Design of Extrapolated Impulse Response FIR Filters With Residual Compensation in Subexpression Space

Ya Jun Yu, *Senior Member, IEEE*, Dong Shi, *Student Member, IEEE*, and Yong Ching Lim, *Fellow, IEEE*

***Abstract*—In this paper, an extrapolated impulse response filter with residual compensation is proposed for the design of discrete coefficient finite-impulse response (FIR) filters using subexpression sharing. The proposed technique utilizes the quasi-periodic nature of the filter impulse response to approximate the filter coefficients. The reduced degree of freedom of filter coefficients due to the quasi-periodic approximation is perfectly restored by introducing a residual compensation technique. The resulting subexpression sharing synthesis of discrete coefficient FIR filters has lower complexities than that of the conventional synthesis techniques in terms of number of adders. To further reduce the synthesis complexity, filter coefficients and residuals may be optimized in subexpression spaces. Mixed integer linear programming is formulated for the optimization. Numerical examples show that the number of adders required by synthesizing the filters in the proposed structure is significantly reduced compared to that of the conventional synthesis schemes synthesized in direct or transposed direct form.**

***Index Terms*—Common subexpression sharing, extrapolated impulse response, finite-impulse response (FIR) filters, mixed integer linear programming, residual compensation, subexpression space.**

I. Introduction

MULTIPLIERLESS finite-impulse response (FIR) filters [1]–[4] are very attractive in VLSI implementation because the multiplication of the input sample data with the filter coefficients is achieved by a limited number of additions and shifts. Since the shifts for fixed coefficient values can be implemented by hardwire, the complexity of multiplierless filters lies mainly in the number of adders. In transposed direct form FIR, there are two types of adders. The adders used to generate the coefficient values are called multiplier block (MB) adders whereas the adders used to add the tap signals into the delay chain are called structural adders, as is shown in Fig. 1. Many efforts have been made to reduce the number of these adders and one of the most efficient algorithms to reduce the number of MB adders is the multiple constant multiplication (MCM) techniques [1]–[3], [5]–[16]. Since most MCM algorithms were considered in transposed direct form FIR, in this paper, we also focus on this platform for expository convenience. However, it is noted that the direct form FIR may synthesize the filter in the same complexity.

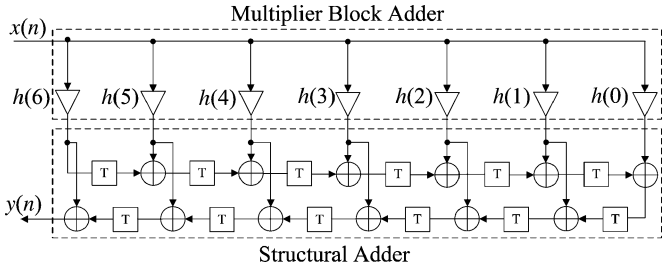


Fig. 1. Transposed direct form of a 12th-order FIR filter.

The MCM techniques can be grouped into two categories. In the first category, algorithms are applied on an FIR design with a given set of discrete coefficients. The common subexpression patterns are extracted and shared among the discrete filter coefficients. These algorithms can be further classified into three types. The first type is common subexpression sharing [1], [5], [6] in which the common digit patterns of the coefficients, usually represented in a canonic signed digital (CSD) form, or binary numbers are found and shared. The second type of algorithms are based on the adder graph [2], [7] in which larger coefficients are realized by shifting and adding those already realized smaller coefficients. The third type is the difference method [3], [8] of which the differences of the coefficient values, instead of the coefficient values, are synthesized to reduce the overall complexity.

The algorithms in the first category, however, suffer from the fact that the searching space is limited by the given discrete coefficient values. Furthermore, it has been proved in [9] that for a filter with L distinct positive non-one coefficient values (after the even and negative coefficient values are transformed to positive odd numbers by scaling the coefficient values with a proper signed power-of-two factor), the lower bound of the number of MB adders of the transposed direct form implementation is equal to the minimum number of adders required to realize the simplest coefficient plus $(L-1)$. Since there are always very small magnitude coefficients which can be realized by one adder in a practical filter, in most cases, the number of adders required to realize the simplest coefficient is one. Therefore, the lower bound is basically determined by the number of distinct coefficient values L. For many benchmark filters, this lower bound has been achieved, for instance, example 1 of [17]. The example is a 120th-order filter with 52 distinct coefficient values. Many algorithms [2], [10]–[12] have achieved this lower bound. To further reduce the number of MB adders, the only way is to reduce the lower bound by reducing the number of distinct coefficient values.

Manuscript received April 28, 2008; revised November 03, 2008. First published February 24, 2009; current version published December 31, 2009. This work was supported in part by Nanyang Technological University (NTU) and Temasek Laboratory @ NTU. This paper was recommended by Associate Editor Y.-P. Lin. .

The authors are with the School of Electrical and Electronic Engineering, Nanyang Technological University, 639798 Singapore (e-mail: eleyuyj@pmail.ntu.edu.sg; shid0001@ntu.edu.sg; elelimyc@pmail.ntu.edu.sg).

Digital Object Identifier 10.1109/TCSI.2009.2016165

In the second category, to overcome the limited searching space in the first category and to reduce the number of distinct coefficient values, filter coefficients are optimized directly in subexpression space to meet the filter specifications [13]–[16]. In [13], the dynamic range of each coefficient is found before an exhaustive search is performed to check all the possible combinations to find the best solution. In the technique proposed in [14], the subexpression sharing problem is modeled as a 0/1 integer linear programming to find the minimum number of adders subject to the filter specification. The algorithm in [15] applies a local search in the vicinity of discrete coefficient values when the coefficients are synthesized in subexpression sharing. The algorithms in [13]–[15] may produce filters with minimum number of adders in direct form implementation. However, they are computationally demanding and thus only suitable for relatively short filters. Recently, a branch and bound (B&B) mixed integer linear programming (MILP) is proposed to optimize FIR filter coefficients directly in a subexpression space which is built upon a predefined subexpression basis set [16]. Though optimum is not guaranteed, the algorithm may design relatively long filters with reduced number of adders.

To reduce the filter complexity by further reducing the lower bound of the number of adders, a residual compensated extrapolated impulse response filter structure is presented in [18]. In [18], the quasi-periodic property of the impulse response of FIR filters is utilized, where the blocks of coefficients with larger magnitude are approximated as scaled versions of smaller ones. The approximation error introduced thereby is compensated by a residual compensation technique. This paper presents a comprehensive study on the residual compensated extrapolated FIR filter proposed in [18]. The new contributions of this paper lie in the following aspects.

1) In view of the superiority of the second category techniques over the first category ones in the MCM algorithm, in this paper, the design of the residual compensated extrapolated FIR filter is formulated as a MILP problem where the filter coefficients and residuals are optimized simultaneously.
2) While the proposed technique reduces the required number of adders, additional delay chain(s) may be used. A low complexity and low power realization of the additional delay chain(s) is proposed.
3) Reduction of the implementation complexity of the proposed filter structure is analyzed with the additional delay chain(s) taken into consideration.
4) Adder depth of the proposed filter structure is investigated based on the design examples. A slight decrease of average adder depth is found in all examples.

Using the proposed technique, the dynamic range of the coefficient multipliers is further reduced, which increases the chance of having coefficients with identical values. This results in a reduction in the lower bound of MB adders. Design examples show that the proposed technique significantly reduces the filter complexity in terms of the number of adders compared with any other existing techniques. In addition, many small coefficient values and residuals diminished to zero, resulting in a reduction in structural adders.

The remaining of this paper is organized as follows. Section II reviews the extrapolated impulse response filters. The residual compensated extrapolated impulse response filter structure [18] is introduced in Section III for the completeness of the paper. A synthesis example is used to illustrate the proposed technique. The optimization of the coefficient values and residuals in the subexpression space is formulated as a MILP problem in Section IV. In Section V, benchmark filters are designed to illustrate the efficiency of the proposed technique. Section VI gives a circuit realization of the extra delay chains. In Section VII, the implementation complexity and adder depth of the proposed technique are discussed and compared with the conventional structures.

II. Extrapolated Impulse Response

The impulse response of a typical linear phase FIR filter is quasi-periodic [19]–[21] as shown in Fig. 2. Most of the energy of the impulse response is concentrated at the center lobe, whereas the side lobes have decreasing magnitudes. If $\mathrm{lobe}_0, \mathrm{lobe}_1$ and lobe_2 (see Fig. 2) have the same number of samples, lobe_1 and lobe_2 can be approximated as scaled versions of lobe_0. Lobe_0 is thus served as a prototype lobe. Once the coefficient multipliers in lobe_0 is implemented, the coefficient multipliers in lobe_1 and lobe_2 do not need to be implemented individually; instead, they could be obtained using a single scalar for each lobe and the implementation complexity of the filters are reduced. Lobe_1 and lobe_2 are thus realized as the extrapolations of lobe_0 and this is the origin of the extrapolated impulse response filters [19]. In the original extrapolated impulse response filters, the lobe closest to the center lobe is usually selected as the prototype lobe. However, it is not necessary the case. Any lobe could be the prototype lobe and the other lobes are approximated as scaled versions of the prototype lobe. In this case, it is no longer an extrapolation, but the name of extrapolated filter is still used.

