OPTIMIZING VERTICAL COMMON SUBEXPRESSION ELIMINATION USING COEFFICIENT PARTITIONING FOR DESIGNING LOW COMPLEXITY SOFTWARE RADIO CHANNELIZERS

A.P.Vinod and E.M-K.Lai

School of Computer Engineering, Nanyang Technological University
Nanyang Avenue, Singapore 639798
Email: {asvinod, asmklai}@ntu.edu.sg

ABSTRACT

The complexity of finite impulse response (FIR) filters used in the channelizer of a software defined radio (SDR) receiver is dominated by the complexity of the coefficient multipliers. A method for designing low complexity channel filters by optimizing vertical common subexpression elimination (VCSE) using coefficient partitioning is presented in this paper. Our algorithm exploits the fact that when multiplication is implemented using shifts and adds, the adder width can be minimized by limiting the shifts of the operands to shorter lengths. Design examples of the channel filters employed in the Digital Advanced Mobile Phone System (D-AMPS) receiver show that the proposed method offers considerable full adder reduction over the VCSE methods.

1. INTRODUCTION

The most computationally intensive part of an SDR receiver is the channelizer since it operates at the highest sampling rate [1]. It extracts multiple narrowband channels from a wideband signal using a bank of FIR filters, called channel filters. Low power and high-speed FIR filters implemented with the minimum number of adders are required in the channelizer. Among the approaches for reducing the number of adders in the multipliers of FIR filters, the CSE techniques in [2]-[4] produced the best hardware reduction since it deals with multiplication of one variable (input signal) with several constants (coefficients). However, the methods in [2]-[4] have not addressed the issue of minimizing the complexity of each adder of the multiplier, which is significant in low power and high-speed implementations. In our recent work [5], we have analyzed the complexity of implementation of FIR filters in terms of the number of full adders (FA's) required for each multiplier. A vertical super-subexpression elimination (VSSE) method for optimizing the VCSE method in [4] to implement low-complexity channel filters using minimum number of FA's has been proposed in [5]. This technique is based on the extension of conventional two-nonzero bit (2-bit) vertical common subexpressions (VCS) in [4] to form three-nonzero bit and four-nonzero bit vertical super-subexpressions (called 3-bit and 4-bit VSS, respectively). The main limitation of the method in [5] is its dependence on the statistical distribution of shifts between the 2-bit VCS in the canonic signed digit (CSD) representations of FIR filter coefficients. Moreover, the routing complexity of the filters designed using the method in [5] is higher than that of the 2-bit VCSE techniques in [2]-[4] as the former method has more number of subexpressions.

In this paper, we show that low complexity coefficient multipliers can be realized by combining three techniques: an efficient coefficient partitioning algorithm, the pseudo floating-point (PFP) representation and the VCSE, which reduces the number of FA's. The FA reduction techniques proposed in this paper do not employ VSS used in [5] and hence they do not have the dependence on statistical distribution of shifts between the 2-bit CS. The problem that we address here is how to minimize the number of FA's required in each adder of a given minimum-adder multiplier filter structure. Though we use the VCSE [4] and VSSE [5] techniques for comparison, our algorithm can also optimize the coefficient multipliers designed using other methods to further minimize the number of FA's.

The paper is organized as follows. In section 2, we provide a brief review of the complexity analysis of coefficient multipliers. Our coefficient partitioning method is presented in section 3. In section 4, we illustrate the implementation of channel filters for the D-AMPS standard using our method and provide comparisons. Section 5 provides our conclusions.

2. MULTIPLIER COMPLEXITY

Definition 1 (Range): The range is analogous to the wordlength, which is equal to the number of bits of an operand (input signal shifted corresponding to the positional weights of the nonzero terms of the coefficient form the *operands* of the adders). For example, if x_1 is an 8-bit quantized signal (as assumed throughout the paper), the range of the operand, $x_1 >> 6$, is fourteen. (Note that *range* is same as *span* in [5]. In this paper, we use the term *span* in the PFP representation. Correspondingly, r_n is the range of the nth operand, which is same as s_n in [5]).

Case I: Odd number of operands: The number of FA's, (N_o), required to compute the output corresponding to a coefficient with n (for n odd*)* operands can be determined using the expression [5]:

$$N_o = r_2 + a_1 r_3 + 2r_4 + a_3 r_5 + r_6 + a_5 2r_7 + 3r_8 + a_7 r_9 + r_{10} + a_9 2r_{11} + 2r_{12} \tag{1}$$

where a_i's are equal to zero except $a_{n-2} = 1$.

