F. Albertuz and M. Garrido, "A Serial Low-Switching FFT Architecture Specifically Tailored for Low Power Consumption," 2024 39th Conference on Design of Circuits and Integrated Systems (DCIS), Catania, Italy, 2024, pp. 1-6, doi: 10.1109/DCIS62603.2024.10769116 Copyright © 2024, IEEE

A Serial Low-Switching FFT Architecture Specifically Tailored for Low Power Consumption

Francisco Albertuz
Department of Electronic Engineering
Universidad Politécnica de Madrid
Madrid, Spain
francisco.albertuz@upm.es

Mario Garrido
Department of Electronic Engineering
Universidad Politécnica de Madrid
Madrid, Spain
mario.garrido@upm.es

***Abstract*—This paper presents a new FFT architecture called serial low-switching FFT, which has been specifically conceived for low power consumption. To achieve this, the architecture has been designed with the aim of reducing the switching activity of the components of the architecture. Experimental results show that this approach reduces the power consumption by approximately one third with respect to equivalent FFT architectures in the literature.**

***Index Terms*—FFT, SLS, pipelined architecture, low power consumption, low switching activity, optimization, digital circuit, FPGA, 6G, radar, space, radio astronomy.**

I. Introduction

The fast Fourier transform (FFT) is one of the most important signal-processing algorithms. Currently, cutting-edge technologies demand higher and higher performance for the FFT. This is the case of 6G communications, where the data rate is expected to reach 1 Tb/s [1]. Likewise, in radar, space, and radio astronomy systems, FFTs must be calculated at very high data rates.

There are two main alternatives to achieve the high-throughput FFTs that these applications demand. One of them is to use a higher clock frequency [2] in the systems and the other is to increase the parallelization of the design, making it capable of processing several samples per clock cycle [2]–[6]. However, in both cases, the higher throughput comes with a higher power consumption.

In a previous study [7] we analyzed the relation between throughput and power consumption in pipelined FFT architectures. For the most efficient FFT designs, doubling the throughput multiplies the power consumption by three. Thus, reducing the power consumption in FFTs has become a crucial metric to consider for the design of high-performance FFTs. In fact, 6G technology states a clear goal to reduce the power consumption in the communication systems [1], which was not present in 5G.

This work was supported in part by MCIN/AEI/10.13039/501100011033 and "ERDF A way of making Europe" under Project PID2021-126991NA-I00 and in part by MCIN/AEI/10.13039/501100011033 and "ESF Investing in your future" under Grant RYC2018-025384-I.

This is a preprint document. To cite the work or retrieve its final Open Access version, follow the DOI: 10.1109/DCIS62603.2024.10769116

The power consumption reduction in FFT architectures can be tackled by following various approaches. Although it is not always the case, reducing the area of the circuit generally leads to less power consumption. This can be achieved through techniques like rotator allocation [8]. This technique explores multiple data orders through the FFT stages to obtain those that reduce the number of rotators and their complexity. Additionally, shift-and-add techniques [9], [10] allow for implementing some of those rotators using only a few adders [2], [11]. Alternatively, rotators on field-programmable gate arrays (FPGAs) may be implemented using DSP slices to reduce the power consumption [12].

The power consumption can also be lowered by reducing the switching activity [13]. Some works in the literature analyze the switching activity of the coefficients in the rotation memories of the FFT [14], [15]. In this context, a way to reduce the switching activity is to rotate by the same coefficient in consecutive clock cycles.

Another way to reduce the switching activity is to use memories instead of registers to implement the buffers in the FFT. It has been proven that, even for small buffer sizes, memories consume less power than buffers [16]. This occurs because in a shift register all the elements must be updated every clock cycle, whereas, in a memory, only the value selected by the address needs to be updated.

As a result, there are multiple techniques to reduce power consumption in FFTs in the literature. However, these are mostly isolated ideas that have not been integrated into a single FFT architecture design. Consequently, previous works do not analyze how much power consumption can be saved if we design an FFT specifically tailored for low power consumption.

This work presents a new FFT architecture that has been entirely designed aiming to achieve low power consumption. From the architecture itself to the smallest details, all the design decisions have been taken towards low power. In fact, the proposed architecture is a new type of architecture called serial low-switching (SLS) FFT, which has not been presented before in the literature. As will be noticed through the explanations in the paper, this architecture has remarkable characteristics that make it more suitable for low power than previous FFT architectures.

