Fine-Grained Critical Path Analysis and Optimization for Area-Time Efficient Realization of Multiple Constant Multiplications

Xin Lou, *Student Member, IEEE*, Ya Jun Yu, *Senior Member, IEEE*, and Pramod Kumar Meher, *Senior Member, IEEE*

***Abstract*—In this paper, critical path of multiple constant multiplication (MCM) block is analyzed precisely and optimized for high-speed and low-complexity implementation. A delay model based on signal propagation path is proposed for more precise estimation of critical path delay of MCM blocks than the conventional adder depth and the number of cascaded full adders. A dual objective configuration optimization (DOCO) algorithm is developed to optimize the shift-add network configuration to derive high-speed and low-complexity implementation of the MCM block for a given fundamental set along with a corresponding additional fundamental set. A genetic algorithm (GA)-based technique is further proposed to search for optimum additional fundamentals. In the evolution process of GA, the DOCO is applied to each searched additional fundamental set to optimize the configuration of the corresponding shift-add network. Experimental results show that the proposed GA-based technique reduces the critical path delay, area, power consumption, area delay product and power delay product by 32.8%, 4.2%, 5.8%, 38.3%, and 41.0%, respectively, over other existing optimization methods.**

***Index Terms*—Critical path, genetic algorithm, high-speed, multiple constant multiplication (MCM).**

I. Introduction

TYPICAL DSP algorithms involve large number of multiplications, and multipliers consume significant amount of area and computation time. It is therefore important to reduce the area and time complexity of implementations of multipliers. In some applications such as linear transformations and transposed form finite impulse response (FIR) filters as illustrated in Fig. 1, the same variable is multiplied by a set of constant coefficients, which are known *a priori*. Such structures are referred to as multiple constant multiplications (MCM) [1]. Efficient implementation of MCM is important for high-speed, low-complexity and low-power DSP systems. Typically the multiplications in an MCM block are realized by a network of adders (subtractors) and hardwired shifts with sharing of partial products across all the multiplications. The MCM problem is extensively studied and many different algorithms have been proposed by researchers to optimize the area consumption and computation time of the MCM block [1]–[13]. All these approaches can be put into two broad categories: i) the common subexpression elimination (CSE) technique proposed by Hartley [14], and ii) the graph-dependence (GD) algorithms of Bull and Horrocks [15].

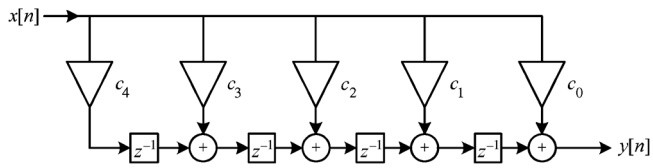


Fig. 1. Transposed structure of a FIR filter with 5 coefficients.

The CSE technique primarily searches for the most frequently occurred common subexpressions which could be maximally shared across the multipliers in the MCM block. Potkonjak *et al.* and Hartley are pioneers to explore the redundancy in MCM blocks using CSE technique. Potkonjak *et al.* [16] represented coefficients in signed digit (SD) form and used a recursive bipartite matching algorithm to identify the maximally-shared common subexpressions. In [17], Hartley expressed the coefficients in canonical signed digit (CSD) form and arranged them in a two-dimensional array to search for identical bit patterns in horizontal (intra-coefficient), vertical and oblique (inter-coefficient) directions. Thereafter, more CSE algorithms [2]–[8] with different common subexpression identification strategies are proposed to reduce the logical operators (LOs) and logic depth (LD). However, the disadvantage of CSE algorithms is that the performance of these algorithms depends on the number representation.

GD algorithms [1], [9]–[13], on the other hand, make no assumptions on the number representation, such that they offer more degrees of freedom to the optimization of the MCM problem. The idea of representing multiplier blocks with directed acyclic graphs (DAGs) was introduced in [15]. As illustrated in Fig. 2, vertices (or nodes) in the graph correspond to additions. Values assigned to vertices representing the results of additions are referred to as fundamentals. Edges in the graph correspond to shift operations, i.e., multiplications by a power-of-two term. Minimized adder graph (MAG) for single constant multiplications are generated by an exhaustive search of all possible graph topologies [10] and further used in the heuristic part of n-dimensional reduced adder graph (RAG-n) algorithm [12]. However, due to the high computation complexity of exhaustive search, the MAG lookup table (LUT) used in RAG-n contains constants only up to 12-bit. Thereafter, several heuristic algorithms [1], [13] with different additional

Manuscript received July 22, 2014; revised November 19, 2014; accepted November 20, 2014. Date of publication December 24, 2014; date of current version February 23, 2015. This paper was recommended by Associate Editor F. Clermidy.

X. Lou and Y. J. Yu are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: xlou001@e.ntu.edu.sg; eleyuyj@pmail.ntu.edu.sg).

P. K. Meher is with the School of Computer Engineering, Nanyang Technological University, Singapore 639798 (e-mail: ASPKMEHER@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2014.2377412

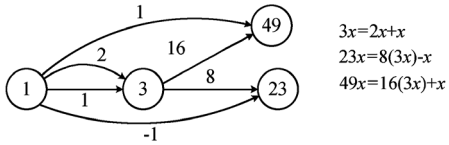


Fig. 2. Directed acyclic graph of constants multipliers of 23 and 49.

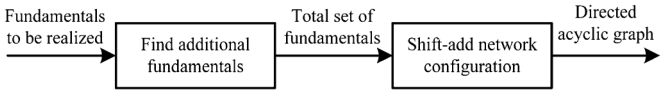


Fig. 3. Design path of MCM optimization at bit-level.

fundamental selection strategies have been proposed to overcome the limitation of the MAG LUT. The GD algorithms share the same optimal part and use different criteria to determine additional fundamentals. While the earlier GD algorithms [1], [9]–[13] focus on reducing the number of additional fundamentals (i.e., word-level adders), in some recent works [18]–[20], the bit-level information of word-level adders has been taken into consideration to minimize the number of full adders. To optimize the bit-level complexity, Kenny *et al.* [21] suggest a two-step design path as shown in Fig. 3. Besides finding the additional fundamentals as those word-level techniques, the interconnection of DAG, referred to as shift-add network configuration (second step in Fig. 3), is further optimized to minimize the FA cost.

Though bit-level information has been considered to estimate the area complexity of MCM blocks [18]–[20], the critical path delay (CPD) and time complexity are still measured imprecisely using adder-depth [3], [22]. This conventional metric is not significantly precise and sometimes is misleading. This is due to the fact that the delay of a word-level adder is closely related to its width, and the critical path of a shift-add network does not increase linearly with number of operands. For example, if it takes T ns to compute $a+b$, then the time needed to compute $a+b+c$ is $T+\delta$, instead of $2T$. Here, δ could be the delay just one or a few full adders, which depends on the adder structure. Based on such a concept, a recent research [23] has shown that the bit-level critical path optimization may achieve high-speed design without significant compromise of clock period of LMS adaptive filters. Shift-inclusive computation reordering, which minimizes the width of adders and reduces the CPD of FIR filters, is suggested in [24]. We find that by bit-level analysis of critical path based on signal propagation across the shift-add network, the CPD of the MCM block can be significantly reduced [25].

In this paper, we propose a delay model based on signal propagation path for fine-grained analysis of critical path of MCM blocks. Based on this model, a dual objective configuration optimization (DOCO) algorithm is developed to optimize the shift-add network configuration (the second step in Fig. 3) for reducing CPD as well as hardware complexity when additional fundamentals are already available. A genetic algorithm (GA)-based technique is further proposed to search for optimum additional fundamentals. Overall, in the evolution process of GA, the DOCO is applied to each searched additional fundamental set to optimize the configuration of the corresponding shift-add network.

The rest of the paper is organized as follows. Section II presents a brief review of background concepts and terminology used in the study of the MCM problem. In Section III, we discuss the proposed signal propagation path based delay model and the optimization of the shift-add network configuration using the proposed DOCO algorithm. In Section IV, the GA-based technique to search for optimum additional fundamentals is discussed. Experimental results are presented in Section V and conclusions are drawn in Section VI.

II. Background Concepts and Terminology

In this section, the bit-level complexity model for the MCM problem based on $\mathcal{A}$-operation [1] is discussed and the shift-add network configuration optimization technique for reducing hardware complexity is introduced. Prior to that, the definition of $\mathcal{A}$-operation is reviewed for the convenience of readers.

