# Resource-Efficient Pipelined Architectures for Radix-2 Real-Valued FFT With Real Datapaths

Xiao-Bo Yin, Feng Yu, *Member, IEEE*, and Zhen-Guo Ma

***Abstract*—This brief presents a new algorithm optimized for radix-2 real-valued fast Fourier transform (RFFT) through rigorous formula derivation. Based on that, a novel two-parallel pipelined radix-2 RFFT architecture is proposed with only real datapaths instead of hybrid datapaths. The architecture takes advantage of saving the arithmetic resource in the time-division multiplexing approach to achieve 100% hardware resource utilization. Thereby, it reduces the required number of complex multipliers from $\log_2 N - 2$ to $(1/2)(\log_2 N - 3)$, in contrast with existing two-parallel pipelined architectures. The experimental result shows that the proposed two-parallel architecture can reduce the slice and power consumption by a factor of 30% compared with a recently published work for a 64-point CFFT. Furthermore, a systematic method is also explicated to generalize the architecture to higher level of parallelism and higher radix.**

***Index Terms*—Parallelism, pipelined architecture, real datapath, real-valued fast Fourier transform (RFFT), resource-efficient.**

## I. Introduction

Fast Fourier transform (FFT) is one of the most widely used algorithms in the field of digital signal processing [1]. Many researches have been made for better computation of FFT [2], which operates over complex numbers and results in complex numbers (CFFT). There has been an increasing interest in the real-valued FFT (RFFT) since many physical signals are real-valued [3]–[5], such as the electrocardiogram and electroencephalograph in biomedical applications. When the input signals are real, the spectrum of the FFT is Hermitian symmetric; therefore, approximately half of the computations are redundant [6].

Specific algorithms have been proposed in many literatures [6], [7] for computation of RFFT. A memory-based architecture for RFFT is proposed by Ayinala [8], achieving less memory but more latency. Nevertheless, no hardware-efficient pipelined architectures for RFFT have been proposed until Garrido [9] presents a four-parallel radix-2 RFFT architecture with real datapaths. In [9], a specific algorithm for RFFT is deduced through a data flow graph to eliminate the redundant calculation. Subsequently, Ayinala [10]–[12], Glittas [13], and Zode [14] have researched on pipelined RFFT computation architectures with hybrid datapaths referred to as a combination of real and complex datapaths. Salehi [15] proposes a two-parallel RFFT architecture with only real datapaths at the first time, which is efficient but does not utilize multipliers and delay registers efficiently.

In this brief, a new algorithm is proposed through formula deduction. This brief also shows that the two-parallel radix-2 RFFT architecture can be implemented using real datapaths with full utilization of hardware resources. The number of complex multiplier (CM), delay registers, and latency is decreased compared with prior works. Another contribution of this brief is that the RFFT architecture of higher parallelism could be achieved from the RFFT and CFFT architecture of lower parallelism easily. The organization of this brief is as follows. Section II details an algorithm through formula derivation optimized for RFFT. The proposed pipelined architecture is presented and extended to higher level of parallelism in Section III. Then, Section IV compares it with recently published architectures and evaluates the architecture through implementation. Finally, the conclusion of this brief is drawn in Section V.

## II. Real-Valued Fourier Transform

Based on the Cooley–Tukey algorithm, this section presents a specific algorithm for RFFT through rigorous formula derivation, contrary to previous approaches [9], [15] which derive the algorithm through a data flow graph.

The $N$-point discrete Fourier transform [6] is defined as follows: $X(k) = \sum_{m=0}^{N-1} x(m) W_N^{mk}, k = 0, 1, \ldots, N-1$, where $x(m)$ is the input signal data, $N$ is a power of two ($N = 2^n$, $n$ is an integer), and $W_N^{mk}$ is the twiddle factor ($W_N = e^{-2j\pi/N}$). The result data set of size $N$ could be divided into $n$ subsets as shown in the following:

$$S = \{X(m) : m = 0, 1, \ldots, N-1\} = \bigcup_{i=0}^{n-1} S_i$$

where $S_0 = \{X(0), X(N/2)\}$ $S_t = \{X((2k+1) \cdot 2^{n-t-1}) : k = 0, 1, \ldots, 2^t - 1\}, t = 1, 2, \ldots, n-1$.

The RFFT considers the input data $x(m)$ as a real sequence. Hence, $X(k)$ is conjugate symmetric: $X(N-k) = X^*(k)$.