The zero phase z-transform transfer function of a $2N$th order filter is given by

$$H(z) = h(0) + \sum_{n=1}^{N} h(n)(z^n + z^{-n}). \tag{1}$$

Assume that $\mathrm{lobe}_l, l = 0, 1, \ldots, L-1$, begins at $n = p_l$ through $n = q_l$ for an L lobe filter, where $q_{L-1} \leq N$, $H(z)$ can be rewritten as

$$\begin{aligned} H(z) = h(0) &+ \sum_{n=1}^{M} h(n)(z^n + z^{-n}) \\ &+ \sum_{n=p_0}^{q_0} h(n)(z^n + z^{-n}) \\ &+ \sum_{n=p_1}^{q_1} h(n)(z^n + z^{-n}) + \cdots \\ &+ \sum_{n=p_{L-1}}^{q_{L-1}} h(n)(z^n + z^{-n}) \\ &+ \sum_{n=q_{L-1}+1}^{N} h(n)(z^n + z^{-n}). \end{aligned} \tag{2}$$

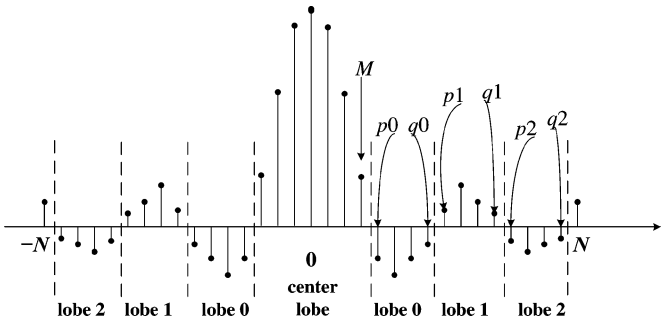


Fig. 2. Typical impulse response of a $2N$th-order FIR filter.

The durations of these lobes are $q_l - p_l + 1$ for $l = 0, 1, \ldots, L-1$, and they might not be all equal. However, they can be separated into several groups, and each group consists of lobes with the same duration. Thus, in each group, any lobe can always be approximated as the scaled version of a prototype lobe in that group. For expository convenience, we assume that all the lobes have the same duration d since other cases are just simple extensions. As a result, (2) can be written as

$$\begin{aligned} H(z) = h(0) &+ \sum_{n=1}^{M} h(n)(z^n + z^{-n}) \\ &+ \sum_{l=0}^{L-1} \sum_{m=1}^{d} h(M+m+ld) \\ &\times (z^{M+m+ld} + z^{-(M+m+ld)}) \\ &+ \sum_{n=M+Ld+1}^{N} h(n)(z^n + z^{-n}) \end{aligned} \tag{3}$$

where d is the duration of each lobe. If the lobe with the smallest magnitude (usually, the $(L-1)$th lobe while lobe_0 is considered as the zeroth lobe) is chosen as the prototype lobe, $H(z)$ can be approximated by

$$\begin{aligned} H(z) \approx \hat{H}(z) = h(0) &+ \sum_{n=1}^{M} h(n)(z^n + z^{-n}) \\ &+ \sum_{m=1}^{d} h(M+m+(L-1)d) \sum_{l=0}^{L-1} \alpha_l (z^{M+m+ld} \\ &+ z^{-(M+m+ld)}) + \sum_{n=M+Ld+1}^{N} h(n)(z^n + z^{-n}) \end{aligned} \tag{4}$$

where α_l is the lth scaling factor and $\alpha_{L-1} = 1$.

A realization of a 12th-order $(N = 6)$ linear phase FIR filter using the above extrapolation technique is given in Fig. 3, where $M = 0$, $L = 2$ and $d = 3$. Coefficients $h(1)$ to $h(3)$ are implemented as scaled version of $h(4)$ to $h(6)$ with a scaling factor α_0.

Optimization techniques have been proposed in [19]–[21] to optimize the filter coefficients of the prototype lobe as well as the scaling factors in continuous spaces. The computational complexity, in terms of the number of multiplication of the resulting extrapolated filter is much reduced when compared with the direct form implementation. The price paid for that is that the order of the extrapolated filter may be slightly higher than that of the minimax optimum [19], [20]. The frequency response of the extrapolated filter is generally degraded if the same order as that of the minimax optimum filter is adopted.

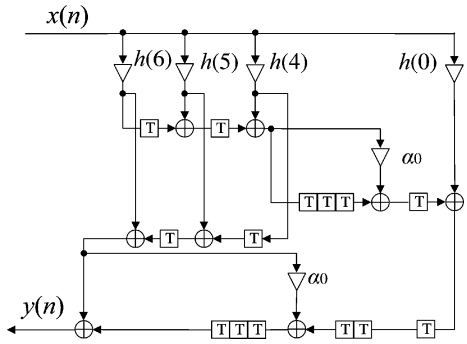


Fig. 3. Structure of extrapolated impulse response.

III. Extrapolated Impulse Response With Residual Compensation

The degradation in the frequency response in the traditional extrapolated filters is due to the fact that the complexity reduction is achieved by reducing the degree of freedom of filter coefficients. Coefficients are only approximations of their optimum values. To meet a given specification, the order of the extrapolated filter is, in general, higher than that of the minimax optimum. In order to overcome this problem, an extrapolated impulse response with residual compensation is introduced to synthesize discrete coefficient filters [18].

A. Structure

Assume that the optimum impulse response of a $2N$th-order linear phase FIR filter for a given specification in a discrete space is $h(n)$ for $-N \le n \le N$. The discrete space, denoted as $\mathcal{D}$, may be the finite-word length space or the signed power-of-two space. Therefore, $h(n) \in \mathcal{D}$, and $h(-n) = h(n)$, for $-N \le n \le N$.

Since the coefficients are symmetric, only the coefficients with non-negative index are considered. Assume further that the coefficients are quasi-periodic with duration d from $n = M+1$ to $n = M + Ld$ for L periods. Thus, $0 \le M < N$, $L \le (N-M)/d$ and both are integers. Following the procedure of the traditional extrapolated filters, the filter coefficients of $h(n)$ for $0 \le n \le M$ and $M + Ld + 1 \le n \le N$ are implemented accurately. For $h(n)$ within the range $M+1 \le n \le M+Ld$, if the period with the smallest magnitude is chosen as the prototype lobe, and all the other periods are approximated as the scaled versions of the prototype lobe, the approximated coefficients, denoted as $h_a(n)$, from $n = M+1$ to $n = M+(L-1)d$ become

$$h_a(n - (L-l-1)d) = \alpha_l h(n)$$

for

$$M + (L-1)d + 1 \le n \le M + Ld$$

and

$$0 \le l \le L-2 \tag{5}$$

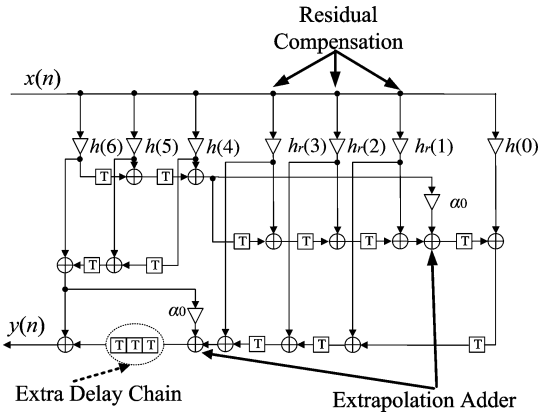


Fig. 4. Extrapolated impulse response with residual compensation.

where the scaling factors α_l are integers or power-of-two numbers for multiplierless implementation. Therefore, the coefficient residuals, denoted as $h_r(n)$, due to the approximation are given by

$$h_r(n) = h(n) - h_a(n), \text{ for } M+1 \le n \le M+(L-1)d. \quad (6)$$

Thus, the z-transform transfer function $H(z)$ can be precisely represented as

$$\begin{aligned} H(z) = h(0) &+ \sum_{n=1}^{M} h(n)(z^n + z^{-n}) \\ &+ \sum_{n=M+Ld+1}^{N} h(n)(z^n + z^{-n}) \\ &+ \sum_{n=M+1}^{M+(L-1)d} h_r(n)(z^n + z^{-n}) \\ &+ \sum_{n=M+(L-1)d+1}^{M+Ld} \sum_{l=0}^{L-1} \alpha_l h(n) \\ &\times (z^{n-(L-1-l)d} + z^{-n+(L-1-l)d}). \end{aligned} \quad (7)$$

In the implementation of the extrapolated filter, the coefficient residuals are compensated by adding the products of the signal samples and the residuals into the tap delay line, as shown in Fig. 4. Thus, the filter impulse response is restored perfectly.

Comparing the residual compensated extrapolated filter implementation shown in Fig. 4 with the transposed direct form implementation shown in Fig. 1, it can be seen that the numbers of structural adders are the same for both structures, since generally each extrapolated coefficient is compensated by a corresponding residual. The MB adders of the proposed technique can be further classified into the prototype coefficient adders and the residual adders, which are used to realize the prototype coefficients and residuals, respectively. Both the prototype coefficients and residuals can share the same MB using MCM algorithms, since they are multiplied by the same signal, as shown in Fig. 4. Since the residual is the difference of the optimum coefficient value and the extrapolated approximation, its magnitude is much smaller than the original optimum value. The magnitudes of the prototype coefficients are small, since the lobe with the smallest magnitude of coefficient values is selected as the prototype lobe. Thus, the dynamic range of the coefficient multiplier is significantly reduced. As a result, the number of distinct odd positive integers is reduced.

