International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 06 | June - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Design of Mixed Radix FFT Algorithm Using FPGA

N Umapathi[1], Ramana[2], Nimisha [3], Sangeetha[4], Anusha[5]
[1]Professor, [2, 3, 4,5] UG Final year
[1, 2, 3, 4,5] Department of Electronics and Communication Engineering,
[1, 2, 3, 4,5] Jyothishmathi Institute of Technology & Science, Karimnagar, Telangana.
nrumapathi@gmail.com

ABSTRACT: FFT algorithms like radix-2 are efficient for signal lengths that are powers of two, many practical applications require FFT computation for arbitrary input sizes. To address this limitation, this project presents the design and FPGA-based implementation of a Mixed Radix FFT algorithm, which accommodates composite input sizes by combining multiple radix strategies (e.g., radix-2, radix-3, radix-5). The proposed architecture leverages the parallel processing capabilities of Field Programmable Gate Arrays (FPGAs) to optimize performance, reduce latency, and improve resource utilization. The design is implemented using hardware description languages (HDLs) such as VHDL or Verilog, synthesized and verified on a suitable FPGA development board. Key aspects of the design include modular decomposition, pipelining, and memory-efficient data management. he results demonstrate that the Mixed Radix FFT design on FPGA offers high flexibility and computational efficiency, making it well-suited for real-time signal processing applications in embedded systems.

INTRODUCTION

The Fast Fourier Transform (FFT) stands out as one of the most critical and widely used algorithms. It enables efficient computation of the Discrete Fourier Transform (DFT), which converts a time-domain signal into its frequency-domain representation. This transformation is fundamental in a wide range of applications. Traditional FFT algorithms such as Radix-2 and Radix-4 are optimized for input sizes that are exact powers of two. However, in many practical scenarios, the input signal length is not always a power of two. In such cases, using fixed-radix FFT algorithms leads to inefficient computation due to unnecessary zero-padding or algorithmic limitations. This inefficiency can result in increased computation time, higher resource usage, and reduced real-time processing capability. To address these limitations, the Mixed Radix FFT algorithm offers a more flexible approach by allowing the input size to be any composite number. It achieves this by decomposing the FFT into smaller DFTs based on the prime factors of the input size (e.g., 2, 3, 5, etc.). This adaptability not only improves the computational efficiency but also broadens the applicability of the FFT algorithm in systems dealing with arbitrary-length signals.

The need for real-time processing, especially in embedded systems and high-speed communication devices, demands hardware implementations that can meet stringent performance requirements. Field Programmable Gate Arrays (FPGAs) are ideal platforms for such applications due to their parallel processing capabilities, reconfigurability, and energy efficiency. Implementing the Mixed Radix FFT algorithm on an FPGA allows for significant acceleration of signal processing tasks while maintaining flexibility in input size handling.

Project focuses on the design, implementation, and analysis of a Mixed Radix FFT algorithm using FPGA technology. The primary goals are to develop an architecture that supports various input sizes, optimizes resource utilization, and achieves high throughput suitable for real-time applications. The design is carried out using Hardware Description Languages (HDLs) such as VHDL or Verilog and synthesized using industry-standard tools like Xilinx Vivado or Intel Quartus. By combining the flexibility of the Mixed Radix algorithm with the speed and parallelism of FPGA hardware, this project aims to deliver an efficient and robust solution for next-generation signal processing systems. The Fast Fourier Transform (FFT) is a fundamental algorithm used to efficiently compute the Discrete Fourier Transform (DFT), which transforms time-domain signals into their frequency components. While the conventional radix-2 FFT algorithm is widely used, it requires the input length to be a power of two, limiting its flexibility.

The Mixed Radix FFT algorithm overcomes this limitation by allowing the FFT length NNN to be factored into smaller radices (such as 2, 3, 4, 5), enabling efficient computation for arbitrary composite lengths. This makes it highly versatile for practical applications where signal lengths vary. Field-Programmable Gate Arrays (FPGAs) provide an excellent hardware platform to implement the Mixed Radix FFT due to their parallel processing capabilities, reconfigurability, and high throughput. By leveraging the FPGA's ability to perform multiple operations concurrently and pipeline data efficiently, the mixed radix FFT design achieves significant acceleration compared to software implementations. The FPGA design typically includes modular butterfly units corresponding to each radix stage, twiddle factor multipliers, and memory blocks for data storage and reordering. This architecture supports scalable and flexible FFT sizes while optimizing resource utilization and power consumption, making it suitable for real-time signal processing in communications, radar, and multimedia applications.