A. $\mathcal{A}$-Operation

The notation of $\mathcal{A}$-operation was first introduced in [1] for a parameterized representation of adds, subtracts and shifts for MCM implementation. The formal definition of $\mathcal{A}$-operation is as follows: let $l_1, l_2, r \geq 0$ be integers and let $s \in \{0,1\}$. An $\mathcal{A}$-operation is defined as

$$\mathcal{A}_p(u,v) = \left|2^{l_1}u + (-1)^s 2^{l_2}v\right| 2^{-r} \tag{1}$$

where u and v are two previously realized fundamentals. $p = \{l_1, l_2, r, s\}$ is the parameter set, where l_1, l_2 are, respectively, the bit positions through which u and v are shifted to left, r is the bit positions through which u and v both are shifted to right, and s indicates if the shifted fundamentals are added or subtracted. The result of an $\mathcal{A}$-operation is a fundamental from the given fundamental set to be realized or an additional fundamental that helps to realize the given fundamentals. We define $\{u, v, l_1, l_2, r, s\}$ as the configuration set of an $\mathcal{A}$-operation.

Since MCM algorithms deal only with multiplications by positive odd integers, certain constraints can be applied to the $\mathcal{A}$-operation. In the MCM problem, there are only two kinds of possible $\mathcal{A}$-operations: i) one of the input fundamentals, say u, is left shifted by one or more bit positions, while the value of the other input fundamental v does not change (otherwise the result is not odd). ii) both input fundamentals u and v are right shifted with the same amount of bit positions. This constraint can be expressed as:

$$\begin{cases} l_1 > 0, l_2 = r = 0 & \text{(only } u \text{ is left shifted), or} \\ l_1 = l_2 = 0, r > 0 & \text{(both } u \text{ and } v \text{ are right shifted)} \end{cases} \tag{2}$$

B. Bit-Level Complexity Model

Based on the $\mathcal{A}$-operation, a complexity model is recently proposed for ripple-carry adders (RCAs) [18]. The complexity model counts the number of full adders needed to realize a word-level adder. In this paper, we use the sign transformation technique of [21] to eliminate half adders in subtract operations. Half adders used for the last bit of word-level adders are counted as full adders. The number of full adders, denoted as N_{fa}, needed to realize an adder in the multiplier block is estimated to be:

$$N_{fa} = \begin{cases} \lceil \log_2(|f_i|) \rceil - l_1 + w_{in} & \text{for } l_1 > 0, l_2 = r = 0 \\ \lceil \log_2(|f_i|) \rceil + w_{in} & \text{for } l_1 = l_2 = 0, r > 0 \end{cases} \tag{3}$$

where f_i is the fundamental to be realized, and w_{in} is the word length of the input to the multiplier block. The total number of

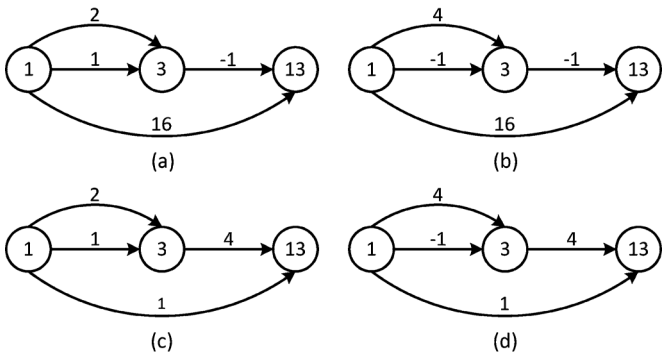


Fig. 4. Different ways to implement coefficient set {3, 13}.

full adders needed to implement the MCM block, denoted as $TotalFA$, is $TotalFA = \sum_{i=1}^{n} N_{fa_i}$, where n is the number of word-level adders.

C. Optimization of Shift-Add Network Configuration for Reduction of Hardware Complexity

For a given fundamental set and a corresponding additional fundamental set, several different DAGs with different FA costs can be constructed to represent the MCM block. In the following example we show that the shift-add network configuration of the DAG can be optimized to reduce the FA cost. Supposing that an 8-bit input is multiplied by a coefficient set {3, 13} using a shift-add network, there are 4 different options to realize the MCM block as shown in Fig. 4. The FA costs for DAGs of (a), (b), (c), and (d) are 17, 16, 19, and 18, respectively. Different shift-add network configurations result in different FA costs. In [21], a greedy algorithm was proposed to optimize the shift-add network of the DAG to reduce the number of full adders. It could be stated as follows: during the construction of the DAG, each time only the fundamental (no matter if it is a given or additional fundamental) which requires the fewest number of full adders according to (3) is realized. This algorithm is referred to as reduced full adder (RFA) algorithm in this paper.

III. Signal Propagation Path Based Critical Path Optimization

While the area complexity is estimated at bit-level, the CPD in our paper is estimated based on the signal propagation path, because in a 1-bit full adder, the delays for different paths may be different. For this reason, a fine-grained signal propagation path based delay model is proposed for the analysis of critical path of MCM blocks. Thereafter, a shift-add network configuration optimization scheme is developed for reducing CPD as well as hardware complexity.

A. Signal Propagation Path Based Critical Path Analysis

In the bit-level delay model proposed in [25], the computation time of the sum and carry-out of a full adder is assumed to be the same. However, a closer look at the full adder reveals that this is not the true case in real digital circuits. Let us take the full adder in Fig. 5(a) as an example. From the implementation perspective of a full adder, the input signals from a/b and c_{in} to the outputs s and c_{out} travel through different paths. This results different intrinsic delays for different signal paths. Furthermore, the loads driven by the outputs of a full adder impact the delays as well. Thus, instead of assuming the computation time of the

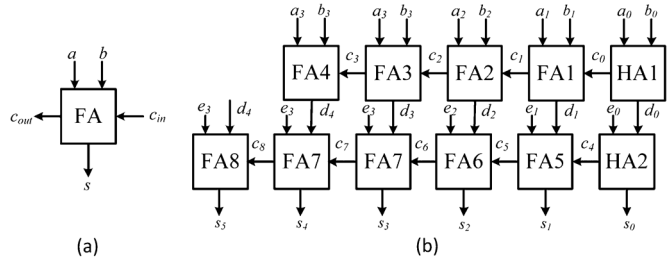


Fig. 5. (a) Full adder. (b) Two consecutive additions.

TABLE I
Propagation Delay Across Different Paths in Full/Half Adders

Structures	Signal Path	Delay (ns)	Normalized Delay (u.t.)
Full adder	$a/b \rightarrow s$	0.080	2.0
	$a/b \rightarrow c_{out}$	0.077	1.9
	$c_{in} \rightarrow s$	0.054	1.3
	$c_{in} \rightarrow c_{out}$	0.041	1.0
Half adder	$a/b \rightarrow s$	0.046	1.1
	$a/b \rightarrow c_{out}$	0.055	1.3

1 u.t. $= 0.041$ ns (the smallest signal path delay) is used to normalize all the delays.

sum and carry out to be the same, we treat the delays of different signal paths in a full/half adder individually.

In order to have a realistic estimation, the two consecutive adder structure shown in Fig. 5(b) is used to estimate the delays of different signal paths of full/half adders. This consecutive addition is a typical operation in a MCM block. The adder structure in Fig. 5(b) is synthesized using Synopsys Design Compiler with STM 65-nm CMOS technology. Place and route is performed using Cadence SOC Encounter. The full/half adder used in this work is the speed-optimized full/half adder from the standard cell library. The delay information of different signal paths of full adder and half adder is obtained from post layout simulation. The results are presented in Table I, where the normalization factor corresponds to 0.041 ns. As we can see from Table I, for a full adder, the propagation delay from two inputs to sum, denoted as $T_{FAab\rightarrow s}$, is almost equal to the delay from two inputs to carry-out, $T_{FAab\rightarrow c_{out}}$. The delay from two inputs to sum is almost two times as large as $T_{FAc_{in}\rightarrow c_{out}}$, which is the delay from carry-in to carry-out.

