IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 June 2023, accepted 18 July 2023, date of publication 24 July 2023, date of current version 8 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3298218

RESEARCH ARTICLE

Radix-2^k MSC FFT Architectures

GUANG-TING DENG[1], MARIO GARRIDO[2], (Senior Member, IEEE), SAU-GEE CHEN[1], AND SHEN-JUI HUANG[3]

[1]Institute of Electronics, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan
[2]Department of Electronic Engineering, ETSI de Telecomunicación, Universidad Politécnica de Madrid, 28040 Madrid, Spain
[3]Hsinchu Science Park, Hsinchu 300, Taiwan

Corresponding author: Mario Garrido (mario.garrido@upm.es)

This work was supported in part by Comunidad de Madrid through the call "Ayudas de Estímulo a la Investigación de Jóvenes Doctores de la Universidad Politécnica de Madrid" under Project APOYO-JOVENES-21-TL23SB-116-I4FOMC; in part by MCIN/AEI/10.13039/501100011033 and "ESF Investing in your future" under Grant RYC2018-025384-I; and in part by the Ministry of Science and Technology from Taiwan under Grants of MOST 108-2221-E-009-100.

ABSTRACT In recent years, the SC FFT architecture has become popular for processing serial data. It requires a small number of components and achieves full utilization of the butterflies, which improves previous serial FFT architectures. By contrast, the MSC FFT architecture, which is the parallel version of the SC FFT, has not been studied in depth in the literature and it has not been analyzed if this new type of FFT architecture improves previous parallel FFTs. The aim of this paper is to provide a rigorous study of MSC architectures that expands the field of FFT architectures by incorporating fundamental knowledge about this promising FFT. With this goal, this paper proposes new MSC FFT architectures for any FFT size, radix, and parallelization. In order to derive these architectures, efficient modules have been developed. These modules are connected by permutation circuits to create the architectures. The optimization of the modules results in a reduction in the number of rotators and their complexity compared to previous designs. As a result, the proposed architectures not only achieve high throughput due to their parallel nature but also the lowest hardware complexity among parallel pipelined FFT architectures so far. To verify the architectures and compare the proposed approach to previous works, a 1024-point MSC FFT architecture has been implemented. Experimental results show that the architecture achieves a throughput of 1.32 gigasamples per second, and reduces the area and power consumption significantly with respect to previous designs.

INDEX TERMS Fast Fourier transform (FFT), multi-path serial-commutator (MSC), pipelined architecture, radix-2^k.

I. INTRODUCTION

The fast Fourier transform (FFT) is a key algorithm in the field of digital signal processing. It plays an important role in multiple applications such as digital communications [1], [2], [3], [4], [5], [6], [7], radio astronomy [8], [9] and medical imaging [10]. In order to meet the stringent requirements of these and other modern applications, hardware designers are constantly searching for new FFT architectures that improve the state-of-the-art and provide highly efficient solutions.

For applications that are implemented on application-specific integrated circuits (ASICs) or field-programmable gate arrays (FPGAs), two main types of FFT architectures can be used: iterative and pipelined. Iterative FFT architectures [4], [11], [12] consist of a bank of memories and one or several processing elements. The processing elements read data from memory, process them, and store the results back in memory until all the computations of the FFT algorithm have been calculated. Conversely, pipelined FFT architectures [13] process data by a set of stages connected in series in a pipeline. Serial pipelined FFT architectures receive one sample per clock cycle. Among them, single-path delay feedback (SDF) [14], [15], single-path delay commutator (SDC) [16], [17] and single-path serial commutator (SC) [18], [19], [20], [21], [22], [23] are the most common architectures. Parallel pipelined FFT architectures process several samples

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

per clock cycle, which results in higher throughput. Among them, multi-path delay feedback (MDF) [1], [2], [3], [5], [6], [24], [25], [26], [27], [28] and multi-path delay commutator (MDC) [29], [30], [31], [32], [33], [34], [35] are the most common architectures. Additionally, in a previous work [36] we proposed the first multi-path serial commutator (MSC) FFT architecture, which is the parallel version of the SC FFT.

Among serial FFT architectures, SC FFTs [19], [20], [21], [22], [23] have become popular in the last years due to the fact that they require a small number of components and achieve full utilization of the butterflies. Although SC FFTs are more efficient than other architectures for the case of serial data, no previous work has studied if MSC FFT architectures, which are the parallel version of SC FFTs, are also more efficient than other parallel pipelined FFT architectures in the literature.

The aim of this paper is to provide a rigorous analysis of MSC architectures that expands the field of FFT architectures by incorporating fundamental knowledge about this promising FFT architecture. With this goal, this paper proposes new MSC FFT architectures for any FFT size, radix-2^k, and parallelization. In order to derive these architectures, efficient modules for radix-2^2, 2^3, 2^4, and 2^5 have been developed. These modules are connected by permutation circuits to create the architectures. The optimization of the modules results in a reduction in the number of rotators and their complexity compared to previous designs. As a result, the proposed architectures not only achieve high throughput due to their parallel nature but also the lowest hardware complexity among parallel pipelined FFT architectures so far.

The contributions of this paper are:

- A thorough analysis of MSC FFT architectures, which is a new type of FFT architecture that has not been studied in depth in the literature yet.
- Derivation and explanation of MSC architectures for any FFT size, radix-2^k, and parallelization.
- A modular approach that allows for designing MSC FFT architectures by connecting basic modules with permutation circuits.
- New MSC architectures that reduce the number of rotators and their complexity with respect to previous architectures in the state-of-the-art.
- Experimental results that show the improvement in area and performance of the proposed approach with respect to the state-of-the-art.

This paper is organized as follows. In Section II, we review the background that is required to understand this work. In Section III, we present the radix-2^k modules, which are the basic elements used to build the proposed architectures. In Section IV, we present the proposed architectures. In Section V, we provide implementation results and compare them to previous approaches. Finally, in Section VI, we summarize the main conclusions of the paper.

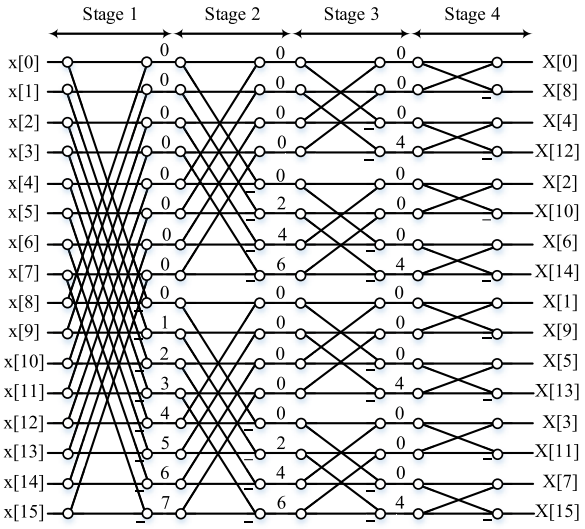

FIGURE 1. Flow graph of a 16-point radix-2 DIF FFT.

II. BACKGROUND

A. THE FAST FOURIER TRANSFORM

The N-point discrete Fourier transform (DFT) of a signal $x[n]$, $n = 0, \ldots, N-1$, is defined as

$$X[k] = \sum_{n=0}^{N-1} x[n]\ W_N^{nk}, k = 0, 1, \ldots, N-1, \qquad (1)$$

where $X[k]$ is the DFT coefficient for frequency k, and the term $W_N^{nk} = e^{-j\frac{2\pi}{N}nk}$ is called twiddle factor. In this work, we assume that N is a power of 2.

The FFT reduces the computational complexity of the DFT from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log_2 N)$. This is possible thanks to the fact that the calculation of the DFT includes operations that are repeated for different frequencies.

FFT algorithms are generally represented by their flow graph. Fig. 1 shows the flow graph of a 16-point FFT [37] decomposed according to the radix-2 decimation in frequency (DIF) algorithm. It consists of $n = \log_2 N = \log_2 16 = 4$ stages. Each stage includes butterflies and rotations. The butterflies calculate additions and subtractions, whereas the rotations calculate the multiplications by the twiddle factors, which are represented by the numbers ϕ in between stages. A rotation by ϕ corresponds to a multiplication by

$$W_N^{\phi} = e^{-j\frac{2\pi}{N}\phi}. \qquad (2)$$

B. RADIX-2^k ALGORITHMS

Apart from radix-2 DIF, there exist other FFT algorithms [37]. Fig. 2 shows the flow graphs to build radix-2^2, 2^3 and 2^4 DIF FFTs. The radix-2^2 DIF algorithm is obtained by breaking down the angle ϕ at each odd stage into a trivial rotation by $\phi_0 = N/4$ and a remaining angle, ϕ', i.e., $\phi = \phi' + N/4$. Then the rotation by ϕ' is moved to the following

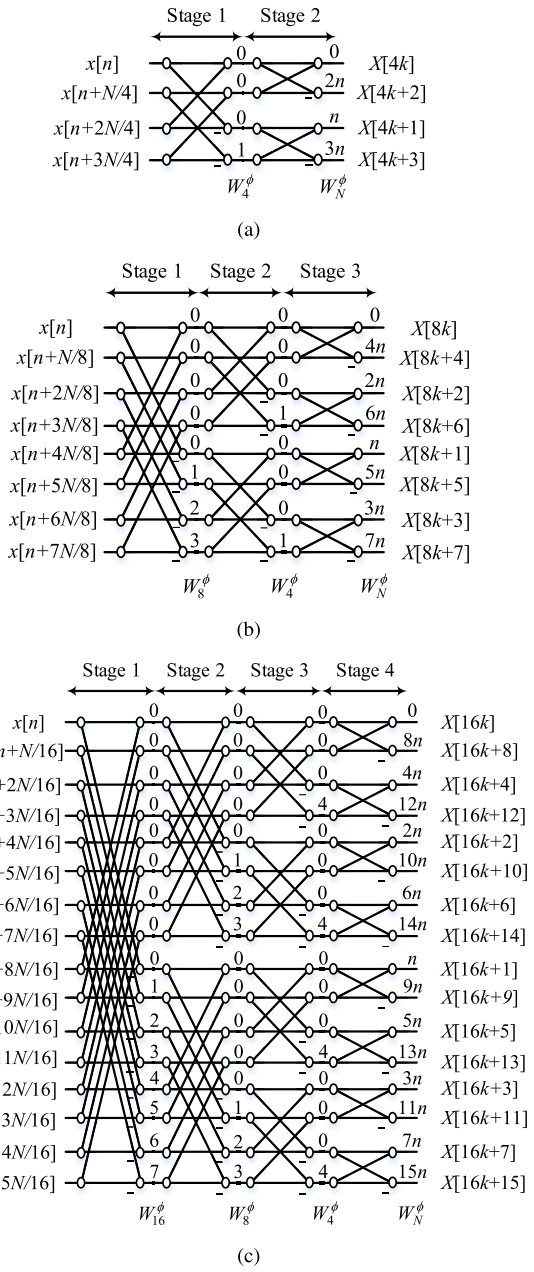

FIGURE 2. Flow graphs for radix-2^k DIF FFT. (a) Radix-2^2. (b) Radix-2^3. (c) Radix-2^4.

stage according to [30]

$$A \cdot e^{-j\frac{2\pi}{N}\phi'} \pm B \cdot e^{-j\frac{2\pi}{N}(\phi'+\frac{N}{4})} = (A \mp jB) \cdot e^{-j\frac{2\pi}{N}\phi'}, \quad (3)$$

where A and B represent the input data of the butterfly. As a result, in radix-2^2 algorithms, odd stages only include trivial rotations. These trivial rotations represent a multiplication by $-j$, which can be calculated by exchanging the real and imaginary parts of the data and changing the sign of the resulting imaginary part. This simplifies the rotators in odd stages of the FFT architecture, placing the most complex rotations in even stages. Likewise, higher radices such as radix-2^3, radix-2^4, and radix-2^5 place the most complicated rotations every three, four, and five FFT stages, respectively.

