High performance IIR filters for interpolation and decimation

Dr David Wheeler, Technical Director, EnSilica, July 2013.

Abstract

This technical note looks at implementing high performance polyphase IIR filters with very low FPGA resource requirements. These structures have the property that order $NK+1$ filters can be implemented with just K multiplications. The filters can be shown to have low sensitivity to coefficient quantisation, leading to efficient implementation in fixed point. Furthermore the maximum gain at any node in a filter stage is bounded meaning that only 1 bit of headroom is required for intermediate calculations. A general architecture suitable for pipelining and mapping to Xilinx DSP slices is presented. Finally a number of multiplierless 5th and 10th order elliptic filter designs are presented which are applicable to efficient polyphase interpolation and decimation. The group delay variation can be minimised by adding all-pass equaliser sections.

Introduction

In most cases a design engineer will choose an FIR filter for his application because these are well understood and supported by good design and IP implementation tools. Xilinx FIR Compiler is an excellent tool for mapping coefficients generated by MATLAB into DSP and FPGA logic resources. However an IIR filter can be designed to meet a particular filtering specification using significantly lower FPGA resources. The main drawback of choosing an IIR filter is that it does require some specialist knowledge to drive the design tools, usually followed by hand coded RTL. However there are modern tools like Xilinx System Generator that can be used to autogenerate HDL once the design is architected and expressed as fixed point. Xilinx provide a good overview of IIR filters in Ref.1.

MATLAB can decompose an IIR filter design into a cascade of lower order sections. The cascade shown in Figure 1 has better numerical properties than implementing the high order difference equation directly. Typically second order biquadratic sections (SOS) are chosen for the low order filters and implemented using the Direct Form 1 structure shown in Figure 2. Each SOS requires 4 multiplies, 3 adds and some rounding to reduce the bit width for the next filter in the cascade. A fixed point implementation usually requires a further multiplier to scale between SOSs.

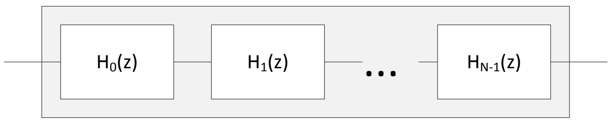


Figure 1: Cascade of filters

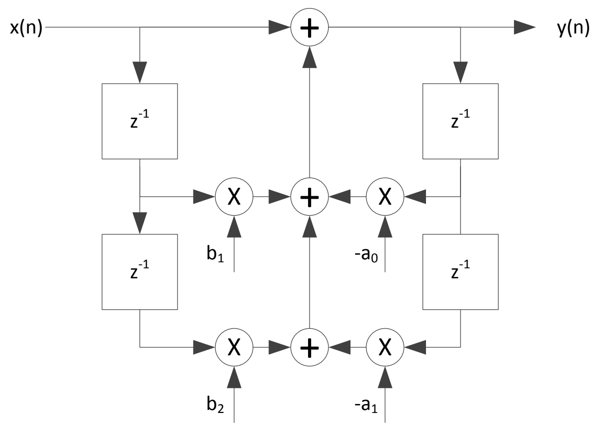


Figure 2: Biquadratic second order section

However the cascade decomposition is not an efficient architecture for interpolators or decimators because it doesn't take advantage of polyphase decomposition. For instance during interpolation the input data is upsampled by inserting $N-1$ zeros between samples, raising the input rate to the output rate, before applying to the filter. The IIR filter cascade cannot take advantage of the zero inputs to reduce computation because of feedback paths in each SOS.

IIR Filters for polyphase decomposition

In this paper we start with an architecture that maps to a N polyphase decomposition and show how this leads to very low complexity filters. We then give an example for the particular case of interpolation and decimation by a factor of 2. This design is compared to an equivalent FIR filter and the resource requirements are determined. This architecture has low sensitivity to coefficient quantisation and this is demonstrated by way of a multiplierless 5th order design. Some other simple multiplierless filters are presented together with cascaded filters giving 10th order response.

The filters in this paper are based on the parallel filter decomposition architecture given in Figure 3a.

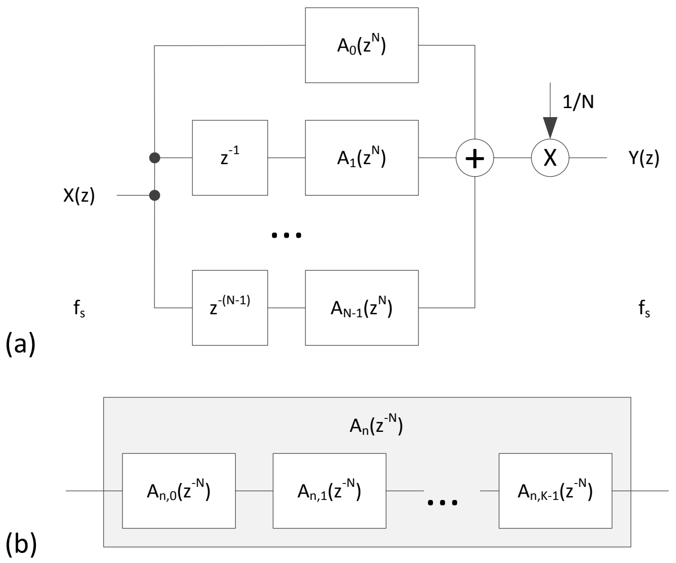


Figure 3: (a) Parallel form of IIR filter, (b) each branch is a cascade of elementary filters

