An Economical Class of Digital Filters for Decimation and Interpolation

EUGENE B. HOGENAUER, MEMBER, IEEE

Abstract—A class of digital linear phase finite impulse response (FIR) filters for decimation (sampling rate decrease) and interpolation (sampling rate increase) are presented. They require no multipliers and use limited storage making them an economical alternative to conventional implementations for certain applications.

A digital filter in this class consists of cascaded ideal integrator stages operating at a high sampling rate and an equal number of comb stages operating at a low sampling rate. Together, a single integrator-comb pair produces a uniform FIR. The number of cascaded integrator-comb pairs is chosen to meet design requirements for aliasing or imaging error.

Design procedures and examples are given for both decimation and interpolation filters with the emphasis on frequency response and register width.

I. Introduction

In recent literature, Crochiere and Rabiner [1]-[3] have presented a general theory for FIR multistage decimators and interpolators with emphasis on optimal designs in terms of minimizing the number of multiplications per second or the required amount of storage. Goodman and Carey [4] have taken the approach that a careful choice of filter coefficients for half-band decimators and interpolators can lead to efficient hardware designs.

In the field of efficient digital filters, Peled and Liu [5] have introduced the "coefficient slicing" approach to filter design. For these filters, multipliers are replaced with adders and ROM look-up tables. This approach can be applied profitably to decimator and interpolator designs.

The essential function of a decimation or interpolation filter is to decrease or increase the sampling rate and to keep the passband aliasing or imaging error within prescribed bounds. In this paper, a class of linear phase FIR filters for decimation and interpolation that fulfill this basic requirement are introduced. The filters require no multipliers and use limited storage thereby leading to more economical hardware implementations. They are designated cascaded integrator-comb (CIC) filters because their structure consists of an integrator section operating at the high sampling rate and a comb section operating at the low sampling rate.

Using CIC filters, the amount of passband aliasing or imaging error can be brought within prescribed bounds by increasing the number of stages in the filter. However, the width of the passband and the frequency characteristics outside the passband are severely limited. For critical applications these limitations can be overcome by using CIC filters to make the transition between high and low sampling rates, and to use conventional filters at the low sampling rate to "shape" or "clean-up" the frequency response. In this manner, CIC filters are used at high sampling rates where economy is critical, and conventional filters are used at low sampling rates where the number of multiplies per second is low.

Like CIC filters, some of the filters described in [4] do not require multipliers; however, these filters are restricted to a rate change factor of two, and have limited attenuation in the aliasing/imaging bands.

The next section describes CIC filters in terms of their functional building blocks, relating their z-transforms to the z-transform of the composite filter. Section III discusses the frequency response of CIC filters giving an approximation that is usable for a wide range of design problems. Tables are provided for determining filter parameters as a function of the desired bandwidth and aliasing/imaging error.

In Section IV, CIC decimation filters are described with particular attention given to the effects of truncation and rounding on the filter's error statistics. Design equations are given and are applied to a specific design problem. In Section V a similar treatment is given for CIC interpolation filters with the major consideration given to register growth and its relation to the filter design.

II. CIC Filter Description

Fig. 1 shows the basic structure of the CIC decimation filter. An analogous structure for the CIC interpolation filter is presented in Fig. 2.

The integrator section of CIC filters consists of N ideal digital integrator stages operating at the high sampling rate, f_s. Each stage is implemented as a one-pole filter with a unity feedback coefficient. The system function for a single integrator is

$$H_I(z) = \frac{1}{1 - z^{-1}}. \tag{1}$$

The comb section operates at the low sampling rate f_s/R where R is the integer rate change factor. This section consists of N comb stages with a differential delay of M samples per stage. The differential delay is a filter design parameter used to control the filter's frequency response. In practice, the differential delay is usually held to $M = 1$ or 2. The system function for a single comb stage referenced to the high sampling rate is

$$H_C(z) = 1 - z^{-RM}. \tag{2}$$

Manuscript received March 17, 1980; revised September 22, 1980.
The author is with ESL, Inc. Sunnyvale, CA 94086.

0096-3518/81/0400-0155$00.75

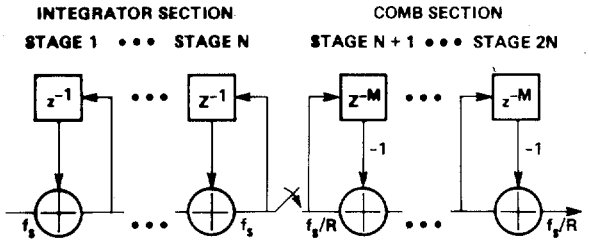

Fig. 1. CIC decimation filter.

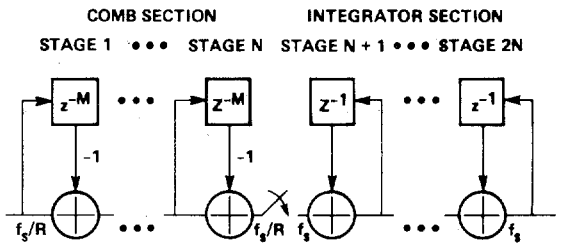

Fig. 2. CIC interpolation filter.