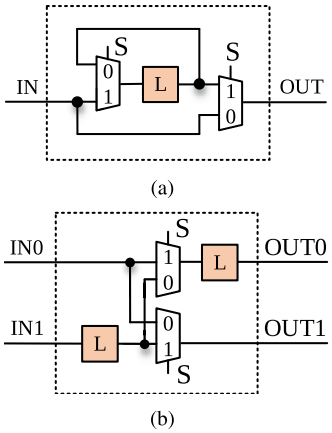

FIGURE 3. Permutation circuits in FFT architectures. (a) Circuit for serial-serial EBE. (b) Circuit for serial-parallel EBE.

C. SYMMETRIC ANGLE SETS

In FFT architectures, the components that calculate the rotations by the angles ϕ according to (2) are called rotators. Generally, the same rotator calculates rotations by several angles. In order to reduce the complexity of the rotators, it is desirable that the rotated angles can be obtained by symmetries. In this context, a symmetric angle set (SAS) [31], [38], [39] is defined as a set of angles $m\pi/2 \pm \alpha$, where $m = \{0, 1, 2, 3\}$ and $\alpha \in [0, \pi/4]$. Any rotation in a symmetric angle set can be calculated as a rotation by an angle in the range $[0, \pi/4]$, plus a trivial rotation and/or an exchange of the real and imaginary parts of the rotation coefficient.

D. M-ROTATOR

Based on the idea of SAS, an M-rotator or M-rot is defined as a rotator that can rotate angles in M different SAS. This is an indication of the complexity of the rotator, as larger M leads to higher complexity due to the fact that a larger number of different SAS must be integrated into the same rotator.

Note that a twiddle factor with the angles $W_L = e^{-j\frac{2\pi}{L}\phi}$, for $\phi = 0, \ldots, L-1$ is an $(L/8+1)$-rot, which means that it includes $L/8+1$ angles in $[0, \pi/4]$. From them, the rest of the angles can be calculated by utilizing symmetries. For instance, for the twiddle factor W_8 only the angles 0 and $\pi/4$ are in $[0, \pi/4]$, whereas the rest of the angles can be obtained by symmetries. As a result, W_8 is a 2-rot. Similarly, W_{16} is a 3-rot and W_{32} is a 5-rot.

E. PERMUTATION CIRCUITS IN FFTs

Apart from rotations and butterflies, FFT architectures include permutation circuits. These permutation circuits are based on the theory of bit-dimension permutations explained in [40].

A bit-dimension permutation considers a set of n dimensions $x_{n-1}, \ldots, x_0$ that can take the values 0 or 1. These

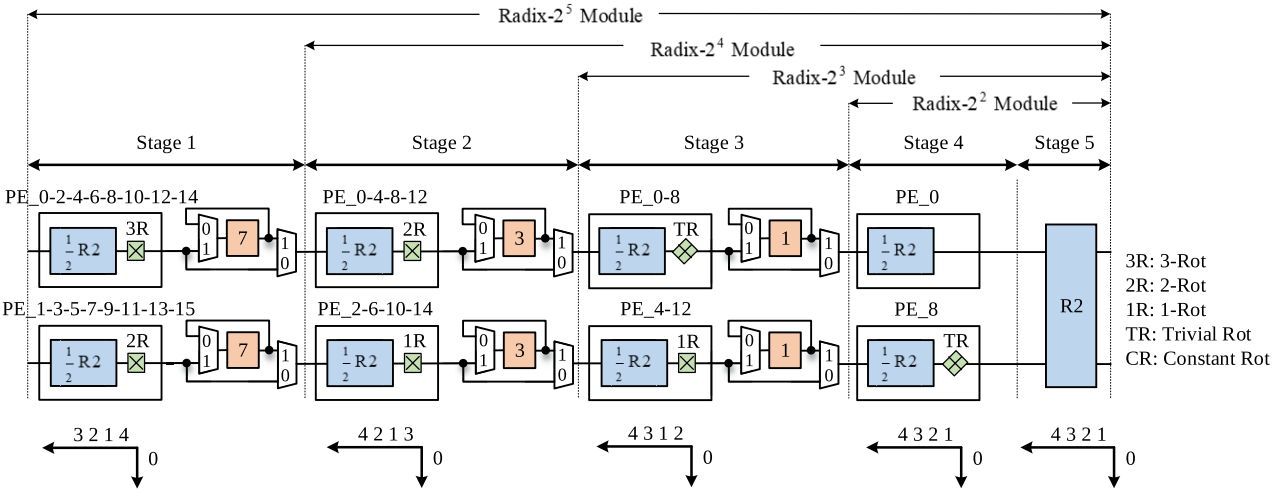

FIGURE 4. Radix-2^k modules for 2-parallel FFT architectures.

values can move to other dimensions creating a permutation, and this permutation of n elements infers a permutation of $N = 2^n$ data [40]. The simplest possible permutation is called elementary bit-exchange (EBE). An EBE exchanges the elements in two dimensions and is represented as $\sigma : x_j \leftrightarrow x_k$, where x_j and x_k represent the j-th and k-th dimensions, and $j > k$. The circuits used to calculate bit-dimension permutations are classified into serial-serial (*ss*), serial-parallel (*sp*), and parallel-parallel (*pp*) [40]. Fig. 3(a) shows the basic circuit used to calculate a serial-serial EBE. It includes a buffer of length L and two multiplexers, where the length of the buffer is calculated as

$$L = \frac{2^j - 2^k}{2^p}, \tag{4}$$

and p is the number of parallel dimensions, which is related to the number of parallel paths of the architecture, P, according to $p = \log_2 P$.

Fig. 3(b) shows the basic circuit used to calculate a serial-parallel EBE. In this case, the length of the buffer is

$$L = \frac{2^j}{2^p}. \tag{5}$$

Finally, a parallel-parallel EBE is simply an interconnection circuit that permutes the parallel paths of the architecture.

F. DATA ORDER IN FFT ARCHITECTURES

The circuits for data permutations explained in Section II-E transform the data order. To model the data order at the different stages of the architecture, parallel branches of the flow graph of the FFT are identified by an index I, being $I = 0$ for the upper branch and $I = N - 1$ for the lower branch. For instance, in the flow graph of Fig. 1, $I = 3$ corresponds to $x[3]$ at the input of the FFT, and $I = 1$ corresponds to $X[8]$ at the output of the FFT.

The binary representation of the index is defined as $I \equiv b_{n-1}, \ldots, b_0$, where the symbol ($\equiv$) is used to relate a number to its binary representation.

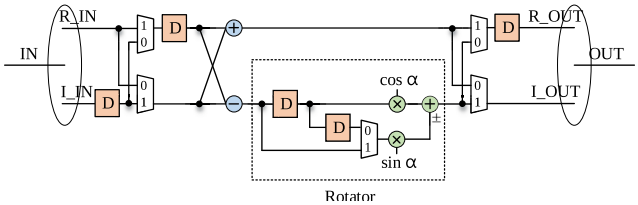

FIGURE 5. Processing element of an SC FFT.

As a result, the data flow at any stage of an FFT architecture can be defined by indicating the placement of the bits b_i. As an example, Fig. 4 include arrows under the architecture. Each set of arrows shows the order at the corresponding stage of the FFT. For instance, at stage 2 the arrow includes the numbers 4213|0, which corresponds to $b_4b_2b_1b_3|b_0$ and the vertical bar distinguishes between data that flow in series and data that flow in parallel at this point of the circuit. The part to the left of the bar indicates the time of arrival relative to the arrival of the first value, t, and the part to the left the parallel terminal, T, being in this example $t \equiv b_4b_2b_1b_3$ and $T \equiv b_0$. For instance, at the second stage of Fig. 4, the value with index $I = 11 \equiv 01011 = b_4b_3b_2b_1b_0$ in the flow graph is received at time $t \equiv b_4b_2b_1b_3 = 0011 \equiv 3$ at terminal $T \equiv b_0 = 1$. Likewise, this can be applied to identify the position of other indexes in the data flow of the architecture, which defines the exact order of the data at any place of the circuit. Further information on this notation can be found in [13] and [40].

G. THE BIT b_{n-s} IN FFTs

In the previous sections of the background, we have discussed important considerations for the rotators and for the permutation circuits in FFT architectures. In this section, we analyze the butterflies.

In a flow graph of the FFT, butterflies operate on pairs of values whose index differ in the bit b_{n-s} [13], [30], [41], where n is the total number of stages and s is the stage. If two

TABLE 1. Rotations at the Stages of the Radix-2^k modules.

