# FIR filter implementation by efficient sharing of horizontal and vertical common sub-expressions

A.P.Vinod, E.M-K.Lai, A.B.Premkumar, and C.T.Lau

The vertical common subexpression elimination (CSE) method proposed by Jang *et al.* does not guarantee hardware reduction over conventional horizontal CSE method in practical linear phase finite impulse response (LPFIR) filter implementations. A method to implement FIR filters with a minimum number of adders by efficiently combining horizontal and vertical common subexpressions is proposed here.

*Introduction:* Multiple constant multiplications (MCM) in digital filters refer to multiplication of one variable with multiple constants [1]. Common subexpression elimination proposed to tackle the MCM problem minimizes the number of additions by extracting the common parts among the constants represented in canonic signed digit (CSD) form [1]-[3]. Methods proposed in [2] and [3] eliminate redundant computations in multiplier blocks by employing the most common horizontal subexpressions among the CSD coefficients. Recently, Jang *et al.* proposed a vertical CSE technique as a better solution to the MCM problem [4]. However, this method does not guarantee hardware savings over conventional horizontal CSE method in practical LPFIR filters. In this letter, an efficient way to combine both horizontal and vertical subexpressions to achieve considerable hardware reduction in high-speed/low-power FIR filters is presented.

*Conventional common subexpression methods:* The number of adders (or subtractors), $N_a$, required to implement an LPFIR filter of length $N$ using CSE method can be computed using the expression:

$$N_a = (N_b - 1) - 2N_s + N_{as} \tag{1}$$

where $N_b$ is the total number of nonzero bits in all $N$ coefficients, $N_s$ is the total number of subexpressions in the $\left\lfloor \frac{N}{2} \right\rfloor$ symmetric coefficients and $N_{as}$ is the number of adders required for distinct subexpressions. A linear phase raised cosine FIR filter used for pulse shaping in the intermediate frequency (IF) processing block of a GSM receiver is considered to illustrate the CSE

method. The filter specifications are: $N = 15,$ cutoff frequency 135.44 kHz, roll-off factor 0.22 and sampling frequency is 541.67 kHz, which is twice the baud rate of GSM. The nonzero bits of the symmetric $\left\lfloor \frac{N}{2} \right\rfloor$ coefficients along with the highest value coefficient, $h(7),$ in 12-bit CSD form are shown in Fig. 1. For notational convenience, -1 is represented by *n.* In the CSD representation shown in our examples, the signs of coefficient values are assumed positive for convenience. Without using CSE, the number of adders required to implement the filter is $N_b - 1,$ which is 30 in this case. The conventional horizontal common subexpressions 101 and 10*n* indicated with bold lines are given by:

$$x_2 = x_1 + x_1 >> 2 \text{ and } x_3 = x_1 - x_1 >> 2 \quad (2)$$

where $>>$ represents the shift operation and $x_1$ represents the input signal. Thus, from (2) we obtain $N_{as} = 2$ and from Fig. 1, $N_s = 5.$ When horizontal CSE is used, the number of adders required is 22 from (1). This offers a reduction rate of 26.7% when compared to direct implementation without CSE. On the other hand, the vertical common subexpressions 10001, 1000*n,* and 101 indicated with dotted lines in Fig. 1 would require 25 adders since $N_s = 4$ and $N_{as} = 3.$ The reduction of 16.7% achieved by the vertical CSE is considerably lower than that using horizontal CSE.

*Distribution of common subexpressions:* It has been reported that in LPFIR filters, the most significant bits (MSB) of adjacent coefficients are identical since they have similar values and therefore a large number of vertical subexpressions occur [4]. However, our observation is that in most of the practical LPFIR filters, the magnitudes of adjacent coefficients are not similar and hence it is unlikely that their MSB are identical when represented in CSD. We observe that many adjacent coefficients have identical least significant bits (LSB) as the wordlength is increased from 8-bit to 16-bit and hence more vertical subexpressions can be obtained for larger wordlengths. However, it is observed that the increase in horizontal common subexpressions with increasing wordlength is even greater. Statistically, horizontal common subexpressions, 101, 10*n,* 1001, and 100*n* occur more frequently in the CSD form of LPFIR filters and hence these subexpressions are the *most common horizontal subexpressions*. The number of vertical common subexpressions that exist in CSD coefficients is fewer than the most common horizontal subexpressions. The CSD form of the raised cosine filter shown in Fig. 1 illustrates this observation. Hence fewer adders are required when horizontal subexpressions are used to realize the filter.