There is a rate change switch between the two filter sections. For decimation, the switch subsamples the output of the last integrator stage, reducing the sampling rate from f_s to f_s/R; and for interpolation, the switch causes a rate increase by a factor of R by inserting $R-1$ zero valued samples between consecutive samples of the comb section output.

It follows from (1) and (2) that the system function for the composite CIC filter referenced to the high sampling rate, f_s, is

$$H(z) = H_I^N(z)H_C^N(z) = \frac{(1-z^{-RM})^N}{(1-z^{-1})^N} = \left[\sum_{k=0}^{RM-1} z^{-k}\right]^N. \quad (3)$$

It is implicit from the last form of the system function that the CIC filter is functionally equivalent to a cascade of N uniform FIR filter stages. A conventional implementation consists of a cascade of N stages each requiring RM storage registers and one accumulator. Taking advantage of the rate change factor, one of the N stages can be simplified to use only M storage registers.

It must be stressed that each integrator has a unity feedback coefficient; for CIC decimators this results in register overflow in all integrator stages. This is of no consequence if the following two conditions are met. 1) The filter is implemented with two's complement arithmetic or other number system which allows "wrap-around" between the most positive and most negative numbers. 2) The range of the number system is equal to or exceeds the maximum magnitude expected at the output of the composite filter. For CIC interpolators, the data are preconditioned by the comb section so that overflow will not occur in the integrator stages.

The economics of CIC filters derive from the following sources: 1) no multipliers are required; 2) no storage is required for filter coefficients; 3) intermediate storage is reduced by integrating at the high sampling rate and comb filtering at the low sampling rate, compared to the equivalent implementation using cascaded uniform FIR filters; 4) the structure of CIC filters is very "regular" consisting of two basic building blocks; 5) little external control or complicated local timing is required; 6) the same filter design can easily be used for a wide range of rate change factors, R, with the addition of a scaling circuit and minimal changes to the filter timing.

Some problems encountered with CIC filters include the following. 1) Register widths can become large for large rate change factors, R. 2) The frequency response is fully determined by only three integer parameters (R, M, and N), resulting in a limited range of filter characteristics.

The application for CIC filters seems to be in areas where high sampling rates make multipliers an uneconomical choice and areas where large rate change factors would require large amounts of coefficient storage or fast impulse response generation. For example, a system has been implemented consisting of 32 digital interpolators operating at about $f_s = 5$ MHz. Each interpolator is built on a single PC board using the CIC technique. The filters have a variable rate change factor of up to $R = 512$ implemented with $N = 4$ stages and a differential delay of $M = 2$, resulting in a stopband attenuation of 53 dB. For the rate change factor of 512, the filter consists of 4093 zeros. Although the number of zeros is large, the implementation is very economical, consisting of 7 adders and 11 storage registers with no coefficient storage or multipliers.

III. Frequency Characteristics

CIC filters have a low-pass frequency characteristic. The frequency response is given by (3) evaluated at

$$z = e^{j(2\pi f/R)} \quad (4)$$

where f is the frequency relative to the low sampling rate f_s/R. As part of the filter design process, R, M, and N are chosen to provide acceptable passband characteristics over the frequency range from zero to a predetermined cutoff frequency f_c expressed relative to the low sampling rate. The power response is

$$P(f) = \left[\frac{\sin \pi Mf}{\sin \frac{\pi f}{R}}\right]^{2N}. \quad (5)$$

For large rate change factors R, the power response can be approximated over a limited frequency range by

$$\hat{P}(f) = \left[RM\frac{\sin \pi Mf}{\pi Mf}\right]^{2N} \quad \text{for } 0 \leqslant f < \frac{1}{M}. \quad (6)$$

This approximation can be used for many practical design problems. For example, the error between P and $\hat{P}$ is less than 1 dB for $RM \geqslant 10$, $1 \leqslant N \leqslant 7$ and $0 \leqslant f \leqslant 255/(256M)$.

For the power response of (5) and (6), nulls exist at multiples of $f = 1/M$. Thus, the differential delay M can be used as a design parameter to control the placement of nulls. For CIC decimation filters, the region around every Mth null is folded into the passband causing aliasing errors; for CIC interpolation filters, imaging occurs in the regions around these same nulls. Specifically, these aliasing/imaging bands are

$$(i - f_c) \leqslant f \leqslant (i + f_c) \quad (7)$$

for $f \leqslant \frac{1}{2}$ and $i = 1, 2, \cdots, \lfloor R/2 \rfloor$ where $\lfloor x \rfloor$ is the largest integer not greater than x.