Therefore, it is unnecessary to calculate all of the FFT coefficients, and the computations of the redundant $(N/2-1)$ conjugate-symmetric samples could be eliminated. Let $\tilde{S}$ denote the RFFT result data set, and it is trivial to prove that $\tilde{S} = \bigcup_{i=0}^{n-1} \tilde{S}_i$, where $\tilde{S}_0 = S_0 = \{X(0), X(N/2)\}$; $\tilde{S}_t = \{X((4k+1) \cdot 2^{n-t-1}) : k = 0, 1, \ldots, 2^{t-1} - 1\}, t = 1, 2, \ldots, n-1$. In set $\tilde{S}_0$, $X(0) = \sum_{m=0}^{N-1} x(m)$ $X(N/2) = \sum_{m=0}^{N/2-1} x(2m) - \sum_{m=0}^{N/2-1} x(2m+1)$. Hence, $X(0)$ and $X(N/2)$ can be calculated by $n$ stages of butterflies. When $t = 1, 2, \ldots, n-1$

Manuscript received October 3, 2015; revised November 18, 2015; accepted February 8, 2016. Date of publication February 18, 2016; date of current version July 28, 2016. This brief was recommended by Associate Editor G. Masera.

The authors are with the Department of Instrument Science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: xiaobo@zju.edu.cn; osfengyu@zju.edu.cn; 850501@zju.edu.cn).

Digital Object Identifier 10.1109/TCSII.2016.2530862

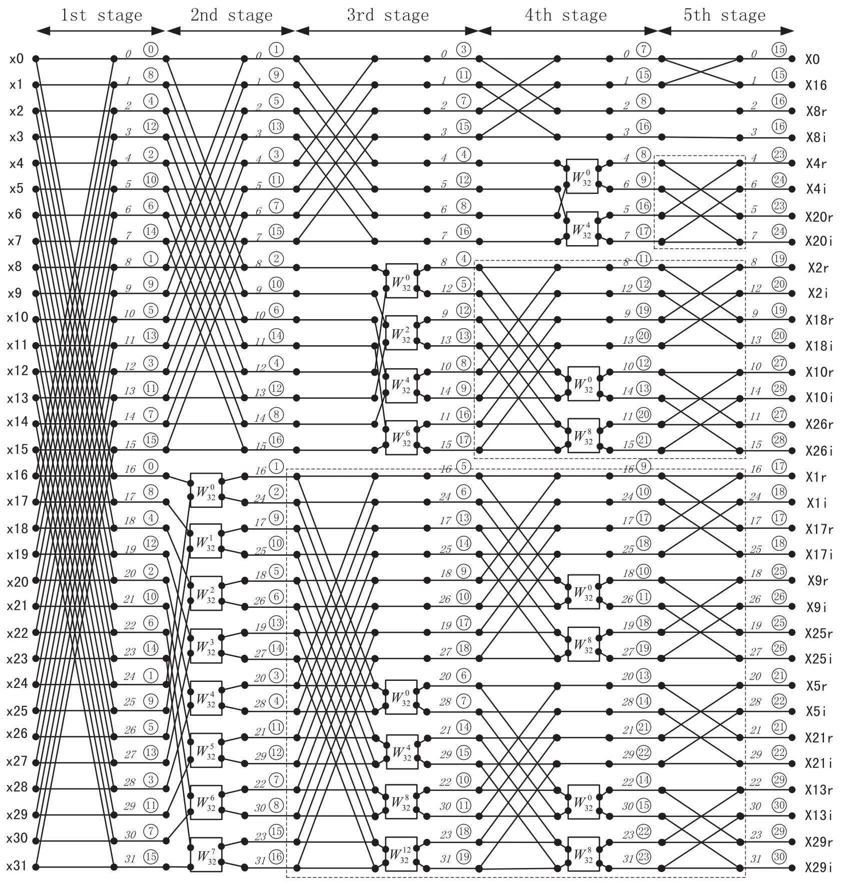

Fig. 1. Real data flow graph of radix-2 RFFT ($N = 32$).

and $k = 0, 1, \ldots, 2^{t-1} - 1$, the result data of set $\tilde{S}_t$ could be calculated as follows:

$$X\left((4k+1)\cdot 2^{n-t-1}\right) = \sum_{m=0}^{2^{t-1}-1} \tilde{X}(t,m) W_{2^{t-1}}^{mk}$$

where $\tilde{X}(t,m) = \sum_{l=0}^{2^{n-t+1}-1} x(l \cdot 2^{t-1} + m) W_4^l W_{2^{t+1}}^m = W_{2^{t+1}}^m \sum_{l=0}^{2^{n-t-1}-1} [(x(l \cdot 2^{t+1} + m) - x(l \cdot 2^{t+1} + 2^t + m)) - j \times (x(l \cdot 2^{t+1} + 2^{t-1} + m) - x(l \cdot 2^{t+1} + 3 \cdot 2^{t-1} + m))]$.

