PAIRING AND ORDERING TO REDUCE HARDWARE COMPLEXITY IN CASCADE FORM FILTER DESIGN

Hyeong-Ju Kang and In-Cheol Park

Division of EE, Dept. of EECS
Korea Advanced Institute of Science and Technology
373-1, Guseong-dong, Yuseong-gu, Daejeon, Korea

ABSTRACT

This paper presents an algorithm that explores all the combinations of sub-modules in the cascade form filter to reduce hardware complexity under design constraints. Though the cascade form structure has freedom in pairing and ordering of its sub-modules, the hardware complexity is subject to the pairing and ordering if the optimization based on the multiplier block concept is applied. The proposed algorithm selects the pairing and ordering that results in the minimal hardware complexity among all the cases that satisfy the frequency response specification. To cope with the case that the objective filter has many taps and the exploration time is too long, a clustering method is also developed. Experimental results on several filters show that the proposed algorithm reduces the hardware complexity by about 10% on the average, while satisfying the filter specification.

1. INTRODUCTION

Digital filters are frequently used in digital signal processing. In applications demanding high throughput and low power, application specific filters are frequently adopted to meet the constraints of performance and power consumption. In the digital filter implementation, the direct form structure and the cascade form structure are usually used by virtue of simplicity. The direct form structure has less hardware complexity, and the cascade form structure is robust to the quantization and roundoff noise [1][2]. The hardware complexity of the digital filters can be reduced by using the concept of multiplier blocks [3]-[5]. In such approaches, all the coefficient multiplications are decomposed and considered as a whole to construct a hardware block called a multiplier block that implements all the coefficient multiplications. In a multiplier block, the adders used in one multiplication can be shared with other multiplications.

In the cascade form structure, a variety of theoretically equivalent implementations can be obtained by simply pairing the poles and zeros and ordering the sections in different ways. In the previous works, a number of factors that affect the functionality, such as coefficient quantization, roundoff noise, and scaling, are considered to determine the pairing and ordering [1] [2] [6]-[10]. However, they do not consider the hardware complexity because the pairing and ordering does not affect the complexity of filters. Although the assumption is valid for the case that the filters are implemented with multipliers, the pairing and ordering has a significant effect on the hardware complexity if the concept of multiplier block is employed.

In this paper, a pairing and ordering algorithm is suggested for the cascade form structure, which considers the hardware complexity in implementing multiplier blocks. Given a filter equation, the proposed algorithm divides it into second-order polynomials and determines a set of second-order polynomials to be implemented in a multiplier block. When there are too many second-order polynomials, the polynomials are clustered into medium-sized groups, and the optimum ordering and pairing is applied to each group.

This paper is organized as follows. In Section 2, the direct form structure and the cascade form structure are explained, and in Section 3, the concept of a multiplier block is explained. Section 4 describes the proposed algorithm in detail. Experimental results are shown in Section 5, and concluding remarks are made in Section 6.

2. FILTER REALIZATIONS

In IIR filter design, the cascade form structure is preferred to the direct form structure. In fact, the cascade form structure requires more hardware complexity than the direct form structure. The cascade form structure, however, has several advantages that compensate the hardware overhead. One of them is that it is generally much less sensitive to coefficient quantization [1]. Another advantage is that its input, x[n], and output, y[n], have less number of fan-outs, leading to a significant profit in the FIR filter system that has dozens of taps.

In implementing the cascade form structure, we have to solve two problems, pairing and ordering. The pairing deals with how to pair a second-order denominator polynomial and a second-order numerator polynomial to form a second-order section. The ordering decides the order of the second-order sections. In the previous works, many algorithms have been proposed for the problems to improve the roundoff noise [1][6]-[10]. An efficient method is presented in [1] as a rule of thumb. The second-order denominator polynomial whose poles are closer to the unit circle should be paired with the second-order numerator polynomial whose zeros are closer to those poles, and the second-order sections should be ordered such that the second-order section whose poles are closer to the unit circle is located closer (or farther) to the filter-input.