Based on the input-output path delay information in Table I, the propagation delay of each path in the MCM block is estimated by adding all the delay components along the signal path. We illustrate the proposed delay model in Fig. 6, which shows the addition of three 4-bit numbers a, b, and e. Ripple carry adder (RCA) is used in most of the works for the implementation of MCM blocks due to their regular implementation pattern. Thus, it is also considered to implement MCM blocks through out this work. (However, the idea of signal propagation path based critical path analysis studied in this work can be applied to other adder structures) Six bits are needed to represent the result of this addition and sign extensions are necessary to get the correct result. Suppose all the bits of a, b, and e are available at time 0. In the worst case, the computation time for bit s_5, which can be computed by adding all the delay components along the signal path marked by the dashed line, is:

$$T_{s_5} = T_{HAab\rightarrow c_{out}} + 3T_{FAc_{in}\rightarrow c_{out}} + T_{FAc_{in}\rightarrow s} + T_{FAab\rightarrow c_{out}} + T_{FAc_{in}\rightarrow s}$$

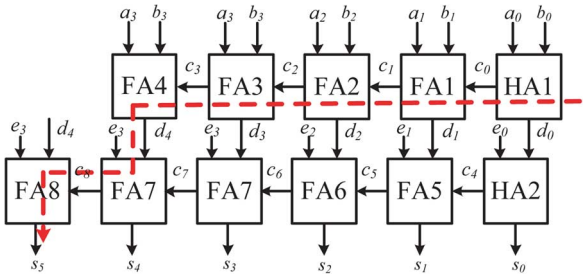


Fig. 6. Critical path of two consecutive additions.

$$= T_{HAab\rightarrow c_{out}} + 3T_{FAc_{in}\rightarrow c_{out}} + 2T_{FAc_{in}\rightarrow s} + T_{FAab\rightarrow c_{out}} \quad (4)$$

where $T_{HAab\rightarrow c_{out}}$ is the delay from inputs a_0 or b_0 to c_{out} of the first half adder, HA1. The term $3T_{FAc_{in}\rightarrow c_{out}}$ is the delay of the three consecutive full adders, FA1, FA2, and FA3. There are two terms of $T_{FAc_{in}\rightarrow s}$, one is the delay from c_3 to d_4 and the other is the delay from c_8 to s_5. The last term $T_{FAab\rightarrow c_{out}}$ is the computation time of FA7.

To estimate the critical path, the delay of each bit of the sum of a word-level adder is computed in an incremental manner. For the level one adder in Fig. 6, the delays of bits d_4, d_3, d_2, d_1, and d_0 can be computed as:

$$\begin{cases} T_{d_0} = \max(T_{a_0}, T_{b_0}) + T_{HAab\rightarrow s} \\ T_{c_0} = \max(T_{a_0}, T_{b_0}) + T_{HAab\rightarrow c_{out}} \\ T_{d_1} = \max(\max(T_{a_1}, T_{b_1}) + T_{FAab\rightarrow s}, T_{c_0} + T_{FAc_{in}\rightarrow s}) \\ T_{c_1} = \max(\max(T_{a_1}, T_{b_1}) + T_{FAab\rightarrow c_{out}}, T_{c_0} + T_{FAc_{in}\rightarrow c_{out}}) \\ T_{d_2} = \max(\max(T_{a_2}, T_{b_2}) + T_{FAab\rightarrow s}, T_{c_1} + T_{FAc_{in}\rightarrow s}) \\ T_{c_2} = \max(\max(T_{a_2}, T_{b_2}) + T_{FAab\rightarrow c_{out}}, T_{c_1} + T_{FAc_{in}\rightarrow c_{out}}) \\ T_{d_3} = \max(\max(T_{a_3}, T_{b_3}) + T_{FAab\rightarrow s}, T_{c_2} + T_{FAc_{in}\rightarrow s}) \\ T_{c_3} = \max(\max(T_{a_3}, T_{b_3}) + T_{FAab\rightarrow c_{out}}, T_{c_2} + T_{FAc_{in}\rightarrow c_{out}}) \\ T_{d_4} = \max(\max(T_{a_3}, T_{b_3}) + T_{FAab\rightarrow s}, T_{c_3} + T_{FAc_{in}\rightarrow s}) \end{cases}$$

Once the delays of the sum bits of level one adder are available, the delays of the sum bits of the level two adder can be computed in the same way. The CPD of the addition in Fig. 6 can thus be found to be:

$$T_c = \max(T_{s_5}, T_{s_4}, T_{s_3}, T_{s_2}, T_{s_1}, T_{s_0}) \quad (5)$$

To illustrate the precision and validity of the proposed delay model, some experimental results are shown in Table II. A parameterized configuration vector $(n, s_2, s_3 \ldots s_n, l)$ is used to represent the configurations of multi-operand shift-adder structures, where n is the number of operands to be added, s_i for $i = 2 \ldots n$ denotes the number of bit positions by which the i-th operand is shifted to the left, and l is the bit-width of operands. Taking the adder structure in Fig. 7 as an example, which shows the addition of three numbers of 8-bit width. The second number is shifted by one bit position to the left and the third is shifted by two bit positions to the left. Thus, the configuration vector in this case is (3, 1, 2, 8).

TABLE II
DELAY ESTIMATION OF DIFFERENT MODELS

Configuration	AD	Estimation of Delay (ns)			% Deviation of Delay Estimation	
		T_{CFA}	T_{PROP}	T_{POST}	ΔCFA	ΔPROP
(2, 1, 8)	1	0.369	0.381	0.389	5.1	2.0
(2, 1, 16)	1	0.697	0.705	0.719	3.1	1.9
(3, 1, 2, 8)	2	0.451	0.500	0.527	14.4	5.1
(3, 2, 1, 8)	2	0.410	0.459	0.474	13.5	3.1
(4, 1, 2, 3, 8)	3	0.533	0.619	0.663	19.6	6.6
(4, 3, 1, 2, 8)	3	0.492	0.541	0.604	18.5	10.4
(4, 2, 3, 1, 8)	3	0.492	0.578	0.563	12.6	2.7
(5, 1, 2, 3, 4, 8)	4	0.615	0.738	0.803	23.4	8.1
(5, 4, 1, 2, 3, 8)	4	0.574	0.660	0.709	19.0	6.9
(5, 3, 4, 1, 2, 8)	4	0.533	0.656	0.685	22.2	4.2
Average	–	–	–	–	15.2	4.6

T_{POST} is the delay estimation obtained from post layout simulation. T_{CFA} and T_{PROP}, are respectively, the delay estimations obtained by the number of CFAs and proposed signal path based delay model. The scaling factor for both T_{CFA} and T_{PROP} is 0.041. $\Delta\text{CFA} = |T_{POST} - T_{CFA}|/T_{POST} \times 100\%$, $\Delta\text{PROP} = |T_{POST} - T_{PROP}|/T_{POST} \times 100\%$

$$\begin{array}{cccccccccccc} & & & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ & & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & \\ + & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & & \\ \hline s_{10} & s_9 & s_8 & s_7 & s_6 & s_5 & s_4 & s_3 & s_2 & s_1 & s_0 \end{array}$$

Fig. 7. Addition of three binary numbers.

Listed in Table II are the estimated propagation delays obtained using different models. It can be found that the adder depth does not contain useful information of propagation delay, and our proposed delay model provides more precise estimate of propagation delay than the estimation in terms of the number of cascaded full adders (CFAs).

B. Optimization of Shift-Add Network Configuration for Reduction of CPD

For a given fundamental set and a corresponding additional fundamental set, different configurations of the shift-add network result in not only different hardware complexity as shown in Section II-C, but also different CPD. Let us take the same four different implementations of the multiplication of a 8-bit input with the coefficient set {3, 13} in Fig. 4 as an example. Using the delay information in Table I, the CPD for scenarios (a), (b), (c), and (d) are estimated to be 13.2 u.t., 12.2 u.t., 11.2 u.t., and 10.3 u.t., respectively. This implies that the configuration of a shift-add network could be optimized to reduce the CPD. The following shift-add network configuration optimization algorithm, named greedy reduced critical path delay configuration (GRCPDC) algorithm, is proposed to optimize the configuration of the shift-add network for reducing the CPD.

1) During the construction of the DAG, each time only one fundamental (no matter if it is a given or additional fundamental) which results in the lowest CPD among all the fundamentals that can be obtained from the previously realized fundamentals is realized. If there are more than one fundamentals that result in the same lowest CPD, the fundamental with the smallest value is selected.
2) Repeat step 1 until all fundamentals have been realized.

