Optimization techniques for hardware efficient FIR filters

高効率FIRフィルタハードウェアの最適化技術

February, 2022

Jinghao YE
葉　静浩

Optimization techniques for hardware efficient FIR filters

高効率FIRフィルタハードウェアの最適化技術

February, 2022

Waseda University Graduate School of Fundamental Science and Engineering

Department of Electronic and Physical Systems, Research on Integrated System Design

Jinghao YE
葉　静浩

Abstract

In recent years, systems containing convolutional operations have been widely used in many fields, in which the convolutional operation plays a role in filtering for selecting specific information. The main operation of convolutional systems is called multiply-accumulate (MAC) which includes multiplication and addition. The amount of MAC operations in convolutional systems is always very large, especially in some high-accuracy applications. Therefore, long computing time, high energy consumption and large circuit area are required in those applications. In order to solve those problems, high area/power-efficiency convolutional systems are required. This research takes the finite impulse response (FIR) filter as the starting point because the FIR filter is a classic convolutional system of one-dimensional or two-dimensional convolution and widely used in various digital systems for its stability. Many techniques of FIR filter designs can be applied to other more complex convolutional systems, such as convolutional neural network accelerators presented recently.

As the demand for high-quality signal processing, high-quality FIR filters generally require a large number of taps, and the corresponding bit-width of signals is also becoming longer and longer. It means a large number of multipliers, adders and registers with long bit-width are required in those FIR filters. It is foreseeable that the latency, power consumption and circuit area of those FIR filters will be very large, which leads to low area/power-efficiency of target systems. Therefore, optimization techniques for high-efficiency FIR filters are necessary. In literature, several design methods ranging from structure-level, unit-level and bit-level have been proposed for area/power-efficient FIR filters. However, there is still room for improvement in the efficiency of existing designs at those three levels. Therefore, optimization methods at three different levels are proposed with the main objective of area/power-efficiency improvement for FIR filter designs.

Firstly, a symmetric hybrid form for high-order FIR filters is proposed as a structure-level optimization method for solving the driving-ability and low-efficiency problems in the existing methods. The evaluation results show that, compared with the existing symmetric systolic form and hybrid form, the proposed symmetric hybrid form can achieve 21% area saving, 28% power reduction, 10% reduction in area delay product (ADP) and 18% reduction in power delay product (PDP) with fewer registers and multipliers. Moreover, when compared with the existing symmetric systolic form, the proposed design can achieve 33% of latency reduction.

Secondly, as a unit-level optimization method, a faithfully truncated adder (FTA) tree-based design approach is presented for designing area/power-efficient FIR filters with predefined output accuracy. With the introduction of a generic methodology for error probability analysis of faithfully truncated adder and adder tree, a method that can determine the truncate bit-width of truncated adder blocks for various area/power-efficient FIR filter designs within the accuracy loss constraint is presented. The evaluation results show that, for uniformly distributed random inputs, the proposed FTA tree-based FIR design can achieve up to 35% area saving and 30% power reduction when compared with the existing FIR designs. In addition, a fixed 6-tap FIR for electrocardiogram (ECG) signal filtering is also implemented as a case study for normally distributed signals, and the corresponding results show that 30.7X improvement of error variations, 9% area saving and 20% PDP reduction can be achieved in the proposed method, which clearly shows the accuracy and performance improvements over the existing design.

Thirdly, an adder-segmentation approach as a bit-level optimization method is introduced to shorten the critical path for performance improvement. The evaluation results show that up to 40%, 30.7% and 22.8% reduction in delay, ADP and energy delay product (EDP) can be achieved when compared with the existing FIR filter designs.

With the development of convolutional systems in signal processing and machine learning, it is believed that the three level optimization methods proposed in this dissertation can be applied and extended to design high-efficiency convolutional neural network accelerators for edge computing. However, with the consideration of the huge amount of data movements in these convolutional neural network accelerators, energy-efficient dataflow including memory access optimization should be developed as a system-level approach, which can be viewed as one of the future research directions in designing more complex convolutional systems with high-efficiency.

Acknowledgments

I would like to express my profound gratitude and appreciation to my advisor, Professor Youhua Shi, for his constant guidance, support, and encouragement throughout my years at Waseda University. He gives many of the high-quality characteristics that I aspire to emulate during my professional and personal life. Working with him has been and will continue to be a source of honor and pride for me.

I also thank Professor Masao Yanagisawa for being my associate advisor and being on my dissertation reading committee. Without him, I can't finish my doctoral study at Waseda University. I am very grateful for his invaluable help during the preparation of this dissertation and several papers.

I would like to also thank Professor Takanobu Watanabe for being on the reading committee of my dissertation. I also show my greatest appreciation to all professors of the department of electronic and physical systems. And I also thank Professor Shinji Kimura, Professor Toshihiko Yoshimasu and Professor Nozomu Togawa for valuable advice.

I have greatly appreciated all of the students and the colleagues at the Information System Lab.

Finally, I deeply appreciate my family and friends for their many years of support.

List of Acronyms

ADP	Area delay product
AMB	Addition of multiplier block
ATB	Addition of adder tree block
BS	Bit shifter
CNN	Convolutional neural network
CSD	Canonical signed digit
CSE	Common sub-expression elimination
DF	Direct form
DSP	Digital signal processing
ECG	Electrocardiogram
EDA	Electronic design automation
EDP	Energy delay product
EEG	Electroencephalogram
FFA	Fast finite impulse response algorithms
FIR	Finite impulse response
FTA	Faithfully truncated adders

HB	High bit
HF	Hybrid form
IB	Important bit
LB	Low bit
LSB	Least significant bit
MAC	Multiply-accumulate
MB	Middle bit
MCM	Multiple constant multipliers
PB	Pattern block
PDP	Power delay product
PE	Processing element
SA	Structural adder
SDF	Symmetric direct form
SF	Systolic form
SHF	Symmetric hybrid form
SSF	Symmetric systolic form
STF	Symmetric transpose form
TB	Truncated bit
TF	Transpose form

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, the motivation for this research is explained. And then, the contribution and the organization of this dissertation are illustrated.

This chapter is organized as follows. Section 1.1 gives an introduction to explain the motivation of this research. Section 1.2 outlines the main contributions of this work. Finally, Section 1.3 presents the organization of the dissertation.

1.1 Motivation

In recent years, systems containing convolutional operations have been widely used in many fields, such as image processing [1], audio processing [2], video processing [3], bioelectric signal processing [4], artificial intelligence [5], and communication [6]. The convolutional operation plays a role in filtering for selecting specific information. The example of the circuits playing this role are filters such as finite impulse response (FIR) filters [1]-[4], [7] (one-dimensional or two-dimensional convolutional systems) and convolutional neural network (CNN) architectures [5], [8]-[13] with its hardware accelerators [14]-[20] (three-dimensional convolutional systems).

The main operation of convolution systems is the multiply-accumulate (MAC) operation containing multiplication and addition. The amount of MAC operations in convolutional systems are always very large, especially in some high-accuracy applications. Therefore, long computing time, high energy consumption and large circuit area are required in those applications, which will affect our lives with long waiting time, short running time for mobile devices powered by batteries and expensive price for digital equipments. To solve these problems, high-efficiency convolutional systems are necessary.

High-efficiency has two meanings in circuit designs, such as high energy-efficiency and high area-efficiency. Depending on different application scenarios, convolutional system designs have two different goals such as high speed for servers and some communication systems and low power consumption for mobile devices. However, even if the target high-speed convolutional system is designed for server-level or desktop equipments, high energy consumption means more electricity costs and more expensive chip packaging, which will greatly increase the using or production cost. Therefore, evaluating the convolutional systems only by performance is not comprehensive, and the power consumption or energy consumption are also important. Thus, power delay product (PDP) and energy delay product (EDP) could be taken as important indicators to measure the energy efficiency of high-performance circuit designs. A lower PDP or EDP means less energy consumption is necessary to complete the same number of operations or to achieve the same operating speed, which can fully reflect the superiority of the circuit design. Similarly, as mentioned before, the production cost of a circuit is proportional to the circuit area, and a high-performance circuit design usually requires a large circuit area. Therefore, when designing circuits for convolutional systems, area delay product (ADP) is also an important indicator to measure the excellence of the design. A lower ADP means less circuit area is required to achieve the same performance (frequency). Therefore, PDP and ADP are usually used as important indicators for circuit measurement.

All convolutional systems have three different design levels from high to low for efficiency improvement, such as structure-level, unit-level, and bit-level. The structure-level designs influence the circuit structure of the computational block, such as the structure of a filter in the target system. The unit-level design focuses on the specific computation unit in a target computation block, such as the multiplier unit in a target filter. The bit-level design affects the bit operations in the target arithmetic resource, such as the carry of an adder or a multiplier. Structure-level, unit-level and bit-level designs are greatly affecting the performance of the circuit. And the design of each level can usually be integrated to achieve the purpose of improving efficiency level by level.

There are a lot of circuits included in convolutional systems. In this research, the FIR filter is taken as the starting point, because the FIR filter is a classic convolution system of one or two-dimensional convolution and is widely used in various digital systems for its stability. Moreover, many techniques of the FIR filter design can be generalized to other more complex convolution systems, such as convolutional neural network accelerators [18]-[20].

An FIR filter is a common circuit used to filter out noise or get the signal of a target frequency band. The equation of FIR filters is shown in the following:

$$Y(t) = \sum_{i=0}^{L-1} h(i)x(t-i) \tag{1.1}$$

in which, L is called the tap number of an FIR filter and $L-1$ is called the order of the FIR filter. From Equation (1.1), it can be found that, typically, an FIR filter can be regarded as a circuit consisting of L multipliers, $L-1$ adders and $L+1$ registers, including input and output registers. An L-tap FIR filter has L FIR coefficients, and they are multiplied by the L discrete input signals of the nearest L clock-cycles. And then, the output signal is generated by accumulating the results of multiplications. Depending on the requirements of different digital signal processing systems, there are mainly two kinds of FIR filters such as reconfigurable and fixed. The reconfigurable FIR filters are usually used in systems with a certain versatility which means the coefficients need to be changed for different applications, such as the digital signal processing (DSP). And the fixed FIR filters are usually used in systems with a specific role, which means the coefficients are predetermined and fixed, such as those in power-line noise filtering in Electrocardiogram (ECG) equipment.

As the demand for high quality signal processing, the bit-width of digital signals is getting larger and larger, and high-precision filtering requires a large number of taps [6], [21]-[22]. It means a large number of long bit-width multipliers, adders and registers are required in FIR filter designs. The power consumption and circuit area of these FIR filters will be very large, which leads to the low area/power-efficiency of the target application. Therefore, optimization techniques for high-efficiency FIR filter designs are necessary. The existing research for the high-efficiency FIR filter designs includes the structure-level, the unit-level and the bit-level optimization methods. The area/power efficiency of the FIR filter design is successfully improved by the existing optimization methods. However, there is still room for improvement in the efficiency of existing designs at these three design levels.

An FIR filter has four basic structure-level designs called basic forms, such as direct form (DF) [7], transpose form (TF) [23]-[28], systolic form (SF) [7], [29] and hybrid form (HF) [7], [30]. The advantages and disadvantages of these four basic forms are as follows. The direct form has the smallest area and power consumption in those four forms because all the registers in the direct form are on the input side. Compared with the registers on the output side, the registers on the input side have only half of the bit-width, which reduces the circuit area and power consumption. However, the delay of the direct form is the longest because of its long critical data path in the adder tree. The transpose form has been proposed as the solution to solve the problem of the long critical data path in the direct form. Compared with the direct form, the circuit area of the transpose form is larger, because all registers of transpose form are on the output side. Moreover, the input register is necessary to drive all multipliers following. When the order of an FIR filter becomes larger, it takes a large amount of time to transmit the input signal to all the following multipliers, which also leads to a long delay. It is called driving ability problem. Systolic form and hybrid form are proposed for solving this driving ability problem. The systolic form has a large number of registers to completely pipe-lining the computation of each tap. Compared with the other three basic forms, the critical data path of the systolic form is the shortest, while the circuit area and power consumption of the systolic form are the largest. The hybrid form is a compromise solution to solve the driving ability problem and the low area/power-efficiency problem by combining the direct form and the transpose form. The critical path of hybrid form contains one multiplier and two adders which is longer than those in systolic form and transpose form. And, moving $L/2$ registers of the transpose form from the output side to the input side, making the circuit area and power consumption of the hybrid form is smaller than those of the transpose form and the systolic forms. The existing FIR filter designs are all based on these four basic forms and the selection of basic form is based on the requirements of applications.

An effective way to improve the area/power-efficiency is to utilize the symmetry of the FIR filter [7], [31]-[32] because the absolute value of the coefficients at the symmetrical position in linear FIR filters are equal to each other. Therefore, the half number of the multipliers can be reduced by pre-adding the corresponding input data in the direct form or transmitting the result of a multiplier to the two corresponding output side adders in the transpose form. The systolic form-based symmetric method is more complicated. To halve the number of multipliers in the systolic form, a large number of extra registers for timing matching are necessary. And, there is no existing research works on the symmetric methods of hybrid form FIR filters.

Another effective way to improve the area/power efficiency is to reduce the complexity of the multiplier block through unit-level optimization because the circuit area and power consumption of a multiplier are times larger than those of an adder or a register. And, the coefficients of target FIR filters are usually pre-determined and fixed such as the FIR filter for filtering the power interference (50 Hz or 60 Hz noise) in Electrocardiogram (ECG) [4] systems, which means the FIR filter having a fixed role and unnecessary to changing the coefficients for different situations. The FIR filter with these properties is called a fixed FIR filter, in which the general multipliers can be replaced by multiple constant multipliers (MCM) [33]-[42]. For area saving and power reduction, the MCM has only shift-and-add operations. Moreover, MCMs can be combined with canonical signed digit (CSD) technologies [35]-[38] for further reduction of the area and power consumption. As the result, the structural adder tree block occupies more than 50% of the cost in the state-of-the-art fixed FIR filters [35], [39], [40]-[41]. Therefore, the unit-level optimization for the adder tree is very important in the current FIR filter design. However, the existing techniques for adder tree optimization achieve only marginal benefits, because the total number of adders or complexity of additions cannot be greatly reduced. On the other hand, the efficiency and performance can be effectively improved by the methods based on approximate computing [35], [43]-[44]. It seems that if approximate computing can be successfully applied to FIR filter designs, delay and design cost can be further reduced.

When the bit-width of input/output signal and coefficients is large, the high complexity of computation not only leads to a high-power consumption and a large circuit area but also leads to a long delay of FIR filter circuit. To solve these problems, methods for computational complexity reduction [35]-[37], [40], [45]-[46] and parallel processing [26], [37], [47] for performance improvement are proposed. And, performance or area/power-efficiency are successfully improved by these methods. However, most of them are focused on optimizing the multiplier block and the hardware complexity of the paralleled methods is also very high. And, optimization of the adder tree is also very important in the state-of-the-art fixed FIR filter designs [35], [39], [40]-[41]. As a solution, [41] proposed an optimization method in the bit-level to reduce the hardware complexity in both multiplier block and adder tree. But the method proposed in [41] is not sufficient for FIR filters with various MCM designs, such as the CSD-based MCMs.

1.2 Contributions

In this dissertation, three optimization methods in structure-level, unit-level and bit level have been proposed for high-efficiency FIR filter designs.

In the structure-level, a symmetric hybrid form for FIR filter designs with symmetric FIR coefficients is proposed for area/power-efficiency optimization. It can solve the driving ability problem of the input register in the transpose form-based FIR filter designs and the long critical data path problem in the direct form-based FIR filter designs. The evaluation results show that, compared with HF and SSF-based FIR filter designs, the proposed SHF-based FIR filter designs can effectively achieve 21% saving in circuit area, 28% reduction in power consumption, 10% reduction in ADP and 18% reduction in PDP. Moreover, compared with the existing SSF the proposed form can effectively achieve 33% reduction in latency.

In the unit-level, a method for transpose form-based FIR filter designs with CSD-based multipliers and approximate tree units is proposed for the high-efficiency FIR filter design with an acceptable error (averagely smaller than 1 *ulp*). The proposed approximate adder tree replaces all of the adders with the proposed faithfully truncated adders (FTAs). The relationship between bit-width of truncated bits in FTA tree and average error of FTA tree has been discussed mathematically for FTA tree design. The implementation results of seven fixed FIR filters show that, replacing the normal adder tree with the proposed FTA tree can achieve up to 44.5% reduction in ADP and 35.85% reduction in PDP. In order to further discuss the effect and influence of utilizing the FTA-based adder tree in FIR filters, a fixed FIR filter with predetermined FIR coefficients for exterminating the 50 Hz power noise is designed by the FTA-based adder tree, the improved GeAr(16,8,0) [43]-based adder tree and the normal adder tree. The results of the output waveforms and the implementations show that, compared with the existing GeAr(16,8,0) adder tree-based FIR filter, the proposed FTA tree-based FIR filter can get 25.7% lower power consumption and 29.3% smaller circuit area with 30.7X accuracy improvement.

In the bit-level, an advanced adder-segmentation-based optimization method for FIR filter designs is proposed in this research. When applying the proposed method, the delay is shortened by the appropriate adder-segmentation-based critical data path optimization. In addition, the optimal point for adder-segmentation with the corresponding mathematical analyzing method is also proposed, which can be applied to optimize the performance of the proposed bit-level optimization method-based FIR filters. When compared with the existing FIR filter designs, the evaluation results show that, up to 40%, 31% and 23% reduction in delay, ADP and EDP can be effectively achieved by the proposed method, respectively.

1.3 Dissertation Organization

This dissertation summarizes my research works in designs for FIR filter efficiency optimization. Detailed description of each design in three design-levels can be found in the appendices included in the published papers at Waseda Univeristy.

This dissertation is organized into five chapters. A brief description of these chapters is shown in the following.

Chapter 1 introduces the motivations and the main contribution of this research. And the whole organization of this dissertation is shown.

Chapter 2 discusses the existing designs of FIR filters in the structure-level. And then, as a solution to those problems, a symmetric hybrid form is proposed for area/power-efficiency improvement.

Chapter 3 discusses the existing methods for unit-level FIR filter designs. And then, an area/power-efficiency optimization method with an approximate adder tree is proposed for FIR filter designs.

Chapter 4 introduces the existing FIR filter designs in the bit-level. And then, an adder-segmentation-based bit-level optimization method is proposed for further performance and efficiency improvement while without accuracy loss.

