ON THE DESIGN AND MULTIPLIER-LESS REALIZATION OF DIGITAL IF FOR SOFTWARE RADIO RECEIVERS

K. S. Yeung and S. C. Chan

Department of Electrical and Electronic Engineering,
The University of Hong Kong, Pokfulam Road, Hong Kong.
e-mail: ksyeung@eee.hku.hk, scchan@eee.hku.hk

ABSTRACT

This paper proposes to reduce the decimation factor of the multistage decimator so that its output can be fed directly to the Farrow structure for sample rate conversion, eliminating the need for another L-band filter for upsampling. Furthermore, it was found out that the programmable FIR filter can be replaced by a half-band filter placed immediately after the Farrow structure, i.e. after sample rate conversion. This significantly reduces the complexity of the proposed software radio receiver because this half-band filter, which consists of fixed filter coefficients, can be implemented efficiently without multiplication using SOPOT coefficients. As the coefficients of the multistage decimators and the subfilters in the Farrow structure are also fixed, they can also be implemented efficiently using the SOPOT coefficients. As a result, apart from the limited number of multipliers required in the Farrow structure, the entire digital IF can be implemented without any multiplication. Design example is given to demonstrate the effectiveness and feasibility of the proposed approach.

I. INTRODUCTION

Software radio is a general hardware/software platform for supporting inter-communication between different wireless communications systems [10][11]. The basic idea of an ideal software radio receiver is to digitize the received signal using high-speed ADCs and to process it by a sophisticated programmable system, probably consisting of a combination of hardware that is re-configurable or programmable, and digital signal processors (DSP). Due to various limitations of current digital technology and signal converters, most software radio architectures considered digitalize the down-converted signal at intermediate frequency (IF). It is envisioned that, with the availability of low-cost and high-speed signal converters with reasonable accuracy, software radio employing digital signal processing technique is a cost effective means to offer more flexibility and less sensitivity to analog components than traditional receiver employing analog IF.

Figure 1 shows a commonly used IF architecture for software radio receiver. The IF-signal is digitized at a bandwidth of 20 to 40 MHz. A programmable digital decimator and a sample rate changer are employed to isolate the desired user's channel from the signal spectrum and convert it to an appropriate sampling rate for further processing in the DSP [10]. The digital decimator will normally consist of multiple stages of decimator. As the sampling rate of the baseband signal is much lower than that at the IF, the output of each stage in the decimator will consist of a bandlimiting (anti-aliasing) digital filter and a downsampler (decimator) to filter out the unwanted signals and lower the sampling rate. By selecting an appropriate number of stages, different integer decimation ratios can be implemented. A programmable FIR is usually needed to remove the residual interference from adjacent channels. It is because the sampling rate is usually not an integer multiples of the channel spacing. Hence, the multiple stages of decimation filters, which implement an integer decimation factor, are unable to remove this residual interference from adjacent channels. Together with the sample rate changer (SRC), which provides the necessary rational or even irrational rate-change factor, it is now possible to accommodate signals with a wide variety of bandwidths, required by different communication standards.

The design and implementation of the programmable decimator and sample rate changer, however, is far more complicated. There are several important contributions in the hardware-efficient structure for implementing the programmable receiver and most of them are based on the CIC filter and its variants [1]-[3]. In addition, it is usually assumed that the programmable FIR and the SRC immediately after it are fast enough to handle the decimated input signal. One drawback of this conventional structure is that the output of the multistage decimator, which is obtained by downsampling the high-rate IF signal from the ADC, has to be upsampled again by the L-band filter in order to carry out the arbitrary sample rate conversion. Another important problem, which limits the throughput of the system for wideband signal, is the high processing requirement of the programmable FIR filter. Considerably number of high-speed general-purpose multipliers is usually required for their implementation for wideband signals.

In this paper, we propose to reduce the decimation factor of the multistage decimator so that its output can be fed directly to the Farrow structure for sample rate conversion, eliminating the need for another L-band filter for upsampling. Furthermore, it was found out that the programmable FIR filter can be replaced by a half-band filter (HBF) placed immediately after the Farrow structure, i.e. after sample rate conversion. This significantly reduces the implementation complexity of the proposed software radio receiver because this half-band filter, which consists of fixed filter coefficients, can be implemented efficiently without multiplication using sum-of-powers-of-two (SOPOT) coefficients. As the coefficients of the multistage decimators and the subfilters in the Farrow structure are also fixed, they can also be implemented efficiently using the SOPOT coefficients. As a result, apart from the limited number of multipliers required in the Farrow structure, the entire digital IF can be implemented without any multiplication.