Path	Stage 1	Stage 2	Stage 3	Stage 4
2-parallel				
0	$W_{32}^0, W_{32}^2, W_{32}^4, W_{32}^6$ $W_{32}^8, W_{32}^{10}, W_{32}^{12}, W_{32}^{14}$	$W_{32}^0, W_{32}^4,$ W_{32}^8, W_{32}^{12}	W_{32}^0, W_{32}^8	W_{32}^0
1	$W_{32}^1, W_{32}^3, W_{32}^5, W_{32}^7$ $W_{32}^9, W_{32}^{11}, W_{32}^{13}, W_{32}^{15}$	$W_{32}^2, W_{32}^6,$ W_{32}^{10}, W_{32}^{14}	W_{32}^4, W_{32}^{12}	W_{32}^8
4-parallel				
0	$W_{32}^0, W_{32}^4, W_{32}^8, W_{32}^{12}$	W_{16}^0, W_{32}^8	W_{32}^0	W_{32}^0
1	$W_{32}^2, W_{32}^6, W_{32}^{10}, W_{32}^{14}$	W_{32}^4, W_{32}^{12}	W_{32}^8	W_{32}^0
2	$W_{32}^1, W_{32}^5, W_{32}^9, W_{32}^{13}$	W_{32}^2, W_{32}^{10}	W_{32}^4	W_{32}^0
3	$W_{32}^3, W_{32}^7, W_{32}^{11}, W_{32}^{15}$	W_{32}^6, W_{32}^{14}	W_{32}^{12}	W_{32}^8
8-parallel				
0	W_{32}^0, W_{32}^8	W_{32}^0	W_{32}^0	W_{32}^0
1	W_{32}^4, W_{32}^{12}	W_{32}^8	W_{32}^0	W_{32}^0
2	W_{32}^1, W_{32}^9	W_{32}^2	W_{32}^0	W_{32}^0
3	W_{32}^5, W_{32}^{13}	W_{32}^{10}	W_{32}^4	W_{32}^8
4	W_{32}^2, W_{32}^{10}	W_{32}^4	W_{32}^0	W_{32}^0
5	W_{32}^6, W_{32}^{14}	W_{32}^{12}	W_{32}^8	W_{32}^0
6	W_{32}^3, W_{32}^{11}	W_{32}^6	W_{32}^0	W_{32}^0
7	W_{32}^7, W_{32}^{15}	W_{32}^{14}	W_{32}^{12}	W_{32}^8

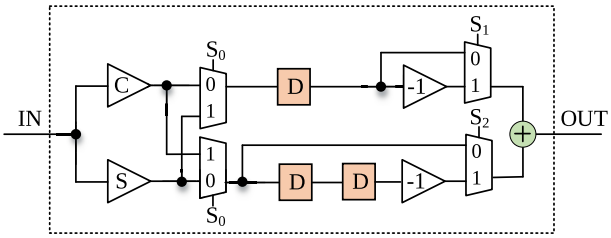

FIGURE 6. General structure for 1-rots in the PEs of the proposed MSC FFT architectures.

TABLE 2. Timing diagram of the rotator in Fig. 6.

Time	IN	S_0	S_1	S_2	OUT	Multiplicand
0	x_0	0	-	-	-	-
1	y_0	0	0	0	$x_0C + y_0S$	$C - jS$
2	x_1	1	0	1	$-x_0S + y_0C$	
3	y_1	1	0	0	$x_1S + y_1C$	$S - jC$
4	x_2	1	0	1	$-x_1C + y_1S$	
5	y_2	1	1	0	$-x_2S + y_2C$	$-S - jC$
6	x_3	0	1	1	$-x_2C - y_2S$	
7	y_3	0	1	0	$-x_3C + y_3S$	$-C - jS$
8	-	-	1	1	$-x_3S - y_3C$	

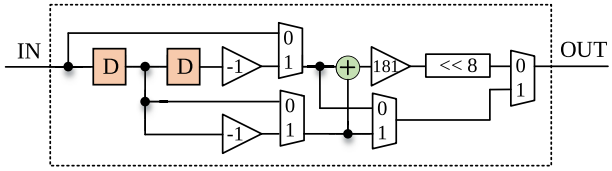

FIGURE 7. Rotator by W_{32}^{ϕ}, $\phi = 0, 4, 8, 12$.

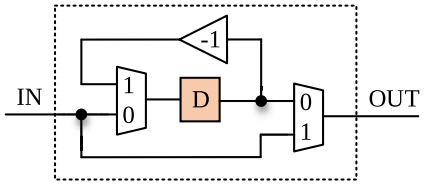

FIGURE 8. Trivial rotator in radix-2^k modules.

indexes I_0 and I_1 differ in b_{n-s}, their binary representations are similar except for the bit b_{n-s}. As the weight of b_{n-s} is 2^{n-s}, then $I_1 - I_0 = 2^{n-s}$.

If we consider the flow graph in Fig. 1, it represents a 16-point FFT, so $n = 4$. For stage $s = 1$, $2^{n-s} = 2^{4-1} = 8$, which corresponds to the separation between values that are processed together in the butterflies of the first stage: $x[0]$ and $x[8]$, $x[1]$ and $x[9]$, etc. For stage $s = 2$, $2^{n-s} = 2^{4-2} = 4$ and it can be observed that pairs of values into a butterfly of the second stage differ in 4. Likewise, b_{n-s} holds for other stages and for any FFT size.

III. RADIX-2^k MODULES FOR THE PROPOSED MSC FFTs

The design of the proposed MSC FFTs is based on a set of modules for radix-2^k that calculate 2^k-point FFTs efficiently. By combining these modules and connecting them with permutation circuits, the proposed MSC FFTs are obtained. In this Section, we present the radix-2^k modules for 2, 4, and 8-parallel data, whereas the proposed MSC FFT architectures are described in Section IV.

Fig. 4 shows the radix-2^k modules when the number of parallel paths is 2. These modules consist of processing elements (PEs) that include half butterflies ($\frac{1}{2}$R2), radix-2 butterflies (R2), rotators, and permutation circuits. Throughout the paper, diamond-shaped rotators are trivial rotators (TR); rotators in a square are constant rotators (CR), 1-rots (1R), 2-rots (2R), or 3-rots (3R); and general rotators (GR) are illustrated as (⊗).

The modules that are used for different radices are highlighted at the top of Fig. 4. Note that the radix-2^2 module corresponds to the last two stages. Likewise, the radix-2^3, 2^4 and 2^5 modules correspond to the last three, four, and five stages, respectively.

As explained in Sections II-F and II-G, arrows at the bottom of the figure define the order of the data at the corresponding stage of the circuit and data that differ in bit b_{n-s} must be placed together at the input of the butterfly. As the PEs of the proposed approach operate on pairs of data that arrive to them in consecutive clock cycles, the bit b_{n-s} must be placed in the LSB of the horizontal arrow at the stages that include PEs. Conversely, for butterflies that process two data in parallel, as in stage 5 of Fig. 4, the bit b_{n-s} must be placed in the LSB of the vertical arrow, which is the bit closer to the intersection of the arrows. These rules hold for all the proposed designs.

The structure of a PE is shown in Fig. 5 and follows the structure of the PEs in the SC FFT [18]. The PEs consist of a half-butterfly and a half-rotator. In the figure, D stands for delay and consists of a register that delays data one clock cycle.

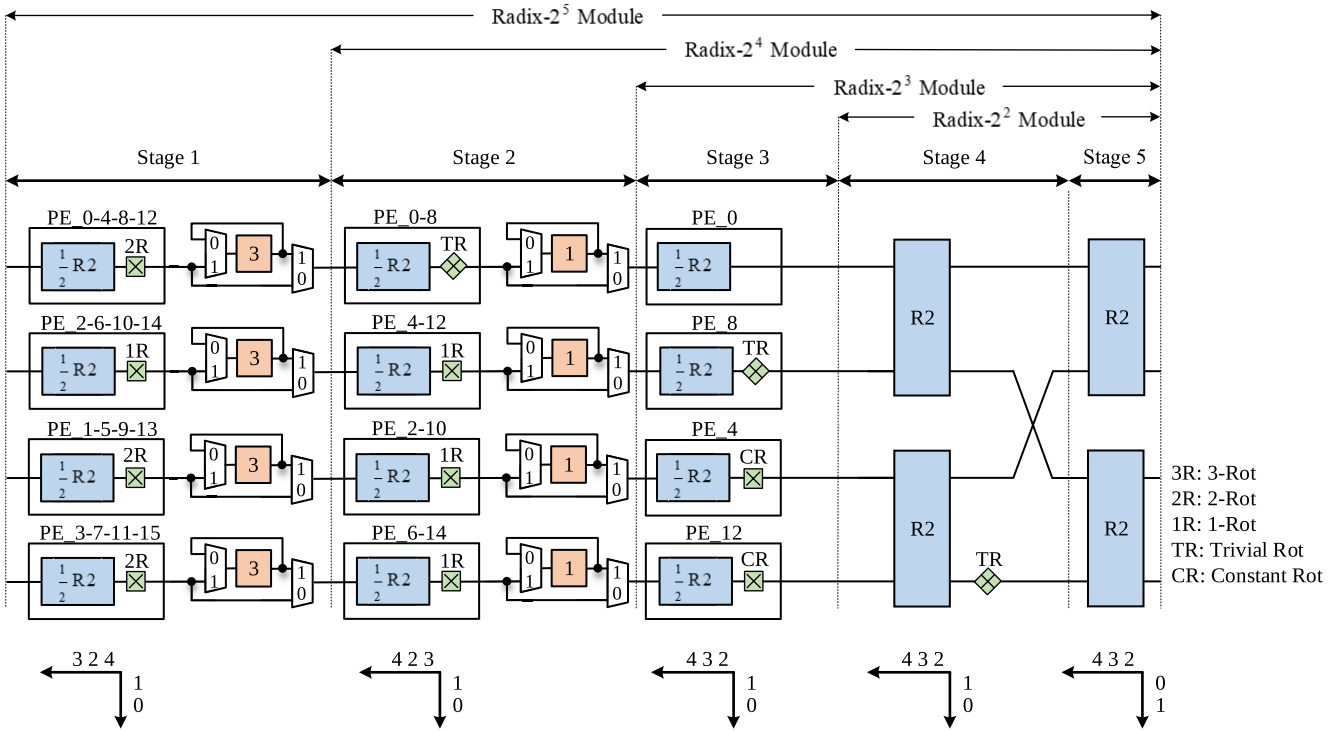

FIGURE 9. Radix-2^k modules for 4-parallel FFT architectures.

The PEs in the proposed architectures differ in the rotation angles. Table 1 shows the rotation angles for different parallel paths and stages. The case of 2-parallel data is the first one in the table. For instance, the PE at the upper path of the second stage rotates by W_{32}^{ϕ} for $\phi = 0, 4, 8, 12$, which is a 2-rot, whereas the PE at the lower path of the second stage rotates by W_{32}^{ϕ} for $\phi = 2, 6, 10, 14$, which is a 1-rot.