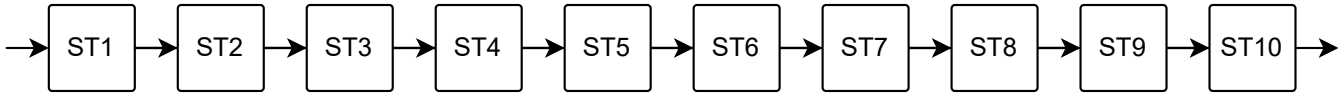

Fig. 1. Block diagram of the stages of a serial 1024-point FFT.

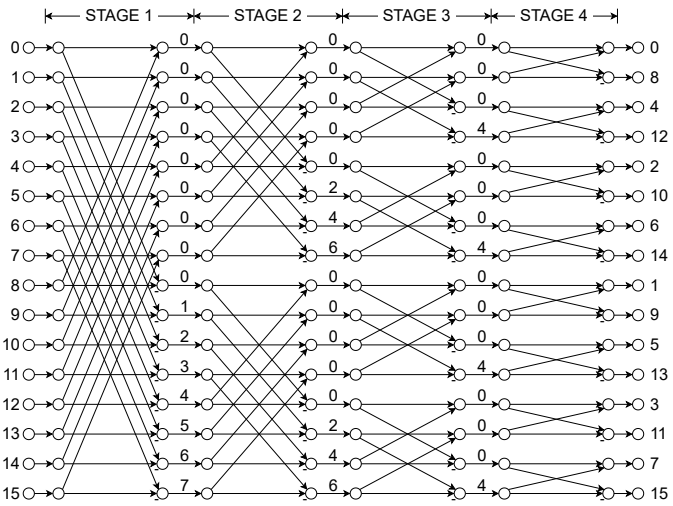

Fig. 2. 16-point radix-2 DIF FFT flow graph.

Although the final future research goal will be to derive highly parallel low-power FFT architectures for 6G, radar, radio astronomy, and space, this work is only the first step in this research line. Therefore, in this work, we present a serial pipelined FFT architecture as a proof of concept to demonstrate that a significant power consumption reduction is feasible in FFT architectures. In future work, we will generalize these ideas to design the high-throughput low-power parallel FFT architectures required in 6G, radar, radio astronomy, and space applications.

This paper is organized as follows. In Section II, we review previous FFT architectures related to the proposed one. In Section III, we describe the proposed architecture and the approaches that we have applied to reduce its power consumption. In Section IV, we present the experimental results and compare them to previous works. Finally, in Section V, we summarize the main conclusions of the paper.

II. Background

Fig. 1 shows a block diagram of a serial architecture to calculate a 1024-point FFT. It consists of 10 stages connected in series. The circuit that calculates the stage depends on the type of architecture. Among serial pipelined FFT architectures, the single-path delay feedback (SDF) [17], [18] and the single-stream feedforward (SFF) [19] FFT architectures have the property that they do not alter the order of the data at the FFT stages. The proposed SLS FFT shares this property, which makes it comparable to the SDF and SFF FFTs. Therefore, these two FFT architectures are reviewed next as the previous work most related to the proposed architecture.

Fig. 2 shows the flow graph of a 16-point radix-2 FFT, decomposed according to decimation in frequency (DIF). In SDF and SFF architectures, data arrive in series at each of the stages. The order of arrival of the data is from the top to the bottom of the flow graph in Fig. 2, and each of the samples consists of real and imaginary parts. The numbers at the input of the flow graph show the index of the input samples, ranging from x_0 to x_{15}. Likewise, the numbers at the output of the flow graph correspond to the frequency index of the outputs, including the values from X_0 to X_{15}.

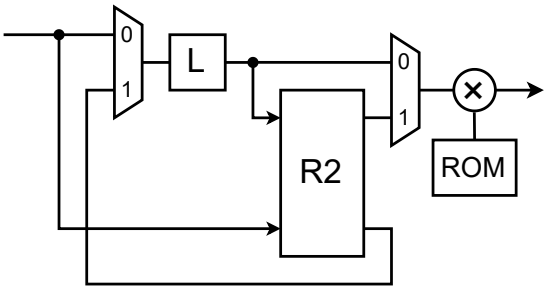

Fig. 3. Stage of a radix-2 SDF FFT architecture.