Case II: Even number of operands: The number of FA's, (N_e), required to compute the output corresponding to a coefficient with *n* operands is given by [5]:

$$N_e = r_2 + 2r_4 + c_0 r_6 + 3r_8 + c_1 r_{10} + 3r_{12} \qquad (2)$$

where $c_0 \equiv \begin{cases} 2, \text{for } n=6 \\ 1, \text{elsewhere} \end{cases}$ and $c_1 \equiv \begin{cases} 2, \text{for } n=10 \\ 1, \text{elsewhere} \end{cases}$.

Note that (1) and (2) are same as in [5], except that we use the notation r_n here instead of s_n and *range* is same as *span* in [5].

The coefficients $h(0)$ and $h(1)$ of an FIR filter expressed in 12-bit CSD form shown in Fig. 1 is used as an example to illustrate the VCSE method and our optimization. The numbers in the first row of Fig. 1 represent the number of bitwise right shifts.

	1	2	3	4	5	6	7	8	9	10	11	12
h(0)	0	0	0	1	0	-1	0	0	-1	0	1	0
h(1)	0	0	0	1	0	-1	0	0	-1	0	0	1

Fig. 1. VCS in filter coefficients.

In direct implementation, (i.e., the implementation using shifts and adds and without VCSE) the outputs of the filter taps are given by (3) and (4).

$$y(0) = 2^{-4}x_1 - 2^{-6}x_1 - 2^{-9}x_1 + 2^{-11}x_1 \qquad (3)$$

$$y(1) = 2^{-4}x_1[-1] - 2^{-6}x_1[-1] - 2^{-9}x_1[-1] + 2^{-12}x_1[-1] \qquad (4)$$

where [-*k*] represents a delay of *k* units. For both (3) and (4), *n* is 4 (even). The ranges r_2 and r_4 in (3) are 14 and 19 respectively. Using (2) the number of FA's required to compute (3) in direct method is $r_2 + 2r_4$, i.e., 52 FA's. Similarly, the ranges r_2 and r_4 in (4) are 14 and 20 respectively and the number of FA's required to compute (4) in direct method is 54. Thus, a total of 106 FA's are needed to compute (3) and (4) using direct implementation.

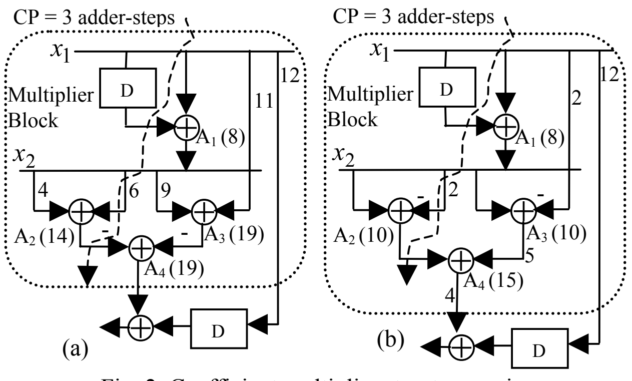

Fig. 2. Coefficient multiplier structures using VCSE (a) and our CPM (b).

The objective of VCSE algorithm is to identify multiple identical bit patterns that exist across the coefficient set and eliminate redundant computations by forming VCS from the bit patterns. The 2-bit VCS, [1 1] shown encircled in Fig. 1 is given by $x_2 = x_1 - x_1[-1]$. Using VCS, the output of the filter can be expressed as

$$y = 2^{-4}x_2 - 2^{-6}x_2 - 2^{-9}x_2 + 2^{-11}x_1 + 2^{-12}x_1[-1] \qquad (5)$$

Fig. 2(a) shows the multiplication structure using VCSE. The numerals adjacent to the data path in Fig. 2 represents the number of bitwise right shifts. The numerals in brackets alongside the adders indicate the number of FA's used in the adder. Thus, the number of FA's required for the multiplier block using VCSE method [4] is 60 in this case, which is a reduction of 43.4% over the direct method.

3. THE COEFFICIENT PARTITIONING METHOD

The key idea in our approach is to reduce the *ranges* of the operands so that the adder width can be reduced which in turn minimizes the number of FA's. To achieve this, firstly the coefficients are encoded using the PFP representation and then partitioned for further reduction of range.