The general structure for the 1-rots in the PEs of the proposed MSC FFT architectures and their timing diagram are shown in Fig. 6 and Table 2, respectively. The rotator receives the real and imaginary parts of the input $x + jy$ in consecutive clock cycles. First, they are multiplied by the real and imaginary part of the coefficient $C + jS$ that represents the angle. Then, the corresponding pairs of values are added to provide the real and imaginary parts of the outputs in consecutive clock cycles. The multiplications by C and S are constant multiplications, so they are easily implemented as shift-and-add. In fact, as the input is multiplied by both C and S simultaneously, they are implemented as a multiple constant multiplication (MCM) [42], [43].

The 2-rots in the proposed architectures are similar to the 1-rots, with the only difference being that the rotator multiplies by two constants, $C_0 + jS_0$ and $C_1 + jS_1$. These multiplications create a reconfigurable MCM (RMCM) problem that is solved according to [44]. Likewise, the constant multiplications in M-rots with M larger than two are designed by solving an RMCM problem.

A special case is the rotator by W_{32}^{ϕ} with $\phi = 0, 4, 8, 12$ at the upper path of the second stage in Fig. 4. W_{32}^{0} is a rotation by 0° whose imaginary part is zero; W_{32}^{8} is a rotation by −90° whose real part is zero; and W_{32}^{4} and W_{32}^{12} are rotations by −45° and −135° whose real and imaginary parts are equal in magnitude, i.e, $|C| = |S|$. All these properties allow for simplifying the rotator to the structure in Fig. 7. It can be observed that this rotator only requires one real multiplier, which is only used for the rotations by −45° and −135°. This is possible due to the fact that the adder is moved before the multiplier, which allows for calculating $(x + y)C$ instead of $xC + yC$, and $(x - y)C$ instead of $xC - yC$.

Another special case is the trivial rotator at the upper path of the third stage of the radix-2^k modules in Fig. 4. According to Table 1, this rotator rotates by W_{32}^{0} and W_{32}^{8}, which are rotations by 0° and −90°, respectively. The corresponding circuit is shown in Fig. 8. For the rotation by 0°, the circuit simply passes the data unchanged by selecting the input 0 in the multiplexers. For the rotation by −90°, the real part of the input is first stored in the register and then the control signal of the multiplexers is set to 0. This changes the sign of the real part and, then, swaps the real and imaginary parts of the data, which calculates the desired rotation as $(x+jy)(-j) = y-jx$.

Fig. 9 shows the radix-2^k modules when the number of parallel paths is 4. The higher parallelization of these modules reduces the complexity of the shuffling circuits with respect to 2-parallel ones. Furthermore, the rotations at the same stage are distributed among all the rotators in parallel. This can be observed by comparing the rotation angles in 2-parallel and 4-parallel radix-2^k modules in Table 1. Thus, although the number of rotators increases, their complexity is reduced. For

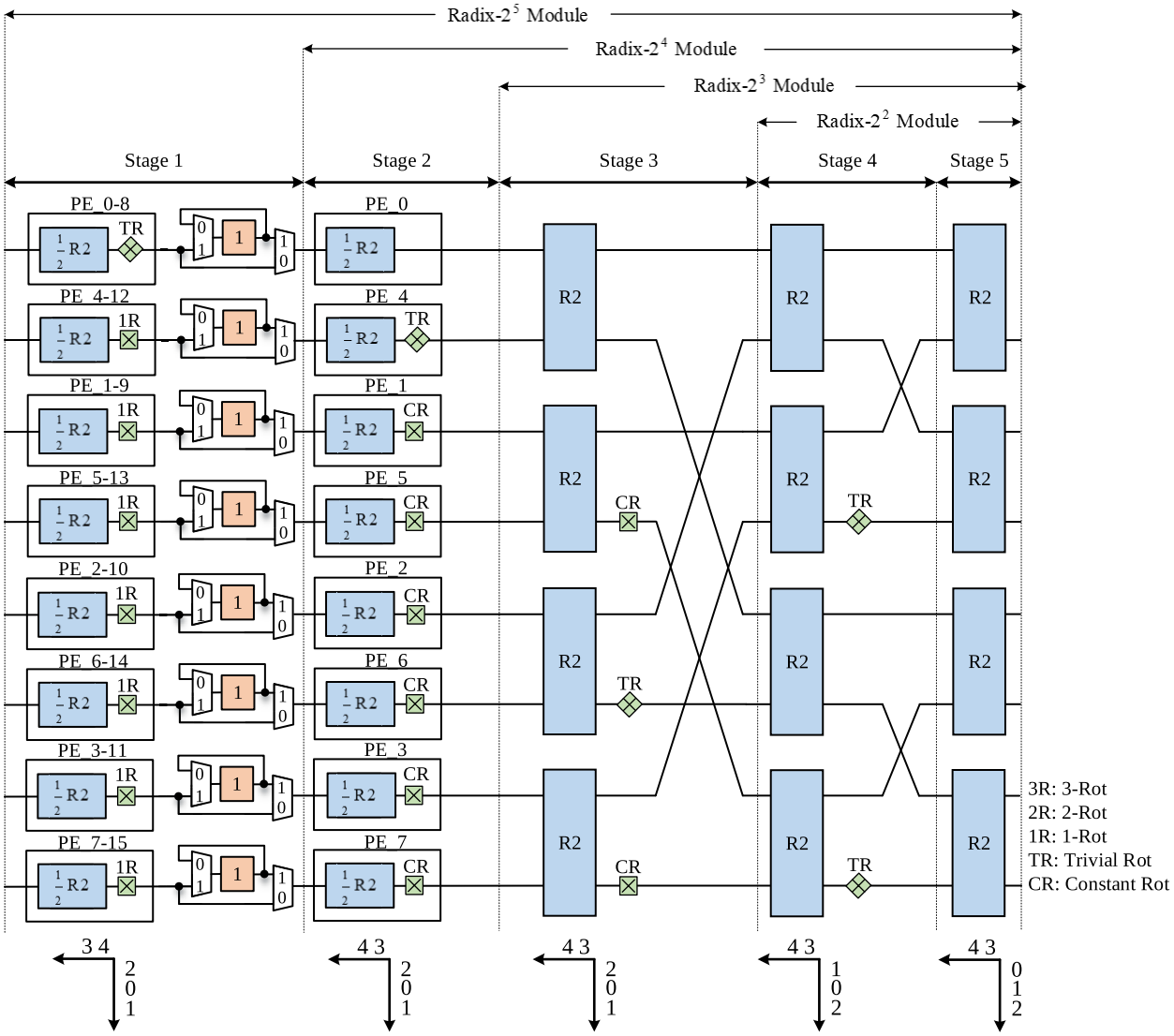

FIGURE 10. Radix-2^k modules for 8-parallel FFT architectures.

instance, the second stage of the 2-parallel radix-2^k modules requires a 2-rot and a 1-rot, whereas the second stage of the 4-parallel case requires 3 1-rots and 1 trivial rotator. Apart from these considerations, the 4-parallel radix-2^k modules are based on the same ideas presented for the 2-parallel radix-2^k modules.

For 8-parallel data, the radix-2^k modules are shown in Fig. 10. These modules reduce the complexity of the permutation circuits and the rotators even further with respect to the 4-parallel case. The exact rotation angles for each rotator are shown in Table 1. The 8-parallel radix-2^k modules are based on the same ideas presented for the 2-parallel radix-2^k modules.

IV. PROPOSED MSC FFT ARCHITECTURES

In this section, we describe the proposed MSC FFT architectures. These architectures are obtained by combining the modules shown in Section III.

For sizes smaller than or equal to 32, the proposed MSC FFT architectures correspond to the radix-2^k modules presented in Section III. Note that any of the proposed radix-2^k modules calculates an N-point FFT, where $N = 2^k$. This holds for the 2, 4, and 8-parallel modules in Figs. 4, 9 and 10.

Fig. 11 shows two architectures to calculate a 64-point 2-parallel radix-2^3 MSC FFT. They consist of two 2-parallel radix-2^3 modules like the module shown in Fig. 4, two general rotators, and permutation circuits that connect the modules. The architectures in Figs. 11(a) and 11(b) differ in the data order at the FFT stages, which is shown at the bottom of the figures. As a consequence, the lengths of the buffers in both architectures are different, although the number of components is the same. The exact EBEs carried out at the stages of the architectures are indicated under the corresponding permutation circuits. As for the 64-point 2-parallel radix-2^3 architectures in Fig. 11, FFT architectures with other sizes and radices also allow for different data orders at the

