IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 20, 2021, accepted February 19, 2021, date of publication February 24, 2021, date of current version March 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3061759

Design of Very High-Speed Pipeline FIR Filter Through Precise Critical Path Analysis

SU MIN CHO[1], PRAMOD KUMAR MEHER[2], (Senior Member, IEEE), LUONG TRAN NHAT TRUNG[3], HYO JIN CHO[1], AND SANG YOON PARK[1], (Member, IEEE)

[1]Department of Electronics Engineering, Myongji University, Yongin 17058, South Korea
[2]Sandhaan Labs Private Limited, Bhubaneswar 751016, India
[3]Solution Group, Synopsys, Da Nang 550000, Vietnam

Corresponding author: Sang Yoon Park (sypark@mju.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government through MSIT under Grant 2019R1F1A1060820.

ABSTRACT In this paper, we propose a new hardware architecture of a very high-speed finite impulse response (FIR) filter using fine-grained seamless pipelining. The proposed full-parallel pipeline FIR filter can produce an output sample in a few gate delays by placing the pipeline registers not only in between components, but also across the components. A precise critical path analysis at the gate level allows to create an appropriate pipelining strategy depending on the throughput requirement. This paper also presents two alternative architectures, each offering different trade-offs in terms of area and throughput rate. The proposed FIR filters are synthesized to measure the maximum throughput and the balance between complexity and speed. The synthesis results show that the proposed fully pipelined FIR filter supports up to throughput of 1.8 Giga samples per second and offers 73.5% less area-delay product (ADP) than the existing systolic designs. Also, the proposed single multiplier-accumulator (MAC) based FIR filter has 3 times higher throughput and 26.0% less area with 75.8% less ADP compared to the existing design.

INDEX TERMS Finite impulse response filters, digital filters, pipeline, Wallace tree, critical path, Booth multiplier.

I. INTRODUCTION

There is an increasing demand for high-speed architectures for digital signal processing (DSP) algorithms, especially in the fields of fiber optic communication [1], ultra high-definition video analysis [2], [3], high-speed convolutions in neural networks [4], [5] etc. Implementation with integrated circuits such as application-specific integrated circuits (ASICs) and field-programmable gate arrays (FPGAs) is essential to meet the speed requirements of real-time applications. In particular, to achieve very high throughput over Giga samples per second (GSPS), extremely fine pipelining or a large number of parallel processing units are required [6]–[8]. There have been a lot of research in recent years to speed up basic arithmetic operators such as adders [9]–[11], multipliers [12], [13], compressors [14], [15], non-linear converters [16], trigonometric operators using coordinate rotation digital computer (CORDIC) [6], [17], [18] etc. Acceleration of digital finite impulse response (FIR) filter is also crucial as it is essential in almost all DSP applications. In addition, the design methodology for fast FIR filters can also provide a good strategic solution for speed-up of the entire digital system, which is basically based on the successive computation of arithmetic operations such as addition and multiplication.

The simplest way to get a high-throughput FIR filter is to use a large number of multiplier-accumulator (MAC) units to increase the level of parallelism [19]–[21]. It is also possible to optimize individual blocks by modifying the Booth encoding for multipliers to reduce the number of partial products (PP's) [12], to optimize an adder network [9]–[11], and to employ fast adders such as a carry look ahead adder [11], [22], [23]. Adopting one of the well-known architectural options such as distributed arithmetic (DA) can also be a good solution [7], [22], [24], [25]. Specifically, DA or a variant thereof, has been proven to optimize both area and speed, especially for higher order FIR filters and adaptive filters. The systolic approach can also be used to dramatically increase

The associate editor coordinating the review of this manuscript and approving it for publication was Tianhua Xu.

throughput by having the effect of increasing the number of pipelining stages [26], [27]. Seamless pipelining offers a different pipelining option without significant structural changes by inserting pipelining registers across arithmetic components [8], [11], which, however, do not provide practical solutions.

To maximize the parallelism of the FIR filter, it is necessary to use the same number of MAC units as the taps of the FIR filter, which is called a reference full-parallel FIR filter (FPFF) in this paper. However, with the typical pipelining method, the throughput of the FIR filter is limited by the propagation delay of the multiplier and adder, even in the full-parallel architecture [24]. In this paper, we redesign the FIR filter in a way that can fully utilize the seamless pipelining at the gate level [8]. For the purpose, we create a new pipeline architecture of a direct-form FIR filter based on the modified Booth encoder (MBE) [12], Wallace reduction tree (WRT) [10], [11] and hierarchical compressor array network [14]. Precise critical path analysis is provided at the gate level to find the optimal number of pipelining stages and register locations that minimize area while meeting specified timing constraints.

If the timing constraint becomes loose enough that a full-parallel architecture is not required, then it may be necessary to reduce the number of processing elements. In this paper, we propose two alternative architectures by modifying the reference FPFF: single-MAC FIR filter (SMFF) and folded FIR filter (FDFF). They compensate for the increase in propagation delay due to recursive computations by appropriately using compressors designed to prevent carry propagation and separate clock sources. In summary, the contribution of this paper is as follows:

- A novel high-speed full-parallel architecture of the FIR filter based on MBE and Wallace tree is designed at the gate level.
- A hierarchical Wallace tree network is employed to facilitate the design and pipelining procedure.
- A method to find the critical path of the proposed architecture is provided.
- A propagation delay of the proposed FIR filter is formulated in terms of unit gate delay to simplify the design process.
- A novel design method for an optimal pipeline architecture suitable for the given timing constraint is proposed.
- Practical design examples based on the proposed structure and analysis are provided with the synthesis results.
- Two alternative architectural options are proposed to reduce the number of MACs by using bit-plane pipelining with compressor arrays.

The rest of this paper is organized as follows. In the next section, a new architecture of the reference full-parallel FIR filter is designed. The detailed algorithm and architecture of MBE and hierarchical WRT network are also described. Section III proposes a precise critical path analysis of the proposed reference FPFF, and its optimal pipelining strategy for a given timing constraint. The design examples are also given in this section. Section IV presents alternative two architectural options to balance between area and throughput by reducing the number of MACs. In Section V, the proposed and existing architectures are synthesized and compared. In Section VI, a discussion on the simulation results is described. Section VII introduces the main DSP applications of the proposed filter and future research directions. Finally, conclusions are given in Section VIII.

II. REFERENCE FULL-PARALLEL ARCHITECTURE

A K-tap FIR filter produces the output sample $y(n)$ using the K most recent input samples of $x(n), x(n-1),\ldots,x(n-K+1)$ as

$$y(n) = \sum_{k=0}^{K-1} w(k)x(n-k), \tag{1}$$

where $w(k)$ for $0 \le k \le K-1$ represents the $(k+1)$-th FIR filter coefficient. The K-tap FIR filter needs to perform K multiplications and $(K-1)$ additions to obtain one output sample $y(n)$. Note that the maximum throughput can be achieved when the K multipliers and the $(K-1)$ adders are used in parallel as shown in Fig. 1. This type of full-parallel FIR filter (FPFF) generates one output sample per clock period.

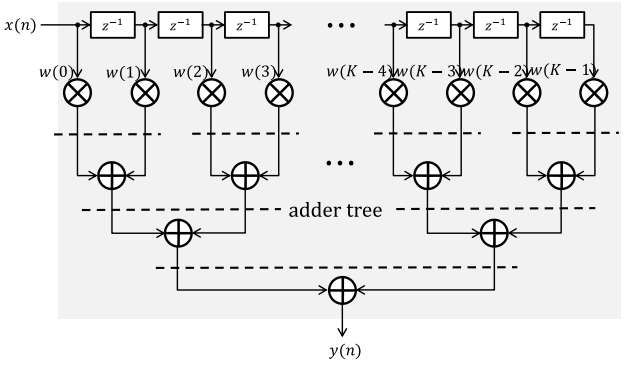


FIGURE 1. Structure of K-tap direct-form full-parallel FIR filter assuming K is a multiple of 4.

A lot of architectures have been proposed for the hardware implementation of FIR filter to improve area, throughput, and power consumption, and to find their optimal trade-offs. The history of the implementation of FIR filters and their pros and cons have already been highlighted in many previous articles [19], [20], [22], [25], [27], [28], so they are not repeated in this paper. Instead, we want to use a full-parallel implementation as a reference design for further discussion in order to target very high throughput applications. The block diagram of the proposed architecture of the reference FPFF, especially for the case of K-taps, m-bit inputs, and m-bit coefficients, is shown in Fig. 2. It consists of modified Booth encoders, a hierarchical Wallace reduction tree (WRT) network, and a ripple carry adder (RCA).

A Booth encoder is widely used as a multiplier for the FIR filter [29], [30], especially, modified Booth encoder (MBE)

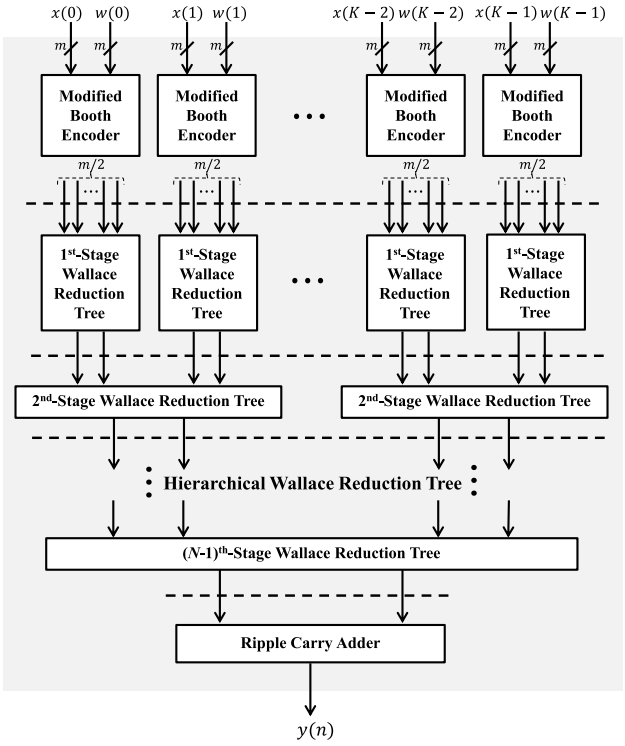


FIGURE 2. Structure of K-tap reference full-parallel FIR filter.

