A Normal I/O Order Radix-2 FFT Architecture to Process Twin Data Streams for MIMO

A Normal I/O Order Radix-2 FFT Architecture to Process Twin Data Streams for MIMO

Antony Xavier Glittas, Mathini Sellathurai, *Senior Member, IEEE* and G Lakshminarayanan

***Abstract*—Nowadays, many applications require simultaneous computation of multiple independent FFT (fast Fourier transform) operations with their outputs in natural order. Therefore, this paper presents a novel pipelined FFT processor for FFT computation of two independent data streams. The proposed architecture is based on MDC (multi-path delay commutator) FFT architecture. It has an $N/2$-point DIT FFT and $N/2$-point DIF FFT to process the odd and even samples of two data streams separately. The main feature of the architecture is that the bit reversal operation is performed by the architecture itself so the outputs are generated in normal order without any dedicated bit reversal circuit. The bit reversal operation is performed by the shift registers in the FFT architecture by interleaving the data. Therefore, the proposed architecture requires less number of registers and has high throughput.**

***Index Terms*—Fast Fourier transform, Multi-path delay commutator FFT, Bit reversed order, Normal order, Bit reversal.**

I. Introduction

Fast Fourier transform is one of the most commonly used operation in the wireless communication applications such as OFDM (Orthogonal frequency division multiple accesses), UWB (Ultra wide band), DVB-T and signal processing application as well. A family of pipelined FFT architectures is discussed in [1] in which SDF (Single path delay feed-back) and MDC (multi-path delay commutator) are very popular. There are applications such as image processing, array signal processing, MIMO OFDM, etc in which more than one data stream need to be processed. Therefore, simultaneous multiple FFT operations are required and a dedicated bit reversal circuit is also required to generate the outputs in natural order.

There are FFT architectures [2]–[5] which can handle multiple independent data streams. However, all the data streams are processed by a single FFT processor in [2], [5]. In [5], four independent data streams are processed one by one. Similarly, eight data streams are processed at two domains in [2]. Thus, the outputs of multiple data streams are not available in parallel. In order to process the data streams simultaneous, more than one FFT processors need to be employed. In [3], one to four data streams are processed using multiple data paths for WLAN application. Data of different data streams are interleaved to process them simultaneously in [4]. Nevertheless, the architectures in [2]–[4] do not have any specific bit reversal circuit.

In [6]–[9], certain circuits are proposed to reorder the bit reversed FFT output into normal order. The bit reversal circuits are proposed for different radices in [8]. In [9], a similar structure is proposed for variable length FFT, whose register complexity is N. These circuits are suitable for bit reversing the data from pipelined FFT architecture. However, only the bit reversal structures are proposed in [8], [9]. In [6], the bit reversal circuit is integrated to FFT architecture as a result the total register requirement of the design is reduced from $5N/2$ to $2N$. Two, four, eight-parallel pipelined radix-2^k DIF feed forward FFT architectures are proposed in [10] and they need extra N registers to generate the output in natural order. Moreover, these two, four and eight-parallel FFT architectures can start its operation only when $x(n+N/2)$, $x(n+3N/4)$ and $x(n+7N/8)$ samples arrive respectively. Therefore, hardware is underutilized and additional registers are required to store the first $N/2$, $3N/4$ and $7N/8$ samples respectively.

In [7], a modified MDC FFT architectures with a new data scheduling method and rearranging structure are proposed in which the draw backs of [10] are eliminated. The architectures in [2]–[6], [8]–[10] operated at the frequency of incoming sample rate but the architecture in [7] operates half the clock frequency to generate same throughput as that of [2]–[4], [6], [8]–[10]. Thus, the throughput of the architecture in [7] is doubled if all the architectures are operated at the same speed. Similarly, a combined SDC-SDF FFT architecture [11] with I/O in natural order is proposed in which the bit reversal is carried out only with $N/2$ registers. However, its throughput is low and required number of register is high. In [12], low complexity FFT architectures are proposed but these architectures can process only real-valued signals (signals only with real part). Moreover, they generate two outputs per clock cycle and these outputs are not in natural order. Thus, most of the recent architectures require bit reversal structures to generate the outputs in natural order.

