Decimation by Non-Integer Factor using CIC Filter and Linear Interpolation

Djordje Babic[1†], Jussi Vesma[2], and Markku Renfors[1]

[1]Tampere University of Technology
Telecommunications Laboratory
P.O. Box 553, FIN-33101 Tampere
FINLAND
[†]Tel. +358-3-365 3910, Fax: +358-3-365 3808
[†]E-mail: babic@cs.tut.fi

[2]NOKIA GROUP
Nokia Research Center
P.O. Box 407 FIN-00045
FINLAND

ABSTRACT

Recently we have developed an efficient flexible multirate signal processing structure with high oversampling ratio and adjustable fractional or irrational sampling rate conversion factor. One application area is a multistandard communication receiver which should be adjustable for different symbol rates utilised in different systems. The proposed decimation filter consists of parallel CIC (cascaded integrator-comb) filters followed by a linear interpolation filter. The idea in this paper is to use two parallel CIC filters to calculate the two needed sample values for linear interpolation. In this paper we give a modification of the proposed structure and its control logic that enables better image and aliasing attenuation. The modification is based on the observation of the dependence of behaviour of the control logic on the fractional part of the sampling rate conversion factor.

1. INTRODUCTION

In multistandard receivers, the hardware should be configurable or programmable for the reception of different types of signals having different symbol rates. After the AD conversion, utilizing commonly the delta-sigma AD-conversion principle and high oversampling ratio, the sampling rate is reduced to be a low integer multiple of the symbol rate. In this decimation, the desired channel is preserved and other channels and noise are attenuated. The problem is that the needed decimation factor can be a difficult fractional number or even an irrational number and, for instance, FIR filters used for integer or fractional decimation cannot be efficiently utilized. Another problem is that there can be disturbing channels that are much stronger (e.g. 80-100 dB) than the desired channel. Therefore, the frequency bands which cause aliasing in decimation should have good attenuation. In addition to these requirements, the overall implementation should be simple because this decimation filter is used in the digital front-end of mobile receivers where the sampling rate is high [1], [2]. Based on these requirements (low complexity and possible irrational decimation factor), in [2] we have introduced a decimation filter structure which consists of two parallel CIC (cascaded integrator-comb) filters followed by linear interpolation. As it was shown this structure is easy to implement because the CIC filter does not need any multiplications and the linear interpolation requires only one multiplication. This structure has good anti-aliasing and anti-imaging properties.

In the general case, the decimation factor is a very difficult non-integer, thus the overall decimation factor is expressed as

$$R = \frac{F_{in}}{F_{out}} = R_{int} + \varepsilon, \qquad (1)$$

where $F_{in} = 1/T_{in}$ and $F_{out} = 1/T_{out}$ are the input and output sampling frequencies, whereas R_{int} is the integer part and ε is the decimal part of the overall decimation ratio. In [2] we have restricted discussion only for $\varepsilon \in [0,1)$. However, it was shown that sometimes it is better to use negative ε in order to increase aliasing band attenuation level. Therefore, in this paper we introduce modifications of the structure and control logic proposed in [2], in order to use the system for $\varepsilon \in (-1,0]$ as well. In that way characteristics of the proposed structure are improved, especially the worst case aliasing attenuation level.

2. BUILDING UNITS

Cascaded integrator-comb (CIC) filters are commonly used for decimation and interpolation by integer ratio providing efficient anti-image and anti-alias filtering [3]. These filters have a simple regular structure without multipliers. CIC decimation filter (see [3]) consists of N cascaded digital integrator stages operating at high input data rate F_{in}, followed by N cascaded comb or differentiator stages operating at low sampling rate F_{in}/R. Its frequency response is given by

$$H_{CIC}\left(e^{j\omega}\right) = e^{-j\omega N(R-1)/2}\left(\frac{\sin(\omega R/2)}{R\sin(\omega/2)}\right)^{N}, \qquad (2)$$

where $\omega = 2\pi f/F_{in}$ is the normalized input frequency.