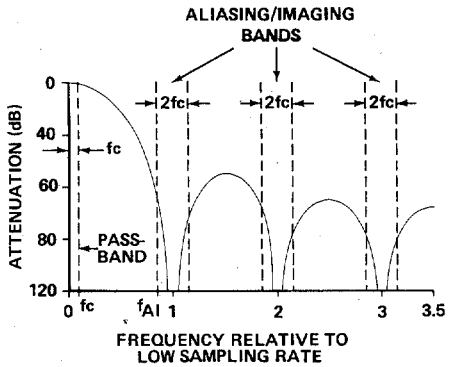

Fig. 3. Example frequency response for $N = 4$, $M = 1$, $R = 7$, and $f_c = \frac{1}{8}$.

TABLE I
PASSBAND ATTENUATION FOR LARGE RATE CHANGE FACTORS

Relative Bandwidth-Differential Delay Product (Mf_c)	Passband Attenuation at f_c (dB) As a Function of Number of Stages (N)					
	1	2	3	4	5	6
1/128	0.00	0.00	0.00	0.00	0.00	0.01
1/64	0.00	0.01	0.01	0.01	0.02	0.02
1/32	0.01	0.03	0.04	0.06	0.07	0.08
1/16	0.06	0.11	0.17	0.22	0.28	0.34
1/8	0.22	0.45	0.67	0.90	1.12	1.35
1/4	0.91	1.82	2.74	3.65	4.56	5.47

An example power response is given in Fig. 3 for an $N = 4$ stage CIC filter with a differential delay of $M = 1$ and a rate change factor of $R = 7$. The passband cutoff is at $f_c = \frac{1}{8}$ with the aliasing/imaging bands centered around the nulls at frequencies of 1, 2, and 3 relative to the low sampling rate.

For practical design problems, the aliasing/imaging errors can be characterized by the maximum error over all aliasing/imaging bands. For a large class of filter design problems where $f_c \leqslant 1/2M$, this maximum occurs at the lower edge of the first aliasing/imaging band at

$$f_{AI} = 1 - f_c. \tag{8}$$

Tables I and II are presented as an aid in determining the tradeoffs between bandwidth, passband attenuation, and aliasing/imaging error. It is assumed that the rate change factor is large, so the power response approximation of (6) can be used. In these tables attenuations are calculated relative to the maximum filter response at $f = 0$.

The passband attenuations given in Table I are constant for a given relative bandwidth-differential delay product (Mf_c); however, this is not the case for the aliasing/imaging attenuation given in Table II. Here, two values of differential delay, $M = 1$ and 2 are tabulated; differential delays greater than these seem to be of less value.

IV. CIC Decimation Filter Design

A. Design Overview

This section presents design considerations for CIC decimation filters. The most significant bit (MSB) of these filters is determined as a function of the overall register growth. This is followed by a demonstration that truncation or rounding may be used at each stage of filtering, the retained number of bits decreasing monotonically from stage to stage. An explanation is given which relates the truncation or rounding in intermediate stages to the total error in the output data stream. This explanation is then turned around so that the filter designer can determine the amount of truncation or rounding to apply at each stage, without violating design constraints.

It is assumed that the desired frequency characteristics have already been determined using information in Section III, resulting in choices for the rate change factor R, differential delay M, and number of stages N. It is also assumed throughout this section and Section V that two's complement arithmetic is being used.

TABLE II
ALIASING/IMAGING ATTENUATION FOR LARGE RATE CHANGE FACTORS

Differential Delay (M)	Relative Bandwidth (f_c)	Aliasing/Imaging Attenuation at f_{IA} (dB) As a Function of Number of Stages (N)					
		1	2	3	4	5	6
1	1/128	42.1	84.2	126.2	168.3	210.4	252.5
1	1/64	36.0	72.0	108.0	144.0	180.0	215.9
1	1/32	29.8	59.7	89.5	119.4	149.2	179.0
1	1/16	23.6	47.2	70.7	94.3	117.9	141.5
1	1/8	17.1	34.3	51.4	68.5	85.6	102.8
1	1/4	10.5	20.9	31.4	41.8	52.3	62.7
2	1/256	48.1	96.3	144.4	192.5	240.7	288.8
2	1/128	42.1	84.2	126.2	168.3	210.4	252.5
2	1/64	36.0	72.0	108.0	144.0	180.0	216.0
2	1/32	29.9	59.8	89.6	119.5	149.4	179.3
2	1/16	23.7	47.5	71.2	95.0	118.7	142.5
2	1/8	17.8	35.6	53.4	71.3	89.1	106.9

B. Register Growth

The system function from the jth stage up to and including the last stage can be expressed as a fully expanded polynomial in z^{-1}. The resulting function is

$$H_j(z) = \begin{cases} H_I^{N-j+1} H_C^N = \displaystyle\sum_{k=0}^{(RM-1)N+j-1} h_j(k) z^{-k}, \\ \qquad\qquad j = 1, 2, \cdots, N \\ H_C^{j-N} = \displaystyle\sum_{k=0}^{2N+1-j} h_j(k) z^{-kRM}, \\ \qquad\qquad j = N+1, \cdots, 2N \end{cases} \tag{9a}$$

where

$$h_j(k) = \begin{cases} \displaystyle\sum_{l=0}^{\lfloor k/RM \rfloor} (-1)^l \binom{N}{l} \binom{N-j+k-RMl}{k-RMl}, \\ \qquad\qquad j = 1, 2, \cdots, N \\ (-1)^k \binom{2N+1-j}{k}, \qquad j = N+1, \cdots, 2N \end{cases} \tag{9b}$$

are the impulse response coefficients. This function is derived in Appendix I.