*Proposed common subexpression sharing method:* Further reduction of adders can be achieved by efficiently combining horizontal and vertical subexpressions. To achieve this, firstly the four most common horizontal subexpressions, 101, 10*n,* 1001, and 100*n,* are extracted from the coefficient set represented in CSD. The remaining nonzero bits are examined for suitable vertical common subexpressions. Consider the same example shown in Fig. 2, where conventional horizontal subexpressions are given by (2). From the remaining bits, two vertical subexpressions, 101 and *n*01, are obtained:

$$x_4 = x_1 + x_1[-2] \text{ and } x_5 = -x_1 + x_1[-2] \tag{3}$$

where [-*k*] represents the delay operation. By combining the common subexpressions (2) and (3), the output of the filter can be represented as:

$$\begin{aligned} y = {} & x_3 >> 5 + x_2 >> 10 + x_3[-2] >> 4 + x_1[-2] >> 11 + x_3[-4] >> 3 + x_4[-4] >> 9 \\ & + x_5[-4] >> 12 + x_2[-6] >> 2 + x_1[-7] >> 1 + x_2[-8] >> 2 + x_4[-8] >> 9 - x_5[-8] >> 12 \\ & + x_3[-10] >> 3 + x_3[-12] >> 4 + x_1[-12] >> 11 + x_3[-14] >> 5 + x_2[-14] >> 10 \end{aligned} \tag{4}$$

We consider the transposed direct form FIR filter structure for implementation. It can be noted that only twenty adders are required to implement the filter, two for horizontal common subexpressions (2), two for vertical common subexpressions (3), and sixteen for filter output (4). This method results in reduction rates of 16.6% and 6.6% when compared to vertical and horizontal common subexpression methods, respectively. We present two examples to show the minimum adder realization of LPFIR filters by efficiently combining horizontal and vertical common subexpressions.

*Example 1:* We consider the Parks-McClellan design of a LPFIR filter whose specifications are $N$=26, pass-band and stop-band edges at $0.2\pi$ and $0.25\pi$ respectively. The infinite-precision filter coefficients and their CSD representation using 8 bits and 16 bits are shown in Table 1. It can be noted that the magnitudes of adjacent coefficients are considerably different. As a consequence, their MSB portions have fewer identical bits. In the proposed method, horizontal common subexpressions of 101, 10*n,* 1001, and 100*n* are first extracted from the CSD representation. From the remaining nonzero bits, vertical common subexpressions 1001, 100*n,* and 10001 are utilized. Comparison of the number of adders required for the filter using the common subexpression methods and reduction rates with respect to the implementation without using any subexpressions are shown in Table 2. The results indicate that the proposed method offers a reduction rate of 11% over vertical common subexpression method [4].

*Example 2:* In this example, a linear phase Parks-McClellan FIR filter with identical taps, $N$=219, as in [4] is considered. The pass-band and stop-band edges of the filter are $0.2\pi$ and $0.25\pi$ respectively. For the 16-bit CSD implementation, 386 adders are required when vertical common subexpression method is used. Employing the proposed method, adder requirement is reduced to 337, which is 8% less.

*Conclusion:* In this letter, we have shown that the vertical CSE method does not guarantee minimum adder implementation of LPFIR filters. We have shown that transposed direct form CSD filter structures with minimum number of adders can be realized by efficiently combining horizontal and vertical common subexpressions that exist in the filter coefficients. The filters realized using the proposed method require fewer adders than conventional horizontal and vertical CSE methods.