It should be noted from Figs. 3 and 4 that, in the extrapolated impulse response structure, besides the structural adders and MB adders, another two types of adders are employed in the residual compensated extrapolated filters. The first type of adder is the extrapolation adders used to add the extrapolated terms into the tap delay line; and the second type is the scaling factor adders used to generate proper scaling factors if they are not power-of-two numbers. To minimize the number of overall adders, the scaling factors are chosen in such a way that as few scaling factor adders as possible are used to reduce the number of distinct residuals as much as possible. If power-of-two numbers are used for the scaling factors, no additional scaling factor adders are requires. However, in some cases, using a few scaling factor adders may achieve more saving in the residual adders.

B. An Example

In spite of the overhead which might be caused by the extrapolation adders and scaling factor adders, the proposed technique still significantly reduce the overall number of adders when compared with any other existing techniques. An example is taken from literature [17] to illustrate the efficiency of the proposed technique.

The discrete filter coefficient values presented in [17] are listed as $h(n)$ in Table I for easy reference. As we have indicated in the introduction, the number of non-one distinct coefficient values (after they have been transformed to positive odd numbers) of the coefficient set of this filter is 52. Therefore, the lower bound of the MB adders is 52, which have been achieved by many algorithms [2], [10]–[12].

By inspecting the coefficient values $h(n)$ in Table I, it is noted that the filter impulse response shows a quasi-periodicity for 5 periods, from $h(4)$ to $h(48)$, with period duration of 9. The period with the minimum coefficient magnitude, i.e., from $h(40)$ to $h(48)$ is chosen as the prototype lobe to approximate the other 4 lobes. The scaling factors for lobes $h(4)$ to $h(12)$, $h(13)$ to $h(21)$, $h(22)$ to $h(30)$ and $h(31)$ to $h(39)$ are chosen to be 20, $-8, 4$, and -2, respectively. Therefore, the approximated coefficient values are given by

$$\begin{aligned} h_a(4+k) &= 20h(40+k) \\ h_a(13+k) &= -8h(40+k) \\ h_a(22+k) &= 4h(40+k) \\ h_a(31+k) &= -2h(40+k), \quad \text{for } k = 0, 1, \ldots, 8. \end{aligned} \quad (8)$$

Thus, the residuals are

$$\begin{aligned} h_r(4+k) &= h(4+k) - 20h(40+k) \\ h_r(13+k) &= h(13+k) + 8h(40+k) \\ h_r(22+k) &= h(22+k) - 4h(40+k) \\ h_r(31+k) &= h(31+k) + 2h(40+k), \quad \text{for } k = 0, 1, \ldots, 8. \end{aligned} \quad (9)$$

Both $h_a(n)$ and $h_r(n)$ for $4 \le n \le 39$ are listed in Table I.

TABLE I
IMPULSE RESPONSE OF A 120TH-ORDER FILTER. $h(n)$ IS THE ORIGINAL DISCRETE COEFFICIENT VALUE OBTAINED IN [17], $h_a(n)$ IS THE EXTRAPOLATED APPROXIMATION OF THE ORIGINAL COEFFICIENT VALUE, AND $h_r(n)$ IS THE RESULTING RESIDUAL. COEFFICIENTS FROM $h(40)$ TO $h(48)$ ARE USED AS THE PROTOTYPE LOBE

n	$h(n)$	$h_a(n)$	$h_r(n)$	n	$h(n)$	$h_a(n)$	$h_r(n)$
0	14686	–	–	31	−78	−72	−6
1	−13494	–	–	32	−148	−140	−8
2	10267	–	–	33	277	260	17
3	−5927	–	–	34	−267	−250	−17
4	1653	720	933	35	143	136	7
5	1515	1400	115	36	23	16	7
6	−3000	−2600	−400	37	−152	−136	−16
7	2823	2500	323	38	193	178	15
8	−1522	−1360	−162	39	−142	−138	−4
9	−109	−160	51	40	**36**	–	–
10	1336	1360	−24	41	**70**	–	–
11	−1739	−1780	41	42	**−130**	–	–
12	1312	1380	−68	43	**125**	–	–
13	−390	−288	−102	44	**−68**	–	–
14	−546	−560	14	45	**−8**	–	–
15	1090	1040	50	46	**68**	–	–
16	−1071	−1000	−71	47	**−89**	–	–
17	585	544	41	48	**69**	–	–
18	84	64	20	49	−24	–	–
19	−616	−544	−72	50	−24	–	–
20	798	712	86	51	55	–	–
21	−600	−552	−48	52	−61	–	–
22	165	144	21	53	45	–	–
23	282	280	2	54	−18	–	–
24	−541	−520	−21	55	−8	–	–
25	524	500	24	56	24	–	–
26	−277	−272	−5	57	−28	–	–
27	−59	−32	−27	58	23	–	–
28	323	272	51	59	−14	–	–
29	−408	−356	−52	60	6	–	–
30	302	276	26				

It can be seen from Table I that the residual coefficient values from $h_r(4)$ to $h_r(39)$ are much smaller than their original values. To implement the filter in residual compensated extrapolated form, the coefficient values to be synthesized are $h(0)$ to $h(3)$, $h_r(4)$ to $h_r(39)$ and $h(40)$ to $h(60)$. Among these coefficient values, the number of distinct non-one coefficient values (after they have been transformed to positive odd numbers) have been reduced to 31. Using RAG-n algorithm [2] to generate the subexpression coefficients, the resulting MB requires 33 adders, which is close to the new lower bound of 31. The reason that 2 more adders are used than the lower bound is that some coefficient values of the center lobe are realized directly, and their magnitudes are large, among which $h(0)$ and $h(2)$, each requires two adders to synthesize.

Symmetric coefficients of linear phase FIR filter can share the MB. However, the extrapolation adders and scaling factor adders for symmetric lobes cannot be shared with each other as shown in Fig. 4. Therefore, besides the MB adders, the extrapolated realization requires additional 8 adders to add the extrapolated lobes into the tap delay line, and additional 2 adders to generate the scaling factor 20. Thus, in total 43 adders are required to synthesize the filter, as shown in Table II. The best

TABLE II
NUMBER OF ADDERS USED TO SYNTHESIZE THE FILTER COEFFICIENTS

	Best result obtained in literature [2], [10]–[12]	Proposed extrapolated structure
Adders in MB	52	**33**
Extrapolation Adders	NA	**8**
Scaling Factor Adders	NA	**2**
Total Adders	52	**43**

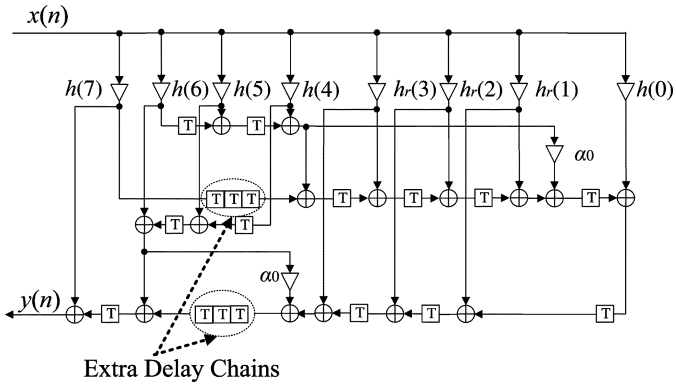


Fig. 5. Extrapolated impulse response with residual compensation without including tail coefficients.

results obtained by synthesizing the same set of discrete coefficients in published literature require 52 adders [2], [10]–[12]; the result is also listed in Table II for comparison.

In the synthesis of the extrapolated impulse response, it is noted that extra delay chains may be used. As indicated by the dashed circles in Fig. 4, a delay chain of length d, $d = 3$ in this example, is used when the coefficient symmetry is exploited. Furthermore, if $N > M + Ld$, the tail coefficients, $h(7)$ as shown in Fig. 5, are not included in any extrapolation lobe, one more delay chain with the same length is required. The delay chain(s) are employed to maintain the delays of tap signals.

The extra delay chains are inherent in the extrapolated impulse response structure. The reason that the extra delay chain was not previously considered in [19]–[21] is because that the overhead introduced by the delay chain is negligible compared with the saving in multipliers. In our proposed technique, however, the arithmetic complexity is reduced to such an extend that it has become necessary to consider the complexity of the delay chain. Fortunately, each delay chain absorbs one tap delay element in the original filter structure, which compensates for the implementation complexity and power consumption. A realization of the delay chains is given in Section VI. A detailed analysis of the power consumed by the delay chain is given in Appendix. The implementation and power consumption overhead due to the delay chains are discussed in Sections VI and VII, respectively.

IV. OPTIMIZING FILTER COEFFICIENTS AND RESIDUALS IN SUBEXPRESSION SPACE

As illustrated in Section III, the proposed technique can significantly reduce the number of adders in the multiplier block. However, the example shown in Section III synthesized the extrapolated impulse response filter based on a given set of discrete coefficients, therefore belongs to the first category optimization

technique for the subexpression sharing. It has been shown in [16] that the second category optimization techniques, where coefficient values are optimized directly in subexpression space, outperformed significantly over the techniques of the first category. In this section, a linear programming problem of the proposed structure is formulated to optimize the filter coefficients and residuals in subexpression space.

A. Review of Subexpression Space

A subexpression space is constructed based on a subexpression basis set [16]. The basis set consists of 0 and odd integers and the order of the basis set is defined as the number of adders needed to generate the values of all the elements in the set. For example, the order of the basis set $\{0, \pm 1, \pm 3, \pm 5, \pm 7\}$ is 3.