In this parallel form the filter on each branch is successively delayed by one more than the previous branch. Furthermore the filters in each branch are constrained to be N-band. The delays in each branch operate at the input/output sample rate, but each $A_n(z^N)$ filter has terms involving only powers of z^{-N}, which means that the difference equation operates on every Nth input/output sample ignoring all samples in-between. An equivalent interpretation for these filters is that the frequency response repeats N times around the unit circle.

The transfer function is given by the sum of delayed N-band all pass filters $A_n(z^N)$.

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{N}\sum_{n=0}^{N-1} A_n(z^N)z^{-n}$$

Furthermore let each all pass filter be expressed as the cascade of elementary all pass sections, as shown in Figure 3b. The total number of all pass sections is K and the order of $H(z)$ is $NK + 1$

$$A_n(z^N) = \prod_{k=0}^{K_n-1} A_{n,k}(z^N)$$

$$A_{n,k}(z^N) = \frac{a_{n,k} + z^{-N}}{1 + a_{n,k}z^{-N}}$$

$$K = \sum_{n=0}^{N-1} K_n$$

An elementary all pass section implemented in Direct Form 1 is shown in Figure 4. It consists of two N sample delay elements and a single coefficient multiplier.

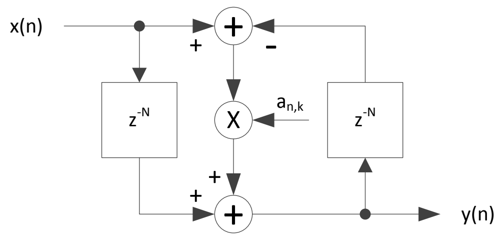
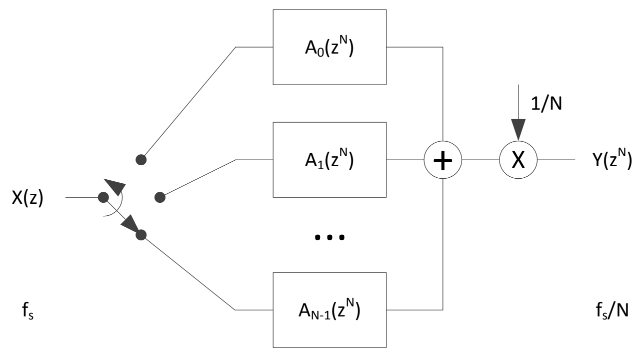


Figure 4: Elementary all pass section

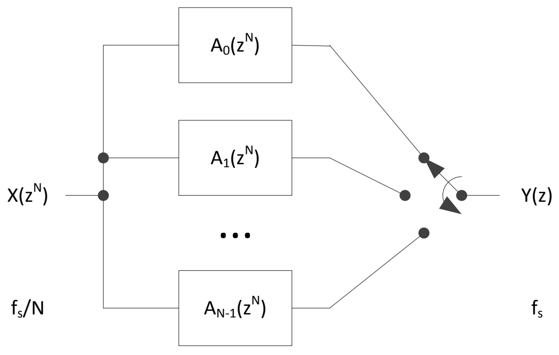
Apart from being canonical in multipliers there are other advantages to using Direct Form 1 which will map efficiently to DSP slices and share the delay resources when cascaded.

Interpolation and decimation

This filter architecture naturally maps to the decimation and interpolation structures in Figure 5. The commutating switches, which move on every sample at the higher sampling rate, replace the delay elements in Figure 3a.



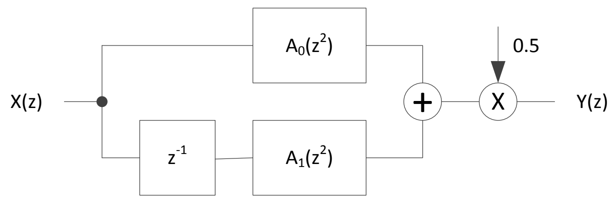
(a)



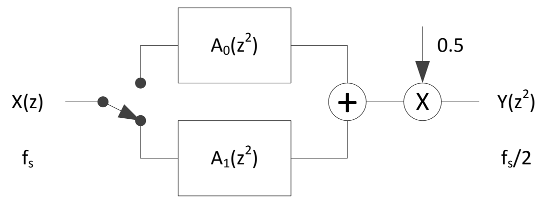
(b)