The rest of this paper is organized as follows: Section II is devoted to the principle and design of the proposed digital IF. The design of the Farrow-based fractional-delay digital filters (FDDF) for the sample rate changer is presented in Section III. Section IV describes the design and multiplier-less implementation of the FDDF, multistage decimator and the half-band filter in the digital IF. This is then followed by a design example in Section V. Finally, conclusion is drawn in Section VI.

II. PROGRAMMABLE DECIMATOR AND SRC

In this section, the design of the programmable decimator and SRC for software radio receivers is outlined. As mentioned earlier, conventional software radio receiver uses multiple stages of downsamplers, followed by a programmable FIR and SRC for sample rate conversion. The multistage decimator may consist of a cascade of CIC and half-band filters or other low order FIR filters such as the ISOP [2]. The design of programmable SRCs with arbitrary conversion factors was first discussed in the paper by Ramstad [12]. The input signal is first up-sampled by a factor L by inserting L-1 zeros between every sample. This creates L-1 images in the frequency domain, which are then removed by an L-band filter with spectral support from $-\pi/L$ to π/L . If L is sufficiently large, further interpolation by an irrational number can be achieved simply by a second or higher order polynomial interpolation. Alternatively, the Farrow structure [5], which is usually used to realize tunable fractional delay digital filter, can also be used to realize the SRC. One drawback of the conventional structure is that the output of the multistage decimator, which is obtained by downsampling the high-rate IF signal from the ADC, has to be upsampled again by the *L*-band filter to carry out the arbitrary sample rate conversion. In this paper, we propose to reduce the decimation factor of the multistage decimator so that its output can be fed directly to the Farrow structure for sample rate conversion, eliminating the need for another *L*-band filter for upsampling. In other words, the high sampling rate of the IF signal of the software radio is utilize to simply the arbitrary sample rate conversion. Furthermore, it was found out that the programmable FIR filter, which is usually a bottleneck in software radio application for wideband signal, can be replaced by a half-band filter placed immediately after the Farrow structure, i.e. after sample rate conversion. This significantly reduces the implementation complexity of the new software radio receiver shown in Fig. 7 because the half-band filter, which consists of fixed filter coefficients, can be implemented without multiplication using SOPOT coefficients. In contrast, the programmable FIR filter usually requires considerably number of high-speed general-purpose multipliers to achieve a high system throughput.

To design the proposed programmable decimator, refer to Fig. 7, the IF-signal from ADC first passes through the optional CIC filter or its variants. The output of CIC filter is then fed to the multistage decimators denoted by LPF#3, LPF#2 and LPF#1. As mentioned earlier, the output of multistage decimators is fed directly to the Farrow-based FDDF for sample rate conversion, eliminating the need for another *L*-band filter for upsampling. Finally, the output of the FDDF is fed to the HBF instead of programmable FIR. To design each anti-aliasing filter, let ω_{pi} and ω_{si} be the passband and stopband edges of the *i*th anti-aliasing filter, respectively (relative to its input sampling rate F_{s-in} = 2). Then, the *i*th filter satisfies the following:

$$\omega_{pi} > \omega'_{pi-1} / M \ , \tag{1}$$

$$\omega_{si} < \left(2-\omega'_{si-1}\right)/ M \ , \tag{2}$$

where ω'_{pi} and ω'_{si} is the overall passband and stopband edges of previous *i* filters, and *M* is the arbitrary down-sampling ratio. The total down-sampling ratio M^* of proposed programmable decimator in Fig. 7 is given by

$$M^* = M_{CIC} \cdot 2^m \cdot M_I \ . \tag{3}$$

where M_{CIC}, which is a positive integer, is the decimation factor of the CIC filter or its variants and M_I, which is either a rational or even an irrational number, is the decimation factor of the SRC and *m* is the remaining number of decimators to be selected. In general, the structure of the FDDF is more complicated than that of other FIR filters. The main reason for putting the FDDF in back is that it can be operated at relatively low sampling rate and hence lower the power consumption. The design and implementation of the FDDF will be described next section in detail. Note that, for each filter design except the HBF and the CIC filter, the stopband edge should start from the transition band of the first aliasing folding of previous filters. Otherwise, the overall frequency response will give worse stopband attenuation because previous transition bands are not fully attenuated to the specification. On the other hand, if M^* is very large, in order to avoid the fractal phenomenon, the frequency response of each anti-aliasing filter must be zero at π .