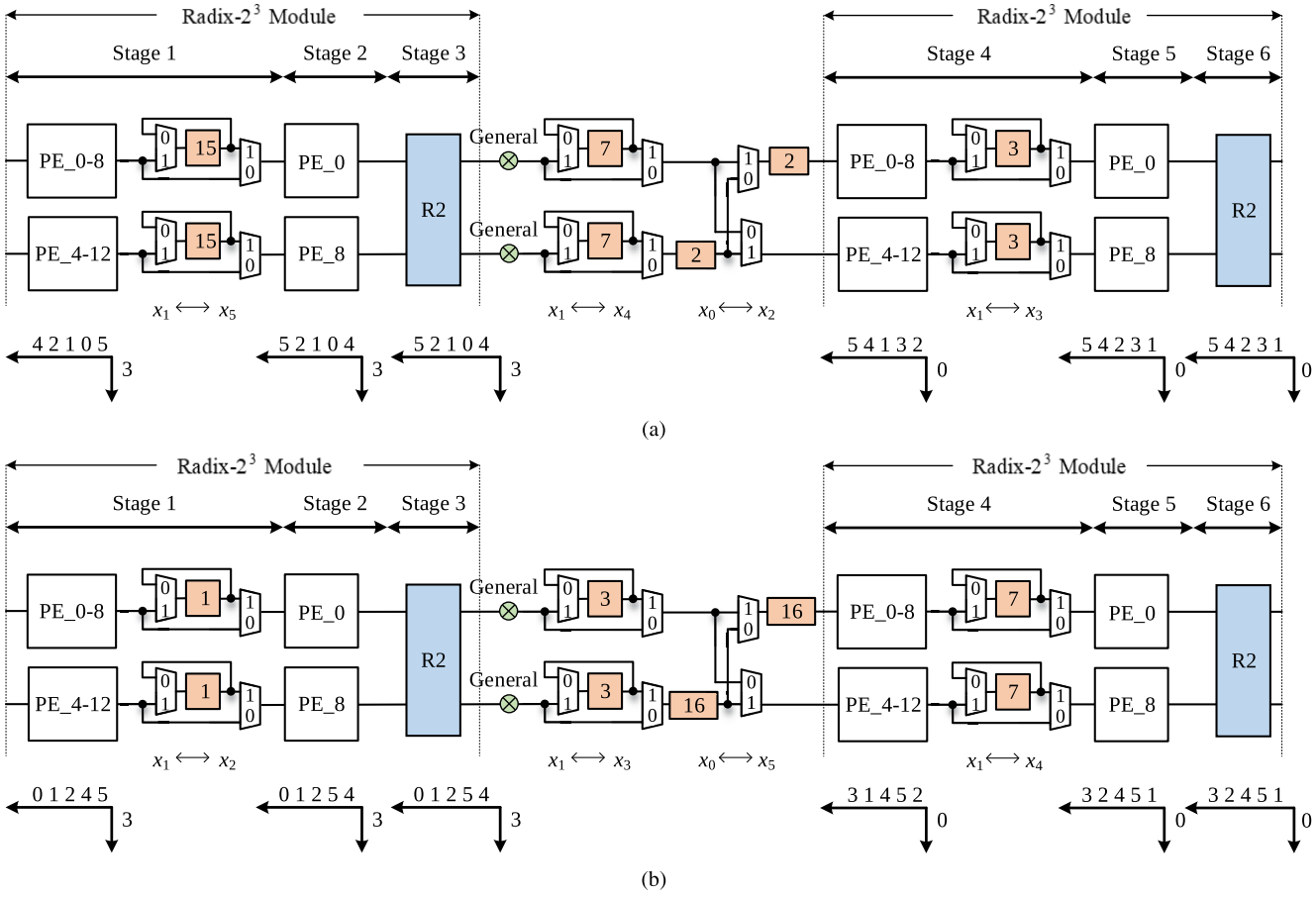

FIGURE 11. Proposed 64-point 2-parallel radix-2^3 MSC FFT architecture with different data orders.

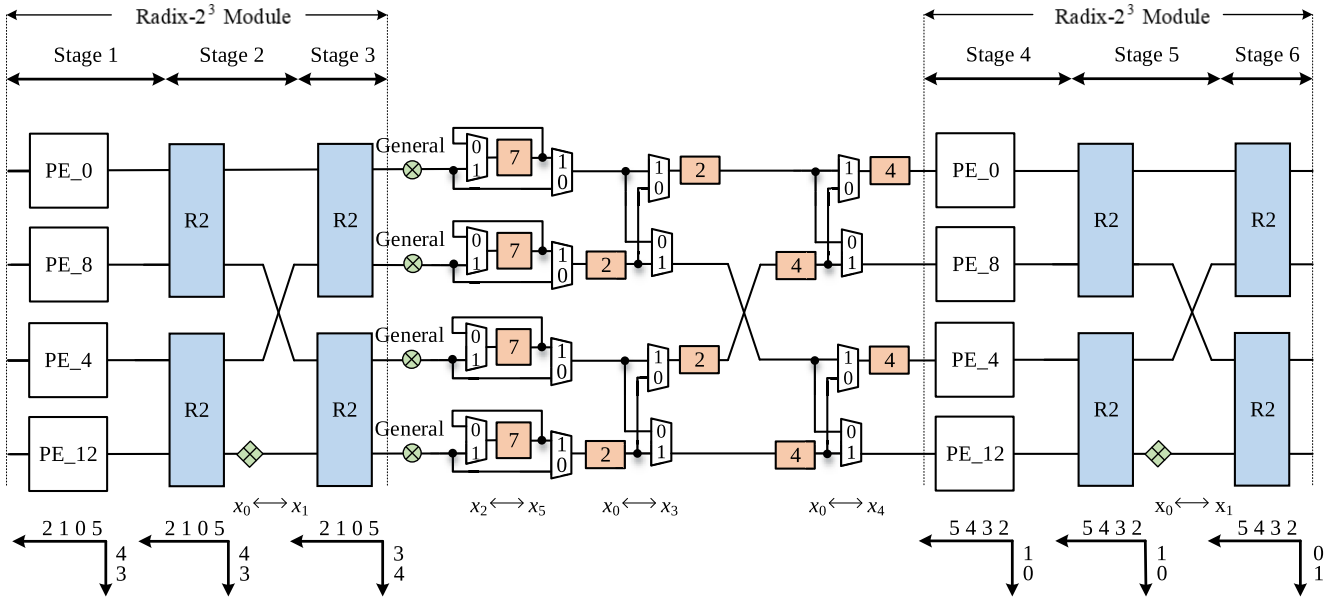

FIGURE 12. Proposed 64-point 4-parallel radix-2^3 MSC FFT architecture.

input, output, and through the FFT. In fact, the number of alternatives is large, as happens for MDC FFTs [45].

For the proposed MSC FFTs, the only considerations for the data order are:

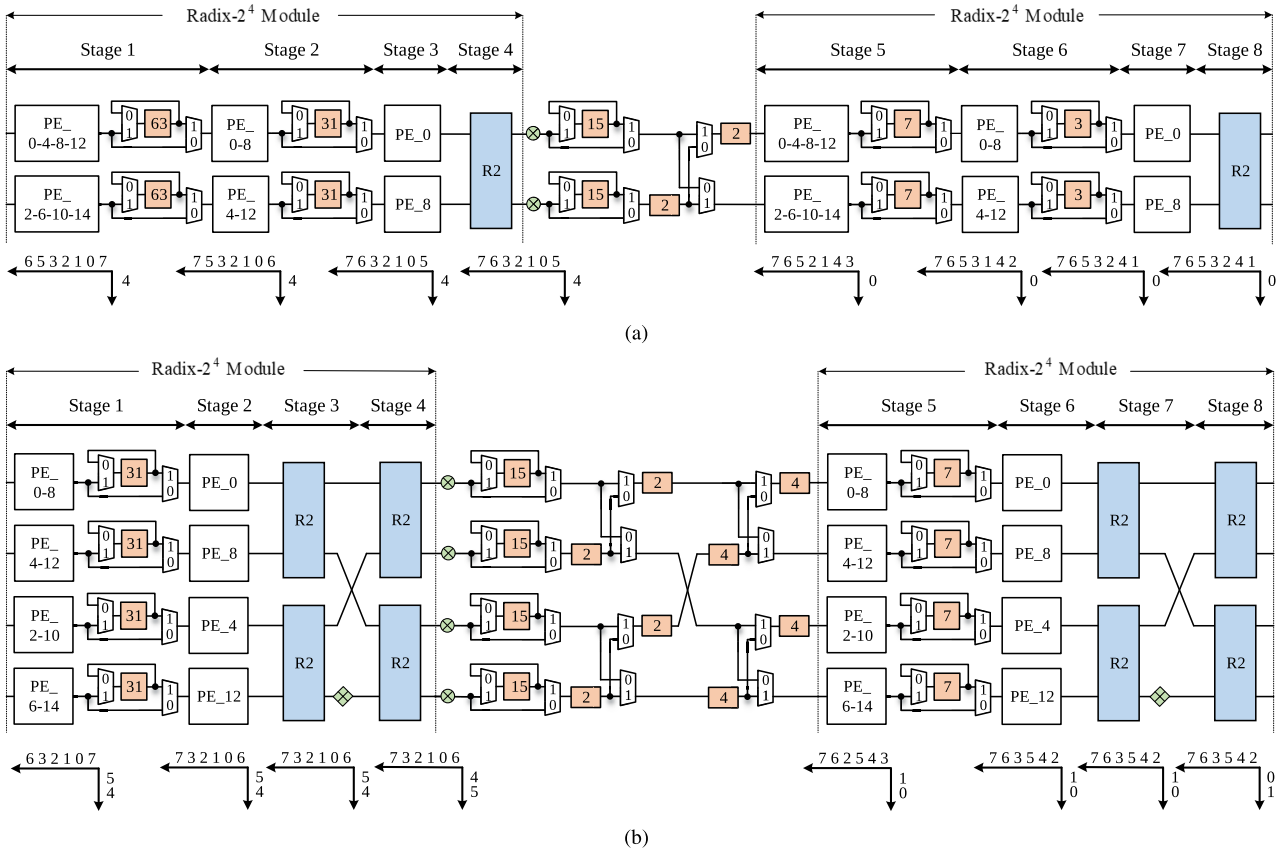

FIGURE 13. Proposed 256-point radix-2^4 MSC FFT architectures. (a) 2-parallel. (b) 4-parallel.

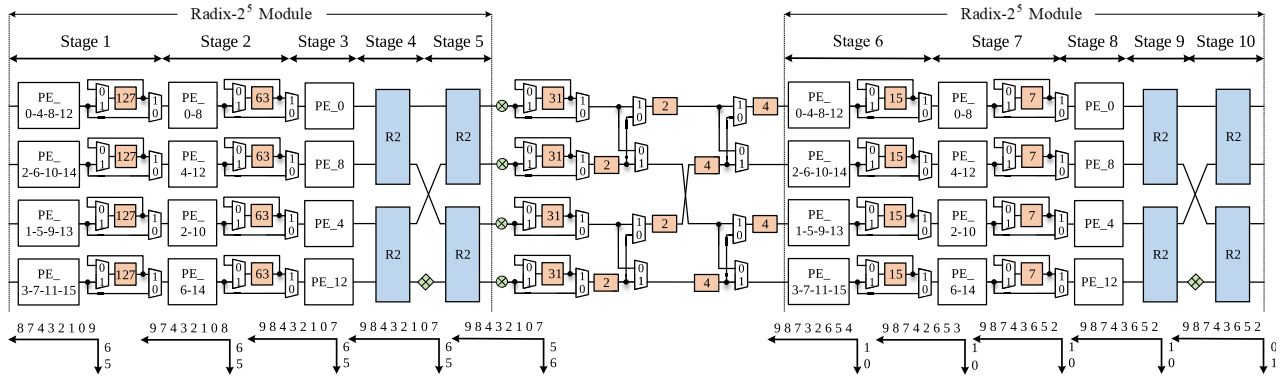

FIGURE 14. Proposed 1024-point 4-parallel radix-2^5 MSC FFT architecture.

- Bit b_{n-s} is used for dealing with the butterfly operation. This bit is placed at the smallest serial dimension, i.e., x_p, for the stages in which SC PEs are used; and in the smallest parallel dimension, i.e., x_0, for the stages where R2 butterflies are used, as they operate on two contiguous parallel paths.
- The bits b_i have been placed in serial or parallel dimensions by following the rotator allocation principle [31]. This allows for simplifying the rotators' complexity.