3. MULTIPLIER BLOCK

If the coefficients of a filter are constant, each constant multiplication can be decomposed into addition, subtraction, and

0-7803-7761-3/03/$17.00 ©2003 IEEE

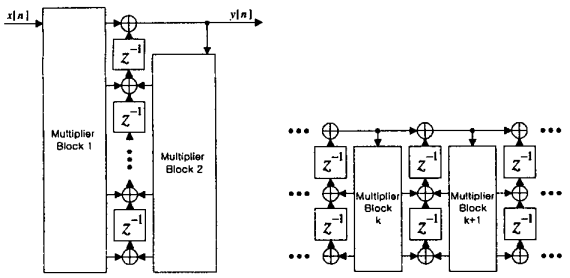

Figure 1. Multiplier blocks (a) in the transposed direct form structure and (b) in the transposed cascade form structure.

shift. The complexity of filters in this case is dominated by the number of additions/subtractions used to implement the coefficient multiplications as the number is proportional to the number of two-input adders and the shifting can be implemented by wire connections. To reduce the complexity, the coefficients can be restricted to powers-of-two or expressed in canonical signed-digit (CSD) representation. However, there is another approach in which coefficient multiplications are considered as a whole. The hardware block called a multiplier block as in Figure 1 is used to implement all coefficient multiplications. Exploiting the concept of a multiplier block enables the additions/subtractions used in one multiplication to be used in other multiplications. Therefore, it can reduce the number of additions/subtractions. Many algorithms have been proposed to make the multiplier block as simple as possible [3-5] In this paper, BHM algorithm [3] is used because it is fast and produces the minimal multiplier block among the algorithms.

4. PAIRING AND ORDERING ALGORITHM

If the concept of multiplier blocks is used, the pairing and ordering affects the hardware complexity of cascade form filters. By changing the pairing and ordering, each multiplier block has a different set of coefficients, and thus it can be implemented with different hardware complexity. In this section, we present the proposed algorithm with considering IIR filters. The algorithm, however, can be easily expanded for FIR filters by eliminating the denominator part. The proposed algorithm assumes that a filter is designed with infinite-precision coefficients and the numerator and the denominator polynomials are factorized into first- or second-order polynomials.

4.1 Clustering

If the target filter is very complex, it will take very long time to search all the possible structures. Therefore, the large problem should be divided into several clusters. There are two issues related with the clustering. The first one is how many polynomials a cluster can have. Since the optimal pairing and ordering is searched in each cluster, increasing the number of polynomials in a cluster expands the search space and leads to better results. The processing time of a cluster, however, increases exponentially according to the cluster size. Experiments show that a 3-cluster (a cluster that has three

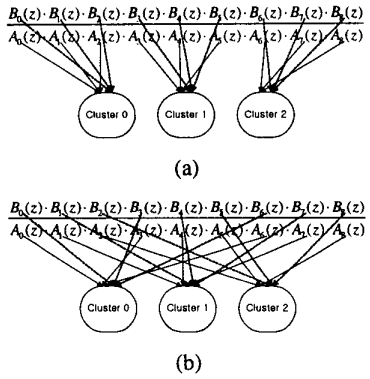

Figure 2. Clustering schemes: (a) Scheme 1 and (b) Scheme 2.

second-order numerator and denominator polynomials) takes about two hours to explore all the structures, and a 4-cluster takes more than two days. The proposed algorithm, therefore, uses 3 or less sized clusters.