A subexpression space is then constructed on a subexpression basis set by defining its element as

$$n = \sum_{i=0}^{K-1} y(i) 2^{q(i)}, y(i) \in S \tag{10}$$

where S is a predefined subexpression basis set and $q(i)$ is an integer. K is the number of allowed subexpression terms. From the equation, it is noted that for a given K and a given subexpression basis set, in general, the constructed subexpression space is a subset of the integer space with the same wordlength.

In optimizing the filter coefficient using B&B MILP in subexpression space [16], the subexpression basis set is first defined and the number of subexpression terms allocated to each coefficient is also specified. The normalized peak ripple magnitude (NPRM) [22] is minimized subject to the filter specification such as band ripple ratios and band edges.

During the branch and bound optimization, filter coefficients are selected for branching and the discrete values assigned to each coefficient is in the predefined subexpression space of the particular coefficient.

B. Formulating the Linear Programming Problem

Based on (7), the frequency response of a linear-phase residual compensated extrapolated impulse response FIR filter can be expressed as

$$\begin{aligned} H(\omega) = {} & h(0) + \sum_{n=1}^{M} h(n)\mathrm{Trig}(\omega, n) + \sum_{n=M+Ld+1}^{N} h(n)\mathrm{Trig}(\omega, n) \\ & + \sum_{n=M+1}^{M+(L-1)d} h_r(n)\mathrm{Trig}(\omega, n) \\ & + \sum_{n=M+(L-1)d+1}^{M+Ld} \sum_{l=0}^{L-1} \alpha_l h(n)\mathrm{Trig}(\omega, n-(L-1-l)d). \end{aligned} \tag{11}$$

In (11), $h(n)$ for $n = 0, 1, \ldots, M, M+Ld+1, \ldots, N$ is the filter coefficient that is synthesized directly; $\alpha_l h(n)$ for $n = M+(L-1)d+1, M+(L-1)d+2, \ldots, M+Ld$ and $l = 0, 1, \ldots, L-1$ is the extrapolated coefficient with the scaling factor α_l prefixed and $\alpha_{L-1} = 1$; and $h_r(n)$ for $n = M+1, M+2, \ldots, M+(L-1)d$ is the residual used to improve the precision of the filter coefficients. Therefore, the total number of the variables to be optimized is equal to the number of the original filter coefficients. The optimization is formulated to find the filter coefficient values of $h(n)$ for $n = 0, 1, \ldots, M, M+Ld+1, \ldots, N$, and residuals $h_r(n)$ for $n = M+1, M+2, \ldots, +M+(L-1)d$, in given spaces, to minimize the frequency response ripples δ

$$\begin{aligned} &\text{minimize:} \quad \delta \\ &\text{subject to:} \quad 1-\delta \le H(\omega) \le 1+\delta, \quad \text{for } \omega \in [0, \omega_p] \\ &\qquad -(\delta_s\delta)/\delta_p \le H(\omega) \le (\delta_s\delta)/\delta_p, \quad \text{for } \omega \in [\omega_s, \pi] \end{aligned} \tag{12}$$

where ω_p and ω_s are the passband and stopband edges, δ_p and δ_s are the passband and stopband ripples, and $H(\omega)$ is the frequency response of the filter given in (11). This problem can be efficiently optimized by B&B MILP [16], [22], [23]. It is noted that the residuals are usually much smaller than their original filter coefficient values; during the branch and bound search, it is more likely to produce zero-valued residuals. A zero-valued residual contributes to two adders reduction in the structural adders and therefore is preferred.

The criteria of the selection for a coefficient for branching could be found in [24]. In particular, the coefficients in the prototype lobe have more impact on the frequency response and hence should be branched at early stage in the branch and bound optimization. The bound condition is determined by the subexpression space predefined for each coefficient [16]. The computation time in optimizing the coefficient values and residuals of the proposed technique is similar to that of MILP in the optimization of filters in other discrete spaces and/or structures. The computation time increases exponentially with the number of variables [16], [24]. For a given number of variables, the computing time required for different specifications differs widely [24]. The typical computation time for designing a 60th-order filter, for instance, ranges from 20 min to 2 h on a Pentium 4 2.4 GHz desktop computer with 3 GB RAM.

V. Design Examples

In this section, several benchmark filters from literature are designed to illustrate the superiority of the proposed technique. The detailed design procedures are given for the first two examples in Sections V-A and V-B, whereas five more examples are collectively presented in Section V-C.

A. Example L2

Example $L2$ is taken from the second example of [17]. The filter specification is as follows: a lowpass filter with order 62 has passband and stopband edges at $\omega_p = 0.2\pi$ and $\omega_s = 0.28\pi$, respectively; the passband and stopband ripples are $\delta_p = 0.028$ and $\delta_s = 0.001$, and the effective wordlength (excluding the sign bit) is 11. By inspection of the optimal continuous coefficients, the 26 coefficients from $h(0)$ to $h(25)$ consist of 2 lobes, each comprising of 13 coefficients. Thus, the coefficients from $h(13)$ to $h(25)$ are chosen as the prototype lobe_1. $Lobe_0$ (from $h(0)$ to $h(12)$) is approximated by the prototype lobe_1 with a scaling factor $\alpha_0 = -4$. This can be expressed mathematically as

$$h_a(k) = -4h(13+k), \quad \text{for } k = 0, 1, 2, \ldots, 8. \tag{13}$$

TABLE III
IMPULSE RESPONSE OF EXAMPLE $L2$. COEFFICIENTS FROM $h(13)$ TO $h(25)$ ARE CHOSEN AS THE PROTOTYPE LOBE. RESIDUALS FROM $h_r(0)$ TO $h_r(12)$ AND COEFFICIENTS FROM $h(13)$ TO $h(31)$ ARE REALIZED IN THE MULTIPLIER BLOCK. $h(n) = h_r(n) + \alpha_0 h(n+13)$ FOR $n = 0, 1, \ldots, 12$, WHERE $\alpha_0 = -4$

n	$h(n)$	$h_a(n)$	$h_r(n)$	n	$h(n)$	n	$h(n)$
0	921	0	$17\times 2^6 - 5\times 2^5 - 7\times 2^0$	13	$\mathbf{0}$	26	-9×2^0
1	841	176	$5\times 2^7 + 7\times 2^2 - 3\times 2^0$	14	$\mathbf{-11\times 2^2}$	27	1×2^0
2	626	236	$3\times 2^7 + 3\times 2^1$	15	$\mathbf{-1\times 2^6 + 5\times 2^0}$	28	7×2^0
3	343	176	$5\times 2^5 + 7\times 2^0$	16	$\mathbf{-11\times 2^2}$	29	1×2^3
4	74	52	11×2^1	17	$\mathbf{-13\times 2^0}$	30	3×2^1
5	-112	-68	-11×2^2	18	$\mathbf{17\times 2^0}$	31	3×2^0
6	-184	-128	-7×2^3	19	$\mathbf{1\times 2^5}$		
7	-152	-112	-5×2^3	20	$\mathbf{7\times 2^2}$		
8	-62	-44	-9×2^1	21	$\mathbf{11\times 2^0}$		
9	34	36	-1×2^1	22	$\mathbf{-9\times 2^0}$		
10	92	92	0	23	$\mathbf{-3\times 2^3 + 1\times 2^0}$		
11	96	104	-1×2^3	24	$\mathbf{-13\times 2^1}$		
12	56	80	-3×2^3	25	$\mathbf{-5\times 2^2}$		

It is noted that, in this example, the center coefficient is included in a lobe extrapolated from the prototype lobe. To avoid the extrapolation of the center coefficient being added twice in the implementation where coefficient symmetry is exploited, a modified structure, illustrated by a 10th-order example, is shown in Fig. 6, where coefficients $h(0)$ to $h(2)$ are extrapolated from $h(3)$ to $h(5)$. In this structure, $\alpha_0 h(3)$ is added into the delay chain only once.

The subexpression space for the design is constructed on a basis set of order 7

$$SS = \{\pm 1, \pm 3, \pm 5, \pm 7, \pm 9, \pm 11, \pm 13, \pm 17\}. \tag{14}$$

The resulting filter coefficients are listed in Table III. It can be seen that residuals $h_r(0)$ and $h_r(1)$ require 2 adders to construct each. However, since they both use the coefficient 167 $(5\times 2^5 + 7\times 2^0)$ which is already synthesized in $h_r(3)$, only one adder each is used in the synthesis of $h_r(0)$ and $h_r(1)$. Furthermore, it is noted that $h_r(10)$ and $h(13)$ are 0 which contributes to 4 adders reduction in the structural adders. As listed in Table IV, both the MB adders and structural adders are reduced by 4 in comparing with the best result obtained in literature [16]. The overall number of adders reduced is 6 after the 2 extrapolation adders are compensated. It is also noted from Table III that the effective wordlength of the filter coefficient is 10 which is one bit less than that in [16].

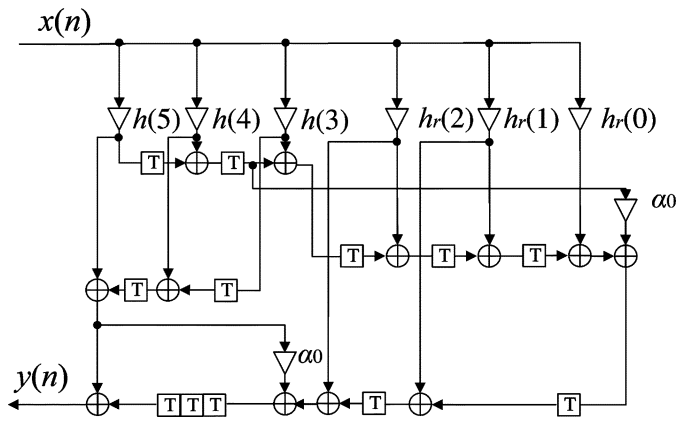


Fig. 6. Modified extrapolated impulse response structure with the center coefficient extrapolation.