LITERATURE REVIEW

Although numerous Fast Fourier Transform (FFT) architectures have been proposed for computing real-valued FFTs (RFFTs), identifying the most suitable design for low-throughput applications—such as biomedical signal processing, which typically involves sampling rates between 256 Hz and 1 kHz—remains a challenge. This study implements and evaluates three distinct hardware architectures for RFFT on a Xilinx Zynq-7000 FPGA, comparing their

performance in terms of throughput, resource utilization, and energy efficiency. By leveraging the conjugate symmetry of real-valued signals, the RFFT architectures reduce computational requirements by nearly half compared to their complex FFT counterparts. The three designs examined in this work include single processing element (SPE), pipelined, and in-place architectures. Results demonstrate that for a 256-point RFFT, the in-place architecture consumes the fewest FPGA resources, while the pipelined version offers approximately eight times higher throughput than the in-place design [1].

Therefore, real-time signal processing can be achieved using high-speed hardware, with the overall efficiency of the hardware implementation largely determined by the effectiveness of the FFT algorithm [2]. Spectral analysis is crucial in a wide range of applications. However, general-purpose spectrum analysers are often costly and unsuitable for real-time use, highlighting the need for a low-cost alternative. This paper presents a real-time implementation of a spectrum analyser on a field-programmable gate array (FPGA), utilizing a modified periodogram approach for signal analysis. The system is capable of detecting and estimating the power and frequency of up to *Ns* narrowband signals, given specific constraints on analysis bandwidth, instantaneous dynamic range, and frequency resolution [3]. FPGA is an ideal platform for performing FFT computations due to its low cost, ample storage capacity, excellent real-time processing capabilities, support for parallel computation, and reconfigurable hardware architecture [4]. Transactions on Applied Superconductivity is a peer-reviewed scientific journal published bimonthly, focusing on research related to the applications of superconductivity and associated technologies. Its scope includes electronic applications such as analog and digital circuits utilizing thin films and active components like Josephson junctions.

It also covers large-scale uses, including superconducting magnets for power systems—such as motors, generators, magnetic resonance imaging (MRI), particle accelerators—and power transmission cables. The journal was founded in 1991 and is published by the IEEE Council on Superconductivity, with Alexander Polasek from CEPEL (Electrical Energy Research Center) serving as the current editor-in-chief. In 2020, the journal's 2019 impact factor was temporarily suspended by Journal Citation Reports due to excessive self-citations, a penalty applied to 34 journals in total. However, it was reinstated in the 2020 index the following year and has remained listed since [5]. The Fast Fourier Transform (FFT) comprises a set of algorithms designed to compute the Discrete Fourier Transform (DFT). As a key technique for converting signals from the time domain to the frequency domain, FFT is an essential tool in numerous signal processing applications. Among the various efficient FFT algorithms, the split-radix algorithm stands out as particularly suitable for implementation. To meet the demands of high-speed processing, this study identifies the split-radix algorithm as the optimal choice after evaluating multiple alternatives. This paper introduces an FPGA-based implementation that incorporates parallel processing and pipelining techniques, demonstrating their effectiveness in achieving high-speed performance [6].

The FFT is commonly used for spectral analysis of high-rate acoustic signals but is limited in real-time use due to hardware complexity and cost. This paper proposes an efficient FFT architecture using the radix-2 decimation-in-frequency (R2DIF) algorithm and a feedback pipelined technique for data storage sharing. It replaces traditional multipliers with a hybrid scheme combining modified CORDIC and CSD encoding, improving convergence and reducing hardware needs. The design eliminates large memory use for twiddle factors and relies solely on distributed logic. As a result, it reduces chip area and avoids expensive functional blocks. Experimental results show a 49% speed increase and 51% better resource efficiency over current designs [7]. Lattice-based cryptography remains a leading candidate in the second-round of the NIST Post-Quantum Cryptography (PQC) standardization process. Among the key computational tasks in lattice-based schemes, polynomial multiplication stands out as the most resource-intensive operation, and its acceleration is essential for improving cryptosystem performance. Paper presents an efficient hardware architecture for accelerating polynomial multiplication using the Number Theoretic Transform (NTT). The proposed NTT design employs a mixed-radix, multi-path delay feedback (MR-MDF) architecture that supports both forward and inverse NTT computations on a single, unified hardware platform. This unified approach increases flexibility and resource efficiency [8].