C. Optimization of Shift-Add Network Configuration for Reduction of CPD and Hardware Complexity

The idea presented in Section III-B is similar to the algorithm that we proposed in [25]. In [25] it was shown that sig-

nificant CPD saving can be achieved at the price of increasing the total number of full adders, the $TotalFA$. In this section, a dual-objective configuration optimization (DOCO) algorithm is proposed to optimize both CPD and $TotalFA$ of the MCM block for a given fundamental set with a corresponding additional fundamental set. The proposed DOCO algorithm is a hybrid algorithm based on three basic greedy optimization techniques: GRCPDC, RFA and the optimal part of RAG-n, referred to as ORAG-n. Here, GRCPDC generates MCM blocks with lowest CPD and higher $TotalFA$ while RFA and ORAG-n focus on reducing $TotalFA$ and the number of adders, respectively. The key idea of DOCO is to reduce the $TotalFA$ of GRCPDC without increasing CPD by introducing solutions of RFA and ORAG-n. Let C be the given fundamental set and A be the corresponding additional fundamental set, and let $U = C \cup A$.

Problem Definition: For a given C and a corresponding A, find a configuration of shift-add network to implement the MCM block with minimum CPD and $TotalFA$.

In this paper, minimization of CPD is the primary objective and minimization of $TotalFA$ is considered an objective with lower priority. In this multiple-objective optimization, the two objectives CPD and $TotalFA$ are linearly combined with a shifting factor to generate the *cost function (CF)* as:

$$CF = \text{CPD} \ll W + TotalFA \tag{6}$$

Here, "$\ll$" means left shift. W is a shifting factor and $W = \lceil \log_{10} TotalFA \rceil + 1$. CPD is shifted W digit positions to the left such that CPD does not affect the W least significant digits of CF and the W least significant digits of CF represent exactly the value of $TotalFA$. This ensures that CPD is optimized prior to $TotalFA$.

The DOCO algorithm is described as follows:

1) Apply GRCPDC, RFA, and ORAG-n on U and get three different shift-add networks. For each shift-add network, record the shift-add configuration of each coefficient of U.
2) Merge the configuration recordings of the three shift-add networks. If there are identical configurations for the same coefficient, keep one and remove the others. Compute the number of full adders needed for each configuration.
3) Solve the combination problem of choosing one configuration for each coefficient of U from the merged configuration set. If the combination number is less than N_{max}, go to step 4. Otherwise, go to step 5. The size of the combination problem increase exponentially with the number of coefficients. Thus, the algorithm uses N_{max} as a threshold to determine either optimal (step 4) or heuristic (step 5–7) method to be used in the following steps. The default value of N_{max} in our design is 25 000.
4) Traverse through all the combinations and test to find if for every combination there exists a feasible solution. If yes, compute the CF of this solution. Choose the combination with the smallest CF and terminate the optimization.
5) Use the shift-add network generated by GRCPDC as a base solution and compute its CF. For each coefficient, remove the configurations in the merged configuration set that need more number of full adders than the corresponding configuration in the base solution.
6) For each fundamental, select one configuration from the merged configuration set to replace the configuration in the base solution and generate a new solution. Test the new solution: if the new solution is a feasible one and its CF is smaller than the base solution, update the base solution with the new solution and delete the configuration in the merged configuration set; otherwise, keep the original base solution.
7) Repeat step 6 until no configuration left in the merged configuration set.

In DOCO, two different optimization strategies are used according to the size of the combination problem. In step 4, when the combnation problem size is less than N_{max}, exhaustive search is used to test all potential solutions to get an optimal solution while steps 5–7 start with a base solution of GRCPDC and improve it gradually with greedy replacement. N_{max} is used as a threshold to determine which strategy to use. In the rest of the paper, we use DOCO (U) to represent the CF value of the shift-add network of U after DOCO optimization.

We use the implementation of a coefficient set $C = \{3, 7, 9, 19, 5, 15, 31, 33, 23, 59, 21, 61, 99, 29, 71, 41, 195, 119, 37, 351, 321, 431\}$ from [21] as an example to illustrate the advantage of DOCO. Table III lists the results of different optimization methods for the coefficient set. We first use RAG-n to find the additional fundamentals and then optimize the shift-add network configuration using RFA, GRCPDC, and DOCO separately. As shown in Table III, RFA reduce the $TotalFA$ at the expense of increased CPD while GRCPDC reduce the CPD by sacrificing $TotalFA$. The last column of Table III shows that the $TotalFA$ of GRCPDC is further optimized by DOCO without increasing the CPD.

TABLE III
RESULTS OF DIFFERENT ALGORITHMS

Metric	RAG-n	RFA	GRCPDC	DOCO
Critical path delay (ns)	1.08	1.19	0.97	0.97
Number of full adders	202	190	236	210

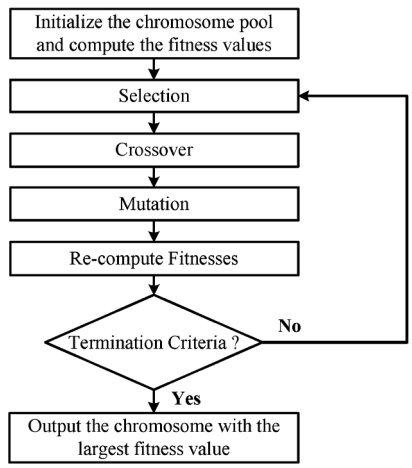


Fig. 8. Procedure of a typical GA.

IV. SEARCHING FOR ADDITIONAL FUNDAMENTALS USING GA-BASED APPROACH

Last section has shown that the proposed DOCO algorithm optimizes the shift-add network configuration when the additional fundamentals are already known (corresponding to the second step of the design path shown in Fig. 3). In this section, a GA-based technique is proposed to find the optimum additional fundamentals (corresponding to the first step of the

TABLE IV
FILTER SPECIFICATIONS

Filters	# Taps	L	Type	f_p	f_s
A [22]	25	12	LP	0.23	0.31
B [28]	60	15	LP	0.20	0.26
C [28]	60	15	LP	0.16	0.25
D [28]	80	15	LP	0.11	0.21
E [29]	60	11	LP	0.04	0.14
F [30]	63	12	LP	0.20	0.28
G [31]	108	9	LP	0.10	0.15
H [32]	695	12	LP	0.62	0.63

TABLE V
ESTIMATED CPD (u.t.)

Filters	C1	MBPG	NRSCSE	MINAS	RAG-n	RFA	Hcub	Hcubdc	Proposed GA
A	24.9	20.2	21.5	21.1	31.5	31.5	32.5	22.1	20.2
B	26.9	27.0	25.1	24.2	29.8	34.4	23.2	23.2	23.2
C	24.1	26.0	24.1	24.2	29.8	36.4	29.8	24.1	23.2
D	25.1	25.0	23.1	23.1	26.0	33.6	27.0	22.1	21.2
E	16.2	16.2	16.2	16.2	16.2	21.9	16.2	16.2	14.3
F	17.2	16.2	16.2	17.2	16.2	18.1	16.2	16.2	14.3
G	20.1	19.1	19.1	18.1	19.1	20.0	19.1	19.1	15.3
H	21.1	21.1	21.1	22.1	21.1	23.0	21.1	21.1	18.2

design path shown in Fig. 3) that will achieve better CPD and hardware complexity. GA was first introduced by John Holland in the 1970s [26], which constitutes a class of adaptive heuristic searches mimicking the process of natural evolution. In the past several decades, GA has been widely studied and applied in highly non-linear optimization problems. The procedure of a typical GA is shown in Fig. 8. It can be described in the following steps.

1) Initialize the potential solutions and encode them as chromosomes. Compute the *fitness* of each chromosome.
2) Apply three genetic operations (selection, crossover and mutation) to the chromosomes and re-compute the *fitness* of each chromosome.
3) Check if the termination criteria is reached. If not, repeat step 2; otherwise, terminate the algorithm.

This proposed GA-based algorithm could be applied on any MCM problem to derive an area-time efficient implementation. Since the delays of different signal propagation path of full/half adders are used in the design process, the technology to which the MCM block is going to be mapped need to be determined prior to the design process of the MCM block.

A. Encoding and Initial Population

In our problem, chromosomes are encoded to be a string of binary digits, representing the potential additional fundamentals of the given fundamental set. The length of chromosome is set to be $N \times \lceil \log_2(C_{max}) \rceil$ initially and increases adaptively. Here, N is the initial number of potential additional fundamentals which is set by users and C_{max} is the maximum value of the coefficient set. The algorithm increases the value of N adaptively if the initial value is not large enough to find a feasible solution. Our design experiments show that $N = 7$ is sufficiently large for coefficient sets consisting of up to 80 distinct positive odd integers.

