COMPLEXITY REDUCTION OF SOFTWARE DEFINED RADIO CHANNELIZERS USING FILTER COEFFICIENT-PARTITIONING

A.P.Vinod, E.M-K.Lai and S.Emmanuel
School of Computer Engineering
Nanyang Technological University, Singapore 639798

ABSTRACT

The computational cost 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. Previous works have focused on minimizing the number of adders employed in the coefficient multipliers. These works have not considered reducing the complexity of each adder, which is significant in SDR applications that require low power and high-speed. In this paper, we present a coefficient-partitioning algorithm for minimizing the complexity of adders used in the multipliers. Our algorithm exploits the fact that when multiplication is realized using shifts and adds, the adder width can be minimized by limiting the shifts of the operands to shorter lengths. Design examples show that the proposed method offers an average full adder reduction of 20% over the common subexpression elimination (CSE) methods.

1. INTRODUCTION

The most computationally intensive part of the wideband receiver of an SDR 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. The complexity of implementation of FIR filters in terms of the number of full adders (FAs) required for each multiplier was presented in [5]. A method for optimizing the CSE method in [2] to implement low-complexity channel filters was also proposed in [5]. This technique is based on the extension of conventional two-nonzero bit (2-bit) common subexpressions (CS) in [2] to form three-nonzero bit and four-nonzero bit super-subexpressions (called 3-bit and 4-bit SS, respectively) by exploiting identical shifts between a 2-bit bit CS and a third nonzero bit, or between two 2-bit CS. Since employing SS reduces the number of adders, the number of FAs is also reduced correspondingly – this is the basic approach adopted in [5]. The main limitation of the method in [5] is its dependence on the statistical distribution of shifts between the 2-bit CS 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 CSE techniques in [2]-[4] as the former method has more number of subexpressions.

In this paper, an efficient coefficient-partitioning (CP) algorithm to implement the multipliers of channel filters with a minimum number of FAs is proposed. We combine three techniques: the CP algorithm, the pseudo floating-point (PFP) representation and the CSE, to reduce the number of FAs. The FA reduction techniques proposed in this paper do not employ super-subexpressions proposed 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 FAs required in each adder of a given minimum-adder filter structure.

The paper is organized as follows. A review of multiplier complexity analysis is provided in section 2. In section 3, we present our CP method. In section 4, we provide design examples to illustrate our method. Section 5 provides our conclusions.

2. MULTIPLIER COMPLEXITY

For completeness, a brief review of the complexity of the multiplier formulated in [5] is provided here. For an adder whose operands have ranges r_1 and r_2 (range is the number of bits of an operand) such that $r_2 > r_1$, the adder width is assumed as r_2. Consequently, at the most r_2 FAs are needed to compute the sum. (If overflow occurs, (r_2+1) FAs are required. For simplicity, we assume only r_2 FAs throughout the paper).

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

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

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

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

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

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 *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]).

The coefficient $h_k = 0.0000101001010101$, is used as an example to illustrate the CSE method [2] here. In direct implementation, (i.e., the implementation using shifts and adds and without CSE or any other multiplier optimization techniques) the filter tap output is

$$y_k = 2^{-5}x_1 + 2^{-7}x_1 + 2^{-10}x_1 + 2^{-12}x_1 + 2^{-14}x_1 + 2^{-16}x_1 \quad (3)$$

where x_1 is the input. In this case, n is 6 (even), r_2, r_4, and r_6 are 15, 20 and 24 respectively. Using (2) the total number of FAs required to compute (3) in direct method is $r_2 + 2r_4 + 2r_6$, i.e., 103 FAs. The goal of the CSE technique [2] is to identify multiple occurrences of identical bit patterns in the coefficient set. The pattern [1 0 1] is present thrice in this example, which can be expressed as a common subexpression (CS),

$$x_2 = x_1 + x_1 >> 2 \quad (4)$$

Using the CS (4), the output can be expressed as [2]

$$y_k = x_2 >> 5 + x_2 >> 10 + x_2 >> 14 \quad (5)$$

Fig. 1 shows the multiplication structure using CSE. The numerals adjacent to the data path represents the number of bitwise right shifts. It requires $r_2 = 10$ FAs for computing (4) and $r_2 + r_3 = 20 + 24 = 44$ FAs for (5) as shown alongside the adders A_1, A_2 and A_3. The numerals in brackets alongside the adders indicate the number of FAs used in the adder. Thus in this case, 54 FAs are required for computing y_k using CSE method [2], which is a reduction of 47% over the direct method.

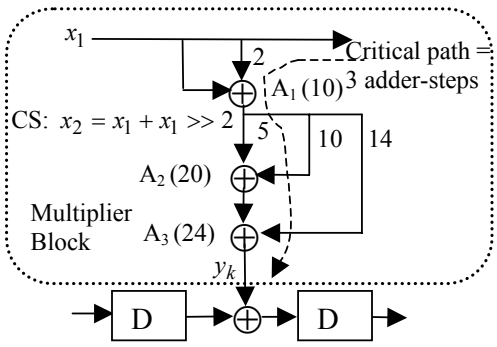

Fig.1. FIR multiplier realization using CSE [2].