Figure 5: (a) Decimation (b) interpolation

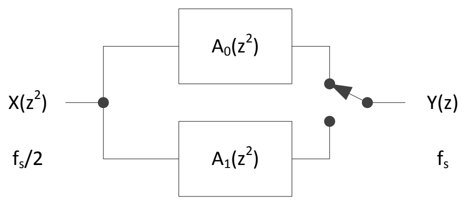
For interpolation and decimation by N then we can either design around the optimum structure in Figure 5 or design a cascade of prime factor rate conversions. An advantage of decomposing into prime factors, Ref.2, is that it can ease the optimization problem because there are fewer free variables, and the resulting filter cascade is very close to the optimum. In many applications interpolation and decimation is by powers of two, which can be achieved by repeated interpolation or decimation by two. Without loss of generality we can turn our attention to the special case of $N = 2$. The 2-branch half-band parallel form of this filter is shown in Figure 6 together with special cases of decimation and interpolation which clearly maps onto this polyphase architecture.



(a)



(b)



(c)

Figure 6: Two branch parallel form (a) lowpass filter (b) decimation (c) interpolation

The second order elementary all pass section is shown in Figure 7. This elementary section, when cascaded forms a ladder structure because the z^{-2} delay between sections can be efficiently shared. For example Figure 8 has a cascade of two sections to show that clearly.

This structure can be efficiently pipelined and mapped to a Xilinx DSP48E1 Slice. To implement the two branch filter of Figure 6a requires the addition of only 4 external registers and one adder as shown in Figure 9. The A, B, C, D and P ports follow the nomenclature given in Ref.3. The DSP Slices can be cascaded to implement higher order filters and because of the 2 branch structure the pipeline delays in each branch should be equal. Note that the DSP slice cannot be fully pipelined, which requires 3 internal registers, because the feedback path only has a delay of 2. It is also noted from Ref.3 that using the M register achieves a higher frequency and lower power than registering at the inputs instead. For $N = 3$ and higher then it should be possible to get the maximum operating frequency from a DSP slice. It is also possible to get the maximum DSP slice pipelining by operating the filter in a 2 channel TDM mode, where the delay elements double up; the A input feedback path would use registers A1 and A2 in the slice and the delay before the C input would increase from 3 to 5.

