

Circuits and Systems Letters

On Prefilters for Digital FIR Filter Design

P. P. VAIDYANATHAN AND G. BEITMAN

Abstract—A new family of digital prefilter structures is introduced, based on the Dolph–Chebyshev function. These prefilters can be combined with appropriately designed “equalizer” filters based on equiripple methods, leading to efficient FIR digital filter designs. Design examples are included, demonstrating the simplicity of the resulting designs, as compared to conventional equiripple designs.

I. INTRODUCTION

In a recent paper [1], Adams and Willson introduced a novel technique for the design of linear phase FIR digital filters with very few multipliers. This technique can be briefly outlined as follows: Given a set of specifications on the frequency response in terms of cutoff frequencies and attenuation tolerances, the transfer function $H(z)$ is obtained as a cascade of two transfer functions $H_1(z)$ and $H_2(z)$. One of these transfer functions $H_2(z)$, which is called the “prefilter,” is extremely simple to implement, requiring very few additions and no multiplications. The prefilter $H_2(z)$ provides a reasonable stopband attenuation, but has poor passband response. The filter $H_1(z)$ which is designed by appropriately modifying the Parks–McClellan algorithm [2] compensates for this, leading to an overall filter $H(z)$ that meets all the specifications. The order of $H_1(z)$, which is called the equalizer, is typically much lower than the order of an equiripple filter designed directly with the Parks–McClellan algorithm. This is partially because $H_2(z)$ already contributes to some attenuation in the stopband. The overall implementation is, therefore, simpler than that of a conventional approach.

A number of prefilter functions are presented in [1] and [3], most of them being based on the “Recursive Running Sum,” (RRS) defined to be

$$R(z) = 1 + z^{-1} + \dots + z^{-(L-1)} = (1 - z^{-L}) / (1 - z^{-1}). \tag{1}$$

The function $R(z)$ is a lowpass function having equispaced zeros on the unit circle, and provides a minimum stopband attenuation of about 13 dB (with respect to the passband peak at $\omega = 0$). The implementation requires only two adders and L delays.

The purpose of this paper is to introduce newer prefilter structures, based on the well-known Dolph–Chebyshev functions [4], which are low-pass functions based on the Chebyshev polynomial. Our motivation for this choice is twofold. First, these functions are optimal in the sense that the stopband has equiripple behavior, offering the largest possible attenuation for a given order. Second, the implementation of these functions involves

Manuscript received July 30, 1984. This work was supported in part by the National Science Foundation under Grant ECS 84-04245.

The authors are with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125.

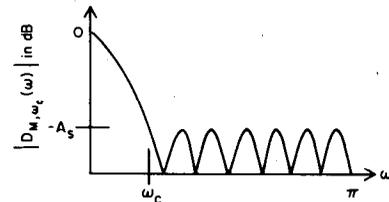


Fig. 1. The Dolph–Chebyshev function.

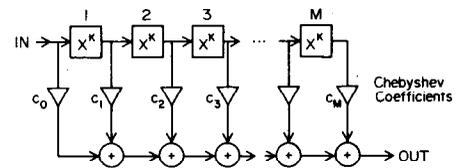


Fig. 2. Implementation of $T_M(X^K)$

only the coefficients of Chebyshev polynomials, which turn out to be extremely simple combinations of powers of two, for low orders. For example, consider the seventh-order Chebyshev polynomial:

$$T_7(X) = 64X^7 - 112X^5 + 56X^3 - 7X. \tag{2}$$

This can be implemented as

$$T_7(X) = 2^6 X^7 - (2^7 - 2^4) X^5 + (2^6 - 2^3) X^3 - (2^3 - 1) X \tag{3}$$

requiring no multiplications.