TABLE 1. Encoding to obtain $m/2$ partial products using m-bit MBE [12].

Expressions for the partial product bits of MBE
$neg_0 = w_1,\ neg_i = w_{2i+1} \cdot (\overline{w_{2i}} + \overline{w_{2i-1}})$ for $1 \le i \le m/2-1$
$\overline{one_0} = \overline{w_0},\ \overline{one_i} = \overline{w_{2i} \oplus w_{2i-1}}$ for $1 \le i \le m/2-1$
$\overline{two_0} = \overline{w_1} \cdot \overline{w_0}$ $\overline{two_i} = \overline{(\overline{w_{2i+1}} \cdot w_{2i} \cdot w_{2i-1}) + (w_{2i+1} \cdot \overline{w_{2i}} \cdot \overline{w_{2i-1}})}$ for $1 \le i \le m/2-1$
$na_{ij} = \overline{x_j \oplus w_{2i+1}}$ for $0 \le i \le m/2-1, 0 \le j \le m-1$, $na_{im} = na_{i(m-1)}$ for $0 \le i \le m/2-1$
$p_{ij} = \overline{(na_{i(j-1)} + \overline{two_i}) \cdot (na_{ij} + \overline{one_i})}$,   for $0 \le i \le m/2-1, 1 \le j \le m$,
$\tau_i = \overline{\overline{one_i} + \overline{x_0}}$ for $0 \le i \le m/2-1$
$c_i = neg_i \cdot (\overline{one_i} + \overline{x_0})$ for $0 \le i \le m/2-2$
$\bar{\epsilon} = \begin{cases} x_1, & \text{if } \overline{x_0 \cdot w_{m-1}} = 0 \\ \overline{x_1}, & \text{otherwise} \end{cases}$
$t_1 = \overline{(\overline{one_{m/2-1}} + \bar{\epsilon}) \cdot (\overline{two_{m/2-1}} + \overline{x_0})}$
$\bar{d} = \overline{(\overline{w_{m-1}} + x_0) \cdot (w_{m-3} + x_1) \cdot (w_{m-2} + x_1) \cdot (w_{m-2} + w_{m-3})}$
$s_i = p_{im}$ for $0 \le i \le m/2-1$
$\alpha_2 = \overline{\overline{s_0 \cdot \bar{d}}}$
$\alpha_1 = s_0 \cdot \bar{d}$
$\alpha_0 = \overline{s_0 \oplus \bar{d}}$

proposed by Kuang *et al.* [12] is known as one of the most efficient multipliers among the existing Booth encoder variants since it results in the smallest number of partial products (PP). In the proposed reference K-tap FPFF, a total K MBEs are used, one per each tap. Assuming m is even, multiplication

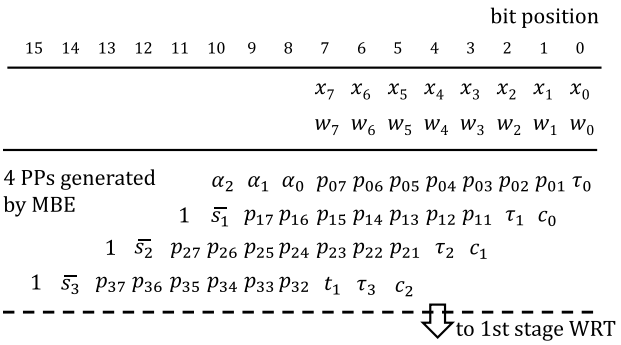


FIGURE 3. Four partial products generated by an 8-bit MBE.

TABLE 2. The required number of WRT levels according to the number of partial products.

Num. of Partial Products	Num. of Levels
3	1
4	2
$5 \le n \le 6$	3
$7 \le n \le 9$	4
$10 \le n \le 13$	5
$14 \le n \le 19$	6
$20 \le n \le 28$	7
$29 \le n \le 42$	8
$43 \le n \le 63$	9

of the m-bit input by the m-bit coefficient using MBE yields a total $m/2$ PPs. Table 1 lists how each bit of $m/2$ PPs is encoded in order, where x_i and w_i denote the i-th bit of the m-bit input x and the m-bit coefficient w, respectively. For more detailed algorithm and circuit diagrams to implement this, refer to [12]. Fig. 3 shows four PPs of the 8-bit input x and 8-bit coefficient w.

The Wallace reduction tree (WRT) is used to implement the adder tree of the FPFF in Fig. 1. WRT is responsible for reducing several PPs generated by MBE into two PPs. The main task of WRT is to group 2 or 3 bits at the same bit position and use a half adder (HA) or a full adder (FA) to reduce to 2 bits over 2 consecutive bit positions. This process continues until two PPs are left over several levels within the WRT. The required number of levels depends on the number of PPs as listed in Table 2. Since the propagation delay of each level is at most FA delay, WRT is suitable for speeding up the additions with the pipelining inside the adder tree of the FIR filter. This is because the operation of one FA does not depend on the result of other FAs or HAs in the same level.

Since one m-bit MBE produces $m/2$ PPs, the total number of PPs in K taps is equal to $m/2 \times K$. Furthermore, as the number of bits or taps of the filter increases, the number of PPs increases, which makes the design of the WRT complicated. In order to avoid the design difficulties, a hierarchical WRT network can be used in the proposed architecture as shown in Fig. 2. Specifically, the role of the first stage WRT is to reduce

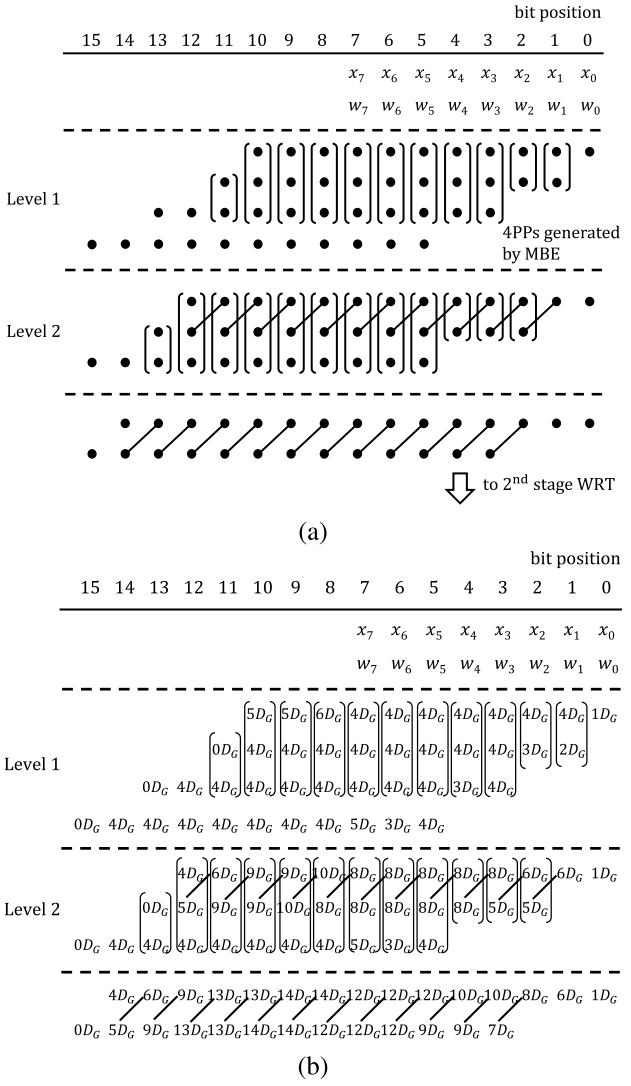


FIGURE 4. **(a) Dot diagram of MBE and the first stage WRT when $m = 8$. (b) Accumulation of the propagation delays without pipelining.**

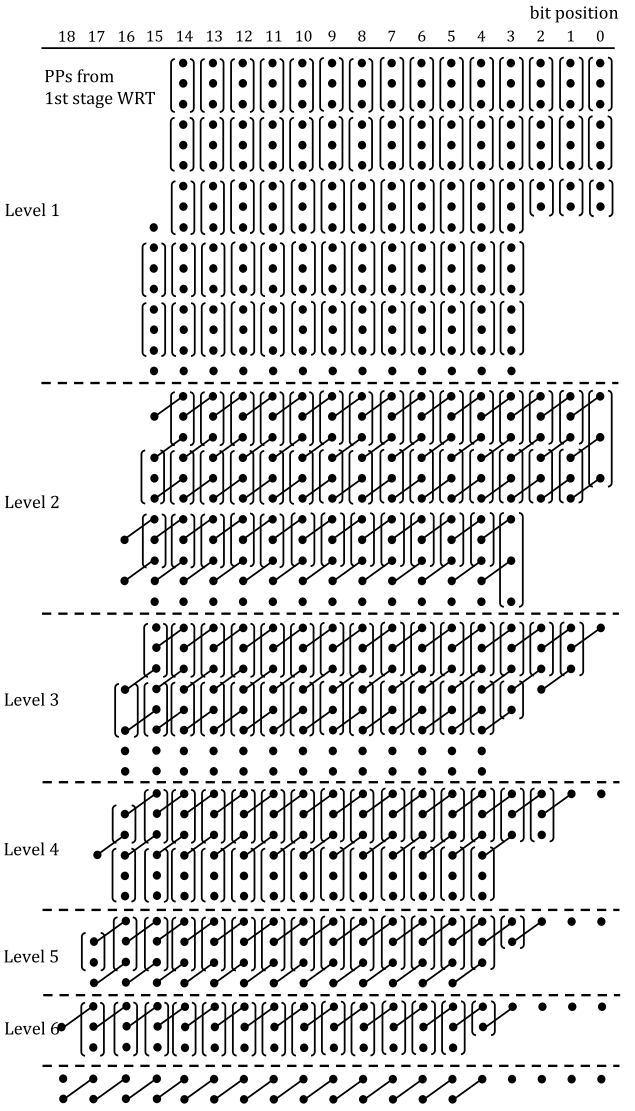


FIGURE 5. **Dot diagram of the second stage WRT with 16 partial products.**

the $m/2$ PPs generated by the MBE of each tap into two PPs. Therefore, the required number of levels depends on the bit-width m. If $m = 8$, two levels are needed to reduce 4 PPs to 2 PPs, whose dot diagram is shown in Fig. 4 (a). Note that the dots representing the initial 4 PPs are equivalent to the results in Fig. 3.