The proposed architecture is designed to process two independent data streams simultaneously with less amount of hardware.The odd inputs which are in natural order are first bit reversed and then they are processed by $N/2$-point DIT FFT. The even samples are directly processed by $N/2$-point DIF FFT so its outputs are in bit reversed order. Therefore, the outputs of $N/2$-point DIF FFT are bit reversed. The outputs of the two $N/2$-point FFTs are further processed by the two parallel butterflies to generate the outputs of N-point FFT in natural order. The bit reversing is carried out by the scheduling registers which are actually used to delay the samples for

Antony Xavier Glittas is with Heriot-Watt University, Edinburgh, UK and the Department of Electronics and Communication Engineering, National Institute of Technology Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India e-mail: (glittas@gmail.com).

Mathini Sellathurai is with Heriot-Watt University, Edinburgh, UK e-mail: (M.Sellathurai@hw.ac.uk).

G. Lakshminarayanan is with the Department of Electronics and Communication Engineering, National Institute of Technology Tiruchirappalli, Tiruchirappalli, Tamil Nadu, India e-mail: (laksh@nitt.edu).

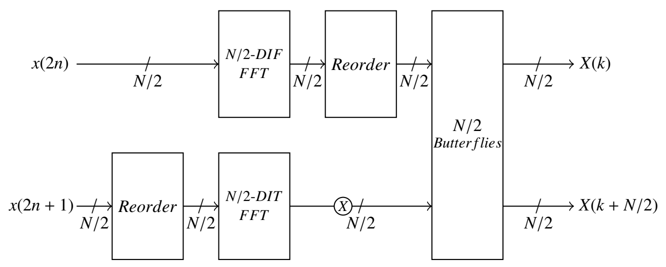


Figure 1: Idea of the proposed method.

performing the butterfly operations. Thus, the FFT architecture does not use any dedicated circuit to bit reverse the data. As a result the proposed architecture requires less number of registers than the prior FFT designs.

The proposed architecture is explained in detail in the section-II. In section-III, the results are compared with prior designs. The paper is concluded in the section-IV.

II. Proposed Pipelined FFT architecture

The idea of computing an N-point FFT using two $N/2$-point FFT operations with additional one stage of butterfly operations is shown in the Fig.1 which is not the exact architecture but provides the methodology. The reordering blocks in the Fig.1 are present merely to state that the $N/2$ odd samples ($x(2n+1)$) are reordered before $N/2$-point DIT FFT operation and $N/2$ even samples ($x(2n)$) are reordered after $N/2$-point DIF FFT operation. In order to compute the N-point DIT FFT from the outputs of two $N/2$-point FFTs, additional one stage of butterfly operations are performed on the results of the two FFTs. Thus, the outputs generated by additional butterfly stage is in natural order.

For the purpose of simplicity, the proposed 16-point FFT architecture in the Fig.2 is explained. It has two 8-point MDC FFT architectures to process two data streams. The delay commutator units present at the left side of SW_1 dissociates the odd and even samples. The shift registers in the delay commutator units which receive inputs, are used to bit reverse the odd input samples. These shift registers are called RSR (reordering shift registers). The RSR in the last stage store outputs from the 8-point DIF FFT and bit reverse them. The BF2 carries out two-parallel butterfly operations between the bit reversed data in the RSR in the last stage and outputs from the 8-point DIT FFT. Thus, the upper and lower BF2 in the last stage generate the FFT outputs of the first and second data streams in normal order. The two data paths from SW_2 are combined together so the word length of the data path in last stage is twice and so thick lines are used for representing the data paths and registers.

A. *Operation of the Proposed Architecture*

The FFT architecture in the Fig.2 is divided into six levels (L_1, L_2, L_3, M_1, M_2 and M_3). The RSR registers in the levels L_1 and M_1 reorder the odd input data and the RSR registers in the level L_3 and M_3 reorder the partially processed even data. The 8-point DIF and DIT FFT operations are performed in the levels L_2 and M_2 respectively. The data from L_1 and

Table I: Data Flow Through Different Levels

Level	Time —–>						
L_1	$E_1(1,1)$	$O_1(1,1)$	$E_1(1,2)$	$O_1(1,2)$			
L_2		$E_1(2,1)$	$E_2(2,1)$	$E_1(2,2)$	$E_2(2,2)$		
L_3			$E_1(3,1)$	$O_1(3,1)$	$E_1(3,2)$	$O_1(3,2)$	
M_1		$E_2(1,1)$	$O_2(1,1)$	$E_2(1,2)$	$O_2(1,2)$		
M_2			$O_1(2,1)$	$O_2(2,1)$	$O_1(2,2)$	$O_2(2,2)$	
M_3				$E_2(3,1)$	$O_2(3,1)$	$E_2(3,2)$	$O_2(3,2)$