Butterflies in the flow graph are represented with arrows. Each butterfly has two inputs and two outputs. The upper output of the butterfly corresponds to the addition and the lower to the subtraction. For instance, the first butterfly at stage 1 adds and subtracts the samples x_0 and x_8. In SDF and SFFT architectures, where data flow from top to bottom in the flow graph, the number of clock cycles that the architecture must wait for the second input of each butterfly can be calculated as

$$L = 2^{n-s}, \tag{1}$$

where $n = \log_2 N$ is the total number of stages and s is the current stage number. For instance, at the first stage of a 16-point FFT $n = \log_2 16$ and $s = 1$, so $L = 8$, which agrees with the separation between x_0 and x_8 in Fig. 2.

The numerical values between stages in Fig. 2 represent the rotation coefficients. Rotations can be calculated as complex multiplications by a constant of the form $C + jS$. Each of these constants is obtained with the expression

$$C + jS = \cos\left(\frac{2\pi}{N}\phi\right) + j \cdot \sin\left(-\frac{2\pi}{N}\phi\right), \tag{2}$$

where N is the number of points of the FFT and ϕ is the rotation value in the flow graph.

A. Stage of an SDF FFT Architecture

Fig. 3 shows the stage of an SDF FFT architecture. In this circuit, input data are first stored in a buffer of length L. When the corresponding samples arrive, they are added to and subtracted from the samples stored in the buffer in a radix-2 butterfly (R2), consisting of an adder and a subtracter that

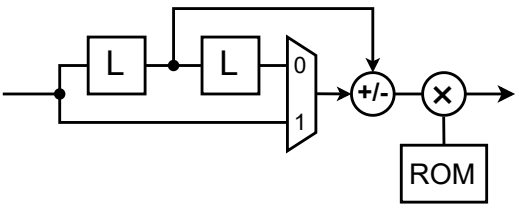

Fig. 4. Stage of a radix-2 SFF FFT architecture.

operate on two input samples. The results of the additions are sent to the rotator directly from the butterfly as it carries out the computations. By contrast, the results of the subtractions are fed back into the buffer. These results are sent to the rotator from the buffer immediately after the results of the additions. Note that this way of carrying out the calculations preserves the order from top to bottom in the flow graph.

This architecture uses one buffer of length L, two multiplexers, one radix-2 butterfly (adder and subtracter), and one rotator with its corresponding rotation memory.

The SDF architecture shows some inefficiencies regarding power consumption. On the one hand, the butterfly of the SDF FFT is used only 50% of the time. However, its inputs are changing 100% of the time. This means that half of the processing time the adder and the subtractor of the butterfly are commuting without actually producing valid data. On the other hand, all the outputs of the butterfly pass through the rotator. However, in Fig. 2 it can be observed that only half of the data must be rotated at each FFT stage, as rotations by ϕ are just multiplications by 1 that do not change the data. This means that the rotator is commuting unnecessarily during half of the processing time.

B. Stage of an SFF FFT Architecture

Fig. 4 shows the stage of an SFF FFT architecture. This structure stores the first batch of data in a buffer of length L. When the next L data arrive, it uses an adder/subtracter to calculate the additions between the input samples and those stored in the buffer. The results of these additions are sent directly to the rotator. While this happens, the second buffer stores the first batch of samples for other L clock cycles, and the second batch is stored in the first buffer. When all additions are outputted, the operator is set to subtraction mode and subtractions are carried out. As for the outputs of the additions, they are sent to the rotator.

This architecture uses two buffers of length L, one multiplexer, one adder/subtracter, and one rotator with its corresponding memory.

Contrary to the SDF FFT, the SFF architecture uses the adder/subtracter during 100% of the processing time. However, as in the SDF FFT, all the results of the additions and subtractions pass through the rotator, even when some of them must not be rotated. Additionally, a drawback of the SFF with respect to the SDF FFT is that it uses two buffers of length L instead of one, which increases the power consumption in the buffers.

