MEASUREMENT SCIENCE REVIEW, **25**, (2025), No. 3, 134-140

sciendo

ISSN 1335-8871

MEASUREMENT SCIENCE REVIEW

Journal homepage: https://content.sciendo.com

Performance Estimation of Low Power and Area-Efficient Parallel Pipelined FFT

Surya P[1]*, Arunachalaperumal C[2], Dhilipkumar S[3]

[1]*Research Scholar, Anna University, Chennai, 600025, Tamil Nadu, India,* suryame3020@gmail.com
[2]*Professor, Department of Electronics and Communication Engineering, Ramco Institute of Technology, Rajapalayam, 626117, Tamil Nadu, India,* arunachalaperumal@ritrjpm.ac.in
[3]*Assistant Professor, Department of Electronics and Communication Engineering, Loyola ICAM College of Engineering and Technology (LICET), Chennai, 600034, Tamil Nadu, India,* dhilipkumar.s@licet.ac.in

Abstract: We present a novel parallel and pipelined fast Fourier transform (FFT) architecture for high-speed and low-power applications, a critical component in wireless communications and digital signal processors. The new FFT model implements a data-inverted Vedic multiplier in the FFT architecture, which reduces data switching activity in the input patterns to minimize dynamic power consumption and computational delay. The proposed architecture incorporates a low-power bit inversion (BI) multiplier scheme for a minimum number of complex multiplications with a high-speed partial product generation technique in FFT computation. This research focuses on the investigation and implementation of a modified butterfly unit as the best choice compared to other low-power and high-speed multipliers, such as Booth and Wallace multipliers for FFT processors. The BI multiplier design was synthesized in a field programmable gate array (FPGA), and the results show that the area efficiency could be improved by about 30 % and the power consumption and delay could be reduced by 56 %. The proposed FFT processor utilizes only 8 % of the available look-up tables (LUTs) with a 1:3 ratio in resource utilization and a 56 % reduction in delays compared to previous research. This makes this architecture best suited for high-speed wireless communications and 5G applications. This BI-Vedic multiplier is used in convolutions, FFT, and digital signal processing (DSP) filters where fast multiplication is critical. Throughput in applications with real-time signals is improved. It is also used in image and video processing and is critical for algorithms that manipulate pixels, scale, and compress data when many multiplications need to be performed quickly. IoT and embedded systems are beneficial for low-power systems as BI reduces power consumption and switching activity.

Keywords: low power fast Fourier transform, Vedic multiplier, bit inversion technique, look-up table, field programmable gate array, 5G application

1. Introduction

The fast Fourier transform (FFT) is a widely used mathematical algorithm in technical computing. It is used in various fields such as digital image processing, radar signal processing, wireless communications, and numerical solvers such as partial differential solvers and mechanical systems [1]. In digital signal processing, a conversion between the time domain and the frequency domain is necessary. For this purpose, the FFT algorithm has become one of the most important. The single-path delay feedback (SDF) FFT is the most commonly used structure for computing the FFT in hardware, as it offers high throughput with fewer hardware resources. Each stage of the SDF FFT consists of a butterfly unit, a rotator, and a buffer stage for the calculation. The butterfly unit performs both additions and subtractions, followed by the buffer stage, and the rotator rotates the data in the complex plane. The rotation angle can be changed at any time, and the set of angles rotated by the rotator is called the twiddle factor.

The rotation operation is costly in hardware, so various techniques have been applied to reduce the hardware required to implement rotators [2]. Rotator allocation can simplify FFT rotators where different coefficients are needed to merge the shift-and-add multiplications to create the rotator stage. This idea can effectively distribute the rotations across the parallel branches in parallel FFT architectures. FFT algorithms have high computational parallelism and a pipelined structure with regular data flow, making them suitable for implementation in GPU-based large-scale parallel data processors [3]. The efficient design of pipelined FFT architectures can achieve high performance for modern real-time applications. They offer low latency and high

DOI: 10.2478/msr-2025-0016

**Corresponding author: suryame3020@gmail.com (Surya P)* throughput suitable for high-speed real-time processing [4]. Advanced communication systems widely use FFT, especially in 4G technologies such as orthogonal frequency division multiplexing (OFDM) and single-carrier frequency division multiple access (SC-FDMA). IEEE 802.11ax and 802.11ay are highly dependent on the speed and throughput of the FFT [5].