B. *Example* $L1$

Example $L1$ is also taken from [17]. The filter specification is as follows: a highpass filter with order $N = 120$ has stopband and passband edges at $\omega_s = 0.74\pi$ and $\omega_p = 0.8\pi$, respectively; the stopband and passband ripples are $\delta_s = 0.0001$ and $\delta_p = 0.0057$ while the effective wordlength (excluding the sign bit) is 14.

Again, by inspecting the optimal continuous coefficient values, it is noted that the filter impulse response shows a quasi-periodicity for five lobes, from $h(1)$ to $h(45)$, with each lobe consisting of nine coefficients. The lobe with the minimum coefficient magnitude, i.e., lobe_4 (from $h(37)$ to $h(45)$) is chosen as the prototype lobe to approximate the other 4 lobes. The scaling factors for lobe_3 ($h(28)$ to $h(36)$), lobe_2 ($h(19)$ to $h(27)$), lobe_1 ($h(10)$ to $h(18)$) and lobe_0 ($h(1)$ to $h(9)$) are chosen to be $-2, 4, -8$, and 16, respectively. Expressed mathematically, we have

$$\begin{aligned} h_a(1+k) &= 16h(37+k) \\ h_a(10+k) &= -8h(37+k) \\ h_a(19+k) &= 4h(37+k) \\ h_a(28+k) &= -2h(37+k), \quad \text{for } k = 0, 1, \ldots, 8. \end{aligned} \tag{15}$$

The other coefficient values are synthesized directly. So, the coefficients to be optimized in this problem are $h(n)$ for $n = 0$ and $n = 37, 38, \ldots, 60$ and $h_r(n)$ for $n = 1, 2, \ldots, 36$.

The multi-step design approach proposed in [16] is employed to accelerate the optimization procedure, as well as to reuse the subexpression bases. The filter coefficients are split into three groups

Group A:$h_r(28)$ to $h_r(36)$ and $h(37)$ to $h(60)$
Group B:$h_r(8)$ to $h_r(27)$
Group C:$h(0)$ and $h_r(1)$ to $h_r(7)$.

In the course of the optimization, the residuals and coefficients in Group A are first optimized in a subexpression space

TABLE IV
NUMBER OF ADDERS USED TO IMPLEMENT THE FILTERS

	Proposed / Best Published References						
Filters	$L2$ / [16]	$L1$ / [16]	$L3$ / [16]	$S1$ / [16]	$S2$ / [16]	$Y1$ / [13] / [15]	$G1$ / [14]
Filter Order	62	120	35	24	59	29	15
Coefficient Effective Wordlength	10 / 11	14 / 14	6 / 8	7 / 8	10 / 10	10 / 10 / 12	6 / 8
MB Adders	13 / 17	30 / 44	5 / 3	4 / 4	18 / 19	3 / 7 / 9	2 / 2
Structural Adders	58 / 62	112 / 120	27 / 35	16 / 24	53 / 59	21 / 23 / 21	11 / 13
Extrapolation Adders	2 / –	8 / –	4 / –	2 / –	2 / –	4 / – / –	2 / –
Scaling Factor Adders	0 / –	0 / –	0 / –	2 / –	0 / –	0 / – / –	0 / –
Total Adders	**73** / 79	**150** / 164	**36** / 38	**24** / 28	**73** / 78	**28** / 30 / 30	**15** / 15
Delay Chain Length d	13 / –	9 / –	4 / –	5 / –	6 / –	5 / – / –	3 / –
Number of Delay Chain(s) k	2 / –	2 / –	2 / –	1 / –	1 / –	1 / – / –	1 / –
Maximum Adder Depth	4 / 4	4 / 4	4 / 3	3 / 3	4 / 4	3 / 3 / 3	3 / 3
Average Adder Depth	2.27 / 2.42	2.47 / 2.86	1.61 / 1.72	1.31 / 1.69	2.5 / 2.53	1.33 / 1.67 / 1.47	1 / 1.75

constructed by an initial subexpression basis set, while the residuals and coefficients in Group B and Group C are left continuous. Once a set of discrete values are obtained for the coefficients and residuals in Group A, the initial subexpression basis set is expanded as new bases are generated. The residuals in Group B are then optimized in the expanded subexpression space, while the coefficients and residuals in Group A are fixed to the obtained discrete values, and those in Group C are left continuous. The subexpression space is further expanded. At last, the coefficients and residuals in Group C are optimized, while those of Group A and B are fixed to the obtained discrete values.

The initial subexpression basis set SS_1 and the subsequently updated basis sets SS_2 and SS_3 are as follows:

$$\begin{aligned} SS_1 &= \{\pm 1, \pm 3, \pm 5, \pm 7, \pm 9\} \\ SS_2 &= \{\pm 1, \pm 3, \pm 5, \pm 7, \pm 9, \pm 13, \pm 15, \\ &\quad \pm 19, \pm 23, \pm 25, \pm 29, \pm 33, \pm 41, \\ &\quad \pm 63, \pm 73, \pm 135, \pm 157, \pm 197\}. \\ SS_3 &= \{\pm 1, \pm 3, \pm 5, \pm 7, \pm 9, \pm 13, \pm 15, \\ &\quad \pm 19, \pm 23, \pm 25, \pm 29, \pm 33, \pm 41, \\ &\quad \pm 63, \pm 73, \pm 89, \pm 111, \pm 135, \\ &\quad \pm 145, \pm 157, \pm 171, \pm 177, \pm 197\} \end{aligned}$$

i.e., the coefficients and residuals of Groups A–C are optimized based on the subexpression space constructed from SS_1, SS_2, and SS_3, respectively.

The resulting filter coefficients and residuals are listed in Table V. It can be seen that the final basis set of SS_3 requires 22 adders, while additional 8 adders are used to generate all the coefficients in the MB. Besides, eight extrapolation adders are used. However, it is noted in Table V that the values of residuals $h_r(24)$, $h_r(32)$, $h_r(34)$, and coefficient $h(45)$ are 0, which contributes to 8 adders reduction in the structural adders. The number of different type adders are listed in Table IV. Thus the total number of adders used to construct the filter using the proposed technique is 14 less than obtained in [16] in which this filter is optimized in transposed direct form. It should also be noted that, compared to the result obtained in Section III, where the extrapolated filter is synthesized based on a given set of discrete coefficient values, 13 more adders are reduced by optimizing the filter coefficients and residuals directly in subexpression space.

C. Five More Benchmark Filters

The proposed technique is further applied on five design examples taken from literature. The multiplier block adders, structural adders and total number of adders used to implement the filters are compared with those of the best available designs. Filters $S1$, $S2$, and $L3$ are taken from [16]. $Y1$ is the first example of [13]. $G1$ is given as example 1 in [14]. As listed in Table IV, our proposed technique synthesizes the most of the filters using less number of adders. It is noted that for filters of order as low as 24, the proposed technique may further reduce the required number of adders. The reduction is achieved due to the fact that many coefficient values and residuals are reduced to 0 because of the extrapolation; zero save the structural adders. However, for extremely low order and short wordlength filter as $G1$, no further reduction is achieved since the number of adders used in the original design is already very small.

In the implementation of the proposed filters, extra delay chains are used. The number of delay chains required and the length of each delay chain for each design are also shown in Table IV. The complexity overhead due to the delay chains is implementation dependent. A realization of the delay chain is proposed in the following section, while the net saving in the implementation is discussed in Section VII.

VI. A REALIZATION OF THE EXTRA DELAY ELEMENTS

A realization of the extra delay chain required in the implementation of the proposed extrapolated filter is introduced in this section. Assume that the data width is B-bits. A delay chain of length d may be realized by using 2 pointers and $d+1$ storage units, as shown in Fig. 7. At any clock cycle, a new data is written into the storage unit indicated by pointer W, while the data in storage unit indicated by R is read out for arithmetic operation. The pointers are circulated in the direction indicated by the solid and dashed arrows in Fig. 7 over the storage units along the clock cycle. The circuit diagram of the delay chain with $d = 3$ is shown in Fig. 8. The circuit consists of two parts: a control logic and $d + 1$ storage units. The control logic is a shifting register constructed by $d + 1$ 1-bit D flip-flops (DFF) connected in a circulative manner, whereas each storage unit