## References


1 POTKONJAK, M., SHRIVASTA, M.B., and CHANDRAKASAN, P.A.: 'Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination'. *IEEE Trans. Computer-Aided Design,* 1996, 15, (2), pp.151-161

2 HARTLEY, R.I.: 'Subexpression sharing in filters using canonic signed digit multipliers'. *IEEE Trans. Circuits Syst. II,* 1996, 43, (10), pp. 677-688

3 YAGYU, M., NISHIHARA, A, and FUJI, N.: 'Fast FIR digital filter structures using minimal number of adders and its application to filter design'. *ICICE Trans. Fundam. Electron. Commun. Comput. Sci.,* 1996, E79-A, (8), pp. 1120-1129

4 JANG, Y., and YANG, S.: 'Low-power CSD linear phase FIR filter structure using vertical common sub-expression'. *Electronics Letters,* 2002, 38, (15), pp. 777-779



**Author's affiliations:**
A.P.Vinod, E.M-K.Lai, A.B.Premkumar, and C.T.Lau (School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798)

Email: asvinod@ntu.edu.sg

**Figure captions:**

Fig. 1 Common sub-expressions (CSE) in 15-tap linear phase raised cosine filter coefficients. Hartley's Horizontal CSE (solid) and Jang et al's Vertical CSE (dotted)

Fig. 2 Combined Horizontal and Vertical common sub-expressions in 15-tap linear phase raised cosine filter coefficients

**Table captions:**

Table 1 Coefficients of the 26-tap Parks-McClellan linear phase FIR filter in example 1

Table 2 Number of adders required to implement the filter in example 1

Figure 1

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

Figure 2

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

Table 1

| Infinite-precision Coefficients $h(0) - h(12)$ | 8-bit CSD form $2^{-1}$ ⋯▶ $2^{-8}$ | 16-bit CSD form $2^{-1}$ ⋯▶ $2^{-16}$ |
|---|---|---|
| -0.00933078669575 | 0 0 0 0 0 0 1 0 | 0 0 0 0 0 0 1 0 1 0 –1 0 0 1 0 -1 |
| 0.07628237421426 | 0 0 0 1 0 1 0 -1 | 0 0 0 1 0 1 0 0 -1 0 0 0 1 0 0 -1 |
| 0.03135623682714 | 0 0 0 0 1 0 0 0 | 0 0 0 0 1 0 0 0 0 0 0 0 1 0 -1 0 |
| 0.01374432164657 | 0 0 0 0 0 1 0 -1 | 0 0 0 0 0 1 0 0 -1 0 0 0 0 1 0 0 |
| -0.00948598843682 | 0 0 0 0 0 0 1 0 | 0 0 0 0 0 0 1 0 1 0 0 -1 0 -1 0 1 |
| -0.03358586396879 | 0 0 0 0 1 0 0 0 | 0 0 0 0 1 0 0 0 1 0 1 0 -1 0 0 1 |
| -0.04680063247432 | 0 0 0 1 0 -1 0 -1 | 0 0 0 1 0 -1 0 0 0 0 0 0 0 -1 0 -1 |
| -0.03819695824263 | 0 0 0 0 1 0 0 1 | 0 0 0 0 1 0 1 0 0 -1 0 0 1 0 0 -1 |
| -0.00271831937636 | 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 1 0 -1 0 -1 0 0 1 0 |
| 0.05563093697248 | 0 0 0 1 0 0 -1 0 | 0 0 0 1 0 0 -1 0 0 1 0 0 0 -1 0 1 |
| 0.12420551537587 | 0 0 1 0 0 0 0 -1 | 0 0 1 0 0 0 0 0 0 -1 0 1 0 -1 0 -1 |
| 0.18473033065671 | 0 1 0 -1 0 0 0 -1 | 0 1 0 -1 0 0 0 -1 0 1 0 0 1 0 1 0 |
| 0.22024453765020 | 0 1 0 0 -1 0 0 0 | 0 1 0 0 -1 0 0 0 1 0 –1 0 0 0 0 1 |

Table 2

| Implementation method | 8-bit CSD | Reduction rate (%) | 16-bit CSD | Reduction rate (%) |
|---|---|---|---|---|
| Vertical CSE [4] | 37 | 17.7 | 84 | 29.4 |
| Conventional horizontal CSE | 35 | 22.2 | 72 | 39.4 |
| Proposed method | 32 | 28.9 | 70 | 41.2 |