It is clear that the elements in set $\tilde{S}_t (t = 1, 2, \ldots, n-1)$ could be computed through $2^{t-1}$-point CFFT after $(n-t)$ stages of butterflies and one stage of complex multiplication.

Based on the aforementioned deduction, the radix-2 RFFT can be expressed in terms of butterfly-style (BF) with only real datapaths. The data flow graph of the 32-point RFFT is illustrated as an example in Fig. 1. The dashed squares in Fig. 1 are referred to as the CFFT data flow graph, with the complex data split into real part and imaginary part. As such, the biggest dashed square processes an 8-point CFFT, while the smallest one processes a 2-point CFFT. Thus, RFFT could be expressed by a combination of CFFT and RFFT of smaller size.

The structure consists of $n$ stages as illustrated, each of which contains only $N$ real samples, therefore with only real datapaths, and is composed of butterflies and $W^k$-blocks. The stage structure of the architecture could be divided into three categories: either of the first stage and the last stage only contains one column of butterfly units, and the second stage contains one half column of butterfly units and one half column of $W^k$-blocks, while each of the other $(n-3)$ stages contains one column of butterfly units, followed by one column of $W^k$-blocks. The italic and circled numbers in Fig. 1 represent the data indices and timing instances of the computations in each stage, respectively. The cycle zero of timing instance is denoted as the first cycle of arithmetical processing.

## III. Proposed Architectures for RFFT Computation

In this section, the proposed structure is mapped to radix-2 pipelined architecture. Moreover, several processing engines (PEs) are designed to make full use of hardware resource.

### A. Proposed RFFT Architecture

As shown in Fig. 2(a), the two-parallel proposed RFFT architecture consists of $n$ stages, each of which contains data commutators and several calculating PEs. The rearrange unit (RU) at the beginning of the architecture is designed to reorder the input data to the required sequence. The data commutator in each stage shuffles the input data to generate a new sequence. Each stage of the architecture (expect RU) processes one stage computation in Fig. 1, respectively. Stage 1 and stage $n$ contain BF PE, stage 2 has one single-path butterfly (SBF) PE and one single-path CM (SCM) PE in either path, while other $(n-3)$ stages only have real-datapath delay commutator (RDC) PEs for calculation. The implementation details of the PEs will be described in Section III-B.

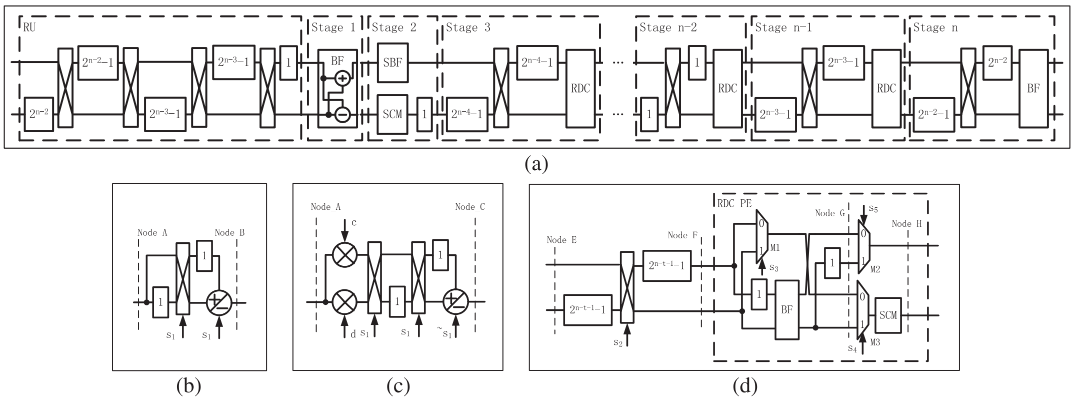

Fig. 2. (a) Block diagram of the proposed two-parallel pipelined architecture. (b) Block diagram of the SBF PE. (c) Block diagram of the SCM PE. (d) Block diagram of stage $3, \ldots, (n-2)$ and the RDC PE. (e) Block diagram of the proposed four-parallel pipelined architecture.