III. Design Of The FDDF

In this section, the design and implementation of the FDDF based on the Farrow structure [4]-[6] is described. More precisely, the output of the FDDF, $y[(m+D+d)T]$, is given by

$$y[(m+D+d)T] = \sum_{n=0}^{N} x[(m-n)T]\cdot h(n,d) \ . \tag{4}$$

where $x[mT]$ is the input signal sampled at a period *T*. $h(n,d)$ is the impulse response of the FDDF with delay $D+d$, where *D* is the group delay of the filter at $d=0$. The *z*-transform $H(z,d)$ of the impulse response associated with delay parameter *d* is given by

$$H(z,d) = \sum_{n=0}^{N} h(n,d) z^{-n} \ . \tag{5}$$

To avoid the implementation of a large number of filters with different delays, Farrow [5] proposed to approximate each impulse response with the L^{th} order polynomial in variable *d* as

$$h(n,d) = \sum_{l=0}^{L} c_{l,n} d^{l} \ . \tag{6}$$

The *z*-transform of (6) is then given by

$$H(z,d) = \sum_{l=0}^{L} \left[\sum_{n=0}^{N} c_{l,n} z^{-n} \right] d^{l} = \sum_{l=0}^{L} C_l(z) d^{l} \ , \tag{7}$$

where $C_l(z) = \sum_{n=0}^{N} c_{l,n} z^{-n}$ are called the subfilters. The coefficients $c_{l,n}$ are determined by polynomial interpolating the impulse response designed with delays equally spaced within a range chosen to be $d = [-0.5, 0.5]$. Alternatively, it can be obtained in using a least squares approach [9]. Thus the FDDF with delay *d* can be implemented by passing the signal through the subfilters followed by the multiplication with the appropriate powers of *d* as shown in Fig. 2.

For sample rate conversion, as mentioned earlier, the output of the multistage decimator is fed directly to the Farrow-based FDDF, eliminating the need for another *L*-band filter for upsampling. It should be noted that the coefficients in the subfilters need not be computed every time, as a new sample gets into the tapped-delay line of the FDDF. Only the delay parameter *d* value is changed as the new sample comes in. The unit based on M_I is required to generate the *d* value for each output sample, it also determines whether to shift *s* input samples into the tapped-delay line of the FDDF. Let d_k be the delay value at the k^{th} output sample and s_k be the number of input samples shifted into the tapped-delay line. Then we have

$$d_k = k \cdot M_I - \left[k \cdot M_I \right], \tag{8}$$

$$s_k = \left[k \cdot M_I \right] - \left[(k-1) \cdot M_I \right]. \tag{9}$$

where $k = 0, 1, 2, ...$ and $[\cdot]$ denotes the largest integer but less than or equal to the value inside the squared bracket.

IV. Multiplier-Less Realization

In this section, we describe the multiplier-less realization for the proposed programmable decimator (including HBF, FDDF, LPF#1, LPF#2 and LPF#3). In particular, the coefficients in the FIR filters and the subfilters are represented as SOPOT coefficients [7]. For further complexity reduction, the multiplier-block (MB) technique [8] is also used. The basic idea of MB is to reduce the redundancies found in implementing all SOPOT coefficients by removing any possible common sub-expressions in their representations. To be more specific, assume that the coefficients $c_{l,n}$ in the subfilters $C_l(z)$ are represented in the following SOPOT form

$$\hat{c}_{l,n} = \sum_{r=1}^{R} u_{l,n,r} 2^{a_r} \ , \tag{10}$$

with $u_{l,n,r} \in \{-1, 1\}$ and $a_r \in \{-g, \ldots, -1, 0, 1, \ldots, g\}$, where g is a positive integer and its value determines the range of the coefficients, and *R* is the number of terms used in the coefficient approximation. The coefficient multiplication can then be implemented as limited number of shifts and additions, giving rise to multiplier-less realization. These SOPOT coefficients can be obtained by a number of methods. Here, we shall employ the random search algorithm reported in [6]. Let the ideal frequency response of the FDDF be

$$H_I(e^{j\omega}, d) = \begin{cases} e^{-j\tau(d)\omega} & ,0 < |\omega| < \pi \cdot \omega_{pi} \\ 0 & ,\pi \cdot \omega_{si} < |\omega| < \pi \end{cases}. \qquad (11)$$