The subsequent WRT collects two PPs from each first stage WRT, then reduces them to two PPs again over several levels to implement the adder tree. Specifically, if $K = 8$, the number of PPs becomes 16, and the reduction procedure can be completed in the second stage WRT with 6 levels, whose dot diagram is shown in Fig. 5. However, the number of PPs handled by one WRT can be limited to 16 or less. As a design example, consider an FIR filter with 16 taps. Then, the first stage WRT generates a total of 32 PPs, i.e., 2 PPs per tap. The second stage WRT is composed of 2 WRTs with 16 PPs or 4 WRTs with 8 PPs. If 2 WRTs with 16 PPs each are selected, 4 PPs are generated by the second stage WRT. The next third stage WRT receives 4 PPs from the second stage WRT and processes them as shown in Figs. 6. Finally, 2 PPs are passed to the RCA.

The ripple carry adder (RCA) is used to add the final two PPs, which are the outputs of the last stage WRT to create the output of the FIR filter. The RCA is the simplest adder that performs m-bit addition by m full adders (FA) connected in series. Alternative fast adders such as a carry look ahead adder (CLA) or other variants may be selected, but RCA is used in this paper in order to have a regular structure with WRT which consists of only bit adders such as FAs.

III. DESIGN OF PROPOSED FINE-GRAINED PIPELINE FIR FILTER

A. SIMPLIFICATION FOR ANALYSIS

Our goal in this Section is to derive a very high speed FPFF through fine-grained seamless pipelining. The throughput of the FIR filter can be improved by inserting pipeline registers

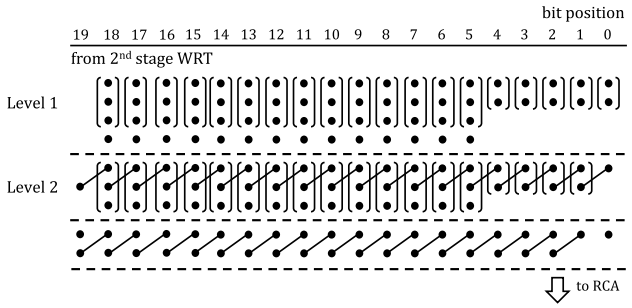


FIGURE 6. Dot diagram of the third stage WRT with 4 partial products.

TABLE 3. Simplification of propagation delays of logic gates to a linear scale of unit gate delay.

Delay of Logic Gate	Delay in nD_G
T_{INV}	0 (ignored)
T_{AND}	D_G
T_{OR}	D_G
T_{XOR}	$2D_G$
$T_{2:1\ \text{MUX}}$	$2D_G$

into the critical path at the expense of increasing latency and area. Therefore, the design of the pipeline FIR filter should begin with finding the critical path that causes the longest propagation delay. There are a lot of paths from each input bit to each output bit, of which the critical path with the longest propagation delay should be found through analysis. However, it is not possible to predict which gates in the standard cell library will be selected during the synthesis process. Also, the delay for each gate is not fixed either as it can change depending on constraints imposed on the synthesis process as well as temperature, load delay factor, output load capacitance, etc. In order to facilitate the analysis procedure and to help in establishing subsequent pipelining strategies, we simplify the analysis as follows:

- The logic circuit to be synthesized is reasonably predicted based on the logic expression to obtain the variable.
- The propagation delay is estimated by simplifying the logic circuit with only AND, OR, XOR, and 2:1 MUX.
- The delay of logic inversion is ignored.
- The gates with multiple inputs are available, but their propagation delays are the same regardless of the number of inputs.
- The propagation delay of each gate does not change depending on the external environment, temperature, load capacitance, etc.
- The propagation delay of each gate is simplified to a linear scale of the unit gate delay, i.e., nD_G. See Table 3.

B. CRITICAL PATH ANALYSIS

Let us begin with the propagation delay analysis of MBE based on the expressions that derive the PP bits summarized in Table 1. Since it is assumed that the delay of inversion is ignored in the analysis, all the bars above the variables in Table 1 can be removed for ease of estimation. Also, remember that it is assumed that the delay of a specific gate does not change according to the number of inputs. The propagation delay to obtain each PP bit is summarized in Table 4. Some variables are also listed in the fourth column as they have different delays in the least significant bit position $i = 0$. As a result, the estimated delay for each PP bit is shown in Fig. 4(b). It should be known that the critical path of MBE is in the calculation of α_0 whose delay is $6D_G$ regardless of the input bit-width of the MBE.

TABLE 4. Propagation delays to obtain partial product bits using MBE.

Variable	Delay	nD_G	nD_G $(i=0)$
T_{neg_i}	$T_{\text{OR}} + T_{\text{AND}}$	$2D_G$	0
$T_{\overline{one_i}}$	T_{XOR}	$2D_G$	0
$T_{\overline{two_i}}$	$T_{\text{AND}} + T_{\text{OR}}$	$2D_G$	$1D_G$
$T_{na_{ij}}$	T_{XOR}	$2D_G$	·
$T_{p_{ij}}$	$T_{na_{ij}} + T_{\text{OR}} + T_{\text{AND}}$	$4D_G$	·
T_{τ_i}	$T_{\overline{one_i}} + T_{\text{OR}}$	$3D_G$	$1D_G$
T_{c_i}	$T_{\overline{one_i}} + T_{\text{OR}} + T_{\text{AND}}$	$4D_G$	$2D_G$
$T_{\overline{\epsilon}}$	$T_{\text{AND}} + T_{\text{MUX}}$	$3D_G$	·
T_{t_1}	$T_{\overline{\epsilon}} + T_{\text{OR}} + T_{\text{AND}}$	$5D_G$	·
$T_{\overline{d}}$	$2T_{\text{AND}} + T_{\text{OR}}$	$3D_G$	·
T_{s_i}	$T_{p_{ij}}$	$4D_G$	·
T_{α_2}	$T_{s_i} + T_{\text{AND}}$	$5D_G$	·
T_{α_1}	$T_{s_i} + T_{\text{AND}}$	$5D_G$	·
T_{α_0}	$T_{s_i} + T_{\text{XOR}}$	$6D_G$	·

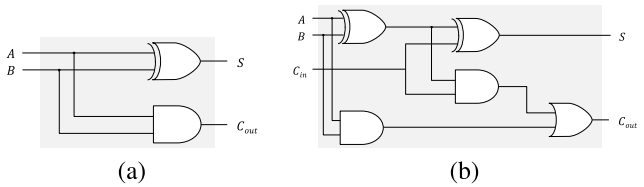


FIGURE 7. (a) Half adder. (b) Full adder.

There are only two components in WRT: a half adder (HA) and a full adder (FA). Assume that HA and FA can be implemented with logic circuits as shown in Fig. 7 and either of their two inputs to the output has the same delay. Then, their propagation delays for all the paths from inputs to outputs can be approximated as listed in Table 5. It is worth knowing that one operation is independent of others within the same level of WRT. Therefore, the maximum propagation delay of one level of WRT cannot exceed $4D_G$, and thus the delay of the entire WRT is equal to $4nD_G$, where n is the number of levels in WRT.

The FIR filter output $y(n)$ can be obtained by adding the two PPs which result from WRT using RCA. Let us examine the propagation delay of the RCA with the dot diagram in Fig. 8 which is for the addition of the last two PPs in Fig. 6.

TABLE 5. Propagation delays for a half adder and a full adder.

Adder	Path	Notation	Delay	nD_G
HA	A to S	T_{HAS}	T_{XOR}	$2D_G$
	A to C_{OUT}	T_{HAC}	T_{AND}	D_G
FA	A to S	T_{FAS}	$2T_{XOR}$	$4D_G$
	A to C_{OUT}	T_{FAC}	$T_{XOR}+T_{AND}+T_{OR}$	$4D_G$
	C_{IN} to S	T_{FCS}	T_{XOR}	$2D_G$
	C_{IN} to C_{OUT}	T_{FCC}	$T_{AND}+T_{OR}$	$2D_G$

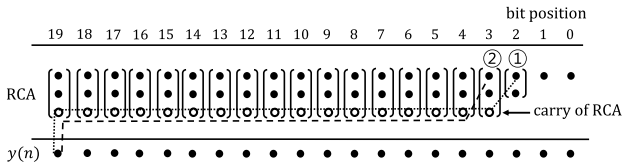


FIGURE 8. Dot diagram of RCA with 2 partial products.

The empty circles mean the C_{OUT} bits propagated from the lower bit position. The RCA consists of one HA in bit position 2 and 17 FAs in bit positions from 3 to 19. Assuming that all the bits of both input PPs arrive at the same time, Paths-① and ② can be the candidates for the critical path. Using approximations from Table 5, we can see that Path-② becomes a critical path by

$$c\text{Path-①}: \quad T_{RCA} = T_{HAC} + 16T_{FCC} + T_{FCS} = 35D_G, \quad (2)$$

$$\text{Path-②}: \quad T_{RCA} = T_{FAC} + 15T_{FCC} + T_{FCS} = 36D_G. \quad (3)$$

Note that C_{OUT} arrives from the HA of bit position 2 before the operation of $A \oplus B$ in the FA of the bit position 3 is completed. Therefore, regardless of whether there is an HA or an FA at the least significant bit position, the propagation delay of an RCA is dependent on only the number of FAs, i.e., $T_{n\text{RCA}}$ which denotes the delay of the RCA having n FAs, can be represented as follows:

$$T_{n\text{RCA}} = T_{FAC} + (n-2)T_{FCC} + T_{FCS} = (2n+2)D_G, \quad (4)$$

C. FINE-GRAINED SEAMLESS PIPELINING

The straightforward way to apply the pipelining to the reference FPFF is to insert the registers after MBE as well as WRTs on the dotted lines in Fig. 2. In order to obtain higher throughput, pipelining registers can be inserted at the gate level through fine-grained seamless pipelining. This section describes how to design a pipeline FPFF at the gate level, how to determine the number of pipeline stages to minimize cost, and how to locate the pipeline registers for a given timing constraint. All the design procedures are based on precise critical path analysis.