In the search process of GA, initial population (chromosomes) plays an important role. In the proposed GA, initial population are the binary representation of randomly generated additional fundamentals which satisfy the following four conditions.

TABLE VI
ESTIMATED FA COST

Filters	C1	MBPG	NRSCSE	MINAS	RAG-n	RFA	Hcub	Hcubdc	Proposed GA
A	222	323	244	195	214	213	210	205	195
B	433	658	562	447	417	387	488	523	393
C	414	664	572	452	421	411	421	523	396
D	465	708	625	483	488	460	489	574	485
E	198	224	188	192	188	170	188	188	191
F	228	214	213	221	202	190	202	209	209
G	234	255	240	220	223	201	223	223	243
H	328	334	323	325	310	268	310	310	364

TABLE VII
REDUCTION (%) OF THE PRODUCT OF CPD AND $TotalFA$

Filters	C1	MBPG	NRSCSE	MINAS	RAG-n	RFA	Hcub	Hcubdc
A	40.3	65.6	33.2	4.5	71.1	70.3	73.3	15.0
B	27.1	93.9	53.9	18.0	35.6	45.3	23.5	26.8
C	9.4	89.3	51.2	20.0	37.6	64.1	37.6	38.2
D	13.5	72.1	40.4	8.5	23.4	50.3	28.4	23.4
E	17.4	32.9	11.5	13.9	11.5	36.3	11.5	11.5
F	30.6	15.4	14.9	26.6	9.0	14.5	9.0	12.7
G	26.5	31.0	23.3	7.1	14.6	8.1	14.6	14.6
H	3.9	5.8	2.3	7.8	−1.8	−7.5	−1.8	−1.8
Average	21.1	50.8	28.8	13.3	25.1	35.2	24.5	17.6

1) Number of additional fundamentals $< 2^{\lceil \log_2(C_{max}) \rceil}$.
2) No repeated values in one chromosome.
3) Not equal to any given fundamentals.
4) Initial additional fundamentals are generated according to the probability mass function $\mathcal{F}_X(x) = (1/x)/(\sum_x (1/x))$ where x is the potential additional fundamental, which is a positive odd integer less than $2^{\lceil \log_2(C_{max}) \rceil}$.

In the last condition, the probabilities of odd integers to be chosen as additional fundamentals are inversely proportional to their values because smaller additional fundamentals possess smaller word length (which is good for smaller CPD and $TotalFA$) and have a higher chance to be reused [12].

B. Fitness Function

With given fundamental set C and additional fundamental set A, $U = C \cup A$ is the union set of C and A. The CF value of U is defined to be:

$$CF = \begin{cases} \text{DOCO}(U) & \text{if } U \text{ is realizable} \\ penalty\ constant & \text{otherwise} \end{cases} \tag{7}$$

Given that a chromosome consists of N potential additional fundamentals, there exist $\sum_{k=0}^{N} \binom{N}{k}$ different combinations of additional set. For example, for 3 potential additional fundamentals a, b, and c, there exist $\sum_{k=0}^{3} \binom{3}{k} = 8$ different combinations of additional fundamental sets; they are $\emptyset$, $\{a\}$, $\{b\}$, $\{c\}$, $\{a, b\}$, $\{a, c\}$, $\{b, c\}$, and $\{a, b, c\}$. Thus, there are $\sum_{k=0}^{N} \binom{N}{k}$ corresponding CFs. The *penalty constant* in (7) is set to be larger than the maximum of all possible CFs. The objective value of the chromosome is defined as:

$$Obj = \min\left(CF_1, CF_2 \ldots CF_{\sum_{k=0}^{N} \binom{N}{k}}\right) \tag{8}$$

After the objective values of all chromosomes are computed, linear ranking based fitness assignment [27] is used to determine the *fitness* of each chromosome.

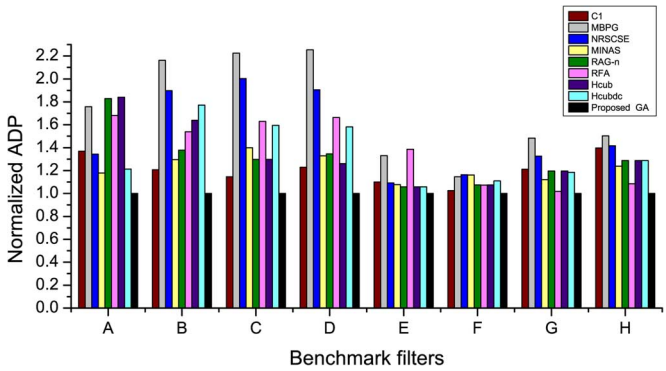


Fig. 9. Normalized ADP results. For each benchmark filter, the algorithms from left to right are: C1, MBPG, NRSCSE, MINAS, RAG-n, RFA, Hcub, Hcubdc, and the proposed GA-based technique, respectively.

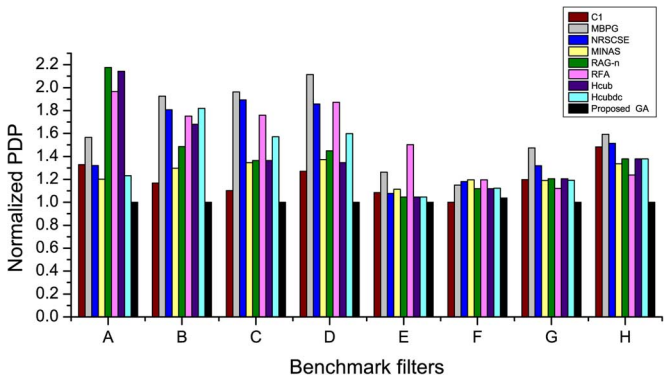


Fig. 10. Normalized PDP results. For each benchmark filter, the algorithms from left to right are: C1, MBPG, NRSCSE, MINAS, RAG-n, RFA, Hcub, Hcubdc, and the proposed GA-based technique, respectively.

TABLE VIII
AVERAGE SAVING OVER ALL BENCHMARKS

Algorithms Compared with	CPD (%)	Area (%)	Power (%)	ADP (%)	PDP (%)
C1	22.6	−0.7	−2.0	21.1	20.0
MBPG	35.0	29.3	20.9	73.2	62.6
NRSCSE	33.9	13.2	11.0	51.9	49.1
MINAS	25.9	−2.4	−0.4	22.5	25.2
RAG-n	33.6	−2.3	3.8	30.8	39.8
RFAI	48.3	−7.1	3.6	38.4	54.6
Hcub	33.9	−0.9	4.0	33.2	40.5
Hcubdc	29.5	4.2	5.2	35.0	36.5

CPD: critical path delay ADP: area delay product PDP: power delay product

TABLE IX
AVERAGE SAVING OF SELF-REALIZABLE AND SELF-UNREALIZABLE SETS

Filters Averaged	Algorithms Compared with	CPD (%)	Area (%)	Power (%)	ADP (%)	PDP (%)
A-D	C1	20.5	2.9	1.0	23.8	21.7
	MBPG	34.0	57.7	41.9	109.9	89.2
	NRSCSE	35.0	31.9	27.0	78.8	72.0
	MINAS	25.8	3.6	3.9	30.1	30.5
	RAG-n	38.2	5.6	16.3	46.3	61.9
	RFAI	59.3	2.6	15.8	62.8	83.7
	Hcub	38.9	8.4	16.7	51.0	63.4
	Hcubdc	30.0	18.2	19.3	54.0	55.6
E-H	C1	24.8	−4.3	−5.1	18.4	18.3
	MBPG	36.0	0.9	0.0	36.5	35.9
	NRSCSE	32.7	−5.5	−5.0	25.0	26.2
	MINAS	25.9	−8.4	−4.7	15.0	19.8
	RAG-n	29.0	−10.2	−8.7	15.4	17.7
	RFAI	37.4	−16.8	−8.6	14.1	25.4
	Hcub	29.0	−10.2	−8.7	15.4	17.7
	Hcubdc	29.0	−9.7	−8.9	15.9	17.5

CPD: critical path delay ADP: area delay product PDP: power delay product

C. Genetic Operations

During the evolution process, three basic genetic operations (selection, crossover and mutation) are applied to the chromosomes iteratively to generate offsprings. In the proposed GA, mating is exhaustive, i.e., every chromosome in a generation enters the crossover process. Multi-point crossover with crossover points in the middle of binary representations of additional fundamentals is applied in the proposed GA-based technique. Analogous to biological evolution processes, mutation operations increase the diversity of the population by introducing new genetic material to the gene pool. In our proposed GA, a fixed rate mutation is used.