II. NEW PREFILTERS BASED ON CHEBYSHEV POLYNOMIALS

The Dolph–Chebyshev function of order M and cutoff frequency ω_c is defined as

$$D_{M, \omega_c}(\omega) = T_M(X) / T_M(X_c),$$

$$X = \cos \frac{\omega}{2} / \cos \frac{\omega_c}{2},$$

$$X_c = 1 / \cos \frac{\omega_c}{2}. \tag{4}$$

Fig. 1 shows a typical magnitude response plot of this low-pass function. The minimum stopband attenuation is

$$A_s = 20 \log_{10} T_M(X_c). \tag{5}$$

It is clear that, for a given M , the attenuation increases as ω_c is increased, and that for fixed ω_c , the attenuation increases with M . In general, a large M implies a larger complexity of the prefilter and hence, a better way of increasing the attenuation is to design the prefilters according to

$$|H_2(e^{j\omega})| = |T_M(X^K) / T_M(X_c^K)| \tag{6}$$

where K is a positive integer > 1 . We, therefore, have a family of

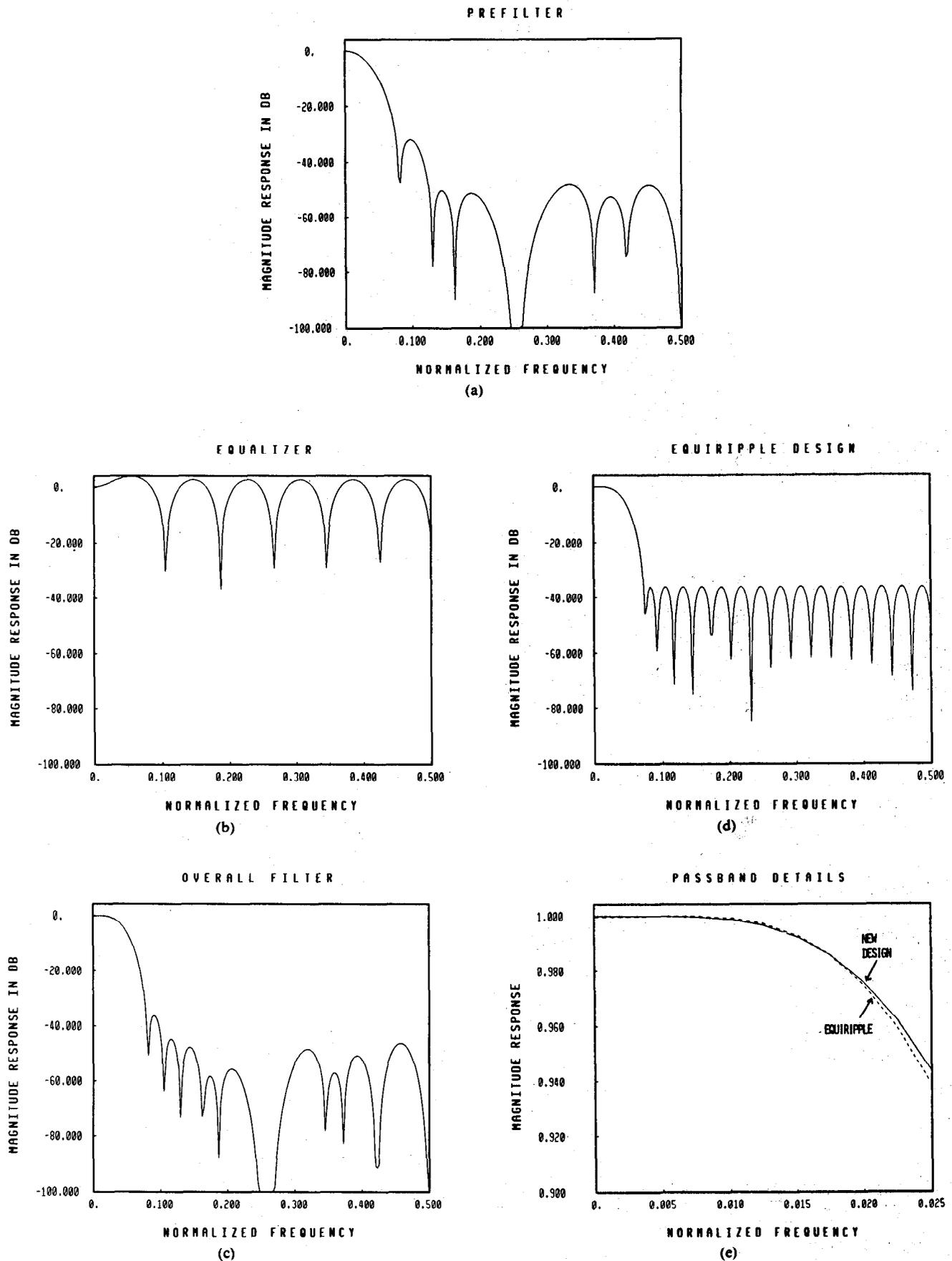


Fig. 3. Example 3.1. (a) Prefilter $H_2(z)$. (b) Equalizer $H_1(z)$. (c) Overall filter $H_1(z)H_2(z)$. (d) Conventional equiripple design. (e) Passband details.

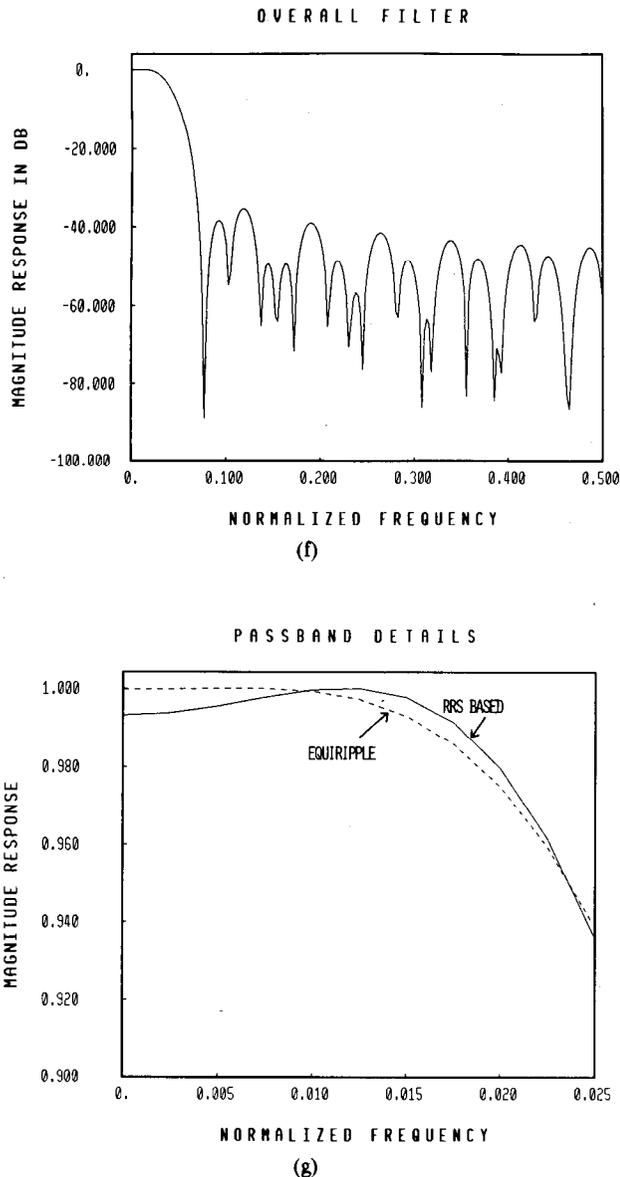


Fig. 3. (Continued). Example 3.1. (f) Overall filter response, with RRS prefilter. (g) Passband details for RRS-based design.

prefilters for lowpass designs, each member in the family being indexed by the triple (M, K, ω_c) . The choice of M , K and ω_c depends upon the given filter specifications. Thus consider a typical lowpass specification, with passband edge at ω_p and stopband edge at ω_s . Clearly ω_c should be in the range $\omega_s \leq \omega_c < \pi$. Moreover, ω_c should be as close to ω_s as possible, so that the attenuation due to the prefilter encompasses the entire stopband $\omega_s \leq \omega \leq \pi$. However, smaller is ω_c , lower is the attenuation obtainable from the prefilter (Equation (6)). Thus for narrow-band filters a large M might be required, which leads to a large dynamic range in the coefficients of $T_M(X)$, and the prefilters then tend to get complicated. This difficulty can be avoided by designing the prefilter $H_2(z)$ according to

$$H_2(z) = \hat{H}_2(z^2) \hat{H}_2(z) \quad (7)$$

where $\hat{H}_2(z)$ is of the form of (6) with ω_c being nearly equal to $2\omega_s$ rather than ω_s . The prefilter of (7) is not equiripple in its stopband anymore but provides excellent attenuation.