When the decimation factor is an irrational number, the filters intended for integer or fractional decimation can not be directly used. One solution is to use polynomial-based interpolation filters. Among them, linear interpolation filter has a simple implementation structure, only one multiplication is needed [4]. Because interpolation is basically a reconstruction problem, polynomial-based interpolation can be analysed using the hybrid analog/digital model shown in Fig. 1, [4]. In this model, the interpolated output samples $y(l)$ are obtained by sampling the reconstructed signal $y_a(t)$ at the time instants $t = (n_l + \mu_l)\, T_{in}$. Here n_l is any integer, $\mu_l \in [0,1)$ is the adjustable fractional interval, and T_{in} is the sampling interval of the input signal $x(n)$.

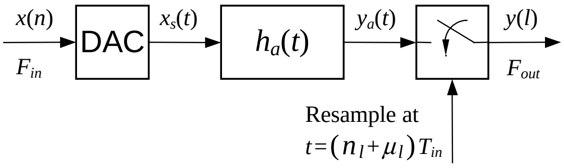


Fig. 1. The hybrid analog/digital model for the linear interpolation filter.

For linear interpolation, the impulse response of the reconstruction filter $h_a(t)$ is a triangular function, and thus, its frequency response is given by

$$H_a(f) = \left(\frac{\sin(\pi f / F_{in})}{\pi f / F_{in}} \right)^2 . \tag{3}$$

The digital implementation of the linear interpolation, which needs only one multiplication, can be based on the following equation:

$$y(l) = x(n_l) + [x(n_l + 1) - x(n_l)]\mu_l . \tag{4}$$

3. PROPOSED STRUCTURE FOR NON-INTEGER DECIMATION IN THE CASE OF $\varepsilon \in (-1,0]$

Figure 2 illustrates the proposed structure for the decimation filter. The input signal $x(n)$ is divided into polyphase components $x_k(m)$ for $k = 0, 1, \cdots, R_{int} - 1$ by using delay line and parallel CIC filters. Therefore, the sampling rate at the output of the CIC filters is F_{in} / R_{int}. The final decimation by $1+\varepsilon / R_{int}$ is done using linear interpolation between some of the two signal pairs $x_k(m)$ and $x_{k\oplus 1}(m)$, where $\oplus$ denotes the modulo R_{int} summation. The linear interpolation block in Fig. 2 is shifted by one branch according to some condition (to be discussed later on). Because of the modulus R_{int} summation mentioned above, the next signal pair for linear interpolation after $x_0(m)$ and $x_1(m)$ is $x_{R_{int}-1}(m)$ and $x_0(m)$. The fractional interval μ_l is recalculated for each output sample $y(l)$ for $l = 0, 1, 2, \cdots$. The time interval between samples $x_k(m)$ and $x_{k\oplus 1}(m)$ equals to T_{in} and, thus, the linear interpolation is done at the high input sampling frequency F_{in}. This means better image attenuation. The CIC filters attenuate the disturbing channels and noise which would cause

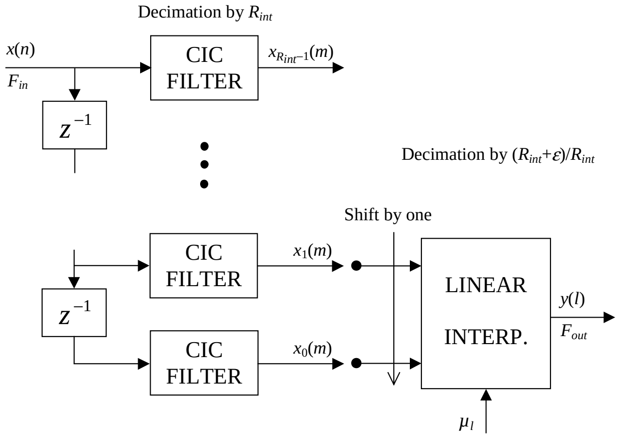


Fig. 2. Model of proposed decimation filter.

aliasing in linear interpolation. In other words, the CIC filters and linear interpolation take care of anti-aliasing and anti-imaging property, respectively. It should be pointed out that the filter structure of Fig. 2 is not the final implementation form. All the CIC filter branches are not needed and some of the blocks can be shared to make the final implementation easier, as will be discussed in Section 3.