The maximum register growth is defined as the maximum output magnitude resulting from the worst possible input signal relative to the maximum input magnitude. This growth is used in the CIC filter design process to insure that no data are lost due to register overflow. Using this definition, the maximum register growth from the first stage up to and including the last stage is simply

$$G_{\max} = \sum_{k=0}^{(RM-1)N} |h_1(k)|. \tag{10a}$$

It is shown in Appendix II that this can be simplified to

$$G_{\max} = (RM)^N. \tag{10b}$$

If the number of bits in the input data stream is B_{in}, then the register growth can be used to calculate $B_{\max}$, the most significant bit at the filter output. That is,

$$B_{\max} = \lceil N \log_2 RM + B_{\text{in}} - 1 \rceil \tag{11}$$

where the least significant bit (LSB) of the input register is considered to be bit number zero and where $\lceil x \rceil$ is the smallest integer not less than x.

Not only is $B_{\max}$ the MSB at the filter output, but it is also the MSB for all stages of the filter. This can be shown by applying modulo arithmetic to the filter output function. For two's complement arithmetic, the modulo operation can be implemented by simply eliminating bit positions above $B_{\max}$.

Since the modulo operation is used at the filter output, the same modulo operation can be applied independently to each integrator and comb stage. This implies that $B_{\max}$ is an upper bound for each filter stage.

It is now shown that $B_{\max}$ is also a lower bound. Since the first N stages of the filter are integrators with unity feedback, it is apparent that the variance of the integrator outputs grow without bound for uncorrelated input data. As seen at the output register, $B_{\max}$ is the MSB for each integrator since this is a significant bit and is the highest order bit that can propogate into the output register. Since a propagation path must be provided through the comb section for this MSB, it can be concluded that $B_{\max}$ must be the MSB not only for the integrators, but also for the combs that follow.

C. Truncation and Rounding

$B_{\max}$ is large for many practical cases and can result in large register widths; however, truncation or rounding may be used at each filter stage reducing register widths significantly.

To calculate the total error at the filter output due to truncation or rounding, the mean and variance of the error at each error source is determined and then the corresponding statistics at the filter output due to the source alone is determined. The total mean and variance at the output is then determined as the sum of the statistics from these individual sources.

There are a total of $2N+1$ error sources: the first $2N$ sources are caused by truncation or rounding at the inputs to the $2N$ filter stages. The last error source is due to truncation or rounding going into the output register. The error sources are given indexes corresponding to the filter stage numbers shown in Fig. 1, with $2N+1$ identifying the error source going into the output register.

It is often assumed that rounding is always better than truncation, however, in the following paragraphs it is shown that except for the first and last error sources, the output error statistics are the same for both truncation and rounding. Furthermore, to keep the output error within bounds, most practical designs will make use of full precision arithmetic at the first error source. As a result, the only place where the designer need worry about truncation versus rounding is at the last error source going into the output register.

It is assumed that each error source produces white noise that is uncorrelated with the input and other error sources. Furthermore, the error at the jth source is assumed to have a uniform probability distribution with a width of

$$E_j = \begin{cases} 0, & \text{if no truncation nor rounding} \\ 2^{B_j}, & \text{otherwise} \end{cases} \tag{12}$$

where B_j is the number of LSB's discarded at the jth source. It can be shown that since the error has a uniform distribution, the mean of the error is

$$\mu_j = \begin{cases} \frac{1}{2} E_j, & \text{if truncation} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

and the variance of the error is

$$\sigma_j^2 = \tfrac{1}{12} E_j^2. \tag{14}$$

To determine the statistics at the output due to the jth error source, we use the system function from the jth stage up through the last comb as given by (9). The impulse response coefficients correspond to independent random processes that are summed together to produce one filter output. The error mean and variance corresponding to the kth coefficient are simply $\mu_j h_j(k)$ and $\sigma_j^2 h_j^2(k)$, respectively, and since the processes are independent over k, the total statistics at the jth stage are the sums of the statistics for each impulse response coefficient. That is, the total mean is

$$\mu_{T_j} = \mu_j D_j \tag{15a}$$

where

$$D_j = \begin{cases} \sum_k h_j(k), & j = 1, 2, \cdots, 2N \\ 1, & j = 2N+1 \end{cases} \tag{15b}$$

is designated the "mean error gain" for the jth error source. Similarly, the total variance is

$$\sigma_{T_j}^2 = \sigma_j^2 F_j^2 \tag{16a}$$

where

$$F_j^2 = \begin{cases} \sum_k h_j^2(k), & j = 1, 2, \cdots, 2N \\ 1, & j = 2N+1 \end{cases} \tag{16b}$$

is designated the "variance error gain" for the jth error source. The two error gains are used to relate the statistics at the error

source to those at the output and are useful in the design process because they are independent of the actual error.