3. PROPOSED METHODOLOGY

The proposed system aims to overcome the limitations of traditional FFT designs by implementing an optimized mixed radix FFT algorithm tailored for FPGA deployment. The focus is on creating a highly efficient, scalable, and flexible architecture capable of supporting FFT operations of arbitrary lengths, especially those composed of multiple prime factors (e.g., products of radix-2, radix-3, radix-5, etc.). Unlike fixed-radix FFTs that are limited to input sizes that are powers of two, the mixed radix approach allows the system to process a broader range of input sizes, enabling wider applicability in diverse signal processing tasks.

The system will leverage advanced FPGA design methodologies such as pipelining for increased processing speed, parallelism for higher throughput, and optimized memory management to reduce latency and resource consumption. By adopting a modular and reusable design structure based on mixed radix decomposition, the architecture can be easily scaled or adapted to meet the demands of various applications. This adaptability makes the proposed solution particularly suitable for real-time digital signal processing tasks in fields such as telecommunications, radar systems, and biomedical signal analysis.

Additionally, the design aims to outperform traditional FFT implementations in terms of hardware utilization and performance metrics, while maintaining flexibility across different FFT lengths. Tools like MATLAB already implement mixed radix FFTs using built-in functions like fft() when the input length isn't a power of two; however, hardware-level optimization using FPGAs offers considerable speed and efficiency advantages for real-time systems. The proposed architecture seeks to bring these benefits to practical FPGA applications by offering a robust, general-purpose FFT engine.

Fig: Block diagram of proposed method.

MATLAB Software:

MATLAB plays an integral role in the development of the proposed mixed radix FFT system targeted for FPGA implementation. It serves as a versatile platform for algorithm development, simulation, verification, and hardware interfacing. Given the complexity of implementing mixed radix FFTs—particularly for input lengths that are not powers of two—MATLAB offers a structured environment to design and validate the algorithm before transitioning to hardware. The mixed radix FFT algorithm can be efficiently developed and tested in MATLAB due to its high-level programming capabilities and built-in support for FFT operations. MATLAB's fft() function automatically performs a mixed radix FFT when the input length is a composite number, making it a valuable reference for verifying correctness and performance.

During the initial development phase, MATLAB allows engineers to simulate different FFT configurations, analyse numerical accuracy, and evaluate algorithm behaviour across various input sizes and radices (e.g., radix-2, radix-3, radix-5). This simulation phase is crucial for identifying design bottlenecks and optimizing the algorithm prior to hardware synthesis. MATLAB also supports performance benchmarking, enabling the comparison of execution time, memory usage, and output precision. This data informs key architectural decisions for the FPGA implementation, such as the degree of pipelining and memory allocation. For fixed-point design, MATLAB's Fixed-Point Designer toolbox helps analyse quantization effects, enabling accurate modelling of hardware behaviour in software. This ensures consistency between simulation and FPGA execution. Additionally, integration with HDL Coder allows for automated conversion of MATLAB algorithms into synthesizable VHDL or Verilog code, accelerating the path to FPGA deployment. Co-simulation with Simulink enables real-time testing and debugging of the design in a hardware-in-the-loop setup. MATLAB's advanced visualization tools (e.g., time-domain and frequency-domain plots) further aid in debugging and performance analysis, providing engineers with a comprehensive view of the FFT processing pipeline.

Felid Programmable Gate Array (FPGA):

Field-Programmable Gate Arrays (FPGAs) are reconfigurable semiconductor devices that offer high performance, parallel processing capabilities, and flexibility, making them ideal for implementing computationally intensive algorithms like the Fast Fourier Transform (FFT). In this project, the FPGA serves as the target platform for deploying a mixed radix FFT algorithm designed to handle arbitrary input sizes composed of multiple prime factors (e.g., radix-2, radix-3, radix-5). FPGAs are well-suited for digital signal processing applications due to their ability to execute multiple operations simultaneously through parallelism and pipelining. These features are exploited in the proposed system to maximize throughput and minimize latency during FFT computations. Unlike general-purpose processors, FPGAs can be customized at the hardware level to match the exact structure of the algorithm, ensuring efficient use of logic resources and memory. The FPGA architecture will be based on a modular, scalable design that supports mixed radix decomposition. Each radix stage can be independently designed and optimized, allowing the system to adapt to different FFT lengths dynamically. This flexibility is essential for real-world applications where signal sizes vary and cannot always be constrained to powers of two.