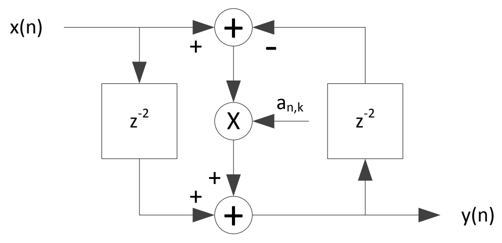


Figure 7: Elementary second order all pass section

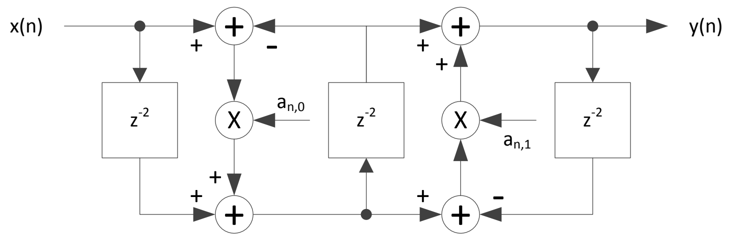


Figure 8: Cascading two elementary second order all pass sections

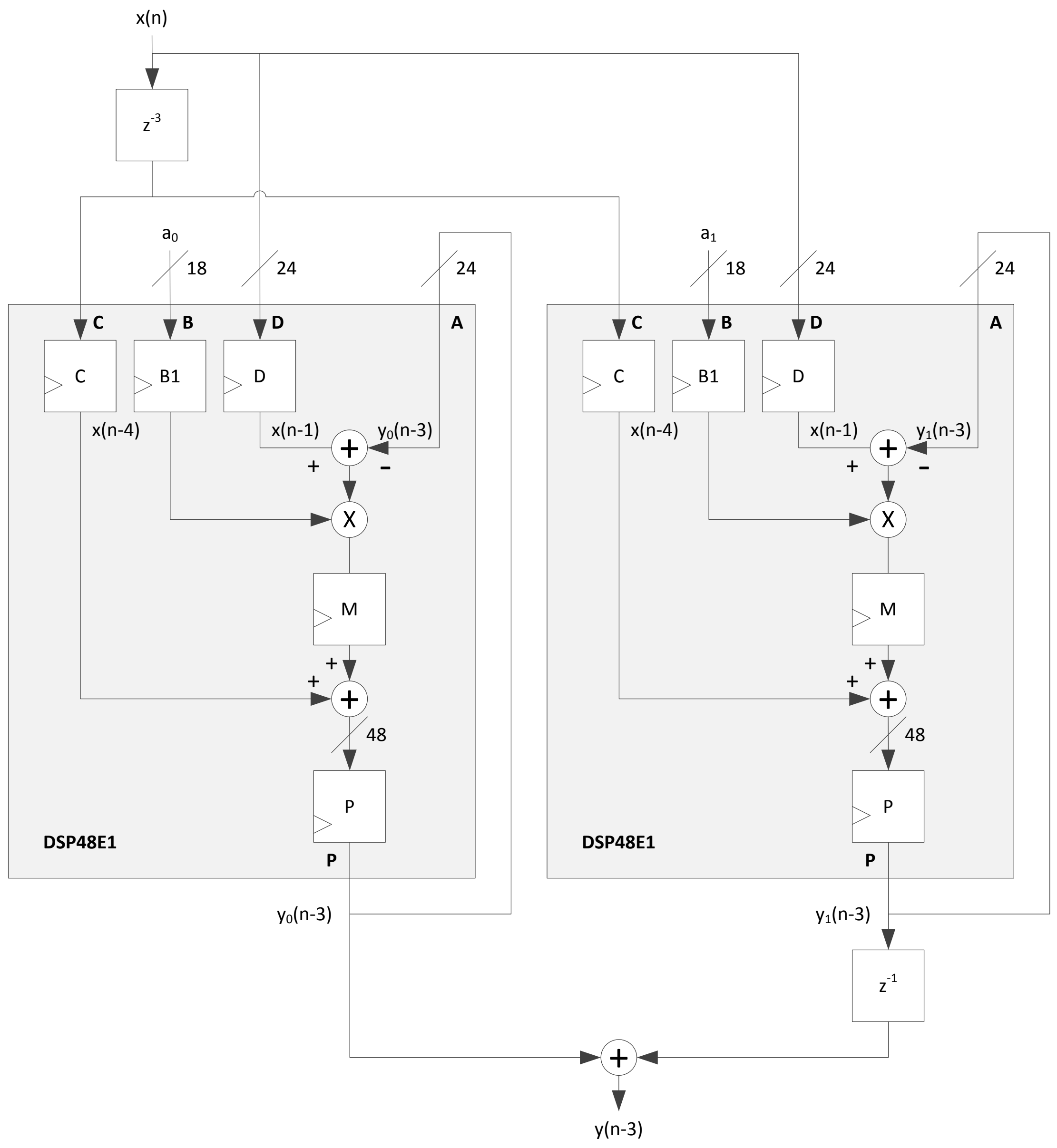


Figure 9: Pipelined 2 branch filter mapped to Xilinx DSP48E1s

It can be shown that the low pass filter in Figure 6a has an overall order of 5, even though it's composed of two second order all-pass sections and a delay element. Now it should be clear that, in an ideal case, only 2 multiplies and 5 adds are required to implement this 5th order IIR filter compared to 10 multiplies and 8 adds if implemented in SOS. And there are even more savings if we consider decimation by 2, where only one branch is active for each input sample. This is equivalent to dividing up the input sequence into two half-rate sequences containing odd and even samples respectively. These are then applied to the all-pass branches and summed. Furthermore the delay elements in each elementary section may then operate at the lower rate so halving the storage requirements. This maps efficiently into a DSP slice because an external adder can be used to accumulate the two branch filter outputs. So only one DSP slice is required to implement a 5th order decimate by 2 IIR filter, as shown in Figure 10.

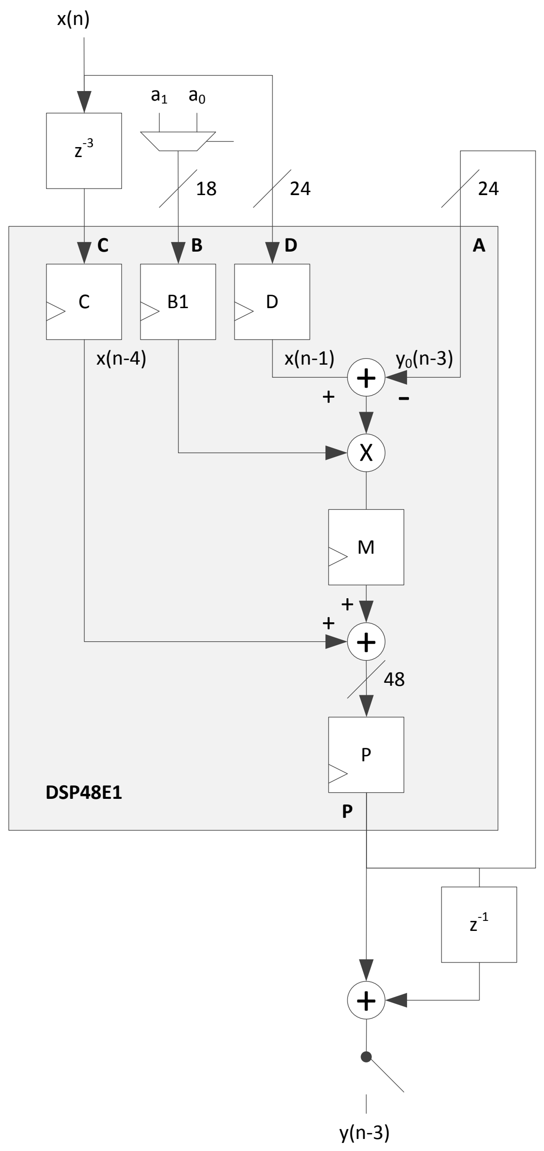


Figure 10: Decimation by 2 mapped to Xilinx DSP48E1

Similar observations apply to interpolation where each branch filter provides one interleaved output sample in turn. Only one DSP Slice is required to support interpolation.