The other issue is how the polynomials are clustered. Since the ordering may degrade the frequency response severely, the clustering method must consider the resulting frequency response. In addition, it is better to put the second-order polynomials that are likely to be merged to a forth-order polynomial in a cluster because a forth-order polynomial may have less hardware complexity than two second-order polynomials. The proposed algorithm has two clustering schemes. The first is that the polynomials are ordered as in the conventional method [1] and then the first three second-order sections are grouped into a cluster, the next three sections are grouped, and so on. This clustering scheme is illustrated in Figure 2(a). This method guarantees that the final frequency response is better than that obtained by the conventional ordering. The other scheme is based on the fact that a polynomial with distant roots suffers less frequency response degradation after quantization. In the conventional ordering, the second-order sections are ordered in order of increasing closeness of the poles to the unit circle or in order of decreasing closeness to the unit circle [1]. The second-order polynomials whose roots are close to each other, therefore, may neighbor with each other. Assigning those polynomials to different clusters can increase the possibility of forth-order polynomials. If N polynomials in the conventional order are to be assigned to M clusters, the first M polynomials are assigned to M clusters one by one, and the next M polynomials are assigned one by one, and so on, as shown in Figure 2(b). As it is not clear which method is better, experimental results will be compared later.

4.2 Scaling and Coefficient Rounding

Many works have treated how to scale each section in filters. The proposed algorithm is based on the scaling method used in [6]. The proposed algorithm uses different rounding schemes according to the order of polynomials. Each coefficient of the second-order polynomials is rounded to the nearest integer. For the forth-order polynomials, the algorithm selects optimal integer values from a small search space. After rounding each coefficient to the nearest integer, it compares the frequency responses of the

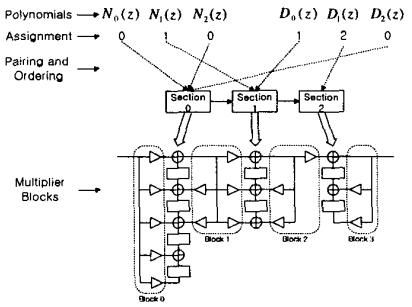

Figure 3. Pairing and ordering.

polynomials obtained by changing the integer values by 0~2. Then it selects the polynomial that has the smallest deviation from the frequency response of the original infinite-precision polynomial. It is not allowed to use more-than-forth-order polynomials for denominators because they degrade the frequency response severely. If higher-order polynomials are used for numerators, each coefficient is rounded to the nearest integer.

4.3 Pairing and Ordering

In a cluster, there are many choices of pairing and ordering. The original pairing problem means only how to pair a denominator polynomial and a numerator polynomial. In this paper, however, the pairing problem also includes how to combine denominator polynomials or numerator polynomials into higher-order polynomials. Combining lower-order polynomials into a higher-order one may reduce the hardware complexity but may deteriorate the frequency response. It is also important to search the best ordering of denominator polynomials and numerator polynomials. The ordering has a significant effect on the hardware complexity and the frequency response.

The cluster located closer to the filter input is searched earlier. Initially, a cluster has as many sections as its second-order denominator polynomials, and each section has its own number. One of the numbers is assigned to each polynomial, as shown in Figure 3. Polynomials are paired and ordered according to those section numbers. The polynomials assigned to the same section are merged to a higher-order polynomial. The proposed algorithm explores all the possible assignments. It can be done in reasonable time since the filter is clustered considering the execution time. It is prohibited to make a more-than-forth-order denominator polynomial because the polynomial degrades the frequency response severely. For each assignment, multiplier blocks are constructed and optimized. This procedure is illustrated in Figure 3, where 3 numerator polynomials and 3 denominator polynomials are paired and ordered to construct a structure. As an example, section number 0, 1, and 0 are assigned to numerator polynomials and section number 1, 2, and 0 are assigned to denominator polynomials. Then the first and third numerator polynomials are inserted into Section 0, the second numerator polynomial into Section 1, the first denominator polynomial into Section 0, and so on. The first and third numerator polynomials are merged into a forth-order polynomial. From this structure, 4 multiplier blocks are constructed.

For each pairing and ordering case, the entire frequency response is estimated and checked whether it satisfies the filter specifications. Since the pairings and orderings of the earlier clusters are already determined, their frequency responses can be obtained. The pairings and orderings of the later clusters, however, are not determined, and thus their frequency responses should be estimated. The proposed algorithm estimates the responses with assuming that the pairings and orderings of those clusters are subject to the conventional method [1].

