

IIR Filter IP

2017.05.04

Kevin Bloom

Overview



- System Design

Overview



- System Design
- IIR IP

Overview



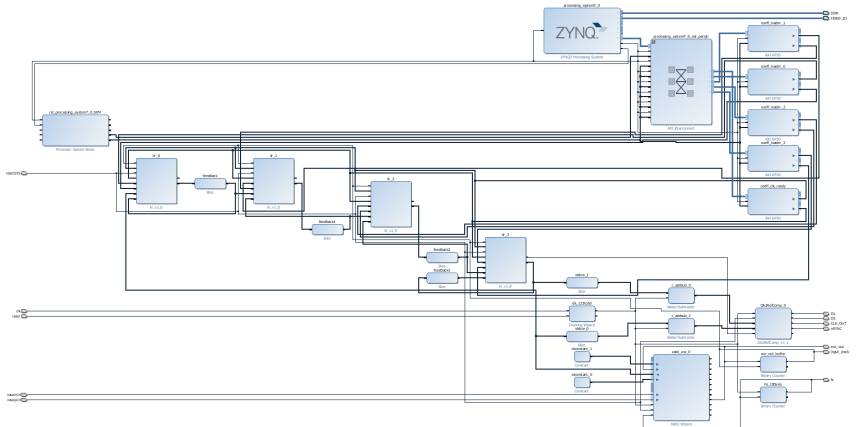
- System Design
- IIR IP
- Zynq Communication

Overview

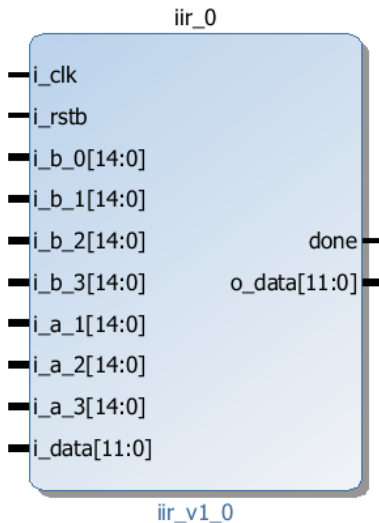


- System Design
- IIR IP
- Zynq Communication
- Example Outputs

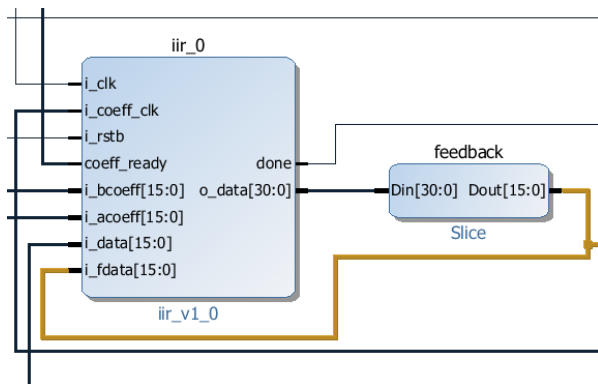
System Design



IIR IP — Hardware Only



IIR IP — Zynq Fed



IIR IP — Data Input Process

```
70--- Data input ---
71
72 p_data_input : process (i_rstb,i_clk)
73 begin
74     if(i_rstb='1') then
75         p_data  <= (others=>(others=>'0'));
76         p_fdata <= (others=>(others=>'0'));
77     elsif(rising_edge(i_clk)) then
78         p_data  <= signed(i_data)&p_data(0 to p_data'length-2);
79         p_fdata <= signed(i_fdata)&p_fdata(0 to p_fdata'length-2);
80     end if;
81 end process p_data_input;
```

IIR IP — Arithmetic

```
85 p_mult : process (i_rstb,i_clk,p_data,r_bcoeff,p_fdata,r_acoeff)
86 begin
87     if(i_rstb='1') then
88         r_mult      <= (others=>(others=>'0'));
89         r_fmuilt     <= (others=>(others=>'0'));
90     elsif(i_clk='1') then
91         for k in 0 to 2 loop
92             r_mult(k) <= p_data(k) * r_bcoeff(k);
93             r_fmuilt(k) <= p_fdata(k) * r_acoeff(k); --k=2, zero
94         end loop;
95     end if;
96 end process p_mult;
```