TABLE V
IMPULSE RESPONSE OF EXAMPLE $L1$. COEFFICIENTS FROM $h(37)$ TO $h(45)$ ARE CHOSEN AS THE PROTOTYPE LOBE. RESIDUALS FROM $h_r(1)$ TO $h_r(36)$, COEFFICIENTS $h(0)$ AND $h(37)$ TO $h(60)$ ARE REALIZED IN THE MULTIPLIER BLOCK. $h(n-ld) = h_r(n-ld) + \alpha_{L-1-l}h(n)$ FOR $n = 37, 38, \ldots, 45$, $l = 1, 2, 3, 4$, $L = 5$ AND $d = 9$, WHERE $\alpha_0 = 16, \alpha_1 = -8, \alpha_2 = 4$ AND $\alpha_3 = -2$

n	$h(n)$	$h_a(n)$	$h_r(n)$	n	$h(n)$	$h_a(n)$	$h_r(n)$	n	$h(n)$
0	14672		$111 \times 2^7 + 29 \times 2^4$	19	-619	-628	9×2^0	37	$\mathbf{-157 \times 2^0}$
1	-13481	-2512	$-43 \times 2^8 + 1 \times 2^5 + 7 \times 2^0$	20	798	788	5×2^1	38	$\mathbf{197 \times 2^0}$
2	10256	3152	111×2^6	21	-599	-576	-23×2^0	39	$\mathbf{-9 \times 2^4}$
3	-5920	-2304	$-111 \times 2^5 + 1 \times 2^6$	22	163	144	19×2^0	40	$\mathbf{9 \times 2^2}$
4	1651	576	$33 \times 2^5 + 19 \times 2^0$	23	283	292	-9×2^0	41	$\mathbf{73 \times 2^0}$
5	1512	1168	$9 \times 2^5 + 7 \times 2^3$	24	-540	-540	0	42	$\mathbf{-135 \times 2^0}$
6	-2991	-2160	$-63 \times 2^4 + 177 \times 2^0$	25	521	528	-7×2^0	43	$\mathbf{33 \times 2^2}$
7	2810	2112	$177 \times 2^2 - 5 \times 2^1$	26	-271	-304	33×2^0	44	$\mathbf{-19 \times 2^2}$
8	-1506	-1216	-145×2^1	27	-66	0	-33×2^1	45	$\mathbf{0}$
9	-126	0	-63×2^1	28	330	314	1×2^4	46	15×2^2
10	1352	1256	3×2^5	29	-414	-394	-5×2^2	47	-41×2^1
11	-1753	-1576	-177×2^0	30	306	288	9×2^1	48	63×2^0
12	1323	1152	171×2^0	31	-81	-72	-9×2^0	49	-19×2^0
13	-399	-288	-111×2^0	32	-146	-146	0	50	-7×2^2
14	-538	-584	23×2^1	33	275	270	5×2^0	51	29×2^1
15	1083	1080	3×2^0	34	-264	-264	0	52	-63×2^0
16	-1064	-1056	-1×2^3	35	139	152	-13×2^0	53	23×2^1
17	579	608	-29×2^0	36	28	0	7×2^2	54	-9×2^1
18	89	0	89×2^0					55	-1×2^3
								56	25×2^0
								57	-29×2^0
								58	3×2^3
								59	-13×2^0
								60	7×2^0

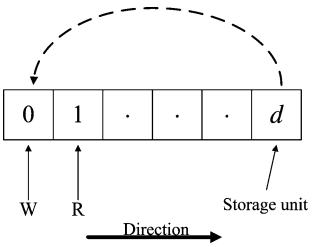


Fig. 7. Illustrative of the delay chain.

TABLE VI
AVERAGE POWER OF B-BIT DELAY CHAIN, FULL ADDER, AND TAP DELAY ELEMENT

	power (E)
P_1 = An adder	$2.5B$
P_2 = A delay Chain	$3 + 2.125B$
P_3 = Two delay chains	$3 + 3.125B$
P_4 = A tap delay	$1.75B$

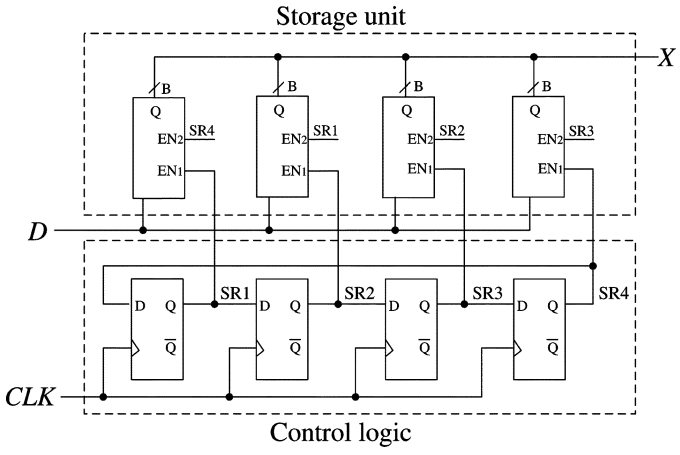


Fig. 8. Circuit diagram of a delay chain.

consists of B-bit D latches connected by the enable signal EN_1. The shifting registers are initialized to have one output HIGH, while the others LOW. The HIGH output is circulated to enable one of the storage unit to be written, and at the same time select another storage unit to be read out by controlling the enable signals EN_1 and EN_2 in the storage units at any clock cycle. During the operation of the delay chain, at any clock cycle, only two storage units and two DFFs may change states. Therefore, the overall power consumption incurred due to the delay chain is low. When two delay chains are used in the extrapolated filter, the control logic of the shifting register may be shared by the two chains.

A detailed analysis and estimation of the power consumption incurred due to the delay chain(s) in comparison with that of full adders and tap delay elements are given in the Appendix; the results are listed in Table VI, where the data width is B-bits. E is a unit of power defined in the Appendix. It is noted that the power consumed by the delay chain(s) is independent of the length of the delay chain. The increased power due to one delay chain and two delay chains, given by $P_2 - P_4$ and $P_3 - 2P_4$ is less than one sixth of the power consumed by an adder with width B, respectively, where P_2, P_3, and P_4 are given in Table VI.

Furthermore, in the estimation of the power consumption, it was assumed that there is no spurious switching in the operation of the circuit. In practice, however, full adders experience substantial spurious switching [25], [26], whereas the delay chains constructed from D-latches and DFFs do not result in much of such switching. Therefore, the overall power consumed due to the implementation of the extra delay chain(s) is negligible in comparing with the saving achieved due to the reduction of adders.

TABLE VII
COMPLEXITY COMPARISON OF DELAY CHAINS AND FULL ADDERS. THE EXTERNAL SIGNAL WORDLENGTH IS ASSUMED TO BE 12 BITS

Filters	$L2$	$L1$	$L3$	$S1$	$S2$	$Y1$	$G1$
Reduction in Number of Adders	6	14	2	4	5	2	0
Intermediate Result Wordlength B	22	26	18	19	22	22	18
Reduction in Number of Transistors for Adders	3696	10192	1008	2128	3080	1232	0
Number of Extra Transistors in Delay Chains	2600	1928	404	362	530	404	136
Reduction in Total Number of Transistors	1096	8264	604	1766	2550	828	-136
Reduction in Equivalent Number of Adders	1.8	11.4	1.2	3.3	4.1	1.3	-0.3

VII. DISCUSSION

It has been shown in Section V that the proposed technique synthesizes filters using less adders but employing extra delay elements. To give a sense on the reduction of implementation complexity, in this section, a transistor level comparison is given based on the realization proposed in Section VI and the Appendix. The equivalent number of adders saved is thus obtained. In addition to the implementation complexity, the adder depth of the filters designed using the proposed technique is investigated and the applicability of the proposed technique is discussed.

A. Implementation Complexity

A simplified comparison is given in this subsection by counting the number of transistors used in the adders and delay chains in the implementation. Assume that the number of adders reduced is A and each adder is of B-bit. Then the number of transistors reduced is equal to $28AB$ (assume that the conventional 28-transistor full adder given in the Appendix is used). On the other hand, the number of transistors used to implement one delay chain and two delay chains of length d are $(d+1)(5B+16)-16B$ and $(d+1)(10B+16)-32B$, respectively. Thus, the number of transistors reduced due to the reduced number of adders and the number of transistors increased due to the extra delay chain are shown in Table VII for each design presented in Section V. In Table VII, the wordlength of the intermediate result, B, is assumed to be the sum of the wordlength of the external signals and the wordlength of the filter coefficients. Assumed also that the wordlength of the external signal is 12 bits. The equivalent number of adders saved after taking the additional delay chains into consideration is listed in the last row of Table VII.

From Table VII, it is shown that the proposed technique reduces the implementation complexity in most cases. The reduction amount, varying from a few adders to more than 10 adders, depends on many factors such as the filter length, coefficient effective wordlength, the number and length of the delay chains. For very short filters with short coefficient wordlength, for instance $G1$ with filter order 15 and coefficient effective wordlength 6, the net saving in implementation is negative; this is because the MB in such a case is small and the saving in MB adders and structural adders are canceled by the extrapolation adders. In general, in order to achieve implementation complexity reduction, the total number of adders reduced, A, using the proposed algorithm should satisfy

$$A > \frac{5kd}{28} \tag{16}$$

where k is the number of delay chains. In (16), it is assumed that B is larger than $(d+1)$, which in general is true.

B. Adder Depth

Adder depth is defined as the number of adders that a signal goes through from the input to a delay element. It was pointed out in [25], [26] that the power consumption of a MB is often not proportional to the number of adders but is rather very much dependent on the adder depth of every coefficient. More specifically, the average dynamic power of a circuit can be expressed by [27]

$$P_{\text{dyn}} = 0.5V_{\text{DD}}^2 f_c C_L \alpha \tag{17}$$

where V_{DD} is the supply voltage, f_c is the clock frequency, C_L is the load capacitance, and α is the switching activity. All the parameters, except α, are defined by the physical layout and specifications of the circuit. Apparently, larger α would result in more dynamic power consumption. It is pointed out in [25], [26] that in a network of adders, for instance the MB, one of the main contributions to the increase of α is due to an undesirable effect called "glitch". In [25] a simple model for estimating the glitch path among all the coefficients is proposed. However, it is shown in [26] that the model in [25] is not always accurate. Nevertheless, it is in general agreed that larger adder depth indicates more glitches and hence higher power consumption [25], [26].