As an example, Fig. 3 shows the input and output signals as well as some of the polyphase signals of the decimation filters for the decimation factor of $R = 3.9$. These polyphase signals $x_0(m)$ and $x_1(m)$ shown in Fig. 3(b) are obtained from $x(n)$ using a delay and two parallel CIC filters as shown in Fig. 2. Linear interpolation is then applied between these two signals to obtain the output samples $y(l) = y(lT_{out})$ for l–1, l, l+1 and l+2. After sample $y(l+1)$, the next output sample $y(l+2)$ falls outside the interval $x_0(m)$ and $x_1(m)$. When this occurs, the linear interpolation is shifted by one interval (as indicated by an arrow in Fig. 2) and the interpolation is done between signals $x_{R_{int}-1}(m)$ and $x_0(m)$.

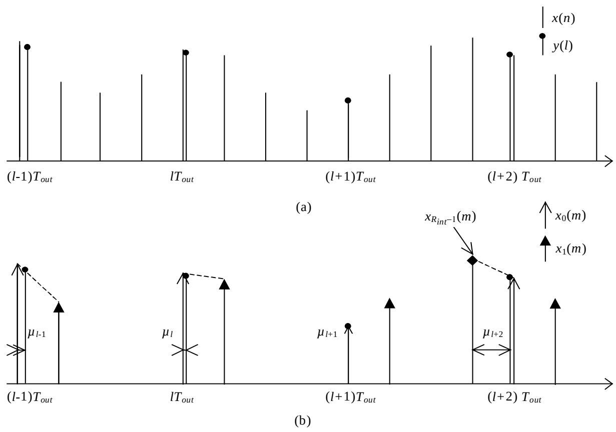


Fig. 3. (a) The input and output samples of the proposed decimation filter for $R = 3.9$. (b) The output samples of the two parallel CIC filter branches $x_0(m)$ and $x_1(m)$.

3.1. The frequency response of the overall system

The overall frequency response of the decimation filter structure in Fig. 2 is a product of the frequency responses of the CIC filter and linear interpolation filter. Note that the former response is periodical whereas the latter is not. The frequency response of the parallel CIC filter stage is simply the same as the response of one CIC filter given by Eq. (2), where, however, R has to be replaced by R_{int}. Since the linear interpolation is done at the higher input rate F_{in}, its frequency response is given by Eq. (3). Consequently, the overall zero-phase frequency response of the proposed decimation filter, relative to the input sampling frequency, is given by

$$H_T(\omega) = H_{CIC}(\omega) H_a\left(\omega F_{in} / 2\pi\right) = \left(\frac{\sin\left(\frac{\omega R_{int}}{2}\right)}{R_{int} \sin\left(\frac{\omega}{2}\right)} \right)^N \left(\frac{\sin\left(\frac{\omega}{2}\right)}{\frac{\omega}{2}} \right)^2 . \quad (5)$$

where $\omega = 2\pi f / F_{in} = 2\pi f / (RF_{out})$.

4. IMPLEMENTATIONS

The implementation structure in the case of negative ε, given in Fig. 4, is exactly the same as in the case of positive ε that is explained in [2], only the control logic is changed. However, here we shortly describe the implementation structure for the completeness of the paper. In the general case the number of the parallel CIC filters B, that is a number of comb filter branches, is given with B=2+N, where N is the order of the CIC filter. Two branches are used for calculating the output samples and the remaining N branches are used for initializing the state-variables of the branches needed later. However, the number of required comb branches can be reduced to the minimum. It is possible to use only B=3 branches in the comb section if following condition holds

$$|\varepsilon| \leq 1/N . \quad (6)$$

The integrator stage is shared among the branches. The commutators COM1 and COM2 are used to select the correct input branch for the B comb sections and for linear interpolation, respectively.