IIR IP — Arithmetic

```
98 p_add_st0 : process (i_rstb,i_clk,r_mult,r_fmult)
99 begin
100   if(i_rstb='1') then
101     r_add_st0    <= (others=>(others=>'0'));
102     r_fadd_st0   <= (others=>(others=>'0'));
103   elsif(i_clk='1') then
104     for k in 0 to 1 loop
105       r_add_st0(k) <= resize(r_mult(2*k),33) + resize(r_mult(2*k+1),33);
106       r_fadd_st0(k) <= resize(r_fmult(2*k),33) + resize(r_fmult(2*k+1),33);
107     end loop;
108   end if;
109 end process p_add_st0;

111 p_add_st1 : process (i_rstb,i_clk,r_add_st0,r_fadd_st0)
112 begin
113   if(i_rstb='1') then
114     r_add_st1 <= (others=>'0');
115     r_fadd_st1 <= (others=>'0');
116   elsif(i_clk='1') then
117     r_add_st1 <= resize(r_add_st0(0),34) + resize(r_add_st0(1),34);
118     r_fadd_st1 <= resize(r_fadd_st0(0),34) + resize(r_fadd_st0(1),34);
119   end if;
120 end process p_add_st1;
```

IIR IP — Arithmetic

```
122 p_final_sum : process (i_rstb,i_clk,r_add_st1,r_fadd_st1)
123 begin
124     if(i_rstb='1') then
125         r_final_sum    <= (others=>'0');
126     elsif(i_clk='1') then
127         r_final_sum    <= r_add_st1 - r_fadd_st1;
128     end if;
129 end process p_final_sum;
```

IIR IP — Data Output

```
131 p_output : process (i_rstb,i_clk,r_final_sum)
132 begin
133     done      <= '0';
134     if(i_rstb='1') then
135         o_data <= (others=>'0');
136         done   <= '0';
137     elsif(i_clk='1') then
138         done   <= '1';
139         o_data <= std_logic_vector(r_final_sum(33 downto 3));
140     end if;
141 end process p_output;
```

IIR IP — IIR Troubles



- Single Stage vs BiQuad

IIR IP — IIR Troubles



- Single Stage vs BiQuad
- Floating Point to Fixed Point

IIR IP — IIR Troubles

- Single Stage vs BiQuad
- Floating Point to Fixed Point
- Gains and Scaling

IIR IP — Single Stage Problems

- Number of coefficients increases dramatically

IIR IP — Single Stage Problems

- Number of coefficients increases dramatically
- Numerator coefficients approach zero

IIR IP — Single Stage Problems

- Number of coefficients increases dramatically
- Numerator coefficients approach zero
- Denominator coefficients approach infinity

IIR IP — Single Stage Problems

- Number of coefficients increases dramatically
- Numerator coefficients approach zero
- Denominator coefficients approach infinity

```
Numerator:  
0.067504806016373181  
0.27001922406549272  
0.40502883609823914  
0.27001922406549272  
0.067504806016373181  
Denominator:  
1  
-0.39064145319446159  
0.53430063715423204  
-0.084233712203843125  
0.020651424506043823
```

IIR IP — Single Stage Problems

- Number of coefficients increases dramatically
- Numerator coefficients approach zero
- Denominator coefficients approach infinity

```
Numerator:
0.00041659920440659937
0.0016663968176263975
0.0024995952264395961
0.0016663968176263975
0.00041659920440659937
Denominator:
1
-3.1806385488747191
3.8611943489942142
-2.1121553551109691
0.43826514226197993
```

```
Numerator:
0.067504806016373181
0.27001922406549272
0.40502883609823914
0.27001922406549272
0.067504806016373181
Denominator:
1
-0.39064145319446159
0.53430063715423204
-0.084233712203843125
0.020651424506043823
```