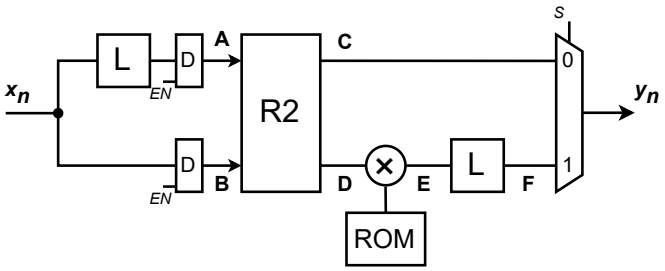

Fig. 5. Stage of the proposed radix-2 SLS FFT architecture.

TABLE I
TIMING DIAGRAM OF THE SLS FFT STAGE IN FIG. 5 FOR $L = 8$.

t (cyc.)	x_n	EN	A	B	C	D	E	F	S	y_n
0	x_0	0								
1	x_1	0								
2	x_2	0								
3	x_3	0								
4	x_4	0								
5	x_5	0								
6	x_6	0								
7	x_7	0								
8	x_8	1								
9	x_9	1	x_0	x_8	ad_0	sb_0	rt_0		0	y_0
10	x_{10}	1	x_1	x_9	ad_1	sb_1	rt_1		0	y_1
11	x_{11}	1	x_2	x_{10}	ad_2	sb_2	rt_2		0	y_2
12	x_{12}	1	x_3	x_{11}	ad_3	sb_3	rt_3		0	y_3
13	x_{13}	1	x_4	x_{12}	ad_4	sb_4	rt_4		0	y_4
14	x_{14}	1	x_5	x_{13}	ad_5	sb_5	rt_5		0	y_5
15	x_{15}	1	x_6	x_{14}	ad_6	sb_6	rt_6		0	y_6
16			x_7	x_{15}	ad_7	sb_7	rt_7		0	y_7
17								rt_0	1	y_8
18								rt_1	1	y_9
19								rt_2	1	y_{10}
20								rt_3	1	y_{11}
21								rt_4	1	y_{12}
22								rt_5	1	y_{13}
23								rt_6	1	y_{14}
24								rt_7	1	y_{15}

III. PROPOSED ARCHITECTURE

The proposed SLS FFT architecture has been designed with the purpose of reducing power consumption. This is achieved by reducing the switching activity in the architecture This way, the SLS FFT presents a new structure that removes the main drawbacks in SDF and SFF FFTs related to power consumption. In Section III-A we provide an overview of the SLS FFT architecture and explain how it works. In the rest of the Section, we analyze the special characteristics of the proposed architecture that allow it to achieve low power consumption.

A. Overview of the Architecture

Fig. 5 shows the stage of the proposed SLS FFT architecture. It consists of an input buffer, a pair of registers (D) with an enable signal (EN), a radix-2 butterfly, a rotator, an output buffer, and a multiplexer. As in SDF and SFF FFT architectures, the first input data are stored in a buffer for L clock cycles. Then, the stored samples and those arriving at the input are added and subtracted in the butterfly. The result of

TABLE II
SUMMARY OF POWER CONSUMPTION AND AREA USAGE RESULTS FOR THE DIFFERENT IMPLEMENTATIONS OF THE INPUT BUFFER.

	SR			**DPLRAM**			**SPLRAM**			**DPBRAM**			**SPBRAM**		
Size	Power	CLB	BRAM	Power	CLB	BRAM	Power	CLB	BRAM	Power	CLB	BRAM	Power	CLB	BRAM
8	5	4	0	6	6	0	7	5	0	9	0	0.5	8	0	0.5
16	5	4	0	6	6	0	7	6	0	9	0	0.5	8	0	0.5
32	6	9	0	7	8	0	7	7	0	9	0	0.5	8	0	0.5
64	9	16	0	9	9	0	9	8	0	9	0	0.5	8	0	0.5
128	15	31	0	14	16	0	14	16	0	9	0	0.5	8	0	0.5
256	25	63	0	26	31	0	27	34	0	9	0	0.5	8	0	0.5
512	44	96	0	50	66	0	51	63	0	9	0	0.5	8	0	0.5
1024	82	160	0	93	124	0	91	123	0	9	0	1	8	0	1
2048	156	287	0	161	200	0	161	198	0	17	0	2	16	0	2
4096	307	543	0	308	355	0	301	328	0	28	0	4	23	0	4

the subtraction is directly connected to the rotator. The rotated value is then stored in another buffer until all additions are outputted. Finally, the results stored in the output buffer are outputted immediately after the results of the additions.