M_1 can be forwarded to L_2 and M_2 respectively or vice versa with help of SW_1. Similarly, the data from L_2 and M_2 can be forwarded to L_3 and M_3 respectively or vice versa with help of SW_2. The SW_1 and SW_2 have two switches (SW) to swap the data path and propagate the data to different levels. During the normal mode, the switches (SW_1 or SW_2) pass the data at u_1, u_2, u_3 and u_4 to v_1, v_2, v_3 and v_4 respectively. However, during the swap mode, the switches (SW_1 or SW_2) pass the data at u_1, u_2, u_3 and u_4 to v_3, v_4, v_1 and v_2 respectively.

The SW_1 is in swap mode during first $N/2$ clock cycles and it is in normal mode during $N/2+1$ to N. On other hand, the SW_2 is in normal mode during first $N/2$ clock cycles and it is in swapl mode during $N/2+1$ to N. Thus, SW_1 and SW_2 are in different mode at any time and change their modes for every $N/2$ clock cycles. Moreover, if there is trasistion of data betweeen L_y to L_{y+1} or M_y to M_{y+1} (where y can be 1 or 2) then the switches (SW_1 or SW_2) are in normal mode and if there is trasistion of data betweeen L_y to M_{y+1} or M_y to L_{y+1} then the switches (SW_1 or SW_2) are in swap mode. Like other control signals in the design, the control signals to the switches SW_1 and SW_2 are provided externally and these switch control signals swap at every $N/2$ clock cycles.

The two input streams to the FFT processor are represented as X_1 and X_2. The odd and even samples of two input streams are disassociated by the delay commutator units in L_1 and M_1 (X_1 is disassociated into $\{E_1(i,j), O_1(i,j)\}$ and X_2 is disassociated into $\{E_2(i,j), O_2(i,j)\}$). In these representations, i defines the nature of the data and j defines the number of the data set whose FFT has to be computed. The even set of input data ($x(0), x(2), x(4)...$) is define as $E(1,j)$ and odd set of input data ($x(1), x(3), x(5)...$) is defined as $O(1,j)$. $E(2,j)/O(2,j)$ is the set of scheduled or ordered even/odd data which are ready to fed to 8-point DIF/DIT FFT. The outputs of 8-point DIF/DIT FFT are define as $E(3,j)/O(3,j)$ which are fed to the third level for 16-point FFT computation. The table I explains the operation of FFT and the data propagation through different levels.

1) The first eight samples of X_1 are loaded into the registers ($4D$ in upper and lower arm of delay commutator unit) in L_1. After eight clock cycles, the switch (SW_1) is set in normal mode and the first eight samples of X_2 are loaded into the registers ($4D$) in M_1. Simultaneously, $E_1(1,1)$ (even samples of X_1) is forward from $L_1$to L_2 as $E_1(2,1)$ to perform 8-point FFT operation. The odd samples of X_1 and X_2 are bit reversed by the RSR in L_1 and L_2. The operation of bit reversing is explained in the next subsection

2) After eight clock cycles, the positions of the switches SW_1 and SW_2 are set in swap mode and normal mode respectively. The odd samples ($O_1(1,1)$) of X_1 are forwarded