The pulse descriptor word (PDW) of the radar pulse train signal is processed using either FFT or finite impulse response (FIR) filters [6]. Maintaining computational accuracy and the ability to derive large discrete Fourier transforms (DFTs) are important considerations when designing digital signal processing (DSP) circuits. These features pave the way for FFT circuits that offer better throughput and performance [7]. The radix-2^k multi-path FFT architecture (MSC-FFT) was designed using the single-carrier FFT architecture (SC-FFT), which allows extensive utilization of hardware. This leads to an area reduction, high data rates, and a simplification of the rotators. To achieve this, the structure of the processing element was optimized [8].

A novel circuit is introduced to calculate bit reversal for parallel data with minimal use of buffers and multiplexers, resulting in insignificant memory usage. The number of multipliers required is proportional to the memory usage [9]. A reconfigurable mixed-radix $2^k \times 3$-point feed-forward architecture is presented, which enables low-power parallel FFT. An 8-parallel 128-2048 points low-power and area-efficient FFT processor has been implemented for the 4G standard using TSMC 90 nm technology to meet the growing demand for low-cost mobile devices.

2. Literature review

To minimize the dynamic power consumption and area, recent research in low-power FFT focuses on optimizing the computational blocks, especially the multipliers: to preserve power during idle cycles, Cheng and Parhi (2021) proposed a low-complexity FFT processor using the radix-2^2 algorithm in conjunction with clock-gating and power-gating approaches. To save hardware and switching operations, Chen et al. (2020) implemented a pipelined FFT design that reuses twiddle factor multipliers. Xiong et al. (2019) have introduced approximate computation approaches in FFT multipliers that significantly reduce power consumption with negligible impact on accuracy. Booth multiplier reduces the number of incomplete products, but can make things more complicated. Dadda multipliers and Wallace trees are fast because they reduce the partial products at the same time, although they can use a lot of power. Though not always the best option for systems with limited power, the Baugh-Wooley multiplier is preferred for signed operations in DSP (2020–2023). Approximate Multipliers Intentional error-tolerance is used in emerging designs to minimize area and switching activities. These consist of error-tolerant multiplier (ETM) and lower-part or adder (LOA). The Vedic multiplication methods have been recently improved to include an energy-efficient Vedic multiplier using clock gating and operand isolation was proposed by Venkatesh et al. (2022). For high-speed applications, Ramesh et al. (2023) implemented a hybrid Vedic multiplier with partial product compression. Bit-level optimization: research has focused on bit reuse, bit truncation, and bit inversion (BI) [10].

The performance of various high-speed multipliers – such as power, area, delay, FFT suitability, and utilization – is compared, as mentioned in Table 1.

Table 1. High speed multipliers comparison.

Multiplier	Power [mW]	Area (LUT)	Delay [ns]	FFT suitability	Utilization
Booth	2.35	150	12.5	Moderate	Moderate area and speed
Wallace tree	2.10	180	10.3	High	High-speed, but complex logic increases area
Dadda	2.25	160	9.8	High	Similar to Wallace
Baugh-Wooley	2.40	170	11.0	Moderate	Best for 2's complement
Approximate LOA	1.50	120	8.5	Medium	Power-efficient, may introduce slight computational error

3. Methodology

This article presents a high-speed and area-efficient FFT processor designed for low-power applications. The processor uses a bit-inversed Vedic multiplier to achieve its high speed and parallel and pipelined architecture. The pipelined FFT method can perform the FFT analysis sequentially so that it can run in real time without interruption. However, the pipelined FFT architecture may not be suitable for processing large FFTs because it consumes a lot of hardware space and may lead to incorrect implementation results on a single field programmable gate array (FPGA) chip. On the other hand, a parallel FFT architecture may offer better performance. The M-point DFT of an M-point series $y(n)$ is defined as:

$$Y(L) = \sum_{n=0}^{M=1} y(n) \cdot X_M^{nl} \tag{1}$$

where X is the twiddle factor value; L is the frequency domain index, ranging from 0 to M - 1 samples; n is the time domain index, ranging from 0 to M - 1 samples;