It can be demonstrated that the mean error gain given by (15b) is zero for all but the first and last error sources and furthermore, the expression for the first error source can be simplified. This results in the form

$$D_j = \begin{cases} (RM)^N, & j = 1 \\ 0, & j = 2, 3, \cdots, 2N \\ 1, & j = 2N+1. \end{cases} \tag{17}$$

From (12) and (14) it is noted that the error variance is the same for either truncation or rounding and the total error mean given by (15a) and (17) is zero for all but the first and last error sources. As a result, the choice of truncation versus rounding does not affect the error statistics except for the first and last error sources.

The total mean and variance at the output due to truncation and/or rounding are

$$\mu_T = \sum_{j=1}^{2N+1} \mu_{T_j} = \mu_{T_1} + \mu_{T_{2N+1}} \tag{18}$$

and

$$\sigma_T^2 = \sum_{j=1}^{2N+1} \sigma_{T_j}^2. \tag{19}$$

Using the foregoing information relating error at the sources to error at the output, we can now work backwards to determine the number of bits to discard given appropriate error constraints. In this process, only the variance is used as a design parameter since it is affected by truncation and rounding at all error sources. On the other hand, the mean is affected by truncation and rounding only at the first and last error sources.

It is assumed that the number of bits retained in the output register is B_{out}, so the number of LSB's discarded is

$$B_{2N+1} = B_{\max} - B_{\text{out}} + 1. \tag{20}$$

The resulting error variance $\sigma_{T_{2N+1}}^2$ is defined by (16).

A legitimate design decision at this point is to make the variance from the first $2N$ error sources less than or equal to the variance for this last error source, and also to distribute the error about equally among these sources. This results in the following design equation for choosing the number of LSB's to discard at each stage:

$$B_j = \left\lfloor -\log_2 F_j + \log_2 \sigma_{T_{2N+1}} + \frac{1}{2}\log_2 \frac{6}{N} \right\rfloor \tag{21}$$

for $j = 1, 2, \cdots, 2N$. This equation is derived in Appendix III.

D. Design Example

We wish to design a decimation filter to reduce the sampling rate from 6 MHz to 240 kHz with a passband of 30 kHz. The aliasing attenuation must be better than 60 dB with a falloff in the passband of less than 3 dB. The number of bits in the input and output registers is $B_{\text{in}} = B_{\text{out}} = 16$.

We note that the rate change factor is $R = 25$ and the bandwidth relative to the low sampling rate is $f_c = \frac{1}{8}$. Referring to Tables I and II, we see that a filter with $N = 4$ stages and a differential delay of $M = 1$ results in an aliasing attenuation of 68.5 dB and a passband attenuation of 0.90 dB. To simplify the design, truncation is used at all stages of the filter. We calculate the MSB for the filter as $B_{\max} = 34$ resulting in $B_{2N+1} = 19$.

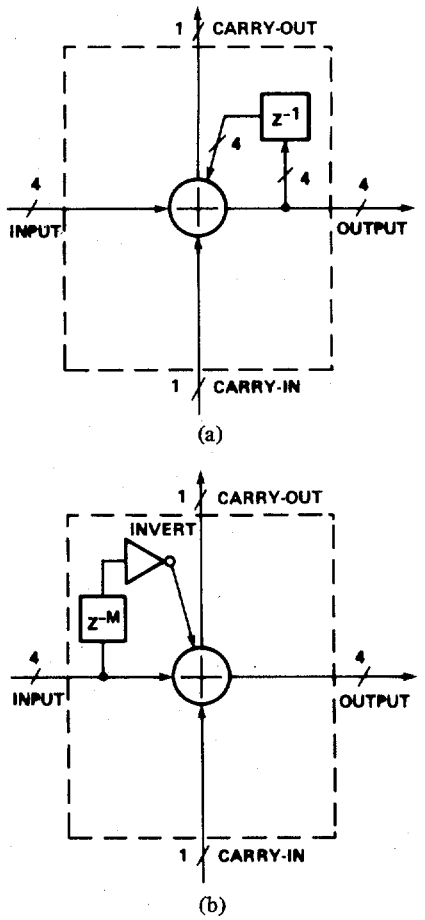

Fig. 4. CIC building blocks. (a) 4 bit integrator; (b) 4 bit comb.

Using (15a), (16), and (17) we calculate the error gain for each stage and using (21) we determine the number of LSB's discarded for the $2N$ filter stages to be 1, 6, 9, 13, 14, 15, 16, and 17, respectively. Using (18) and (19) we calculate the total mean and standard deviation assuming a binary point to the right of the LSB of the output register. The mean is $\mu_T/2^{19} = 1.245$ and the standard deviation of $\sigma_T/2^{19} = 0.373$.

The decimator is to be implemented in hardware using 4 bit parts. As a result, the register lengths in this example, except for the first integrator, can be truncated up to the nearest multiple of 4 bits resulting in LSB's to be discarded for the $2N$ stages of 0, 3, 7, 11, 11, 15, 15, and 15, respectively. The mean is now reduced to $\mu_T/2^{19} = 0.500$ and the standard deviation is slightly better at $\sigma_T/2^{19} = 0.301$.