Definition 2 (Pseudo floating-point (PFP) representation): The CSD representation for the ith filter coefficient of wordlength *B* is $h_i = \sum_{j=0}^{B-1} 2^{a_{ij}}$. The PFP representation of h_i is [6]

$$h_i = 2^{a_{i0}} \cdot \sum_{j=0}^{B-1} 2^{a_{ij} - a_{i0}} = 2^{a_{i0}} \cdot \left[\sum_{j=0}^{B-1} 2^{c_{ij}} \right] \qquad (6)$$

where $c_{ij} = a_{ij} - a_{i0}$. The term a_{i0} is known as the *shift* and the upper limit value, $(a_{i(B-1)} - a_{i0})$, is known as the *span.* Instead of expressing the coefficients using *B*-bit CSD, it can be expressed as a (*shift, span*) pair using fewer bits. For example, the PFP form of the coefficient $h(0)$ in the example in Fig. 1 is $2^{-4}(2^0 - 2^2 - 2^{-5} + 2^{-7})$. The term 2^{-4} is the *shift* part (implying 'right shift by 4'), and the bracketed term is the *span* part. Note that the shift operation can be performed after the addition of all the terms of the span part. The cost of shifts is negligible as they can be hardwired. This reduces the effective wordlength of the coefficient to that of the span (7 bits), which in turn reduces the ranges of the operands. Using (2), the number of FA's required to implement the coefficient multipliers in Fig. 1 when the coefficients are coded using PFP is 82. Though this FA requirement is less than that of direct implementation, it must be noted that the PFP implementation needs more FA's when compared with the VCSE method in Fig. 2(a). We shall now show that by combining the PFP coding scheme with the VCSE and then partitioning the resulting expression, considerable reduction of FA's can be achieved.

3.1 FA Reduction Using Coefficient Partitioning

The basic idea is to reduce the range of the span part of PFP by partitioning it into two sub-components, called *sub-filters.*

Definition 3 (Order): The most significant bit of a filter coefficient represented in CSD form is defined as the order of the coefficient. For instance, the order of a coefficient $h(n)=2^{-6}+2^{-8}+2^{-11}+2^{-14}+2^{-16}$ is 2^{-6}.

Firstly, the CSD coefficient is expressed using VCS and the resulting expression is then coded using PFP representation. Let M represents the span of the PFP representation. The span part is partitioned into two sub-components (sub-filters) of length $M/2$ (or two sub-components of lengths $\lfloor M/2 \rfloor$ and $\lceil M/2 \rceil$ if M is odd). The latter sub-component is then scaled by its *order* to reduce its span. The 'partitioned and scaled' versions of the PFP coefficients thus obtained can be added using fewer numbers of FA's since their ranges are reduced. Consider the same example shown in Fig. 1. Using PFP, the filter output corresponding to the nonzero bits of $h(0)$ and the VCS formed by $h(0)$ with $h(1)$ obtained in VCSE method can be expressed as $2^{-4}(x_2-2^{-2}x_2-2^{-5}x_2+2^{-7}x_1)$. In this case, the span (M) is 7 and the shift is 4. Partitioning the span part into two sub-filters, $h_1(n)$ and $h_2(n)$, we have

$$h_1(n)=x_2-2^{-2}x_2 \text{ and } h_2(n)=-2^{-5}x_2+2^{-7}x_1 \tag{7}$$

where $h(n)$ is the sum of $h_1(n)$ (MSB half) and $h_2(n)$ (LSB half). The LSB sub-filter is further scaled by its order, 2^{-5}, and expressed as $h_2(n)=-2^{-5}(x_2-2^{-2}x_1)$. Fig. 2(b) shows the implementation of the filter taps using our coefficient partitioning method (CPM). When compared with the VCSE method in Fig. 2(a), the adders A_2 and A_3, have shorter widths since the ranges of their operands are shorter. The shift 2^{-5} of $h_2(n)$ and that of the final expression $2^{-4}(x_2-2^{-2}x_2-2^{-5}x_2+2^{-7}x_1)$ are performed after the addition stages as shown alongside the data paths at the outputs of adders A_3 and A_4 respectively. Thus, our method requires only 43 FA's to implement the filter tap, which is a reduction of 28.3% over the VCSE method [4]. Note that both methods have identical critical path (CP) lengths (3 adder-steps) and hence their multiplier delays are same.

In order to meet the stringent adjacent channel interference specifications of wireless communications standards, higher order FIR filters are needed in SDR channelizers. It has been observed that the reduction rates offered by our method increases with the filter order. This can be explained by considering the numerical property of the *end-coefficients* of higher-order FIR filters.