To analyze $Y(L)$, we must consider that $y(n)$ can be real or complex. This requires complicated multiplications and complex additions for each value of l. Therefore, to calculate the M-point, $Y(L)$ values require M^2 complex multiplications with additions. The DIF-FFT algorithm uses the radix-r^2 FFT to divide a DFT into smaller DFTs. The input data $x(n)$ (32 bits) is represented by the 2-dimensional array of data $x(8\,l + m)$, stored in the memory block. Here, l is the length of the FFT; m is the number of stages. We calculate the DFT by performing 8-point DFTs, and the T results are multiplied by the twiddle factors. The butterfly unit calculates each DFT operation, referred to as the FFT information path for FFT calculations. The basic technique used in the FFT method is a "divide and conquer" approach. This involves decomposing M-point DFTs into successively smaller DFTs [11]. Assume that the length of $y(n)$ is even (i.e., M is divisible by 2) and $y(n)$ is decimated into 2 series of length $M/2$; computing the $M/2$-point DFT of each series requires random $(M/2)^2$ multiplications and a similar number of additions. It is possible to determine the M-point DFT of $y(n)$ by using two $M/2$-point DFTs that requires less than $M^2/2$, which facilitates storage. In this research, a high-speed and low-power FFT architecture using a modified Vedic multiplier is proposed. This architecture includes hardware components for radix-2 algorithms that can compute an FFT based on user requirements. The generalized radix-2 FFT architecture uses a data-inverted Vedic multiplier instead of a complex multiplier to reduce the partial products in complex multiplications. In this work, low-power techniques are used to reduce the power consumption using a BI algorithm. Power consumption during multiplication has become a significant design issue in FFT architecture. Therefore, minimizing the switching activity during multiplication can lead to significant savings in the overall power budget of the FFT design. To reduce the multiplication cost, a BI technique is proposed to reduce the computational complexity. In this approach, an encoder increases the run length of '1' or '0'. In this way, the internal computation of the multiplier is reduced by bypassing the successive partial product generation. The required partial product generators (PPGs) are reduced in this phase, resulting in low propagation delay [12]. An innovative architecture has been presented that uses the FFT to reduce the switching activity of the internal computations in the Vedic multiplier. This is achieved by independently controlling the bits at odd and even positions of the multiplier input, reducing the randomness of the input pattern. A significant reduction in power consumption can be achieved by identifying the run length of the '0's and '1's in the multiplier input and applying one of four possible switching schemes (full inversion, no inversion, odd inversion, and even inversion). The power consumption is higher for input patterns that switch between 0→1 and 1→0 transitions than for patterns that include 0→0 and 1→1 transitions.

Table 2 describes the BI scheme:

- In pattern 1, there is a run length of 7 1's, greater than the applied '0' full inversion.
- In pattern 2, where the number of 1's and 0's is the same, but the number of 1's in odd positions is greater than in even positions, the input pattern is switched with odd inversion.
- For pattern 3, where the number of 1's and 0's is the same, but the number of 1's in the even positions is greater than in the odd positions, the input pattern is switched with even inversion. If the number of 0's is greater than that of 1's, no BI scheme is applied.

Table 2. Different inversion schemes.

Input Pattern	Inversion scheme		Output Pattern
11111110	'1's – 7 / '0's – 1 (1>0)	Full inversion	00000001
10101001	'1's – 4 / '0's – 4 (1=0)	Odd inversion	00000011
01010110	'1's – 4 / '0's – 4 (1=0)	Even inversion	00000011
00000001	'1's – 1 / '0's – 7 (0>1)	No inversion	00000001

Fig. 1 shows the proposed BI multiplier-based FFT architecture. The architecture includes: a serial to parallel converter, a bit stream analyzer, an FFT/IFFT module, an I/O controller, and a bit selection unit as the primary functional units of the proposed FFT architecture. A serial-to-parallel converter is a digital preprocessing circuit that converts serial data into parallel data for high-speed processing in the FFT [12]. The serial input data stream is received and analyzed to find out the run length of 0's and 1's in the patterns.

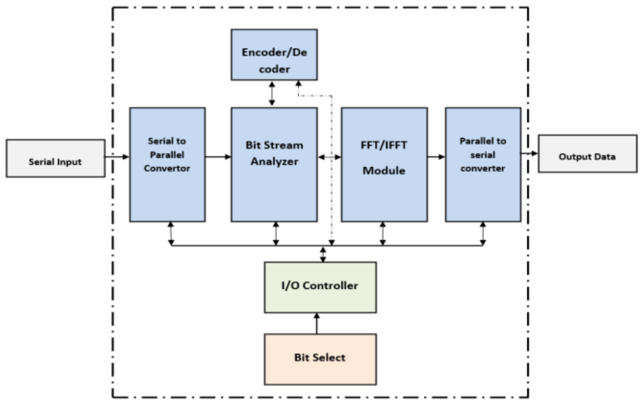

Fig. 1. FFT architecture with BI scheme.