Quantization and headroom

These second order structures have been termed Wave Digital Filters (WDF) because they emulate a doubly terminated lossless ladder network in a classical analogue filtering. The design of such filters having elliptic response is comprehensively covered in the literature. This work together with the bilinear transform for translating between the analogue and digital domain provide a powerful way to design digital elliptic filters. Another advantage of being derived from a ladder filter is that these structures inherit low sensitivity to coefficient quantization. This means that the 18-bit wordlength for coefficients will be more than sufficient to implement filters with 100 dB stopband requirements without loosing the equiripple properties. In Ref.4 an analysis of the steady state gain characteristics of an elementary section showed that the maximum gain at the worst case arithmetic node was 2.0, occurring at the pre-adder output. This has important consequences for fixed point implementation because it means we only need 1 additional bit at the pre-adder output. In a Xilinx DSP48E1 slice there's no saturation logic at the pre-adder output, so limiting the A and D inputs to 24 bits will allow the filter to operate without numerical overflow, but more importantly to the maximum internal precision of the DSP slice. It has been noted that although the steady state gain is limited to 2.0, transients in a step response can exceed 2.0, so for this reason it is recommended that only 23-bits are used and sign extended for safe operation. For some data sources like 24-bit music the full dynamic range can be applied taking advantage of prior knowledge of the signal characteristics.

If your input data is less than 24-bits then it should be left aligned in the word, thereby providing some fractional bits. For instance with 16 bit input data a good choice would be [1 guard bit, 16 data bits, 7 fractional bits]. In general 3 fractional bits are sufficient for good accuracy compared to floating point.

Generating filter coefficients

Methods to generate coefficients for any order elliptic filter implemented by two-branch structures are adequately covered in Ref.5.

Examples

Figure 11 shows a 5th order (2 coefficients $a_0 = 0.1380$, $a_1 = 0.5847$) WDF filter with normalised passband 0.125, compared to a 22 tap FIR filter, designed using Parks-McClellan to meet the same passband ripple and stopband attenuation. This shows a typical 4 to 1 saving in filter order and the WDF requires just 2 multiplies compared to 11 for the FIR (taking the symmetry into account)

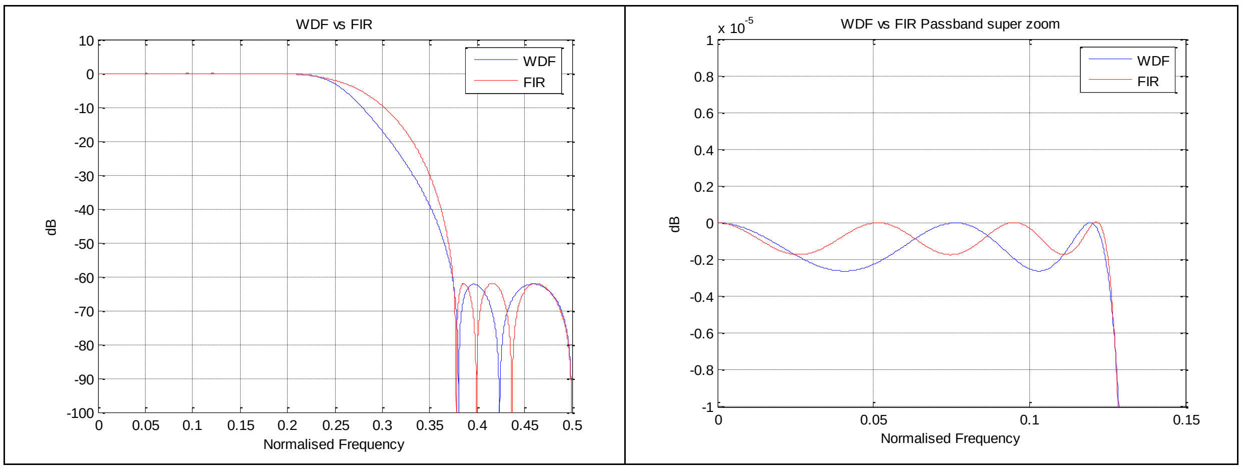


Figure 11: Comparison of FIR and WDF

In a halfband WDF the stopband and passband ripples are not independent, however setting any reasonable stopband attenuation leads to negligible passband ripple, for example it was 10^{-6} dB for the filter in Figure 11. The FIR has an advantage in being able to specify these two design parameters independently, so that a higher passband ripple can be specified to aid meeting a stopband requirement for a given order.