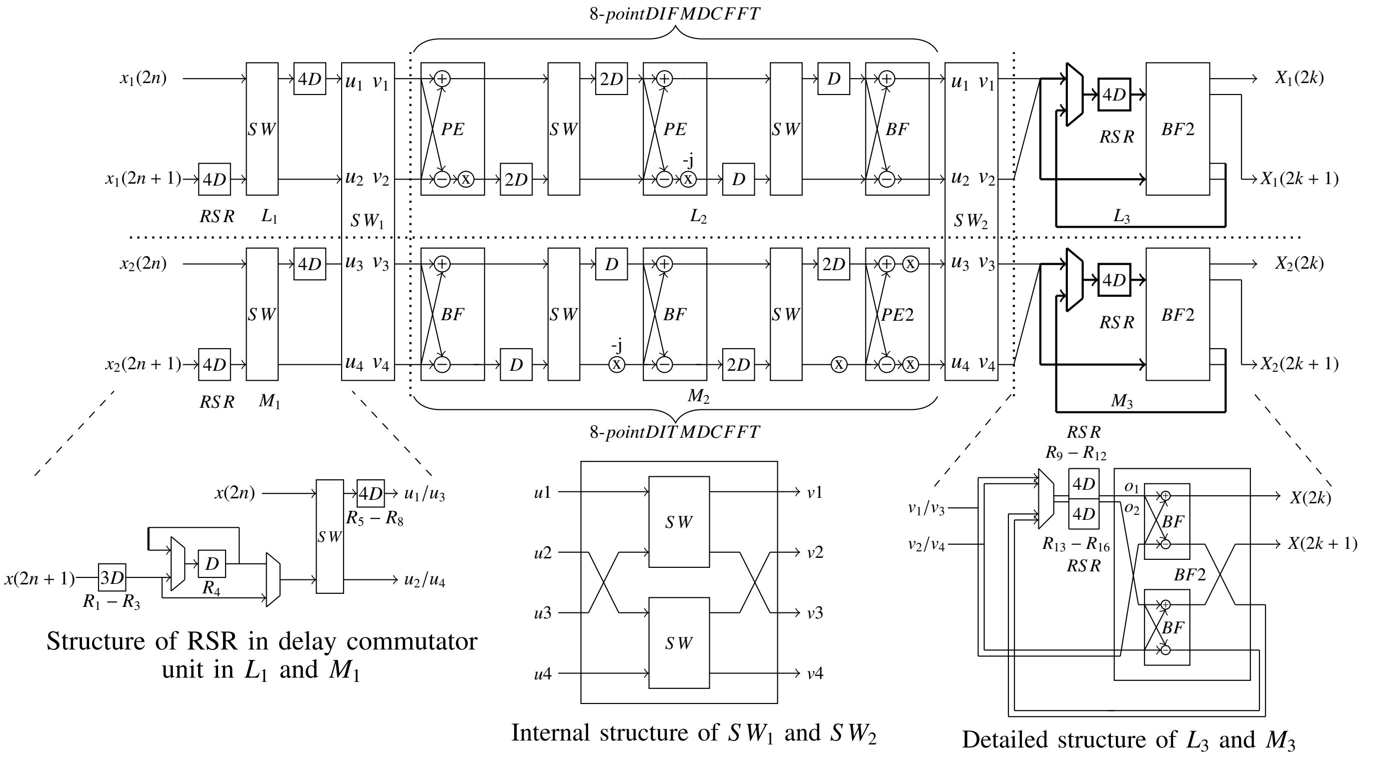


Figure 2: Proposed 16-point radix-2 FFT architecture with outputs in natural order

from L_1 to M_2 as $O_1(2,1)$ and the even samples $(E_2(1,1))$ of X_2 is forwarded from M_1 to L_2 as $E_2(2,1)$. Simultaneously, $E_1(2,1)$ is forward from L_2 to L_3 as $E_1(3,1)$ and reordering is performed.

3) After eight clock cycles, the SW_1 and SW_2 are set in normal mode and swap mode respectively. The odd samples of X_2 $(O_2(1,1))$ are forwarded from M_1 to M_2 as $O_2(2,1)$ and $O_1(2,1)$ is forwarded from M_2 as $O_1(3,1)$ to L_3 where the butterfly operations with $E_1(3,1)$ corresponding to the last stage (of the data stream X_1) are performed. In the meantime, $E_2(2,1)$ from L_2 are forwarded to M_3 as $E_2(3,1)$ and reordering is performed in the RSR.

4) After eight clock cycles, the switch (SW_2) is set to normal position to allow the partially processed odd samples $(O_2(3,1))$ from M_2 to M_3 and perform the butterfly operations of the last stage (of the data stream X_2).

Instead of using radix-2 FFTs as in Fig.2, any higher radix FFTs architecture can be used. In the Fig.3, two radix-2^3 64-point FFTs are used to realize 128-point FFT whose multiplier complexity is $4(log_8(N/2) - .5)$ and working is almost same as 16-point FFT. The multiplier complexity of N-point radix-k FFT algorithm is $4(log_k(N/2) - .5)$.