Once we comply with the principles described above, the rest of the bits can be arranged arbitrarily, leading to different data orders. Then, the length of the permutation circuits inside the modules is determined by the EBEs that are calculated, and the permutation circuits between the radix-2^k modules are obtained according to the methodology presented in [40].

Fig. 12 shows the proposed 64-point 4-parallel radix-2^3 MSC FFT architecture. The architecture consists of two 4-parallel radix-2^3 modules like the module shown in Fig. 9, four general rotators, and permutation circuits that connect

both modules. Compared to the 2-parallel cases, the 64-point 4-parallel radix-2^3 MSC FFT does not include serial-serial permutation circuits in the radix-2^3 modules. By contrast, the number of EBEs in between the radix-2^3 modules is larger. The data order at the FFT stages is shown at the bottom of the figure and the exact dimensions that are swapped by the permutation circuits are shown at the bottom of these circuits.

The proposed architectures can be extended to different number of parallel paths, radix, and FFT size by using the modules described in Section III and connecting them with general rotators and permutation circuits to adapt the data orders. As examples of larger MSC FFT architectures, Fig. 12 shows two 256-point radix-2^4 MSC FFTs for 2 and 4-parallel data, whereas Fig. 14 shows a 1024-point 4-parallel radix-2^5 MSC FFT architecture. Note that the 256-point MSC FFTs combine radix-2^4 modules, and the 1024-point MSC FFT combines two radix-2^5 modules.

In general, a P-parallel N-point radix-2^k MSC FFT architecture has a number of complex adders in the butterflies equal to

$$\text{Complex Adders} = P \cdot \log_2 N, \tag{6}$$

which can be derived easily since each stage has P complex adders regardless of the type of PE used for the stage. The calculation of the complex memory can be split into the memory in permutation circuits (M_{PC}) and the memory in MSC PEs (M_{PE}). For the latter, we consider 2.5 complex memory elements (5 real) per MSC PE according to Fig. 5. This leads to:

$$\text{Total Complex Memory} = M_{PC} + M_{PE} = M_{PC} + 2.5 \cdot PE, \tag{7}$$

where PE is the total number of processing elements. This results in:

$$\text{Total Complex Memory} = (M_{PC} + PE) + 1.5 \cdot PE, \tag{8}$$

where the term in parenthesis is equal to $N - P$. By substituting this value and calculating the total number of processing elements, the total memory in MSC FFT architectures can be obtained as

$$\text{Total Complex Memory} = (N - P) + \frac{3P}{2} \cdot (n - \log_2 P \cdot \lceil n/k \rceil), \tag{9}$$

where $\lceil \cdot \rceil$ is the ceiling operation, $n = \log_2 N$ is the number of stages, and $P \cdot (n - \log_2 P \cdot \lceil n/k \rceil)$ is the number of processing elements in the architecture.

The calculation of the latency is analogous to the calculation of the total memory. The latency of an MSC PE is 2, which results in

$$\text{Latency} = L_{PC} + L_{PE} = \left(L_{PC} + \frac{PE}{P}\right) + \frac{PE}{P}, \tag{10}$$

The latency in the term inside the parenthesis is $(N/P - 1)$, leading to a total latency of

$$\text{Latency (cycles)} = (\frac{N}{P} - 1) + (n - \log_2 P \cdot \lceil n/k \rceil). \tag{11}$$

The throughput of the proposed architectures is equal to the number of parallel data of the architecture, i.e.,

$$\text{Throughput (samples/cycle)} = P. \tag{12}$$

V. COMPARISON AND IMPLEMENTATION RESULTS

Table 3 compares the proposed architectures with previous radix-2^k pipelined FFTs. The table is divided into architectures for 2, 4, and 8 parallel paths. The first two columns show the type and the radix of the architectures. The third to seventh columns show the number of general rotators, 3-rots, 2-rots, 1-rots, and constant rotators. Columns eight and nine show the number of complex adders in butterflies and the total memory size considering complex data. Finally, the last columns show the performance in terms of latency in cycles and throughput in samples per cycle (S/c).

Regarding rotators, the proposed architectures reduce the total number of rotators and their complexity. Whereas previous 2-parallel radix-2^4 FFT architectures require $2\lfloor n/4 \rfloor$ or more non-general rotators, the proposed 2-parallel radix-2^4 architecture needs only a total of $1.5\lfloor n/4 \rfloor$. This represents a saving of 25% in terms of non-general rotators. Regarding the complexity of non-general roators, the proposed 2-parallel radix-2^4 MSC FFT architecture uses 2-rots and 1-rots, but no 3-rot. Conversely, previous 2-parallel radix-2^4 FFT architectures use 3-rots [26], [30], [31] and in some cases also 2-rots [31].

The simultaneous reduction in the number of non-general rotators and their complexity can also be observed by comparing 4-parallel radix-2^4 FFT architectures. The number of non-general rotators is $4\lfloor n/4 \rfloor$ in [2] and [25], and $3\lfloor n/4 \rfloor$ in [30] and [31]. Conversely, in the proposed radix-2^4 4-parallel MSC FFT the total number of non-general rotators is approximately $2.5\lfloor n/4 \rfloor$, which represents a saving of 17% in terms of non-general rotators. Additionally, the non-general rotators in the proposed radix-2^4 4-parallel MSC FFT have lower complexity than those in previous 4-parallel FFTs. The architectures in [2], [25], and [30] use 3-rots for non-general rotators, the architecture in [31] uses 2-rots and 1-rot, and the proposed architecture uses 1-rots and CRs, which are the simplest ones.

The proposed 8-parallel radix-2^4 MSC FFT also reduces the number and complexity of the rotators with respect to previous works. In the proposed radix-2^4 8-parallel MSC FFT, the number of non-general rotators is approximately $5\lfloor n/4 \rfloor$, and all of them are CRs. This improves [27] and [30], which use $8\lfloor n/4 \rfloor$ and $6\lfloor n/4 \rfloor$ 3-rots, respectively. Compared to [31], the proposed radix-2^4 FFT reduces the complexity of the rotators, as it only requires CRs, while the number of rotators is similar in both designs.

The proposed radix-2^4 and radix-2^5 MSC architectures are also compared in Table 3. It can be observed that the number of general rotators in the proposed radix-2^5 FFTs depends on $\lceil n/5 \rceil$, whereas for radix-2^4 they depend on $\lceil n/4 \rceil$. By contrast, radix-2^5 FFTs include a larger number of non-general rotators. Thus, the selection between radix-2^4 or

TABLE 3. Comparison of radix-2^k pipelined FFT architectures for an N-point FFT.

PIPELINED ARCHITECTURE		AREA							PERFORMANCE	
		Rotators					Complex	Complex	Latency	Th.
Type	Radix	GR	3-rot	2-rot	1-rot	CR	Adders	Memory	(cycles)	(S/c)
2-PARALLEL ARCHITECTURES										
MDF [24]	2	$2\lceil n/2\rceil$	0	0	0	0	$2n$	$2N$-1 ††	$3N/2+3n-1$ ††	2
MDF [1]	2^2	$2\lceil n/2\rceil-2$	0	0	0	0	$4n$	N	$N/2$	2
MDF [26]	2^4	$2\lceil n/4\rceil-2$	$2\lfloor n/4\rfloor$	0	0	0	$4n$	N	$N/2$	2
MDC [30]	2^4	$2\lceil n/4\rceil-2$	$2\lfloor n/4\rfloor$	0	0	0	$2n$	N	$N/2$	2
MDC [31]	2^4	$2\lceil n/4\rceil-2$	$\lfloor n/4\rfloor$	$\lfloor (n+1)/4\rfloor$	0	0	$2n$	N	$N/2$	2
Proposed	2^2	$2\lceil n/2\rceil-2$	0	0	0	0	$2n$	$N-2+3(n-\lceil n/2\rceil)$	$N/2-1+n-\lceil n/2\rceil$	2
	2^3	$2\lceil n/3\rceil-2$	0	0	$0.5\lfloor n/3\rfloor$	0	$2n$	$N-2+3(n-\lceil n/3\rceil)$	$N/2-1+n-\lceil n/3\rceil$	2
	2^4	$2\lceil n/4\rceil-2$	0	$0.5\lfloor n/4\rfloor$	$\lfloor n/4\rfloor$ $+(0.5)^{\triangle}$	0	$2n$	$N-2+3(n-\lceil n/4\rceil)$	$N/2-1+n-\lceil n/4\rceil$	2
	2^5	$2\lceil n/5\rceil-2$	$0.5\lfloor n/5\rfloor$	$\lfloor n/5\rfloor$ $+(0.5)^{*}$	$\lfloor (n+1)/5\rfloor$ $+(0.5)^{\dagger}$	0	$2n$	$N-2+3(n-\lceil n/5\rceil)$	$N/2-1+n-\lceil n/5\rceil$	2
4-PARALLEL ARCHITECTURES										
MDC [35]	4	$3\lceil n/2\rceil-3$	0	0	0	0	$4n$	$8N/3$	$N/3$	4
MDF [2]	2^4	$4\lceil n/4\rceil-4$	$4\lfloor n/4\rfloor$	0	0	0	$8n$	N	$N/4$	4
MDF [25]	2^4	$4\lceil n/4\rceil-4$	$4\lfloor n/4\rfloor$	0	0	0	$8n$	N	$N/4$	4
MDC [30]	2^4	$4\lceil n/4\rceil-4$	$3\lfloor n/4\rfloor$	0	0	0	$4n$	N	$N/4$	4
MDC [31]	2^4	$4\lceil n/4\rceil-4$	0	$\lfloor n/4\rfloor$	$2\lfloor n/4\rfloor$ $+(1)^{\triangle}$	0	$4n$	N	$N/4$	4
Proposed	2^3	$4\lceil n/3\rceil-4$	0	0	0	$\lfloor n/3\rfloor$	$4n$	$N-4+6(n-2\lceil n/3\rceil)$	$N/4-1+n-2\lceil n/3\rceil$	4
	2^4	$4\lceil n/4\rceil-4$	0	0	$1.5\lfloor n/4\rfloor$	$\lfloor (n+1)/4\rfloor$	$4n$	$N-4+6(n-2\lceil n/4\rceil)$	$N/4-1+n-2\lceil n/4\rceil$	4
	2^5	$4\lceil n/5\rceil-4$	0	$1.5\lfloor n/5\rfloor$	$2\lfloor n/5\rfloor$ $+(1.5)^{*}$	$\lfloor (n+2)/5\rfloor$	$4n$	$N-4+6(n-2\lceil n/5\rceil)$	$N/4-1+n-2\lceil n/5\rceil$	4
8-PARALLEL ARCHITECTURES										
MDF [27]	2^4	$8\lceil n/4\rceil-8$	$8\lfloor n/4\rfloor$	0	0	0	$16n$	N	$N/8$	8
MDC [30]	2^4	$8\lceil n/4\rceil-8$	$6\lfloor n/4\rfloor$	0	0	0	$8n$	N	$N/8$	8
MDC [31]	2^4	$8\lceil n/4\rceil-8$	0	0	$3\lfloor n/4\rfloor$	$2\lfloor (n+1)/4\rfloor$	$8n$	N	$N/8$	8
Proposed	2^4	$8\lceil n/4\rceil-8$	0	0	0	$5\lfloor n/4\rfloor$ $+(2)^{\triangle}$	$8n$	$N-8+12(n-3\lceil n/4\rceil)$	$N/8-1+n-3\lceil n/4\rceil$	8
	2^5	$8\lceil n/5\rceil-8$	0	0	$3.5\lfloor n/5\rfloor$	$5\lfloor (n+1)/5\rfloor$ $+(2)^{\dagger}$	$8n$	$N-8+12(n-3\lceil n/5\rceil)$	$N/8-1+n-3\lceil n/5\rceil$	8

$\triangle$: When (n mod 4)=3. $*$: When (n mod 5)=4. †: When (n mod 5)=3. ††: The increases in memory and latency are justified by the processing of two FFT channels.