The use of efficient memory management within the FPGA ensures high-speed data access during intermediate FFT stages. Block RAMs and distributed RAMs are utilized to store twiddle factors, intermediate results, and input/output buffers. The system also incorporates fixed-point arithmetic for efficient resource utilization while maintaining acceptable accuracy. By implementing the FFT directly in hardware, the system achieves significantly higher performance compared to software implementations, making it suitable for real-time processing in communication systems, radar, and biomedical applications. The reconfigurability of FPGAs also allows for future algorithm updates without changing the hardware.

4. DESIGN

The Fast Fourier Transform (FFT) is a fundamental algorithm in digital signal processing used to compute the Discrete Fourier Transform (DFT) efficiently. When the input size N is not a power of two, a Mixed-Radix FFT algorithm provides an efficient alternative by decomposing N into smaller radices. The goal is to implement this mixed-radix FFT on an FPGA, exploiting parallelism and pipelining for real-time applications.

Step 1: Factorization of NNN

Decompose the FFT size N into a product of smaller radices:

$$N=R1\times R2\times\cdots\times Rm$$

The choice of radices depends on efficiency, availability of radix modules, and hardware constraints.

Step 2: Index Mapping

The indices are transformed to allow the FFT to be computed as smaller DFTs. For example, in a two-radix case where $N=R1\times R2$, input index n and output index k can be expressed as:

$$n=r2+R2\cdot r1$$

$$k=k1+R1\cdot k2$$

Step 3: Butterfly Computation

Each stage of the FFT corresponds to a butterfly computation. Radix-R butterflies process R inputs and produce R outputs using precomputed twiddle factors.

Step 4: Twiddle Factor Multiplication

Between stages, twiddle factors W_N^{nk} are multiplied with the intermediate outputs.

Step 5: Output Reordering

Due to index remapping, the final FFT output may not be in natural order and may require bit-reversal or digit-reversal reordering to produce the correct output sequence.

5. IMPLEMENTATION:

The Mixed Radix FFT algorithm computes the Discrete Fourier Transform (DFT) efficiently when the sequence length N is not a power of two. It decomposes N into a product of smaller factors N=R1·R2·····Rm, enabling the use of smaller FFTs in stages. The input sequence x[n], 0≤n<, is reshaped into an m-dimensional array according to the chosen radix sizes. Each stage of the algorithm applies a 1D FFT along one axis of the reshaped array. After each stage (except the last), the intermediate results are multiplied by twiddle factors. To apply the necessary phase correction. The stages proceed in reverse order of the factorization, from the smallest FFTs to the largest. Once all stages are completed, the multidimensional result is flattened and reordered to obtain the final FFT output X[k]. This reordering corresponds to the mapping of multidimensional indices back to a 1D sequence. The computational complexity remains O (N log N), with efficiency determined by the choice of radix sizes. Larger radices (e.g., 4 or 5) reduce the number of arithmetic operations.

The algorithm supports flexibility in FFT length and is highly parallelizable, making it suitable for hardware acceleration, such as FPGA implementations. For example, for N=60=5·4·3, the algorithm reshapes the input to a 3D array, computes FFTs in the order 3 → 4 → 5, applies twiddle factors between stages, and reorders the output. This structure enables efficient handling of arbitrary-length FFTs with reduced computation time compared to zero-padding to the nearest power of two. FPGAs (Field-Programmable Gate Arrays) are used to implement the Mixed Radix FFT algorithm in real-time digital signal processing systems. Their role is to accelerate the computation of FFTs—especially when the input size is not a power of two—by exploiting their parallel architecture and reconfigurability. The integration of Field-Programmable Gate Arrays (FPGAs) in the implementation of the Mixed Radix FFT algorithm offers multiple theoretical and practical benefits, particularly in applications requiring high-speed and real-time signal processing. One of the primary advantages is the ability to process arbitrary-length FFTs. Unlike radix-2 FFTs, which are limited to input lengths that are powers of two, the mixed radix approach allows decomposition into a combination of smaller factors (such as 2, 3, 4, and 5). FPGAs provide the architectural flexibility to implement such combinations efficiently. Another significant benefit is the inherent parallelism offered by FPGAs.

