A New Horizontal and Vertical Common Subexpression Elimination Method for Multiple Constant Multiplication

Kazunari Kato
Graduate School of Engineering
Gifu University,
1-1 Yanagido, Gifu-shi 501-1193 Japan
Email: n3124006@edu.gifu-u.ac.jp

Yasuhiro Takahashi and Toshikazu Sekine
Department of Electrical and Electronic Engineering
Gifu University,
1-1 Yanagido, Gifu-shi 501-1193 Japan
Email: {yasut, sekine}@gifu-u.ac.jp

***Abstract*— The common subexpression elimination (CSE) techniques address the issue of minimizing the number of adders needed to implement the multiple constant multiplication (MCM) blocks. In this paper, we propose a new CSE method using a combining horizontal and vertical technique. The proposed method searches firstly the frequency of higher order horizontal common subexpression, i.e., 3–5 bits, and then searches vertical. Our simulation results show that our method offers a good tradeoff between the implementation cost and the synthesis run-time in comparison with conventional methods.**

I. Introduction

In the digital signal processing (DSP) algorithms, many fixed transforms (e.g., FIR/IIR filter with fixed coefficients, DCT, DFT, etc) do not require the flexibility of a general-purpose multiplier as the multiplicand has a limited number of values. From this reason, it is attractive to carry out the multiplication by using shifts and adds. The shifts can be realized by using hard-wired shifters and hence they are essentially free. Furthermore, we can reduce the adder area by using the common subexpression elimination (CSE) techniques. The CSE tackles the multiple constant multiplication (MCM) problem [1], [2] by minimizing the number of additions through extracting common parts among the constants represented in canonic signed digit (CSD) [3]–[11]. There are three different kinds of common subexpressions (CSs): horizontal, vertical and oblique. Due to the computational complexity and the fact that linear phase FIR filters are symmetrical, the search for redundant computations in multiplier block is normally confined to horizontal CSs. Recently, Jang et al. [5] proposed a method of further reducing the number of adders by using vertical CSE, and Vinod et al. [6] proposed a combining horizontal and vertical CSE. However, the structures for these techniques are designed without any consideration of the number of registers (i.e. time delay elements). The gate number ratio of adders to registers is 1 : 0.6–0.8 [12]; therefore, in case of structure with many registers, the implementation cost cannot be reduced. In our previous paper [7] we have presented an improved horizontal and vertical CSE which is able to reduce the number of adders and registers, but the previous proposed method has needed a long simulation run-time in order to search all horizontal CS patterns.

In this paper, we propose a new CSE method using a combining horizontal and vertical technique in realizing multiple constant multipliers (MCM). The proposed method is firstly the frequency of higher order horizontal CS, i.e., 3–5 bits, and then searches vertical. The rest of this paper is organized in four sections. Section II describes the definition of the MCM. Section III provides a brief review of CSE approaches and then is presented a new CSE method. Section IV presents the synthesis results of the 20 MCM design examples. Finally, conclusions are drawn in section V.

II. Multiple Constant Multiplication

A common feature of many digital signal processing algorithms is that they involve computations of the form

$$Y_i = a_{ij} X_i \ (i = 0, 1, \cdots, N-1; \ j = 0, 1, \cdots, M-1), \quad (1)$$

where X_i and Y_i are input and output variable vectors, respectively. Also, a_{ij} is a set of constant coefficients, N is the number of coefficients and M is the word length. One typical example is the transposed form FIR filter that one input data is multiplied with the filter coefficients, as shown in Fig. 1. In this paper, we perform multiple multiplications in Equation (1) using the registers and the adders/subtracters in order to reduce the area. Then the problem of reducing the costs is stated as the problem of minimizing the weighted sum of the numbers of the registers and adders/subtracters which are needed to perform all of the multiplications. That is, the objective cost function (CF) to be minimized is written as:

$$\mathrm{CF} = \beta N_{\mathrm{reg}} + \gamma N_{\mathrm{as}} \quad (\beta > 0, \gamma > 0), \qquad (2)$$

where N_{reg} and N_{as} are the number of registers and adders/subtracters, respectively, β and γ are weights.

The above is called the multiple constant multiplication (MCM) problem. But the MCM problem is very complex that it is believed to be NP-hard. Hence, we have to find heuristics referred to as the CSE.

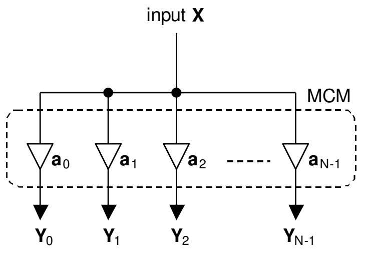

Fig. 1. Block diagram of MCM circuit.