Listed in Table IV are the maximum adder depth and the average adder depth of each design. It is shown that the filters designed using our proposed technique have the same maximum adder depth in most cases and smaller average adder depth in all cases when compared with the best previously published results. The reduction in the average adder depth is due to the fact that the dynamic range of the residuals is significantly reduced compared with their original coefficients. These results show that the proposed filter structure will not incur any power consumption overhead due to increased adder depth; rather in addition to the power consumption reduced due to the reduced adders, the power consumption may be slightly further reduced because of the smaller average adder depth.

C. Applicability and Limitation

The proposed technique is applicable to the design of any FIR filter since the quasi-periodic property is inherent in the impulse response. While the inaccuracy of the periodicity may degrade the frequency response in the traditional extrapolated filter, the proposed technique is immune to the problem of imperfect periodicity due to the introduction of residual compensation. The

magnitude of the residual might be larger than that of the original coefficient value; such situation, however, does not exist in the design of the current examples, whose specifications vary from lowpass to highpass, narrow band to wide band, and moderate ripple requirements to stringent ones. When the length and location of the prototype lobe is properly selected, the magnitude of residual hardly increases. Even if one or two residuals increase the magnitudes in exceptional examples, the overall dynamic range of coefficient values and residuals resulted from the extrapolation decreases for sure; the proposed optimization technique further ensures that the values with increased magnitudes (if any) are synthesized using not more than the required number of adders.

Therefore, in the proposed technique, what actually matters is if the reduction in the MB adders and structural adders is large enough to compensate for the increase in the extrapolation adders and scaling factor adders. As shown in Section VII-A, when both filter length and coefficient wordlength are short, net implementation complexity reduction is not guaranteed. Furthermore, the saving on the adders may be canceled by the additional delay chain if the delay chain is long. For instance, the proposed design of $L2$ uses six adders less than that in [16] as shown in Table IV. However, the equivalent saving after compensating the delay chains is only 1.8 adders as shown in Table VII due to the long delay chains of length 13. If the implementation complexity is the major concern, a discrete solution using four adders less than that in [16] could be obtained using the proposed technique where a prototype lobe with length five is adopted; in this design, the saving on the number of adders is reduced, but the equivalent saving after compensating for the delay chain increases to 3.2 adders. Nevertheless, the design with delay chain length of 13 achieves more power saving since the power incurred due to the delay chains is independent of the length of delay chains and is negligible. Therefore, depending on the design requirements, complexity priority or power priority design could be chosen by selecting different prototype lobes.

VIII. Conclusion

In this paper, an extrapolated impulse response with residual compensation is proposed for synthesizing linear phase FIR filter in subexpression sharing. The proposed filter structure reduces the dynamic range of the coefficient values to be synthesized and this further reduces the lower bound of the number of adders required to synthesize a filter. Both the filter coefficients and the residuals may be optimized directly in subexpression space. MILP is formulated to optimize the filter coefficients and the residuals. Examples have shown that the number of adders used to synthesize the filter is significantly reduced. The reduction varies from a few adders to more than 10 adders for different designs if the filter order is not lower than 20. Extra delay chains are used in the proposed structure. The power overhead incurred due to the proposed delay chains is negligible, whereas the increase in the implementation complexity is insignificant if the prototype lobe is chosen properly.

Appendix
Average Power Estimation of Delay Chain, Full Adder, and Tap Delay Element

The power consumption of the delay chain described in Section VI as well as the conventional 28-transistor 1-bit full adder [28] and tap delay element is estimated in this section. To simplify the analysis, the first-order estimation is used and the following assumptions are made.

1) Static power consumption is much smaller than dynamic power consumption, and therefore ignored.
2) Gate capacitance is considered as load capacitance. Drain and source capacitances are ignored.
3) All p-transistors used in the full adder, delay chain, and tap delay element adopt the same gate size and therefore have the same gate capacitance C_p, whereas all n-transistors have the same gate capacitance C_n. $(C_p + C_n)$ is considered as a unit load capacitance.
4) The external load of the delay chain, the tap delay element, and the sum output of the full adder is the addend input of the subsequent full adder, whereas the external load of the carry out of the full adder is the carry in input of the subsequent bit of full adder. This assumption is consistent with the filter structure shown in Fig. 5.
5) Input data is random where 0 and 1 have equal probability.

Based on the above assumptions, the power consumed in a clock cycle by a circuit to charge a node with a unit load capacitance (consisting of a p-gate and an n-gate) from 0 to 1 is considered as a unit of power consumption, and denoted as E. $E = E_n + E_p$, where E_n and E_p are the power consumed when capacitance C_p and C_n are charged from 0 to 1, respectively.

A. Delay Chain

Two types of components, D-latch and D-flip-flop, are used in the realization of the delay chain. Consider a 1-bit storage unit consisting of a D-latch. The gate level diagram is shown in Fig. 9. According to the structure in Fig. 8, each storage unit only has three possible states: ST0, ST1, and ST2, as listed in Table IX. In order to estimate the power of the whole circuit in one clock cycle, the voltage change of both internal and external nodes are considered.

In the operation of the circuit, at any clock cycle, one latch transits from ST0 to ST2, one from ST2 to ST1 and one from ST1 to ST0. All the other storage units remain the state ST0 unchanged. The storage units remaining the states or changing the state from ST1 to ST0 consume 0 power, regardless the values of the data D.

When the storage unit changes the state from ST2 to ST1, only node $\bar{Q}$ with capacitance load C_p may be charged from 0 to 1, if the present data D is 0 and the previous data D is 1. When the present and previous data of D are in other combinations, no power is consumed. In average, $0.25E_p$ is consumed.

When the storage unit changes the state from ST0 to ST2, if the node Q, which is the addend input of a full adder, is changed from 0 to 1, $4E$ is consumed. If Q remains unchanged, or changes from 1 to 0, no power is consumed. Therefore, in average, $1E$ is consumed.

TABLE VIII
FIRST-ORDER ESTIMATION OF THE POWER CONSUMED BY THE FULL ADDER IN ALL TRANSITIONS

	Power consumed(E)	Final state ABC_{in}								Average
		000	001	010	011	100	101	110	111	
Initial state ABC_{in}	000	0	4	4	3	4	3	3	7	3.5
	001	1	0	0	4	0	4	4	3	2
	010	1	0	0	4	0	4	4	3	2
	011	2	6	6	0	6	0	0	4	3
	100	1	0	0	4	0	4	4	3	2
	101	2	6	6	0	6	0	0	4	3
	110	2	6	6	0	6	0	0	4	3
	111	3	2	2	1	2	1	1	0	1.5
	Overall									2.5

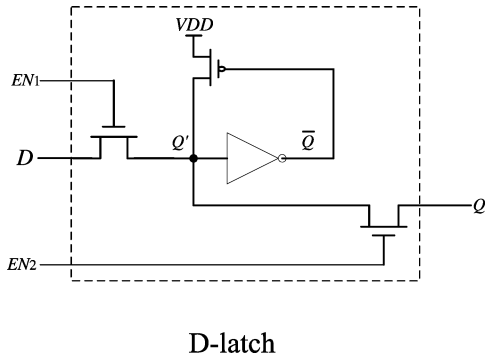


Fig. 9. Circuit diagram of a D-latch.

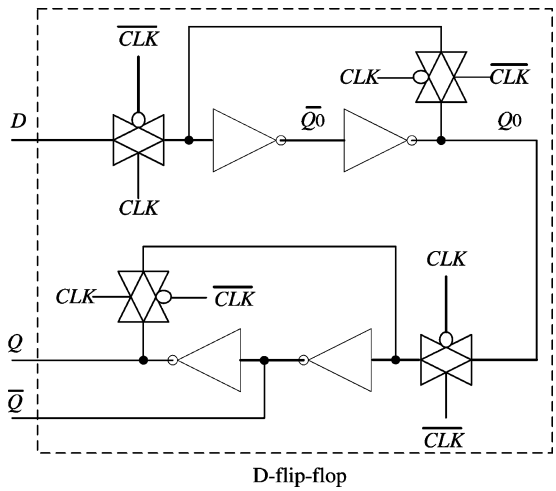


Fig. 10. Circuit diagram of a D-flip-flop.

TABLE IX
STATES OF THE STORAGE UNIT

	EN1	EN2	Remark
ST0	0	0	–
ST1	1	0	A new data is written into the unit
ST2	0	1	The data in the unit is read

From the above analysis, in one clock cycle, $E + 0.25E_p$ is consumed for 1-bit storage unit. A B-bit storage unit, thus consumes power of $p_1 = (0.25E_p + E)B$.

A master-slave DFF is constructed by using 2 D-latches, connected as shown in Fig. 10. For a B-bit delay chain, nodes $\overline{Q_0}, Q_0$ and $\bar{Q}$ consist of 1 unit capacitance each. Since Q is connected to the subsequent DFF as well as the control signals of EN_1 and EN_2 of the storage units, as shown in Figs. 8 and 9, node Q consists of 1 unit capacitance plus $2B$ of capacitance C_n. In the operation of the shift register, at any clock cycle, the data stored in one DFF changes from 0 to 1 and another one from 1 to 0, All the other DFFs keep the values unchanged and thus do not consume any power. The two DFFs, which change states, consume $p_2 = E + 2BE_n$ and $p_3 = 2E$, respectively, for a delay chain with data width B. Thus, the total power consumed by one delay chain in a clock cycle is

$$\begin{aligned} P_2 &= p_1 + p_2 + p_3 \\ &= (E + 0.25E_p)B + (E + 2BE_n) + 2E \\ &= 3E + (1.25E_p + 1.75E_n)B. \end{aligned} \tag{18}$$

Assume that $E_n = 0.5E$, we have $P_2 = (3 + 2.125B)E$. Note that E_n is in general smaller than $0.5E$ since C_n is generally smaller than C_p.

