International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 09 | Sep 2025 www.irjet.net p-ISSN: 2395-0072

Optimized Radix-2 FFT Processor for FPGA: A VHDL Design and Synthesis Study

Dr. Sarita Sanap[1], Shrushti Baviskar[2]

1Department of Electronics and Computer, MIT, Chht. Sambhajinagar, Maharashtra, India,
2Post Graduate Student, MIT, Chht. Sambhajinagar, Maharashtra, India

***

Abstract - *The paper details the design and implementation of a high-speed Fast Fourier Transform (FFT) processor at the Register Transfer Level (RTL) using VHDL, optimized for FPGA deployment. The FFT is integral to digital signal processing, and the need for efficient, high-throughput hardware solutions is emphasized by the growing demands in applications like 5G communications, audio/image processing, and medical imaging. The project focuses on a radix-2 approach for simplicity and scalability, developing essential components such as the butterfly computation unit, twiddle factor ROM, shift registers, and a finite state machine controller. Xilinx tools were used for simulation and synthesis, verifying correctness and resource efficiency. The modular, pipelined architecture achieves high throughput and is well suited for real-time DSP systems, bridging theoretical concepts and practical hardware realization. Future work may extend the design to support higher-point FFTs and reconfigurable architecture.*

***Key Words*: Fast Fourier Transform (FFT), Digital Signal Processing (DSP), FPGA (Field-Programmable Gate Array), VHDL (VHSIC Hardware Description Language), Radix-2 algorithm, Butterfly unit, Twiddle factor, Register Transfer Level (RTL)**

1.INTRODUCTION

The Fast Fourier Transform is a cornerstone algorithm in Digital Signal Processing, which is crucial for analyzing and transforming signals between the time and frequency domains. Its efficiency has made it indispensable in a vast array of applications, including telecommunications, audio and image processing, medical imaging, and radar systems. Modern DSP systems, particularly those in rapidly evolving fields such as 5G communication, require high-speed, low-latency, and resource-efficient implementations of the FFT. The increasing complexity and performance demands of these applications necessitate robust and optimized hardware designs for FFT processors.

Despite the widespread use of FFT, the challenges of realizing these algorithms efficiently in hardware remain significant. Achieving high throughput while minimizing resource consumption and power is a persistent design goal, particularly for embedded and real-time systems. Existing research has explored various optimization techniques, including different radix algorithms, pipelined architectures, and multiplier-free designs, often leveraging Field-Programmable Gate Arrays due to their flexibility and parallelism.

This project directly addresses the need for optimized FFT hardware by focusing on the design and simulation of a high-speed FFT processor using a Very High-Speed Integrated Circuit Hardware Description Language. The primary objectives of this study are as follows:

- To design the Register Transfer Level architecture of an FFT processor using VHDL.
- Individual VHDL modules for essential components, including the butterfly unit, twiddle factor ROM, shift registers, and control unit, were developed.
- The design was simulated and rigorously verified using Xilinx tools to ensure functional correctness and performance.
- To synthesize the FFT processor for FPGA implementation, with a focus on achieving high speed, low latency, and efficient resource utilization.

Through this project, we aim to bridge theoretical DSP concepts with practical hardware realization, demonstrating a VHDL-based RTL design that yields an optimized digital hardware solution for critical signal-processing tasks. The remainder of this paper details the theoretical background of the Fourier Transform, architectural design of the proposed FFT processor, implementation and verification methodologies, and discussion of the results obtained from synthesis and simulation

II. Literature Review

Fast Fourier Transform (FFT) has been extensively studied in hardware design because of its fundamental role in various digital signal processing applications. This review explores key developments and architectural considerations in FFT processor design, particularly focusing on FPGA implementations and their relevance to modern communication systems.

Early efforts in FFT hardware design, such as those by Josue Saenz S. et al., demonstrated the feasibility of implementing radix-2 DIT FFT processors on FPGAs for fixed-point transformations (e.g., 16- and 32-point). This study highlighted the use of custom VHDL packages for complex number representation and effective data movement using shift registers to optimize FPGA resource utilization, confirming the practicality of small-scale, accurate FFT processors for fixed-size transformations.

As communication technologies have evolved, the demand for more adaptable and higher-performance FFT solutions has become paramount. Daud Khan et al. addressed this need by proposing a reconfigurable and scalable FFT processor specifically optimized for LTE and 5G networks. Their architecture features a run-time configurable butterfly structure, pipelining for high throughput, in-place memory computation, and dynamic twiddle factor generation, achieving throughputs of over 245.76 Msamples/s. Complementing this, Bautista et al. introduced efficient serial butterfly structures for FFT architectures with non-power-of-two lengths, catering to the flexible demands of 5G and beyond-5G systems. Their design minimizes hardware resources and memory overhead while maintaining a competitive throughput.