III. Common Subexpression Elimination Method

A. Review

Common subexpression elimination (CSE) proposed to tackle the MCM problem minimizes the number of additions by extracting the common parts among the constructs represented in binary form [1], [2]. In the past, some new techniques for CSE, oblique (i.e. Hartley) [3], vertical (Jang et al.) [5], and combining horizontal and vertical (Vinod et al.) CSE technique [6] have been proposed a powerful solution to reduce the complexity in MCM, using the CSD representation.

Figure 2 illustrates the three different (horizontal, vertical and oblique) types of CSs where $\overline{1}$ denotes -1. The shared cells in this figure indicate contentions, where two or more CSs share the same nonzero digit. A contention implies a potential inhibition of sharing some CSs. The notations shown in Fig. 3 are used to express the three different types of subexpression. In this figure, $x[-i]$ denotes the value of x after i sample delays and $<< j$ stands for left shift of j digits. i and j can be deemed as the height and length of the CSs. We so assume that shift operations are essentially free, they can be hard-wired. However, the structures for vertical and oblique techniques are designed without any consideration of the number of registers. Vinod et al.'s CSE technique [6] has used combining horizontal and vertical CS to reduce the number of adders. However, the structures generated by using vertical CSE technique are still designed without any consideration of the number of registers. Therefore, if the structure of MCM contains many registers, the implementation cost cannot be reduced. Our improved horizontal and vertical CSE technique (Takahashi et al.) [7] has been proposed an efficient way to find the correct bit-patterns for horizontal and vertical CSEs. The proposed CSE has stated as the problem of minimizing the numbers of the delay and adders/subtracter blocks which are needed to perform all of the multiplications. Using Takahashi et al.'s method, the MCM area of the FIR filters has been reduced by an average of 20%. However, its CSE technique needs a long time of synthesis simulation because it searches all frequency of CS in the MCM.

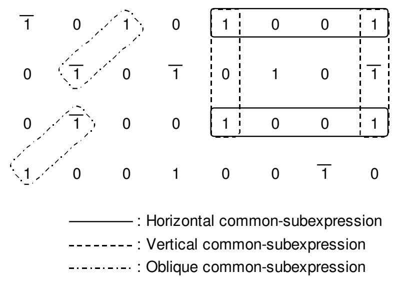
$\overline{1}$ 0 1 0 1 0 0 1
0 $\overline{1}$ 0 $\overline{1}$ 0 1 0 $\overline{1}$
0 $\overline{1}$ 0 0 1 0 0 1
1 0 0 1 0 0 $\overline{1}$ 0
: Horizontal common-subexpression
: Vertical common-subexpression
: Oblique common-subexpression

Fig. 2. Common subexpressions in the constant coefficients.

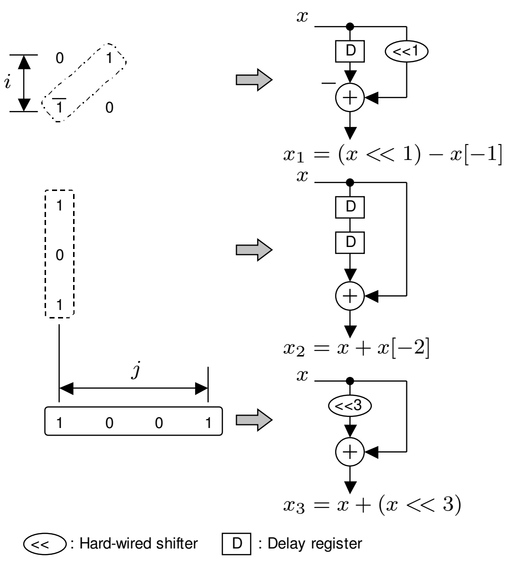

Fig. 3. Notations and circuit configurations of common subexpressions

B. Frequency of Common Subexpressions

Frequency of occurrence of a CS is defined as the number of times the same CS is being reused or repeated in the MCM. Thus, high frequency of occurrences of CSs means that most of the bits in the MCM will be grouped as CSs, leaving behind fewer numbers of unpaired bits. In this paper, we firstly analyzed the frequency of occurrences of MCMs. The MCMs were randomly designed as 100 FIR filters, and then all coefficients were represented in CSD format using MATLAB. Figures 4 and 5 show frequency of occurrence of the horizontal and vertical CS, respectively. From these figure we found that 3-, 4-, and 5-b horizontal CSs, i.e., [101], [10$\overline{1}$], [1001] and [10001], etc., have a high correlation with the frequency of occurrence, on the other hand there is no association between the frequency of occurrence and vertical CSs.