Table I shows the timing diagram of the SLS stage in Fig. 5, for the case of $L = 8$, which corresponds to the first stage of the flow graph in Fig. 2. The first column of the table represents the time measured in clock cycles (cyc.). The second and last columns represent the input (x_n) and output (y_n) of the circuit, respectively. The rest of the columns show the evolution of different points in the circuit marked with letters in Fig. 5, as well as the control signals EN and S. We can observe that the enable signal is set to 1 after eight clock cycles, which is equal to the value of L for this stage. One clock cycle later, the first sample that was stored in the input buffer appears at A, and sample x_8 is registered at B. At this point, the butterfly calculates the addition (ad_0) and subtraction (sb_0) between these samples, and the rotator calculates the rotated value (rt_0) of the subtraction. Also, the selection signal for the multiplexer is set to 0, so that the result of the addition is outputted. The rotated value is stored in the output buffer and is outputted after eight cycles as the selection signal changes to 1. This process is done analogously for the rest of the samples, calculating additions ad_{1-7}, subtractions sb_{1-7}, and rotations rt_{1-7} in the same way.

The stage of the proposed SLS FFT architecture has two buffers of length L, one radix-2 butterfly, one rotator with its rotation memory, and one multiplexer. Its latency from first input to first output is $L + 1$ clock cycles.

B. Low-Power Features

To develop the SLS FFT architecture, we have applied different techniques to our design that allowed us to achieve low switching activity:

- First, the registers before the butterfly are enabled only when the butterfly performs calculations of valid data. Conversely, during the clock cycles where the butterfly is not used, the register values do not change and there is no switching in the butterfly, so no power is lost in unuseful operations.
- The rotator in this architecture is located right after the subtracter. Thus, only those values that must be rotated pass through the rotator, whereas the outputs of the addition do not cross it, removing the inefficiency of previous architectures.
- For the rotator itself, we use a power-efficient low-area complex multiplier that uses only three DSP slices [12]. Note that this rotator has a delay of four clock cycles, so the output buffer length must be reduced by four.
- As in other previous radix-2 FFT architectures, in the last stage, no rotation is calculated, so the rotator can be eliminated. The rotations in the second-to-last stage can be performed with a trivial rotator, which only changes the sign of the real part and then interchanges the real and imaginary parts. These simplifications also help reduce the power consumption.
- A study of the buffers has been carried out to use the most power-efficient ones. This study is detailed in Section III-C.
- The control of the architecture has been designed thoroughly in order to reduce the power consumption of the architecture, which is explained in Section III-D.

C. Buffer Selection

For the design of the architecture, we have studied different alternatives for implementing the buffers and selected those that minimize the power consumption depending on their size. Table II shows the implementation results regarding area usage, expressed in CLBs, and power consumption, expressed in mW, for different ways of implementing a buffer. We consider power-of-two buffer sizes and five different implementations, which are shift register (SR), dual-port look-up table random-access memory (DPLRAM), single-port LUT RAM (SPLRAM), dual-port block RAM (DPBRAM), and single-port block RAM (SPBRAM). All buffers are implemented on a Virtex Ultrascale+ XCVU37P-L2FSVH2892E FPGA for a clock frequency of 600 MHz and a word length of 32 bits.

We can see that the shift register performs well for small buffers but, as we increase their size, power consumption and area increase significantly. For LUT RAM we obtain similar results since both alternatives make use of CLBs to implement the logic. For BRAM, however, the power consumption depends on the number of memory blocks used, which results in much better performance for the greatest buffer sizes.