TABLE I
DATA ORDER OF THE PROPOSED 32-POINT RFFT ARCHITECTURE

| cycle | input | $s_1s_2s_3s_4$ | stage1 | stage2 | stage3 | stage4 | stage5 |
|---|---|---|---|---|---|---|---|
| 0 | 0,1 | 0100 | - | - | - | - | - |
| 1 | 2,3 | 0001 | - | - | - | - | - |
| … | … | … | … | … | … | … | … |
| 11 | 22,23 | 1011 | 0,16 | - | - | - | - |
| 12 | 24,25 | 1110 | 8,24 | 0,- | - | - | - |
| 13 | 26,27 | 1011 | 4,20 | 8,16 | - | - | - |
| 14 | 28,29 | 1110 | 12,28 | 4,24 | 0,- | - | - |
| 15 | 30,31 | 1001 | 2,18 | 12,20 | 4,8 | - | - |
| 16 | - | 0100 | 10,26 | 2,28 | 16,12 | - | - |
| 17 | - | 0001 | 6,22 | 10,18 | 24,20 | - | - |
| 18 | - | 0100 | 14,30 | 6,26 | 2,28 | 0,- | - |
| 19 | - | 0011 | 1,17 | 14,22 | 6,10 | 2,4 | - |
| 20 | - | 0110 | 9,25 | 1,30 | 18,14 | 16,6 | - |
| 21 | - | 0011 | 5,21 | 9,17 | 26,22 | 24,18 | - |
| 22 | - | 0110 | 13,29 | 5,25 | 1,30 | 8,26 | - |
| 23 | - | 0001 | 3,19 | 13,21 | 5,9 | 12,10 | - |
| 24 | - | 1100 | 11,27 | 3,29 | 17,13 | 20,14 | - |
| 25 | - | 1001 | 7,23 | 11,19 | 25,21 | 28,22 | - |
| 26 | - | 1100 | 15,31 | 7,27 | 3,29 | 1,30 | 0,1 |
| 27 | - | 1011 | - | 15,23 | 7,11 | 3,5 | 2,3 |
| 28 | - | 1110 | - | -,31 | 19,15 | 17,7 | 16,17 |
| 29 | - | 1011 | - | - | 27,23 | 25,19 | 24,25 |
| 30 | - | 1110 | - | - | -,31 | 9,27 | 8,9 |
| … | … | … | … | … | … | … | … |
| 41 | - | 1001 | - | - | - | - | 30,31 |

TABLE II
DATA SEQUENCE OF SBF AND SCM

| cycle | Node A | $s_1$ | Node B | Node C | Node D |
|---|---|---|---|---|---|
| 0 | $x_1$ | 0 | - | $c \cdot x_1, d \cdot x_1$ | - |
| 1 | $x_2$ | 1 | $x_1 + x_2$ | $d \cdot x_2, c \cdot x_2$ | $c \cdot x_1 - d \cdot x_2$ |
| 2 | $x_3$ | 0 | $x_1 - x_2$ | $c \cdot x_3, d \cdot x_3$ | $d \cdot x_1 + c \cdot x_2$ |
| 3 | $x_4$ | 1 | $x_3 + x_4$ | $d \cdot x_4, c \cdot x_4$ | $c \cdot x_3 - d \cdot x_4$ |
| 4 | $x_5$ | 0 | $x_3 - x_4$ | $c \cdot x_5, d \cdot x_5$ | $d \cdot x_3 + c \cdot x_4$ |

TABLE III
DATA SEQUENCE OF STAGE 3 FOR THE 32-POINT RFFT

| cycle | Node E | $s_2s_3s_4s_5$ | Node F | Node G | Node H |
|---|---|---|---|---|---|
| 0 | 0, - | 0110 | - | - | - |
| 1 | 8,16 | 0010 | 0,- | - | - |
| 2 | 4,24 | 1100 | 8, 4 | 0,-, 8, - | 0,- |
| 3 | 12,20 | 1001 | 16,12 | -,4,12, - | 4,8 |
| 4 | 2,28 | 0110 | 24,20 | 16,-, -,20 | 16,12 |
| 5 | 10,18 | 0010 | 2,28 | 24, -, -28 | 24,20 |
| 6 | 6,26 | 1100 | 10, 6 | 2,-,10, - | 2,28 |
| 7 | 14,22 | 1001 | 18,14 | -,6,14, - | 6,10 |
| 8 | 1,30 | 0110 | 26,22 | 18,-, -,22 | 18,14 |
| 9 | 9,17 | 0010 | 1,30 | 26,-, -,30 | 26,22 |
| 10 | 5,25 | 1100 | 9, 5 | 1,-, 9, - | 1,30 |
| 11 | 13,21 | 1001 | 17,13 | -,5,13, - | 5,9 |
| 12 | 3,29 | 0110 | 25,21 | 17,-, -,21 | 17,13 |
| 13 | 11,19 | 0010 | 3,29 | 25,-, -,29 | 25,21 |
| 14 | 7,27 | 1100 | 11, 7 | 3,-,11, - | 3,29 |
| 15 | 15,23 | 1001 | 19,15 | -,7,15, - | 7,11 |
| 16 | -,31 | 0110 | 27,23 | 19,-, -,23 | 19,15 |
| 17 | - | 0010 | -,31 | 27,-, -,31 | 27,23 |
| 18 | - | 1100 | - | - | -,31 |

