out std_logic_vector(17 downto 0);
filter_ready: out std_logic;

coef_b0      : in std_logic_vector(31 downto 0);
coef_b1      : in std_logic_vector(31 downto 0);
coef_b2      : in std_logic_vector(31 downto 0);
coef_b3      : in std_logic_vector(31 downto 0);
coef_b4      : in std_logic_vector(31 downto 0);

coef_a1      : in std_logic_vector(31 downto 0);
coef_a2      : in std_logic_vector(31 downto 0);
coef_a3      : in std_logic_vector(31 downto 0);
coef_a4      : in std_logic_vector(31 downto 0)
```

4.6.2 Cara Kerja

Keluaran filter dapat dijabarkan dalam persamaan 4.5 berikut.

$$y(n)=\sum_{l=0}^{4} b(l)x(n-l)-\sum_{l=1}^{4} a(l)y(n-l) \quad (4.5)$$

Proses Filter membagi perhitungan dalam beberapa tahap. Gambar 4.6 menunjukan flowchart tahap proses perhitungan.



Gambar 4.6. flowchart proses filter

4.7 Latch

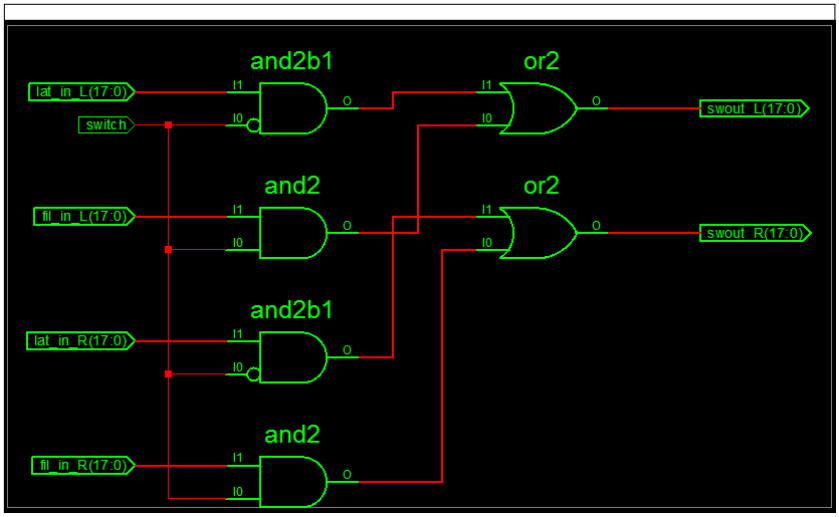
Blok Latch berfungsi melewatkan data tanpa melakukan proses. Blok sistem latch berguna agar sinyal asli dapat diperhatikan tanpa perlu memprogram ulang FPGA. Port Latch ditunjukan sebagai berikut.

```
clk          : in std_logic;
n_reset      : in std_logic;
ready        : in std_logic;
L_bus_out    : in std_logic_vector(17 downto 0);
R_bus_out    : in std_logic_vector(17 downto 0);
L_bus        : out std_logic_vector(17 downto 0);
R_bus        : out std_logic_vector(17 downto 0)
```

4.8 Switch

Blok Switch berfungsi sebagai multiplexer untuk memilih memproses sinyal masukan atau hanya meneruskan saja. Port Switch ditunjukan sebagai berikut.

```
lat_in_R : in  STD_LOGIC_VECTOR (17 downto 0);
lat_in_L : in  STD_LOGIC_VECTOR (17 downto 0);
fil_in_R : in  STD_LOGIC_VECTOR (17 downto 0);
fil_in_L : in  STD_LOGIC_VECTOR (17 downto 0);
swout_R : out  STD_LOGIC_VECTOR (17 downto 0);
swout_L : out  STD_LOGIC_VECTOR (17 downto 0);
switch      : in  STD_LOGIC
```



Gambar 4.7. Skematik blok Switch

4.9 Perancangan Keseluruhan

Perancangan keseluruhan digunakan untuk menghubungkan seluruh komponen sub-sistem agar menjadi filter utuh.