The butterfly units are a crucial component of all FFT architectures. They are defined by the implementation of complex multiplication and addition. The efficiency of the FFT in terms of area, power, speed, and throughput is determined by the BI multiplier used. We have developed an encoder module, which is shown in Fig. 2. This module encodes the transmitting signal based on the number of zeros and ones. It defines the data to be inverted based on zeros and ones and consists of internal modules such as shift register, even counter, odd counter, comparator, and inverter. Power consumption during multiplication has become one of the main problems in the development of FFT. Therefore, minimizing the switching activity during multiplication can lead to significant power savings. To reduce the cost of multiplication, we propose a BI technique that reduces the computational cost.

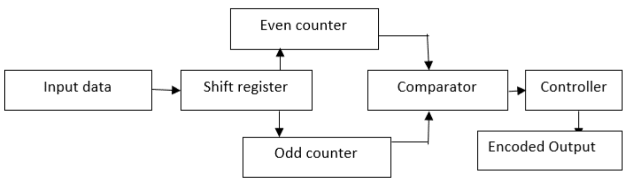

Fig. 2. Encoder module.

In this paper, we propose a high-speed, reconfigurable, and low-power FFT architecture, which is a key component in wireless communications.

Vedic multiplier algorithm

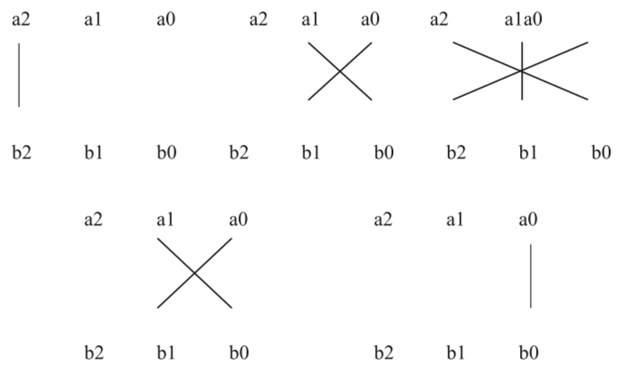

Fig. 3. A 3x3 multiplication using a Vedic multiplier.

Fig. 3 shows a 3x3 multiplication performed with a Vedic multiplier. This architecture is faster than the existing multipliers and can be used in all numerical schemes [13].

- Step 1: Multiply the least significant bit (LSB) of the multiplicand and the multiplier vertically to obtain the final result of the LSB.
- Step 2: To perform a binary multiplication, multiply the LSB of the multiplicand by the most significant bit (MSB) of the multiplier. Then multiply the MSB of the multiplicand by the LSB of the multiplier. Add the two products obtained in the previous steps. The result will give you the second bit of the result.
- Step 3: To multiply two numbers using the multiplication method, we must first multiply the most significant digit of the multiplicand by the most significant digit of the multiplier. This should be done vertically. The resulting product is then added to the previous carry, which has already been determined in the previous steps. The resulting sum and carry are then measured as the MSB of the result.

Let us consider two 3-bit binary numbers, *A* and *B*:
A=a2a1a0,
B=b2b1b0

S0 = a0b0
C1S1 = a0b1 + a1b0
C2S2 = C1 + a1b1 + a0b2 + a2b0
C3S3 = C2 + a1b2 + a2b1
C4S4 = C3 + a2b2

To understand the difference between the conventional UT-Vedic multiplier and the BI-UT Vedic multiplier, a detailed explanation is provided below:

Classical UT-Vedic Multiplier:

Let
A = 101,
B = 111

S0 = a0b0	=> 1 × 1 = 1	(CS = 1)
C1S1 = a0b1 + a1b0	=> 1 × 1 + 1 × 1 = 10	(CS = 2)
C2S2 = C1 + a1b1 + a0b2 + a2b0	=> 1 + 0 × 1 + 1 × 1 + 1 × 1 = 11	(CS = 3)
C3S3 = C2 + a1b2 + a2b1	=> 1 + 0 × 1 + 1 × 1 = 10	(CS = 2)
C4S4 = C3 + a2b2	=> 0 + 1×1 = 01	(CS = 1)

where CS – computational cost
Total computational cost = 1+2+3+2+1 = 9

BI-UT-Vedic Multiplier:

The bit switch scheme is applied to A and B:
A = 101 – 1 > 0 = 010,
B = 111 – 1 > 0 = 000

S0 = a0b0	=> 0 × 0 = 1	(CS = 0)
C1S1 = a0b1 + a1b0	=> 0 × 0 + 1 × 0 = 01	(CS = 1)
C2S2 = 0 + a1b1 + a0b2 + a2b0	=> 1 + 1 × 0 + 0 × 0 + 0 × 0 = 01	(CS = 1)
C3S3 = 0 + a1b2 + a2b1	=> 1 + 1 × 0 + 0 × 0 = 01	(CS = 1)
C4S4 = C3 + a2b2	=> 0 + 0 × 0 = 0	(CS = 0)

Total computational cost = 0+1+1+1+0 = 3

Based on the computations performed above, the bit-inversed Vedic multiplier has a computational cost of 3, while the conventional Vedic multiplier has a computational cost of 9. This proves that the proposed Vedic multiplier can significantly reduce the power consumption and delay in the FFT architecture without compromising the resource utilization as stated in reference [14].

4. Results

In this section, the experimental results and discussion of the proposed BI-Vedic multiplier based FFT architecture and a detailed performance analysis are described. The performance analysis is performed for both FPGA and ASIC. The simulation is performed in ModelSim. The proposed method is synthesized using CADENCE Genus solution and compiled in CADENCE INNOVUS for RTL to GDSII conversion based on 90 nm technology [15]. Fig. 4 shows the simulation output waveform of the BI scheme using ModelSim.

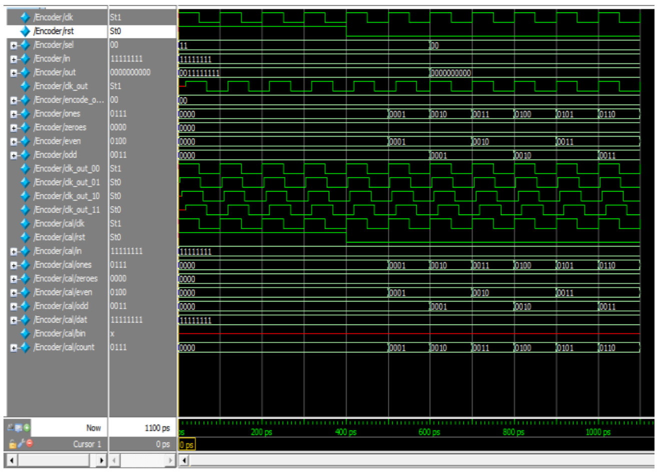

Fig. 4. Simulation waveform – BI scheme.

5. Performance evaluation of FFT

This section discusses the performance analysis of the BI multiplier FFT (BIM-FFT) architecture technique. The performances are analyzed in four different Xilinx FPGA families, namely Spartan-6, Virtex-4, Virtex-5, Virtex-6, and Zynq. The analysis focuses on various aspects such as the number of slices, flip-flops, look-up tables (LUTs), operating frequency, and I/O delay in the FFT architecture.

The hardware utilization in Spartan-6, Virtex-4, Virtex-5, and Zynq FPGA devices is analyzed in Table 3. In addition, the communication performance of four Xilinx devices is shown in Table 4. The number of registers, LUTs, and slices was measured using Xilinx ISE. It was found that the number of slices used in the Virtex-6 and dyn devices is lower compared to the other FPGA families. Similarly, LUTs used by the Virtex-6 and Zynq devices are also lower compared to other FPGA families. However, the Spartan-6 FPGA operated at a higher frequency and had a higher throughput than any other FPGA family. In addition, the delay in the Spartan-6 FPGA was lower compared to Virtex-4, Virtex-5, and Virtex-6 FPGA devices [16].

Table 3. Performance analysis of the proposed BIM-FFT.