Next we look at what can be achieved without using DSP48E1, but quantising the coefficients heavily to implement a filter in just a few LEs. Recall that since WDF filters are based on ladder analogue prototypes then we should be able to find low bit density coefficients for a range of specified passbands. Where possible the bits are written in a canonical decomposition

Normalised passband	Coefficients	Stopband attenuation (dB)
0.1	1/8 9/16 = 1/2+1/16	69
0.175	3/16 = 1/8+1/16 21/32 = 1/2+1/8+1/32	44
0.19	7/32 = 1/4-1/32 89/128 = 1/2+1/8+1/16+1/128	39

Although the stopband attenuation of a 5th order filter with 0.19 passband is modest, simply cascading two of these filters produces a 10th order with -78 dB stopband.

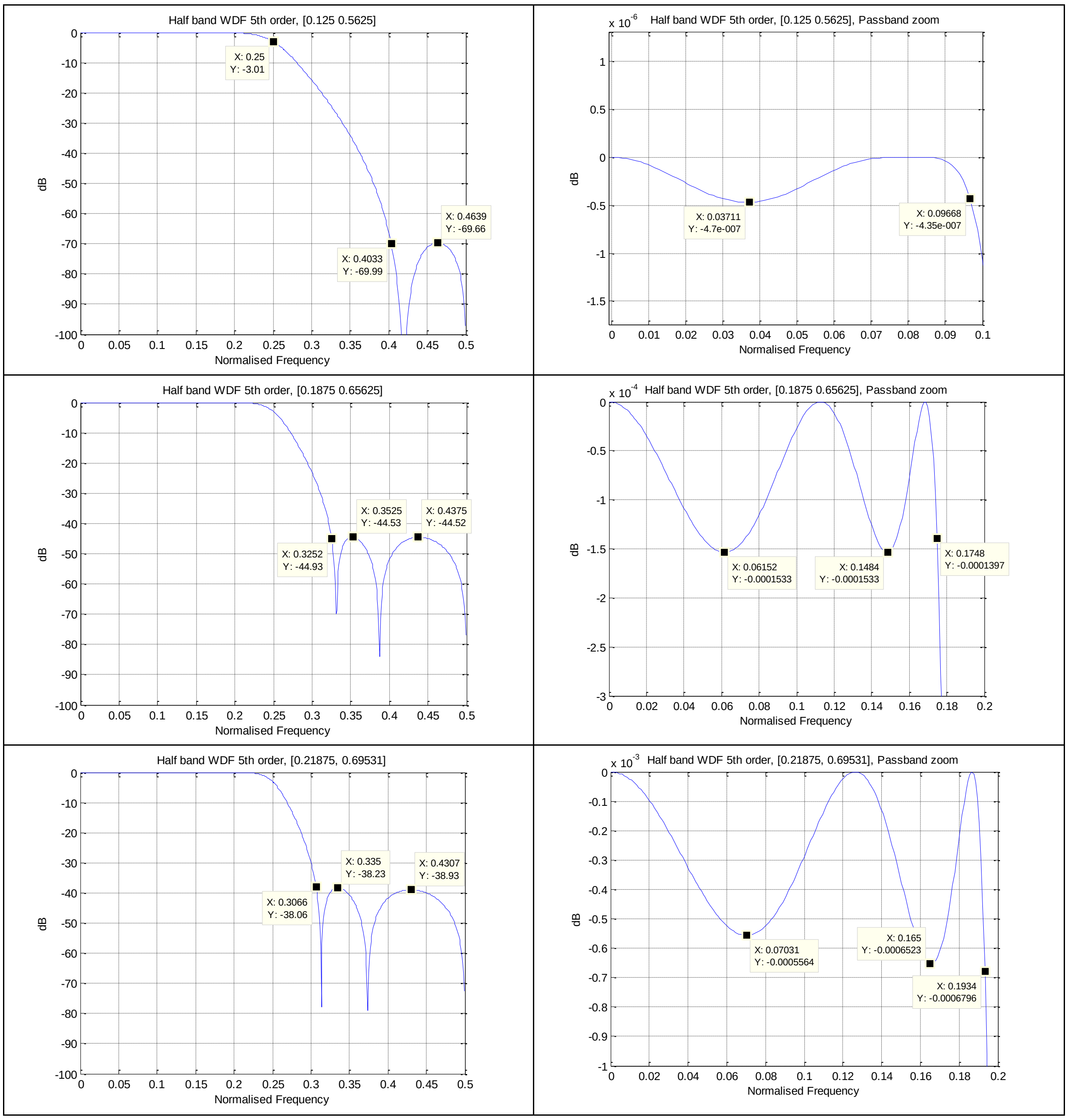


Figure 12: Multiplierless filters (a) 5th order 0.1 passband (b) 5th order 0.175 passband (c) 5th order 0.19 passband