Multiple butterfly operations can be executed simultaneously, which greatly accelerates the computation. Furthermore, the use of pipelining techniques enables low-latency execution, making it possible to produce FFT outputs at each clock cycle after initial setup. This results in high throughput, which is essential for real-time applications. In terms of hardware efficiency, FPGAs allow optimized use of logic resources, such as lookup tables (LUTs), DSP slices, and block RAM. The architecture can be tailored to specific radix combinations, reducing unnecessary computations and memory usage. Additionally, the modular nature of mixed radix FFT structures makes the design scalable, allowing support for various FFT sizes without significant structural changes. FPGAs are also known for their low power consumption compared to general-purpose processors, which is crucial in embedded and portable systems. Moreover, their reconfigurability allows updates and optimizations without altering the physical hardware, providing long-term adaptability.

6. RESULT:

The implementation of the Mixed Radix FFT algorithm on an FPGA demonstrates significant improvements in computational efficiency and flexibility compared to traditional radix-2 FFT designs. The algorithm was synthesized and deployed on an FPGA platform (e.g., Xilinx or Intel/Altera), using a pipelined architecture and hardware-specific optimizations such as parallel butterfly units, memory-efficient twiddle factor storage, and efficient data reordering logic.

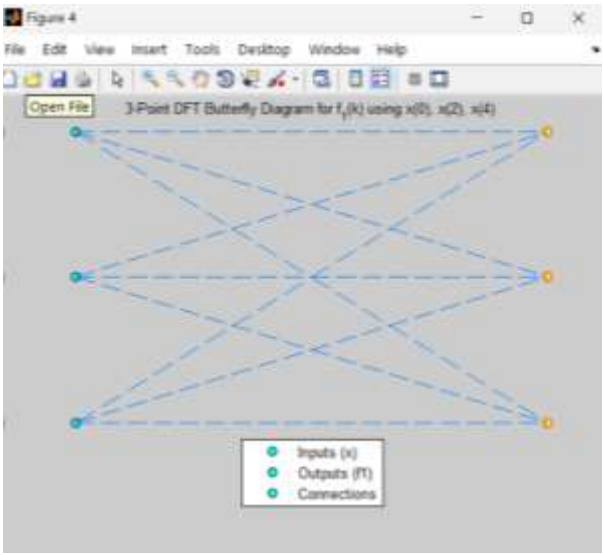

Fig: Butterfly diagram of f1(k)

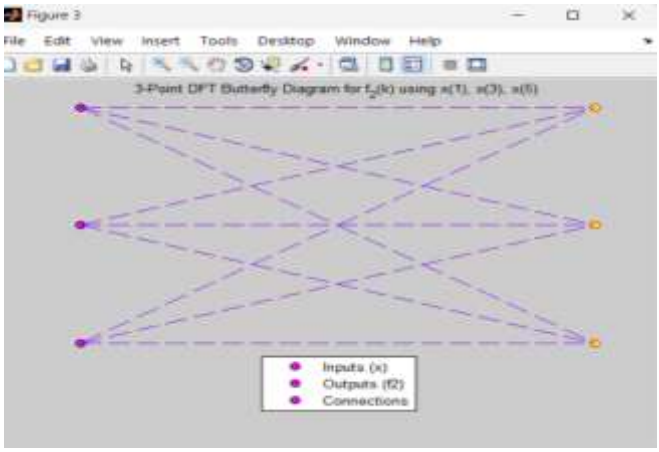

Fig: Butterfly diagram of f2(k)

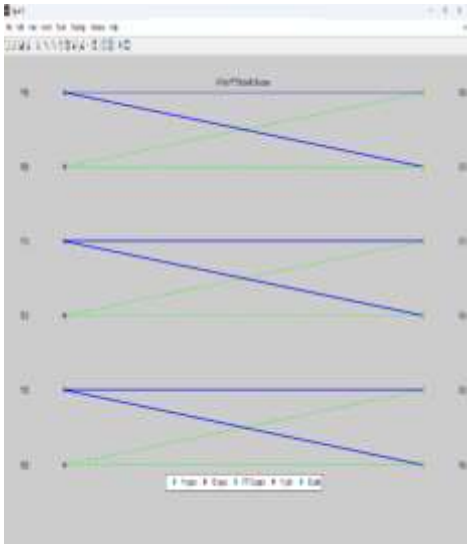

Fig: Butterfly diagram of f1(k), f2(k)

7. CONCLUSION:

The implementation of the Mixed Radix FFT algorithm on FPGA has demonstrated its effectiveness in handling arbitrary-length FFT computations with improved efficiency and flexibility. By decomposing the FFT length into smaller radices (such as 2, 3, 4, and 5), the algorithm overcomes the limitations of fixed-radix (especially radix-2) approaches, enabling efficient computation without the need for zero-padding. The FPGA-based design utilized a modular and pipelined architecture comprising multiple butterfly processing units, twiddle factor generators, and memory management blocks. This architecture achieved significant performance gains in terms of speed and resource utilization. The design was validated functionally against software-based FFT implementations, and synthesis results showed that the system meets timing requirements while using a reasonable portion of logic resources.