B. Bit Reversing

The proposed architecture is inspired from the architecture in [7] where $N/2$ data scheduling registers before the first butterfly unit, are used to separate odd samples from the even samples and delay them to generate $x(n)$ and $x(n + N/2)$ in parallel. In proposed architecture, this data scheduling registers are reused to bit reverse odd samples. Similarly, $N/2$ data

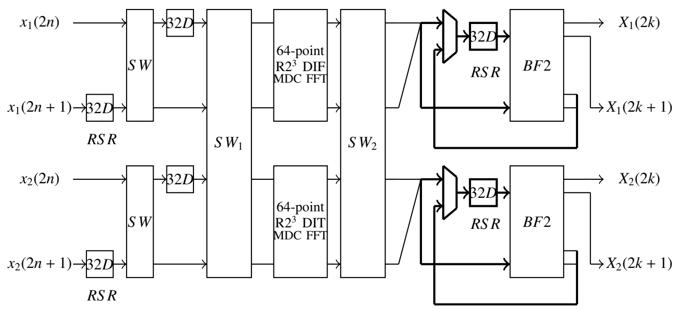


Figure 3: Proposed 128-point radix-2^3 FFT architecture

scheduling register are used before the last butterfly unit to store the partially processed even samples until the arrival odd samples in [7] and here, these registers are reused to bit reverse the partially processed even samples (outputs from DIF FFT). In [8], circuits that use multiplexers and shift registers for bit reversal, are proposed. According to [8], if N is even power of r, the number of registers required to bit reverse N data is $(\sqrt{N} - 1)^2$. If N is odd power of r, the number of registers required to bit reverse N data is $(\sqrt{rN}-1)(\sqrt{N/r}-1)$ where r is the radix of the FFT algorithm. In the proposed architecture, these bit reversal circuits are incorporated in the data scheduling register to perform dual role.

The RSR used in the 16-point FFT and 64-point FFT architectures are show in the Fig.4a and Fig.4b. Actually, these structure are present in the places of the shift registers marked with RSR. Generalized RSR for N-point is shown in the Fig.4c in which c_0 is $N/4 - (\sqrt{N/4} - 1)^2$ or $N/4-(\sqrt{(Nr)/4}-1)(\sqrt{N/(4r)}-1)$. These registers in c_0 do not

Table II: Bit Reversal Operation In The Levels L_1 & M_1

Clk	$x(2n)$	$x(2n+1)$	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	u_3	u_4
0	$x(0)$	$x(1)$	-	-	-	-	-	-	-	-	-	-
1	$x(2)$	$x(3)$	$x(1)$	-	-	-	$x(0)$	-	-	-	-	-
2	$x(4)$	$x(5)$	$x(3)$	$x(1)$	-	-	$x(2)$	$x(0)$	-	-	-	-
3	$x(6)$	$x(7)$	$x(5)$	$x(3)$	$x(1)$	-	$x(4)$	$x(2)$	$x(0)$	-	-	-
4	$x(8)$	$x(9)$	$x(7)$	$x(5)$	$x(3)$	$x(1)$	$x(6)$	$x(4)$	$x(2)$	$x(0)$	$x(0)$	$x(8)$
5	$x(10)$	$x(11)$	$x(9)$	$x(7)$	$x(5)$	$x(3)$	$x(1)$	$x(6)$	$x(4)$	$x(2)$	$x(2)$	$x(10)$
6	$x(12)$	$x(13)$	$x(11)$	$x(9)$	$x(7)$	$x(3)$	$x(5)$	$x(1)$	$x(6)$	$x(4)$	$x(4)$	$x(12)$
7	$x(14)$	$x(15)$	$x(13)$	$x(11)$	$x(9)$	$x(7)$	$x(3)$	$x(5)$	$x(1)$	$x(6)$	$x(6)$	$x(14)$
8	-	-	$x(15)$	$x(13)$	$x(11)$	$x(9)$	$x(7)$	$x(3)$	$x(5)$	$x(1)$	$x(1)$	$x(9)$
9	-	-	-	$x(15)$	$x(13)$	$x(11)$	-	$x(7)$	$x(3)$	$x(5)$	$x(5)$	$x(13)$
10	-	-	-	-	$x(15)$	$x(11)$	-	-	$x(7)$	$x(3)$	$x(3)$	$x(11)$
11	-	-	-	-	-	$x(15)$	-	-	-	$x(7)$	$x(7)$	$x(15)$