C. Proposed combining horizontal and vertical method

The proposed technique is described by the pseudo C language code shown in Fig. 6. Let us see this in more detail in the next subsubsection.

1) Proposed I– Approximate method: The proposed I method is searched as local horizontal CSs, i.e., [101], [10$\overline{1}$], [1001], [100$\overline{1}$], [10001], [1000$\overline{1}$] and their negated versions. Firstly, most common horizontal subexpressions resulting from the proposed method extracted from the matrix table in the MCM, and then remaining non-zero bits are examined for optimum vertical CS. This method has a short run-time since it becomes a local search for the horizontal CSs.

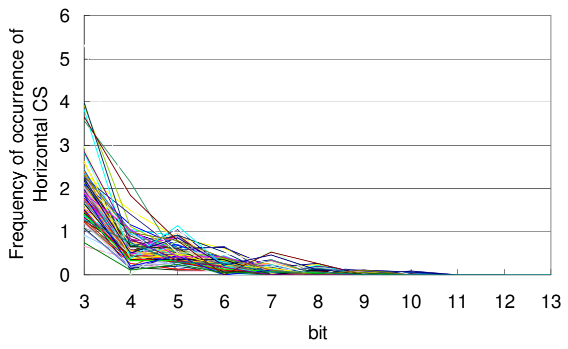

Fig. 4. Frequency of occurrence of a horizontal common subexpression for 100 examples.

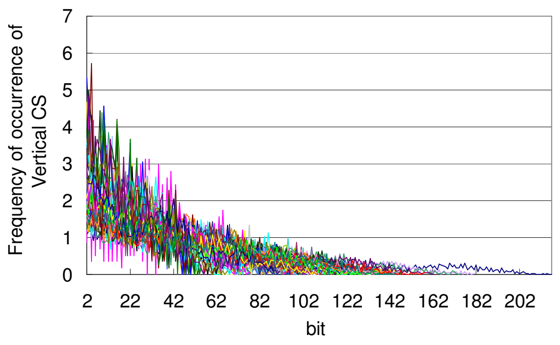

Fig. 5. Frequency of occurrence of a vertical common subexpression for 100 examples.

2) Proposed II– Exact method: This proposed exact method is applied as a brute force search. As the number of permutation is $12! = 479,001,600$, we have the MCM which consists of the smallest number of adders after 12! search iterations. Then, the remaining non-zero bits are examined for optimum vertical CS. Pattern identification of vertical CSE is the same as Takahashi et al.'s method. Finally, the number of adders and registers in the MCM are the smallest values by using this exact method however the solution is a local-minimum.

IV. Design Example Results

In this section, we present the results of the approximate (Proposed I) and exact (Proposed II) methods on randomly generated and FIR filter instances, and compare with the results of previously proposed heuristics. As the first experiment set, we used randomly generated 20 instances where the size of bitwidth (b) are defined in 13- or 14-b, and the number of coefficients (N) ranges between 70 and 220. Our experiments were implemented in MATLAB using a 2.2 GHz computer with 1024 MB memory. In these examples we set the parameters $\gamma = 1.0$, $\beta = 0.6$ in Equation (2) if we assume that the MCMs are fabricated in a 0.35 μm standard CMOS process [7], [10].

Run-time and implementation cost for benchmark examples are shown in Table 1 and 2, respectively. From these results we found that the Proposed I method can be used to design the MCM which has a small implementation compared to

```
void main()
{
   Eliminate zero coefficients;
   Merge coefficients with the same value;
   Construct the initial coefficient matrix;
     for Aproximate horizontal CSE (Proposed I)
       {
         Find coefficients with identical 3-5b CSs;
         Extract identical pattern;
         Update the coefficient matrix;
               if (identical pattern =0) {
                        break;
               }
               else {
                        return;
               }
       }
     for Exact horizontal CSE (Proposed II)
       {
         c = first(P)
          /* P= initial solution having the */
          /*    smallest number of adders   */
         while c!= NULL do
               if valid(P,c) then output(P,c) {
                  c = next(P,c)
               }
       }
     for vertical CSE
       {
         Find coefficients with identical pattern;
         Extract identical pattern;
         Find coefficients with similar pattern;
         Calculate the cost function of pattern;
         Extract similar/identical pattern;
         Update the coefficient matrix;
               if (identical pattern =0)   {
                   Output signal flow graph;
                        exit(0);
               }
               else {
                       return;
               }
       }
}
```

Fig. 6. Pseudo C code of the proposed CSE algorithm.