V. EXPERIMENTAL RESULTS

In this section, numerical results are presented to demonstrate the effectiveness of the proposed GA-based technique for reducing the CPD as well as hardware complexity of MCM blocks. The efficiency of our proposed approach is compared with several CSE algorithms [3], [24] and GD-based algorithms [1], [12], [19], [21], [22] in terms of area, time complexity, and power consumption. We have taken only positive odd integers as coefficients since even values can be realized by shifting an odd integer to the left and multiplying by negative coefficients can be subtracted instead of being added. An input word-length of 8-bit is assumed for all the design examples.

In this experiment, eight commonly referenced benchmark FIR filter coefficient sets are tested. The specifications of all these benchmark filters are listed in Table IV, where L denotes the word-length of the filter coefficients, and # Taps is the number of filter taps. The information regarding types of filters is given in the Type column, where LP and HP, respectively, denote low-pass and high-pass filters. The last two columns list the passband edge f_{p} and the stopband edge f_{s} of the benchmark filters.

Table V and Table VI show respectively the estimated CPD and $TotalFA$ of the proposed GA-based technique and other competing methods. $TotalFA$ and CPD are estimated using the methods discussed in Section II-C and Section III-A, respectively. In general, the proposed GA-based technique performs well in terms of both CPD and $TotalFA$. From Table V, it can be found that the proposed GA-based technique outperforms other methods in terms of CPD for all benchmarks. The average improvements of CPD over C1 [22], MBPG [24], NRSCSE [3], MINAS [19] (with an adder depth constraint), RAG-n [12], RFA [21], Hcub [1], and Hcubdc [1] (Hcub with an adder depth constraint) are 17.8%, 14.2%, 11.9%, 11.9%, 25.4%, 44.6%, 23.0%, and 10.6%, respectively. The proposed GA-based technique achieves the minimum $TotalFA$ for benchmarks A and C. However, in some cases, the proposed GA-based technique may consumes slightly more number of full adders than other methods in order to lower the CPD. The average saving of $TotalFA$ over C1, MBPG, NRSCSE, MINAS, Hcub, and Hcubdc are 2.9%, 32.8%, 16.1%, 1.6%, 1.3%, and 7.2%, respectively, and is 0.7% and 7.2% worse than RAG-n and RFA, respectively. The comparison result of the product of CPD and $TotalFA$, which reflects the area-delay performance, is presented in Table VII. Due to the significant reduction of CPD, the average reduction of the product of CPD

TABLE X
DETAIL EXPERIMENTAL RESULTS BASED ON POST LAYOUT SIMULATION USING 65-nm TECHNOLOGY

Filters	Algorithms	CPD(ns)	Area(μm^2)	Power(μw)	ADP(μm^2*ns)	PDP(μw*ns)	CPD Saving(%)	Area Saving(%)	Power Saving(%)	ADP Saving(%)	PDP Saving(%)
A	C1	1.34	10748	1.25	14380.5	1.67	21.0	13.2	9.7	36.9	32.8
	MBPG	1.17	15727	1.68	18447.9	1.98	6.1	65.6	47.7	75.7	56.7
	NRSCSE	1.22	11619	1.37	14117.2	1.67	9.9	22.4	20.3	34.4	32.1
	MINAS	1.30	9550	1.17	12386.9	1.52	17.3	0.6	2.5	18.0	20.3
	RAG-n	1.75	10974	1.57	19204.9	2.74	58.2	15.6	37.5	82.9	117.5
	RFA	1.65	10669	1.50	17647.1	2.48	49.5	12.4	31.5	68.0	96.6
	Hcub	1.84	10499	1.47	19329.3	2.70	66.5	10.6	28.8	84.1	114.3
	Hcubdc	1.25	10217	1.25	12741.0	1.55	12.7	7.6	9.3	21.3	23.2
	Proposed GA	1.11	9495	1.14	10501.6	1.26	–	–	–	–	–
B	C1	1.47	20609	2.60	30275.1	3.82	18.9	1.6	−1.8	20.8	16.7
	MBPG	1.69	32024	3.73	54184.6	6.30	36.9	57.9	40.7	116.2	92.6
	NRSCSE	1.71	27753	3.45	47568.5	5.92	38.7	36.8	30.3	89.8	80.7
	MINAS	1.53	21272	2.79	32460.4	4.25	23.5	4.9	5.2	29.5	29.9
	RAG-n	1.71	20185	2.84	34557.3	4.86	38.5	−0.5	7.3	37.9	48.6
	RFA	1.99	19388	2.88	38601.5	5.73	61.1	−4.4	8.7	54.0	75.1
	Hcub	1.74	23624	3.17	41058.9	5.50	40.6	16.5	19.6	63.8	68.1
	Hcubdc	1.81	24497	3.29	44388.0	5.96	46.6	20.8	24.1	77.1	82.0
	Proposed GA	1.24	20281	2.65	25067.5	3.27	–	–	–	–	–
C	C1	1.35	19914	2.50	26903.5	3.38	11.7	2.7	−1.2	14.7	10.3
	MBPG	1.63	32015	3.69	52183.9	6.02	34.7	65.1	45.7	122.4	96.3
	NRSCSE	1.72	27278	3.37	47000.0	5.80	42.4	40.7	32.9	100.3	89.3
	MINAS	1.50	21954	2.76	32821.9	4.12	23.6	13.2	8.9	39.9	34.5
	RAG-n	1.45	20950	2.88	30461.6	4.18	20.2	8.1	13.5	29.8	36.4
	RFA	1.85	20704	2.92	38198.9	5.39	52.5	6.8	15.4	62.8	75.9
	Hcub	1.45	20950	2.88	30461.6	4.18	20.2	8.1	13.5	29.8	36.4
	Hcubdc	1.51	24818	3.20	37400.4	4.81	24.5	28.0	26.1	59.4	57.1
	Proposed GA	1.21	19388	2.53	23459.5	3.06	–	–	–	–	–
D	C1	1.44	22551	2.96	32540.6	4.27	30.5	−5.8	−2.6	22.9	27.1
	MBPG	1.75	34044	4.05	59678.7	7.10	58.5	42.2	33.4	125.4	111.4
	NRSCSE	1.65	30574	3.78	50446.7	6.24	49.2	27.7	24.5	90.5	85.8
	MINAS	1.54	22892	3.00	35184.3	4.61	39.0	−4.4	−1.2	32.9	37.3
	RAG-n	1.50	23737	3.24	35629.7	4.87	35.7	−0.8	6.8	34.6	44.9
	RFA	1.92	22889	3.27	44039.1	6.29	74.0	−4.4	7.6	66.3	87.2
	Hcub	1.42	23546	3.19	33387.6	4.52	28.2	−1.6	5.0	26.1	34.6
	Hcubdc	1.50	27830	3.57	41857.0	5.37	36.0	16.3	17.6	58.1	59.9
	Proposed GA	1.11	23937	3.04	26474.7	3.36	–	–	–	–	–
E	C1	0.80	9268	0.96	7423.3	0.77	15.4	-4.6	-6.1	10.1	8.4
	MBPG	0.83	10867	1.09	8976.1	0.90	19.0	11.8	6.1	33.1	26.2
	NRSCSE	0.80	9191	0.95	7371.4	0.77	15.6	−5.4	−6.8	9.3	7.7
	MINAS	0.82	8839	0.96	7265.6	0.79	18.4	−9.1	−6.0	7.7	11.4
	RAG-n	0.79	9050	0.94	7131.6	0.74	13.5	−6.9	−7.9	5.7	4.6
	RFA	1.12	8333	0.95	9349.3	1.07	61.7	−14.3	−7.0	38.6	50.3
	Hcub	0.79	9050	0.94	7131.6	0.74	13.5	−6.9	−7.9	5.7	4.6
	Hcubdc	0.79	9050	0.94	7131.6	0.74	13.5	−6.9	−7.9	5.7	4.6
	Proposed GA	0.69	9719	1.02	6745.2	0.71	–	–	–	–	–
F	C1	0.74	10927	1.08	8031.3	0.79	−1.5	4.1	−2.1	2.6	−3.5
	MBPG	0.90	9986	1.01	8977.6	0.91	20.5	−4.9	−8.0	14.7	10.9
	NRSCSE	0.90	10140	1.04	9115.9	0.93	20.5	−3.4	−5.5	16.4	13.8
	MINAS	0.87	10514	1.10	9095.0	0.95	16.0	0.2	−0.5	16.2	15.3
	RAG-n	0.88	9570	1.01	8421.8	0.89	18.0	−8.8	−8.4	7.6	8.0
	RFA	0.90	9312	1.05	8399.0	0.95	20.9	−11.3	−4.6	7.3	15.3
	Hcub	0.88	9570	1.01	8421.8	0.89	18.0	−8.8	−8.4	7.6	8.0
	Hcubdc	0.89	9759	1.00	8685.2	0.89	19.3	−7.0	−9.2	10.9	8.4
	Proposed GA	0.75	10496	1.10	7830.0	0.82	–	–	–	–	–
G	C1	0.97	10989	1.12	10692.6	1.09	29.9	−6.7	−7.8	21.2	19.8
	MBPG	1.06	12301	1.26	13088.1	1.34	42.1	4.4	3.8	48.3	47.5
	NRSCSE	1.02	11447	1.17	11710.2	1.20	36.6	−2.9	−3.4	32.7	32.0
	MINAS	0.95	10413	1.14	9892.0	1.08	26.8	−11.6	−6.1	12.1	19.1
	RAG-n	1.00	10562	1.09	10551.2	1.09	33.4	−10.4	−9.7	19.6	20.5
	RFA	0.92	9792	1.11	8989.3	1.02	22.6	−16.9	−8.5	1.9	12.1
	Hcub	1.00	10562	1.09	10551.2	1.09	33.4	−10.4	−9.7	19.6	20.5
	Hcubdc	0.99	10562	1.09	10445.6	1.08	32.0	−10.4	−9.7	18.4	19.3
	Proposed GA	0.75	11783	1.21	8825.6	0.91	–	–	–	–	–
H	C1	1.19	15762	1.75	18709.2	2.07	55.2	−9.9	−4.4	39.8	48.3
	MBPG	1.24	16166	1.79	20094.6	2.23	62.5	−7.6	-2.0	50.2	59.2
	NRSCSE	1.21	15666	1.75	18955.6	2.12	58.2	−10.4	−4.3	41.7	51.3
	MINAS	1.09	15219	1.72	16573.1	1.87	42.4	−13.0	−6.2	23.9	33.6
	RAG-n	1.16	14918	1.67	17230.5	1.93	51.0	−14.7	−8.7	28.8	37.8
	RFA	1.11	13139	1.57	14518.2	1.73	44.4	−24.9	−14.2	8.5	23.9
	Hcub	1.16	14918	1.67	17230.5	1.93	51.0	−14.7	−8.7	28.8	37.8
	Hcubdc	1.16	14918	1.67	17230.5	1.93	51.0	−14.7	−8.7	28.8	37.8
	Proposed GA	0.77	17492	1.83	13381.0	1.40	–	–	–	–	–