The data output sequence of each stage of the two-parallel 32-point RFFT architecture is illustrated in Table I. The real input data at cycle $m$ is $(2*m, 2*m+1)$. The data sequence of RU is not included since RU has the same output order as stage 1. Moreover, the index differences of the commutators in RU are $2^{n-2}, 1, 2^{n-3}, 1$, respectively (when the index difference is $b$, the control signal of the commutator is of cycle period $2b$: it maintains zeros at the first $b$ cycles, then turns to one at the second $b$ cycles, and so forth. The control signals of the commutators in RU are denoted as $s_1, s_2, s_3, s_4$, shown in Table I). In stage $t$, the index differences are $2^{n-t-1}$ for stage $3, \ldots, (n-2)$ and $2^{t-2}$ for the last two stages. The new data sequence is critical to the calculating unit, requiring only one butterfly unit (BF PE or SBF PE) and at most one SCM PE for each stage. Moreover, BF PE, SBF PE, and SCM PE preserve the data sequence.

## *B. PEs*

Aiming at the goal of full source utilization, SCM PE [2], SBF PE, and RDC PE are designed for computation, as illustrated in Fig. 2(b)–(d).

The SBF PE (SCM PE) behaves as SBF (CM) only in the upper (lower) path of stage 2. The data sequence of SBF PE and SCM PE is shown in Table II. In addition, $c$ and $d$ in Fig. 2(c) are denoted as the real and imaginary parts of twiddle factors, respectively. Both SBF PE and SCM PE share the adders and multipliers in the time-multiplexing approach, therefore reducing arithmetic resources by half.

In stage $t$ $(t = 3, \ldots, n-3)$, according to the real data flow graph of RFFT as shown in Fig. 1, $(2^n - 2^{n-t+1})$ data require butterfly structure, and one half of data require complex multiplication. Therefore, at least one butterfly unit (BF PE) and one multiplying unit (SCM PE) are required in each stage. As shown in Fig. 2(d), the RDC PE consists of one BF PE, one SCM PE, and three multiplexers ($M1$, $M2$, and $M3$). Table III illustrates the data sequence of stage 3 for 32-point RFFT computation. The data commutator shuffles the input data (node E) of stage $t$ to generate a new data sequence

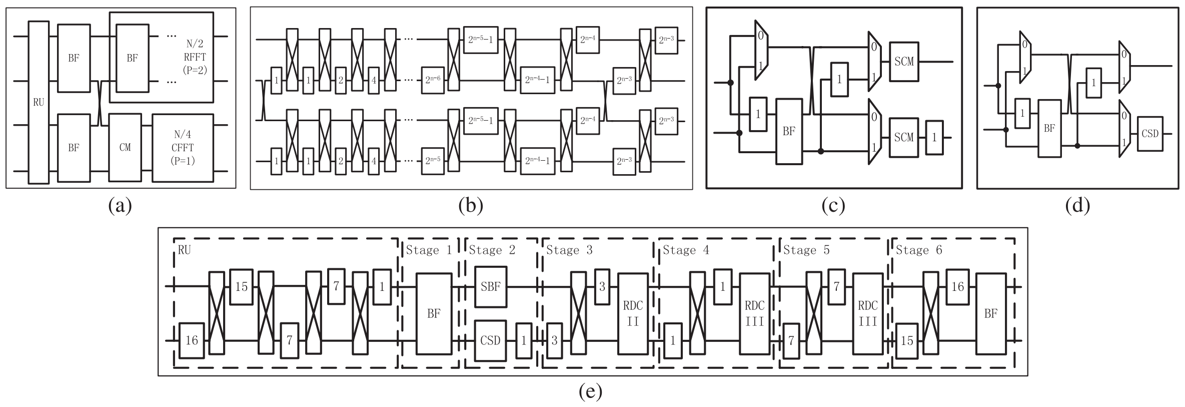

Fig. 3. (a) Block diagram of the proposed four-parallel pipelined architecture. (b) Block diagram of the RU of the four-parallel architecture. (c) Block diagram of the RDC II PE. (d) Block diagram of the RDC III PE. (e) Block diagram of the proposed two-parallel pipelined radix-$2^3$ 64-point architecture.