TABLE 4. Comparison of the rotators complexity in 4-Parallel 128-Point MSC FFTs.

FFT Architecture	Rotators				
	GR	2-rot	1-rot	CR	Total
[36]	4	4	3	2	13
Proposed	4	0	3	4	11

radix-2^5 mostly depends on the FFT size. For instance, for $N=1024$ points, $\lceil n/5\rceil=2$ and $\lceil n/4\rceil=3$. This results in fewer general rotators for radix-2^5, being this radix preferable for this FFT size. Conversely, for $N=4096$, $\lceil n/5\rceil=3$ and $\lceil n/4\rceil=3$, so radix-2^4 is preferred, as it requires the same number of general rotators and less non-general rotators than radix-2^5.

Regarding complex adders in butterflies, the proposed architectures achieve the theoretical minimum number of adders, which is also achieved in other previous designs. This theoretical minimum corresponds to one adder per parallel branch in each stage and is due to the fact that, at each stage, all branches in the flow graph include one adder. As a consequence, data flowing through any branch of the architecture must encounter at least one adder at each stage.

The complex memory and the latency of the proposed architectures increase slightly with respect to previous approaches. This is the price that the proposed architectures must pay in order to achieve the improvement in the number of rotators and their complexity. However, the increase in memory and latency is small and it is almost negligible for medium-size and large FFTs. For instance, the proposed 2-parallel radix-2^4 MSC FFT for 1024 points increases the memory from 1024 to 1043, which is only 1.8 %, and the latency from 512 to 518, which is an increase of 1.1 %. These increases are small compared to the reduction of up to 25% in the number of rotators and their complexity.

Finally, for all architectures in the table, the throughput in terms of samples per cycle only depends on the parallelization and is equal to P.

As a conclusion, compared to the state-of-the-art, the proposed architectures are the most efficient ones in terms of rotators. They reduce both the number and also the complexity of non-general rotators with respect to previous designs. The number of non-general rotators is reduced up to 25% and the complexity is reduced by using simpler M-rots than in previous approaches. This improvement comes at the cost of slight increases in latency and memory, which are small

TABLE 5. Comparison of parallel pipelined architectures on ASICs.

	[33]	[34]	[28]	[7]				Proposed
Architecture	MDC	MDC	MDF	MDC				MSC
FFT Size (N)	1024	1024	1024	2048	1024	512	128	1024
Radix	Radix-4	Radix-2^2	Mixed radix	Radix-4/8				Radix-2^5
Parallel Streams (P)	8	2	4	4				4
Word Length (WL)	16	In: 32 / Out:40	16	In:8 / Out:12				16
Technology (nm)	65	65	65	90				40
Voltage (V)	0.6	0.6	0.45	1.0				0.9
Clock Rate (MHz)	290	400	20	40				330
Throughput (GS/s)	2.32	0.80	0.08	0.16				1.32
Core Area (mm^2)	8.30	3.60	3.34	3.10				0.18
Power (mW)	82.00	60.30	13.83	63.72	62.92	57.51	51.69	30.90
@ throughput rate	@ 2.32 GS/s	@ 0.80 GS/s	@ 0.08 GS/s	@ 0.16 GS/s				@ 1.32 GS/s
Normalized Area (mm^2)	1.57	1.21	1.26	0.38				0.18
Normalized Power (mW)	64.60	68.87	561.66	118.28	116.78	106.73	95.95	30.90
@ throughput rate	@ 1.32 GS/s	@ 1.32 GS/s	@ 1.32 GS/s	@ 1.32 GS/s				@ 1.32 GS/s
SQNR (dB)	NP	NP	NP	NP				40.58

NP: Not provided.

compared to the savings in the number and complexity of the rotators.

In order to show the improvement of the proposed approach with respect to [36], Table 4 compares the rotators' complexity in 128-point MSC architectures. The table divides the rotators into general rotators, 2-rots, 1-rots, and constant rotators. The last column of the table is the total number of non-trivial rotators in each approach. The architecture in [36] includes 4 2-rots in the first stage, 4 GR in stage 3, 3 1-rots in stage 4, and 2 constant rotators in stage 5, leading to a total of 13 non-trivial rotators. To calculate a 128-point FFT, the proposed approach combines the radix-2^3 and radix-2^4 modules in Fig. 9. The former includes 2 constant rotators, whereas the latter uses 3 1-rots and 2 constant rotators. The connection of these modules requires 4 general rotators, leading to a total of 11 non-trivial rotators. As a result, the proposed approach reduces both the total number of rotators and their complexity with respect to [36].

To validate the architectures and compare them with the implementations in previous works, a test chip for the 4-parallel 1024-point radix-2^5 MSC FFT in Fig. 14 has been designed by using the TSMC 40 nm CMOS Logic technology. The architecture has a word length of 16 bits for each real and imaginary part of the data. Table 5 shows the experimental results and compares them to previous approaches on ASICs. For a fair comparison, the table only considers post-layout results.

The experimental results for the proposed architecture show that the core size of the test chip is 0.18 mm^2, the throughput is 1.32 GS/s at a clock frequency of 330 MHz, and the power consumption at this high rate is only 30.9 mW.

In order to make results from different technologies comparable, the area and power consumption of the architectures have been normalized according to

$$A_N\ [\text{mm}^2] = \frac{\text{Area}\ [\text{mm}^2]}{\left(\frac{\text{Tech.}\ [\text{nm}]}{40}\right)^2 \cdot \frac{WL}{16} \cdot \frac{P}{4}}, \tag{13}$$

where A_N denotes the normalized area with respect to the technology process (Tech.), word length (WL), and number of parallel branches. Likewise, the power is normalized as

$$P_N\ [\text{mW}] = \frac{\text{Power}\ [\text{mW}]}{\frac{\text{Tech.}\ [\text{nm}]}{40} \cdot \frac{\text{Th.}\ [\text{GS/s}]}{1.32} \cdot \frac{WL}{16} \cdot \left(\frac{\text{V}\ [\text{V}]}{0.9}\right)^2}, \tag{14}$$

where P_N denotes the normalized power consumption according to the technology process, throughput (Th.), word length, and working voltage (V). For the normalization of [7] and [34], we have considered their average word lengths of 36 and 10 bits, respectively.

In Table 5, it can be observed that the proposed architecture is the first MSC FFT among 1024-point FFTs on ASICs. Among them, it is the only one that uses radix-2^5, being this radix the most efficient one in terms of rotations. Regarding area and power, the proposed architecture reduces the normalized area and power consumption by more than 50% with respect to previous approaches.

VI. CONCLUSION

In this paper, we have proposed a new family of multi-path SC FFT architectures for 2, 4, and 8 parallel data, radix-2^k, and any power-of-two size. The proposed architectures consist of efficient radix-2^k modules that are connected by permutation circuits. In this paper, we have proposed a new family of multi-path SC FFT architectures for 2, 4, and 8 parallel data,

radix-2^k, and any power-of-two size. The proposed architectures consist of efficient radix-2^k modules that are connected by permutation circuits.

Compared to previous approaches, the proposed MSC FFT architectures reduce the number of non-general rotators up to 25% and their complexity. They also have a slight increase in data memory and latency, which is negligible for large FFTs.

Finally, implementation results show that the proposed radix-2^5 4-parallel 1024-point MSC FFT architecture achieves a throughput of 1.32 GS/s. The FFT core has an area of 0.18 mm^2 and a power consumption of 30.9 mW. The normalized values of area and power are significantly smaller than the values reported in previous approaches.

REFERENCES

[1] N. Li and N. P. van der Meijs, "A radix 2^2 based parallel pipeline FFT processor for MB-OFDM UWB system," in *Proc. IEEE Int. SOC Conf. (SOCC)*, Sep. 2009, pp. 383–386.

[2] H. Liu and H. Lee, "A high performance four-parallel 128/64-point radix-2^4 FFT/IFFT processor for MIMO-OFDM systems," in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, Dec. 2008, pp. 834–837.

[3] L. Liu, J. Ren, X. Wang, and F. Ye, "Design of low-power, 1GS/s throughput FFT processor for MIMO-OFDM UWB communication system," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2007, pp. 2594–2597.

[4] S. Liu and D. Liu, "A high-flexible low-latency memory-based FFT processor for 4G, WLAN, and future 5G," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 3, pp. 511–523, Mar. 2019.

[5] J. Wang, C. Xiong, K. Zhang, and J. Wei, "A mixed-decimation MDF architecture for radix-2^k parallel FFT," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 1, pp. 67–78, Jan. 2016.

[6] C.-H. Yang, T.-H. Yu, and D. Markovic, "Power and area minimization of reconfigurable FFT processors: A 3GPP-LTE example," *IEEE J. Solid-State Circuits*, vol. 47, no. 3, pp. 757–768, Mar. 2012.

[7] K.-J. Yang, S.-H. Tsai, and G. C. H. Chuang, "MDC FFT/IFFT processor with variable length for MIMO-OFDM systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 4, pp. 720–731, Apr. 2013.