The choice of M should be made carefully. If M is large, then $H_2(z)$ provides a good attenuation in the stopband, but at the

TABLE I
IMPULSE RESPONSE COEFFICIENTS $h_1(n)$ FOR THE EQUALIZER OF
EXAMPLE 3.1

Filter Length = 14	
impulse response	
$h(1) = -0.5745 = h(14)$	
$h(2) = 0.2029 = h(13)$	
$h(3) = 0.1847 = h(12)$	
$h(4) = 0.1758 = h(11)$	
$h(5) = 0.1734 = h(10)$	
$h(6) = 0.1718 = h(9)$	
$h(7) = 0.1731 = h(8)$	

same time, spans a large dynamic range in its passband. This means that an equalizer $H_1(z)$ is required, with a large dynamic range of passband response. As a result, the coefficients $h_1(n)$ of the impulse response have large magnitudes, even though they add up approximately to unity near $\omega = 0$. In essence, this means the sensitivity of the overall passband response with respect to the coefficients $h_1(n)$ is large, and, therefore, large M should be avoided.

There is another important factor that governs the choice of ω_c , as can be seen from Fig. 2, which shows an implementation of $T_M(X^K)$. The quantity $M_c = X_c^K$, which is a multiplier coefficient, appears M times. In order to keep the prefilter as simple as possible, ω_c should be chosen so that M_c is a power of 2, or a sum of two or three such powers.

Unlike in designs based on the RRS [1], [3], we now have a wide choice of prefilter parameters. Even though this is an advantage, it is not clear at this point in time as to how to uniquely choose the triple (M, K, ω_c) in order to minimize the overall complexity. However, we believe that the above discussions do give an approximate guideline for this choice.

III. DESIGN EXAMPLES

Example 3.1

Consider a low-pass specification with bandedges $\omega_p = 0.042\pi$, $\omega_s = 0.146\pi$, $A_p =$ maximum passband attenuation $= 0.28$ dB, $A_s =$ minimum stopband attenuation $= -36$ dB. A direct design based on the Parks-McClellan algorithm leads to an equiripple filter of order 33, requiring 17 multipliers.

A prefilter of the form: $H_2(z) = \hat{H}_2(z^2) \hat{H}_2(z)$ where $|\hat{H}_2(e^{j\omega})| = |T_5(X^2)/T_5(X_c^2)|$ with $\omega_c = 0.2951672\pi$ is ideally suited to obtain an efficient design. Fig. 3(a) shows the prefilter response. The above value of ω_c corresponds to the value of $M_c = 1.25 = (1.01)_2$ which is a "2-bit" multiplier. With this choice of $H_2(z)$, the equalizer $H_1(z)$ requires an order of only 13, which corresponds to 7 multipliers.

Fig. 3 also shows the frequency-response plots of the equalizer $H_1(z)$ and the overall filter $H_1(z)H_2(z)$, and also of the conventional equiripple designs. Note that the equalizer contributes very little to the stopband attenuation. The impulse response coefficients $h_1(n)$ of $H_1(z)$, shown in Table I, are reasonably small numbers and do not lead to any sensitivity problems.

The specifications of this example can also be satisfied by using a recursive running sum as a prefilter. Thus following the design rules in [1], the number of taps for the RRS would be $L = 2\pi/\omega_s = 13.7$, and the choice $L = 13$ is proper. With the