When two delay chains are employed, node Q in Fig. 10 consists of 1 unit capacitance plus $4B$ of capacitance C_n. Thus, the power consumed by the shift register for two delay chains is (3+3.125B)E. The overall power consumed by the delay chains are listed in Table VI.

B. Tap Delay

Tap delays are realized by DFF too. For each bit of tap delay, $\overline{Q_0}, Q_0, \bar{Q}$ and Q consist of 1, 1, 1, and 4 unit capacitance, respectively. In one clock cycle, if the data in the DFF remains unchange, no power is consumed; if the data changes from 0 to 1, $5E$ is consumed; and if the data changes from 1 to 0, $2E$ is consumed instead. In average, 1.75E is consumed for 1-bit tap delay. The value for tap delay listed in Table VI is the power consumption of a B-bit tap delay.

C. Full Adder

The conventional CMOS 1-bit full adder has three inputs (A, B, C_{in}), two outputs (S, C_{out}). Table VIII lists all possible state transitions of a full adder and the corresponding power consumption where the transitions of both internal and external nodes are considered. It is shown that in average, a 1-bit full adder consumes power of $2.5E$. The value listed in Table VI is a B-bit adder.

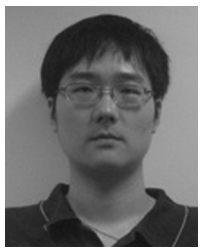
REFERENCES

[1] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 43, no. 5, pp. 677–688, Oct. 1996.

[2] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 42, no. 6, pp. 569–577, Sep. 1995.
[3] O. Gustafsson and L. Wanhammar, "Improved multiple constant multiplication using minimum spanning trees," in *Proc. Asilomar Conf. Sigals, Syst., Comp.*, 2004, pp. 63–66.
[4] M. Aktan, A. Yurdakul, and G. Dundar, "An algorithm for the design of low-power hardware-efficient FIR filters," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 7, pp. 1536–1545, Jul. 2008.
[5] F. Xu, C.-H. Chang, and C.-C. Jong, "Contention resolution algorithm for common subexpression elimination in digital filter design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 10, pp. 695–700, Oct. 2005.
[6] I.-C. Park and H.-J. Kang, "Digital filter synthesis based on an algorithm to generate all minimal signed digit representation," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 21, no. 12, pp. 1525–1529, Dec. 2002.
[7] D. R. Bull and D. H. Horrocks, "Primitive operator digital filters," *IEE Proc. G*, vol. 138, pp. 401–412, Jun. 1991.
[8] N. Sankarayya and K. Roy, "Algorithms for lower power and high speed FIR filter realization using differential coefficients," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 43, no. 6, pp. 488–497, Jun. 1997.
[9] O. Gustafsson, "Lower bounds for constant multiplication problems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 11, pp. 974–978, Nov. 2007.
[10] C.-Y. Yao, H.-H. Chen, T.-F. Lin, C.-J. Chien, and C.-T. Hsu, "A novel common-subexpression-elimination method for synthesizing fixed-point FIR filters," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 11, pp. 2215–2221, Nov. 2004.
[11] O. Gustafsson, "A difference based adder graph heuristic for multiple constant multiplication problems," in *Proc. IEEE ISCAS*, New Orleans, LA, 2007, pp. 1097–1100.
[12] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Trans. Algorithms*, vol. 3, pp. 1–38, May 2007.
[13] J. Yli-Kaakinen and T. Saramäki, "A systematic algorithm for the design of multiplierless FIR filters," in *Proc. IEEE ISCAS*, Sydney, Australia, 2001, pp. 185–188.
[14] O. Gustafsson, H. Ohlsson, and L. Wanhammar, "Design of linear-phase FIR filters combining subexpression sharing with MILP," in *Proc. MWSCAS*, 2002, pp. 9–12.
[15] F. Xu, C.-H. Chang, and C.-C. Jong, "Design of low-complexity FIR filters based on signed-powers-of-two coefficients with reusable common subexpressions," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 10, pp. 1898–1907, Oct. 2007.
[16] Y. J. Yu and Y. C. Lim, "Design of linear phase FIR filters in subexpression space using mixed integer linear programming," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 10, pp. 2330–2338, Oct. 2007.
[17] Y. C. Lim and S. R. Parker, "Discrete coefficient FIR digital filter design based upon an LMS criteria," *IEEE Trans. Circuits Syst.*, vol. 30, no. 5, pp. 723–739, Oct. 1983.
[18] Y. J. Yu, D. Shi, and Y. C. Lim, "Subexpression encoded extrapolated impulse response FIR filter with perfect residual compensation," in *Proc. IEEE ISCAS*, 2008, pp. 2446–2449.
[19] Y. C. Lim and B. Liu, "Extrapolated impulse response FIR filters," *IEEE Trans. Circuits Syst.*, vol. 37, no. 12, pp. 1548–1551, Dec. 1990.
[20] Y. J. Yu, K. L. Teo, Y. C. Lim, and G. H. Zhao, "Extrapolated impulse response filter and its application in the synthesis of digital filters using the frequency-response masking technique," *Signal Process. (Elsevier)*, vol. 85, no. 3, pp. 581–590, Mar. 2005.
[21] L. Zhou, W. Pei, and Z. He, "Extrapolated impulse response filter design utilizing principal components analysis," *Electron. Lett.*, vol. 43, pp. 483–484, 2007.
[22] Y. C. Lim, "Design of discrete-coefficient-value linear phase FIR filters with optimum normalized peak ripple magnitude," *IEEE Trans. Circuits Syst.*, vol. 37, no. 12, pp. 1480–1486, Dec. 1990.
[23] Y. C. Lim, Y. Sun, and Y. J. Yu, "Design of discrete-coefficient FIR filters on loosely connected parallel machines," *IEEE Trans. Signal Process.*, vol. 50, no. 6, pp. 1409–1416, Jun. 2002.
[24] Y. C. Lim and S. R. Parker, "FIR filter design over a discrete power-of-two coefficient space," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-31, no. 3, pp. 583–591, Jun. 1983.
[25] S. S. Demirsoy, A. G. Dempster, and I. Kale, "Transition analysis on FPGA for multiplier-block based fir filter struc," in *Proc. IEEE ICECS*, Dec. 2000, vol. 2, pp. 17–12.
[26] K. Johansson, O. Gustafsson, and L. Wanhammar, "Switching activity estimation for shift-and-add based constant multipliers," in *Proc. IEEE ISCAS*, 2008, pp. 676–679.
[27] F. Najm, "A survey of power estimation techniques in VLSI circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 2, no. 6, pp. 446–455, Dec. 1994.
[28] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Reading, MA: Addison-Wesley, 1993.

Ya Jun Yu (S'99–M'05–SM'09) received the B.Sc. and M.Eng. degrees in biomedical engineering from Zhejiang University, Hangzhou, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2004.

Since 2005, she has been with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, where she is currently an Assistant Professor. From 1997 to 1998, she was a Teaching Assistant with Zhejiang University. She joined the Department of Electrical and Computer Engineering, National University of Singapore as a Post Master Fellow in 1998 and remained in the same department as a Research Engineer until 2004. In 2002, she was a Visiting Researcher at the Tampere University of Technology, Tampere, Finland, and the Hong Kong Polytechnic University, Hong Kong, China. She joined the Temasek Laboratories at Nanyang Technological University as a Research Fellow in 2004. Her research interests include digital signal processing and VLSI circuits and systems design.



Dong Shi (S'07) received the B.Eng. degree in electronic engineering from Shanghai Jiaotong University, Shanghai, China, in 2006. He is currently working towards the Ph.D. degree at Nanyang Technological University, Singapore.

His current research interests include low power and low complexity digital filters design and implementation.

Yong Ching Lim (S'79–M'82–SM'92–F'00) received the A.C.G.I., B.Sc., D.I.C., and Ph.D. degrees in electrical engineering from Imperial College, University of London, U.K., in 1977, 1977, 1980, and 1980, respectively.

Since 2003, he has been with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, where he is currently a Professor. From 1980 to 1982, he was a National Research Council Research Associate in the Naval Postgraduate School, Monterey, CA. From 1982 to 2003, he was with the Department of Electrical Engineering, National University of Singapore. His research interests include digital signal processing and VLSI circuits and systems design.

Dr. Lim was a recipient of the 1996 IEEE Circuits and Systems Society's Guillemin-Cauer Award, the 1990 IREE (Australia) Norman Hayes Memorial Award, 1977 IEE (U.K.) Prize and the 1974–77 Siemens Memorial (Imperial College) Award. He served as a lecturer for the IEEE Circuits and Systems Society under the distinguished lecturer program from 2001 to 2002 and as an Associate Editor for the IEEE Transactions on Circuits and Systems from 1991 to 1993 and from 1999 to 2001. He has also served as an Associate Editor for Circuits, Systems and Signal Processing from 1993 to 2000. He served as the Chairman of the DSP Technical Committee of the IEEE Circuits and Systems Society from 1998 to 2000. He served in the Technical Program Committee's DSP Track as the Chairman in IEEE ISCAS'97 and IEEE ISCAS'00 and as a Co-chairman in IEEE ISCAS'99. He is the General Chairman for IEEE APCCAS 2006 and a Co-General Chairman for IEEE ISCAS 2009. He is a member of Eta Kappa Nu.