Family	Registers	LUT's	Slices
Spartan-6 - xc6slx9	1179/11440	1126/5720	475/1430
Virtex-4 - xc4vsx55	1246/49152	1618/49152	1187/24576
Virtex-5 - xc5vsx50t	1239/32640	1548/32640	710/8160
Virtex-6 - xc6vlx75t	106/93129	266/46560	95/11640
Zynq - xc7z010	106/35200	265/17,600	85/4400

Table 4. Performance analysis of the proposed DPR-FFT architecture with frequency and delay.

Family	Frequency [MHz]	Delay [ns]	Power [mW]
Spartan-6 - xc6slx9	335.768	2.97	64.15
Virtex-4 - xc4vsx55	190.81	5.241	76.49
Virtex-5 - xc5vsx50t	188.79	5.297	75.38
Virtex-6 - xc6vlx75t	246.77	4.052	79.01
Zynq - xc7z010	261.55	3.823	64.23

Fig. 5 shows a performance comparison graph of an FPGA for the BIM-FFT architecture. The FPGA performance is more efficient with the Virtex-6 and Zynq FPGAs. As can be seen in Fig. 7, the Zynq FPGA and Virtex-6 FPGA families consume a lower number of registers, LUTs, and slices compared to other FPGA families for the DPR-FFT architecture. In addition, the frequency is lower for Zynq and Virtex-6 FPGAs. Virtex-4 and Virtex-5 FPGAs consume more LUT resources compared to other families.

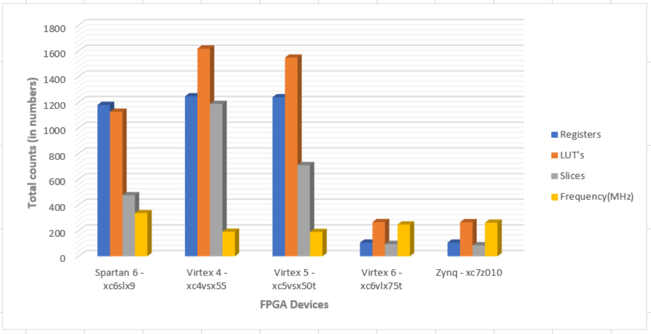

Fig. 5. Performance analysis of BIM-FFT for different FPGA families.

From the data presented in Fig. 6, it is evident that the BIM-FFT architecture is capable of operating at high speeds in Spartan-6 FPGA. The delay time of this architecture is 2.97 ns, which is lower than that of the Zynq and Virtex FPGA families. However, it should be noted that the delay time in Zynq and Virtex-6 FPGAs for R2RMDC FFT is even lower.

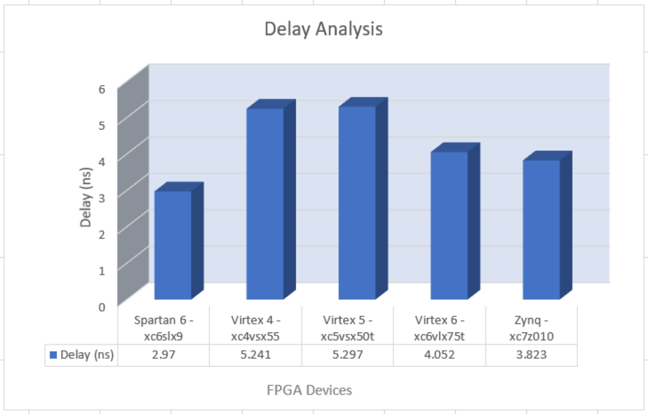

Fig. 6. Delay-FFT – Delay analysis for different FPGA family.

The functional efficiency of the proposed BIM-FFT architecture is compared with the dynamically reconfigurable FFT processors presented by Sivakumar et al. (2018). The existing FFT architectures are executed to verify the parameter measurement.

Table 5. Performance comparison for FFT.

Device	Methodology	LUTs	Flip-flops	Slices	Frequency [MHz]	Delay [ns]	Power [mW]
ASAP 7 nm	R2MDC	12686	7416	4005	105.234	2.835	128
	Proposed	9548	7239	3634	188.790	1.914	75.38

When implementing FPGAs, parameters such as LUTs, flip-flops, slices, frequency, and throughput are critical to the performance of a system. This phase is crucial for the selection of the appropriate device. The proposed FPGA provides a configurable structure through customizable modules, connected by programmable control elements and enclosed by an input and output block (IOB). Table 5 shows the results of the existing DPR and R2RMDC-OMS FFT architectures. The proposed FFT design was synthesized and compared with theVirtex-5 FPGA. Table 5 shows that the DPR-FFT is compared with an R2MDC FFT proposed by Sivakumar et al. (2018), in which the number of slice LUTs (area) is minimized to 9548 and the delay is reduced from 2.835 ns to 1.914 ns. The proposed BIM-FFT utilizes an 8:1 ratio of logical resources with a 38 % reduction in delay and operates at 58 % higher speed [17].