Chapter 5 presents the conclusions and several future research directions which can be expanded based on the optimization methods of this dissertation.

Chapter 2

Structure-Level Optimization

This chapter introduces the existing designs of structure-level and discusses the problems of each existing form for FIR filter designs. Moreover, as a solution to those problems, a symmetric hybrid form for area/power-efficiency improvement is proposed.

The content of this chapter is shown as following. Section 2.1 discusses the existing four basic forms and three symmetric forms for FIR filters. Section 2.2 presents the proposed symmetric hybrid form for FIR filters. Section 2.3 presents the evaluation and comparison result by FIR filter implementations. Finally, Section 2.4 gives a summary of this chapter.

2.1 Existing Forms of FIR Filters

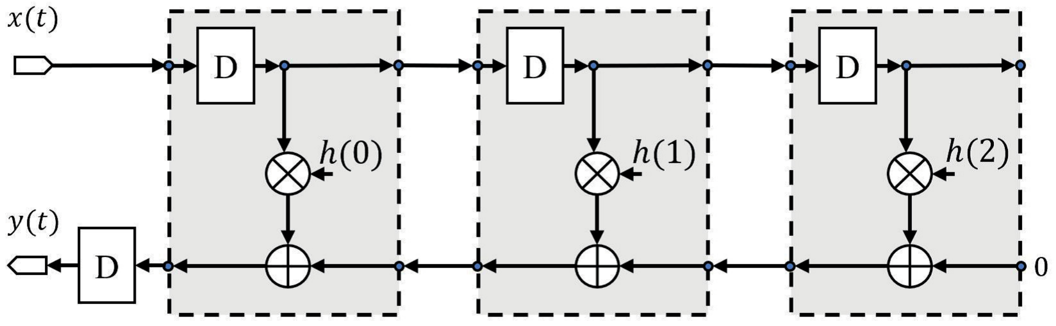
2.1.1 Basic Forms

Most of the existing FIR filters are based on the two basic forms shown in Fig. 2.1, in which (a) is the direct form and (b) is the transpose form. As shown in Fig. 2.1, the circuit of the one-tap FIR filter in the botted boxes can be seen as the basic-units of those two forms. Obviously, in the direct form, the critical data path increases with the number of taps. Totally, $1 + \lfloor log_2 L \rfloor$ adders and one multiplier are contained in this critical data path, in which L is the tap number. Therefore, in the direct form, when the tap number is large, the delay will be very long. On the other hand, no matter how large the tap number L is, only one multiplier and one adder are contained in the critical data path of the transpose form. Therefore, compared with the direct form, higher performance can be achieved in the transpose form, which is the main reason that most of the FIR filter designs such as [22]-[28] are based on the transpose form.

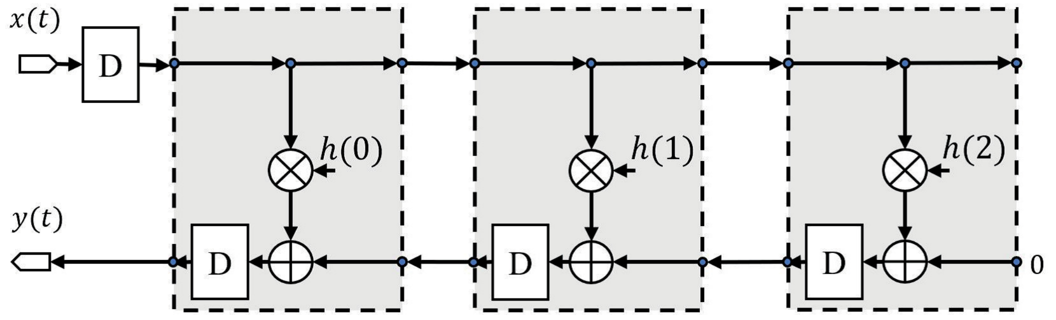
Generally, a large L is required in FIR filter designs, such as the filter presented in [48] contains 512-tap, due to the accuracy of frequency domain usually needs to be high. However, the registers for storing the input data of transposed form need to driver all the connected L multipliers. When the tap number of a target FIR filter is very large, it will lead to a long delay and the reduction of filter frequency, which is called the driving ability problem. To solve this problem, Fig. 2.2 shows two solutions to this problem for high-speed filtering, such as systolic form [7], [29] and hybrid form [7], [30], in which the basic-unit of each form is in black dotted box. There are two advantages when applying the systolic form. First, no matter how many taps an FIR filter has, the registers are only necessary to drive one multiplier and one register. Second, only one adder and one multiplier are contained in the critical data path. However, one disadvantage exists in the systolic form. In each tap, two extra registers, one on the input side and one in the adder tree, are necessary, which not only generate low area/power-efficiency but also lead to a long latency between input and output.

2.1.2 Basic Symmetric Forms

The coefficients of linear-phase FIR filters are symmetric, which means $|h(i)| = |h(L-1-i)|$, where L is the tap number. Based on this symmetry, half the number of multipliers can be reduced for area and power reduction. Fig. 2.3 shows the symmetric direct form and transpose form (SDF & STF) [7] with their basic-units in dotted boxes. Because area and power consumption of multipliers are generally much larger than those of adders and registers, significant improvement of area/power efficient can be achieved by utilizing the symmetry of the coefficients. But when combined with the systolic form, a large number of extra registers are necessary to match the timing. As an example, a symmetric systolic form (SSF) is proposed in [31] as shown



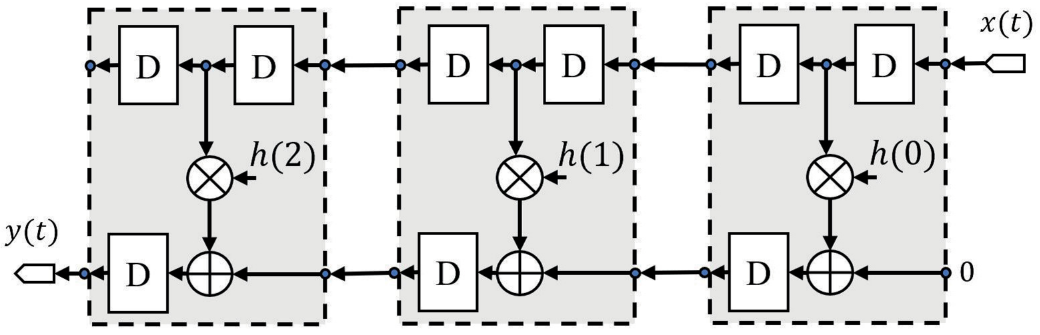
(a)



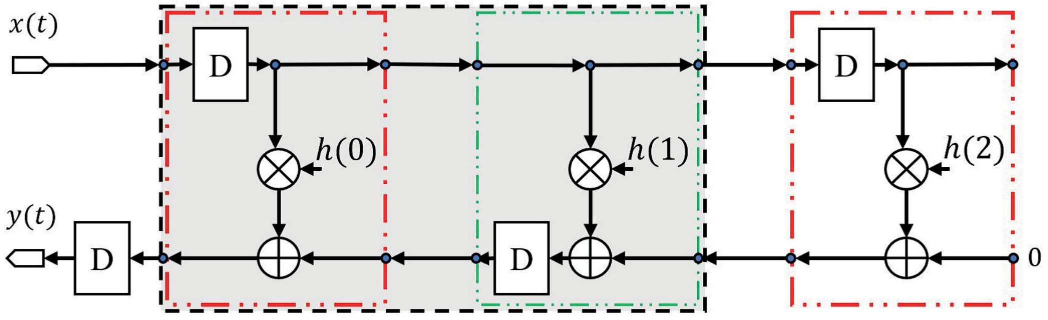
(b)

Figure 2.1: General 3-tap FIR implementations: (a) direct form and (b) transpose form.

in Fig. 2.4 (a), in which the number of multipliers is half-subtracted by the symmetry of a linear-phase FIR filter. However, the circuit in Fig. 2.4 (a) is not completely pipelined. The input register has the driving ability problem because all adders on the input side are necessary to be driven by the input register. A more complex symmetric systolic form as shown in Fig 2.4 (b) was proposed in [32], in which the basic-units have been completely pipelined [32]. However, in each basic-unit, three extra registers in Fig. 2.4 (b) (two on the input side and one on the output side) are necessary, which leads to not only additional area overhead and power consumption overhead but also long latency. When the tap number is not very large (e.g., 16-tap), the performance of symmetric systolic form proposed in [31] is better than that proposed in [32], because the driving ability problem is not obvious. However, when



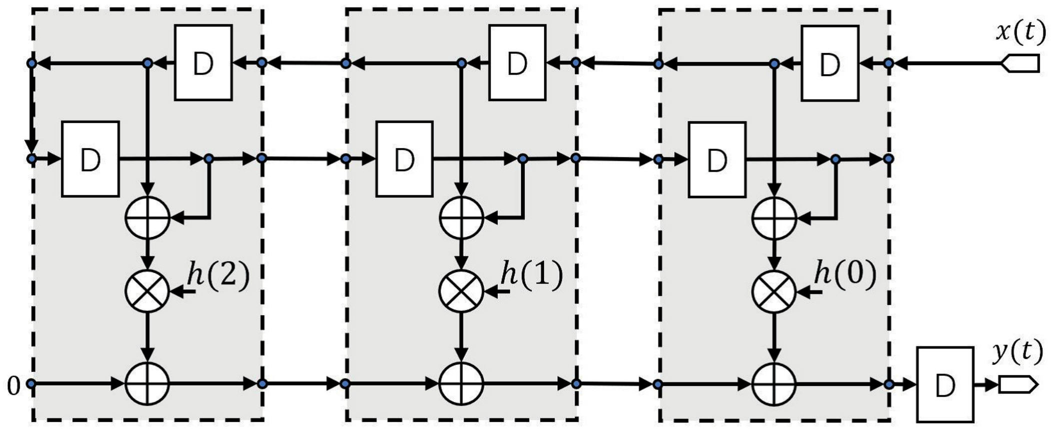
(a)



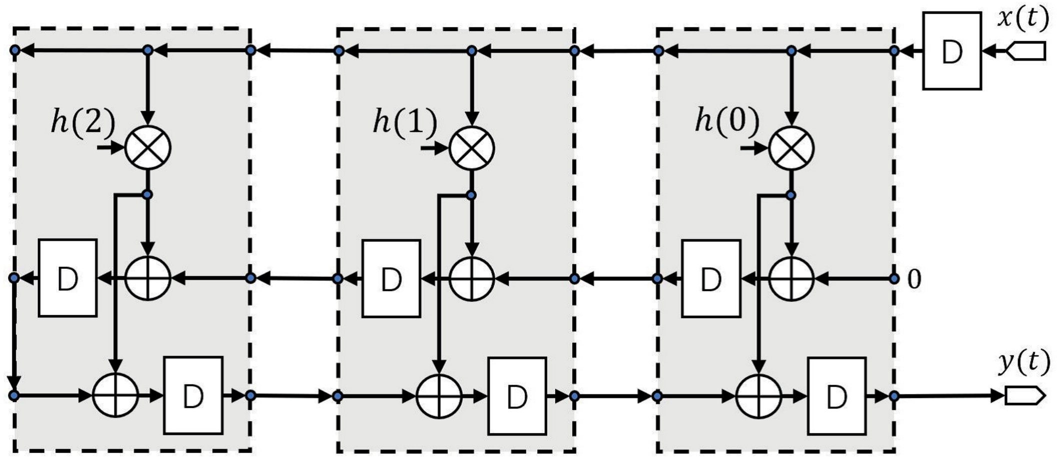
(b)

Figure 2.2: 3-tap FIR implementations: (a) systolic form and (b) hybrid form.

the number of taps is very large (e.g., 512-tap), the driving ability problem of the existing symmetric systolic form in [31] will be obvious and lead to a long delay. On the other hand, the symmetric systolic form [32] requires times of more registers with the corresponding latency.



(a)



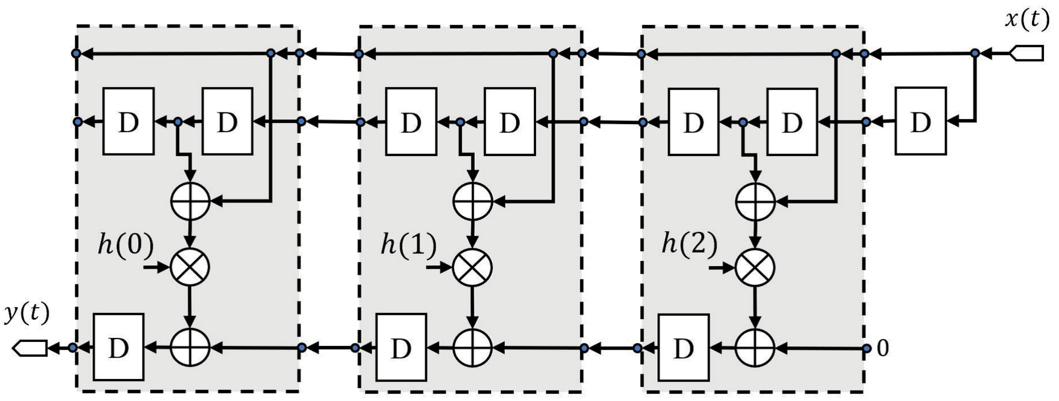
(b)

Figure 2.3: 6-tap symmetric FIR implementations: (a) symmetric direct form and (b) symmetric transpose form.

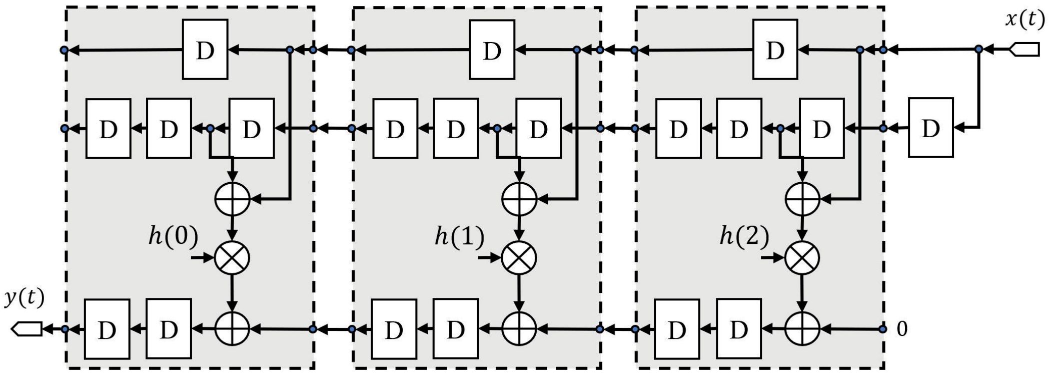
2.1.3 Discussions

The previous two subsections discussed the existing four basic forms and three corresponding symmetric forms of FIR filters. In conclusion, their problems are shown in Table 2.1.

Compared with the systolic form FIR, the number of registers required in hybrid form FIR is much smaller. And, different from the direct form, the length of the critical data path of hybrid form is stable, which means no matter how many taps the target FIR filter has, the resources contained in the critical data path are only two



(a)



(b)

Figure 2.4: 6-tap FIR implementations in two types of symmetric systolic forms.

Table 2.1: Problems of the existing FIR forms

FIR forms	Problems
DF/SDF	The delay is increasing with the tap number.
TF/STF/SSF [31]	The driving ability problem becomes severe in large FIR filters.
SF/SSF [32]	The circuit area is large and the power consumption is high.
HF	The symmetry is very difficult to apply.

adders and one multiplier. Moreover, different from the transpose form, the hybrid form does not have the driving ability problem. Therefore, when the tap number of a specified FIR filter is large, the hybrid form shows its great potential. Moreover, no existing research work realizes the symmetric hybrid form (SHF) FIR filter, because of its difficulty of halving the number of multiplier number by applying coefficient symmetry. Therefore, a symmetric hybrid form is proposed in this research for FIR filter designs in order to solve the driving ability problem of the input register in large-tap FIR filters for performance and efficiency improvement. And, controlling the latency of the proposed form is the smallest, which means $L-1$ clock cycles for a L-tap FIR filter.

2.2 Symmetric Hybrid Form

2.2.1 Basic Unit and Its Connection

Fig. 2.2 (b) shows the basic-unit of HF, which is the combination of the TF and the DF. Like that, the basic-unit of the proposed SHF is also composed of SDF and STF shown in Fig. 2.3.

Unlike the basic-unit of TF and DF that can be directly connected together while not causing any timing confusion, the basic-unit of STF and SDF cannot be easily connected together in the SHF. As shown in Fig. 2.3, the basic-unit of STF has two paths on the output side and only one path on the input side. On the other hand, the basic-unit of SDF has two paths on the input side and only one path on the output side. Therefore, due to the input/output difference, they cannot be easily connected together.

The basic-unit of the proposed symmetric hybrid form and the corresponding connection are shown in Fig. 2.5 as an example of the proposed SHF-based 8-tap

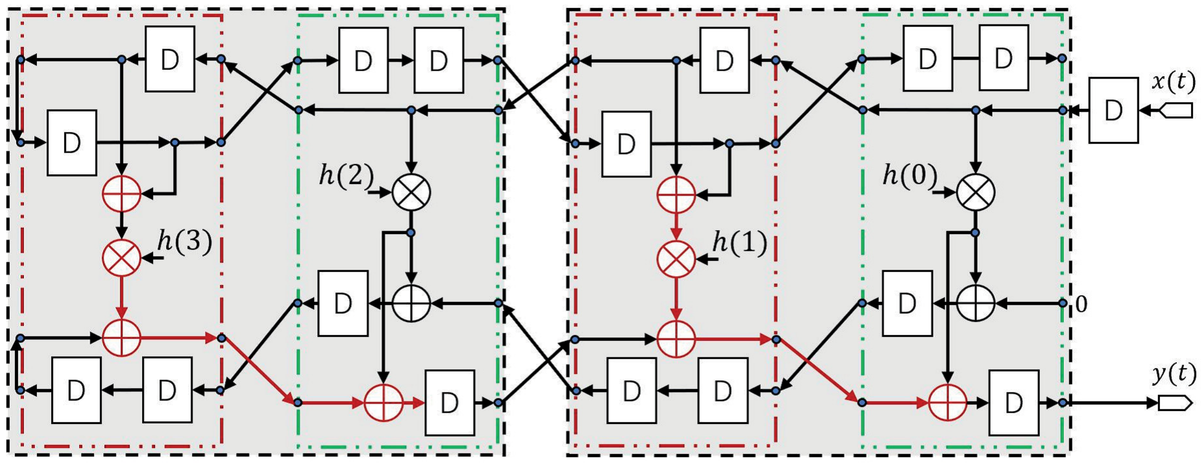


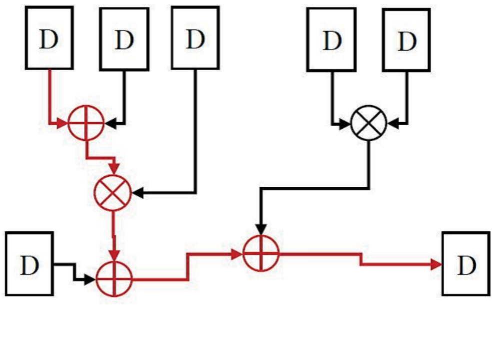
Figure 2.5: An example of 8-tap FIR implementation with the proposed basic-unit of symmetric hybrid form (in black dotted box); in which the circuit which like the basic-unit of symmetric transpose form is in green dotted box, and the circuit which like the basic-unit of symmetric direct form is in red dotted box. The red path combine the resources contained in the critical data path (© [2020] IEEE).