As it was mentioned the control logic algorithm is different in the case of negative ε. Using analysis in time as in Fig. 3, one can notice that operations for ε'>0 and ε<0 are complementary, where ε'=1+ε. That means, there is shifting performed for the case of ε<0 whenever there is no shifting in the case of ε'>0. Using this observation the structure of the algorithm remains the same as in [2], however some changes are required. In Fig. 5(a) the control logic in the case of negative ε is given. The first step in this algorithm is the initial set up of the index value l as well as the fractional interval $\mu_0 = 0$. The next step is the interpolation which is expressed by

$$y(l) = I(\mu_l, u_0(m), u_1(m)), \quad (7)$$

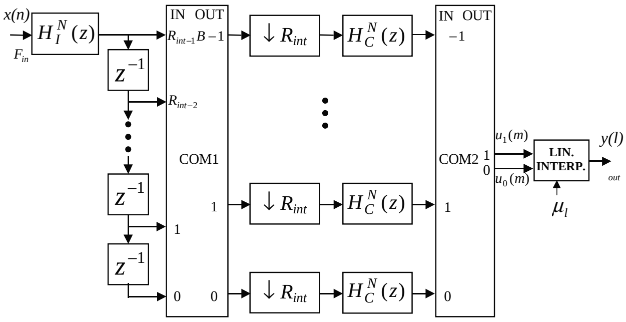


Fig. 4. Implementation structure for the proposed decimation filter.

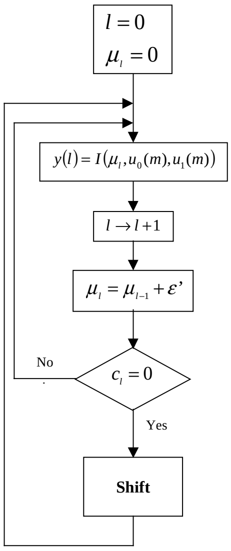
where $I(\ )$denotes the linear interpolation between the samples $u_0(m)$ and $u_1(m)$ with the fractional interval of μ_l. After interpolation, l is incremented by one and the fractional interval can be computed by

$$\mu_l = \mu_{l-1} \oplus \varepsilon', \quad (8)$$

with the initial condition $\mu_0 = 0$, note that here we use complementary value ε' instead of ε and this is a main difference in the algorithm. In Eq. (8) the modulo summation indicates that only the decimal part of the result is used. According to Eq. (8), the calculation of μ_l can be implemented by using an adder with fixed point arithmetic. The shifting in the interpolation has to be performed whenever there is no overflow while calculating μ_l. Therefore, the overflow bit c_l of the adder can be used as a shifting condition. The shift block in Fig. 5(a) means that the interpolation is shifted by one branch (see Fig. 2). This shifting operation is implemented using the commutators COM1 and COM2 as it is shown in Fig. 5(b). The commutator COM1 has R_{int} inputs and B outputs. The commutator COM2 has B inputs and two outputs. In order to describe the function of the commutators we use variables for the outputs of the commutators. There are B variables for the outputs of COM1 denoted by OUT_i^1 for i=0,1… B–1 and two variables for COM2 denoted by OUT_i^2 for i=1 and 2. The values of these variables determine what input sample is connected to the i^{th} output. The switching algorithm for COM1 and COM2 is given in Fig. 5(b). When shifting occurs, only one output of COM1, numbered by p, should be switched to the another input. Hence, only the value of the variable OUT_p^1 is changed. In COM2, when shifting occurs, both output branches should be switched to the another input. This is done because the order of the interpolator inputs must be preserved.