3. COEFFICIENT-PARTITIONING

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 FAs. To achieve this, firstly the coefficients are encoded using the PFP representation and then partitioned for further reduction of range.

Definition1 (Pseudo floating-point (PFP) representation): The general representation of CSD for the i[th] filter coefficient that has a wordlength B is $h_i = \sum_{j=0}^{B-1} 2^{a_{ij}}$. The PFP representation of h_i is [6]

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

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 in the example in Fig. 1 is $2^{-5}(2^0 + 2^{-2} + 2^{-5} + 2^{-7} + 2^{-9} + 2^{-11})$. The term 2^{-5} is the *shift* part, and the bracketed term is the *span* part. The shift operation can be performed after the addition of all the terms of the span part. This reduces the effective wordlength of the coefficient to that of the span (11 bits), which in turn reduces the ranges of the operands. Using (2), the number of FAs required to implement the PFP coefficient multiplier is 78. We shall now show that by combining the PFP coding scheme with the CSE and then partitioning the resulting expression, further reduction of FAs can be achieved.

3.1. FA Reduction Using Coefficient-Partitioning

The basic idea of CP is to reduce the range of the span part of PFP by partitioning it into two parts.

Definition 2 (Order): The most significant bit of a filter coefficient represented in CSD form is defined as the order of the coefficient.

Firstly, the CSD coefficient is expressed using CS 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 parts 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 FAs since their ranges are reduced. Consider the same example of the filter tap shown in Fig. 1. Using PFP, the filter output obtained in CSE method (5) can be expressed as $2^{-5}(x_2 + 2^{-5}x_2 + 2^{-9}x_2)$. In this case, the span (M) is 9 and the shift is 5. Partitioning the span part into two parts, $h_1(n)$ and $h_2(n)$, we have

$$h_1(n) = x_2 \text{ and } h_2(n) = 2^{-5}x_2 + 2^{-9}x_2 \quad (7)$$

where $h(n)$ is the sum of $h_1(n)$ (MSB half) and $h_2(n)$ (LSB half). The LSB part is further scaled by its order, 2^{-5}, and expressed as $h_2(n) = 2^{-5}(x_2 + 2^{-4}x_2)$. Fig. 2 shows the implementation of the filter tap using our CP method. When compared with the CSE method in Fig. 1, 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 $2^{-5}(x_2 + 2^{-5}x_2 + 2^{-9}x_2)$ are performed after the addition stages as shown alongside the data paths at the outputs of adders A_2 and A_3 respectively.

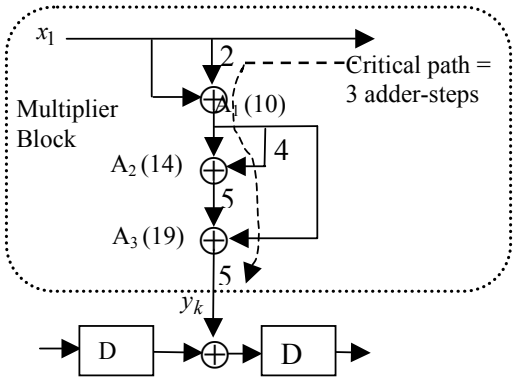

Fig. 2. Multiplier realization using CP method.

Thus, our method requires only 43 FAs to implement the filter tap, which is a reduction of 20.4% compared with the CSE method [2]. Note that both methods have identical critical path lengths (3 adder-steps) and hence their multiplier delays are same.

The steps of the CP algorithm are as follows.

Step 1: Design the filter of length N.

Step 2: Obtain the CSD representation of the coefficients for a desired wordlength. Set $k = 0$.

Step 3: Identify the CS [1 0 1] and [1 0 –1] and their negated versions in $h(k)$. Express the filter output corresponding to the coefficient $h(k)$ using HCSE.

Step 4: Express the HCSE output corresponding to $h(k)$ in PFP. Set *M = span.*

Step 5: Partition the span part into two parts of length $M/2$. Scale the latter part by its *order*.

Step 6: Increment *k.* If $k \neq N$, go to Step 3. Otherwise, terminate the program.