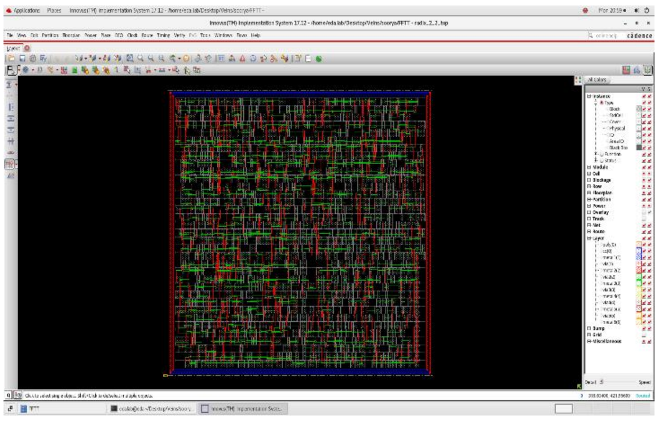

Fig. 7. Physical design of parallel and pipelined CTS-BIS FFT.

Fig. 7 shows the ASIC implementation for the Vedic FFT architecture. The parallel and pipelined FFT with a 7-BI scheme was implemented in CMOS 90 nm technology. The Cadence Genus synthesis solution was used to generate reports on power, area, and timing. The proposed BI-Vedic FFT operates with an input size of 32 bits at an operating frequency of 50 MHz. It occupies an area of 4.91 mm^2 and consumes 54 mW at an operating voltage of 1.8 V.

6. FUTURE WORK

Even though the current implementation of the BI-Vedic Multiplier shows remarkable gains in power and delay, there are a number of improvements that could be explored.

Pipelining and Parallelization:

In high-speed systems, adding pipelined stages to the multiplier can help further reduce the latency in the critical path and increase throughput.

Hybrid Designs:

For certain workloads, performance can be improved by combining the bit-inverted approach with additional optimization strategies such as Booth encoding or carry save adders.

Dynamic BI Logic:

Using data-dependent inversion logic instead of static BI could adaptively reduce switching activity depending on the operand pattern.

FPGA/ASIC Custom Optimization:

By using vendor-specific tools and logic usage, the design can be tailored to specific hardware platforms (such as Xilinx, Intel, or low-power ASICs) to unlock additional area and power gains. The advantages of the proposed multiplier—low power, speed, and area efficiency—make it a compelling contender for use in a number of areas outside of FFT processing.

Real-Time DSP Systems:

Software-defined radios can benefit from fast and energy-efficient multiplication of modulation/demodulation blocks, convolutional neural networks (CNNs), and FIR/IIR filters.

IoT and Edge AI Devices:

Without sacrificing functionality, the proposed design could extend the battery life of devices with limited power consumption, such as wearable technology or smart sensors.

Image and Video Processing Pipelines:

The architecture is suitable for hardware accelerators in multimedia applications, as tasks such as scaling, filtering, and color space conversion require frequent multiplications.

Higher Bit-Width Implementations:

The scalability of the multiplier for more complex applications, such as FFT cores in radar or biomedical signal processing, would be confirmed by testing and evaluating 32-bit and 64-bit versions.

Multi-Operand Multiplication:

The architecture could be applicable to AI/ML hardware if extended for multiplication with multiple operands (e.g., for fused multiply-accumulate units).

7. CONCLUSION

A novel architecture for BIM-FFT with maximum throughput and minimum latency has been proposed and implemented in Xilinx Virtex-7 FPGA. The performance of the proposed FFT architecture was analyzed in terms of throughput, speed, and resource utilization. The power consumed by the BI-Vedic multiplier has been reduced, which contributes to an overall reduction in the power consumption of the complex FFT architecture. An innovative encoding technique, called data inversion scheme, was presented to minimize the switching activity of the system-level data inputs. The FFT processor presented in this research utilizes only 8 % and 9 % of the available LUTs and has a 56 % lower delay compared to previous research, making it suitable for IoT 4G and 5G applications.