Table III: Bit Reversal Operation In The Levels L_3 & M_3

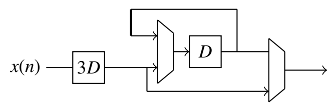
Clk	v_3	v_4	R_9	R_{10}	R_{11}	R_{12}	R_{13}	R_{14}	R_{15}	R_{16}	o_1	o_2
0	$X(0)$	$X(4)$	-	-	-	-	-	-	-	-	-	-
1	$X(1)$	$X(5)$	$X(0)$	-	-	-	$X(4)$	-	-	-	-	-
2	$X(2)$	$X(6)$	$X(1)$	$X(0)$	-	-	$X(5)$	$X(4)$	-	-	-	-
3	$X(3)$	$X(7)$	$X(2)$	$X(1)$	$X(0)$	-	$X(6)$	$X(5)$	$X(4)$	-	-	-
4	-	-	$X(3)$	$X(2)$	$X(1)$	$X(0)$	$X(7)$	$X(6)$	$X(5)$	$X(4)$	$X(0)$	$X(8)$
5	-	-	-	$X(3)$	$X(2)$	$X(1)$	-	$X(7)$	$X(6)$	$X(5)$	$X(4)$	$X(12)$
6	-	-	-	-	$X(3)$	$X(2)$	-	-	$X(7)$	$X(6)$	$X(2)$	$X(10)$
7	-	-	-	-	-	$X(3)$	-	-	-	$X(7)$	$X(6)$	$X(14)$

involve in reordering. The control signals to the multiplexer in RSR are properly varied to interleave the data the data. If log_2N is even, log_2N-2 multiplexers are required otherwise log_2N multipliers are required for bit reversal. For more details on bit reversal [8] may be referred.

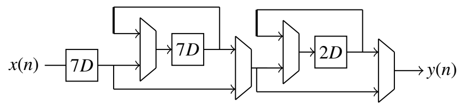
In the proposed FFT architecture, the first $N/4$ and next $N/4$ odd input data to DIF FFT are separately bit reversed as they are required in parallel. Thus, $N/4$-point bit reversing algorithm is enough and the number of registers required to bit reverse $N/4$ data is either $(\sqrt{N/4}-1)^2$ or $(\sqrt{(Nr)/4}-1)(\sqrt{N/(4r)}-1)$ depending upon the power of two. In the Fig.2, the RSR (R_1-R_4) in M_1 bit reverse the first $N/4$ odd input data $(x(1), x(3), x(5), x(7))$ and store them in R_5-R_8 $(x(1), x(5), x(3), x(7))$. After that, the next $N/4$ odd input data $(x(9), x(11), x(13), x(15))$ are bit reversed in R_1-R_4 $(x(9), x(13), x(11), x(15))$ which is explained in the table II. The delay commutator unit in L_1 and M_1 feed the bit reversed odd input samples to $u_1\&u_2$ and $u_3\&u_4$ (in SW_1) respectively. Similarly, in M_3, the RSR (R_9-R_{12}) bit reverse the first $N/4$ output data $(X(0), X(2), X(4), X(6))$ and the RSR $(R_{13}-R_{16})$ bit reverse the next $N/4$ output data $(X(8), X(10), X(12), X(14))$ of DIT FFT separately, which is explained in the table III. Thus, the RSR in L_3 and M_3 bit reverse the partially processed even data samples from $v_1\&v_2$ and $v_3\&v_4$ (in SW_2) respectively and feeds to BF2 (via $o_1\&o_2$).

III. Comparison

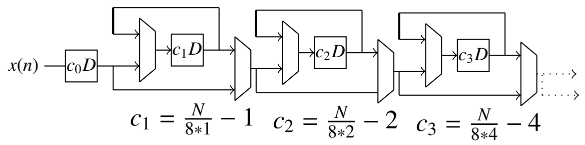
Though several architectures [2]–[5] are proposed for processing multiple independent data streams, each stream is not processed by an individual FFT processor simultaneously. Moreover, the architecture in [5] has disadvantage such as throughput per data stream is low and complicated dual port RAMs are required. The computation complexities of the architecture [5] listed in the table IV correspond to two streams $(N_S = 2)$ as proposed architecture processes two streams. If the single stream architectures [6], [7] need to process two or more streams, two or more such architectures are required in parallel.