(node F). Three multiplexers are used for data selection to reorder the data and to maximize the utilization of arithmetic resources. The multiplier $M1$ selects the data without butterfly calculation, $M3$ allows the ones requiring multiplying with twiddling factors to pass through, and $M2$ selects the ones only requiring butterfly. The data in the last two paths of node G require a complex multiplication, while those of the first two paths do not. After the aforementioned progress, the output data sequence (node H) can be achieved, which is also the input data sequence (node D) of stage $(t+1)$. The aforementioned mechanism can be iterated by applying to stage $(n-1)$. Additionally, stage $(n-1)$ actually needs only $W_8^1$ multiplication for $W^k$-block. Moreover, it can be simply implemented by canonical-signed-digital (CSD) multiplication [16].

In stage $3,\ldots,n-1$, only the operation of BF requires interactive processing between the two paths. Therefore, data alignment of two paths should be guaranteed only before BF PE. Accordingly, the structures of stage $3,\ldots,n-1$ can be optimized, resulting in that delay registers are decreased by $O(\log_2 N)$ and that the data of the first path is always one cycle ahead of the second path in stage $3,\ldots,n-1$ in Table I.

The commutator and multiplexer in the architecture can be easily implemented. All of the commutators and most of the multiplexers can be simply controlled by an $n$-bit counter, such as $s_1$ in Table II and $s_2, s_3, s_4$ in Table III. In addition, $s_5$ is designed with a combination of $s_3$ and $s_4$ ($s_5 = !(s_3|s_4)$).

### C. Generalization of the Architectures

The proposed algorithm can also be extended to higher parallelism and higher radix (i.e., radix-$2^i$).

A four-parallel pipelined architecture is made as an example to describe the systematic method. Under the analysis of Section II, it is easy to notice that $\bigcup_{i=0}^{n-2} \tilde{S}_i$ can also be computed through $2^{n-1}$-point RFFT after a stage of butterfly computation, while $\tilde{S}_{n-1}$ can be computed through $2^{n-2}$-point CFFT. Fig. 3(a) presents the proposed four-parallel $N$-point RFFT architecture, in which $P$ is denoted as the level of parallelism. The architecture consists of four parts: the RU to reorder the input data, two BF PEs, one two-parallel $N/2$-point RFFT (without RU), and one four-parallel $N/4$-point CFFT with one CM. The RU of the four-parallel architecture is designed as shown in Fig. 3(b) with $2^{n-2}$ delay registers, in which the index difference of the commutator is $1, 2, \ldots, 2^{n-4}$, respectively. The architecture of [2] is applied as the CFFT architecture in Fig. 3(a) with 100% utilization of hardware resources, of which the two input datapaths stand for the real and imaginary parts of the data, respectively. Furthermore, the RFFT architectures with higher parallelization can be designed with the iterative method in the same way.

The radix-$2^i$ RFFT algorithms are also applied to reduce the number of multiplication with twiddle factors, therefore decreasing the number of multipliers. Radix-$2^i$ RFFT can be implemented with reduced multiplication complexity. For example, in the radix-$2^3$ 64-point RFFT architecture shown in Fig. 3(e), each of stages 2, 4, and 5 has one half column of $W_8^k$-blocks, which can be replaced with CSD multiplication, and stage 3 has one column of $W^k$-blocks, requiring two SCM PEs. As a result, the multipliers of the radix-$2^3$ architecture can be reduced by one third.

## IV. Comparison and Experimental Results

The hardware complexities of the proposed and other two-parallel pipelined RFFT architectures are summarized in Table IV, in terms of CMs, real adders, delay registers (without twiddle factor memory), and latency. Since the input data of the architectures in [9] and [15] are average-cut ordered, the input data should be rearranged in the beginning with the extra cost of at least $((l-1)/l)N$ delay registers and $((l-1)/l^2)N$ latency for $l$-parallel architecture. One CM can be referred to as four multipliers and two adders; therefore, one SCM is considered as one-half CM. When it is implemented by three multipliers and four adders, one SCM is no more than two-third CM since its resource is halved. Compared with those of [9], [13], and [15], the proposed two-parallel architecture requires the same number of real adders, while the number of multipliers is halved, and the numbers of delay registers and latency are both decreased with respect to radix-2 and radix-$2^3$, as shown in Table IV. When the level of parallelism is higher than 2, the proposed architectures have slightly more delay registers and latency.