Optimization techniques for hardware efficiency are a recurring theme in the design of FFT processors. Taesang Cho and Hanho Lee presented a modified Radix-25 FFT processor for high-speed, low-power Wireless Personal Area Network applications, emphasizing a simplified twiddle factor approach and a multi-path delay commutator to reduce complexity. In another significant contribution, Godi et al. explored multiplier-free parallel-pipelined FFT designs for FPGAs. By employing addition-based twiddle factor calculations and digit slicing, they significantly reduced the area and power consumption, making these designs suitable for resource-constrained or battery-powered DSP applications.

Beyond traditional DSP, FFT processors are becoming increasingly integral to emerging applications. Chowdary A. et al. investigated hybrid radar and communication systems where FFT-based signal processing enables joint sensing and data transmission, demonstrating the potential for shared hardware and multi-functional roles in 6G and IoT infrastructure. Furthermore, the role of FFT in OFDM-based systems is critical, as shown by Liu et al. in their work on low-complexity Peak-to-Average Power Ratio reduction using real-valued neural networks, where FFT plays a key role in signal transformation. Similarly, Baig et al. discussed new multi-carrier waveform designs for 5G and future wireless systems, highlighting the importance of FFT/IFFT-based architectures for flexible subcarrier spacing and dynamic bandwidth allocation.

Khan et al. (2025) proposed a run-time reconfigurable FFT engine capable of performing Radix-8, Radix-4, Radix-3, and Radix-2 computations, targeting the high-throughput and flexible bandwidth demands of LTE and 5G networks. Their architecture addresses the challenge of supporting both power-of-two and non-power-of-two FFT sizes such as 1536, which is essential for LTE bandwidth allocation. By sharing resources across different butterfly configurations and employing in-place multi-bank memory with dynamic twiddle factor generation, the design optimizes hardware utilization and reduces latency while maintaining high throughput. The design achieves a throughput exceeding 245 Msamples/sec at a 310 MHz clock frequency, validated through simulation and synthesis for FPGA platforms. This work highlights the importance of adaptability, efficient memory management, and multi-radix computation to meet the evolving requirements of next-generation wireless communication systems.

In summary, the reviewed literature highlights a clear progression in FFT processor design from fixed and small-scale implementations to scalable, reconfigurable, and application-specific architectures. Key advancements include the development of optimized algorithms, efficient hardware mapping techniques (e.g., multiplier-free designs and pipelining), and integration into complex systems beyond traditional signal processing. FPGA technology, due to its inherent flexibility and parallelism, has consistently served as a crucial platform for implementing these advancements, providing a robust foundation for high-speed, low-latency, and resource-efficient FFT solutions. This extensive body of work underpins the focus of the current project on designing and implementing a high-speed RTL-level FFT processor using VHDL and Xilinx tools.

III. Architectural Design of the FFT Processor

The design of the Fast Fourier Transform processor outlined in this project emphasizes modularity, high-speed computation, and efficient resource utilization, targeting Field-Programmable Gate Array implementation. The overall operation of the processor was partitioned into three primary processes: Data Input, FFT Computation, and Data Output. This sequential processing allows a streamlined flow from raw sampled data to transformed frequency-domain results.

1. Overall Architecture

The architecture of the FFT processor is built around a single radix-2 butterfly unit optimized for iterative computation. The core components include the following:

- **Butterfly Processing Element**: The fundamental computational unit.

- **Dual-Port FIFO RAM**: For data storage and retrieval.
- **Coefficient ROM**: To store pre-computed twiddle factors.
- **Controller**: To manage the operational flow.
- **Address Generation Unit**: To Supplies the correct memory and ROM addresses.
- **Cycles Unit**: This synchronizes internal operations across different clock phases.
- **Counter Unit**: For timing-specific delays within the processing cycle.

Data pathways within the processor are configured for 32-bit signed fractions, whereas coefficients are stored as 32-bit words, ensuring appropriate precision for DSP operations.

2. Key Components

2.1 Butterfly Processing Element

The butterfly is the core computational block responsible for performing a two-point FFT. It operates in an in-place manner, meaning that the results overwrite the input data in memory, thereby optimizing memory usage. Each butterfly computation requires four clock cycles and has a latency of five cycles, which accounts for input dependencies and pipelined RAM operations. The BPE architecture consists of one multiplier and two adders, enabling the complex additions and multiplications essential for the FFT algorithm.