Definition 4 (End-Coefficients): We designate the first $N/4$ coefficients, $h(0)$ to $h(\lceil N/4 \rceil -1)$, of an FIR filter with N taps as the *end-coefficients*. For example if N is 40, the coefficients, $h(0)$ to $h(9)$, of one half of the symmetric set, $h(0)$ to $h(19)$, are called its end-coefficients. As the filter order increases, the side-lobes of the impulse response decrease and hence the magnitudes of the end-coefficients of $h(n)$ will also decrease. Due to their lower magnitudes, most of the nonzero bits of the CSD representations of end-coefficients occur in the LSB part. In conventional implementation, this will lead to the use of longer *shifts* which will in turn increase the *ranges* of the operands and correspondingly the number of FA's. On the other hand, the use of shorter *shifts* in our method results in considerable reduction of FA's. Therefore, our method offers considerable FA reduction in the channelization application where higher order FIR filters are needed.

We also examined the adder complexity reduction achieved by partitioning the coefficient into more than two sub-components. If x_2 and x_3 are the VCS obtained from the input x_1, and x_{k_j} represents the data from the set $\{x_1, x_2, x_3\}$ that has to be shifted corresponding to the position of the *j*-th CSD bit, the general expression for filter output corresponding to a coefficient $h(n)$ of wordlength B is

$$y(n)=\sum_{j=1}^{z}(s_j 2^{-p_j})(x_{k_j}) \tag{8}$$

where $s_j \in \{-1,0,1\}$, $p_j \in \{0,1,........\text{B}\}$, and z is the number of nonzero digits. If p_{s_1} is the shift, (8) can be expressed in PFP form as

$$y(n)=2^{-p_{s1}}\sum_{j=1}^{z}(s_j 2^{-(p_j-p_{s1})})(x_{k_j}) \tag{9}$$

Partitioning $h(n)$ into n sub-components at equal intervals (i.e., n_1, n_2,.... n), (9) can be written as

$$y(n)=2^{-p_{s1}}\Bigg[\sum_{j_1=1}^{n_1}s_{j_1}2^{-(p_{j_1}-p_{s1})}x_{k_{j_1}}+ 2^{-p_{s2}}(\sum_{j_2=1}^{n_2}s_{j_2}2^{-(p_{j2}-p_{s1}-p_{s2})}x_{k_{j2}})+.......... ....+2^{-p_{sn}}(\sum_{j_n=1}^{n}s_{j_n}2^{-(p_{j_n}-p_{s1}-p_{s_n})}x_{k_{j_n}})\Bigg] \tag{10}$$

In this case, the widths of the adders in the intermediate-stages of the multiplier are larger since the multiple inner shifts, $(2^{-p_{s2}}, 2^{-p_{s3}}, ..., 2^{-p_{sn}})$, in (10) need to be performed prior to the intermediate additions. Hence, each of these intermediate-stage adders would require more FA's. On the other hand, when the coefficient is partitioned into two sub-components, only one inner shift operation exists (i.e., $2^{-p_{s2}}$) and this is done just before the final-stage adder of the multiplier. Therefore, the widths of the adders in the preceding stages that compute the sum of the bracketed term of $2^{-p_{s2}}$ are less and only the final-stage adder requires the highest width. Hence, partitioning a coefficient into two halves offers the best reduction of FA's than partitioning into multiple parts.

The steps of our CPM are as follows.

Step 1: Set $k=0$. Identify the VCS [1 1], [1 –1], [1 0 1] and [1 0 –1] and their negated versions in the CSD representation of $h(k)$. Express the output corresponding to $h(k)$ using VCSE.

Step 2: Express the VCSE output corresponding to $h(k)$ in PFP. Set $M = span$.

Step 3: Partition the span part into two sub-filters of length $M/2$. Scale the latter sub-filter by its *order*.

Step 4: Increment k. If $k \neq N$, go to Step 1. Otherwise, terminate the program.

4. DESIGN EXAMPLES

The FIR filters employed in the D-AMPS Channelizer [7] are considered. The sampling rate of the wideband signal chosen is 34.02 MHz as in [7]. The channel filters extract 30 kHz D-AMPS channels from the wideband signal after downsampling by a factor of 350. The pass-band and stop-band edges are 30 kHz and 30.5 kHz respectively. The peak pass-band ripple specification is 0.1 dB. The peak stop-band ripple (PSR) specifications at different frequencies and respective filter lengths (N) are chosen to be as in the D-AMPS standard. These parameters are shown in Table I.

Table I Specifications of the D-AMPS channel filters

PSR (dB)	-48	-65	-85	-96
N	260	610	940	1180

The reduction of FA's over the direct implementation in designing the channel filters whose coefficients are coded using 16-bit CSD, for different filter lengths are shown in Fig. 3. For the filter with 1180 taps (corresponding to the most stringent blocking specification), our method (CPM) offers a reduction of 71%, whereas the reductions offered by the VSSE [5] and the VCSE [4] methods are 56.7% and 33.8% respectively. The average reduction of FA's for different filter lengths achieved using the VCSE [4] is 30.5% and the VSSE [5] is 50.2%. On the other hand, our method offers an average reduction of 64%.