IIR IP — Single Stage Problems

- Number of coefficients increases dramatically
- Numerator coefficients approach zero
- Denominator coefficients approach infinity

```
Numerator:
0.067504806016373181
0.27001922406549272
0.40502883609823914
0.27001922406549272
0.067504806016373181
Denominator:
1
-0.39064145319446159
0.53430063715423204
-0.084233712203843125
0.020651424506043823
```

```
Numerator:
0.00041659920440659937
0.0016663968176263975
0.0024995952264395961
0.0016663968176263975
0.00041659920440659937
Denominator:
1
-3.1806385488747191
3.8611943489942142
-2.1121553551109691
0.43826514226197993
```

```
Numerator:
0.0047786506212785162
0.03822920497022813
0.13380221739579848
0.2676044347915969
0.33450554348949618
0.2676044347915969
0.13380221739579845
0.03822920497022813
0.0047786506212785162
Denominator:
1
-0.79504924865005511
1.316171195258671
-0.63488252768501452
0.42215843994659008
-0.11480285264112501
0.033379585260840769
-0.0039477257655511615
0.00030769332294375583
```

IIR IP — BiQuad

Pros

- Number of coefficients never change

IIR IP — BiQuad

Pros

- Number of coefficients never change
- Small coefficient magnitudes

IIR IP — BiQuad

Pros

- Number of coefficients never change
- Small coefficient magnitudes
- Easy to make generic

IIR IP — BiQuad

Pros

- Number of coefficients never change
- Small coefficient magnitudes
- Easy to make generic

Cons

- Numerator coefficients need scaled

IIR IP — BiQuad

Pros

- Number of coefficients never change
- Small coefficient magnitudes
- Easy to make generic

Cons

- Numerator coefficients need scaled
- Requires more hardware

IIR IP — BiQuad

Pros

- Number of coefficients never change
- Small coefficient magnitudes
- Easy to make generic

Cons

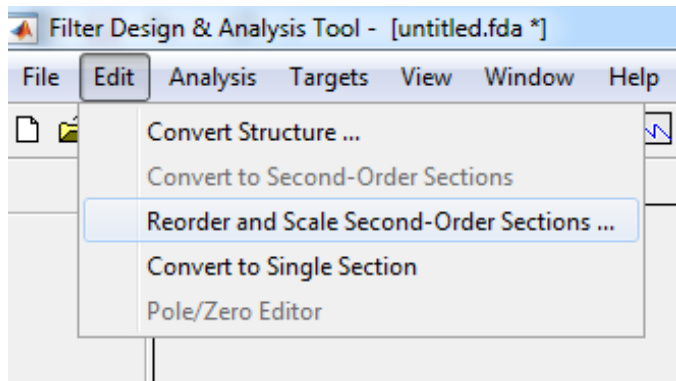
- Numerator coefficients need scaled
- Requires more hardware

```
-----  
Section #1  
-----  
Numerator:  
1  
2  
1  
Denominator:  
1  
-0.22705028708083497  
0.4514083390923061  
Gain:  
0.30608951300286774  
-----  
Section #2  
-----  
Numerator:  
1  
2  
1  
Denominator:  
1  
-0.16359116611362662  
0.045748876831938463  
Gain:  
0.22053942767957796  
-----
```

IIR IP — Fixed Point

Quantisation 1.15		
Numerator:	Fixed Point:	Hex:
1	32768	8000
2	65536	10000
1	32768	8000
Denominator:		
1	32768	8000
-0.262322431	-8596	FFFFFFDE6C
0.676883869	22180	56A4

IIR IP — Scaling



IIR IP — Scaling

Reordering and Scaling of Second-Order Sections

Reordering

☒ (None)
☐ Auto
☐ Least selective section to most selective section
☐ Most selective section to least selective section
☐ Custom reordering

Numerator Order: [1:4]