2.2 Dual-Port FIFO RAM

The RAM serves as the primary storage for both the input data and intermediate FFT computation results. Its dual-port nature allows simultaneous read and write operations, which is crucial for the pipelined execution of the FFT algorithm. During the data input process, the sampled data were written into the RAM. For the FFT computation, data are read, transformed by the butterfly unit, and the results are written back to the same memory locations. Finally, during the output process, the data are read from the RAM and provided to the external environment, with bit-reversal applied to ensure the correct output order.

2.3 Coefficient ROM

The ROM stores the sine and cosine coefficients (twiddle factors) required by the butterfly unit for complex multiplication. The Address Generation Unit provides the correct address to the ROM, ensuring that the appropriate twiddle factor is supplied for each butterfly operation.

2.4 Address Generation Unit

The AGU is a critical component responsible for generating accurate read and write addresses for the RAM and the read addresses for the coefficient ROM. Its functions are highly dependent on the current operational mode (input, FFT computation, or output) and specific stage of the FFT computation. The AGU comprises several subunits:

- **Butterfly Generator**: Tracks which butterfly is being computed within a stage.
- **Stage Generator**: Keeps track of the current FFT stage (e.g., for an 8-point FFT, there are three stages).
- **Stage done_IO done block**: Generates control signals indicating the completion of I/O operations, stages, or the entire FFT computation.
- **IO-Address Generator**: Manages RAM addressing during data input and output, including bit-reversed addressing for the output.
- **Base Index Generator**: Complex logic to generate addresses for the butterfly input data (A and B) based on the stage, butterfly count, and cycles.
- **Shifters**: Implement read address shifting necessary to account for the pipeline latency between reading input data and writing back the transformed output.
- **ROM Address Generator**: Provides the coefficient ROM with the correct addresses based on the signal flow graph.

2.5 Controller

The Controller acts as a Finite State Machine that governs the overall operation of the FFT processor. It transitions through various states (e.g., rst1 to rst7) to manage the Data Input, FFT Computation, and Data Output processes. It supplies critical mode information to the AGU and other units, ensuring the synchronized and correct execution of the FFT algorithm.

IV. Technology Stack

The design and implementation of this FFT processor leverage a specific technology stack to achieve its objectives, as summarized in Table 4.1.

Technology Stack for FFT Processor Design			
Layer	**Tools**	**Purpose**	**Layer**
Algorithm Design	VHDL	Signal modelling, validation	Algorithm Design
HDL Design	VHDL	RTL architecture of FFT	HDL Design
Simulation	ISE Simulator	Verify functionality and timing	Simulation
Synthesis	Xilinx Vivado / ISE	FPGA compilation & resource estimation	Synthesis
Synthesis	Xilinx Vivado / ISE	FPGA compilation & resource estimation	Synthesis
Target Device	Spartan-3 FPGA	Hardware platform for deployment	Target Device
Waveform Analysis	ISE Waveform Viewer	Analyze input-output timing and correctness	Waveform Analysis

Table 1. Technology Stack for FFT Processor Design

This architectural breakdown provides a clear understanding of how the FFT processor is designed to handle high-speed signal-processing tasks efficiently.

V. Results

The Fast Fourier Transform processor, designed using VHDL, underwent rigorous verification through simulation and synthesis using Xilinx tools. This section presents the key results, including the Register Transfer Level (RTL) schematics, behavioral simulation waveforms, and detailed synthesis reports, demonstrating the successful implementation and performance characteristics of the proposed architecture.

5.1 RTL Schematic

The Register Transfer Level schematic provides a visual representation of the hardware architecture of the FFT processor, as automatically generated by Xilinx ISE 10.1.

- **Overall Processor Schematic**: The top-level RTL schematic illustrates the interconnection of multiple modular blocks, including butterfly units, twiddle read-only memory (ROM), shift registers, and finite state machine controller. This structural integration confirms the successful mapping of the VHDL code to a hardware representation, thereby validating the modular-design approach.
- **Butterfly Unit Schematic**: A detailed schematic of the internal structure of the Butterfly Unit, the core computational element, shows its components: multipliers and adders. This unit performs complex addition and multiplication on the input samples. The schematic highlights the pipelining paths, indicating a design optimized for speed and reduced latency.