FIR filter. As shown in Fig. 2.5, the basic-unit of the proposed symmetric hybrid form is connected by the basic-unit of symmetric transpose form and symmetric direct form in green dotted box and red dotted box. In the basic-unit of the proposed form, two extra paths are necessary. One path is in the input side of the basic-unit of symmetric transpose form. And, the other path is in the output side of the basic-unit of symmetric direct form. Due to the basic-unit of symmetric transpose form and symmetric direct form contain the computation of two taps, the increased paths need to insert two extra registers in each of them to match the time of nearby basic-units. Therefore, the basic-unit of the proposed symmetric hybrid form has eight registers, four adders and two multipliers for the computation of four taps. The hardware resources that need to be driven by each register are only one register, one adder and one multiplier.

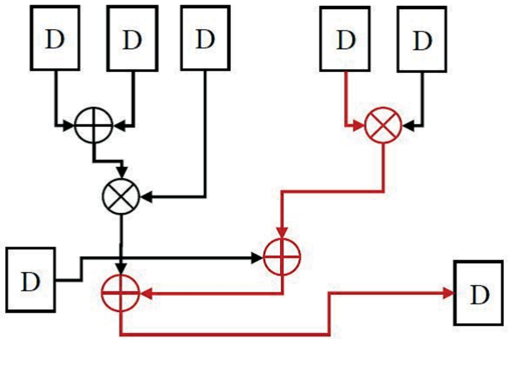
As the path in red shown in Fig. 2.5, hardware resources contained in the critical data path of the proposed symmetric hybrid form are three adders and one multiplier. However, there is still room for critical data path optimization. Fig. 2.6 (a) shows the resources related to the critical data path, and it can be optimized to the circuit shown in Fig. 2.6 (b) through reconnection. The hardware resources colored by red in Fig. 2.6 are located in the critical data path of each circuit. After optimizing, the critical data path contains only two adders and one multiplier. It is necessary to be mentioned that the bit-width of adders before and after multipliers are different. After multiplying an n-bit FIR coefficient with an n-bit input data, the result of the multiplier is in $2n$-bit. Therefore, the input bit-width of the adders before the multipliers is n-bit, and the input bit-width of the adders after multipliers is $2n$-bit. As shown in Fig. 2.6 (b), two adders after multipliers and one multiplier are contained in the critical data path of the proposed symmetric hybrid form. It is the same with the length of the critical data path of hybrid form. But, the critical data path of the proposed form is longer than that of the symmetric systolic form, because one of the two adders contained in the critical data path of the symmetric systolic form is before multipliers.

2.2.2 Extension to Arbitrary-Tap FIR Filters

As mentioned in the previous subsection, the computation of four taps is contained in one basic-unit of the proposed SHF. Therefore, FIR filters cannot be designed by directly connecting the basic-unit of the proposed SHF together, if the tap number of them is not multiple of four. To solve this problem, extended forms of the proposed SHF for arbitrary-tap FIR filters are presented in Fig. 2.7. As shown in Fig. 2.7 (a), when L=4, 8, 12, ..., $4k$, the FIR filters can be designed by directly connecting the basic-unit of the proposed SHF together. When L=1,5,9, ..., $4k$+1, except the basic-unit of the proposed SHF, an extra circuit is required for the left one-tap computation. The basic-unit of the TF can be utilized here because the final circuit in basic-unit of



(a)



(b)

Figure 2.6: Optimization of the critical data path, (a) critical data path before optimization and (b) critical data path after optimization (© [2020] IEEE).

the proposed form is like the basic-unit of the SDF and connecting it by the basic-unit of the DF or the SDF will increase one adder in the critical data path. When L=2, 6, 10, ..., $4k$+2, the proposed design is shown in Fig. 2.7 (c), except for the proposed basic-unit, an extra circuit is required for the computation of the left two taps. The

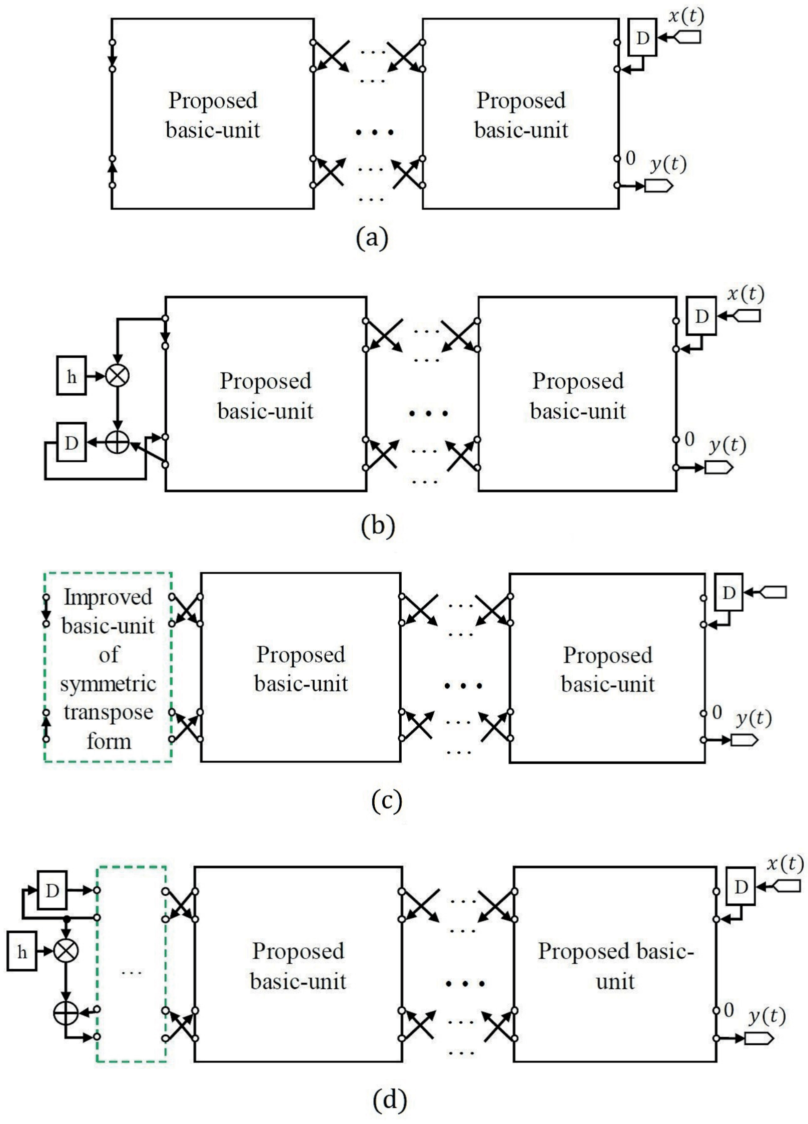


Figure 2.7: Proposed symmetric hybrid form for arbitrary-tap FIR filter implementations: (a) L = $4k$, (b) L = $4k$+1, (b) L = $4k$+2 and (c) L = $4k$+3 (© [2020] IEEE).

circuit shown in the green dotted box in Fig. 2.5 can be used here. When L=3,7,11, ..., $4k$+3, the basic-unit of the DF and the circuit shown in the green dotted box in Fig. 2.5 can be utilized for the left 3-tap computation, and the final result is shown in Fig. 2.7 (d).

Therefore, regardless how many taps an FIR filter has, the proposed SHF can be used to implement it and the critical data path in the proposed SHF only contains one multiplier and two adders. Moreover, each register in the proposed symmetric hybrid form needs to drive at most one register, one adder and one multiplier. In addition, the latency between the input register and the output register in the proposed SHF is only $L + 1$ clock cycles, which is the minimum latency of an L-tap FIR filter.

Table 2.2: Comparisons of hardware complexity for an L-tap FIR filter (© [2020] IEEE)

FIR Form	Register Number	Adder Number	Multiplier Number	Latency (clock cycle)	Driving Resources	Critical Data Path
SF [7]	$3L-1$	$L-1$	L	$2L$	$1mul+1reg$	$1mul+1add$
HF [7]	$L+1$	$L-1$	L	$L+1$	$2mul+1reg$	$1mul+2add$
SSF [31]	$3\lceil L/2\rceil+1$	$L-1$	$\lceil L/2\rceil$	$L+1$	$\lceil L/2\rceil add+1reg$	$1mul+2add$
SSF [32]	$3L-2$	$L-1$	$\lceil L/2\rceil$	$L+\lceil L/2\rceil$	$1add+1reg$	$1mul+2add$
Proposed SHF	$2L-1$	$L-1$	$\lceil L/2\rceil$	$L+1$	$1mul+1add+1reg$	$1mul+2add$

* The bit-width of adders are varied with methods.
* The input and output registers are included in the register number of each form.
* The latency indicates the number of clock cycles between the first valid output sample and the first valid input sample is the latency.

2.3 Evaluation and Comparison Results

To prove the effectiveness of the proposed symmetric hybrid form, the implementation and comparison results of various FIR filters are provided in this section.

2.3.1 Hardware Complexity

As a general form for arbitrary FIR filter designs, the hardware complexity of the proposed symmetric hybrid form is compared with those of the systolic form [7], the hybrid form [7] and the symmetric systolic forms [31]-[32]. And, the results are shown in Table. 2.2.

Compared with the hybrid form, the proposed symmetric hybrid form adds extra $L-2$ registers to achieve the purpose of half-subtracting the number of multipliers by the symmetry of coefficients. And compared with the systolic form proposed in [32], the proposed form can save $L-1$ registers. Except for the symmetric systolic form proposed in [31], which has the driving ability problem, the proposed form is the most economical in terms of required hardware resources. Moreover, the latency of the proposed form is only $L+1$ clock cycle, which is the limit of an L-tap FIR filter. However, compared with the SSF proposed in [31]-[32], the critical data path of the proposed SHF is longer because it contains one adder with a larger bit-width.

Table 2.3: Implementation results of 16-tap and 512-tap 12-bit FIR filters in various forms

L (tap)	FIR Form	Delay (ns)	Area (um^2)	Power (mw)
16-tap	SF [7]	6.9	352529	19.6
	HF [7]	8.6	236924	13.8
	SSF [31]	7.8	171489	13.3
	SSF [32]	7.8	240810	18.2
	Proposed SHF	8.8	169850	11.8
512-tap	SF [7]	6.9	11224875	359.1
	HF [7]	8.8	7606045	208.8
	SSF [31]	11.8	4632771	242.1
	[32]	7.8	6915051	255.8
	Proposed SHF	8.9	5472521	184.4

2.3.2 Comparison Results of 16-tap and 512-tap FIR Filters

For evaluations and comparisons, 12-bit (include input data, output data and FIR coefficients) reconfigurable FIR filters in 16-tap and 512-tap are implemented by applying the proposed symmetric hybrid form and other existing forms. The corresponding logic synthesis results are shown in Table 2.3, by utilizing the logic synthesis tool Design Complier (Synopsys) with the library of Rohm 180nm, in which delay, area and power consumption are the comparative issues. Power delay product, and area delay product are compared and shown in Fig. 2.8, in order to present a more comprehensive evaluation.

As shown in Fig. 2.8 (a), for the 16-tap FIR filters, the SSF [31] has the best ADP and PDP, which is because 1) 16-tap is not large, 2) driving an adder is easier than driving a multiplier and 3) compared with the SSF [32] and the proposed SHF, the SSF [31] has the minimum number of registers. However, when the tap number grows up to 512, due to the driving ability problem, the delay of the SSF [31] becomes extremely long. Therefore, the corresponding PDP of SSF [31] becomes the worst of the five forms.

Compared with the SF [32], it is obvious that the proposed SHF can improve the performance by reducing the number of registers and shortening the latency to only $L + 1$ clock cycles. For the 16-tap FIR filters, the proposed SHF can not only save 29% of circuit area, 20% of ADP, 35% of power consumption and 27% of PDP, but also reduce the latency of seven clock cycles. For the 512-tap FIR filters, the proposed SHF can not only save 21% of circuit area, 10% of ADP, 28% of power consumption and 18% of PDP, but also reduce the latency by 255 clock cycles. Compared with the HF introduced in [7], the proposed SHF can achieve 28% of area saving, 27% of ADP reduction, 12% of power saving and 11% of PDP reduction.

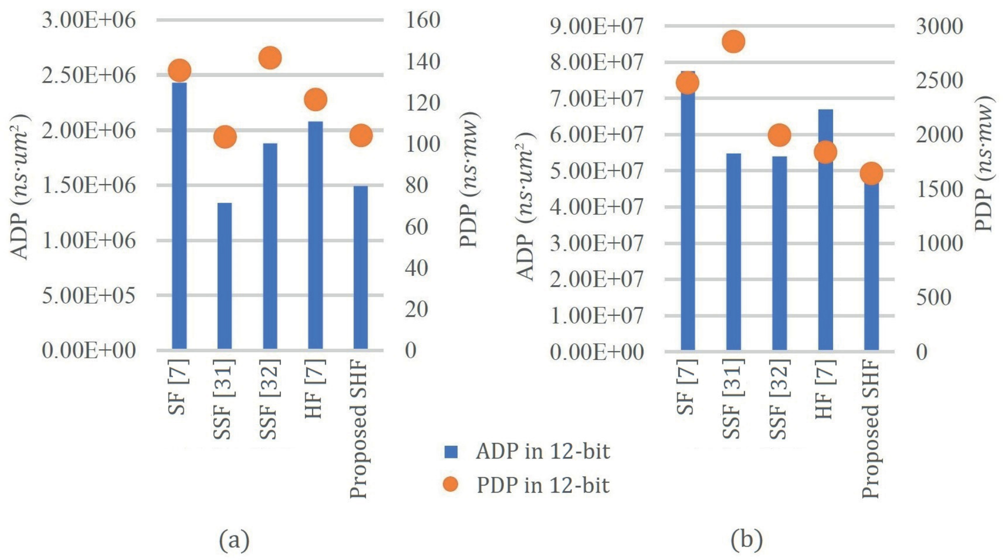


Figure 2.8: ADP and PDP comparisons in various 12-bit FIR filters, (a) 16-tap and (b) 512-tap.

2.4 Chapter Conclusion

In this chapter, a symmetric hybrid form is proposed as a structure-level optimization method for FIR filters. Firstly, the existing forms and their problems are discussed. Secondly, the proposed symmetric hybrid form for FIR filter to solve the problems of the existing forms is presented. Finally, the evaluation results show that, compared with the existing symmetric systolic form [32] and hybrid form [7], the proposed form can achieve great area and power saving with PDP and ADP improvement and latency reduction, which clearly shows the improvement in performance and efficiency of the proposed method. In conclusion, the proposed symmetric hybrid form is the most suitable in the implementation of high-order FIR filters for high-speed and real-time applications.

Chapter 3

Unit-Level Optimization

This chapter introduces the existing unit-level optimization methods and discuses their problems. As a solution, an approximate computing-based unit-level optimization method of unit-level is proposed.

The content of this chapter is shown as following. Section 3.1 discusses the existing unit-level optimization methods and shows the necessity of the optimization in the adder tree of an FIR filter. And, as an effective solution of unit-level optimization, approximate computing-based methods are also introduced. Section 3.2 shows the proposed faithfully truncated adder-based (FTA-based) unit-level optimization method. Section 3.3 presents the static error analysis and the optimization methods of the proposed FTA and FTA tree. Section 3.4 and 3.5 evaluate the proposed method with uniformed distributed random input and ECG signals to show the improvement of the proposed method. Finally, Section 3.6 gives the concluding mark of this chapter.

3.1 Existing Unit-Level Optimization Methods

As mentioned in the previous chapter, an FIR filter is mainly implemented either in direct form or in transpose form as shown in Fig. 2.1. In an FIR filter, input data is multiplied by a series of FIR coefficients. And then, the results of multipliers are accumulated to generate the output data. As a result, a large number of the MAC operations are required in FIR filters, which leads to very high computation complexity. For this reason, various high-efficiency FIR designs and algorithms [33]-[42] in the unit-level have been proposed in the past decades.

Since the FIR coefficients are usually pre-determined and fixed, which can be viewed as constants, the general multipliers in an FIR filter can be replaced by multiple constant multipliers (MCM) which contain only shift-and-add operations for the purpose of area saving and power reduction [35]. Moreover, if combined these MCMs with the methods of canonical signed digit (CSD) [35]-[38], the number of the shift-and-add operations can be reduced by ensuring no consecutive non-zero bits, and further design cost reduction can be achieved. As an example, Fig. 3.1 gives the corresponding MCM and CSD-based MCM operations of a 4-bit coefficient and the input data $X(t)$.

As the result of the existing multiplication optimization, it was reported in [35], [38], [40] and [41] that in the state-of-the-art fixed FIR filter implementations, the structural adder tree block occupies more than 50% of the power consumption and the circuit area. For this reason, several unit-level optimization methods have been

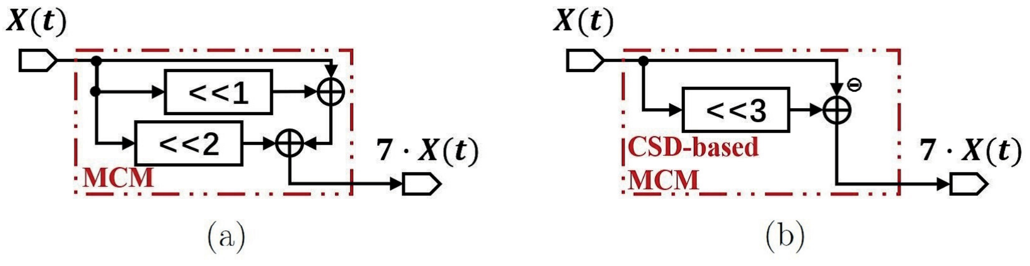


Figure 3.1: Operations of coefficient = 2b'0111, (a) MCM and (b) CSD-based MCM.

proposed for improving the efficiency of the adder tree block [41]-[42]. However, these techniques achieve only marginal benefits, because either the total number of adders or the complexity of additions cannot be greatly reduced.