Hartley, Jang, and Vinod methods, The Proposed I method also performs about times as fast as Takahashi et al.'s method. On the other hand, the Proposed II method has the results of the smallest implementation cost but it needs a long computational time in order to use a brute force search.

V. Conclusion

This paper has been presented a new horizontal and vertical CSE method in realizing constant multipliers. The proposed method has searched firstly 3-, 4-, and 5-b horizontal common subexpressions within MCM blocks and then searched vertical. From the MATLAB synthesis results included in 20 MCM examples, we have found that the proposed method has reduced area of constant multiplier and decreased run-time compared with our previous method.

References

[1] M. Mehendale, S. D. Sherlekar, and G. Venkatesh, "Synthesis of multiplier-less FIR filters with minimum number of additions," in *Proc. 1995 IEEE/ACM Int. Conf. on Computer-Aided Design (ICCAD 1995)*, San Jose, CA, Nov. 5–9, 1995, pp. 668–671.

TABLE I

COMPARISON OF RUN-TIME FOR BENCHMARK EXAMPLES.

$N \times b$	Hartley [s]	Jang [s]	Vinod [s]	Takahashi [s]	Proposed I [s]	Proposed II [s]
77×13	0.074	$\sim$0	$\sim$0	0.250	0.094	340.0
116×14	0.063	0.016	0.016	0.390	0.125	471.9
138×14	0.063	0.015	0.015	0.453	0.156	564.4
136×14	0.063	$\sim$0	0.015	0.469	0.140	564.0
92×13	0.031	$\sim$0	0.016	0.297	0.094	387.5
142×14	0.063	0.016	0.015	0.469	0.156	596.9
141×14	0.078	$\sim$0	0.015	0.484	0.141	581.2
146×13	0.078	0.015	0.015	0.438	0.141	597.8
146×14	0.079	0.015	0.015	0.484	0.156	588.8
142×14	0.078	$\sim$0	0.031	0.469	0.156	628.3
101×13	0.047	0.016	0.015	0.328	0.110	490.5
98×13	0.047	$\sim$0	0.015	0.344	0.109	480.1
102×13	0.047	0.015	0.016	0.328	0.109	494.1
105×13	0.062	$\sim$0	0.016	0.328	0.110	484.2
101×13	0.063	$\sim$0	0.015	0.328	0.109	493.7
105×14	0.047	0.016	0.015	0.359	0.109	495.8
143×14	0.062	$\sim$0	0.016	0.484	0.157	627.9
142×14	0.047	0.015	0.016	0.344	0.109	492.4
216×13	0.093	0.016	0.031	0.594	0.188	789.0
144×14	0.063	0.016	0.015	0.469	0.140	614.0

TABLE II

COMPARISON OF IMPLEMENTATION COST FOR BENCHMARK EXAMPLES.