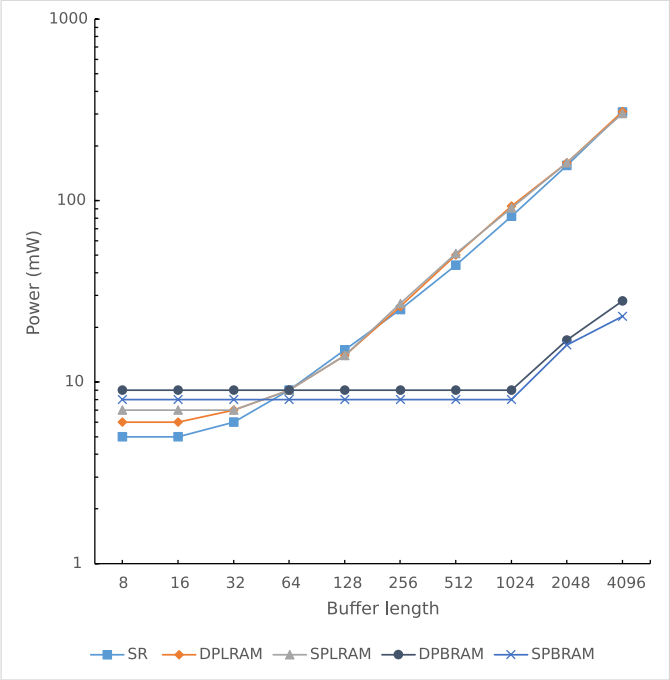

Fig. 6. Buffer power consumption for various implementation alternatives.

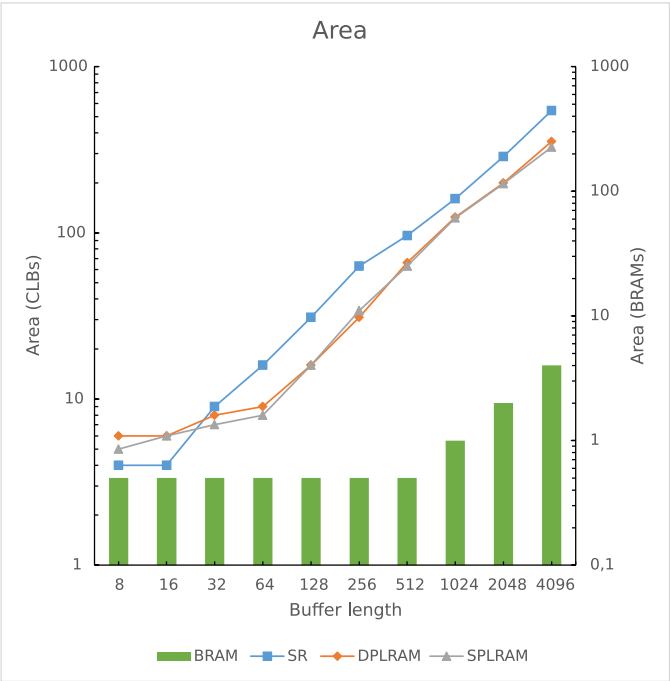

Fig. 7. Buffer area for various implementation alternatives.

In Fig. 6 a graph of the power consumption for the different buffer sizes and implementations is shown. In the Y-axis we plot the power expressed in mW, and in the X-axis, the considered buffer sizes. Each implementation is represented with a line with different marking and color. Coinciding with the results presented in Table II, the line that corresponds to the shift register implementation is the lowest for buffer size 32 and below. For larger buffers, we can see a clear difference between the alternatives that use BRAM and the rest, being the SPBRAM the optimal approach among them.

Fig. 7 shows a similar graph showcasing area usage. The lines represent used CLBs (Y-axis, left) in SR, DPLRAM, and SPLRAM, whereas the bars represent BRAM usage (Y-axis, right) in DPBRAM and SPBRAM implementations. The X-axis shows the different buffer sizes. The number of CLBs used increases consistently, whereas the number of used BRAMs only starts to increase from size 1024. This happens because the minimum number of synthesizable BRAM is 0.5, so all buffer sizes below 512 use the same area even though they need fewer memory slots.

In this paper, we present a 1024-point FFT circuit using the new proposed SLS architecture. As seen in Fig. 1, this design consists of ten stages with buffer sizes 1 to 512. Choosing the alternative with the lowest power consumption for each buffer size leads to stages 5 to 10 being implemented using shift registers, and stages 1 to 4 with BRAM. In these stages, the input buffer is implemented with single-port BRAM, whereas the output buffer is implemented using dual-port BRAM. This is because there are some clock cycles at which we have to read the output memory while the values coming from the rotator still need to be stored, as seen in Table I. All memories also incorporate a write-enable (we) signal, which ensures that data are not updated when there are no new values to store.