[8] T. Kamazaki, S. K. Okumura, Y. Chikada, T. Okuda, Y. Kurono, S. Iguchi, S. Mitsuishi, Y. Murakami, N. Nishimuta, H. Mita, and R. Sano, "Digital spectro-correlator system for the Atacama Compact Array of the Atacama Large Millimeter/submillimeter Array," *Publications Astronomical Soc. Jpn.*, vol. 64, no. 2, p. 29, Apr. 2012.

[9] H. Kanders, T. Mellqvist, M. Garrido, K. Palmkvist, and O. Gustafsson, "A 1 million-point FFT on a single FPGA," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 10, pp. 3863–3873, Oct. 2019.

[10] L. Li and A. M. Wyrwicz, "Parallel 2D FFT implementation on FPGA suitable for real-time MR image processing," *Rev. Sci. Instrum.*, vol. 89, no. 9, pp. 1–9, Sep. 2018.

[11] Z. Kaya, M. Garrido, and J. Takala, "Memory-based FFT architecture with optimized number of multiplexers and memory usage," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 8, pp. 3084–3088, Aug. 2023.

[12] Y. Guo, Z. Wang, Q. Hong, H. Luo, X. Qiu, and L. Liang, "A 60-mode high-throughput parallel-processing FFT processor for 5G/4G applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 31, no. 2, pp. 219–232, Feb. 2023.

[13] M. Garrido, "A survey on pipelined FFT hardware architectures," *J. Signal Process. Syst.*, vol. 94, no. 11, pp. 1345–1364, Nov. 2022.

[14] M. Garrido, R. Andersson, F. Qureshi, and O. Gustafsson, "Multiplierless unity-gain SDF FFTs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 9, pp. 3003–3007, Sep. 2016.

[15] S. He and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Proc. IEEE Custom Integr. Circuits Conf.*, May 1998, pp. 131–134.

[16] Y.-N. Chang, "An efficient VLSI architecture for normal I/O order pipeline FFT design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 12, pp. 1234–1238, Dec. 2008.

[17] X. Liu, F. Yu, and Z.-K. Wang, "A pipelined architecture for normal I/O order FFT," *J. Zhejiang Univ. Sci. C*, vol. 12, no. 1, pp. 76–82, Jan. 2011.

[18] M. Garrido, S.-J. Huang, S.-G. Chen, and O. Gustafsson, "The serial commutator (SC) FFT," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 63, no. 10, pp. 974–978, Oct. 2016.

[19] M. Garrido, N. K. Unnikrishnan, and K. K. Parhi, "A serial commutator fast Fourier transform architecture for real-valued signals," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 65, no. 11, pp. 1693–1697, Nov. 2018.

[20] N. K. Unnikrishnan, M. Garrido, and K. K. Parhi, "Effect of finite word-length on SQNR, area and power for real-valued serial FFT," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2019, pp. 1–5.

[21] H. Fang, B. Zhang, F. Yu, B. Zhao, and Z. Ma, "A pipelined algorithm and area-efficient architecture for serial real-valued FFT," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 11, pp. 4533–4537, Nov. 2022.

[22] S. Park and D. Jeon, "A modified serial commutator architecture for real-valued fast Fourier transform," in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Oct. 2020, pp. 1–6.

[23] J. Hazarika, M. T. Khan, S. R. Ahamed, and H. B. Nemade, "Energy efficient VLSI architecture of real-valued serial pipelined FFT," *IET Comp. Digital Tech.*, vol. 13, no. 6, pp. 461–469, Jun. 2019.

[24] X. Zhou, X. Chen, Y. He, and X. Mou, "A flexible-channel MDF architecture for pipelined radix-2 FFT," *IEEE Access*, vol. 11, pp. 38023–38033, Apr. 2023.

[25] S.-I. Cho, K.-M. Kang, and S.-S. Choi, "Implemention of 128-point fast Fourier transform processor for UWB systems," in *Proc. Int. Wireless Commun. Mobile Comput. Conf.*, Aug. 2008, pp. 210–213.

[26] J. Lee, H. Lee, S.-i. Cho, and S.-S. Choi, "A high-speed, low-complexity radix-2^4 FFT processor for MB-OFDM UWB systems," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2006, p. 4.

[27] S.-N. Tang, J.-W. Tsai, and T.-Y. Chang, "A 2.4-GS/s FFT processor for OFDM-based WPAN applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 6, pp. 451–455, Jun. 2010.

[28] C.-H. Yang, T.-H. Yu, and D. Markovic, "A 5.8 mW 3GPP-LTE compliant 8×8 MIMO sphere decoder chip with soft-outputs," in *Proc. Symp. VLSI Circuits*, Jun. 2010, pp. 209–210.

[29] T. Ahmed, "A low-power time-interleaved 128-point FFT for IEEE 802.15.3c standard," in *Proc. Int. Conf. Inform., Electron. Vis. (ICIEV)*, May 2013, pp. 1–5.

[30] M. Garrido, J. Grajal, M. A. Sánchez, and O. Gustafsson, "Pipelined radix-2^k feedforward FFT architectures," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 1, pp. 23–32, Jan. 2013.

[31] M. Garrido, S.-J. Huang, and S.-G. Chen, "Feedforward FFT hardware architectures based on rotator allocation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 2, pp. 581–592, Feb. 2018.

[32] M. Garrido and P. Malagón, "The constant multiplier FFT," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 1, pp. 322–335, Jan. 2021.

[33] D. Jeon, M. Seok, C. Chakrabarti, D. Blaauw, and D. Sylvester, "A super-pipelined energy efficient subthreshold 240 MS/s FFT core in 65 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 47, no. 1, pp. 23–34, Jan. 2012.

[34] N. Le Ba and T. T. Kim, "An area efficient 1024-point low power radix-2^2 FFT processor with feed-forward multiple delay commutators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 10, pp. 3291–3299, Oct. 2018.

[35] M. A. Sanchez, M. Garrido, M. Lopez-Vallejo, and J. Grajal, "Implementing FFT-based digital channelized receivers on FPGA platforms," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 44, no. 4, pp. 1567–1585, Oct. 2008.

[36] S.-C. Hsu, S.-J. Huang, S.-G. Chen, S.-C. Lin, and M. Garrido, "A 128-point multi-path SC FFT architecture," in *Proc. IEEE Int. Symp. Circuits Syst.*, Oct. 2020, pp. 1–5.

[37] M. Garrido, "A new representation of FFT algorithms using triangular matrices," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 63, no. 10, pp. 1737–1745, Oct. 2016.

[38] R. Andersson, "FFT hardware architectures with reduced twiddle factor sets," M.S. thesis, Dept. Elect. Eng., Linköping Univ., Linköping, Sweden, Jun. 2014.

[39] R. Andersson and M. Garrido, "Using rotator transformations to simplify FFT hardware architectures," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 12, pp. 4784–4793, Dec. 2020.

[40] M. Garrido, J. Grajal, and O. Gustafsson, "Optimum circuits for bit-dimension permutations," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 5, pp. 1148–1160, May 2019.

[41] M. Garrido, F. Qureshi, J. Takala, and O. Gustafsson, "Hardware architectures for the fast Fourier transform," in *Handbook of Signal Processing Systems*, 3rd ed., S. S. Bhattacharyya, E. F. Deprettere, R. Leupers, and J. Takala, Eds. Cham, Switzerland: Springer, 2019.

[42] O. Gustafsson, "A difference based adder graph heuristic for multiple constant multiplication problems," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2007, pp. 1097–1100.
[43] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Trans. Algorithms*, vol. 3, no. 2, pp. 1–39, May 2007.
[44] K. Möller, M. Kumm, M. Kleinlein, and P. Zipf, "Reconfigurable constant multiplication for FPGAs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 6, pp. 927–937, Jun. 2017.
[45] M. Garrido and P. Paz, "Optimum MDC FFT hardware architectures in terms of delays and multiplexers," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 3, pp. 1003–1007, Mar. 2021.

GUANG-TING DENG received the B.S. degree from the Department of Electrical Engineering, National Central University (NCU), in 2019, and the M.S. degree in electronics from National Yang Ming Chiao Tung University (NYCU), in 2021. His research interests include baseband signal processing and fast Fourier transform (FFT) circuit implementation.

MARIO GARRIDO (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering from Universidad Politécnica de Madrid (UPM), Spain, in 2004 and 2009, respectively.

In 2010, he moved to Sweden to work as a Postdoctoral Researcher with the Department of Electrical Engineering, Linköping University. From 2012 to 2019, he was an Associate Professor with the Department of Electrical Engineering. In 2019, he moved back to UPM, where he holds a Ramón y Cajal Research Fellowship. He was the author of more than 50 scientific publications, and he appeared in the "World's Top 2% Scientists List" elaborated by Standford University, in 2022. His research interest includes optimized hardware design for signal-processing applications. This includes the design of hardware architectures for the fast Fourier transform (FFT), circuits for data management, the CORDIC algorithm, neural networks, and circuits to calculate statistical and mathematical operations. His research covers high-performance circuits for real-time computation, and designs for small area and low power consumption.

SAU-GEE CHEN received the B.S. degree from National Tsing Hua University, Taiwan, in 1978, and the M.S. and Ph.D. degrees in electrical engineering from the State University of New York, Buffalo, NY, USA, in 1984 and 1988, respectively. He is currently the Director of the Board of ICP DAS Technology. Prior to that, he was a Professor with the Department of Electronics Engineering, National Yang Ming Chiao Tung University (NYCU), Taiwan, where the submitted work was done. He was the Director of Honors Program, College of Electrical and Computer Engineering/College of Computer Science, NYCU, from 2011 to 2012. He was the Department Chair of the Department of Electronics Engineering, NYCU, from 2012 to 2015. He was the Associate Dean of the Office of International Affairs, from March 2011 to July 2011, and the Director of the Institute of Electronics, from 2003 to 2006, NYCU. He has published more than 100 conference and journal papers, and holds 20 U.S. and Taiwan patents. His research interests include digital communication, multi-media computing, digital signal processing, and VLSI signal processing. He was a member of the Board of Governor and IEEE Taipei Section, from 2013 to 2020. He was the Coordinator of IEEE VTS Asia–Pacific Chapters, from 2014 to 2016. He was the Chair of the IEEE Vehicular Technology Society and Taipei Chapter, from 2012 to 2013. From 2004 to 2006, he served as an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS.

SHEN-JUI HUANG received the M.S. degree from the Department of Electrical Engineering, National Taiwan University, in 1997, and the Ph.D. degree in electronics from National Yang Ming Chiao Tung University, in 2012. His research interests include baseband signal processing and circuit implementation.