The proposed architectures can be applied to both fixed- and floating-point data, and the accuracy for fixed-point FFT

TABLE IV
HARDWARE COMPLEXITY COMPARISON OF ARCHITECTURES FOR PARALLEL PIPELINED RFFT WITH $N$-POINT

| Architecture | Parallel | Input Order | Complex Multiplier[3] | Real Adder | Delay Registers | Latency |
|---|---|---|---|---|---|---|
| radix-2 [13] | 2 | Normal[1] | $log_2N-2$ | $2log_2N-2$ | $\frac{7}{4}N$ | $\frac{7}{8}N$ |
| radix-2 [9] | 4 | Average-Cut[2] | $log_2N-2$ | $4log_2N$ | $\frac{7}{4}N$ | $\frac{7}{16}N$ |
| radix-2 [15] | 2 | Average-Cut | $log_2N-3$ | $2log_2N$ | $2N-6$ | $N-3$ |
|  | 4 |  | $log_2N-3$ | $4log_2N-2$ | $\frac{7}{4}N-4$ | $\frac{7}{16}N-1$ |
| radix-2 proposed | 2 | Normal[1] | $\frac{1}{2}(log_2N-3)$ | $2log_2N-1$ | $\frac{7}{4}N+2log_2N-8$ | $\frac{7}{8}N-2$ |
|  | 4 |  | $log_2N-4$ | $4log_2N-6$ | $\frac{15}{8}N+3log_2N-13$ | $\frac{15}{32}N-2$ |
| radix-$2^3$ [11] | 2 | Normal | $\frac{1}{3}log_2N-1$ | $4log_2N-2$ | $N-2$ | $\frac{1}{2}N-1$ |
| radix-$2^3$ [12] | 2 | Normal | $\frac{1}{3}log_2N-1$ | $2log_2N$ | $\frac{2}{3}N-2$ | $\frac{1}{3}N-1$ |
|  | 4 |  | $\frac{2}{3}log_2N-2$ | $4log_2N-2$ | $\frac{7}{4}N-4$ | $\frac{7}{16}N-1$ |
| radix-$2^3$ [15] | 2 | Average-Cut | $\frac{2}{3}log_2N-2$ | $2log_2N$ | $2N-6$ | $N-3$ |
|  | 4 |  | $\frac{2}{3}log_2N-2$ | $4log_2N-2$ | $\frac{7}{4}N-4$ | $\frac{7}{16}N-1$ |
| radix-$2^3$ proposed | 2 | Normal | $\frac{1}{3}log_2N-1$ | $2log_2N$ | $\frac{7}{4}N+3log_2N-11$ | $\frac{7}{8}N+\frac{1}{3}log_2N-4$ |
|  | 4 |  | $\frac{2}{3}log_2N-2$ | $4log_2N-5$ | $\frac{15}{8}N+4log_2N-16$ | $\frac{15}{32}N+\frac{1}{3}log_2N-4$ |

[1] Normal order of $l$-parallel represents the input data with index $i-1, i-1+l, \cdots, i-1+(\frac{N}{l}-1)l$ on the $i$-th path.
[2] Average-Cut order of $l$-parallel represents the input data with index $\frac{i-1}{l}N, 1+\frac{i-1}{l}N, \cdots, \frac{N}{l}-1+\frac{i-1}{l}N$ on the $i$-th path.
[3] Complex multiplier mentioned herein is referred as four real multipliers and two real adders. When complex multiplier is implemented by three real multipliers and four real adders, the resource for complex multipliers in proposed architectures increases by no more than $\frac{1}{3}$.

TABLE V
EXPERIMENTAL RESULTS FOR THE TWO-PARALLEL PIPELINED RADIX-2 RFFT ARCHITECTURES

| Archi-tecture | RFFT Size | Area |  | Max Freq | Power[1] |
|---|---|---|---|---|---|
|  |  | Slices | BRAMs[2] | (MHz) | (mW) |
| [15] | 16 | 612 | 0 | 461 | 26.8 |
|  | 32 | 795 | 0 | 461 | 35.2 |
|  | 64 | 994 | 0 | 459 | 41.6 |
| proposed | 16 | 381 | 0 | 464 | 16.7 |
|  | 32 | 513 | 0 | 464 | 24.8 |
|  | 64 | 618 | 0 | 463 | 28.5 |
|  | 256 | 825 | 0 | 462 | 36.4 |
|  | 2048 | 1288 | 0 | 460 | 72.6 |
|  | 4096 | 1455 | 5 | 430 | 83.4 |
|  | 16384 | 1545 | 29 | 423 | 92.1 |

[1] The power consumption is evaluated under 200MHz.
[2] The size of Block-RAM used in the implementation is 18kb.

is investigated in [17]. To be fair, this brief makes signal data of 16-bit length as an example. Table V shows the synthesis results achieved by the implementation of the proposed two-parallel radix-2 architectures on Xilinx Virtex-5 FPGA, XC5VSX110T. The performance is evaluated based on the post place and route results from Xilinx toolset. Block-RAMs are used only when the size of RFFT reaches 4096; otherwise, only distributed memory is used. The results of Table V show that the proposed two-parallel architectures achieve 30% smaller number of slices and lower power consumption compared with [15], when the size of CFFT is 64 or less. Therefore, the proposed architectures achieve small area, high frequency, and low power consumption.