(a) RSR used in 16-point FFT architecture



(b) RSR used in 64-point FFT architecture


$x(n)$
c_0D
c_1D
c_2D
c_3D
$c_1 = \frac{N}{8*1} - 1$ $c_2 = \frac{N}{8*2} - 2$ $c_3 = \frac{N}{8*4} - 4$

(c) RSR used in N-point FFT architecture

Figure 4: Reordering shift registers

Although the four parallel architecture [10] generates four outputs in parallel, its hardware utilization is only 25% and require additional $3N/4$ registers to hold the first $3N/4$ data in real-time applications. In [8], [9], only the reordering circuits are proposed for SDF and MDC architectures respectively so the adder and internal register complexities in the table IV remain same as MDC and SDF architecture. The table IV has two types of registers which are internal registers (Int. registers, used to perform FFT operation) and reordering registers (Reo. Registers, used to reorder the FFT outputs in normal order). N_S is number of independent data streams processed. The extend to which the architecture is utilized during FFT computation of a single set of data is called utilization and it is equal $T_{operational}/(T_{operational} + T_{idle})$. $T_{operational}$ and T_{idle} are the time period for which the architecture is operational and idle respectively. N_P is number of parallel outputs. TP is throughput of the architecture which is the product of utilization and N_P.

The total number of registers (including reordering registers) required by an FFT processor [6]–[11] is greater than or approximately equal to $2N$. If these architectures are employed to process two independent data streams simultaneously, their hardware requirements have to be doubled. As a result the required number of registers may increase to $4N$ or more. The proposed architecture uses two $N/2$-point MDC FFT architectures and two butterflies (BF2) to process two data streams but its register requirement is only $3N-4$.

Though the architectures in [6], [11] require less number of adders when compare to the proposed architecture, its throughput is lesser than the proposed architecture. Moreover, complexity of adder and multiplier depend logarithmically on N whereas the complexity of register depends linearly on N. Hence, registers occupy more area of the design than the adders and multipliers. The typical number of transistors required for one bit register and full adder are eight and fourteen respectively but these numbers may vary with different logic styles. We use these data to estimate the hardware

Table IV: Comparison Of The Proposed Architectures With Prior Designs For The FFT Computation Of N-Points

Architecture	Adders	Int. Buffer	Reo. Buffer	Multiplier	N_S	Utilization	Latency	N_P	TP
SDF R2 FFT	$2log_2N$	$N-1$	N	$2log_4N-1$	1	100%	$N+N-1$	1	1
MDC R2 FFT	$2log_2N$	$3N/2-2$	N	$2log_4N-1$	1	50%	$3N/4+N-1$	2	1
MDC R2 FFT [5]	$2log_2N$	$2N-2$	$9N/8+192$	$2log_4N-1$	2	100%	Not specified	2	2
MDC R2 FFT [6]	log_2N	$3N/2$	$N/2$	$2log_4N-1$	1	100%	Not specified	1	1
MDC R2 FFT [7]	$2log_2N$	$3N/2-2$	$5N/8-3$	$2log_4N-1$	1	100%	$(11N/8)-5$	2	2
SDF R2 FFT [8]	$2log_2N$	$N-1$	$(\sqrt{N}-1)^2$	$2log_4N-1$	1	100%	$N+(\sqrt{N}-1)^2-1$	1	1
MDC R2 FFT [9]	$2log_2N$	$3N/2-2$	N	$2log_4N-1$	1	50%	$3N/4+L_{26}^c-1$	2	1
MDC R4 FFT [10]	$8log_4N$	$N+3N/4$	N	$3log_4N-3$	1	25%	$3N/4+N/4+N$	4	1
SDF-SDC R2 FFT [11]	log_2N+1	$1.5N+1.5log_2N-1.5$	$0.5N$	$log_4N-.5$	1	100%	$2N+log_2N-1$	1	1
Proposed	$4log_2N+4$	$3N-4$	0	$4log_4N-4$	2	100%	$3N/4-1$	4	4
Normalized	$2log_2N+2$	$3N/2-2$	0	$2log_4N-2$	1	100%	$3N/4-1$	2	2

requirement of the proposed design with a word length of 16-bit. The hardware cost of 16X16 complex multiplier is approximately equal to 62 (15*4+2) 16-bit adders. The total number of transistors required for $3N-4$ complex registers, $4log_2N+4$ complex adders and $4log_4N-4$ complex multipliers are $2*8*(3N-4)*10$, $2*14*(4log_2N+4)*10$ and $(14*(4log_4N-4)*10)*62$ respectively. Nowadays, DVB and LTE demand FFT architectures with large value of N (2K to 32K). Thus, for large value of N (say, N=8K), the number of transistors required by registers (95%) is much greater than transistors required by the adders (0.33%) and multipliers (4.61%). The multiplier complexity can be reduced by using higher radix FFT algorithm. Therefore, low register complexity is crucial for a pipelined FFT processor.