where $\tau(d) = D + d$. To determine the SOPOT coefficients, the following objective function is minimized for $d = [-0.5, 0.5]$:

$$\min \langle T_{SOPOT} \rangle \text{ subject to } \begin{cases} \delta_p < \delta_{p-\max} \\ \delta_s < \delta_{s-\max} \\ \delta_d < \delta_{d-\max} \end{cases}. \qquad (12)$$

where δ_p is the passband peak ripple error

$$\delta_p = \max_{0<|\omega|<\pi \cdot \omega_{pi}} \left\langle \left| H_I(e^{j\omega}, d) - \hat{H}(e^{j\omega}, d) \right| \right\rangle, \qquad (13)$$

δ_s is the stopband peak ripple error

$$\delta_s = \max_{\pi \cdot \omega_{si}<|\omega|<\pi} \left\langle \left| H_I(e^{j\omega}, d) - \hat{H}(e^{j\omega}, d) \right| \right\rangle, \qquad (14)$$

and δ_d is the group delay peak ripple error

$$\delta_d = \max_{0<|\omega|<\pi \cdot \omega_{pi}} \left\langle \left| \tau(d) - \hat{\tau}(d) \right| \right\rangle. \qquad (15)$$

T_{SOPOT} is the total number of terms for implementing total SOPOT coefficients, $\hat{H}(e^{j\omega}, d)$ is the frequency response of the given SOPOT coefficients and $\delta_{p-\max}$, $\delta_{s-\max}$ and $\delta_{d-\max}$ are given in the specifications. In the random search algorithm, the real-valued coefficients using the least squares approach in [9] are obtained. Let $\boldsymbol{b}$ be the vector containing these real-valued coefficients. Then, the algorithm repetitively calculates a candidate SOPOT vector $\boldsymbol{b}_c$ given by

$$\boldsymbol{b}_c = \lfloor \boldsymbol{b} + \lambda \boldsymbol{b}_p \rfloor_{SOPOT}, \qquad (16)$$

where $\boldsymbol{b}_p$ is a random vector with elements chosen in the range ± 1 , λ is a user-defined variable used to control the size of the neighborhood to be searched, and $\lfloor \cdot \rfloor_{SOPOT}$ is the rounding operator that converts every element inside the input vector to its closest SOPOT value with a given value of *g*. The performance measures δ_p , δ_s and δ_d of the new coefficients are then calculated. The set that yields the minimum total terms for implementing total SOPOT coefficients while satisfying the given specifications and wordlength constraint *g* is declared as the optimum solution. Since this is a random search algorithm, the longer the searching time, the higher the chance of finding the optimal solution. To implement this multiplier-less FDDF using MB, consider its implementation in Fig. 2. Here, each sub-filter is implemented in their transposed form, where the input signal $x(n)$ is multiplied with a large number of constant coefficients in SOPOT form. The redundant additions in these SOPOT products can be reduced using a multiplier-block, greatly reducing the arithmetic complexity. The other FIR filters are implemented using a similar approach. We now present a design example.

V. Design Example

A programmable decimator using the multistage architecture with specifications has been designed as shown in Fig. 7. Note that the design result of the CIC filter is not mentioned in this section because its design and implementation is well known [1]-[3]. The target specifications of proposed programmable decimator are $\delta_{p-\max} = 0.00116$ (0.01-dB in passband deviation), $\delta_{s-\max} = 0.0001$ (80-dB in stopband attenuation) and $\delta_{d-\max} = 0.003$. The design results are summarized in Table 1. The frequency responses of the HBF, LPF#1, LPF#2 and LPF#3 are shown in Fig. 3 and the frequency responses and the group delays of the FDDF with $d = \{-0.5, -0.4, -0.3, -0.2, -0.1, 0\}$ are shown in Fig. 4. Also, the frequency responses of cascading the HBF and the FDDF with $M_I = \{1,2\}$ are shown in Fig. 5. Their worst-case passband deviation and stopband attenuation are 0.00997-dB and 80.37-dB respectively. The overall frequency responses of the proposed programmable decimator with $M_I = \{1,2\}$ are shown in Fig. 6. The overall worst-case passband deviation, stopband attenuation and group delay peak ripple error are 0.00955-dB, 81.65-dB and 0.00192 respectively. It should be noted that the total number of adders required for implementing all SOPOT coefficients before and after using MB are 249 and 110 adders, respectively, which is about 44% of the original hardware resources for all SOPOT coefficients.