5.2 Behavioral Simulation Waveform

Behavioral simulation was conducted using the Xilinx ISim simulator to verify the functional correctness of the fft_controller module.

- **Controller Validation**: The waveform shows the clk, reset, start, done, state, and stage_sel signals. Critically, it demonstrates that at approximately 500 ns, the start signal triggers the controller to sequence through the FFT stages, and the done signal asserts upon completion, thereby confirming the FSM logic functions.

5.3 LUT 3 Analysis

The implementation utilizes Look-Up Tables for combinatorial logic. The LUT3 example illustrates how logic gates are synthesized to achieve specific functions.

- **Logic Schematic**: The schematic for LUT3 demonstrates the use of NOT, AND, and OR gates to combine input terms, which is an autogenerated realization based on the defined truth table.

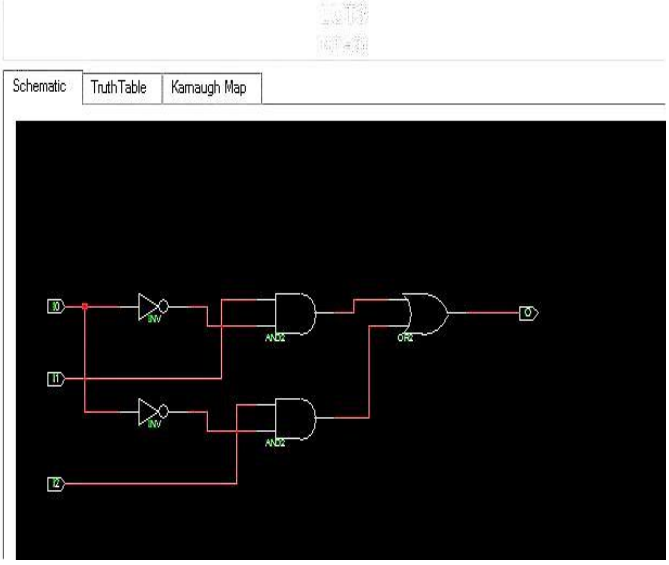
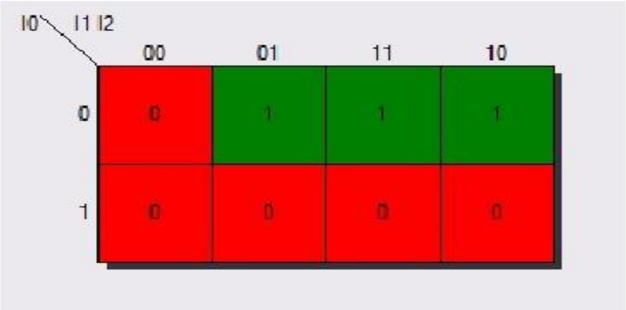


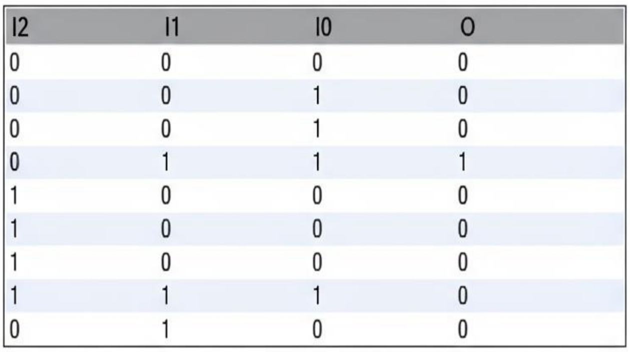
Fig. (a) Logic Schematic

- **Truth Table and K-Map**: The provided truth table defines the output for all possible combinations of threehe 3 input bits. A Karnaugh Map further visualizes the LUT3's behavior, showing outputs (1s and 0s) for various input combinations, confirming specific logic conditions for an output of 1.



I0 \ I1 I2	00	01	11	10
0	0	1	1	1
1	0	0	0	0

Fig (b) K-Map



I2	I1	I0	O
0	0	0	0
0	0	1	0
0	0	1	0
0	1	1	1
1	0	0	0
1	0	0	0
1	0	0	0
1	1	1	0
0	1	0	0

Fig (c) Truth Table

5.4 Synthesis Report

The synthesis report generated by Xilinx provides crucial insights into the resource utilization and timing performance of the design when mapped to the target FPGA (Spartan-3 3s1000fg320-4).