$N \times b$ (Cost [%])	Not-CSE	Hartley	Jang	Vinod	Takahashi	Proposed I	Proposed II
77×13	144(100)	114.4(79.4)	107.4(74.6)	101.2(70.3)	95.4(66.3)	94.2(65.4)	92.4(64.2)
116×14	209(100)	166.4(79.6)	158.4(75.8)	138.4(66.2)	134.0(64.1)	134.6(64.4)	132.4(63.3)
138×14	226(100)	182.4(80.7)	184.6(81.7)	160.4(71.0)	147.0(65.0)	149.6(66.2)	143.0(63.3)
136×14	239(100)	183.4(76.7)	179.6(75.1)	155.2(64.9)	153.2(64.1)	154.2(64.5)	149.2(62.4)
92×13	152(100)	122.8(80.8)	118.6(78.0)	111.4(73.3)	102.2(67.2)	102.2(67.2)	102.2(67.2)
142×14	246(100)	190.4(77.4)	202.6(82.4)	171.4(69.7)	151.4(61.5)	167.6(68.1)	147.4(59.9)
141×14	240(100)	187.4(82.2)	177.6(74.0)	167.2(70.0)	159.8(66.6)	159.6(66.5)	157.8(65.8)
146×13	240(100)	199.8(82.3)	197.4(82.3)	173.4(72.3)	143.4(59.8)	168.6(70.3)	138.8(57.8)
146×14	228(100)	187.4(82.3)	176.4(77.4)	162.4(71.2)	145.8(63.9)	151.2(66.3)	145.8(63.9)
142×14	251(100)	195.4(77.8)	211.2(84.1)	175.2(69.8)	155.8(62.1)	161.6(64.4)	153.8(61.3)
101×13	180(100)	144.8(80.4)	146.6(81.4)	126.4(70.2)	114.0(63.3)	118.2(65.7)	113.0(62.8)
98×13	188(100)	139.4(74.1)	152.6(81.2)	122.2(65.0)	115.6(61.5)	116.6(62.0)	116.2(61.8)
102×13	189(100)	157.8(83.5)	147.6(78.1)	136.2(72.1)	123.0(65.1)	124.6(65.9)	120.4(63.7)
105×13	174(100)	143.4(82.4)	123.6(71.0)	121.2(69.7)	112.4(64.6)	115.2(66.2)	111.4(64.0)
101×13	198(100)	150.8(76.2)	155.6(78.6)	130.4(65.9)	122.4(61.8)	124.6(62.9)	120.6(60.9)
105×14	170(100)	139.4(82.0)	141.6(83.3)	124.2(73.1)	109.8(64.6)	122.6(72.1)	108.8(64.0)
143×14	224(100)	179.4(81.0)	164.4(73.4)	162.4(72.5)	146.4(65.4)	148.2(66.2)	143.4(64.0)
142×14	187(100)	146.4(78.3)	147.6(78.0)	121.2(64.8)	117.8(63.0)	118.6(63.4)	116.8(62.5)
216×13	230(100)	188.4(81.9)	175.4(76.3)	190.4(82.8)	151.6(65.9)	163.2(71.0)	149.6(65.0)
144×14	207(100)	156.4(75.6)	159.4(77.0)	160.4(77.5)	148.0(71.5)	149.2(72.1)	143.6(69.4)
Reduction Ave.	–	20.3%	21.8%	29.4%	35.6%	33.5%	36.6%

[2] M. Potkonjak, M. B. Srivastava, and A. Chandrakasan, "Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination," *IEEE Trans. Computer-Aided Design*, vol. 15, no. 2, pp. 151–165, Feb. 1996.

[3] R. I. Hartley, "Optimization of canonic signed digit multipliers for filter design," in *Proc. 1991 IEEE Int. Symp. on Circuits and Systems (ISCAS 1991)*, Singapore, June 11–14, 1991, pp. 1992–1995.

[4] R. Paško, P. Schaumout, V. Derudder, S. Vernalde, and D. Ďuračková, "A new algorithm for elimination of common subexpressions," *IEEE Trans. Computer-Aided Design*, vol. 18, no. 1, pp. 58–68, Jan. 1999.

[5] Y. Jang and S. Yang, "Low-power CSD linear phase FIR filter structure using vertical common sub-expression," *Electron. Lett.*, vol. 38, no. 15, pp. 777–779, July 2002.

[6] A. P. Vinod, E. M-K. Lai, A. B. Premkumar, and C. T. Lau, "FIR filter implementation by efficient sharing of horizontal and vertical common subexpressions," *Electron. Lett.*, vol. 39, no. 2, pp. 251–253, Jan. 2003.

[7] Y. Takahashi and M. Yokoyama, "New cost-effective VLSI implementation of multiplierless FIR filter using common subexpression elimination," in *Proc. ISCAS 2005*, Kobe, Japan, May 23–26, 2005, pp. 845–848.

[8] 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*, vol. 51, no. 11, pp. 2215–2221, Nov. 2004.

[9] A. P. Vinod and E. M-K. Lai, "Comparison of the horizontal and the vertical common subexpression elimination methods for realizing digital filters," in *Proc. ISCAS 2005*, pp. 496–498.

[10] Y. Takahashi, T. Sekine, and M. Yokoyama, "70 MHz multiplierless FIR Hilbert transformer in 0.35 μm standard CMOS library," *IEICE Trans. Fundamentals*, vol. E90-A, no. 7, pp. 1376–1383, July 2007.

[11] N. Banerjee, J. H. Choi, and K. Roy, "A process variation aware low power synthesis methodology for fixed-point FIR filters," in *Proc. 2007 ACM/IEEE Int. Symp. on Low Power Design (ISLPED 2007)*, Portland, OR, Aug. 27–29, 2007, pp. 147–152.

[12] K. Suzuki, H. Ochi, and S. Kinjo, "A design of FIR filter using CSD with minimum number of registers," in *Proc. 1996 IEEE Asia Pacific Conf. on Circuits and Systems (APCCAS 1996)*, Seoul, Korea, Nov. 18-21, 1996, pp. 227–230.