Since the approximate computing-based methods such as those proposed in [35], [43] and [44] are effective ways in efficiency improvement while at the cost of accuracy loss, it seems that if approximate computing can be successfully applied to the FIR filter designs, delay and design cost can be further reduced.

For the above reasons, an error-controllable approximate computing method-based adder tree is proposed as a unit-level optimization method for improving the area/power-efficiency of FIR filters while without causing an unacceptable accuracy loss.

3.2 FTA Based FIR Filter Design

The proposed faithfully truncated adder (FTA) -based FIR filter design is shown in Fig. 3.2, which is built on the top of state-of-the-art CSD-based FIR filter designs. The CSD-pattern multiplication block proposed in [37] was adopted due to its low-power consumption and small-circuit area. The structural adders (SAs) are replaced by shorter bit-width adders called faithfully truncated adders.

Typically, as mentioned in the previous chapter, an L-tap FIR filter consists of L multipliers, $L-1$ adders, and $L+1$ registers (including registers for storing the input data and output data), therefore the hardware resources required in FIR filter designs depends on the number of taps (L) and the bit-width of input/output data and FIR coefficients for a required accuracy. Due to the requirement of accuracy in the frequency domain, L is generally assumed to be large, which incurs a large silicon area and leads to significant power consumption as well. Therefore, the number of registers and SAs needed in the accumulation block increases linearly with the order of the FIR filter. Moreover, in an FIR filter, the accumulation process leads to the increase of the bit-width of the corresponding registers and SAs. However, the final output

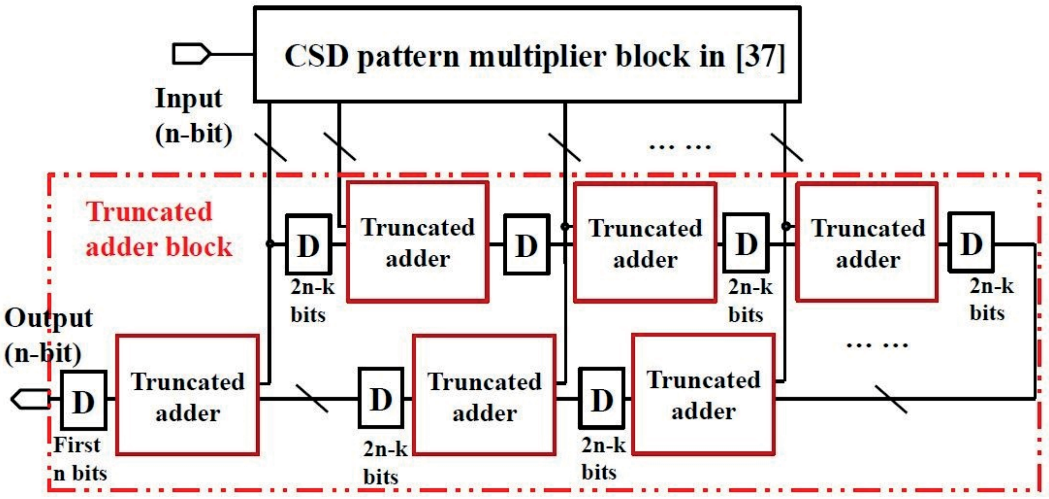


Figure 3.2: Proposed n-bit faithfully truncated adder-based FIR filter design with CSD-pattern multiplier block (Copyright (c) 2020 IEICE, [34] Fig. 2).

data of the FIR filter has the same bit-width as the input data, which indicates the possibility of bit-width reduction of SAs for area/power-efficiency improvement while minimizing the output accuracy loss. Therefore, the analysis and the optimization method of static error will be discussed in the following for the proposed FTA-based FIR filter designs.

3.3 Static Error Analysis and Optimization

As shown in Fig. 2.1 and mentioned in the previous chapter, when considering an L-tap n-bit FIR filter, typically, the addition is performed by adding L $2n$-bit data from L multipliers, while only the most significant n bits in the result of the adder tree will be finally output [44]. Due to its bit-width is different between the input data and output data of the adder tree, the least significant n bits of the input data of adders (i.e., the outputs of multipliers) cause significant power consumption and need large circuit area for computation while having little effects on the final output, which can be effectively explored for further power consumption reduction and circuit area saving.

As show in Fig. 3.2, three kinds of adders can be utilized for adder tree unit designs, they are $2n$-bit normal adder, n-bit adder truncated adder [50] and $(2n-k)$-bit truncated adder, respectively. When shorter adders are used in the adder tree, lower power consumption and smaller circuit area can be achieved while with higher accuracy loss. Therefore, with the considerations of area saving and power reduction with acceptable accuracy loss, an improved adder tree design with the truncated adder for area/power-efficient FIR filter designs is proposed in this section. The problem that needs to be solved can be defined as how to determine the bit-width of the adders utilized in the adder tree shown in Fig. 3.2(c) (i.e., the method to determine the largest truncated bit-width (k)) for various FIR designs to get significant area saving and power consumption reduction while meeting the accuracy loss constraint.

According to [44], the average accuracy loss of FIR filter designs should be smaller than one unit of last place (ulp), which is defined as the weight of the least significant bit (LSB) in the output. Therefore, ulp is utilized as the unit of accuracy measurement.

In the following, static error analysis of both the single individual truncated adder and the truncated adder tree will be performed. And then, an accuracy/efficiency-optimal truncated adder tree configuration method for the L-tap n-bit FIR filter will be presented.

3.3.1 Error Analysis of Individual Truncated Adder

Assuming that the inputs of an adder are in $2n$-bit, for a $(2n-k)$-bit truncated adder, the least significant k bits of each $2n$-bit input will be truncated, and only the most significant $(2n-k)$-bit of the inputs are added together. As a result, the $2n$-bit adder is now changed to the $(2n-k)$-bit adder shown in Fig. 3.2 (c), in which power and area efficiency can be improved due to the reduction of bit-width. In addition, it also slightly helps to raise the speed of addition. However, due to the truncated bits, the accuracy loss will occur.

As shown in Fig. 3.3, in the following, the most significant n bits, the middle $(n-k)$ bits, and the least significant k bits in a $2n$-bit input are indicated as important bits (IBs), middle bits (MBs) and truncated bits (TBs), respectively. The shaded parts in Fig. 3.3 are the computations of bits that we want to eliminate or truncate for area saving and power reduction.

From Fig. 3.3, it is very clear that, if the least significant k bits in the $2n$-bit inputs are truncated, 1 *ulp* error in the final n-bit output would occur when the generated carry of MB adder (C_{MB}) produced by the original generated carry of TB adder (C_{TB}) and affects the final n-bit output.

First, the probability of the carry generation by truncating k bits in the k-bit TBs

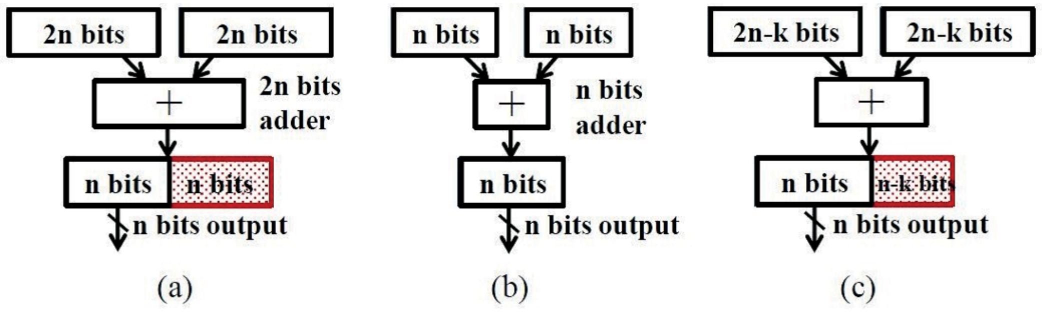


Figure 3.3: Three possible adder designs with various bit-width for FIR filters, (a) normal adder, (b) truncated adder and (c) proposed truncated adder (Copyright (c) 2020 IEICE, [34] Fig. 3).

adder can be calculated as the following equation:

$$p_{C\_TB(k)} = (2^k - 1)/2^{k+1} \tag{3.1}$$

Next, the probability of the generated carry of TBs adder(C_{TB}) will propagate through the $(n-k)$ bits MBs adder and generate a carry (C_{MB}) to affect the LSB of the final n-bit output causing one *ulp* error. Considering the MBs adder shown in Fig.3.3, if the bit-width of MBs is one (i.e., n-k=1), when the addition of MBs is $1+1$, no matter 0 or 1 the (C_{TB}) from the TBs adder is, a carry (C_{MB}) will be generated by the MB adder. Similarly, if the addition of MB is $0+0$, no matter the C_{TB} is 1 or 0, no carry will be generated in the MB adder. Only when the addition of MB is $1+0$, the carry from the TBs adder will affect the carry of MB adder. Taking these into consideration, for a $(n-k)$-bit MBs adder, the probability for a generated carry from TBs propagated through the $(n-k)$-bit MBs adder and affecting the LSB of the n bits output causing one *ulp* error can be calculated through Equation (3.2) and expressed in following.

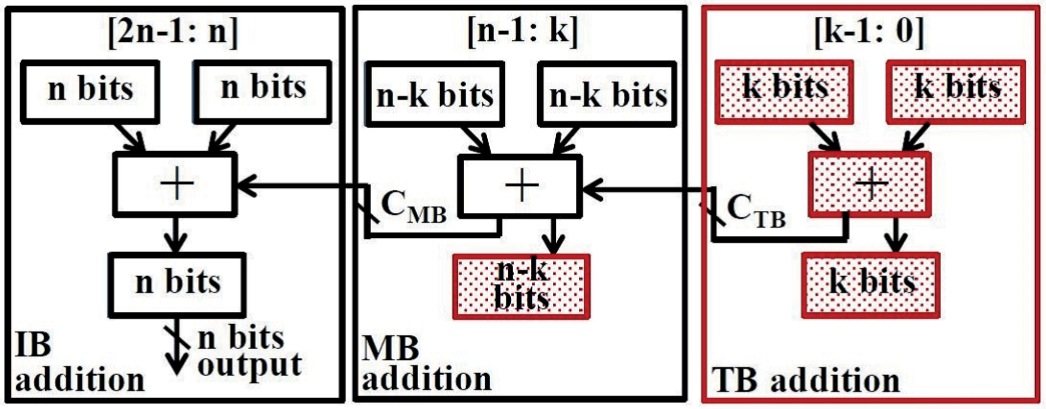


Figure 3.4: Details of $(2n-k)$-bit Truncated Adder (Copyright (c) 2020 IEICE, [34] Fig. 4).

$$p_{C\_MB(k)} = \frac{(2^k - 1)}{2^{k+1}} \times \frac{1}{2^{n-k}} = \frac{2^k - 1}{2^{n+1}} \tag{3.2}$$

3.3.2 Error Analysis for Accuracy-Area Optimization

From the discussion of the previous subsection, it is easy to calculate the probability of error in an individual truncated adder. However, in FIR filter designs, generally, there are more than two outputs from multipliers, and they would be added together by applying an adder tree. Therefore, the error analysis for an adder tree using identically truncated adders is performed in this work. For example, in a small 6-tap FIR filter with n-bit inputs, n-bit coefficients and n-bit outputs, the adder tree has 5 identical individual adders which perform the addition of 6 independent $2n$-bit inputs to generate one n-bit output as shown in Fig. 3.4, in which the equivalent adder tree unit with IB, MB, and TB adder trees are also shown. The carries generated by MBs and TBs adder tree blocks are indicated as C_{MB} and C_{TB}, respectively. It should be noted that, unlike a single individual adder, for an adder tree block, the bit-width of generated carries (i.e., C_{MB} and C_{TB}) is not in one bit but depends on the tap number of FIR filters. To analyze the error in the final result, carries generated by

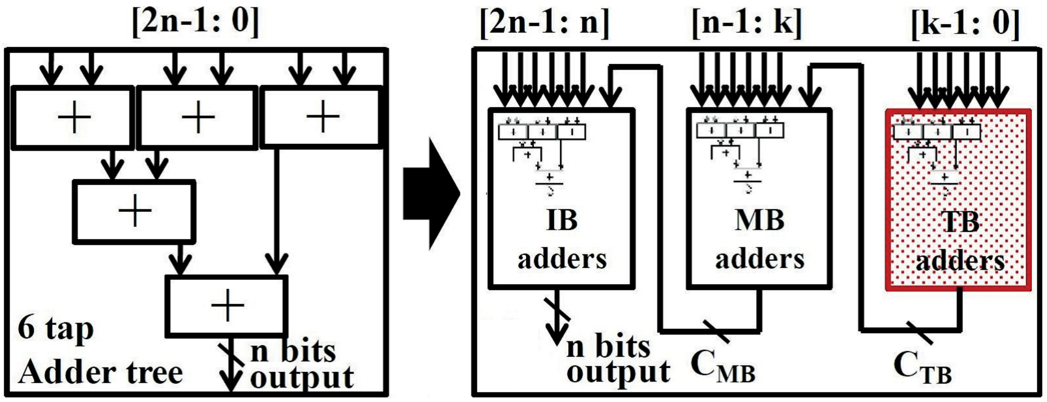


Figure 3.5: A 6-tap n-bit adder tree block with the equivalent representation (Copyright (c) 2020 IEICE, [34] Fig. 5).

TBs adder tree block that will affect the final n bits output should be considered.

According to Equation (3.1), the probability of a carry generation in each individual k-bit adder is $\frac{(2^k-1)}{2^{k+1}}$, therefore the probability ($P_{(k,N,L)}$) of each generated carry (N) in the TBs adder tree can be expressed as

$$P_{(k,N,L)} = C_{L-1}^{N} \times (\frac{2^k-1}{2^{k+1}})^N \times (1 - \frac{2^k-1}{2^{k+1}})^{L-N-1} \tag{3.3}$$

where k and L indicates the truncated bit length and the tap number, respectively, and N is the possible generated carries in the ranging of 0 to $L-1$. Because the bit-width of the MBs adder tree in this work is $(n-k)$, the conditions of the relationship between $n-k$ and N will be discussed in the following.

Firstly, assuming $N < 2^{n-k}$ (i.e., the carries (N) generated in TBs adder tree is less than 2^{n-k}), the generated carries (N) cannot directly affect the final n-bit output; however, it might be combined with the addition of corresponding MBs to generate a carry to affect the LSB of the final n-bit output causing one *ulp* error. The carriers generated in the TBs addition (N) can affect the LSB of the final n-bit output requires that the sum of the MBs should be larger than $2^{n-k} - N$, and then the probability of error (1 *ulp*) generation in the final n-bit output can be calculated as:

$$P_{1ulp} = \frac{N}{2^{n-k}} \times P_{(k,N,L)} \tag{3.4}$$

Next, assuming N is greater than or equal to 2^{n-k}, because the TBs adder tree consists of $L-1$ k-bit adders, the carries (N) generated in TBs adder tree in the range of 0 to $L-1$. If $N \geq 2^{n-k}$, the part beyond MBs (i.e. $\lfloor N/2^{n-k} \rfloor$) in the generated carries (N) will directly affect the final n-bit output and cause $\lfloor N/2^{n-k} \rfloor$ *ulp* errors. Now, rewrite N as $N1 + N2$, where $N1 = 2^{n-k} \times \lfloor N/2^{n-k} \rfloor$ and $N2 = N - 2^{n-k} \times \lfloor N/2^{n-k} \rfloor$. If $N1 \geq 2^{n-k}$, which means the part beyond MB exists, and the number of this part is $\lfloor N/2^{n-k} \rfloor$, it will directly affect the final n-bit output and cause $\lfloor N/2^{n-k} \rfloor$ *ulp* errors. On the other hand, N2 (the part in the range of MB) can lead to at most one *ulp* error when the sum of MBs and N2 would generate a carry to affect the LSB of the output. Therefore, the possible errors generated in the final n-bit output are in the following two situations: 1) $\lfloor N/2^{n-k} \rfloor$ *ulp*, when only $N1$ affects the LSB of the final n-bit output and causing errors; or 2) $\lceil N/2^{n-k} \rceil ulp$, when both $N1$ and $N2$ affect the LSB of output and contribute to the final errors.

For situation 1, it requires that the sum of MBs addition should be less than $2^{n-k} - N2$, and the probability of $\lfloor N/2^{n-k} \rfloor$ *ulp* error can be calculated as:

$$\begin{aligned} P_{\lfloor \frac{N}{2^{N-k}} \rfloor} &= \frac{2^{n-k} - N2}{2^{n-k}} \times P_{(k,N,L)} \\ &= (1 - \frac{N - 2^{n-k} \times \lfloor \frac{N}{2^{n-k}} \rfloor}{2^{n-k}}) \times P_{(k,N,L)} \end{aligned} \tag{3.5}$$

While for situation 2, because both $N1$ and $N2$ affect the output and contribute to the final errors, the sum of MBs addition must be greater than $2^{n-k} - N2$. And then, the probability of $\lceil N/2^{n-k} \rceil ulp$ error can be calculated as following:

$$\begin{aligned} P_{\lceil \frac{N}{2^{n-k}} \rceil ulp} &= \frac{N2}{2^{n-k}} \times P_{(k,N,L)} \\ &= \frac{N - 2^{n-k} \times \lfloor \frac{N}{2^{n-k}} \rfloor}{2^{n-k}} \times P_{(k,N,L)} \end{aligned} \tag{3.6}$$

In Equations (3.5) and (3.6), $P_{(k,N,L)}$ can be calculated according to Equations (3.2) and (3.3). And then the expected error of L inputs adder tree with k-bit truncation can be expressed as following:

$$E_{k-bit} = \sum_{N=1}^{L-1} \{ \lceil \frac{N}{2^{n-k}} \rceil P_{\lceil \frac{N}{2^{n-k}} \rceil} + \lfloor \frac{N}{2^{n-k}} \rfloor \times P_{\lfloor \frac{N}{2^{n-k}} \rfloor} \} \tag{3.7}$$

Therefore, for an L-tap n-bit input coefficients and output FIR filter, with the required accuracy (E_{k-bit}) *ulp*, the greatest k can be determined by Equation (3.7), which can be used for power consumption and circuit area minimization to improve the efficiency of the circuit.