- **Finite State Machine Details**
    - States: 5
    - Transitions: 6
    - Inputs: 1
    - Outputs: 5
    - Clock: clk
    - Reset: reset (asynchronous)
    - Encoding: Automatic
- **Resource Utilization**:
    - **Logic Elements**: The design utilized 10 Basic Elements (LUT2, LUT2_L, LUT3, LUT3_L, LUT4, and MUXF5).
    - **Flip-Flops/Latches**: Six FDC flip-flops were used for state elements and registers (done, stage).
- **Clock Buffers**: One BUFGP was used for the clock signal.
- **IO Buffers**: Five IO buffers were used for external connections.
- **Device Utilization Summary (for** 3s1000fg320-4**)**:
    - **Number of Slices:** 5 out of 7680 (0%)
    - **Number of Slice Flip Flops:** 6 out of 15360 (0%)
    - **Number of 4 input LUTs:** 9 out of 15360 (0%)
    - **Number of Bonded IOBs: 6 out of 221 (2%)**
    - **Number of GCLKs:** 1 out of 8 (12%)
- **Timing Summary**:
    - Speed Grade: -4
    - Minimum period: 3.429 ns (corresponding to a maximum clock frequency of approximately 291.6 MHz)
    - Minimum input arrival time before clock: 3.508 ns
    - Maximum output required time after clock: 7.241 ns
    - Maximum combinational path delay: No path found

These results indicate that the design is highly efficient in terms of resource utilization, occupying a negligible percentage of the target FPGA logic cells, flip-flops, and LUTs. The achieved minimum period of 3.429 ns demonstrates the high-speed capability of the implemented FFT controller.

5.5 Programming File Generation

The final step in the FPGA implementation flow is the generation of a programming file (bitstream). This process converts the synthesized design into a binary file that can be loaded onto the target FPGA. The key options configured during bitstream generation include compression, readback, and various pin and cycle settings.

VI. Conclusion

This project successfully demonstrates the comprehensive process of modelling and designing a high-speed Fast Fourier processor at the Register Transfer Level using VHDL. The core of this work involved a meticulous analysis and implementation of the FFT algorithm, a fundamental component in digital signal processing, adapted to a hardware-friendly architecture based on the radix-2 approach for both simplicity and scalability.

Key components of the FFT processor, including the butterfly computation unit, twiddle factor ROM, shift registers, and finite state machine controller, were meticulously designed in VHDL and structurally integrated. The functionality and performance of the design were rigorously verified through behavioral simulations using Xilinx tools, confirming its correctness and practical viability for real-world applications.

The adoption of fixed-point arithmetic in the design ensured optimal resource efficiency while maintaining acceptable levels of precision, which is necessary for DSP operations. Furthermore, the pipelined and modular nature of the architecture contributes to the high throughput, rendering the design well-suited for integration into various real-time DSP systems, such as those used in OFDM, audio and image processing, and communication receivers.

In essence, this project serves as a bridge between theoretical DSP concepts and their practical hardware realizations. This demonstrates how a VHDL-based RTL design can be leveraged to produce an optimized digital hardware solution. The successful completion of this endeavor not only fulfils the project's primary objectives but also establishes a solid foundation for future enhancements, including the potential for supporting higher-point FFTs, exploring radix-4 or split-radix implementations, and incorporating dynamic FFT size configurability.

VII. REFERENCES

[1] S. Josue Saenz, “FPGA design and implementation of radix-2 Fast Fourier Transform algorithm with 16 and 32 points,” *2015 IEEE International Autumn Meeting on Power, Electronics and Computing*.

[2] Daud Khan, Latif Jan, Mohammad Haseeb Zafar, “Optimized FFT Designs for High-Performance LTE and 5G Networks,” *Arabian Journal for Science and Engineering*, received: 24 May 2024 / Accepted: 23 January 2025 Research Article-Electrical Engineering.

[3] Víctor Manuel Bautista, Mario Garrido, Senior Member, IEEE, and Marisa López-Vallejo, Senior Member, IEEE, “Serial Butterflies for Non-Power-of-Two FFT Architectures in 5G and Beyond,” *IEEE Transactions on Circuits and Systems—I: Regular Papers*, Vol. 70, No. 10, October 2023.

[4] Taesang Cho, Hanho Lee, “A High-Speed Low-Complexity Modified Radix−25 FFT Processor for High Rate WPAN Applications,” *IEEE Transactions on Very Large-Scale Integration Systems* (Volume: 21, Issue: 1, January 2013).

[5] Daud Khan; Latif Jan; Mohammad Haseeb Zafar; Optimized FFT Designs for High-Performance LTE and 5G Networks https://doi.org/10.1007/s13369-025-10009-z