REFERENCES

[1] Adámek, K., Novotný, J., Thiyagalingam, J., Armour, W. (2021). Efficiency near the edge: Increasing the energy efficiency of FFTs on GPUs for real-time edge computing. *IEEE Access*, 9, 18167-18182. https://ieeexplore.ieee.org/document/9330509

[2] Andersson, R., Garrido, M. (2020). Using rotator transformations to simplify FFT hardware architectures. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 67 (12), 4784-4793. https://doi.org/10.1109/TCSI.2020.3006253

[3] Dai, J., Yin, H. (2020). Design and realization of non-radix-2 FFT prime factor processor for 5G broadcasting in release 16. In *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 406-409. https://doi.org/10.1109/ISCID51228.2020.00098

[4] Garrido, M., Malagón, P. (2021). The constant multiplier FFT. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 68 (1), 322-335. https://doi.org/10.1109/TCSI.2020.3031688

[5] Hsu, S.-C., Huang, S.-J., Chen, S.-G., Lin, S.-C., Garrido, M. (2020). A 128-point multi-path SC FFT architecture. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. https://doi.org/10.1109/ISCAS45731.2020.9180883

[6] Garrido, M., Huang, S.-J., Chen, S.-G., Gustafsson, O. (2016). The serial commutator FFT. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 63 (10), 974-978. https://doi.org/10.1109/TCSII.2016.2538119

[7] Juliano, J., Lin, J., Erdogan, A. George, K. (2020). Radar pulse on pulse identification parallel FFT and power envelope algorithm. In *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 0609-0613. https://doi.org/10.1109/UEMCON51285.2020.9298132

[8] Priya, A. L., Deepthi, V. M. (2022). Design and implementation of area efficient pipelined FFT processor. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12 (11), 7245-7251. https://doi.org/10.17762/turcomat.v12i11.11940

[9] Kanders, H., Mellqvist, T., Garrido, M., Palmkvist, K., Gustafsson, O. (2019). A 1 million-point FFT on a single FPGA. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 66 (10), 3863-3873. https://doi.org/10.1109/TCSI.2019.2918403

[10] Kang, H., Lee, J., Kim, D. (2021). HI-FFT: Heterogeneous parallel in-place algorithm for large-scale 2D-FFT. *IEEE Access*, 9, 120261-120273. https://doi.org/10.1109/ACCESS.2021.3108404

[11] Lin, C.-H., Liu, J.-C., Yang, P.-K. (2020). Performance enhancement of GPU parallel computing using memory allocation optimization. In *2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM)*. IEEE. https://doi.org/10.1109/IMCOM48794.2020.9001771

[12] Madanayake, A., Cintra, R. J., Akram, N., Ariyarathna, V., Mandal, S., Coutinho, V. A., Bayer, F. M., Coelho, D., Rappaport, T. S. (2020). Fast radix-32 approximate DFTs for 1024-beam digital RF beamforming. *IEEE Access*, 8, 96613-96627. https://doi.org/10.1109/ACCESS.2020.2994550

[13] Garrido, M. (2019). Multiplexer and memory-efficient circuits for parallel bit reversal. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 66 (4), 657-661. https://doi.org/10.1109/TCSII.2018.2880921

[14] Tsai, W.-L., Chen, S.-G., Huang, S.-J. (2019). Reconfigurable radix-2^k×3 feedforward FFT architectures. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. https://doi.org/10.1109/ISCAS.2019.8702346

[15] Wang, J., Li, X., Fan, G., Tuo, Z. (2019). A parallel radix-2^k FFT processor using single-port merged-bank memory. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE. https://doi.org/10.1109/ISCAS.2019.8702088

[16] Tsai P.-Y., Lin, C.-Y. (2011). A generalized conflict-free memory addressing scheme for continuous-flow parallel-processing FFT processors with rescheduling. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 19 (12), 2290-2302. https://doi.org/10.1109/TVLSI.2010.2077314

[17] Luo, H.-F., Liu, Y.-J., Shieh, M.-D. (2015). Efficient memory-addressing algorithms for FFT processor design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 23 (10), 2162-2172. https://doi.org/10.1109/TVLSI.2014.2361209

Received March 18, 2024
Accepted May 7, 2025