3.4 Evaluations of Uniformly Distributed Random Inputs

To check the correctness of the static error analysis for the proposed unit-level optimization method by the proposed FTA-based adder tree for FIR filter designs, the mathematic calculation results and the exhaustive simulation results with 100 thousand uniformly distributed inputs for a 6-tap 12-bit FIR filter are presented and compared in Fig. 3.6, where the truncated bits (k) are in the range of 8 to 12. The simulation results are in good agreement with the proposed mathematic analysis. Moreover, it can be observed from Fig. 3.6 that, if the requirement of accuracy loss is set to be less than one *ulp*, mostly, 10 bits can be truncated. Therefore, the optimal bit-width of the adder tree blocks can be determined theoretically with the proposed static error analysis method. And, a significant reduction in the power consumption and saving in the circuit area can be achieved by applying the proposed FTAs.

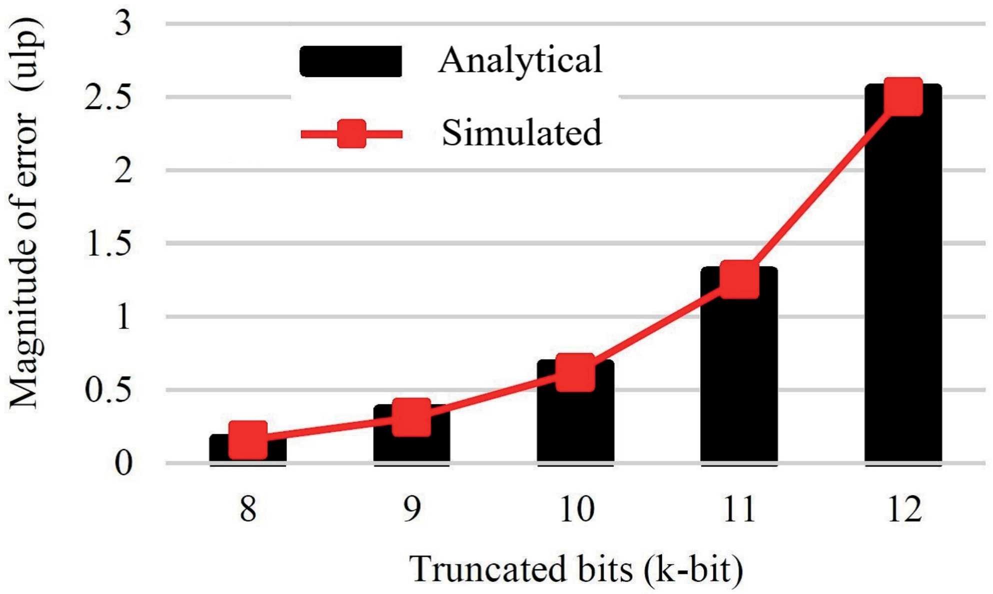


Figure 3.6: Simulation vs. analysis for uniformly distributed random inputs in a 6-tap 12-bit FIR filter (Copyright (c) 2020 IEICE, [34] Fig. 6).

Table 3.1: Implementation results of various FIR filter designs (Copyright (c) 2020 IEICE, [34] Table 1)

FIR Filter	Tap (L)	n (bits)	Design Methods	Delay (ns)	Area (um^2)	Area Reduction	Power (mw)	Power Reduction
MIRZAEI10_06	6	16	Normal adder	14.66	21634		0.85	
[45]			Proposed ($k = 15$)	11.81	15528	28.2%	0.75	11.8%
MIRZAEI10_13	13	16	Normal adder	14.61	34536		1.46	
[45]			Proposed ($k = 14$)	12.56	22302	35.4%	1.09	25.3%
MIRZAEI10_28	28	16	Normal adder	14.20	153358		7.66	
[45]			Proposed ($k = 13$)	13.53	110267	28.1%	5.70	25.6%
MIRZAEI10_61	61	16	Normal adder	15.63	336353		15.39	
[45]			Proposed ($k = 12$)	14.11	241959	28.1%	12.11	21.3%
MIRZAEI10_119	119	16	Normal adder	16.58	637217		27.68	
[45]			Proposed ($k = 11$)	15.81	440117	30.9%	19.94	27.9%
MASKELL07_A108	108	9	Normal adder	12.98	296165		15.72	
[46]			Proposed ($k = 4$)	12.16	246304	16.8%	13.67	13.0%
DAMPS_279	279	12	Normal adder	16.22	974163		44.57	
[49]			Proposed ($k = 6$)	15.55	765912	21.4%	36.59	17.9%

*: k is determined when $E_{k-bit} <= 1ulp$ in all the FIR filter designs

To further evaluate the proposed method with hardware implementation, seven fixed STF FIR filters in [45]-[46], [49] with a various number of taps (in the range of 6 to 279) and bit-widths (9-bit, 12-bit and 16-bit) are implemented in this work by utilizing the logic synthesis tool Design Complier (Synopsys) with the library of Rohm 180nm. By utilizing this large set of FIR implementations, a fair comparison can be performed for the effectiveness evaluation of the proposed unit-level optimization method. All the FIR designs are implemented by utilizing the FTA-based design as shown in Fig. 3.2, where the CSD multiplier block proposed in [37] is applied for area reduction of multipliers.

Table 3.1 shows the evaluation results of seven fixed STF FIR filters implemented with the normal adder unit and the truncated adder unit in terms of delay, area and power consumption. In the proposed method, the maximum number of the truncated bits (k) is calculated by Equation (3.7) to make sure that E_{k-bit} is less than 1 ulp

to guarantee the accuracy requirement. It can be observed from Table 3.1 that the circuit area can be saved up to 35.4% and power consumption can be reduced up to 27.9% by applying the proposed optimal FTA designs.

Moreover, the corresponding PDP, ADP reductions and the ratio of truncated bits in the adder tree are shown in Fig. 3.7. The first five designs are all in 16-bit, but with increasing tap numbers. From Fig. 3.7, it can be observed that the ADP reduction matches the corresponding bit-width reduction. The bit-widths that can be truncated by the proposed method is decreased with the increasing of tap numbers, while still more than 34.4% bit-widths in the adder trees can be reduced. It is clearly shown that the proposed FTAs can achieve significant power and area reduction while meeting the specified requirement of accuracy.

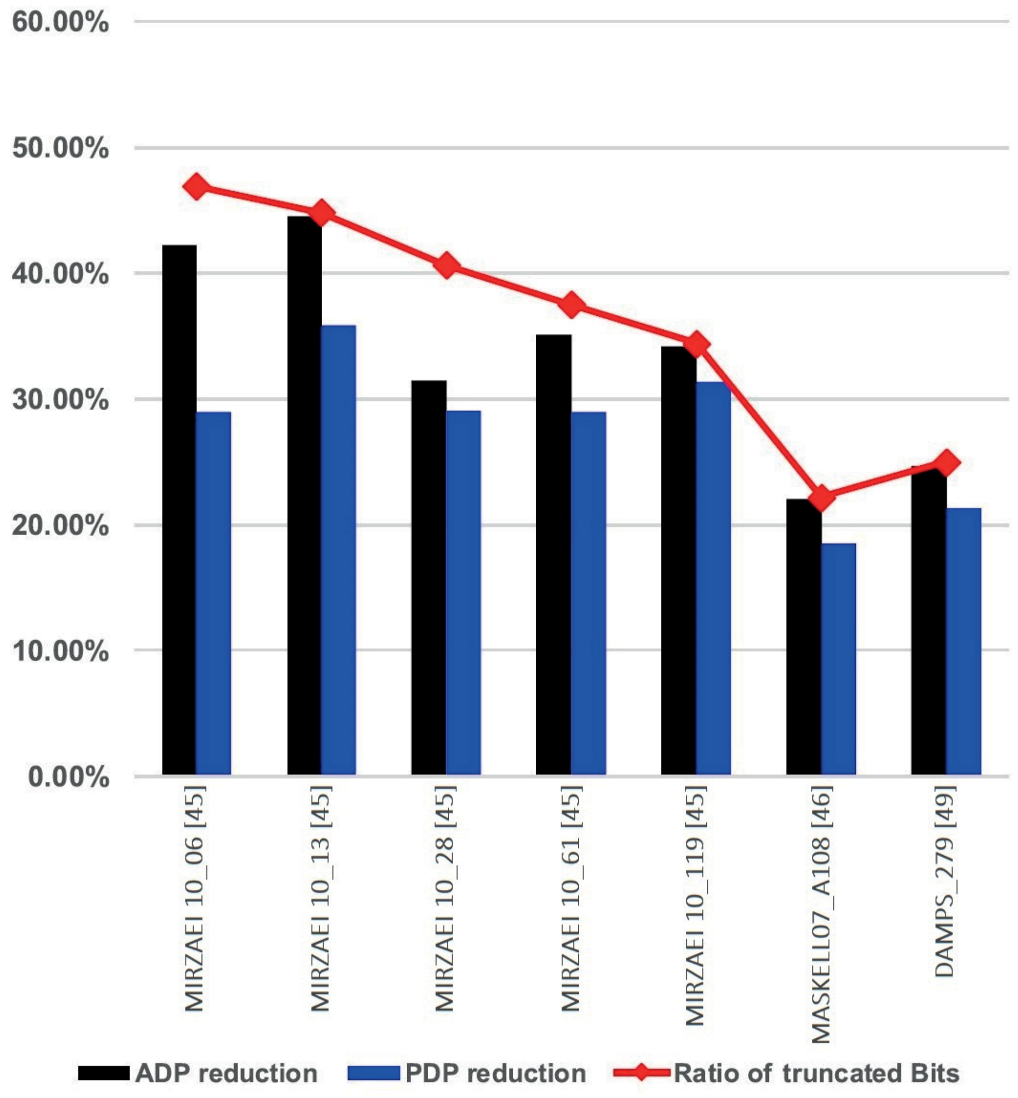


Figure 3.7: Results on ADP and PDP reductions with the ratio of truncated bits in the proposed FTA-based unit-level optimization method (Copyright (c) 2020 IEICE, [34] Fig. 7).

3.5 Evaluations of Normally Distributed ECG Signals

To evaluate the effectiveness of the proposed FTA-based unit-level optimization method with normally distributed inputs, a fixed 6-tap FIR filter for removing the power-line noise in the electrocardiogram (ECG) signal is implemented. The multiplier unit in this FIR filter utilizes the CSD method proposed in [37] and the adder unit is designed with the proposed FTA-based method. The FIR filter is implemented by utilizing the logic synthesis tool Design Complier (Synopsys) with the library of Rohm 180nm. The corresponding input signal is a 10 second raw ECG data (f_s=360 Hz, 12-bit) from file '100.dat' of MIT-BIH arrhythmia database [51] contaminated with synthetic noise (i.e., 50 Hz power-line noise).

3.5.1 Implementation and Evaluation Results

Firstly, in Fig. 3.8, the amplitude-frequency response of the proposed FTA-based FIR filter with k=10 is shown as an example and the corresponding standard results from MATLAB is also shown for comparison. It can be observed that the power-line noise in 50 Hz is successfully removed, and the FIR filter with the proposed design can achieve nearly perfect accuracy even with 10 bits truncated in every adder for normally distributed ECG signals. It should be noted that the number of truncated bits (k) is equal to 10 according to Equation (3.7) in the proposed method while the MATLAB result is obtained after performing 32-bit floating point calculations.

The output waveforms generated by the FIR filter design utilizing the proposed FTA tree with a various number of truncated bits (in the range of 8 to 12) are shown in Fig. 3.9, in which seven waveforms with different heights are presented for visibility improvement. In Fig. 3.9, the standard waveform indicates the result that is obtained from MATLAB with 32-bit floating-point calculations, and the normal adder waveform indicates the result with normal adder tree (i.e., no truncated bits ($k = 0$) in Fig. 3.9), respectively. From the figure, it can be observed that the standard result obtained from MATLAB provides the smoothest curve by eliminating the 50

Hz power-line noise and 32-bit floating-point calculations. Compared with the results of the proposed FTA tree-based FIR filters, there is no obvious difference between the standard implementation with a normal adder tree and the proposed FTA tree-based design.

To check the accuracy loss due to the errors generated by utilizing the truncated adder tree, Maximal Absolute Error ($\max(E)$) and Mean Absolute Error (MAE ($\bar{E}$)) are applied for quantitative error evaluations in the following.

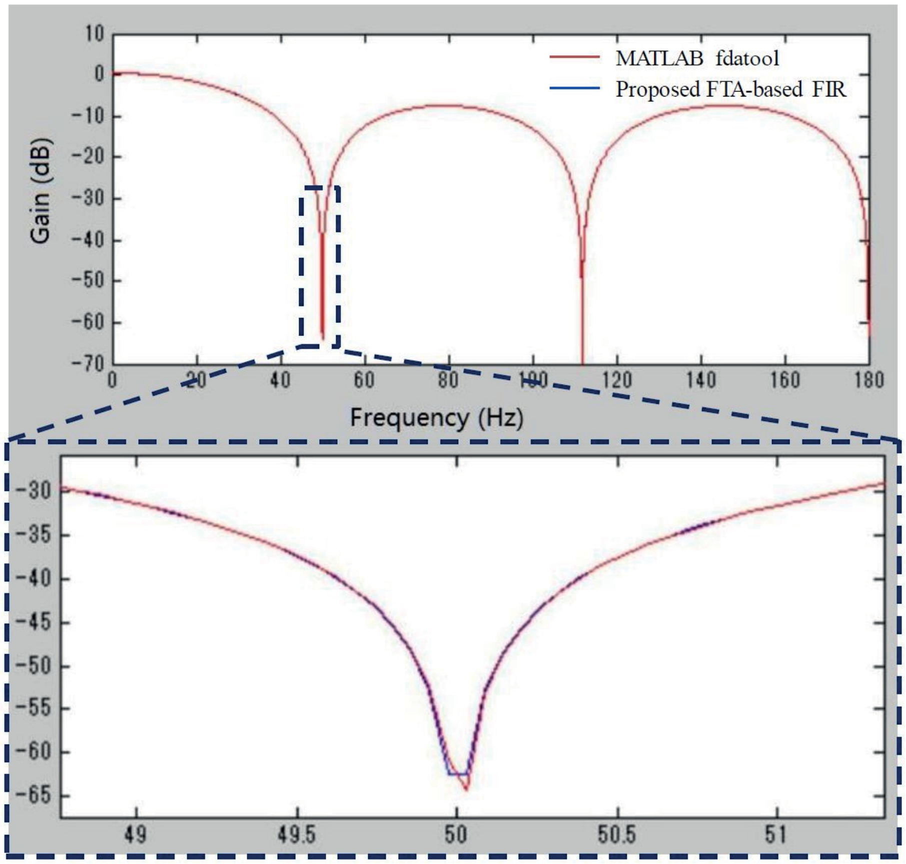


Figure 3.8: Evaluation of the amplitude-frequency response of the proposed FTA tree-based FIR filter design and MATLAB results for normally distributed ECG signals, where in the proposed method, the number of truncated bits (k) is equal to 10; and the result of MATLAB is obtained after performing 32-bit floating point calculations (Copyright (c) 2020 IEICE, [34] Fig. 8).

$$\bar{E} = \frac{1}{S}\sum_{t=1}^{S}(Z(t) - Y(t)) \tag{3.8}$$

and

$$max(E) = max(Z(t) - Y(t)) \tag{3.9}$$

in which, $Y(t)$ and $Z(t)$ indicate the FTA-based designs with various truncated bits (k) and standard signal-based design, respectively. S is the total number of input and output ECG signals which is equal to 3600 in this experiment. The results of error evaluation of FTA-based designs with various truncated bits are shown in Fig. 3.10. According to this figure, the Maximal Absolute Error ($max(E)$) is one *ulp* for k=8 and

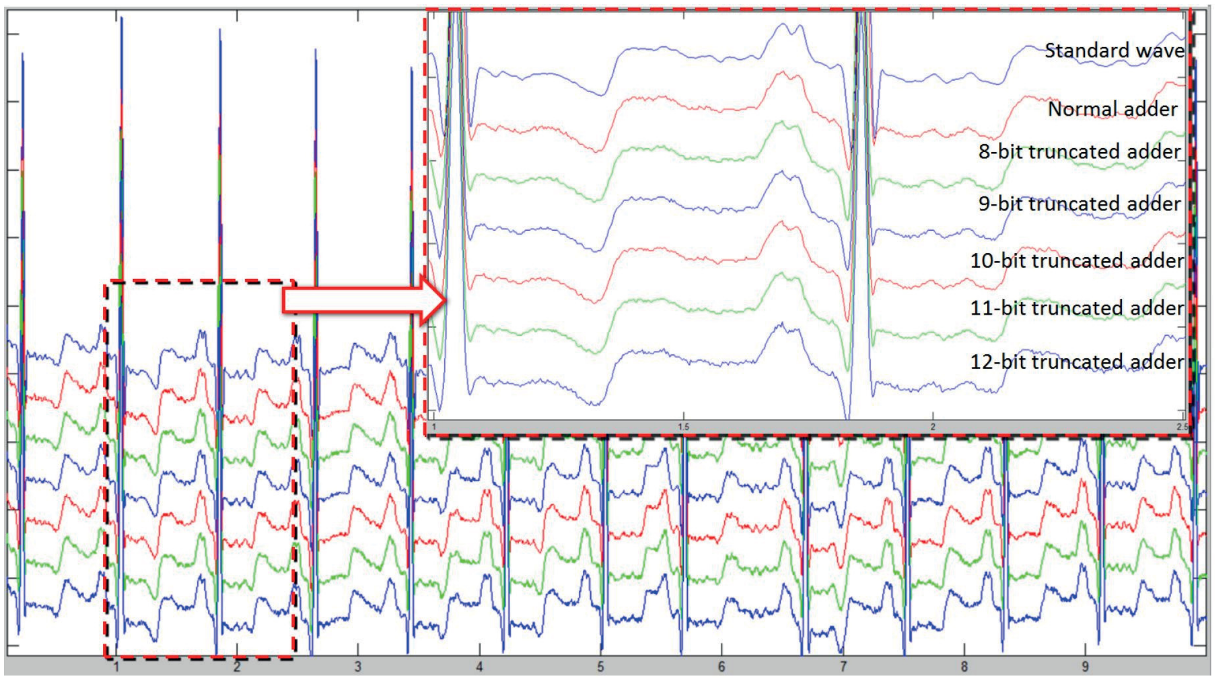


Figure 3.9: The output ECG waveforms of 6-tap FIR filters with various truncated adders (Copyright (c) 2020 IEICE, [34] Fig. 9).