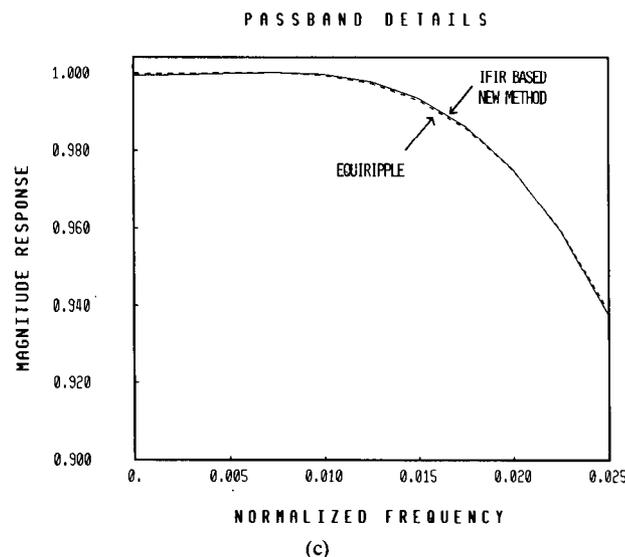
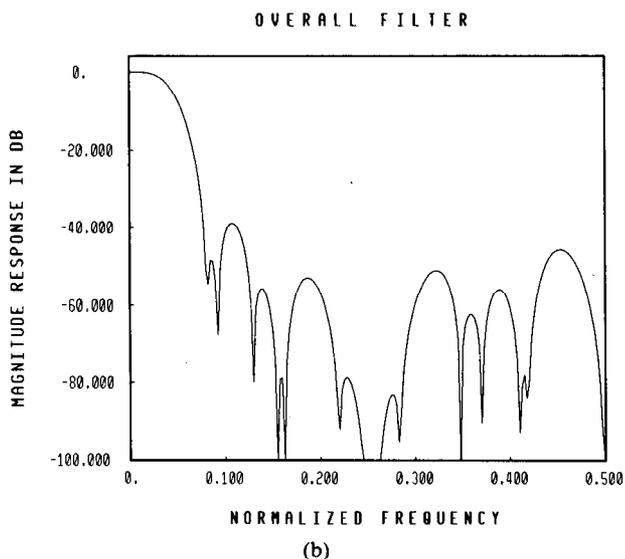
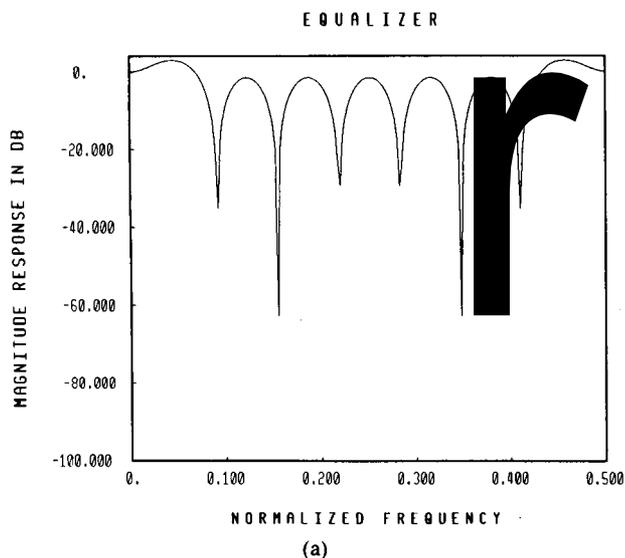


Fig. 4. Example 3.2. (a) Equalizer $H_1(z)$. (b) Overall filter $H_1(z)H_2(z)$. (c) Passband details.

prefilter, therefore, chosen as

$$H_2^{(r)}(z) = \frac{1 - z^{-13}}{1 - z^{-1}}$$

the optimal equalizer $H_1^{(r)}(z)$, designed using McClellan's program, requires a length of 28. The overall design therefore requires 14 multipliers and $27 + 2 = 29$ adders. Fig. 3(f) and (g) show the relevant frequency response plots, for the design based on RRS.

For this design example, the RRS-based design is marginally better than a direct equiripple design. The new design, based on Dolph-Chebyshev functions, is considerably more efficient than an equiripple design, and requires less than half as many multipliers.