Denominator Order

☒ Use Numerator Order
☐ Specify: [1:4]

Scale Values Order

☒ Use Numerator Order
☐ Specify: [1:5]

Scaling

☒ Scale

I1 Linf L2 L1 linf

Less Overflow Highest SNR

Maximum Numerator: 2

Numerator Constraint: None

Overflow Mode: Saturate

Scale Value Constraint: Powers of Two

Max Scale Value: 16

Revert to Original Filter

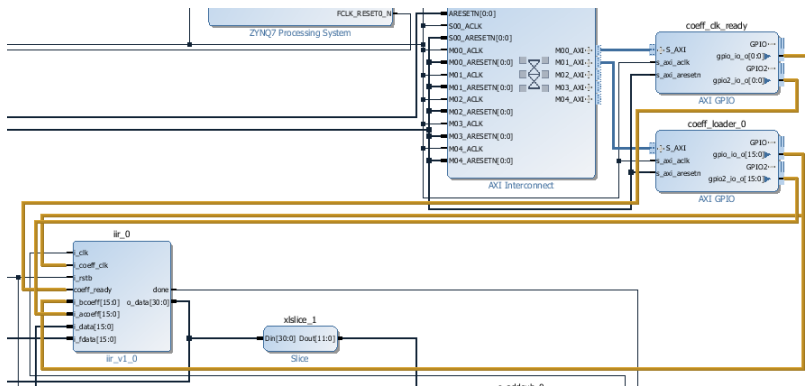
OK Cancel Help Apply

IIR IP — Scaling

Scaled with I1		
Numerator:	Fixed Point:	Hex:
0.380710926	12475	30BB
0.761421852	24950	6176
0.380710926	12475	30BB
Denominator:		
1	32768	8000
-0.262322431	-8596	FFFFFFDE6C
0.676883869	22180	56A4
Gain:	2.329372168	

Scaled with L2		
Numerator:	Fixed Point:	Hex:
0.408937915	13400	3458
0.817875829	26800	68B0
0.408937915	13400	3458
Denominator:		
1	32768	8000
-0.262322431	-8596	FFFFFFDE6C
0.676883869	22180	56A4
Gain:	1.212884367	

Zynq Communication — Outside the IP



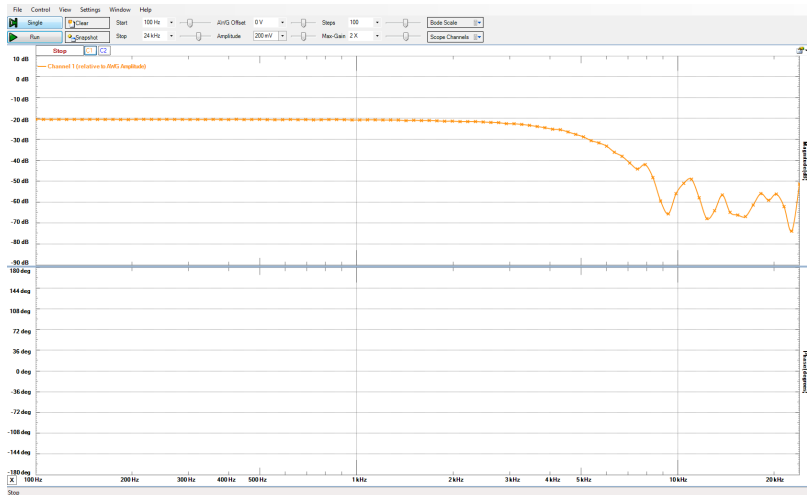
Zynq Communication — Inside the IP

```
53 p_coeff_input : process (i_rstb, i_coeff_clk, coeff_ready)
54 begin
55     if(coeff_ready='1') then
56         r_bcoeff    <= (others=>(others=>'0'));
57         r_acoeff    <= (others=>(others=>'0'));
58         coeff_loop  <= 0;
59     elsif(rising_edge(i_coeff_clk)) then
60         if(coeff_loop /= 4) then
61             r_bcoeff(coeff_loop) <= signed(i_bcoeff);
62             r_acoeff(coeff_loop) <= signed(i_acoeff);
63             coeff_loop <= coeff_loop + 1;
64         elsif(coeff_loop = 4) then
65             -- do nothing
66         end if;
67     end if;
68 end process p_coeff_input;
```