The first possible location of the pipeline registers is indicated by a dotted line before level 1 in Fig. 4, which stores the results of the hidden MBE calculation. Then, the delay of the first pipeline stage becomes $6D_G$, that is, to obtain α_0 in the first PP. Next, for pipelining of the WRT, the registers

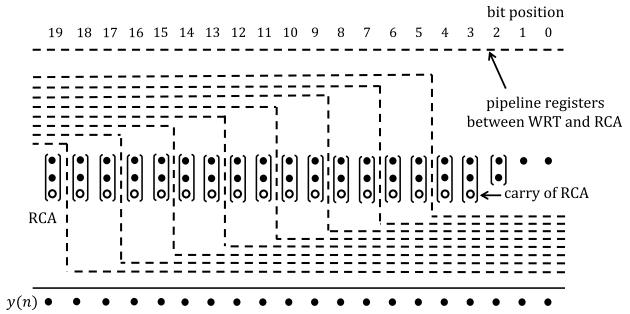


FIGURE 9. Dot diagram for pipelining inside RCA.

TABLE 6. Approximate propagation delay of pipeline stage.

Type	Type of Stage	Delay in nD_G
Type-1	MBE	$6D_G$
Type-2	n_1WRT	$4n_1D_G$
Type-3	n_2RCA	$(2n_2+2)D_G$
Type-4	MBE+n_1WRT	$(4n_1+6)D_G$
Type-5	n_1WRT+n_2RCA	$(4n_1+2n_2+2)D_G$
Type-6	MBE+n_1WRT+n_2RCA	$(4n_1+2n_2+6)D_G$

nWRT: WRT with n levels, nRCA: RCA with n FAs.

can be inserted before and after the WRTs, as well as between levels inside WRTs. The dotted lines in Figs. 4, 5 and 6 show their possible locations. If one pipeline stage contains n levels of WRT, the delay for that pipeline stage, denoted as $T_{n\text{WRT}}$, becomes $4nD_G$, where one $4D_G$ corresponds to T_{FAS} or T_{FAC}.

The pipeline registers can also be inserted between the FAs inside the RCA. The possible locations of the pipeline registers are shown as the dotted line in Fig. 9, where the top dotted line is between the last level of WRT and RCA. If a pipeline stage contains n FAs where $n \geq 2$, the delay denoted as $T_{n\text{RCA}}$ can be calculated as (4). Note that inserting a pipeline register after every individual FA results in a $4D_G$ delay, and it does not help to decrease the delay of whole FIR filter since MBE's delay is $6D_G$. If the pipeline registers are placed for each 2 FA as shown in Fig. 9, the delay of the RCA becomes $6D_G$ except for the last FA with $4D_G$, and the delay of RCA is balanced with that of MBE.

Meanwhile, some pipeline stages may involve a mixture of several levels of WRT, MBE, and FAs in RCA. All the types of pipeline stages and their propagation delays are summarized in Table 6. Types-1, 2, and 3 refer to pipeline stages with only MBE, WRT, and RCA, where n_1 and n_2 represent the number of levels in WRT and number of FAs in RCA, respectively. The delay for Types-4 is also straightforward because it is the sum of the delays of Types-1 and 2.

For Type-5, where one pipeline stage contains n_1 levels of WRT as well as n_2 FAs in RCA, it is worth a closer look. Let us locate the critical path of Type-5 first in order to formulate its propagation delay. Note that the critical path can be formed in one of two paths in WRT area, that is, one is from A to S of the FA in the same bit position, that is, in the vertical down

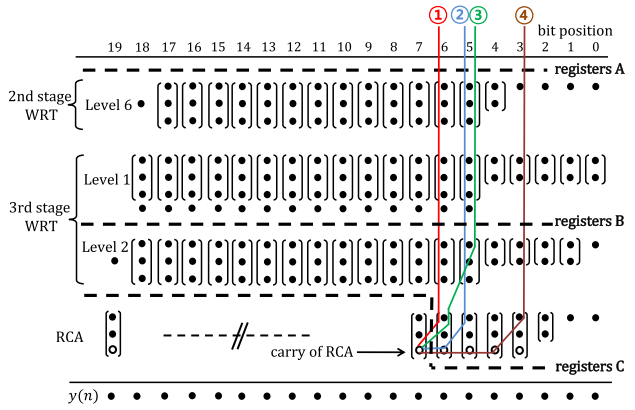


FIGURE 10. Propagation delay in pipeline stage containing WRT and RCA.

direction, and the other is from A to C_{OUT} in the diagonal down direction. On the other hand, in RCA area, we know that the critical path goes from a lower bit position to higher bit positions (from right to left) via carry propagation. Now, let us see how the critical path is formed in Type-5 with the example as shown in Fig. 10. In the figure, the last level-6 in Fig. 5 and the third stage WRT in Fig. 6 are combined with RCA in order to constitute Type-5. First let us consider only the area higher than the fifth bit position having the regular dot diagram, where both WRT and RCA are filled with only FAs. In this area, the following two observations prove that the path down vertically from the rightmost bit position becomes the critical path:

1) Among the adjacent bit positions full of FAs, the lower bit position forms the critical path when comparing the delay between Path-① and Path-② as shown in the second and third rows in Table 7.
2) In WRT, the critical path is formed vertically downward rather than diagonally downward when comparing the delay between Path-② and Path ③ as shown in third and fourth rows in Table 7.

Thus, in the regular area which is full of FAs, Path-③ vertically descending from the rightmost 5th bit position and moving left through the carry bits of the RCA until it encounters the pipeline register becomes the critical path.

However, if the bit positions where HA and FA are irregularly mixed in the WRT area, the 4th or lower bit positions can be included in the possible candidates for the critical path so that decision is not straightforward. That is, as shown in Table 7, in the case of the pipeline stage starting from register A and ending at register C, Path-② becomes the critical path, but when starting from register B and ending at register C, Path-④ becomes the critical path. It is inefficient to determine the critical path after exhaustively searching all the possible paths in the irregular bit positions. Instead, if the critical path is determined based on the bit position where the FA appears first in the RCA as in Path-④, and registers are placed for each of the two FAs after the bit position as shown in Fig. 9, the delay will not increase even if the critical path is not actually on this position. In conclusion, the propagation delay

TABLE 7. The delays to reach the input C_{IN} of the FA at 7th bit position in Fig. 10.

Design	Path	Delay	nD_G
Delay From A to C	Path-①	$3T_{FAS} + T_{FAC}$	$16D_G$
	Path-②	$3T_{FAS} + T_{FAC} + T_{FCC}$	$18D_G$
	Path-③	$2T_{FAS} + 2T_{FAC}$	$16D_G$
	Path-④	$2T_{HAS} + T_{FAC} + 3T_{FCC}$	$14D_G$
Delay From B to C	Path-①	$T_{FAS} + T_{FAC}$	$8D_G$
	Path-②	$T_{FAS} + T_{FAC} + T_{FCC}$	$10D_G$
	Path-③	$2T_{FAC}$	$8D_G$
	Path-④	$T_{HAS} + T_{FAC} + 3T_{FCC}$	$12D_G$

The delay of flip-flop is omitted from the table.

of Type-5 can be obtained by $(4n_1 + 2n_2 + 2)D_G$ where n_1 and n_2 are the number of WRT levels and the number of only FAs in RCA, respectively.

Now, let us consider the last Type-6, which contains all of MBE, WRTs, and RCA. It is equivalent to the entire structure of a non-pipeline FPFF. Note that the critical path of this type descends vertically from the bit position, which is the rightmost of the FA-filled bits, and travels to the left through the carry bit of the RCA. Then, the propagation delay of MBE becomes $4D_G$ instead of $6D_G$ because the critical path is not on the bit position with α_0. Also, the delays of WRTs and RCA are $4n_1D_G$ and $(2n_2+2)D_G$, where n_1 and n_2 are the sum of all WRT levels and number of FAs in RCA, respectively. Therefore, the formula in the last row of Table 6 can be obtained by summing the delays of all the components.