CPD: critical path delay. ADP: area delay product. PDP: power delay product.

and $TotalFA$ over C1, MBPG, NRSCSE, MINAS, RAG-n, RFA, Hcub, and Hcubdc are 21.1%, 50.8%, 28.8%, 13.3%, 25.1%, 35.2%, 24.5%, and 17.6%, respectively.

Beside the estimations, all the designs are further implemented in verilog HDL and synthesized using the Synopsys Design Compiler with the STM 65-nm CMOS library. Place and route is performed using Cadence SOC encounter. The detail information of CPD, area, power consumption, area-delay product (ADP), power delay product (PDP), along with the savings of CPD, area, power consumption, ADP, and PDP are

presented in Table X in Appendix. All the results are obtained from post layout simulation. Power consumptions are estimated at a frequency of 50 MHz. The proposed GA is found to provide minimum CPD for all benchmarks, which is in conformity with the estimated CPD listed in Table V. The proposed GA-based technique incorporated with the DOCO optimization performs much better than all the other methods for most of the benchmarks in terms of CPD, area and power consumption. This is due to the fact that both hardware complexity and CPD are optimized in the proposed GA-based technique. We can observe from Figs. 9 and 10 that our proposed GA-based technique outperforms all other methods in terms of both ADP and PDP for almost all the benchmarks.

In Table VIII, the average savings of CPD, area, power consumption, ADP, and PDP of the proposed GA-based technique for different benchmarks over the other methods are listed. Our proposed algorithm performs much better in CPD, ADP and PDP than all the other methods. The average improvements of CPD for different benchmarks over C1, MBPG, NRSCSE, MINAS, RAG-n, RFA, Hcub, and Hcubdc are 22.6%, 35.0%, 33.9%, 25.9%, 33.6%, 48.3%, 33.9%, and 29.5%, respectively. The solutions generated by the proposed GA-based technique have significant reduction of area and power consumption over solutions of MBPG, NRSCSE, while they consume a little bit more area and power than solutions of C1 and MINAS. However, due to the significant amount of CPD reduction, solutions generated by the proposed GA enjoy substantial reduction in both ADP and PDP. The average ADP reduction over C1, MBPG, NRSCSE, MINAS, RAG-n, RFA, Hcub, and Hcubdc are 21.1%, 73.2%, 51.9%, 22.5%, 30.8%, 38.4%, 33.2%, and 35.0% respectively and the average PDP reduction is 20.0%, 62.6%, 49.1%, 25.2%, 39.8%, 54.6%, 40.5%, and 36.5% respectively.

In MCM problems, the given fundamental set that need to be implemented can be classified into two categories. One is the fundamentals sets that can be realized using shift-add operation without additional fundamentals, i.e., the given fundamental sets can be realized with the optimal part of RAG-n, while the other is the fundamental sets that can not be realized using shift-add operations without additional fundamentals. We refer to the former as self-realizable sets and the later as self-unrealizable sets. In Table IX, sets A-D are self-unrealizable and sets E-H are self-realizable. Table IX shows the comparison results of self-realizable sets and self-unrealizable sets. Generally, the proposed GA-based technique performs better for self-unrealizable sets. This is because for self-unrealizable sets, it is necessary for all MCM algorithms to find valid additional fundamentals. For the proposed GA-based technique, both CPD and $TotalFA$ have been taken into consideration during the searching process of additional fundamentals, and the evolution property of GA leads to a better set of additional fundamentals. Nevertheless, for self-realizable sets, the other GD-based algorithms just use the optimal part of RAG-n to realize the set while the proposed GA still try to search for additional fundamentals that can lower the critical path, which leads to more area and power consumption.

VI. Conclusion

In this paper, the critical path of MCM blocks are analyzed based on the signal path and a fine-grained delay model for CPD estimation is proposed. Based on precise estimate of critical path, we have proposed an algorithm named DOCO to optimize the shift-add network configuration of MCM blocks for the reduction of CPD and hardware complexity subject to an additional fundamental set. In order to find the optimum additional fundamentals for a given fundamental set, a GA-based search method is proposed. The DOCO algorithm is adopted in the proposed GA-based technique to optimize the shift-add network configurations. Experimental results show that solutions generated by the proposed GA-based technique outperforms existing algorithms in terms of CPD, area, power consumption, ADP and PDP. The CPD, area, power, ADP, and PDP are reduced by 32.8%, 4.2%, 5.8%, 38.3%, and 41.0%, respectively, in average over the existing algorithms.

Appendix

The detail experimental results for all the design examples are presented in Table X.

References

[1] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Trans. Algorithms*, vol. 3, no. 2, p. 11, Sep. 2007.

[2] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova, "A new algorithm for elimination of common subexpressions," *IEEE Trans. Comput.-Aided Design Integr. Circuit Syst.*, vol. 18, no. 1, pp. 58–68, Jan. 1999.

[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, Analog Digit. Signal Process.*, vol. 49, no. 3, pp. 196–203, Mar. 2002.

[4] F. Xu, C. H. Chang, and C. C. Jong, "Contention resolution algorithm for common subexpression elimination in digital filter design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 10, pp. 695–700, Oct. 2005.