Zynq Communication — Inside the Zynq

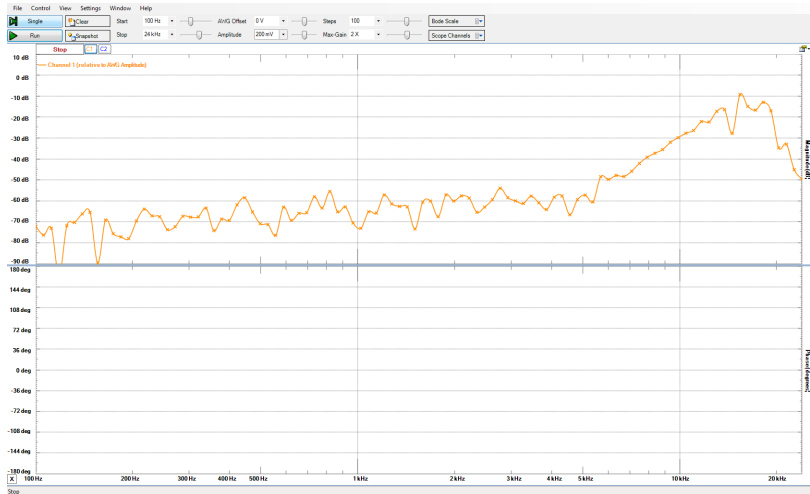
```
102     /* Stage 0 load*/
103     for(i = 0; i < 3; i++){
104         usleep(300);
105         XGpio_DiscreteWrite(&Gpio0, 1, 1); //coeff clk
106         XGpio_DiscreteWrite(&Gpio1, B_CH, b_coeff[i]);
107         XGpio_DiscreteWrite(&Gpio1, A_CH, a_coeff[i]);
108         XGpio_DiscreteWrite(&Gpio0, 1, 0);
109     }
110 }
```

Example Outputs — Lowpass



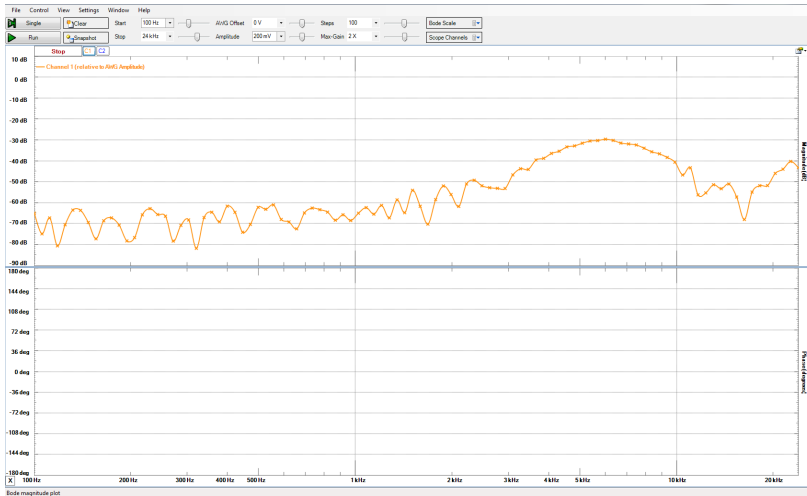
$$F_s = 48\text{kHz}, F_c = 10.8\text{kHz}$$

Example Outputs — Highpass



$$F_s = 48\text{kHz}, F_c = 10.8\text{kHz}$$

Example Outputs — Bandpass



$$F_s = 48\text{kHz}, \text{ Pass band} = 8.4\text{kHz to } 13.2\text{kHz}$$

Conclusion