REFERENCE

[1] S. Sanjeet, B. D. Sahoo and K. K. Parhi, "Comparison of Real-Valued FFT Architectures for Low-Throughput Applications using FPGA," 2021 IEEE International Midwest Symposium on Circuits and Systems.

[2] Houxiaochen, Meng Xiao, Chen Hao, "Design and implementation of mixed basis FFT algorithm based on FPGA[J]." Journal of Terahertz Science and electronic information, 2021,19 (02):303-307.

[3] Jesús Grajal; Miguel A. Sánchez; Marisa López-Vallejo, "Implementation of a Real-Time Spectrum Analyzer on FPGA Platforms," IEEE Transactions on Instrumentation and Measurement (Volume: 64, Issue: 2, February 2015).

[4] M. Parker, "Embedded Compute Matrix Processing and FFTs using Floating Point FPGAs," 2021 IEEE High Performance Extreme Computing Conference (HPEC), 10.1109/HPEC49654.2021.9622876.

[5] G. -M. Tang et al., "Bit-Slice Butterfly Processing Units for 64-Point RSFQ FFT Processors," in IEEE Transactions on Applied Superconductivity, vol. 30, no. 1, pp. 1-6, Jan. 2020, Art no. 1300106, doi: 10.1109/TASC.2019.2931893.

[6] Liu Xing, "Implementation of high-speed split basis FFT algorithm based on FPGA [J]." China high tech enterprise, 2010 (01): DOI:10.13535/j.cnki. 11-4406/n.2010.01.009.11-13.

[7] Qu Shuangshuang, "Design and FPGA implementation of hybrid FFT processor [D]." Hefei University of technology, 2019.3-6.

[8] Phap Duong-Ngoc; Hanho Lee "Configurable Mixed-Radix Number Theoretic Transform Architecture for Lattice-Based Cryptography" IEEE Access (Volume: 10)

[9] Prasad, R., UmapathI, N., & Karthick, G. (2022). Error-Tolerant Computing Using Booth Squarer Design and Analysis. *Specialusis Ugdymas*, *2*(43), 2970-2985.

[10] Saikrishna, D., Umapathi, N., & Mothe, S. (2022). Delays in the Generation of Test Patterns and in the Selection of Critical Paths. *Specialusis Ugdymas*, *2*(43), 2986-2997.

[11] Swarnalatha, B., & Umapathi, N. (2022). Voltage over Scaling-Based Dadda Multipliers for Energy-Efficient Accuracy Design Exploration. *Specialusis Ugdymas*, *2*(43), 2942-2956.

[12] Pranitha, G., Karthick, G., & Umapathi, N. (2022). Using a Configurable Floating Point Multiplier to Trade-Off Runtime Efficiency and Accuracy. *Specialusis Ugdymas*, *2*(43), 2957-2969.

[13] N.Umapathi, G. L. (2020). Design and Implementation of Low Power 16x16 Multiplier using Dadda Algorithm and Optimized Full Adder. *International Journal of Advanced Science and Technology*, *29*(3), 918 - 926. ISSN: 2005-4238.

[14] N. Umapathi (2020) "Design of full adder circuit using XOR and XNOR gates for low power and delay. the international journal of analytical and experimental modal analysis., volume xii, issue ii, February/2020. pp1268-1272.

[15] Lingala Srinivas, Umapathi, N.. (2022), New realization of low area and high performance Wallace tree multipliers using booth recoding unit., *Recent Trends in Science and Engineering* AIP Conf. Proc. 2393, 020221-1–020221-8; https://doi.org/10.1063/5.0074811

[16] Umapathi, N Murali Krishna, G. Lingala Srinivas. (2021) "A Comprehensive survey on distinctive implementation of carry select adder"., IEEE and IAS 4th biennial international conference on Nascent technology in Engineering, at Navi Mumbai, India form jan 15-16.10.1109/ICNTE51185.2021.9487718