[5] F. Xu, C. H. Chang, and C. C. Jong, "Contention resolution algorithm for common subexpression elimination in digital filter design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 10, pp. 695–700, Oct. 2005.

[6] O. Gustafsson and L. Wanhammar, "Ilp modelling of the common subexpression sharing problem," in *Proc. IEEE 9th Int. Conf. Electron.,Circuits Syst.*, Sep. 2002, vol. 3, pp. 1171–1174.

[7] A. Yurdakul and G. Dundar, "Multiplierless realization of linear DSP transforms by using common two-term expressions," *J. VLSI Signal Process.*, vol. 22, pp. 163–172, Sep. 1999.

[8] L. Aksoy, E. da Costa, P. Flores, and J. Monteiro, "Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications," *IEEE Trans. Comput.-Aided Design Integr. Circuit Syst.*, vol. 27, no. 6, pp. 1013–1026, Jan. 2008.

[9] D. R. Bull and D. H. Horrocks, "Primitive operator digital filters," *IEE Proc. G*, vol. 138, pp. 401–412, Jun. 1991.

[10] A. G. Dempster and M. D. Macleod, "Constant integer multiplication using minimum adders," in *IEE Proc. Inst. Elec. Eng. Circuits, Devices, Syst.*, Oct. 1994, vol. 141, no. 5, pp. 407–413.

[11] K. Johansson and O. Gustafsson, "Extended results for minimum-adder constant integer multipliers," in *Proc. IEEE Int. Symp. Circuits Syst.*, Scottsdale, AZ, USA, May 2002, vol. 1, pp. 73–76.

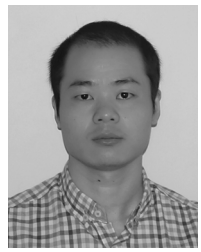
[12] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 42, no. 9, pp. 569–577, Sep. 1995.

[13] F. Xu, J. J. Chen, C. H. Chang, and C. C. Jong, "A modified reduced adder graph algorithm for multiplier block minimization in digital filters," in *Proc. IEEE Asia-Pacific Conf. Circuits Syst.*, Dec. 2004, vol. 2, pp. 705–708.

[14] R. I. Hartley, "Optimization of canonic signed digit multipliers for filter design," in *Proc. IEEE Int. Symp. Circuits Syst.*, Singapore, Jun. 1991, vol. 4, pp. 1992–1995.

[15] D. Bull and D. Horrocks, "Primitive operator digital filters," in *IEE Proc. Inst. Elec. Eng. Circuits, Devices, Syst.*, Jun. 1991, vol. 138, no. 3, pp. 401–412.

[16] M. Potkonjak, M. B. Srivastava, and A. P. Chandrakasan, "Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 15, no. 2, pp. 151–165, Feb. 1996.
[17] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 43, no. 10, pp. 677–688, Oct. 1996.
[18] K. Johansson, O. Gustafsson, and L. Wanhammar, "A detailed complexity model for multiple constant multiplication and an algorithm to minimize the complexity," in *Proc. Eur. Conf. Circuit Theory Design*, Cork, Ireland, Aug. 28–Sep. 2 2005, vol. 3, pp. 465–468.
[19] L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Finding the optimal tradeoff between area and delay in multiple constant multiplications," *Microprocess. Microsyst.*, vol. 35, no. 8, pp. 729–741, Nov. 2011.
[20] Y. Pan and P. K. Meher, "Bit-level optimization of adder-trees for multiple constant multiplications for efficient FIR filter implementation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 2, pp. 455–462, Feb. 2014.
[21] K. Johansson, O. Gustafsson, and L. Wanhammar, "Bit-level optimization of shift-and-add based FIR filters," in *Proc. IEEE Int. Conf. Electron. Circuits Syst.*, Marrakech, Morocco, Dec. 2007, vol. 3, pp. 713–716.
[22] A. G. Dempster, S. S. Dimirsoy, and I. Kale, "Designing multiplier blocks with low logic depth," in *Proc. IEEE Int. Symp. Circuits Syst.*, Scottsdale, AZ, USA, May 2002, vol. 5, pp. 773–776.
[23] P. K. Meher and S. Y. Park, "Critical-path analysis and low-complexity implementation of the LMS adaptive algorithm," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 3, pp. 778–788, Mar. 2014.
[24] C. H. Chang, J. J. Chen, and A. P. Vinod, "Information theoretic approach to complexity reduction of FIR filter design," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 8, pp. 2310–2321, Sep. 2008.
[25] X. Lou, Y. J. Yu, and P. K. Meher, "High-speed multiplier block design based on bit-level critical path optimization," in *Proc. IEEE Int. Symp. Circuits Syst.*, Melbourne, Australia, Jun. 2014, vol. 3, pp. 465–468.
[26] J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI, USA: Univ. Michigan Press, 1975.
[27] J. E. Baker, "Adaptive selection methods for genetic algorithms," in *Proc. ICGA 1*, 1985, pp. 101–111.
[28] L. Aksoy, E. O. Gunes, and P. Flores, "An exact breadth-first search algorithm for the multiple constant multiplications problem," in *Proc. IEEE NORCHIP Conf.*, Tallinn, Estonia, Nov. 2008, pp. 41–46.
[29] A. Shahein, Q. Zhang, N. Lotze, and Y. Manoli, "A novel hybrid monotonic local search algorithm for FIR filter coefficients optimization," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 3, pp. 616–627, Mar. 2012.
[30] Y. C. Lim and S. Parker, "Discrete coefficient FIR digital filter design based upon an LMS criteria," *IEEE Trans. Circuits Syst.*, vol. CAS-30, no. 10, pp. 723–739, Oct. 1983.
[31] D. L. Maskell, "Design of efficient multiplierless FIR filters," *IET Circuits, Devices Syst.*, vol. 1, no. 2, pp. 175–180, Apr. 2007.
[32] C. H. Chang and M. Faust, "On 'a new common subexpression elimination algorithm for realizing low-complexity higher order digital filters'," *IEEE Trans. Comput.-Aided Design Integr. Circuit Syst.*, vol. 29, no. 5, pp. 844–848, May 2010.



Xin Lou (S'14) received the B.Eng. degrees in electronic information technology & instruments from Zhejiang University, Hangzhou, China, in 2010, and the M.Sc. degree in electrical engineering from Royal Institute of Technology, Sweden, in 2012. He is currently working towards the Ph.D. degree at Nanyang Technological University, Singapore.

His research interests include digital filter design and implementation, and digital signal processing for communications.

Ya Jun Yu (S'99–M'05–SM'09) received the B.Sc. and M.Eng. degrees in biomedical engineering from Zhejiang University, Hangzhou, China, in 1994 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2004.

From 1997 to 1998, she was a Teaching Assistant with Zhejiang University. She joined the Department of Electrical and Computer Engineering, National University of Singapore as a Post Master Fellow in 1998 and remained in the same department as a Research Engineer until 2004. She joined the Temasek Laboratories at Nanyang Technological University as a Research Fellow in 2004. Since 2005, she has been with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, where she is currently an Assistant Professor. Her research interests include digital signal processing and VLSI circuits and systems design.

Dr. Yu has served as an associate editor for Circuits Systems and Signal Processing since 2009 and for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II for 2010–2013, respectively.

Pramod Kumar Meher (SM'03) received the B.Sc. (Honours) and M.Sc. degree in physics, and the Ph.D. degree in science from Sambalpur University, India, in 1976, 1978, and 1996, respectively. Currently, he is a Senior Research Scientist with Nanyang Technological University, Singapore. Previously, he was a Professor of Computer Applications with Utkal University, India, from 1997 to 2002, and a Reader in electronics with Berhampur University, India, from 1993 to 1997.

His research interest includes design of dedicated and reconfigurable architectures for computation-intensive algorithms pertaining to signal, image and video processing, communication, bio-informatics and intelligent computing. He has contributed more than 200 technical papers to various reputed journals and conference proceedings.

Dr. Meher has served as a speaker for the Distinguished Lecturer Program (DLP) of IEEE Circuits Systems Society during 2011 and 2012 and Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS during 2008 to 2011, and Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS during 2012–2013. Currently, he is serving as Associate Editor for the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, and *Journal of Circuits, Systems, and Signal Processing*. Dr. Meher is a Fellow of the Institution of Electronics and Telecommunication Engineers, India. He was the recipient of the Samanta Chandrasekhar Award for excellence in research in engineering and technology for 1999.