D. DESIGN EXAMPLE

In the last Section, the propagation delay of the pipeline FPFF is expressed in terms of nD_G. An approximation of D_G can be obtained from a CMOS library for synthesis, which can speed up pipelining decisions in the design. Otherwise, the design procedure can begin with a non-pipeline reference FPFF, and increase the number of pipeline stages until the timing constraint is met. As a design example, let us consider a pipelining of an 8-bit, 16-tap reference FPFF which consists of the MBE in Fig. 3, hierarchical 3 stages WRTs in Figs. 4(a), 5, 6, and RCA in Fig. 8. The total number of levels of WRTs becomes 10 where the 1st, 2nd, and 3rd WRTs have 2, 6, and 2 stages, respectively. The RCA has 17 FAs from bit positions 3 to 19. Table 8 lists six design examples with the different number of pipeline stages. The design in the second row represents the non-pipeline FPFF, and its propagation delay can be obtained by substituting 10 and 17 for n_1 and n_2 of Type-6, respectively, then resulting in $80D_G$. The design with 2 pipeline stages in the third row can be obtained by inserting the registers after the ninth level of WRT ($n_1 = 9$), that is, after level-1 of the third stage WRT. Then, the first pipeline stage has Type-4 with $n_1 = 9$. Its propagation delay becomes $42D_G$. where $6D_G$ is for MBE and $36D_G$ for WRTs. The second pipeline stage has $40D_G$ with $4D_G$ for remaining 1 level for WRT and $36D_G$

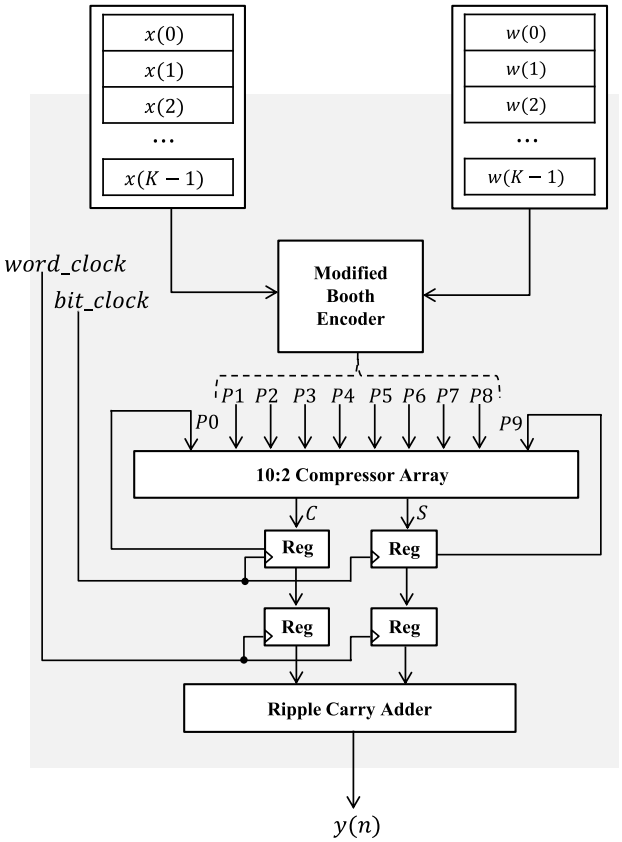


FIGURE 11. Structure of a single MAC FIR filter (SMFF) with $m = 16$.

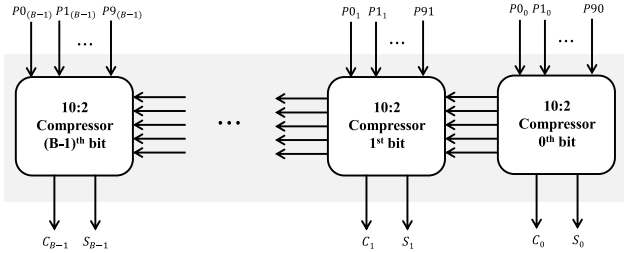


FIGURE 12. A 10:2 compressor array consisting of m 10:2 compressors.

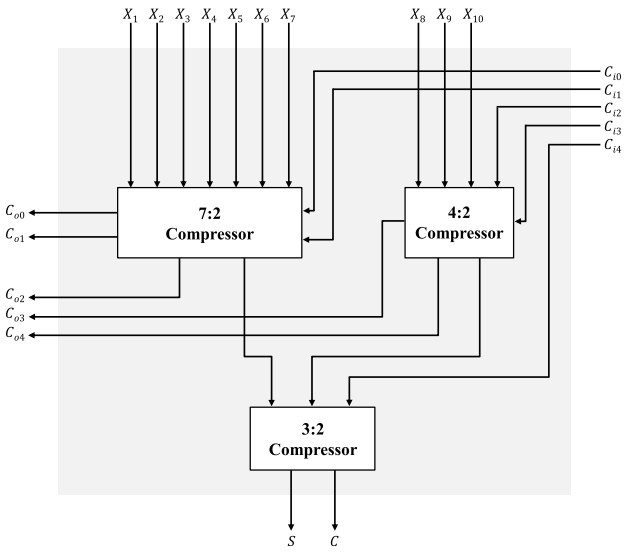


FIGURE 13. Hierarchical structure for 10:2 compressor.

for 17 FAs in RCA, i.e., Type-5 where n_1 and n_2 are set to 1 and 17, respectively. It is known that the first pipeline stage with a longer propagation delay contains the critical path. If the timing constraint is not met for 2-stage pipeline design, additional pipeline stages can be added based on the analysis. The locations of the pipeline registers and corresponding parameters of n_1 and n_2 are summarized in Table 8. Note that the maximally pipelined FPFF can be obtained by inserting pipeline registers after MBE, every WRT level, and every 2 FAs in RCA having total 20 pipeline stages, and the maximum throughput becomes $1/6D_G$.

IV. DESIGN ALTERNATIVES

A. SINGLE-MAC FIR FILTER

Based on the reference full-parallel architecture, let us consider a design alternative to find different trade-offs between area and throughput. A single-MAC FIR filter (SMFF) uses components of the reference FPFF, but involves only one MAC unit. The SMFF performs K-tap FIR filtering through K recursive MAC computations over K clock cycles. Therefore, the area of the SMFF can be significantly reduced at the expense of throughput. The detail structure of the proposed SMFF with K 16-bit inputs and K 16-bit coefficients is shown in Fig. 11. The MBE receives one input and one coefficient every clock cycle and then produces $m/2$ PPs. The compressor array reduces $m/2 + 2$ PPs, i.e., the $m/2$ PPs from the MBE and the other 2 PPs from the registers, to 2 PPs. The generated 2 PPs are stored in the registers and taken by the compressor array again in the next cycle.

Although the first stage WRT shown in Fig. 4(a) can be reused as a compressor array, one of the various compressors found in the literature can also be employed for the purpose. The compressor proposed recently in [14] can be one of the best candidates as it offers a shorter critical path by limiting carry propagation compared to the existing compressors. In addition, it provides compressors of various sizes in a hierarchical structure, so fits well with the proposed FIR filter. If m is 16 bits, the compressor array should serve to reduce 10 PPs to 2 PPs, which can be implemented by connecting 10:2 compressors in series as shown in Fig. 12. Note that the number of compressors is equal to the number of bits of the PPs. Fig. 13 shows the hierarchical structure of a 10:2 compressor which consists of 3:2, 4:2, and 7:2 compressors, whose circuit diagrams are shown in Fig. 14(a), (b), and (c), respectively.

It is known from Fig. 13 that the propagation delay of the 10:2 compressor is the sum of the delays of 7:2 and 3:2 compressors. Also, as can be seen in Figs. 14(c) and (a), the propagation delay of 7:2 compressor is 6 XOR gate delays and the one of 3:2 compressor is 2 XOR gate delays. As a result, the delay of the 10:2 compressor becomes 8 XOR gate delays. Since the carry propagation is avoided in the 10:2

TABLE 8. Design examples of pipeline FPFF with 8 bits and 16 taps.

Num. of Stages	Delay	Stage-1	Stage-2	Stage-3	Stage-4	Stage-5	Stage-6	Stage-7
1	$80D_G$	Type-6 (n_1=10,n_2=17)	-	-	-	-	-	-
2	$42D_G$	Type-4 (n_1=9)	Type-5 (n_1=1,n_2=17)	-	-	-	-	-
3	$30D_G$	Type-4 (n_1=6)	Type-5 (n_1=4,n_2=4)	Type-3 (n_2=13)	-	-	-	-
4	$22D_G$	Type-4 (n_1=4)	Type-2 (n_1=5)	Type-5 (n_1=1,n_2=7)	Type-3 (n_2=10)	-	-	-
7	$14D_G$	Type-4 (n_1=2)	Type-2 (n_1=3)	Type-2 (n_1=3)	Type-5 (n_1=2,n_2=2)	Type-3 (n_2=5)	Type-3 (n_2=5)	Type-3 (n_2=5)
20	$6D_G$	Maximally Pipelined, 1 Type-1 Stage, 10 Type-2 (n_1=1) Stages, 8 Type-3 (n_2=2) Stages, 1 Type-3 (n_2=1) Stage						

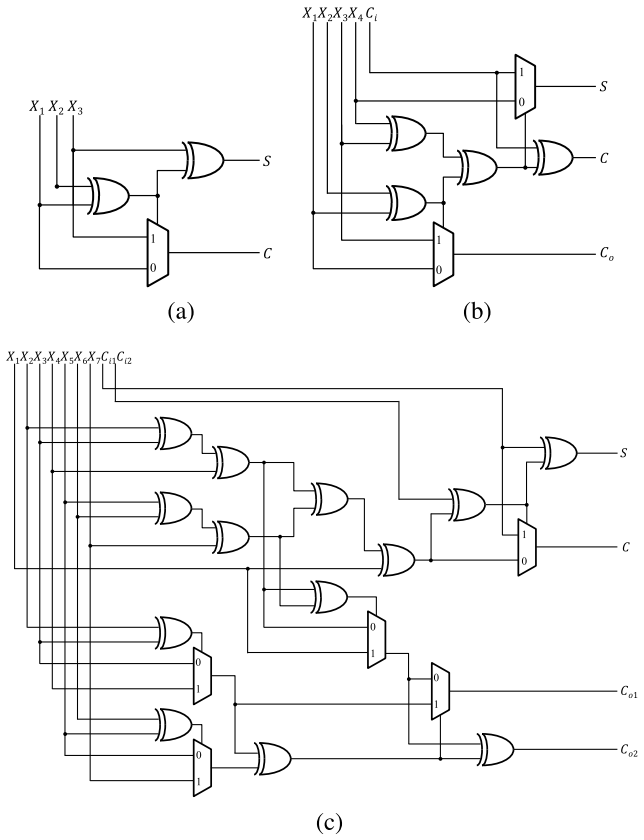


FIGURE 14. (a) 3:2 compressor. (b) 4:2 compressor. (c) 7:2 compressor.

compressor array, the total delay of the 10:2 compressor array is equivalent to the one of each 10:2 compressor. Note that the 10:2 compressor array has a shorter propagation delay than a corresponding 5 level WRT with 10 PPs as input with 10 XOR gate delays. However, for the proposed full-parallel architecture, WRT is more advantageous to have a regular structure to apply the proposed pipelining technique. On the other hand, in SMFF, the compressor array is recommended since reducing the delay of the compressor is key to increase the throughput.