Example 3.2:

This is a repetition of Example 3.1, with the exception that the equalizer $H_1(z)$ is designed using the interpolated FIR (IFIR) approach [5]. Essentially, instead of designing the filter $H_1(z)$, a filter $\hat{H}_1(z)$ is first designed with ω_p and ω_s equal to two times the specified values. The filter $\hat{H}_1(z)$ requires a lower order than $H_1(z)$ because of a wider transition bandwidth. Then the equalizer is taken to be $\hat{H}_1(z^2)$, which now has the desired cutoff frequencies ω_p and ω_s , but has an additional passband around $\omega = \pi$. But this passband is automatically suppressed by the prefilter $H_2(z)$, used in Example 3.1. In summary, $\hat{H}_1(z^2)H_2(z)$ meets all the desired specifications.

In our example, an order of 8 was found to be sufficient for $\hat{H}_1(z)$. We thus have only five multipliers in the implementation. Fig. 4 shows the response of $\hat{H}_1(z^2)H_2(z)$. This example demonstrates how the powerful IFIR approach can be judiciously combined with the use of Dolph-Chebyshev prefilters.

Example 3.3

A bandpass design example is next considered.

Consider the following bandpass specifications:

$$\text{stopbands: } 0 \leq \frac{\omega}{2\pi} \leq 0.126 \text{ and } 0.374 \leq \frac{\omega}{2\pi} \leq 0.5$$

$$\text{passband: } 0.18 \leq \frac{\omega}{2\pi} \leq 0.32$$

$$A_p = \text{max passband attenuation} \leq 0.36 \text{ dB,}$$

$$A_s = \text{min stopband attenuation} \geq 32 \text{ dB.}$$

A direct equiripple design requires an order of 29. A prefilter-based design, with the following prefilter:

$$|H_2(e^{j\omega})| = T_5 \left[\left(\frac{\sin \omega}{\sin \omega_0} \right)^2 \right]$$

with $\omega_0 = \pi/4$ was found most appropriate for this design, and required an equalizer of order 16. Fig. 5(a) shows the prefilter response, and Fig. 5(b) shows the overall frequency response of $H_1(z)H_2(z)$. The figure also shows the response of the conventional equiripple design. The complexity of implementations for the three examples is summarized in Table II. Note that a causal version of "cos $\omega/2$ " corresponds to $((1 + z^{-1})/2)$, requiring one addition operation.

CONCLUDING REMARKS

The results reported in this paper and those reported in [1] and [3] indicate that a wide family of prefilter structures can be constructed. The choice of a most appropriate prefilter for a