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 CS 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

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

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

$$y(n) = 2^{-p_{s_1}} \sum_{j=1}^{z} (s_j 2^{-(p_j - p_{s_1})})(x_{k_j}) \qquad (9)$$

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

$$y(n) = 2^{-p_{s_1}} \left[\sum_{j_1=1}^{n_1} s_{j_1} 2^{-(p_{j_1} - p_{s_1})} x_{k_{j_1}} + \right.$$

$$2^{-p_{s_2}} \left(\sum_{j_2=1}^{n_2} s_{j_2} 2^{-(p_{j_2} - p_{s_1} - p_{s_2})} x_{k_{j_2}} \right) + ...$$

$$\left. .....+ 2^{-p_{s_n}} \left(\sum_{j_n=1}^{n} s_{j_n} 2^{-(p_{j_n} - p_{s_1} - p_{s_n})} x_{k_{j_n}} \right) \right] \qquad (10)$$

In this case, the widths of the adders in the intermediate-stages of the multiplier are larger since the multiple inner shifts, $(2^{-p_{s_2}}, 2^{-p_{s_3}}, ..., 2^{-p_{s_n}})$, in (10) need to be performed prior to the intermediate additions. Hence, each of these intermediate-stage adders would require more FAs. On the other hand, when the coefficient is partitioned into two sub-components, only one inner shift operation exists (i.e., $2^{-p_{s_2}}$) 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 $2^{-p_{s_2}}$ are less and only the final-stage adder requires the highest width. Hence, partitioning a coefficient into two halves offers the best reduction of FAs than partitioning into multiple parts.

4. DESIGN EXAMPLES

Example 1: The FIR filters employed in the channelizer of the D-AMPS in [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 [8]. 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 FAs 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 74%, whereas the reductions offered by the SSE [5] and the CSE methods [2] are 62.1% and 35.6% respectively. The average reduction of FAs for different filter lengths achieved using the CSE [2] is 31.4% and the SSE [5] is

54.2%. On the other hand, our CP method offers an average reduction of 66%.

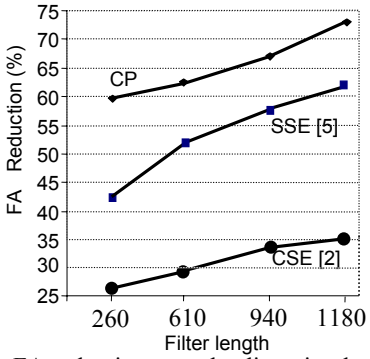

Fig. 3. FA reduction over the direct implementation in designing the DAMPS channel filters for different filter lengths.

Further, we examine the number of FAs 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 analysed 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 shows the FA reduction achieved using different optimization methods over the direct implementation as a function of the number of extracted channels.

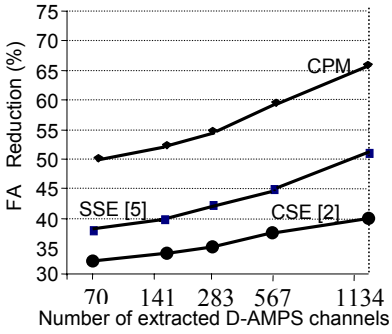

Fig.4. FA reduction in implementing D-AMPS channel filters for different number of channels.

The average reduction of FAs using CP method is 56.6%whereas the reductions achieved using the CSE [2] and the SSE [5] methods are 35.6% and 43.2% respectively.

Example 2: In this example, we consider the channel filters employed in receivers for the PDC standard. The sampling rate of the wideband signal is 25.6 MHz, which covers 1024 channels of 25 kHz spacing. The filter length is 1000 to meet the maximum attenuation requirement of –90 dB and 24-bit coefficients are considered. Fig. 5 shows the reduction of FAs achieved using various methods over the direct method for extracting different number of channels (128, 256, 512, 768 and 1024). Our CP method offers an average FA reduction of 55%, which is better than that of the CSE [2] and SSE [5] methods by 15% and 22% respectively.

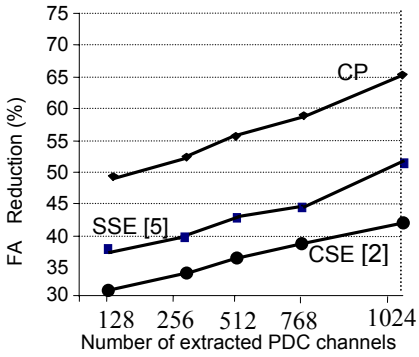

Fig. 5. FA reduction for the PDC channel filters for different number of channels extracted.

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 20% over the CSE method [2] and 15% over the SSE method [5]. Though we used the common subexpression techniques to compare our method, our algorithm can also be applied to reduce the FA requirement of minimum-adder FIR filter coefficient multipliers designed using any other methods. Therefore, our approach in this paper offers a more general solution to multiplier complexity reduction.

6. RFERENCES

[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. Circuits 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. Circuits 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 and E. M-K. Lai, "On the implementation of efficient channel filters for wideband receivers by optimizing common subexpression elimination methods," *IEEE Trans. On Computer-Aided Design of Integrated Circuits Syst.,* vol. 24, no. 2, pp. 295-304, Feb. 2005.

[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. And 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.

[8] N. Spencer, "An overview of digital telephony standards," *IEE Colloquium on the Design of Digital Cellular Handsets,* pp. 1/1-1/7, March 1998.