VI. Conclusion

The programmable multistage decimator has been proposed. Its output is fed directly to the Farrow structure for sample rate conversion by eliminating the need for another *L*-band filter for upsampling. Furthermore, it was found out that the programmable FIR filter can be replaced by a half-band filter placed immediately after the Farrow structure. This significantly reduces the implementation complexity of the proposed software radio receiver because this half-band filter, which consists of fixed filter coefficients, can be implemented efficiently using SOPOT coefficients. As the coefficients of the multistage decimators and the subfilters in the Farrow structure are also fixed, they can also be implemented efficiently using the SOPOT coefficients. As a result, apart from the limited number of multipliers required in the Farrow structure, the entire digital IF can be implemented without any multiplication. Design example has been given to demonstrate the effectiveness and feasibility of the proposed approach.

References

[1] S. K. Mitra, *Digital Signal Processing: A Computer-Based Approach*, Singapore, McGraw-Hill, 1998.

[2] H. J. Oh, S. Kim, G. Choi and Y. H. Lee, "On the use of interpolated second-order polynomials for efficient filter design in programmable downconversion," *IEEE J. Select. Areas Commun.*, April 1999, pp. 551-560.

[3] A. Y. Kwentus, Z. Jiang and A. N. Willson, "Application of filter sharpening to cascaded integrator-comb decimation filters," *IEEE Trans. Signal Processing*, vol. 45, pp. 457-467, Feb 1997.

[4] K. Rajamani, Y. S. Lai and C. W. Farrow, "An efficient algorithm for sample rate conversion from CD to DAT," *IEEE Signal Processing Lett.*, vol. 7, no. 10, pp. 288-290, Oct 2000.

[5] C. W. Farrow, "A continuously variable digital delay element," *IEEE Int'l. Conf. Circuits and Sys. 1988*, pp. 2641-2645.

[6] C. K. S. Pun, Y. C. Wu, S. C. Chan and K. L. Ho, "An efficient design of fractional-delay digital FIR filter using Farrow structure," *Proceedings of the 11th IEEE Signal Processing Workshop on Statistical Signal Processing*, pp. 595-598, 2001.

[7] Y. C. Lim and S. R. Parker, "FIR filter design over a discrete power-of-two coefficient space," *IEEE Trans. ASSP-31*, pp. 583-591, April 1983.

[8] A. G. Dempster and M. D. MacLeod, " Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Trans. Circuits Syst. II*, pp. 569-577, Sept. 1995.

[9] C. K. S. Pun, S. C. Chan, K. S. Yeung and K. L. Ho, "On the design and implementation of FIR and IIR digital filters with variable frequency characteristics," *Submitted to IEEE ISCAS'2002*.

[10] T. Hentschel and G. Fettweis, "Sample rate conversion for software radio," *IEEE Commun. Mag.*, pp. 142-150, Aug. 2000.

[11] C. Y. Fung and S. C. Chan, "A multistage filterbank-based channelizer for software radio base stations," *Submitted to IEEE ISCAS'2002*.

[12] T. A. Ramstad, "Digital methods for conversion between arbitrary sampling frequencies," *IEEE Trans. ASSP*, vol. 32, no. 3, 1984.

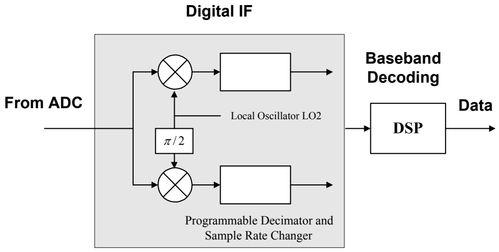
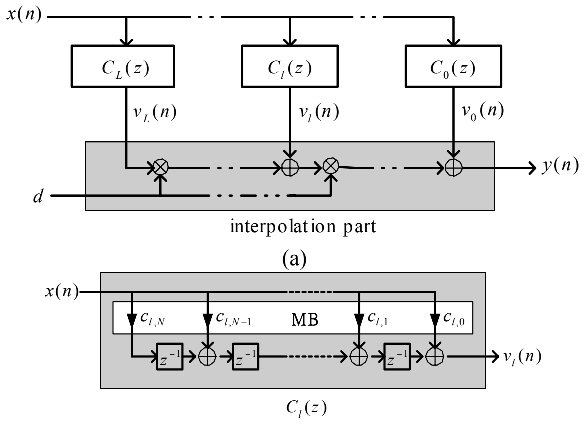


Fig. 1. Digital IF for software radio receiver.