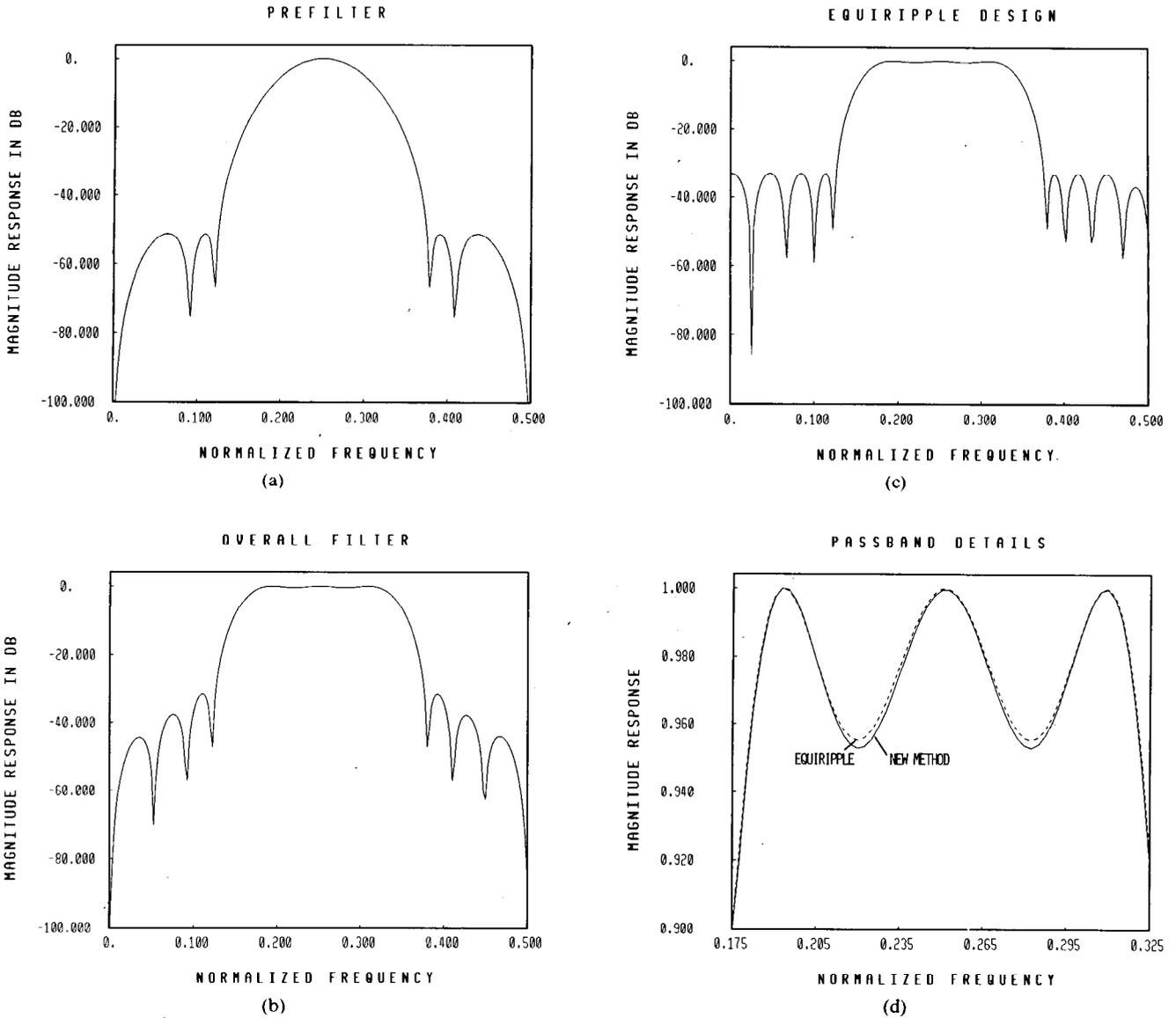


Fig. 5. Example 3.3. (a) Prefilter $H_2(z)$. (b) Overall filter $H_1(z)H_2(z)$
 (c) Conventional equiripple design. (d) Passband details.

TABLE II
 COMPLEXITY COMPARISON FOR THE EXAMPLES

	Prefilter	Equalizer	Total for $H_1(z)H_2(z)$	Direct Equiripple Design
<u>Ex 3.1</u> (new approach)	38 adders	13 adders 7 multipliers	51 adders 7 multipliers	33 adders 17 multipliers
<u>Ex 3.1</u> (RRS-based)	2 adders (RRS)	27 adders 14 multipliers	29 adders 14 multipliers	33 adders 17 multipliers
<u>Ex 3.2</u> (new approach with IFIR)	38 adders	8 adders 5 multipliers	46 adders 5 multipliers	33 adders 17 multipliers
<u>Ex 3.3</u> (new approach, bandpass)	14 adders	16 adders 9 multipliers	30 adders 9 multipliers	29 adders 15 multipliers

given application does not seem to be unique, but based on intuitive guidelines one can generally obtain a very efficient overall design, as demonstrated by examples in Section III, and in [1], [3]. Moreover, in conjunction with the novel IFIR concept [5], the efficiency of the overall implementation improves dramatically for narrow-band filters, as shown by Table II.

ACKNOWLEDGMENT

The authors would like to thank Dr. J. F. Kaiser of the Bell Communications Research, Inc., for useful discussions. This work was part of a 3rd term project, for first year MS students at Caltech, Pasadena, CA.

REFERENCES

- [1] J. W. Adams and A. N. Willson, Jr., "A new approach to FIR digital filters with fewer multipliers and reduced sensitivity," *IEEE Trans. Circuits Syst.*, vol. CAS-30, pp. 277-283, May 1983.
- [2] T. W. Parks and J. H. McClellan, "Chebyshev approximation for nonrecursive digital filters with linear phase," *IEEE Trans. Circuit Theory*, vol. CT-19, pp. 189-194, Mar. 1972.
- [3] J. W. Adams and A. N. Willson, Jr., "Some efficient digital prefilter structures," *IEEE Trans. Circuits. Syst.*, vol. CAS-31, pp. 260-265, Mar. 1984.
- [4] H. D. Helms, "Nonrecursive digital filters: Design methods for achieving specifications on frequency response," *IEEE Trans. Audio Electroacoust.*, vol. AU-16, pp. 336-342, Sept. 1968.
- [5] Y. Neuvo, C. Y. Dong, and S. K. Mitra, "Interpolated finite impulse response filters," *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-32, pp. 563-570, June 1984.