Table 3.2: Implementation results of the 6-tap FIR filters with various truncated bits and the existing optimized GeAr (16,8,0) [43]-based FIR filter for 10 second normally distributed ECG signals (Copyright (c) 2020 IEICE, [34] Table 2).

FIRs with FTA/GeAr	Delay (ns)	Area (um^2)	Power (mw)	σ (ulp)
FTA ($k = 12$)	8.32	13608	0.93	0.70
FTA ($k = 11$)	8.38	14253	1.00	0.42
FTA ($k = 10$)	8.65	14925	1.01	0.32
FTA ($k = 9$)	8.94	15580	1.19	0.29
FTA ($k = 8$)	9.32	16263	1.25	0.29
FTA ($k = 0$*)	10.15	21112	1.36	0.29
GeAr(16,8,0) [43]	8.24	16347	1.32	9.81

*: FIR filter design with normal adders

9 FTA-based FIRs, and when k is less than 10 bit, the $MAE(\bar{E})$ is less than one ulp which completely matches the mathematical analysis presented in Subsection 3.3.2 and shows that the proposed method can be effectively applied to both the normally distributed ECG signals and the uniformly distributed random inputs. Moreover, the error occurrence results in each FTA-based design with different k are provided in Fig. 3.11, which further confirms the results of $max(E)$ and $\bar{E}$ as shown in Fig. 3.10.

In addition, to numerical analysis applying $\bar{E}$ and $max(E)$, σ is introduced to evaluate the error variation around $MAE(\bar{E})$, which is calculated as:

$$\sigma = \sqrt{\frac{1}{S}\sum_{t=1}^{S}(X(t) - Y(t) - \bar{E})^2} \tag{3.10}$$

in which, $Y(t)$ and $X(t)$ indicate the results of the FTA tree-based FIR filter designs with various truncated bits (k) and the result obtained from MATLAB with 32-bit floating point calculations, respectively. As mentioned before, S is the total number of ECG signals in this experiment and is equal to 3600. In this work, σ is applied to describe the error variation around $\bar{E}$. As shown in Fig. 3.12, when compared with the result generated by MATLAB with 32-bit floating point calculations, error variations exist even in the normal adder-based design ($k = 0$). Considering the proposed FTA-based FIR filter designs, when k is small (i.e., $k <= 10$), compared with the normal adder-based design ($k = 0$) the error variation is very small. This can also be observed from the waveforms shown in Fig. 3.9, in which the difference between the corresponding waveforms is little. With the growth of the truncated bits (i.e., k is increasing), the error variation grows significantly larger, and this is also proved in Fig. 3.11 where the error range increases significantly with the increased k.

The corresponding implementation results of the proposed FTA tree-based FIR filter designs with truncated bits (k) in the range of 8 to 12 are shown in Table 3.2, which shows that significant circuit area saving and power consumption reduction can be achieved by the proposed method. As an example, when $k = 10$, up to 29.3% area saving and 25.7% power reduction can be achieved while guaranteeing the output accuracy (i.e. $\bar{E}$ is smaller than 1 ulp) for processing the normally distributed ECG signals.

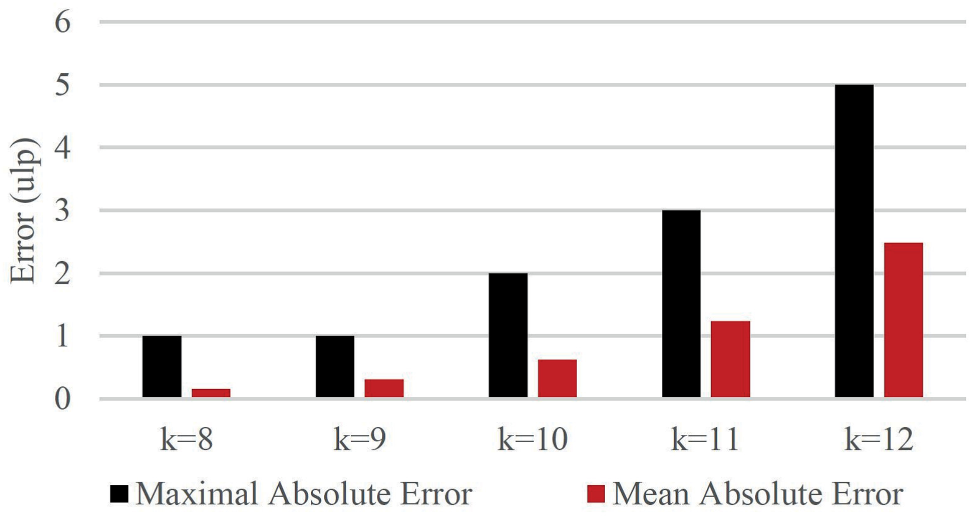


Figure 3.10: Error evaluation results of ECG signals with various truncated adders (Copyright (c) 2020 IEICE, [34] Fig. 10).

3.5.2 Comparisons of Approximate-Computing-Based Methods

GeAr was proposed in [52] as a generic accuracy configurable adder design method, which can be expressed in the form of GeAr(n,R,P) where n is the number of input bits, R is the number of adoption bits and P is the number of bits that is used to predict the internal carry. As a design model, GeAr is effective in design-space exploration for power/area-efficient adder designs, with a low error rate, however the error magnitude might be very large. And then an optimized GeAr design is proposed in [43]. According to [43], considering power consumption, circuit area and error, the GeAr(16,8,0) adder tree-based FIR filter design has the best performance. Therefore, the GeAr(16,8,0) adder has been implemented in the previous 6-tap FIR filter for further evaluation, and the result is also shown in Table 3.2. Moreover, the

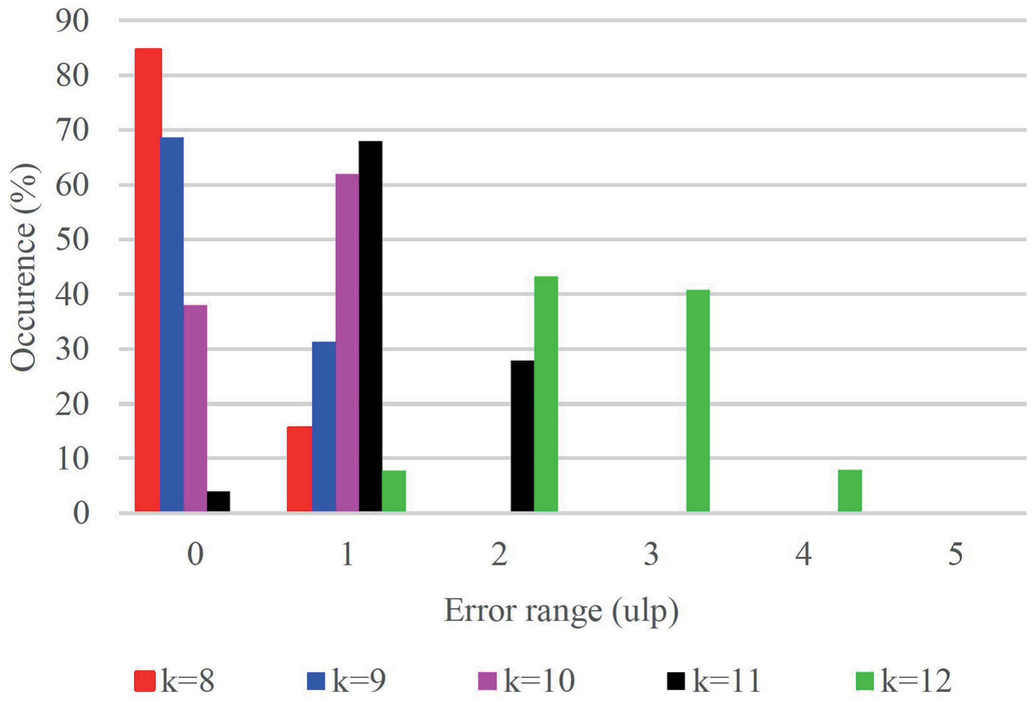


Figure 3.11: Error occurrence in various FTA-based FIR filters (Copyright (c) 2020 IEICE, [34] Fig. 11).

comparison of the output waveform is shown in Fig. 3.13.

As shown in Table 3.2, compared with the GeAr(16,8,0) [43]-based FIR filter, the proposed FDA tree-based FIR filter (k=10) can save 8.7% of circuit area and reduce 19.7% of PDP. Moreover, the proposed method can achieve a 30.7X improvement

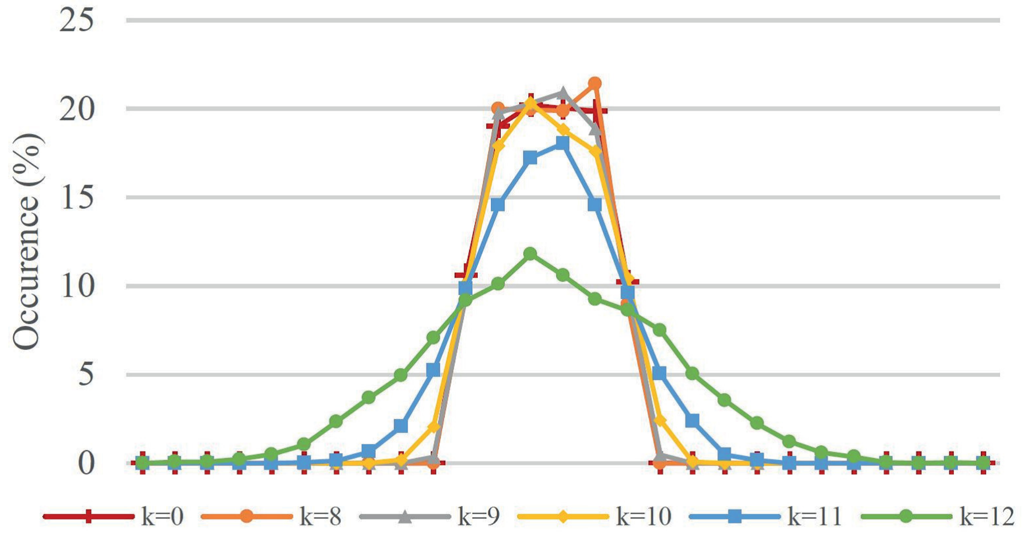


Figure 3.12: Error variations in various FTA-based FIR filters (Copyright (c) 2020 IEICE, [34] Fig. 12).

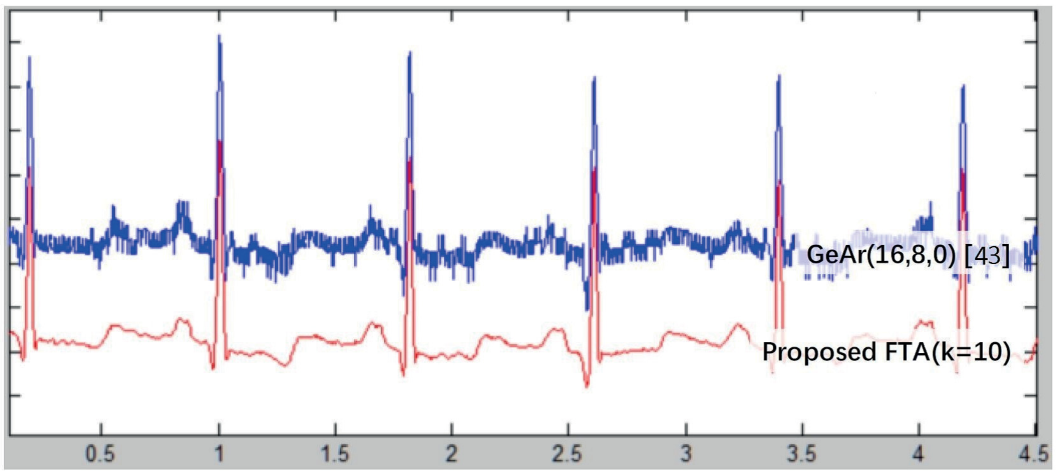


Figure 3.13: Waveform comparison (Copyright (c) 2020 IEICE, [34] Fig. 13).

of σ, which clearly shows the improvement of accuracy over the design proposed in [43]. It can also be confirmed according to the output waveforms shown in Fig. 3.13. Compared with the GeAr-based FIR filter, a more smooth output waveform can be generated by the proposed FTA-based FIR filter, which clearly shows the improvement of the proposed method.

3.6 Chapter Conclusion

This chapter proposed a faithfully truncated adder-based unit-level optimization method for area/power-efficient FIR filter implementation with the predefined output accuracy.

After discussing the existing unit-level optimization methods, a generic methodology for error probability analysis of the truncated adder was presented. Then, based on the error probability, how to determine the width of truncated adder blocks for various FIR filters for area/power-efficient designs while meeting the accuracy loss constraint was presented. Evaluation results showed that a significant reduction in area and power can be achieved with the proposed FTA-based FIR design. As a case study, a fixed 6-tap FIR filter was implemented for electrocardiogram (ECG) signal filtering, and both the numerical analysis and the hardware implementation results showed the effectiveness of the proposed method. The proposed method can effectively support both high-performance and low-cost applications with FIR filters that does not require high accuracy.

Chapter 4

Bit-Level Optimization

An adder-segmentation-based bit-level optimization method is proposed in this chapter for high-performance and high-efficiency FIR filter designs.

This chapter is organized as follows. Section 4.1 introduces the necessity of bit-level optimization and the existing research on bit-level optimization. Section 4.2 and 4.3 present the proposed bit-level optimization method. And then Section 4.4 evaluates the proposed method. Finally, Section 4.5 summarizes the contents of this chapter.

4.1 Existing Research and Its Problem

As mentioned in the previous chapter, to meet the demand for high-quality signal processing, the bit-width of digital signals is getting larger and larger, which requires long bit-width registers, multipliers and adders for FIR filter designs. It means the delay, circuit area and power consumption of these FIR filters will be very large.

There are two directions for solving those problems. First, the direction of speed up the computation of FIR filter, such as the methods of parallel processing [26], [37], [47]. Second, the direction of area/power-efficiency improvement by reducing the circuit area and power consumption such as the existing and the proposed unit-level optimization methods presented in Chapter 3. However, the hardware complexity of the paralleled methods is also very high and the unit-level optimization methods are focused on optimizing the multiplier block or adder tree block but not both of them. Moreover, the FTA tree-based FIR filter design proposed in the previous chapter has accuracy loss, which makes it not suitable for systems with high precision requirements.

In order to optimize both multiplier block and adder tree while without accuracy loss to support the high-performance and high-efficiency applications, a bit-level optimization method is proposed in [41] for FIR filter designs by inserting a register in the middle position of every adder in the adder tree and the last adder in every multiplier to shorting the critical data path for performance improvement. However, the method proposed in [41] can not effectively support the common sub-expression elimination (CSE) [39], [49] or CSD-based MCM designs. Therefore, this chapter proposes an adder-segmentation-based bit-level optimization method for improving the performance and area/power-efficiency to support FIR filters with various MCM designs.

4.2 Basic Idea of Adder-Segmentation-Based Bit-Level Optimization

The STF FIR filter with CSD pattern multipliers as an area/power-efficient FIR implementation is shown in Fig. 4.1 and can be divided into four parts that are pattern block (PB), bit shifter (BS), the addition of multiplier block (AMB) and adder tree block (ATB) from the input side to the output side. The operations of these blocks are shown in Fig. 4.2, in which PB calculates three CSD patterns, AMB finishes the multiplication of FIR coefficients by shifting and adding the CSD patterns generated by PB, and ATB adds the results of AMB together to get the final output of an FIR filter. The circuit complexity is mainly generated by the PB, AMB and ATB because the BS only contains wire connections. According to the CSD pattern-based methods [37], the results of PB are stored in the registers after. Therefore, the delay of these CSD pattern-based FIR filter implementations is determined by the computing complexity of AMB and ATB blocks. As explained previously, the addition of redundant bits causes a limited effect on the accuracy of the output, while leading to a large circuit area and power consumption overhead and long delay. The FTA tree-based unit-level optimization method proposed in Chapter 3 is a promising way for delay/area/power reduction. To avoid the accuracy loss generated by the FTA tree-based designs and to optimize both the multiplier block and the adder tree, this chapter proposes a simple but effective method for delay reduction through segmenting all adders in AMB and ATB as shown in Fig. 4.3. In the proposed design, the adder tree after bit shifter (BS) is divided into two blocks, such as the high bit addition block (HB block) and the low bit addition block (LB block). The HB block and the LB block can complete their additions independently, and then the results generated by the HB addition can be added with the carries presented by the LB block to get the final n-bit output.

Therefore, no accuracy loss will be caused in the proposed design based on adder-segmentation. Moreover, it should be mentioned here that, if some accuracy loss is allowed, the proposed method can also support the truncated adder-based design for further power consumption reduction through LB block gating.