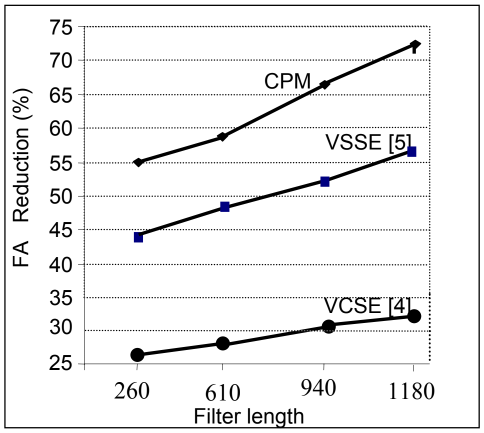

Fig. 3. Reduction of FA's over the direct implementation in designing the D-AMPS channel filters for different filter lengths.

Further, we examine the number of FA's needed to employ the filter bank channelizer, where extraction of each channel requires a separate narrowband filter. The wideband signal considered for channelization consists of 1134 D-AMPS channels, each occupying 30 kHz. We analyzed the requirement of adders to implement the filters for extracting 70, 141, 283, 567, and 1134 channels. The number of filter taps chosen is 1180 and the coefficient wordlength considered is 16 bits. Fig. 4 depicts the FA reduction achieved using different optimization methods over the direct implementation as a function of the number of extracted channels. The average reduction of FA's offered by our CPM is 54.8% whereas the reductions achieved using the VCSE [4] and the VSSE [5] methods are 32.7% and 41.8% respectively. Note that the reduction rate offered by our method increases when the number of channels extracted increases.

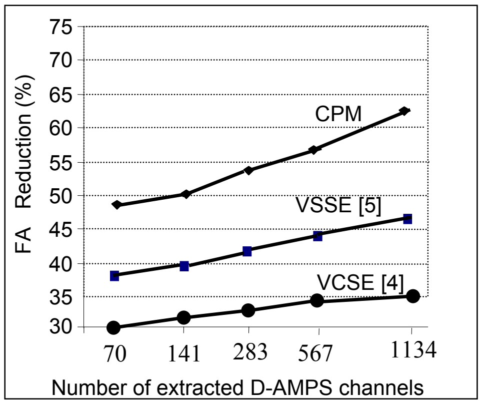

Fig. 4. Reduction of FA's to implement the D-AMPS channel filters for different number of channels extracted.

5. CONCLUSIONS

We have proposed a coefficient partitioning technique to efficiently implement low-complexity channel filters for SDR receivers. The design examples show that our method offers average FA reductions of 22% over the VCSE method [4] and 13% over the VSSE method [5]. Though we used the common subexpression techniques to compare our method, it must be noted that our algorithm can also be applied to reduce the FA requirement of minimum-adder FIR filter coefficient multipliers designed using other methods. Therefore, our approach offers a more general solution to multiplier complexity reduction.

6. REFERENCES

[1] J. Mitola, *Software Radio Architecture.* New York: Wiley, 2000.

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

[3] M. M. Peiro, E. I. Boemo, and L. Wanhammar, "Design of high-speed multiplierless filters using a nonrecursive signed common subexpression algorithm," *IEEE Trans. Ckts. Syst. II,* vol. 49, no. 3, pp. 196-203, March 2002.

[4] Y. Jang and S.Yang, "Low-power CSD linear phase FIR filter structure using vertical common subexpression," *Electronics Letters,* vol. 38, no. 15, pp. 777-779, July 2002.

[5] A. P. Vinod, E. M-K. Lai, A. B. Premkumar and C. T. Lau, "Optimization method for designing filter bank channelizer of a software defined radio using vertical common subexpression elimination," *Proceedings of the IEEE International symposium on Ckts. Syst.,* vol. 4, pp. 437-440, Vancouver, Canada, May 2004.

[6] A. P. Vinod, A. B. Premkumar and E. M-K. Lai, "An optimal entropy coding scheme for efficient implementation of pulse shaping FIR filters in digital receivers," *Proceedings of the IEEE International Symposium on Ckts. Syst.,* vol. 4, pp. 229-232, Bangkok, Thailand, May 2003.

[7] K. C. Zangi, R. D. Koilpillai, "Software radio issues in cellular base stations," *IEEE Journal on Selected Areas in Communication,* vol. 17, no. 4, pp. 561-573, April 1999.