Table V: FPGA Implementation Results

N	Registers	LUTs	DSP48Es	Latency	Throughput
256 [7]	1038	1594	28	$937ns$	$720MS/s$
512 [7]	1186	1824	32	$1889ns$	$720MS/s$
256 [8]	968	1178	28	$1263ns$	$380MS/s$
512 [8]	1126	1497	32	$2568ns$	$380MS/s$
256 Prop.	1536	2420	56	$505ns$	$1520MS/s$
512 Prop.	1728	2952	64	$1010ns$	$1520MS/s$
1024 Prop.	1920	4116	72	$2021ns$	$1520MS/s$
2048 Prop.	2112	5662	80	$4042ns$	$1520MS/s$

The critical path of the proposed architecture is $2T_{MUX}+T_A+T_M$ (the path along SW, SW_1 and PE). If the value of N is large, the critical path may depend upon the bit reversal circuits as the multiplexers in the reordering circuit are connected in serial. The proposed architecture has two N-point FFT architectures and can process two data streams simultaneously. Therefore, the complexity of proposed is normalized to single data stream, which results in $2log_2N+2$ adders, $3N/2-2$ registers and log_2N-2 multipliers. Thus, the proposed architecture remains dominant in terms of register complexity and throughput. The outputs of [7] have to propagate through the additional registers so the designs in [7] have longer latency. The proposed architecture is implemented and validated on Xilinx Vertex-5 FPGA (XC5VLX110T-1 FF1136) for different values of N with word length of 16 bits and the results are shown in the table V. The architectures operate up to 380 MHz and generates four outputs per clock.

Unlike the proposed architecture, the architectures implemented in [10], [11] have only one FFT processor and their throughput is one fourth of the proposed architecture as the architecture [10] is underutilized by 75% and the architecture [10] generates one output per clock. Thus, proposed FFT architectures has high throughput and less area.

IV. Conclusion

This paper has presented a novel FFT processor whose outputs are generated in natural order. The proposed processor can process two independent data streams simultaneously, makes it suitable for many high speed real-time applications. The bit reversal circuit present in prior designs are eliminated by integrating two FFT processors and the registers which are present in the architecture are reused for bit reversal. As a result, the need of additional registers to bit reverse the outputs is avoided. Moreover, the proposed architecture provides throughput higher than the prior architectures. These attributes make the proposed FFT processor superior in sense of hardware complexity and performance.

V. Acknowledgement

The authors acknowledge the UKIERI project (UKUTP201100134) and the second author MS acknowledges the EPSRC project EP/M014126/1

References

[1] S. He and M. Torkelson, "A new approach to pipeline FFT processor," in *Proc.10th Int. Parallel Process. Symp.*, 1996, pp. 766–770.

[2] Y. Chen, Y. W. Lin, Y. C. Tsao, and C. Y. Lee, "A 2.4-Gsample/s DVFS FFT processor for MIMO OFDM communication systems," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1260–1273, May. 2008.

[3] S.-N. Tang, C.-H. Liao, and T.-Y. Chang, "An area- and energy-efficient multimode FFT processor for WPAN/WLAN/WMAN systems," *IEEE J. Solid-State Circuits*, vol. 47, no. 6, pp. 1419–1435, jul. 2012.

[4] M. G. Padma Prasad Boopal and O. Gustafsson, "A reconfigurable FFT architecture for variable-length and multi-streaming OFDM standards," *ISCAS*, 2013.

[5] 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, May 2013.

[6] 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.

[7] 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.

[8] M. Garrido, J. Grajal, and O. Gustafsson, "Optimum circuits for bit reversal," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 58, no. 10, pp. 657–661, Oct. 2011.

[9] S.-G. Chen, S.-J. Huang, M. Garrido, and S.-J. Jou, "Continuous-flow parallel bit-reversal circuit for MDF and MDC FFT architectures," *IEEE Trans. Circuits Syst. I, Reg Papers*, vol. 61, no. 10, pp. 2869–, Oct. 2014.

[10] M. Mario Garrido, M. A. S. J. Grajal, 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.

[11] Z. Wang, X. Liu, B. He, and F. Yul, “A combined SDC-SDF architecture for normal I/O pipelined radix-2 FFT,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 2015.

[12] A. X. Glittas and G. Lakshminarayanan, “Pipelined FFT architectures for real-time signal processing and wireless communication applications,” in *18th International Symposium on VLSI Design and Test Coimbatore*, Jul. 2014, pp. 1–2.