The multiply-and-accumulation process is repeated K times, and the final two PPs are added using the RCA. It is important to know that the last addition only needs to be

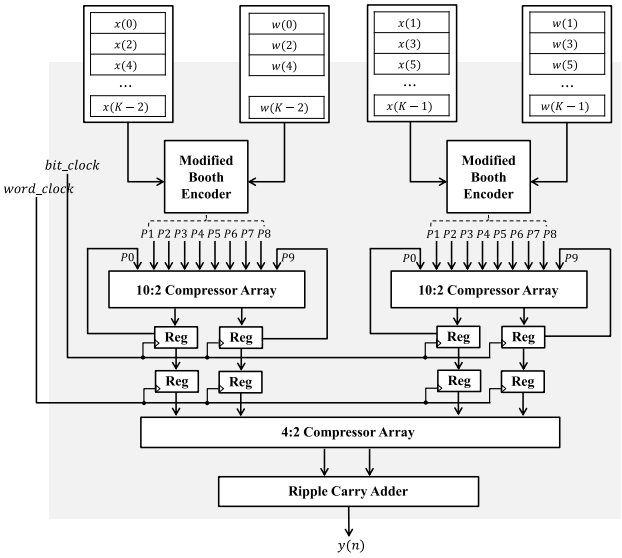


FIGURE 15. Structure of a folded FIR filter (FDFF) with $m = 16$.

performed once to finally obtain the output $y(n)$ without having to perform the operation for each tap. Therefore, in Fig. 11, the upper two registers for accumulation and the lower two registers for storing the final two PPs are synchronized to different clocks, i.e., bit-clock and word-clock, respectively. The word-clock period is set to be K times longer than that of the bit-clock, that is, $T_{\text{word-clock}} = KT_{\text{bit-clock}}$. Although the RCA has a longer delay than the MBE or compressor array in the proposed SMFF, it is not on the critical path since it is synchronized to the word-clock, and therefore, does not affect the throughput of the FIR filter. The SMFF can produce one output sample every K bit-clock cycles where $T_{\text{bit-clock}}$ is set to have a longer period than $T_{\text{FF}} + T_{\text{MBE}} + 6T_{\text{XOR}}$. It should be noted that the proposed SMFF can provide much higher throughput than the conventional single-MAC-based FIR filter [19], where the clock period is set to the sum of the delays of the discrete components of multiplier and adder.

B. FOLDED FIR FILTER

A folded FIR filter (FDFF) uses two or more MBEs and compressor arrays, but less than the number of taps of the FIR filter. The FDFF is slower than FPFF but occupies less area,

TABLE 9. Comparison of synthesis results of full-parallel FIR filters when $m = 8$ and $K = 16$.

Design	Delay	EDAT	DAT	TPT	Area	ADP	PC	EPS
Design of [25]	–	–	5.20	192	18257	94936	3.24	32.07
Design of [26]	–	–	1.79	588	71195	127439	6.08	59.39
Design of [29]	–	–	1.58	632	25163	39757	1.30	12.46
Design of [32]	–	–	1.13	885	72648	82092	1.39	12.77
Design of [33]	–	–	15.28	65	8235	125831	0.75	11.47
Non-Pipeline FPFF-1	$80D_G$	4.55	4.55	220	25294	115089	1.25	11.93
Pipeline FPFF-2	$42D_G$	2.50	2.52	397	25955	65406	1.36	13.02
Pipeline FPFF-3	$30D_G$	1.85	1.89	529	27191	51391	1.56	15.04
Pipeline FPFF-4	$22D_G$	1.42	1.50	667	29127	43691	1.87	18.10
Pipeline FPFF-5	$14D_G$	0.99	0.99	1010	34918	34569	2.76	26.85
Pipeline FPFF-6	$6D_G$	0.55	0.54	1852	62634	33822	6.96	68.29

DAT: data arrival time (ns), EDAT: estimated DAT (ns), TPT: throughput (MSPS, Mega samples per second), Area (μm^2), ADP: area-delay product (μm$^2\times$ns), PC: power consumption at 100 MHz operating clock (mW), EPS: energy per sample (mW$\times$ns).

and in contrast, has larger area than the SMFF but runs faster. That is, an optimal trade-off between SMFF and FPFF can be found by adjusting the number of MAC units according to the required timing constraint. The structure of the proposed FDFF, especially with two MAC units is shown in Fig. 15. Two pairs of inputs and coefficients are fed into MBE every $T_{\text{bit-clock}}$. If the input and coefficients are m-bits each, then each MBE produces $m/2$ PPs, which are used as inputs to the $m/2 + 2$:2 compressor array. If the bit-widths of input and coefficient are 16-bits, two 10:2 compressor arrays, each of which is the same as the one in the SMFF, and one 4:2 compressor array are employed. The architecture of the 4:2 compressor is shown in Fig. 14 (b). Each 10:2 compressor array produces two PPs, i.e., a total of 4 PPs, which are stored in the upper 4 registers, then fed back to the compressor arrays with the next MBE output on the next bit-clock cycle. If the number of taps of the FIR filter is K where K is even, this process is repeated $K/2$ times to have one output sample. After $K/2$ repetitions, the 4 PPs accumulated in the registers are reduced to 2 PPs using a 4:2 compressor array, and finally the output of the filter is obtained using the RCA. Since the operations of the 4:2 compressor array and RCA only need to be performed once per $K/2$ bit-clock cycles, $T_{\text{word-clock}}$ can be set to have a $K/2$ times longer period than the $T_{\text{bit-clock}}$. Also, note that $T_{\text{bit-clock}}$ can be set to equal to the one of the SMFF, that is, $T_{\text{FF}} + T_{\text{MBE}} + 6T_{\text{XOR}}$ because their critical paths are the same.

V. SIMULATION RESULTS

We have coded all the proposed pipeline and non-pipeline FPFFs presented as design examples in Table 8 in VHSIC hardware description language (VHDL). The number of taps is set to 16, i.e., $K = 16$ and the bit-width of input and coefficients are chosen to be 8. Since the internal data are not truncated so that no quantization error occurs, the output of the FIR filter is 20 bits. For the comparison of the proposed high-throughput FPFFs with the existing full-parallel FIR filters [25], [26], [28], [31], [32], we have synthesized them using the Synopsys Design Compiler with TSMC 90-nm standard CMOS library [33].

Table 9 shows the synthesis results in terms of data arrival time (DAT), the throughput (TPT), area, area-delay product (ADP), power consumption (PC), and energy per sample (EPS). Delays expressed as the number of unit gate delays, nD_G in Table 8 are listed again in the second column of Table 9. The third column shows the estimated DAT, namely, EDAT which is the sum of the logic delay and flip-flop delay in the second column. The values of D_G and T_{FF} are set to 0.054ns and 0.23ns, respectively, which were obtained from simulation with the technical library. EDAT can be compared to DAT, which is the actual synthesis result listed in the fourth column. The value of EDAT matches well with the value of DAT. EDAT has proven to be a useful tool in determining the number of pipeline stages to satisfy a given timing constraint.

The synthesis results show that the DAT of the proposed non-pipeline architecture, namely non-pipline FPFF-1 is 4.55ns. TPT becomes the reciprocal of DAT because a full-parallel design can produce one output sample per clock cycle. Therefore, the TPT for a non-pipeline FPFF-1 is 220 Mega samples per second (MSPS). The proposed two-stages pipeline FPFF denoted as pipeline FPFF-2 is estimated to have about half the DAT compared to the non-pipeline design, and the synthesis result proves it. As expected, DAT continues to decrease as the number of pipeline stages increases. Among the proposed FIR filters, the design with the highest throughput has 6 unit gate delays as shown in the last row of Table 9, and its actual synthesis result shows that this value is 0.54ns. The area of the proposed filter increases as the number of pipeline stages increases, but the increase is mostly the register area. FPFF-6 has more than twice the area of FPFF-1, but FPFF-1 is entirely made up of logic, indicating that more than half of the area of FPFF-6 is occupied by registers.

ADP is an indicator of the area and time efficiency for which the FIR filter is designed, taking into account the trade-off between throughput and area. It can be seen from Table 9

TABLE 10. Comparison of the synthesis results of SMFF and FDFF when $m = 16$ and $K = 16$.

Design		NMAC	DAT	NOC	MSP	TPT	Area	ADP	PC	EPS
Single MAC	Design of [19]	1	3.38	1/16	62.08	16.10	14887	924184	1.50	240.44
	Design of [31]	1	1.61	1/16	25.76	38.82	30850	794696	1.05	161.64
	SMFF	1	1.27	1/16	20.32	49.21	11017	223870	0.55	84.98
Multiple MAC	Design of [19]	8	4.92	1/2	9.84	101.62	45995	452590	3.92	78.48
	FDFF	2	1.27	1/8	10.16	98.43	21789	221376	1.23	94.60

NMAC: number of MAC units, DAT: data arrival time (ns), NOC: number of output samples per cycle, MSP: minimum sampling period (ns), TPT: throughput (MSPS, Mega samples per second), Area (μm^2), ADP: area-delay product ($\mu m^2 \times$ns), PC: power consumption at 100 MHz operating clock (mW), EPS: energy per sample (mW$\times$ns).

that the ADP value continues to decrease as the number of pipeline stages increases. Meanwhile, the metric for measuring energy efficiency is the EPS shown in the last column of Table 9, which means the energy required to acquire one output sample. As expected, the EPS increases as the number of pipeline stages of the FIR filter increases.

The proposed pipeline FPFF-6 has a larger area than the designs in [25], [28], and [32], but offers the highest throughput and lowest ADP compared to the existing FIR filters shown in Table 9. Specifically, the throughput of the proposed FPFF-6 provides 9.65, 3.15, 2.93, 2.09, and 28.49 times higher throughput and 64.4%, 73.5%, 14.9%, 58.8%, and 73.1% less ADP compared with the ones proposed in [25], [26], [28], [31], and [32], respectively.