D. Control

To manage the control signals and the addresses of the memories, we implement a 10-bit global counter $c_9 \dots c_0$ that increases every clock cycle. In each stage s, the addresses of the ROM memories and input buffers, if implemented with BRAM, correspond to the bits $c_{n-s-1} \dots c_0$ less significant bits of the counter. The rest of the control signals (we_{in}, we_{out}, EN and S) depend on the bit c_{n-s} of the counter. Therefore, most of the signals of the architecture are generated easily from the same counter. The only exception are the output buffers in the stages implemented with BRAM, which require another counter independent from the global one since their size is not a power of two.

IV. Experimental Results and Comparison

In this section, we analyze the experimental results of a 1024-point FFT implementation using the proposed SLS architecture. The design is implemented in a Virtex Ultrascale+ XCVU37P-L2FSVH2892E FPGA with a target clock frequency of 360 MHz and a word length (WL) of 16 bits for each real and imaginary parts of the data. Table III shows a comparison between our design and other similar architectures presented in the literature. The type of FPGA used for the implementations can be Virtex 5 (V5), Virtex 7 (V7), or Virtex UltraScale+ (VU+). In the comparison, area is measured in terms of look-up tables (LUT), registers (FF), configurable logic blocks (CLB), CARRY8, F7MUX, slice blocks (Slices), digital signal processing slices (DSP), and BRAM. The performance comparison includes the clock frequency used (f_{CLK}),

TABLE III

SUMMARY OF PERFORMANCE AND AREA USAGE RESULTS FOR DIFFERENT FFT IMPLEMENTATIONS.

Architecture	[18]	[20]	[21]	[22]	[23]	Prop.
N	1024	1024	256	32	16	1024
P	1	1	8	1	1	1
***WL* (bits)**	16	16	16	-	17	16
FPGA	VU+	V7	V5	V7	V7	VU+
LUT	2305	1849	3582	4028	3588	1543
FF	1821	1292	298	2297	2551	1029
Slices		585	1237	-	1411	
CLB	533					323
DSP	27	12	48	0	0	24
BRAM	2	4	0	0	0	4
$f_{\mathbf{CLK}}$ **(MHz)**	500	360	35.76	25	78.739	360
Lat. (cyc.)	1087	1063	32	-	-	1033
Lat. (ns)	2174	2942	895	-	-	2869
Th. (S/cyc.)	1	1	8	1	1	1
Th. (MS/s)	500	360	286	25	78	360
SQNR (dB)	54.43	56.25	-	-	-	56.22
Power (mW)	490	312	181	112	159	220
NP (μW/(MHz·P))	980	866	633	4480	2019	611

latency (Lat.), throughput (Th.), signal-to-quantization-noise ratio (SQNR), power, and normalized power (NP) expressed in μW per MHz and number of parallel samples (P). Those parameters that are not mentioned in the cited papers are represented with a dash (-).

To measure dynamic power accurately, a register is placed at the input of the system, which allows input data to be synchronized with the clock. Also, a simulation under a representative test bench is run to establish the estimated switching activity of the signals.

Compared to architectures with the same FFT size, our approach archives a reduction of 37.64% of the power consumption with respect to the SDF architecture presented in [18], and 29.5% with respect to the basic SDF architecture described in [20]. [21] presents an eight-parallel multi-path delay commutator (MDC) FFT architecture that is also efficient in terms of power consumption. If we normalize the power consumed by this FFT by frequency and number of parallel paths, we find that our design is 3.41% more efficient even when the cited architecture implements a 256-point FFT, two stages smaller than ours. Other designs in the literature that are not optimized for low power like [22] and [23], obtain higher measurements of power consumption and area usage, even for a limited number of points.

V. CONCLUSIONS

We have successfully designed and implemented a novel FFT architecture specifically tailored for lowering power consumption. To achieve it, we have applied techniques that allow us to reduce the switching activity of the signals in the circuit. Experimental results show that the proposed architecture achieves savings of over 29% in power consumption with respect to equivalent FFT architectures.

REFERENCES

[1] N. Rajatheva et al., "White paper on broadband connectivity in 6G," 6G Research Vision, University of Oulu, Tech. Rep. 10, Jun. 2020.