The design uses two basic building blocks: a 4 bit integrator shown in Fig. 4(a) and a 4 bit comb shown in Fig. 4(b). These are combined in Fig. 5 to form the composite CIC decimator. Each building block has a 4 bit input, 4 bit output, carry-in and carry-out. A comb stage built from the 4 bit comb building blocks requires a subtraction on the feed-forward path. It

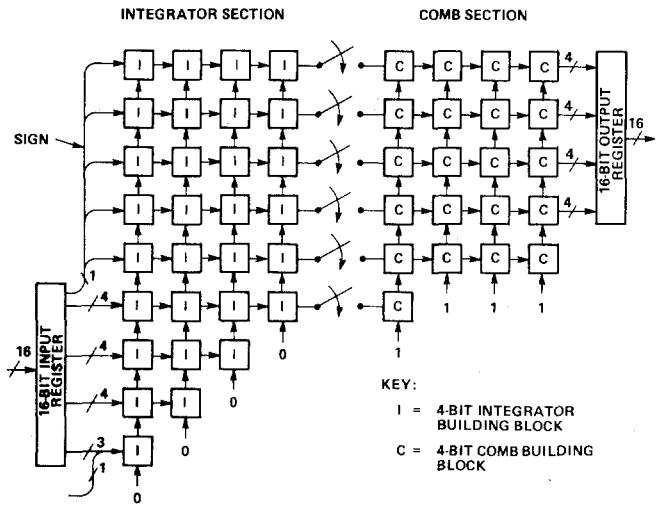

Fig. 5. Example CIC decimation filter for $N = 4$, $M = 1$, and $R = 25$.

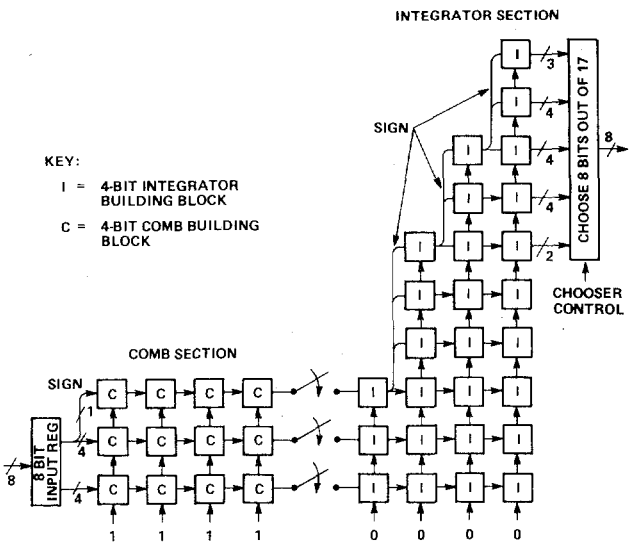

Fig. 6. Example CIC interpolation filter $N = 4$, $M = 2$, and $R = 64$ to 512.

is implemented by taking the one's complement (inverting) and using the low order carry-in port to form the two's complement result.

V. CIC Interpolation Filter Design

A. Design Overview

This section presents design considerations for CIC interpolation filters. For each filter stage the minimum register width is determined. Rounding cannot be used for CIC interpolators (except going into the output register); the introduction of small errors in the integrator stages causes the variance of the error to grow without bound resulting in an unstable filter.

B. Register Growth

The derivation of minimum register width for the jth filter stage is rather straightforward. First, the system function from the filter input up to and including the jth stage is determined. The system function together with a worst case input signal are used to evaluate the maximum register growth up to that point, and the growth together with the input register width is used to determine the minimum register width at the jth stage.

Using this approach, the maximum register growth up to the jth stage can be shown to be

$$G_j = \begin{cases} 2^j, & j = 1, 2, \cdots, N \\ \dfrac{2^{2N-j}(RM)^{j-N}}{R}, & j = N+1, \cdots, 2N \end{cases} \tag{22}$$

assuming that the input signal producing this register growth is at the low sampling rate f_s/R. The minimum register width based on this growth is

$$W_j = \lceil B_{\text{in}} + \log_2 G_j \rceil \tag{23}$$

where B_{in} is the input register width.

When the differential delay is one, then, according to (23), the width of the last comb is actually larger than the width of the first integrator that follows. Using the modulo arithmetic argument introduced in Section IV, we can establish the special condition for the last comb such that

$$W_N = B_{\text{in}} + N - 1 \quad \text{if } M = 1. \tag{24}$$

After the last integrator, truncation, or rounding can be used going into the output register. This is the only source of arithmetic error in CIC interpolators. If the number of bits in the output register is B_{out}, then the number of LSB's discarded is

$$B_T = W_{2N} - B_{\text{out}}. \tag{25}$$

Assuming the error has a uniform probability distribution and with the binary point to the right of the LSB of the output register, the error mean is 0.5 for truncation and zero (0) for rounding; the error standard deviation is $\sqrt{1/12} = 0.289$.