A Non-Exact-Multiplication Scheme for Digital Filter Implementation Using Bit-Slice Components

Y. C. LIM

Abstract—In the implementation of digital filters using bit-slice components, a significant amount of hardware can be saved with insignificant loss of performance if the multiplications involving the least significant bits of the signal and coefficient values are ignored. The effect is the introduction of an error noise source in the multiplier. Optimum hardware count for a given total output noise power is achieved when the noise power arises from ignoring the product of the least significant bits equals the noise power due to arithmetic rounding.

I. INTRODUCTION

Recent research trends in digital filter design, synthesis, and implementation techniques focus on the issue of reducing hardware complexity for a given performance figure. Among these techniques are the multiplierless designs [1]-[5], coefficient wordlength optimizations [2]-[10], low sensitivity structures [11]-[15], and efficient implementation methods [16]-[18]. In this letter, we show that hardware complexity can be reduced with

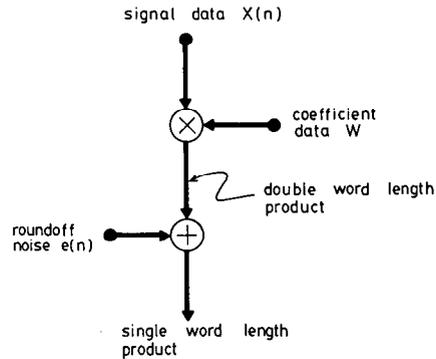


Fig. 1. Roundoff noise model for product wordlength truncation.

insignificant loss of performance by using non-exact-multiplication method.

When a signal data is multiplied by a coefficient data, the product is a double wordlength data. The wordlength of the product is then truncated to minimize hardware complexity. The effect of this truncation process is well known [19] and can be modeled by introducing a roundoff noise $e(n)$, as shown in Fig. 1. The roundoff noise power $E\{e(n)^2\}$ is $Q^2/12$ where Q is the quantization step size [19]. Since the least significant bits of the product is to be truncated, they need not be computed exactly. Non-exact computation of the least significant bits produces significant saving in hardware with insignificant loss of performance.

II. NON-EXACT-MULTIPLICATION SCHEME

Consider multiplying an unsigned signal data $X(n)$ by an unsigned coefficient data W . Without loss of generality let the magnitudes of $X(n)$ and W be each less than unity. If their word lengths are $(B+1)R$ bits and $(A+1)R$ bits, respectively, they can be represented as

$$X(n) = \sum_{b=0}^B x(b, n)2^{-bR} \tag{1a}$$

$$W = \sum_{a=0}^A w(a)2^{-aR}. \tag{1b}$$

The wordlength of $w(a)$ and $x(b, n)$ are each R bits. Also the magnitudes of $w(a)$ and $x(b, n)$ are each less than unity. The product $WX(n)$ (assuming $B < A$) is given by

$$\begin{aligned} WX(n) = & \sum_{m=0}^B \sum_{a=0}^m w(a)x(m-a, n)2^{-mR} \\ & + \sum_{m=B+1}^A \sum_{a=m-B}^m w(a)x(m-a, n)2^{-mR} \\ & + \sum_{m=A+1}^{A+B} \sum_{a=m-B}^A w(a)x(m-a, n)2^{-mR}. \end{aligned} \tag{2}$$

Each of the $w(a)x(m-a, n)$ product term in (2) can be implemented using an R by R bit-slice multiplier. If all product terms with weighting factor less than or equal to 2^{-DR} are ignored,

Manuscript received October 30, 1984.
The author is with the Electrical Engineering Department, National University of Singapore, Kent Ridge, Singapore 0511.