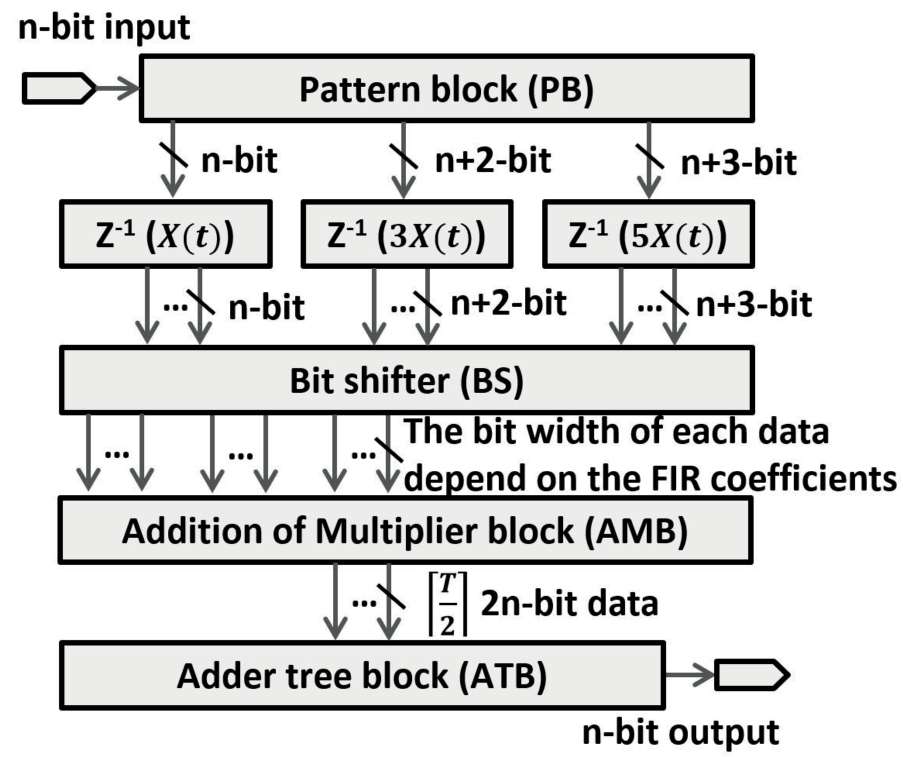
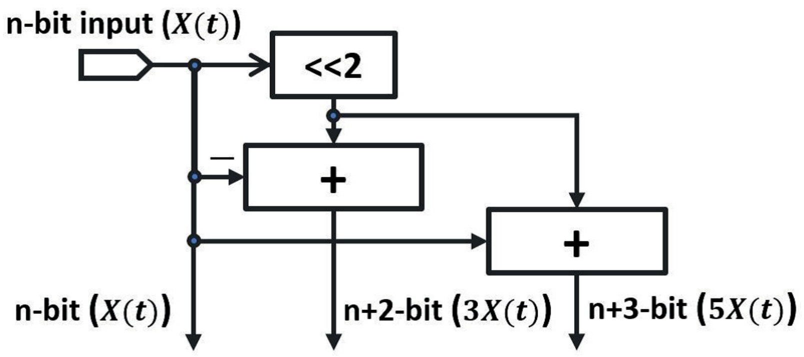
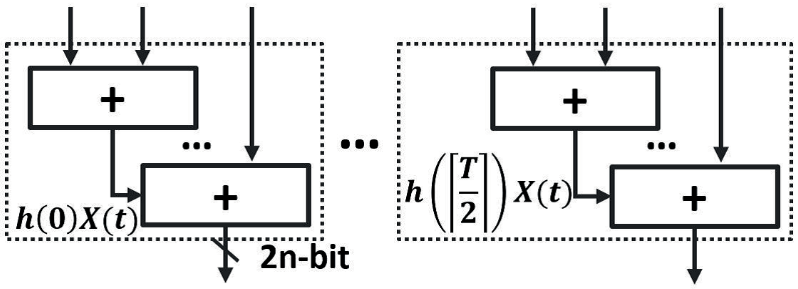


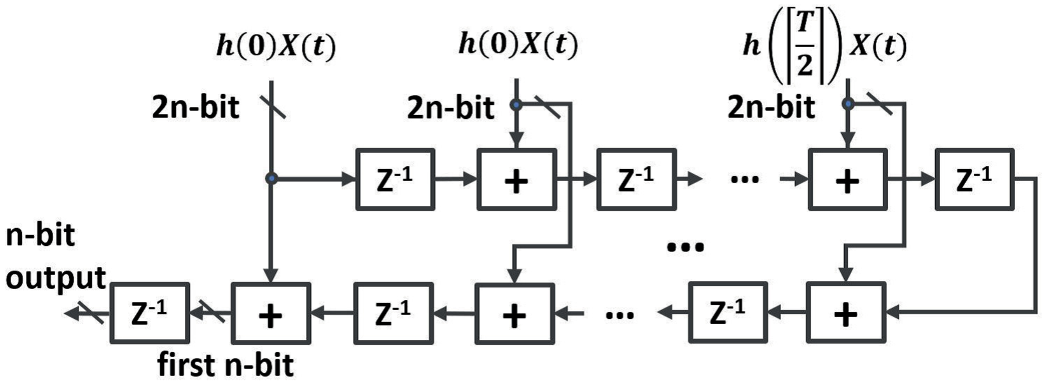
Figure 4.1: CSD-based STF FIR filter.



(a)



(b)



(c)

Figure 4.2: Operation of each block FIR filter design in Fig. 4.1, (a) PB, (b) AMB and (c) ATB.

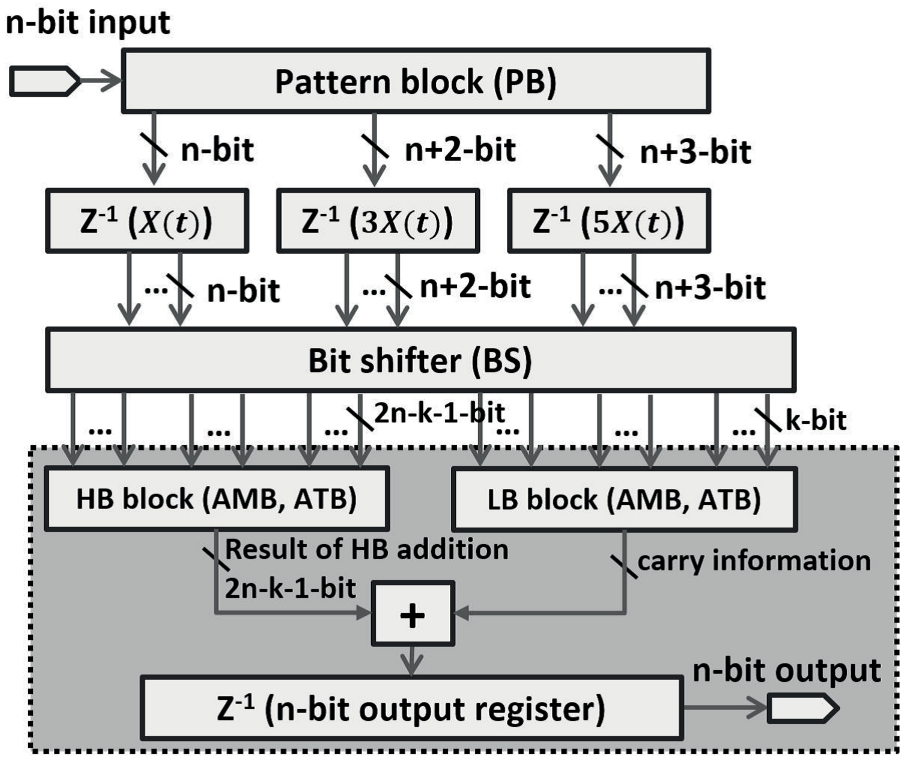


Figure 4.3: FIR filter with the proposed adder-segmentation-based bit-level optimization method.

4.3 Adder Segmentation Optimization

To balance the delay distribution of the proposed adder-segmentation-based filter designs, determining the bit-widths of the HB block and the LB block is necessary. Therefore, an optimization method for adder segmentation is proposed in this section.

Fig. 4.4 introduces the operations and the corresponding circuits of one tap in a CSD-based STF FIR filter. Fig. 4.4 (a) is the existing method with the operations on the left side and the corresponding circuit on the right side. Fig 4.4 (b) shows the proposed method with the operations on the top and the corresponding circuit at the bottom. Assuming the bit-width of the LB block is k, an optimal k determination method is proposed to balance the delay between the HB block and the LB block for maximizing the performance of the proposed method.

The CSD-based STF FIR filter design is considered here because the CSD pattern multiplier is a start-of-the-art technology for efficient MCM designs in a fixed FIR filter implementation. According to [35]-[37], the CSD code has no continuous non-zero bit. Therefore, the most complicated n-bit coefficient coded by CSD-based methods will be "±00±00±00..." or "±0±0±0±0...", in which "±" means plus or minus input data ($X(t)$) after shifting corresponding bits, and "±0±" means $(\pm X(t) << 2) \pm X(t)$ (i.e., $\pm 3X(t)$ or $\pm 5X(t)$). All the CSD codes can be expressed with $\pm$ and the CSD patterns $3X(t)$ or $5X(t)$ [36]-[37]. The multiplication of a n-bit FIR coefficient is performs as "$\pm(X(t) << (n-1)) \pm (X(t) << (n-4)) \pm (X(t) << (n-7))$..." or "$\pm(3X(t), 5X(t) << (n-3)) \pm (3X(t), 5X(t) << (n-7)) \pm (3X(t), 5X(t) << (n-11))$...". Therefore, due to these characteristics of the CSD pattern-based multiplier block, at most, $\lceil n/3 \rceil - 1$ adders in AMB are necessary for a single FIR coefficient.

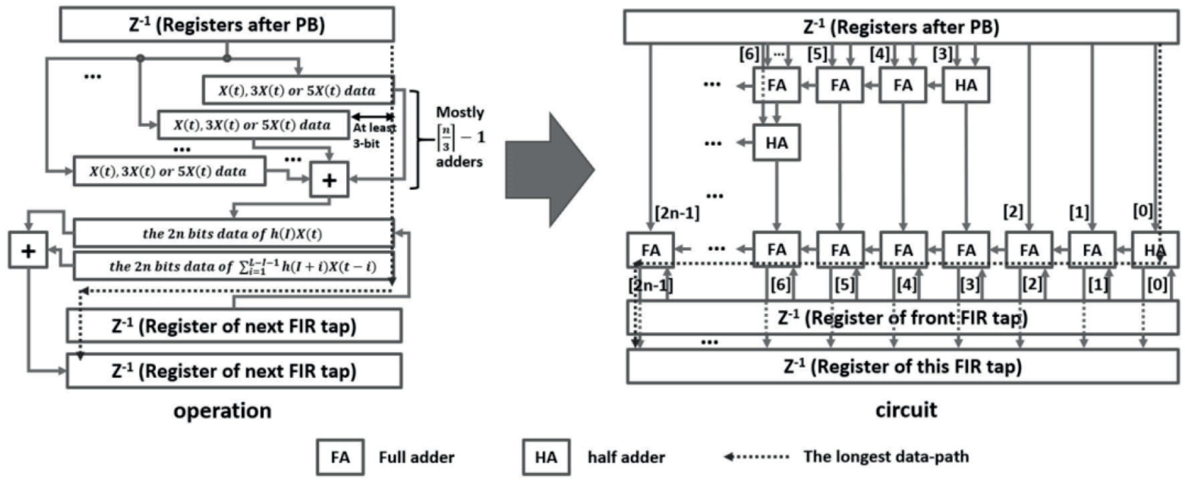
On the other hand, since the above operations in the AMB are divided into the HB block and the LB block by segmentation, the adders in the HB block part are in $2n-k$ bits. On the other hand, due to the generated carriers of the adders in the LB block, the largest bit-width of the adders in the LB block part should be in $k + \lceil log_2 \lceil n/3 \rceil \rceil$ bits. There are $L-1$ adders in the ATB of L-tap FIR filters. The addition in the HB block does not generate any carries due to the properties of linear FIR filters. Therefore, the adders in the HB block are all in $(2n - k)$ bits, no matter in ATB or AMB. And, the inputs of the LB blocks in the ATB are all coded by $(k + \lceil log_2 \lceil n/3 \rceil \rceil)$ bits. Due to the carries needing to be correctly generated by the adder tree located in ATB, the adder tree of a L-tap FIR in the LB block needs extra $log_2 L$ bits for the generated carries. As a result, the bit-width of the largest adders in the LB block is $(k + \lceil log_2 \lceil n/3 \rceil \rceil + log_2 L)$. Therefore, to balance the delay of the LB block and the HB block for circuit speedup, the bit-width of the adders in the HB block should be equal to the bit-width of the largest adder in the LB block as following:

$$k + \lceil log_2 \lceil n/3 \rceil \rceil + log_2 L = 2n - k$$

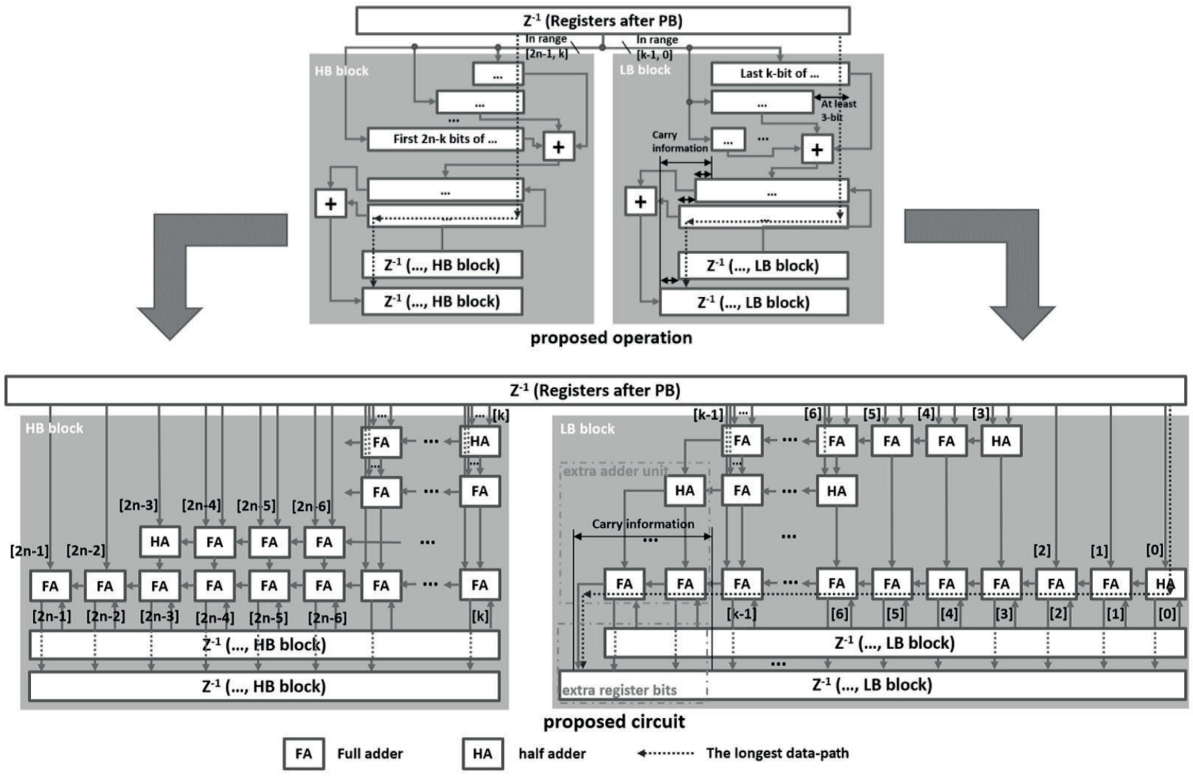
And then, the optimal k should be:

$$k = \frac{2n - \lceil log_2 \lceil n/3 \rceil \rceil - log_2 L}{2} \tag{4.1}$$

Through this equation, the optimal k can be generated.


Z-1 (Registers after PB)
X(t), 3X(t) or 5X(t) data
Mostly ⌈n/3⌉ − 1 adders
At least 3-bit
the 2n bits data of h(l)X(t)
Z-1 (Register of next FIR tap)
operation
circuit
Z-1 (Register of front FIR tap)
Z-1 (Register of this FIR tap)
FA Full adder
HA half adder
The longest data-path

(a)



(b)

Figure 4.4: The operations and the corresponding circuits of one tap in (a) the existing CSD-based FIR filter and (b) the proposed adder-segmentation-based FIR filter (© [2019] IEEE).

4.4 Evaluation

To evaluate the performance, the 6-tap fixed CSD-based STF FIR filter proposed in the previous chapter with 12-bit input output and 12-bit FIR coefficients was implemented with the design proposed in [37] and the proposed adder-segmentation method.

Table 4.1 shows a set of evaluation results on delay, circuit area and power consumption with various LB block bit-widths (k). The baseline design is the CSD-based STF FIR filter proposed in [37] (with $k = 0$ in Table 4.1). According to Equation (4.1), in the proposed method, when $L = 6$ and $n = 12$, the optimal k is 9. However, the implementation results in Table 4.1 show that the fastest design is obtained when $k = 8$. The reason for this difference is that the most significant bit of the data generated by the applied FIR coefficients in the HB is always " 0 ". Therefore, in this specific case, the optimal k is not the same as the result from Equation (4.1). According to the implementation results in Table (4.1), when $k = 8$, the delay is reduced from 9.93 ns to 6.20 ns. By applying the proposed bit-level optimization method of this chapter, up to 39.8% reduction in delay can be achieved with 11.0% additional circuit area.

The proposed bit-level optimization method can be applied to most existing structure-level and unit-level designs for FIR filters. But as the previous analysis shows, the effectiveness of speed-up of the circuits is related to the ratio of the bit-width of the carry information in the LB block to the bit-width of input/output signal and FIR coefficients. When considering the complexity of multiplication, the proposed method is more suitable for designs with start-of-the-art unit-level optimization methods for the multiplication block of an FIR filter such as CSE or CSD [35]-[37], [39], [49] than ordinary MCMs with binary-multiplier-based designs, because of the fewer additions in the AMB. Thus, it can lead to shorter bit-width of carry information. For a similar reason, it should be mentioned that the proposed bit-level optimization method is not suitable for FIR filters with a large number of taps, because the bit-width of the carry information in the LB block increases with the tap number.

Table 4.1: Evaluation results with various k-bits in the proposed adder-segmentation-based FIR filter designs (© [2019] IEEE)

k (bit)	Delay (ns)	Area (um^2)	Power (mw)	EDP ($ns^2 mw$)	EDP Reduction (%)	ADP ($um^2 ns$)	ADP Reduction (%)
0*	9.93	22947	1.45	142.98	-	227864	-
6	6.65	25669	2.70	119.40	16.5	170699	25.1
7	6.42	25553	2.87	114.58	19.9	164050	28.0
8	6.20	25466	2.87	110.32	22.8	157889	30.7
9	6.65	25792	2.67	118.07	17.4	171517	24.7
10	7.17	25375	2.47	126.98	11.2	181939	20.2
11	7.59	25169	2.29	131.92	7.7	191033	16.2
12	8.09	24743	2.11	138.10	3.4	200171	12.2

*The baseline design for k=0 indicates the CSD-based STF FIR proposed in [37]

4.5 Chapter Conclusion

A bit-level optimization method for FIR filter design is proposed in this chapter.

After discussing the necessity and the existing bit-level optimization methods, an adder-segmentation method is proposed to shorten the critical path for performance improvement. And then explains how the proposed adder-segmentation technique can optimize the performance by balancing the length of the critical data path in the LB and the HB block for performance improvement. The evaluation results show the significant improvement can be achieved when compared with the existing FIR filter designs, which clearly shows the efficiency of the proposed method for supporting the high-performance applications with FIR filters.