C. Design Example

We are to design an interpolation filter to handle rate change factors of $R = 64$, 128, 256, and 512, resulting in a final sample rate of 5 MHz. The input and output register widths are $B_{\text{in}} = B_{\text{out}} = 8$. Truncation is used going into the output register. We know from other considerations that an $N = 4$ stage filter with a differential delay of $M = 2$ will meet frequency design requirements.

Since the same filter will be used over a range of rate change factors, maximum register widths must be chosen over all rate change factors. These maximum widths occur for $R = 512$, the maximum rate change factor. The hardware design must include shifting hardware to choose the appropriate bits from the filter output as a function of the current rate change factor.

The register widths are calculated using (23) resulting in values of 9, 10, 11, 12, 12, 21, 30, and 39, respectively. Since 4 bit parts are to be used, the actual widths implemented are 12, 12, 12, 12, 12, 24, 32, and 40. The number of LSB's discarded going into the output register (as controlled by the shifting hardware) is $B_T = 22, 25, 28$, or 31 for the four rate change factors.

Fig. 6 shows the implementation of the interpolator using the two basic building blocks shown in Fig. 4. Hardware is

required to vary the rate change factor. In addition, a chooser is required to select the output bits. This selection is a function of the rate change factor.

VI. Conclusions

It has been shown that CIC filters are an economical alternative to conventional decimation and interpolation filters.

CIC filters are implemented using a cascade of ideal integrator stages operating at a high sampling rate and an equal number of comb stages operating at a low sampling rate. These filters require no multipliers and use limited storage; their regular structure, demonstrated by Figs. 5 and 6, simplify their implementation in hardware; they can be applied easily to problems requiring a rate change factor that is selectable over a wide operating range.

The frequency response of CIC filters is fully determined by only three integer parameters resulting in a limited range of filter characteristics. The aliasing/imaging error in the passband can be held within arbitrary bounds by appropriate choice of these parameters. However, the bandwidth and the frequency response outside the passband are severely limited.

For CIC decimation filters, truncation or rounding may be used at each stage of the filter with a nondecreasing number of LSB's discarded at successive stages. The MSB of each stage is proportional to the maximum register growth expected at the filter output. This requires that all stages have the same MSB.

For CIC interpolation filters, the use of truncation or rounding will produce an unstable filter response. As a result, full precision arithmatic must be used at each stage of the filter. Unlike CIC decimators, however, the MSB increases in successive stages with the MSB of each stage being proportional to the register growth from the filter input up to the stage in question.

Appendix I
System Function for CIC Decimators

In this Appendix we derive (9), the system function for CIC decimators from the jth stage up to and including the last stage. The form of the function is that of a fully expanded polynomial in z^{-1}. There are two cases expressed by (9): case 1, where j is in the range 1 to N and case 2, where j is in the range $N+1$ to $2N$.

First we derive case 1. In this case there are $N-j+1$ integrators and N combs. The system function is simply

$$H_j(z) = H_I^{N-j+1} H_C^N, \quad j = 1, \cdots, N. \tag{A1}$$

Substituting (1) and (2) into (A1) results in

$$H_j(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^{N-j+1}}. \tag{A2}$$

This equation can be expanded by dividing the denominator into the numerator resulting in

$$H_j(z) = (1 - z^{-RM})^{j-1} \left[\sum_{k=0}^{RM-1} z^{-k} \right]^{N-j+1}. \tag{A3}$$

A dimensional analysis of (A3) indicates that the order of the polynomial in terms of z^{-1} is

$$RM(j-1) + (RM-1)(N-j+1) = (RM-1)N + j - 1. \tag{A4}$$

Thus, the system function can be expressed as a fully expanded polynomial of the same order. The polynomial has the form

$$H_j(z) = \sum_{k=0}^{(RM-1)N+j-1} h_j(k) z^{-k} \tag{A5}$$

where $h_j(k)$ are the polynomial coefficients.

Another way of expressing (A2) is in terms of its binomial expansion. This results in

$$H_j(z) = \left[\sum_{l=0}^{N} (-1)^l \binom{N}{l} z^{-RMl} \right] \cdot \left[\sum_{v=0}^{\infty} \binom{N-j+v}{v} z^{-v} \right] \tag{A6}$$

and taking the cross product of the two polynomials results in

$$H_j(z) = \sum_{l=0}^{N} \sum_{v=0}^{\infty} (-1)^l \binom{N}{l} \binom{N-j+v}{v} z^{-(RMl+v)}. \tag{A7}$$

In this expression, terms with identical powers of z^{-1} can be collected together. Thus, for a particular nonnegative value k, where

$$k = RMl + v \tag{A8}$$

it is apparent that l can range over the integers

$$l = 0, 1, \cdots, \lfloor k/RM \rfloor \tag{A9}$$

without forcing v out of range. Using (A8) and (A9) we can now collect terms resulting in the fully expanded polynomial

$$H_j(z) = \sum_{k \geq 0} \left[\sum_{l=0}^{\lfloor k/RM \rfloor} (-1)^l \binom{N}{l} \binom{N-j+k-RMl}{k-RMl} \right] z^{-k}, \quad j = 1, 2, \cdots, N. \tag{A10}$$

The form of this polynomial is the same as (A5) where the range of k is established as $k = 0, 1, \cdots, (RM-1)N + j - 1$. This results in (9) for case 1.