The code for the first of these decimating filters is shown in Figure 13, which highlights how simple the implementation is involving just a few registers and adders to get a 70 dB stopband.

```
module wdf5 #(parameter WIDTH = 16, FRAC = 3, GUARD = 2)
(
    input                             clk,
    input                             reset_n,
    input                             enable,
    input      signed [WIDTH-1:0]     data_in,
    output reg signed [WIDTH:0]       data_out,
    output reg                        valid_out
);

    reg  signed [WIDTH+FRAC+GUARD-1:0] x, x_1r, x_2r;
    reg  signed [WIDTH+FRAC+GUARD-1:0] sum_1r, sum_2r;
    reg                                state;

    wire signed [WIDTH+FRAC+GUARD-1:0] p, p0, p1;
    wire signed [WIDTH+FRAC+GUARD-1:0] sum, diff, dec;

    always @(posedge clk)
    begin
        if (reset_n == 1'b0)
        begin
            x <= 0;
            x_1r <= 0;
            x_2r <= 0;
            sum_1r <= 0;
            sum_2r <= 0;
            state <= 0;
            data_out <= 0;
            valid_out <= 0;
        end
        else if (enable == 1'b1)
        begin
            x <= data_in <<< FRAC;
            x_1r <= x;
            x_2r <= x_1r;
            sum_1r <= sum;
            sum_2r <= sum_1r;
            state <= ~state;
            data_out <= (dec >>> FRAC+1) + dec[FRAC];
            valid_out <= state;
        end
    end

    assign diff = x - sum_2r;
    assign p0 = diff >>> 3;                      // 0.125 = 1/8
    assign p1 = (diff >>> 1) + (diff >>> 4);     // 0.5625 = 1/2+1/16
    assign p = state ? p1 : p0;
    assign sum = p + x_2r;
    assign dec = sum_1r + sum_2r;

endmodule
```

Figure 13: Code example for multiplierless 5th order IIR decimator

Group delay

One of the criticisms of IIR filters is that they are not linear phase. In practical terms this means that the group delay is not flat, so spectral components close to the band edge are delayed by more than those at lower frequencies. However it is relatively easy to design an all-pass equaliser to make the variation arbitrarily flat over a passband. The equaliser may be implemented as a cascade of elementary all-pass sections, very similar to those already discussed. A N-band equaliser, shown in Figure 14, can be written as

$$E(z^N) = \frac{e - z^{-N}}{1 - ez^{-N}}$$

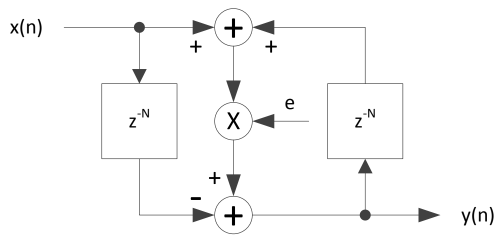


Figure 14: Elementary all-pass equaliser

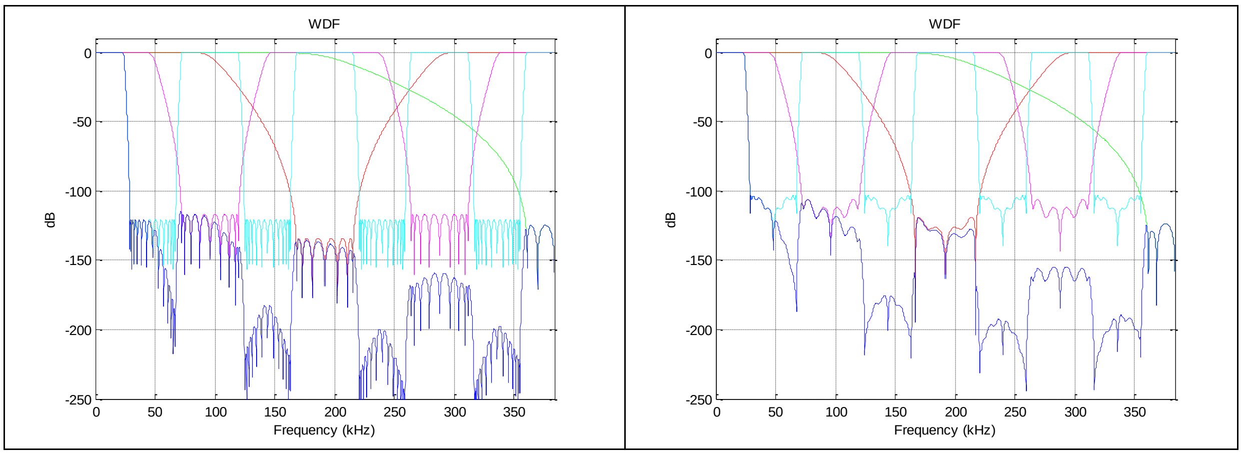
This has the same desirable properties already covered, for instance it can be implemented at the lower sample rate for both decimation and interpolation. Designing this filter is a classical problem well covered in existing literature.