5. EXAMPLES

The bands that cause aliasing to the desired band are positioned around frequencies that are multiples of F_{out}. However the zeros of CIC filters are at the points which are multiples of RF_{out}/R_{int}. The minimum aliasing attenuation occurs at the edge of the first aliased band. Figure 6 shows the minimum attenuation of the aliasing



(*initial set up*)

$OUT_0^1 = 0$, $OUT_1^1 = 1$;
for *i*=2 to *B*-1
$OUT_i^1 = R_{int} - B + i$;
end
$OUT_0^2 = 0$, $OUT_1^2 = 1$;
p=1;

(*shifting*)

if $c_l = 0$ then
$OUT_p^1 = OUT_p^1 \oplus_{R_{int}} (-B)$;
$p = p \oplus_B (-1)$;
$OUT_0^2 = OUT_0^2 \oplus_B (-1)$;
$OUT_1^2 = OUT_1^2 \oplus_B (-1)$;
end.

(a) (b)

Fig. 5. (a) The state flow diagram of the control logic. (b) Algorithm for switching of COM1 and COM2. Here $\oplus_i$ denotes modulo *i* summation.

bands in a range of the fractional decimation factor ranging from 32 to 34 in the case of the third order CIC filter. As it can be seen the minimum attenuation of the aliasing bands depends on ε. As ε increases the minimum attenuation reduces, but this is avoided if we use positive ε algorithm when ε<0.5 and negative ε algorithm when ε>0.5, as explained above.

6. CONCLUSIONS

Since the whole structure requires only one multiplier and since it offers good anti-aliasing and anti-imaging properties, the proposed decimation filter is considered as power-efficient, relatively simple, and flexible solution for non-integer factor decimation in the multistandard radio receivers. We have shown that the negative ε algorithm can be implemented to the proposed structure. It was also shown that the negative ε algorithm allows us to use reduced number of comb branches in the actual implementation, and that means reduction in power consumption of the overall structure. Further, better aliasing attenuation is achieved using proposed negative ε algorithm for certain range of ε and positive ε algorithm for other values of ε.

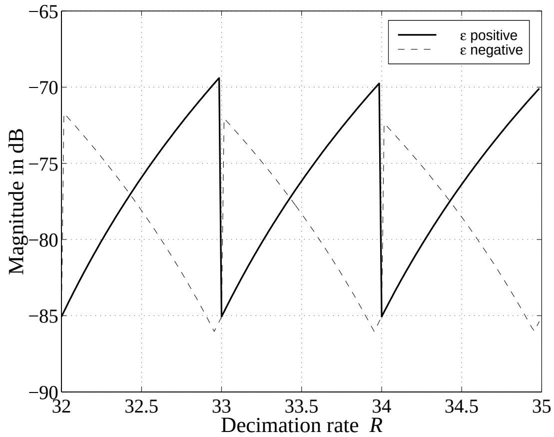


Fig. 6. The maximum value of the aliasing bands for positive and negative ε and the third order CIC filter.

ACKNOWLEDGMENT

This work has been supported by Graduate School TISE.

REFERENCES

[1] T. Hentschel and G. Fettweis, "Software radio receivers," Chapter 10 in *CDMA Techniques for Third Generation Mobile Systems*, edited by F. Swarts, P. van Rooyan, I. Oppermann, and M. P. Lötter, Kluwer Academic Publishers, 1999, pp. 255-283.

[2] D. Babic, J. Vesma, M. Renfors, "Decimation by irrational factor using CIC filter and linear interpolation," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing, ICASSP2001*, Salt Lake City, USA, 2001, in press.

[3] E. B. Hogenauer, "An economical class of digital filters for decimation and interpolation," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-29, pp. 155-162, April 1981.

[4] J. Vesma, *Optimization and Applications of Polynomial-Based Interpolation Filters*; Doctoral Thesis, Tampere University of Technology, Publications 254, 1999.