In the proposed algorithm, the frequency response must satisfy three specifications: passband ripple, stopband ripple, and roundoff noise variance. If not, the pairing and ordering is rejected. The algorithm compares the areas of the structures that satisfy the specifications and selects the pairing and ordering that produces the minimal area.

5. EXPERIMENTAL RESULTS

The proposed algorithm has been applied to a set of IIR filters. The specifications of the filters are shown in Table Ⅰ, where N is the number of taps, Rp and Rs are the ripples in the passband and stopband, and Wn is the normalized cut-off frequency. All the IIR filters are elliptic filters that are designed with infinite-precision coefficients by using MATLAB. The proposed algorithm is performed assuming that the coefficients are to be quantized to 11-bit fixed-point values.

The results are summarized in Table Ⅱ. The column of *Cascade* shows the results obtained by using the cascade form structure, and *Direct* is obtained by using the direct form structure. The last two columns are for the proposed algorithm, where *Proposed1* shows the results obtained with the first clustering scheme, and *Proposed2* shows the results with the second clustering scheme. In Table Ⅱ, the resulting ripples in the passband and stopband are denoted as Rp and Rs, respectively, and the roundoff noise normalized with respect to the rounding noise variance $\sigma^2 = 2^{-2B}/12$ is Ri. The resulting area is estimated as a weighted sum of the number of adders and the number of registers. Given a filter specification, the proposed algorithm selects the minimal-area structure under the condition that its passband ripple, stopband ripple, and roundoff noise can be changed by 0.1dB, 2dB, and 6dB, respectively, with respect to those of the initial cascade form structure.

The direct form structure gives the minimum hardware complexity. As there are more coefficients in a section of the direct form structure, the multiplier block synthesis algorithm can make more adders be shared. However, the direct form structure is not usually used in IIR filter design because it is heavily affected by the coefficient quantization and roundoff noise. Coefficient rounding degrades the frequency response of the direct form structure so severely that the response is far from the original filter specification. The empty columns in Table Ⅱ denote that the frequency response is beyond the specification. On the contrary, the proposed algorithm generates filters whose passband and stopband ripples and roundoff noises are similar to those of the floating-point or the cascade form filters. It means that the proposed algorithm generates feasible filters, while reducing the area by 10%. Compared to the direct form structure that fails to meet the frequency specification, the proposed

Table I Filter Specifications

	N	Rp(dB)	Rs(dB)	Wn		N	Rp(dB)	Rs(dB)	Wn
IIR1	6	1.00	60.0	0.05	IIR7	10	0.67	80.0	0.05
IIR2	6	1.00	60.0	0.10	IIR8	10	0.67	80.0	0.10
IIR3	6	1.00	60.0	0.15	IIR9	10	0.67	80.0	0.15
IIR4	8	0.83	60.0	0.05	IIR10	12	0.50	80.0	0.05
IIR5	8	0.83	60.0	0.10	IIR11	12	0.50	80.0	0.10
IIR6	8	0.83	60.0	0.15	IIR12	12	0.50	80.0	0.15