(a)

(b)

Fig. 2. Implementation of the fractional-delay digital filter (FDDF) (a) Farrow structure (b) transposed form of sub-filters.

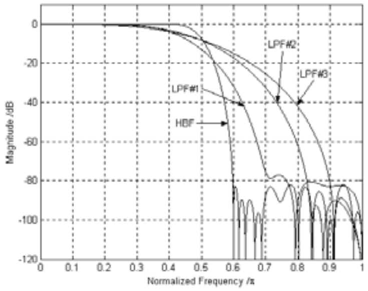
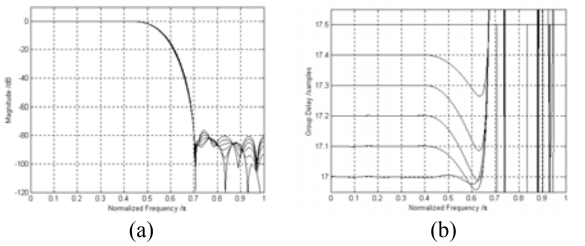


Fig. 3. Frequency responses of HBF, LPF#1, LPF#2 and LPF#3.



(a)

(b)

Fig. 4. (a) Frequency responses and (b) Group delays of the FDDF with $d = \{-0.5, -0.4, -0.3, -0.2, -0.1, 0\}$.

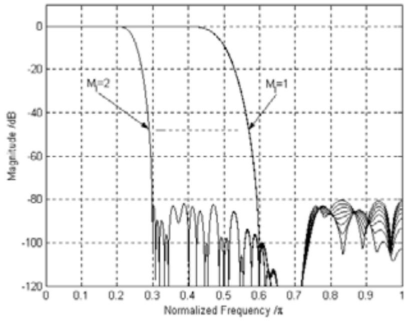


Fig. 5. Frequency responses of cascading the HBF and the FDDF with $M_I = \{1,2\}$.

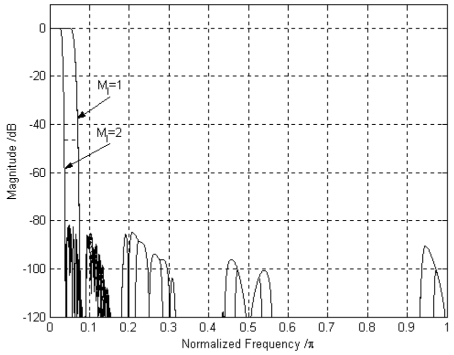


Fig. 6. Overall frequency responses of the proposed programmable decimator $M_I = \{1,2\}$ without CIC filter.

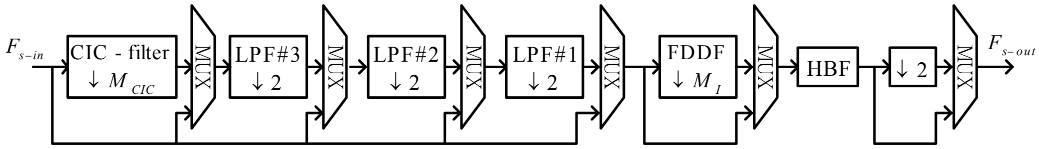


Fig. 7. Proposed programmable decimator structure.

Filter Symbol	HBF	FDDF	LPF#1	LPF#2	LPF#3
Passband edge ω_{pi}	0.4	0.4	0.2	0.1	0.05
Stopband edge ω_{si}	0.6	0.7	0.7	0.85	0.925
Filter order N	47	35	17	11	7
Interpolation order L	N/A	3	N/A	N/A	N/A
Wordlength (bits)	16	16	16	16	16
Passband deviation (in dB)	0.0023	0.00898	0.00248	0.00117	0.00113
Stopband attenuation (in dB)	81.95	76.48	77	90.13	88.68
Group delay peak ripple error	N/A	0.00192	N/A	N/A	N/A
Average SOPOT terms per coeff.	3.29	3.29	3.67	3.67	4.5
Adders required for SOPOT before MB	52	145	22	16	14
Adders required for SOPOT after MB	24	54	14	9	9

Table 1. Summarized results of each anti-aliasing filter.