We now derive case 2. In this case, where j is in the range $j = N+1, \cdots, 2N$, there are $2N+1-j$ combs and no integrators. The system function is simply

$$H_j(z) = H_C^{2N+1-j}, \quad j = N+1, \cdots, 2N \tag{A11}$$

and substituting (2) into (A11) results in

$$H_j(z) = (1 - z^{-RM})^{2N+1-j}, \quad j = N+1, \cdots, 2N. \tag{A12}$$

The binomial expansion of (A12) results in (9) for case 2.

Appendix II
Maximum Register Growth in CIC Decimators

Equation (10b) is derived resulting in a simplified expression for the maximum register growth in CIC decimators. This register growth is defined by (10a).

It is apparent that (9a), evaluated at $j = 1$, is the system function for the composite CIC filter and is just an alternate

form of (3). Combining these two equations results in

$$H_1(z) = \sum_{k=0}^{(RM-1)N} h_1(k)z^{-k} = \left[\sum_{k=0}^{RM-1} z^{-k}\right]^N \tag{A13}$$

and evaluating (A13) at $z = 1$ results in

$$H_1(1) = \sum_{k=0}^{(RM-1)N} h_1(k) = (RM)^N. \tag{A14}$$

In equation (A13) it is noted that the system function is the product of N system functions of the form

$$\sum_k z^{-k}. \tag{A15}$$

Since this polynomial has all positive coefficients, it follows that the product of two or more of these polynomials results in a polynomial that also has all positive coefficients. As a result we can equate the coefficients with their absolute values. This results in a version of (A14) with the form

$$\sum_{k=0}^{(RM-1)N} |h_1(k)| = (RM)^N. \tag{A16}$$

Substituting this expression into (10a) results in (10b), the equation to be derived.

Appendix III
Design Equation for CIC Decimators

In this Appendix, (21) is derived. The equation is used in the design of CIC decimators to determine the number of LSB's to discard at each stage of filtering. The design criteria is to make the variance from the first $2N$ error sources less than or equal to the variance from the last error source (i.e., error source $2N + 1$), and also to distribute the error about equally among these sources. These criteria can be expressed by the inequality

$$\sigma_{T_j}^2 \leqslant \frac{1}{2N}\sigma_{T_{2N+1}}^2, \quad j = 1, 2, \cdots, 2N. \tag{A17}$$

In (12) if one assumes that either truncation or rounding is being used then (12), (14), and (16a) can be combined resulting in

$$\sigma_{T_j}^2 = \frac{1}{12}2^{2B_j}F_j^2. \tag{A18}$$

This equation combined with (A17) results in the inequality

$$2^{2B_j}F_j^2 \leqslant \frac{6}{N}\sigma_{T_{2N+1}}^2. \tag{A19}$$

Taking the logarithm base 2 of (A19) and rearranging terms results in

$$B_j \leqslant -\log_2 F_j + \log_2 \sigma_{T_{2N+1}} + \frac{1}{2}\log_2 \frac{6}{N}. \tag{A20}$$

One choice of B_j is the largest integer not greater than the expression on the right-hand side of (A20). This choice results in (21).

References

[1] R. E. Crochiere and L. R. Rabiner, "Optimum FIR digital filter implementations for decimation, interpolation, and narrowband filtering," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 444–456, Oct. 1975.

[2] L. R. Rabiner and R. E. Crochiere, "A novel implementation for narrowband FIR digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 457–464, Oct. 1975.

[3] R. E. Crochiere and L. R. Rabiner, "Further considerations in the design of decimators and interpolators," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 296–311, Aug. 1976.

[4] D. J. Goodman and M. J. Carey, "Nine digital filters for decimation and interpolation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 121–126, Apr. 1977.

[5] A. Peled and B. Liu, "New hardware realizations of digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-22, pp. 456–462, Dec. 1974.

Eugene B. Hogenauer (M'80) was born in Port Chester, NY, on May 30, 1943. He received the B.S.E.E. degree from Clarkson College of Technology, Potsdam, NY, in 1965, and the M.S. degree in applied mathematics from Northwestern University, Evanston, IL, in 1967.

From 1967 to 1971 he worked at GTE Sylvania, Mountain View, CA, as a Scientific Programmer in the areas of optimization techniques and system simulation. From 1971 to 1975 he worked at Systems Control, Inc., Palo Alto, CA, specializing in algorithm design for digital signal processing applications. Since 1975 he has been a member of the Technical Staff at ESL, Inc., Sunnyvale, CA, where he has concentrated on software engineering of real-time digital signal processing systems. He has also contributed to the functional design of digital filtering hardware.