Table II Results for IIR Filters

	Cascade				Direct				Proposed1				Proposed2			
	Rp	Rs	Ri	Area	Rp	Rs	Ri	Area	Rp	Rs	Ri	Area	Rp	Rs	Ri	Area
IIR1	1.75	60.0	37.4	346	3.26	13.6	64.1	259	1.76	60.1	39.1	304	1.76	60.1	39.1	304
IIR2	1.04	59.9	26.2	297	6.20	54.6	56.0	280	1.06	59.8	26.3	269	1.06	59.8	26.3	269
IIR3	1.03	60.0	20.6	318	1.65	59.5	39.6	238	1.06	60.0	26.5	288	1.06	60.0	26.5	288
IIR4	1.61	60.2	42.9	424				381	1.39	60.1	41.1	403	1.42	60.1	42.8	389
IIR5	0.91	60.0	30.1	431				367	0.91	60.0	28.8	389	0.96	59.8	30.3	382
IIR6	0.93	59.8	23.7	389	4.26	57.4	60.2	353	0.95	59.9	29.6	366	0.88	59.6	27.7	356
IIR7	1.19	78.3	43.8	523				353	1.16	80.2	41.8	495	1.09	78.5	37.7	490
IIR8	0.86	79.7	32.0	537				391	0.94	79.9	29.5	494	0.94	78.9	28.9	460
IIR9	0.84	80.0	26.0	488				384	0.79	79.9	31.7	458	0.82	79.0	31.9	413
IIR10	2.93	81.5	47.3	601				520	2.98	81.4	47.2	580	2.21	80.2	45.6	559
IIR11	0.76	78.6	34.3	608				527	0.84	78.9	31.6	573	0.76	79.2	31.6	552
IIR12	0.83	80.0	27.9	608				478	0.72	80.0	32.1	536	0.68	79.9	30.6	526
Avg.	-	-	-	100%	-	-	-	81.7%	-	-	-	92.3%	-	-	-	89.6%

algorithm enables a trade-off between area and frequency response.

6. CONCLUSION

In this paper, we have proposed a new pairing and ordering algorithm to determine cascade form filter structures. In the previous works, the pairing and ordering problem is considered to improve the frequency response. However, the proposed algorithm explores the pairing and ordering to select a structure that has the least hardware complexity and satisfies the frequency-response specification. The target filter is clustered to reduce the execution time, and the proposed algorithm takes into account higher-order polynomials to broaden the design space. The experimental results show that the proposed algorithm reduces the hardware complexity by 10% on the average, while achieving almost the same frequency response. In other words, the proposed algorithm gives a structure that is closer to the optimal structure.

7. ACKNOWLEDGMENT

This work was supported by the Korea Science and Engineering Foundation through the MICROS center, by the Ministry of Science and Technology and the Ministry of Commerce, Industry, and Energy through the project System IC 2010, and by IC Design Education Center (IDEC).

8. REFERENCES

[1] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, Prentice Hall, 1989.

[2] L. S. DeBrunner, V. DeBrunner, and P. Pinault, "Variable wordlength IIR filter implementations for reduced space designs," in *Proc. IEEE Workshop on Signal Processing Systems*, 2000, pp. 326-335.

[3] A. G. Dempster and M. D. Macleod, "Use of minimum adder multiplier blocks in FIR digital filters," *IEEE Trans. Circuits Syst. II*, vol. 42, pp. 569-577, Sept. 1995.

[4] A. G. Dempster and M. D. Macleod, "IIR digital filter design using minimum adder multiplier blocks," *IEEE Trans. Circuits Syst. II*, vol. 45, pp. 761-763, June 1998.

[5] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Trans. Circuits Syst. II*, vol. 43, pp. 677-688, Oct. 1996.

[6] L. B. Jackson, "On the interaction of roundoff noise and dynamic range in digital filters," *Bell Sys. Tech. J.*, vol. 49, no. 2, Feb. 1970.

[7] L. B. Jackson, "Roundoff-Noise Analysis for Fixed-Point Digital Filters Realized in Cascade or Parallel Form," *IEEE Trans. Audio Electroacoust.*, vol. AU-18, pp. 107-122, June 1970.

[8] S. Y. Hwang, "On Optimization of Cascade Fixed-Point Digital Filters," *IEEE Trans. Circuits Syst.*, vol. 23, pp. 163-166, Jan. 1974.

[9] B. Liu and A. Peled, "Heuristic Optimization of the Cascade Realization of Fixed-Point Digital Filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 464-473, Oct. 1975.

[10] G. Dehner, "On the noise behaviour of a digital filter in cascade structure," in *Proc. Int. Symp. Circuits Syst.*, 1976, pp. 348-351.