Audio interpolation example

Very higher order filters can be easily designed and their coefficients quantised to low bit densities. To illustrate the point we designed an interpolating WDF filter taking DVD audio data at 16-bits, 48 kHz and upsampled it 16x. This was used to drive an audiophile DAC, bypassing the internal upsampling filter to provide better audio quality. The final sample rate was 768 kHz, and the filter was specified flat to 19.2 kHz, -3 dB at 24 kHz. This design had to replace a 60th order Butterworth filter implemented as SOSs, having a maximally flat passband and very steep rolloff. The solution was to use a cascade of four interpolation by 2 filters with the following specification

H2: passband = 1/5,K = 7, stopband >100 dB
H4: passband = 1/8, K = 4, stopband > 100 dB
H8: passband = 1/16, K = 3, stopband >100 dB
H16: passband = 1/32, K = 2, stopband > 100 dB

This filter requires (1*7)+(2*4)+(4*3)+(8*2) = 43 multiplies per input sample to generate the 16 output samples. Comparing this to the original Butterworth filter, which required 16*(60/2)*4=1920 multiplies, shows the efficiency of these WDF filters.



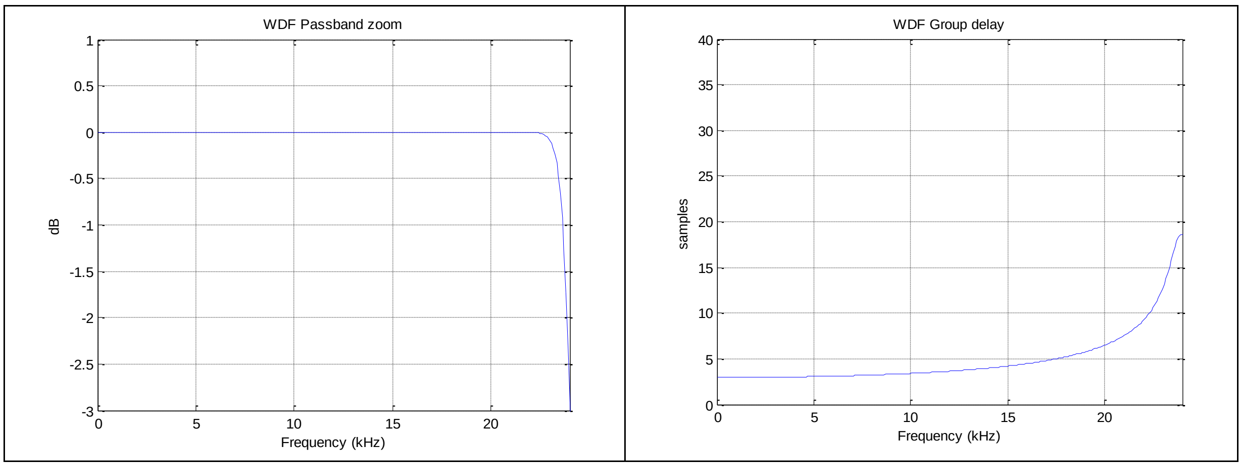


Figure 15: High specification multistage interpolation filter design (a) stage responses, blue is the combined (b) quantised coefficients to 18-bits (c) passband response (d) group delay

With the coefficients quantised to 18 bits the frequency response is still good for a 100 dB stopband. Whilst the group delay is not completely flat it is well controlled out to 20 kHz.

Conclusion

We have presented an IIR filter architecture that naturally maps to interpolation and decimation. This structure is shown to be more resource efficient than IIR second order sections or even a FIR. It maps well to the preadd/multiply/postadd of a DSP48E1 slice, and is robust to fixed point quantisation effects of coefficients to 18-bits to give a controlled 100 dB stopband. Some multiplierless 5th order designs are permissible for special passband widths and they can be mapped to a few registers and adders in logic to save DSP resources.

References

[1] Xilinx white paper WP330, "Infinite Impulse Response Filter Structures in Xilinx FPGAs", August 2009

[2] P.P. Vaidyanathan. "Multirate Systems and filter banks". Prentice-Hall, Englewood Cliffs, NJ, 1993.

[3] Xilinx, 7 Series DSP48E1 Slice, User Guide, v1.5 April 3, 2013.

[4] Artur Krukowski, Richard Morling, Izzet Kale. "Quantization effects in Polyphase N-Path IIR Structure", IEEE Transactions on instrumentation and measurement, vol.51, no.6, pp. 1271-1278, December 2002.

[5] R.A. Valenzuela and A.G.Constantinides, "Digital signal processing schemes for efficient interpolation and decimation" Proc.IEE, pt G, vol.130, no.6, pp. 225-235, 1983.