The proposed SMFF and FDFF have also been synthesized with $m = 16$ and $K = 16$, and compared with the corresponding designs in [19] and [31] in Table 10. According to the synthesis results, both proposed designs have the same DAT, and the value is 1.27ns. Since the number of output samples per clock cycle (NOC) for both designs are 1/16 and 1/8, the throughput (TPT) will be NOC divided by DAT, resulting in 49.21 MSPS and 98.43 MSPS, respectively. The single MAC designs of [19], [31] and the proposed SMFF have only one MAC, but the proposed SMFF has 3.06 and 1.27 times higher throughput, and 75.8% and 71.8% less ADP than the designs of [19] and [31], respectively. The proposed FDFF also has an ADP of 51.1% less than the folded design of [19], which efficiently trades off area and throughput.

VI. DISCUSSION

A. DISCUSSION OF RESULTS

The area complexity of the proposed filter is approximately proportional to the increase of the input bit-width, coefficient bit-width, or the number of tabs. However, more discussion is needed on the speed performance of the proposed structure. As listed in Table 9, the proposed FPFF-6 shows the maximum operating speed that the proposed full-parallel architecture can support. Specifically, the speed of 1.85 GSPS is a very high speed which is not found in the literature. For conventional FIR filter structures, pipeline registers are placed before and after individual components such as multipliers or adders, therefore, the TPT structurally decreases as the bit-width increases. Also, as the number of taps of the FIR filter increases, the TPT decreases. In contrast, the proposed FPFF is designed to provide $6D_G$ DAT regardless of the bit-width or number of taps when maximally pipelined. More specifically, when pipelining is not applied, the propagation delay of RCA and the first stage WRT is proportional to the bit-width. Meanwhile, the propagation delay of the WRT after the second stage increases in proportion to the number of taps of the filter. However, we have taken $6D_G$ as the delay in the critical path by reducing all delays except MBE to $4D_G$ through fine-grained seamless pipelining. The proposed design offers a very low DAT of 0.54ns, resulting in the throughput over 1.85 Giga samples per second (GSPS). It proves that the throughput of the proposed full-parallel design can be significantly improved only at the cost of the register area without additional combinational logic. This is why the proposed design has a significantly lower ADP than the existing design. On the other hand, as the number of registers increases, the power consumed by the registers also increases, so the energy efficiency decreases. But, energy efficiency should not be reduced too much unless excessive pipelines are used so that the sequential logic becomes dominant.

Throughput for SMFF and FDFF is calculated in terms of NOC/DAT. In the design process, we focused on maximizing TPT by reducing DAT and hypothesized that DAT should be $T_{FF} + T_{MBE} + 6T_{XOR}$. By identifying the critical path in the synthesis report and looking at the actual synthesis results in Table 10, we could see that this value is close to a few gate delays of 1.27ns, which is one of the design goals of this work.

B. LIMITATIONS OF THE PROPOSED ARCHITECTURE

FIR filters are generally used as part of a system rather than a complete system by itself. It is demonstrated in this paper that the proposed FIR filter is capable of high-speed processing, but to be deployed in actual system, we need to look at the speed requirement of the whole system. In certain applications, where very high-speed is required, the speed of all subsystems including the FIR filter unit need to be enhanced according to the system level specification. However, most of the signal processing algorithms consist of basic operations such as addition/subtraction, multiplication, addition chain, and MAC, and each basic operation has a method capable of high-speed processing [8]. Therefore, the proposed FIR filter

can be combined with other high-speed implementations to contribute to the speed improvement of the entire system.

On the other hand, the Wallace tree design is not always straightforward. In particular, the design of the Wallace tree should be changed depending on the bit-width of the coefficients or the truncation of the intermediate result. This kind of limitation is observed in all such high-speed and low-area implementation of FIR filters, and not specific to the proposed design. Moreover, the Wallace tree does not need to be modified to change the coefficient values of the FIR filter. Also, the truncation process can be simplified by pruning the lower bits of the Wallace tree.

C. APPROXIMATION OF FIR FILTER COEFFICIENTS

As in this paper, FIR filter designs are targeting the integrated circuit implementations such as the ASICs or FPGAs which use fixed-point arithmetic rather than floating-point arithmetic to reduce the implementation complexity and overall chip cost. For the proposed FPFF, we have used an 8-bit fixed-point representation for the filter coefficients, and a 16-bit representation for the coefficients of SMFF and FDFF. If the bit-width of the filter coefficients is increased, the signal-to-quantization noise ratio decreases, resulting in an increase in the stopband attenuation of the FIR filter. This is a trade-off to increase the resolution, which will increase the number of partial products and bit-width that are produced by MBE. Finally, this increases the complexity of the FIR filter by increasing the volume of the Wallace tree and compressor. However, even if the complexity of the proposed FIR filter increases, the DAT of the maximally pipelined FPFF remains at $6D_G$. That is, the high throughput, which is a main advantage of the proposed structure, can be maintained regardless of the increase in the resolution.

As an alternative, if the coefficients of the FIR filter are fixed, the filter coefficients can be approximated in the form of a sum of powers-of-two [34]. Then, MBE can be removed from the proposed architecture and instead, the input to the first stage WRT will be a number of shifted inputs equal to the number of powers-of-two terms. As a result, MBE is removed, and the volume of the Wallace tree is also reduced, simplifying the entire FIR filter realization. Use of approximate coefficients degrades the frequency response of the FIR filter, but can significantly reduce the complexity of the proposed architecture. Also, since MBE disappears, the minimum DAT can be reduced to $4D_G$ equal to the propagation delay of one FA. Based on the application requirement on the sensitivity of frequency response and the chip cost, one can consider the approximation strategy.

VII. APPLICATIONS OF THE PROPOSED FIR FILTERS

The proposed structure can be used in a variety of high-throughput DSP applications or in the design of digital communication modules based on FIR filters [8].

- It can be used directly as an interpolation filter in digital transmitters that require a sampling rate of 1 Gigahertz or higher [34]. Since the coefficients of the FIR filter for a given application is generally are constant, and also can be expressed in powers-of-two form to reduce computational complexity, the proposed FIR filter can be modified by the method described in Section VI-C.
- Least-mean-square (LMS) adaptive filters are widely used in communication applications, where several levels of pipelining are difficult to be applied, even if high throughput is required. Thus, the proposed non-pipeline FPFF structure can be used to obtain high throughput without increasing the adaptation delay [24].
- Matrix multiplications and discrete transforms basically consist of many dot product operations, but their high complexity makes it difficult to achieve high throughput. Very high throughput can be achieved by using the proposed pipeline FPFF in parallel [35], [36]. However, further study is still needed to find an efficient implementation in terms of area and power consumption, rather than replicating a few simple FIR filters and arranging them in parallel.
- Convolutional neural networks (CNNs) require very high throughput, and on the other hand involves many layers of convolutional operations. In order to design processing elements for CNNs, very high throughput can be achieved by designing the tiles using MACs based on MBE, compressor arrays, and registers used in the proposed SMFF and FDFF [5].
- The proposed MAC architecture used in SMFF and FDFF method could also be used to accelerate the CORDIC or quaternion-based high-dimensional vector rotation. The method of applying the proposed structure to CORDIC based activation function of artificial neural networks that require very high throughput may also be an area of future research. [37]
- High-throughput low-latency digital FIR filter is required for read channels of modern disk drives. Specifically, the equalization step of the read channel is performed by an FIR filter whose data rate should support up to 1 GHz frequency [38].

Among the examples presented, the proposed architecture can be applied without major changes in the case of a general FIR filter or the adaptive LMS filter. For other applications, the proposed FPFF or SMFF/FDFF can be selectively applied depending on whether it has a parallel structure or a MAC-based recursive structure.

VIII. CONCLUSION

In this paper, we have proposed pipeline FIR filters that can provide very high throughput of over 1.85 GSPS. The proposed design begins with a reference full-parallel design based on MBE, hierarchical Wallace tree network, and RCA. The propagation delay of the FIR filter has been approximated in terms of unit gate delay through precise analysis, which helped to establish an efficient pipelining strategy at gate level. As a result, the proposed full-parallel design can

achieve very high throughput through fine-grained seamless pipelining. In this paper, we have also proposed alternative structures, which provides relatively high throughput while significantly reducing the area. The significance of this paper is that the proposed FIR filter can provide scalability to DSP applications that require very high throughput rate, which is more than Giga samples per second.

ACKNOWLEDGMENT

The EDA Tool was supported by the IC Design Education Center.