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

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

[4] P. Paz and M. Garrido, "A 5.2-GSps 8-parallel 1024-point MDC FFT," in *Proc. Conf. Design Circuits Integrated Syst.*, Nov. 2023, pp. 55–60.

[5] A. X. Glittas, M. Sellathurai, and G. Lakshminarayanan, "A normal I/O order radix-2 FFT architecture to process twin data streams for MIMO," *IEEE Trans. VLSI Syst.*, vol. 24, no. 6, pp. 2402–2406, Jun. 2016.

[6] Z. Kaya and M. Garrido, "Low-latency 64-parallel 4096-point memory-based FFT for 6G," *IEEE Trans. Circuits Syst. I*, vol. 70, no. 10, pp. 4004–4014, Oct. 2023.

[7] M. Garrido, "Evolution of the performance of pipelined FFT architectures through the years," in *Proc. Conf. Design Circuits Integrated Syst.*, Nov. 2020, pp. 1–6.

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

[9] F. d. Dinechin and M. Kumm, *Application-Specific Arithmetic - Computing Just Right for the Reconfigurable Computer and the Dark Silicon Era*. Springer, 2024.

[10] R. Garcia and A. Volkova, "Toward the multiple constant multiplication at minimal hardware cost," *IEEE Trans. Circuits Syst. I*, vol. 70, no. 5, pp. 1976 – 1988, May 2023.

[11] M. Garrido, F. Qureshi, and O. Gustafsson, "Low-complexity multiplierless constant rotators based on combined coefficient selection and shift-and-add implementation (CCSSI)," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 7, pp. 2002–2012, Jul. 2014.

[12] P. Paz and M. Garrido, "Efficient implementation of complex multipliers on FPGAs using DSP slices," *J. Signal Process. Syst.*, vol. 95, pp. 543–550, Apr. 2023.

[13] A. P. Chandrakasan and R. W. Brodersen, "Minimizing power consumption in digital CMOS circuits," *Proc. IEEE*, vol. 83, no. 4, pp. 498–523, Apr. 1995.

[14] F. Qureshi and O. Gustafsson, "Twiddle factor memory switching activity analysis of radix-2^2 and equivalent FFT algorithms," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2010, pp. 4145–4148.

[15] I. Qureshi and F. Qureshi, "Impact of FFT algorithm selection on switching activity and coefficient memory size," *Indonesian J. Electrical Eng.*, vol. 12, no. 8, pp. 6224–6229, Aug. 2014.

[16] T. Ahmed, M. Garrido, and O. Gustafsson, "A 512-point 8-parallel pipelined feedforward FFT for WPAN," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Nov. 2011, pp. 981–984.

[17] X.-Y. Shih, H.-R. Chou, and Y.-Q. Liu, "Design and implementation of flexible and reconfigurable SDF-based FFT chip architecture with changeable-radix processing elements," *IEEE Trans. Circuits Syst. I*, vol. 65, no. 11, pp. 3942–3955, Nov. 2018.

[18] V. M. Bautista and M. Garrido, "An automatic generator of non-power-of-two SDF FFT architectures for 5G and beyond," in *Proc. Conf. Design Circuits Integrated Syst.*, Nov. 2023, pp. 61–66.

[19] C. Ingemarsson and O. Gustafsson, "SFF—The single-stream FPGA-optimized feedforward FFT hardware architecture," *J. Signal Process. Syst.*, vol. 90, p. 1583–1592, Nov. 2018.

[20] M. Garrido, V. M. Bautista, A. Portas, and J. Hormigo, "Advanced quantization schemes to increase accuracy, reduce area, and lower power consumption in FFT architectures," *IEEE Trans. Circuits Syst. I*, 2024, Early Access.

[21] S. Mookherjee, L. DeBrunner, and V. DeBrunner, "A high throughput and low power radix-4 FFT architecture," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Nov. 2014, pp. 1266–1270.

[22] X. Han, J. Chen, B. Qin, and S. Rahardja, "A novel area-power efficient design for approximated small-point FFT architecture," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 39, no. 12, pp. 4816–4827, Mar. 2020.

[23] M. Chouhan, A. S. Raghuvanshi, and D. Muchahary, "FPGA implementation of high performance and energy efficient radix-4 based FFT," in *Proc. Asian Conf. Innov. Tech.*, Aug. 2022, pp. 1–5.