Chapter 5

Conclusions and Future Research

This chapter makes a conclusion of this dissertation and shows two directions for future research directions on high energy-efficiency and area-efficiency optimization methods for convolutional systems.

5.1 Technical Summary

The main objective of this research is to improve the area/power-efficiency of convolutional system designs. Optimization methods in three different design levels are proposed in this dissertation. To evaluate these three methods, the proposed methods and the existing relevant methods are compared by implementing various FIR filters.

Firstly, a symmetric hybrid form is proposed as the structure-level optimization method for designing an FIR filter to solve the driving ability problem and to improve the area/power-efficiency of existing forms. The evaluation results show that, compared with the existing symmetric systolic form in [32] and the hybrid form in [7], the proposed form can achieve 21% saving in the circuit area, 28% reduction in the power consumption, 10% reduction in ADP and 18% reduction in PDP. Moreover, compared with the symmetric systolic form in [32], the proposed form can achieve a 33% reduction in latency.

Secondly, a faithfully truncated adder tree-based area/power-efficient unit-level optimization method for FIR filter implementations with predefined output accuracy is proposed. Evaluation results show that, compared with the normal adder tree, up to 35% area saving and 30% power reduction can be achieved by the FIR filters with the proposed FTA-based adder tree. As a case study, a fixed 6-tap FIR filter is implemented for electrocardiogram (ECG) signal filtering with the proposed adder tree. Compared with GeAr-based design, 30.7X improvement in error variation can be achieved and a more smooth output waveform can be generated by the proposed method while with 9% saving in the area saving 20% reduction in the PDP.

Finally, an adder segmentation-based bit-level optimization method for FIR filter design is proposed for improving the area/power-efficiency and the performance while without accuracy loss. The evaluation results show that up to 40% reduction in delay, 31% reduction in ADP and 23% reduction in EDP can be achieved when compared with the existing CSD-based FIR filter designs proposed in [37].

5.2 Future Directions

In this dissertation, three optimization methods for convolutional system design in three design levels are proposed for area/power-efficiency improvement, which can be viewed as I) SHF design for structure-level optimization, II) FTA-based design for unit-level optimization, III) adder segmentation-based design for bit-level optimization. The utilizations and the evaluations are based on the FIR filters which are one-dimensional convolutional systems.

An extending case of the optimization methods for FIR filters is to utilize the fast finite impulse response algorithms (FFAs) [53]-[54] for parallel processing with less multiplications. The FFA-based method is one of the well-known parallel computation methods based on transpose form FIR filter, in which the number of multiplications can be reduced for power-efficiency improvement and the computation can be paralleled for performance improvement. For example, a 2-tap, 2-parallel FFA-based FIR filter only has three multipliers, while for every clock cycle two outputs can be generated, in which the computation can be speed-up by nearly two times with 25% reduction of multiplications. Some related works presented in [18]-[20] are utilizing the FFA methods for designing the processing element (PE) blocks as a structure-level optimization method for performance improvement and power reduction in CNN accelerator designs. In these works, 1/3 and 1/2 multiplications can be saved for convolutions with 3×3 and 5×5 kernels, respectively.

Therefore, the first future research direction is to extend the methods proposed in this dissertation to more complex two-dimensional and three-dimensional convolutional systems, such as image processing and convolutional neural network.

On the other hand, when the system grows larger, an extra topper design level called system-level is necessary to be taken into consideration. The system-level design influence such as the dataflow of the target system, which includes off-chip and on-chip memory communication. It is very important when designing a large circuit such as a CNN accelerator [14]-[18]. The reason of do not optimize the system-level in the previous chapters is that when the circuit is not so large or it is only a computational block in the target system, such as an FIR filter design, due to the simple dataflow and infrequent data exchange, system-level optimization is no need to be considered.

For large high-efficiency convolutional systems, such as CNN accelerators, the optimizations in the system-level for circuit module decision, control module design and dataflow optimization are more important than those in the structure-level, the unit-level and the bit-level. The on-chip memory resources are very limited (typically smaller than 500KB) in low-cost CNN accelerators [14]-[17], which means, it is generally impossible to save all the intermediate results or the parameters on-chip even for one layer. An example of accelerating the CNN architecture VGG [8] on the accelerator Eyeriss [16] shows that even with data compression, tens of megabytes are required to be moved in and out of the chip in each layer, which requires much more energy-consuming than the corresponding MAC computation, because the power consumption of communicating n-bit data with off-chip memory is hundreds of times more than that of the n-bit MAC computation [14]. Moreover, depending on [16], the amount of on-chip GLB accesses can reach up to 50X of that of off-chip DRAM accesses, causing large on-chip power consumption and access conflict.

Therefore, another direction of the future work is to develop a dataflow optimization method as a system-level design approach for high area/power-efficiency convolutional system designs.

Bibliography

[1] J. Lin and M. J. T. Smith, "Two-band hybrid FIR-IIR filters for image compression," IEEE Transactions on Image Processing, vol. 20, no. 11, pp. 3063 - 3072, Nov. 2011.

[2] A. Mukhtar, H. Jamal and U. Farooq, "An area efficient interpolation filter for digital audio applications," IEEE Transactions Consumer Electronics, vol. 55, no. 2, pp. 768 - 772, May 2009.

[3] N. Shlyakhov, M. Goncharenko and A. Leonenko, "FIR/IIR cascade approximating H.264/AVC interpolation filter," in. Proc. IEEE International Conference on Image Processing, Oct. 2014, pp. 3156 - 3159.

[4] J. A. V. Alste and T. S. Schilder, "Removal of base-line wander and power-line interference from the ECG by an efficient FIR filter with a reduced number of taps," IEEE Transactions Bioelectric signal processing, vol. 20, no. 11, pp. 1052 - 1060, Dec. 1985.

[5] Y. Lecun, Y. Bengio and G. Hinton, "Deep Learning," Nature, vol. 521, pp. 436 - 444, May 2015.

[6] T. Hentschel and G. Fettweis, "Software radio receivers," in CDMA Techniques for Third Generation Mobile Systems, The Netherlands: Kluwer, 1999, ch. 10, pp. 257-283.

[7] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation, Wiley, New York, 1999.

[8] K. Simonyan, and A. Zisserman, "Very deep convolutional neural networks for large scale image recognition," in Proc. International Conference on Learning Representations, May 2015, pp. 1 - 14.

[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2016, pp. 770 - 778.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2015, pp. 1 - 9.

[11] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K, He, “Aggregated residual transformations for deep neural networks,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Jul, 2017, pp. 1492 - 1500.

[12] G. Huang, Z. Liu, L. Maaten, and K. Weinberger, “Densely connected convolutional networks,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Jul, 2017, pp. 4700 - 4708.

[13] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2018, pp. 4510 - 4520.

[14] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz and W. J. Dally, “EIE: efficient inference engine on compressed deep neural network,” in Proc. IEEE Annual International Symposium on Computer Architecture, Jun. 2016, pp. 243 - 254.

[15] J. Li, G. Yan, W. Lu, S. Jiang, S. Gong, J. Wu and X. Li, “SmartShuttle: Optimizing off-chip memory accesses for deep learning accelerators,” in Proc. Design, Automation and Test in Europe Conference and Exhibition, Mar. 2018, pp. 343 - 348.

[16] Y. Chen, T. Krishna, J. S. Emer and V. Sze, “Eyeriss: An energy efficient reconfigurable accelerator for deep convolutional neural networks,” IEEE Journal of Solid-State Circuits, vol. 52, no. 1, pp. 127 - 138, Jan. 2017.

[17] H. Ahmad, T. Arif, M. A. Hanif, R. Hafiz and M. Shafique, “SuperSlash: A unified design space exploration and model compression methodology for design of deep learning accelerators with reduced off-chip memory access volume,” IEEE Transactions Computer-Aided Design of Integrated Circuits and System, vol. 39, no. 11, pp. 4191 - 4204, Oct. 2020.

[18] J. Wang, J. Lin and Z. Wang, "Efficient convolution architectures for convolutional neural network," in Proc. IEEE International Conference on Wireless Communications & Signal Processing, Oct. 2016, pp. 2472 - 7628.

[19] J. Wang, J. Lin and Z. Wang, "Efficient hardware architectures for deep convolutional neural network," IEEE Transactions Circuits and System I, vol. 65, no. 6, pp. 1941 - 1953, Jun. 2018.

[20] H. Wang, W. Xu, Z. Zhang, X. You and C. Zhang, "An efficient stochastic convolution architecture based on fast FIR algorithm," IEEE Transactions Circuits and System II, pp. 1 - 5, Oct. 2021.

[21] E. Mirchandani, R. L. Zinser and J. B. Evans, "A new adaptive noise cancellation scheme in the presence of crosstalk (speech signals)," IEEE Transactions Circuits and System II, Analog Digit. Signal Processign, vol. 39, no. 10, pp. 681 - 694, Oct. 1995.

[22] D. Xu and J. Chiu, "Design of a high-order FIR digital filtering and variable gain ranging seismic data acquisition system," in Proc. Southeastcon, Apr. 1993, pp. 1 - 6.

[23] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," IEEE Transactions Circuits and System II, vol. 42, no. 9, pp. 569 - 577, Sep. 1995.

[24] M. Potkonjak, M. B. Shrivasta, and P. A. Chandrakasan, "Multiple constant multiplication: Efficient and versatile framework and algorithms for exploring common subexpression elimination," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 15, pp. 151 - 161, Feb. 1996.

[25] X. Lou, Y. J. Yu, and P. K. Meher, "Fine-grained critical path analysis and optimization for area-time efficient realization of multiple constant multiplications," IEEE Transactions on Circuits and Systems I, vol. 62, no. 3, pp. 863 - 872, Mar. 2015.

[26] J. Park, W. Jeong, H. M. Meimand, Y. Wang, H. Choo, and K. Roy, "Computation sharing programmable FIR filter for low-power and high-performance applications," IEEE Journal of Solid-State Circuits, vol. 39, no. 2, pp. 348 - 357, Feb. 2004.

[27] B. K. Mohanty, P. K. Meher, S. Al-Maadeed, and A. Amira, “Memory footprint reduction for power-efficient realization of 2-D finite impulse response filters,” IEEE Transactions on Circuits and Systems I, vol. 61, no. 1, pp. 120 - 133, Jan. 2014.

[28] B. Mohanty and P. Meher, “A High Performance FIR Filter Architecture for Fixed and Reconfigurable Applications,” IEEE Transactions on Very Large Scale Integration Systems, vol. 24, no. 2, pp. 444 - 452, Feb. 2016.

[29] K. Azadet and C.J. Nicole, “Low-power equalizer architectures for high speed modems,” IEEE Communications Magazine, vol. 36, no. 10, pp. 118 - 126, Oct. 1998.

[30] K. Khoo, Z. Yu and A.N. Willson, “Design of optimal hybrid form FIR filter,” in Proc. IEEE International Symposium on Circuits and Systems, May 2001, pp. 621 - 624.

[31] E. Abdel-Raheem, F. Elguibaly, and A. Tawfik, “Systolic implementation of linear-phase FIR filters,” in Proc. Canadian Conference Electronics and Computer Engineering, Nov. 1993, pp. 680 - 683.

[32] H. K. Kwan and T. S. Okullo-Oballa, “Systolic realization of linear-phase FIR digital filters,” IEEE Transactions Circuits and Systems, vol. 34, pp. 1604 - 1605, Dec. 1987.

[33] J. Ye, N. Togawa, M. Yanagisawa and Y. Shi, “Static error analysis and optimization of faithfully truncated adders for area-power efficient FIR designs,” in Proc. IEEE International Symposium on Circuits and Systems, May 2019, pp. 1 - 4.

[34] J. Ye, M. Yanagisawa and Y. Shi, “Faithfully Truncated Adder-based Area-Power Efficient FIR Design with Predefined Output Accuracy,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E103-A, no. 9, pp. 1063 - 1070, Sep. 2020.

[35] N. Goel and A. Nandi, “Design of FIR filter using FCSD representation,” in Proc. IEEE International Conference on Computational Intelligence & Communication Technology, Apr. 2015, pp. 617 - 620.

[36] W. Ding and J. Chen, “Design of low complexity programmable FIR filters using multiplexers array optimization,” in Proc. IEEE International Symposium on Circuits and Systems, Jul. 2015, pp. 2960 - 2963.

[37] J. Ye, Y. Shi, N. Togawa and M. Yanagisawa, "A low cost and high speed CSD-based symmetric transpose block FIR implementation," in Proc. IEEE International Conference on ASIC, Oct. 2017, pp. 311 - 314.

[38] J. Chen and C. Chang, "Novel design algorithm for low complexity programmable FIR filters based on extended double base number system," IEEE Transactions on Circuits and Systems I, vol. 62, no. 1, pp. 224 - 233, Oct. 2015.

[39] X. Lou and W. Ye, "Low complexity and low power multiplierless FIR filter implementation," in Proc. IEEE International Conference on ASIC, Oct. 2017, pp. 612 - 615.

[40] W. Ye, X. Lou and Y. Yu, "Design of low-power multiplierless linear-phase FIR filters," IEEE Access, vol. 5, pp. 23466 - 23472, Aug. 2017.

[41] X. Lou, Y. Yu and P. K. Meher, "Analysis and optimization of product-accumulation section for efficient implementation of FIR filters," IEEE Transactions on Circuits and Systems I, vol. 63, no. 10, pp. 1701 - 1713, Oct. 2016.

[42] J. Ye, M. Yanagisawa, and Y. Shi, "An adder-segmentation-based FIR for high speed signal processing," in Proc. IEEE International Conference on ASIC, Oct. 2019, pp. 1 - 4.

[43] K. Hayamizu, N. Togawa, M. Yanagisawa and Y. Shi, "Extension and performance/accuracy formulation for optimal GeAr-based approximate adder designs," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E101-A, no. 7, pp. 1014 - 1024, Jul. 2018.

[44] S. Hsiao, J. Hong, Z. Jian and M. Chen, "Low-cost FIR filter designs based on faithfully rounded truncated multiple constant multiplication/accumulation," IEEE Transactions on Circuits and Systems II, vol. 60, no. 5, pp. 287 - 291, Apr. 2013.

[45] D. L. Maskell, "Design of efficient multiplierless FIR filters," IET Circuits, Devices & Systems, vol. 1, no. 2, pp. 175 - 180, Apr. 2007.

[46] S. Mirzaei, R. Kastner, and A. Hosangadi, "Layout aware optimization of high speed fixed coefficient FIR filters for FPGAs," International Journal of Reconfigurable Computing, vol. 2010, no. 1, pp. 1 - 17, Jan. 2010.

[47] B. K. Mohanty and P. K. Meher, “A high-performance energy-efficient architecture for FIR adaptive filter based on new distributed arithmetic formulation of block LMS algorithm,” IEEE Transactions Signal Processign, vol. 61, no. 4, pp. 921 - 932, Feb. 2013.

[48] A. Kovalev, O. Gustafsson, and M. Garrido, “Implementation approaches for 512-tap 60 GSa/s chromatic dispersion FIR filters,” in Proc. IEEE Asilomar Conference on Signals, Systems, and Computers, Nov. 2017, pp. 1779 - 1783.

[49] C. H. Chang and M. Faust, “On a new common subexpression elimination algorithm for realizing low-complexity higher order digital filters,” IEEE Transactions Computer-Aided Design of Integrated Circuits and Systems, vol. 29, no. 5, pp. 844 - 848, May 2010.

[50] V. Gupta, D. Mohapatra, A. Raghunathan and K. Roy, “Low-power digital signal processing using approximate adders,” IEEE Transactions Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 1, pp. 124 - 137, Jan. 2013.

[51] G.B. Moody and R.G. Mark, “The impact of the MIT-BIH arrhythmia database,” IEEE Engineering in Medicine and Biology Magazine, vol.20, no.3, pp.45 - 50, May-Jun. 2001.

[52] M. Shafique, W. Ahmad, R. Hafiz and J. Henkel, “A low latency generic accuracy configurable adder,” in Proc. IEEE Design Automation Conference, Jun. 2015, pp.1 - 6.

[53] Y. Tsao and K. Choi, “Area-efficient parallel FIR digital filter structures for symmetric convolutions based on fast FIR algorithm,” IEEE Transactions on Very Large Scale Integration Systems, vol. 20, no. 2, pp. 366 - 371, Feb. 2012.

[54] Y. Tsao and K. Choi, “Area-efficient VLSI implementation for parallel linear-phase FIR digital filters of odd length based on fast FIR algorithm,” IEEE Transactions on Circuits and Systems II, vol. 59, no. 6, pp. 371 - 375, Jun. 2012.

Publication List

論文 (学術誌原著論文)

1. L. Ye, J. Ye, M. Yanagisawa and Y. Shi, "Power-efficient deep convolutional neural network design through zero-gating PEs and partial-sum reuse centric dataflow", IEEE Access, vol. 9, pp. 17411 - 17420, Jan. 2021.
2. ◯ J. Ye, M. Yanagisawa and Y. Shi, "Faithfully truncated adder-based area-power efficient FIR design with predefined output accuracy," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E103-A, no. 9, pp. 1063 - 1070, Sep. 2020.

国際学会 (口頭発表)

1. ◯ J. Ye, M. Yanagisawa and Y. Shi, "A high-performance symmetric hybrid form design for high-order FIR filters," in Proc. IEEE Asia Pacific Conference on Circuits and Systems, Dec. 2020, pp. 121 - 124.
2. ◯ J. Ye, M. Yanagisawa and Y. Shi, "A bit-segmented adder chain based symmetric transpose two-block FIR design for high-speed signal processing," in Proc. IEEE Asia Pacific Conference on Circuits and Systems, Nov. 2019, pp. 29 - 32.
3. ◯ J. Ye, M. Yanagisawa and Y. Shi, "An adder-segmentation-based FIR for high speed signal processing," in Proc. IEEE International Conference on ASIC, Oct. 2019, pp. 1 - 4.

4. ◯ J. Ye, N. Togawa, M. Yanagisawa and Y. Shi, "Static error analysis and optimization of faithfully truncated adders for area-power efficient FIR designs," in Proc. IEEE International Symposium on Circuits and Systems, May 2019, pp. 1 - 4.

業績賞等

1. Best student paper award, IEEE Asia Pacific Conference on Circuits and Systems, 2020.

研究費・助成金

1. 2019 年度 早大理工総研-キオクシア (旧・東芝メモリ) 若手奨励研究
2. 2021 年度 早大理工総研-キオクシア (旧・東芝メモリ) 若手奨励研究