## V. CONCLUSION

This brief has proposed a radix-2 algorithm with only real datapaths through formula deduction. Based on the algorithm, an efficient two-parallel architecture of pipelined RFFT has been presented. The proposed architecture achieves higher hardware resource utilization of the PEs and delay registers; therefore, it is very efficient for implementation. An iterative method is also presented for the architectures of pipelined RFFT with higher parallelism and higher radix. The experimental result also shows that the proposed architectures are resource-efficient and also work well for low-power scenarios.

## REFERENCES

[1] H. L. Groginsky and G. A. Works, "A pipeline fast Fourier transform," *IEEE Trans. Comput.*, vol. C-19, no. 11, pp. 1015–1019, Nov. 1970.

[2] Z. Wang, X. Liu, B. He, and F. Yu, "A combined SDC–SDF architecture for normal I/O pipelined radix-2 FFT," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 5, pp. 973–977, May 2015.

[3] H.-F. Chi and Z.-H. Lai, "A cost-effective memory-based real-valued FFT and Hermitian symmetric IFFT processor for DMT-based wire-line transmission systems," in *Proc. IEEE ISCAS*, May 2005, vol. 6, pp. 6006–6009.

[4] M.-H. Cheng, L.-C. Chen, Y.-C. Hung, and C. M. Yang, "A real-time maximum-likelihood heart-rate estimator for wearable textile sensors," in *Proc. IEEE 30th Annu. Int. Conf. EMBS*, Aug. 2008, pp. 254–257.

[5] Y. S. Park, T. I. Netoff, and K. K. Parhi, "Reducing the number of features for seizure prediction of spectral power in intracranial EEG," in *Proc. Conf. Rec. 46th ASILOMAR*, Nov. 2012, pp. 770–774.

[6] H. V. Sorensen, D. L. Jones, M. Heideman, and C. S. Burrus, "Real-valued fast Fourier transform algorithms," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 35, no. 6, pp. 849–863, Jun. 1987.

[7] I. S. Uzun, A. Amira, and A. Bouridane, "FPGA implementations of fast Fourier transforms for real-time signal and image processing," *Proc. Inst. Elect. Eng.—Vis., Image Signal Process.*, vol. 152, no. 3, pp. 283–296, Jun. 2005.

[8] M. Ayinala, Y. Lao, and K. K. Parhi, "An in-place FFT architecture for real-valued signals," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 60, no. 10, pp. 652–656, Oct. 2013.

[9] M. Garrido, K. K. Parhi, and J. Grajal, "A pipelined FFT architecture for real-valued signals," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 12, pp. 2634–2643, Dec. 2009.

[10] M. Ayinala and K. K. Parhi, "Parallel-pipelined radix-$2^2$ FFT architecture for real valued signals," in *Proc. Conf. Rec. 44th ASILOMAR*, Nov. 2010, pp. 1274–1278.

[11] M. Ayinala, M. Brown, and K. K. Parhi, "Pipelined parallel FFT architectures via folding transformation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 6, pp. 1068–1081, Jun. 2012.

[12] M. Ayinala and K. K. Parhi, "FFT architectures for real-valued signals based on radix-$2^3$ and radix-$2^4$ algorithms," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 9, pp. 2422–2430, Sep. 2013.

[13] A. X. Glittas and G. Lakshminarayanan, "Pipelined FFT architectures for real-time signal processing and wireless communication applications," in *Proc. 18th Int. Symp. VLSI Des. Test*, Jul. 2014, pp. 1–2.

[14] P. Zode, A. Thor, and A. Y. Deshmukh, "Folded FFT architecture for real-valued signals based on radix-$2^3$ algorithm," in *Proc. IEEE 2nd ICDCS*, Mar. 2014, pp. 1–4.

[15] S. A. Salehi, R. Amirfattahi, and K. K. Parhi, "Pipelined architectures for real-valued FFT and Hermitian-symmetric IFFT with real datapaths," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 60, no. 8, pp. 507–511, Aug. 2013.

[16] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*, vol. 9. New York, NY, USA: Springer-Verlag, no. 1, 1999, pp. 165–170.

[17] W.-H. Chang and T. Q. Nguyen, "On the fixed-point accuracy analysis of FFT algorithms," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4673–4682, Oct. 2008.