REFERENCES

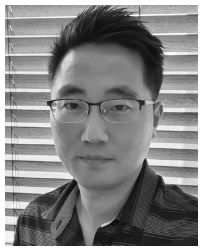
[1] A. Eghbali, H. Johansson, O. Gustafsson, and S. J. Savory, "Optimal least-squares FIR digital filters for compensation of chromatic dispersion in digital coherent optical receivers," *J. Lightw. Technol.*, vol. 32, no. 8, pp. 1449–1456, Apr. 2014.
[2] D. Kang, Y. Kang, and Y. Hong, "VLSI implementation of fractional motion estimation interpolation for high efficiency video coding," *Electron. Lett.*, vol. 51, no. 15, pp. 1163–1165, Jul. 2015.
[3] M. S. Hosseini and K. N. Plataniotis, "High-accuracy total variation with application to compressed video sensing," *IEEE Trans. Image Process.*, vol. 23, no. 9, pp. 3869–3884, Sep. 2014.
[4] J. Wang, J. Lin, and Z. Wang, "Efficient hardware architectures for deep convolutional neural network," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 6, pp. 1941–1953, Jun. 2018.
[5] A. Ardakani, C. Condo, M. Ahmadi, and W. J. Gross, "An architecture to accelerate convolution in deep neural networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 4, pp. 1349–1362, Apr. 2018.
[6] P. Meher and S. Park, "Design of cascaded CORDIC based on precise analysis of critical path," *Electronics*, vol. 8, no. 4, p. 382, Mar. 2019.
[7] P. K. Meher and M. Maheshwari, "A high-speed FIR adaptive filter architecture using a modified delayed LMS algorithm," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2011, pp. 121–124.
[8] P. K. Meher, "Seamless pipelining of DSP circuits," *Circuits, Syst., Signal Process.*, vol. 35, no. 4, pp. 1147–1162, Apr. 2016.
[9] S. K. Patel and S. K. Singhal, "Area–delay and energy efficient multi-operand binary tree adder," *IET Circuits, Devices Syst.*, vol. 14, no. 5, pp. 586–593, Aug. 2020.
[10] L. Dadda and V. Piuri, "Pipelined adders," *IEEE Trans. Comput.*, vol. 45, no. 3, pp. 348–356, Mar. 1996.
[11] S. A. Khan, *Digital Design of Signal Processing Systems: A Practical Approach*. Hoboken, NJ, USA: Wiley, 2011.
[12] S.-R. Kuang, J.-P. Wang, and C.-Y. Guo, "Modified booth multipliers with a regular partial product array," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 56, no. 5, pp. 404–408, May 2009.
[13] J.-Y. Kang and J.-L. Gaudiot, "A simple high-speed multiplier design," *IEEE Trans. Comput.*, vol. 55, no. 10, pp. 1253–1258, Oct. 2006.
[14] T. V. Fontanari, G. Paim, L. M. G. Rocha, P. Ucker, E. Costa, and S. Bampi, "An efficient N-bit 8-2 adder compressor with a constant internal carry propagation delay," in *Proc. IEEE 11th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2020.
[15] A. Fathi, B. Mashoufi, and S. Azizian, "Very fast, high-performance 5-2 and 7-2 compressors in CMOS process for rapid parallel accumulations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 6, pp. 1403–1412, Jun. 2020.
[16] T.-B. Juang, P. K. Meher, and K.-S. Jan, "High-performance logarithmic converters using novel two-region bit-level manipulation schemes," in *Proc. Int. Symp. VLSI Design, Autom. Test*, Apr. 2011, pp. 1–4.
[17] P. K. Meher, J. Valls, T.-B. Juang, K. Sridharan, and K. Maharatna, "50 years of CORDIC: Algorithms, architectures, and applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 9, pp. 1893–1907, Sep. 2009.
[18] P. K. Meher and S. Y. Park, "CORDIC designs for fixed angle of rotation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 2, pp. 217–228, Feb. 2013.
[19] P. K. Meher and S. Y. Park, "Reconfigurable FIR filter for dynamic variation of filter order and filter coefficients," *J. Semicond. Technol. Sci.*, vol. 16, no. 3, pp. 261–273, Jun. 2016.
[20] C. Cheng and K. K. Parhi, "Hardware efficient fast parallel FIR filter structures based on iterated short convolution," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 8, pp. 1492–1500, Aug. 2004.
[21] Y.-T. Hwang and C.-L. Su, "Parallel and pipelined architecture designs for distributed arithmetic-based recursive digital filters," in *Proc. 9th IEEE Workshop VLSI Signal Process.*, Oct. 1996, pp. 35–44.
[22] S. Y. Park and P. K. Meher, "Low-power, high-throughput, and low-area adaptive FIR filter based on distributed arithmetic," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 60, no. 6, pp. 346–350, Jun. 2013.
[23] A. Weinberger, "4-2 carry-save adder module," *IBM Tech. Disclosure Bull.*, vol. 23, no. 8, pp. 3811–3814, 1981.
[24] 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.
[25] P. K. Meher and S. Y. Park, "High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic," in *Proc. IEEE/IFIP 19th Int. Conf. VLSI Syst.-on-Chip*, Oct. 2011, pp. 428–433.
[26] P. K. Meher, "Hardware-efficient systolization of DA-based calculation of finite digital convolution," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 53, no. 8, pp. 707–711, Aug. 2006.
[27] P. K. Meher, S. Chandrasekaran, and A. Amira, "FPGA realization of FIR filters by efficient and flexible systolization using distributed arithmetic," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3009–3017, Jul. 2008.
[28] S. Y. Park and P. K. Meher, "Efficient FPGA and ASIC realizations of a DA-based reconfigurable FIR digital filter," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 7, pp. 511–515, Jul. 2014.
[29] H. Jiang, L. Liu, P. P. Jonker, D. G. Elliott, F. Lombardi, and J. Han, "A high-performance and energy-efficient FIR adaptive filter using approximate distributed arithmetic circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 1, pp. 313–326, Jan. 2019.
[30] L.-H. Chen, W.-L. Liu, and O. T.-C. Chen, "Determination of radix numbers of the booth algorithm for the optimized programmable FIR architecture," in *Proc. IEEE Int. Symp. Circuits Syst. Emerg. Technol. 21st Century.*, vol. 2, Aug. 2000, pp. 345–348.
[31] Synposys, Inc. (Jun. 2018). *DesignWare Building Block IP Documentation Overview*. DWBB_201806.0. [Online]. Available: http://www.synopsys.com/
[32] P. K. Meher and S. Y. Park, "A novel DA-based architecture for efficient computation of inner-product of variable vectors," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Jun. 2014, pp. 369–372.
[33] *TSMC 90nm General-Purpose CMOS Standard Cell Libraries-tcb-n90ghp*. [Online]. Available: https://www.tsmc.com/
[34] B. Parent, J. Muller, A. Kaiser, and A. Cathelin, "Design of 10 GHz sampling rate digital FIR filters with powers-of-two coefficients," in *Proc. 20th Eur. Conf. Circuit Theory Design (ECCTD)*, Aug. 2011, pp. 584–587.
[35] S. S. Nayak and P. K. Meher, "High throughput VLSI implementation of discrete orthogonal transforms using bit-level vector-matrix multiplier," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 46, no. 5, pp. 655–658, May 1999.
[36] G. S. Maharana and P. K. Meher, "Parallel algorithms and systolic architectures for 1-and 2-D interpolation using discrete Hartley transform," *Int. J. Comput. Appl.*, vol. 22, no. 1, pp. 1–7, Jan. 2000.
[37] G. Raut, S. Rai, S. K. Vishvakarma, and A. Kumar, "A CORDIC based configurable activation function for ANN applications," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2020, pp. 78–83.
[38] M. Singh, J. A. Tierno, A. Rylyakov, S. Rylov, and S. M. Nowick, "An adaptively pipelined mixed synchronous-asynchronous digital FIR filter chip operating at 1.3 gigahertz," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 7, pp. 1043–1056, Jul. 2010.

SU MIN CHO received the B.S. degree in electronic engineering from Myongji University, Yongin, Republic of Korea, in 2018, where she is currently pursuing the M.S. degree with the Department of Electronic Engineering. Her research interests include computer arithmetic, low-power architectures for digital signal processing, embedded system design, and system-on-chip design.

PRAMOD KUMAR MEHER (Senior Member, IEEE) received the B.Sc. (Hons.) and M.Sc. degrees in physics and the Ph.D. degree in science from Sambalpur University, Sambalpur, India, in 1976, 1978, and 1996, respectively. He was a Reader in electronics with Berhampur University, Berhampur, India, from 1993 to 1997, and a Professor of computer applications with Utkal University, Bhubaneswar, India, from 1997 to 2002. He has served as a Technical Consultant for the Cyber Security Research Centre, Nanyang Technological University, Singapore. He was a Senior Fellow with the School of Computer Engineering, Nanyang Technological University, from 2005 to 2009. He was a Senior Scientist with the Institute for Infocom Research, Singapore, from 2009 to 2013, and a Senior Research Scientist with the School of Computer Science and Engineering, Nanyang Technological University, from 2013 to 2016. He was a Professor-cum-Research Advisor with the C. V. Raman College of Engineering, Bhubaneswar, from 2016 to 2019. He is currently the Chief Research Advisor with Sandhaan Labs Private Limited. He has contributed nearly 250 technical papers to various reputed journals and conference proceedings, including nearly 90 articles in various IEEE TRANSACTIONS. His current research interests include signal processing, cybersecurity, intelligent computing for smart systems, the IoT, and analytics. He is a Fellow of the Institution of Electronics and Telecommunication Engineers, India. He was a recipient of the Samanta Chandrasekhar Award for excellence in research on engineering and technology in 1999. He was a Speaker of the Distinguished Lecturer Program of the IEEE Circuits Systems Society, from 2011 to 2012. He has served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, and the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION SYSTEMS. He is also a Co-Editor of the book *Arithmetic Circuits for DSP Applications* (Wiley-IEEE Press) and an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS and the *International Journal of Circuits, Systems, and Signal Processing*.

LUONG TRAN NHAT TRUNG received the B.S. degree in electronic and communication engineering from the Da Nang University of Science and Technology, Da Nang, Vietnam, in 2016, and the M.S. degree in electronic engineering from Myongji University, Yongin, Republic of Korea, in 2019. Since 2019, he has been working as an ASIC Digital Design Engineer with Solution Group, Synopsys, Da Nang. His main research interests include high-throughput and low-power digital circuit design.

HYO JIN CHO received the B.S. degree in electronic engineering from Myongji University, Yongin, Republic of Korea, in 2019, where she is currently pursuing the M.S. degree with the Department of Electronic Engineering. Her research interests include low-power digital system design, finite impulse response filter design, and adaptive filter design.



SANG YOON PARK (Member, IEEE) received the B.S. degree in electrical engineering and the M.S. and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, Republic of Korea, in 2000, 2002, and 2006, respectively. He joined the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, as a Research Fellow, in 2007. From 2008 to 2014, he was a Research Scientist with the Institute for Infocomm Research, Singapore. Since 2014, he has been with the Department of Electronics Engineering, Myongji University, Yongin, Republic of Korea, where he is currently an Associate Professor. His research interests include design of dedicated and reconfigurable architectures for low-power and high-performance digital communication systems.