Design of RNS-KSA based 2D FIR Filter

S.Vamsi krishna[1], S.Salma [2], S.Sajida[3],

[1]*PG Scholar,* [2]*Assistant Professor,*[3]*Assistant Professor, Dept of ECE, Anantha Lakshmi institute of technology & sciences, Anantapuramu*

***Abstract* - Two-dimensional finite impulse response filters have the benefits of phase linearity and assured stability, making them appealing in applications. 2D Finite Impulse Response (FIR) filters serve a significant role in different applications, such as image processing, medical field, pattern recognition, robot vision, and seismic or geophysical signal processing. This paper presents a novel RNS-based 2D Finite Impulse Response (FIR) filter using Residue Number System (RNS) and floating- point arithmetic operations, in which Residue Number System (RNS) provides substantially lower ranges of numbers in arithmetic computations than starting numbers and Floating- point arithmetic provides more accuracy and dynamic range than fixed-point arithmetic. The Floating Point Adder in the filter is designed using a Kogge-Stone parallel prefix adder (KSA) to improve addition speed. The proposed novel 2D FIR filter can process 32, 48, and 64-bit data by just changing the corresponding RNS moduli selection which makes it a better choice over a fixed-point-based 2D FIR filter. This filter is designed in Verilog HDL for kernel sizes 3X3, 5X5 and 7X7 and synthesized using Altera's Quartus software for FPGA Board Cyclone III (EP3C16F484C6) to evaluate the design parameters of the filter.**

Keywords—2D FIR filters, Residue Number System (RNS), floating-point arithmetic, Kogge-Stone parallel prefix adder, Verilog HDL, Xilinx Vivado.

I. INTRODUCTION

Digital image processing relies heavily on filters to manipulate and enhance two-dimensional (2D) signals[2]. Among these filters, 2D finite impulse response (FIR) filters have grown in popularity because of their inherent stability, ease of design, & efficient implementation. Unlike their infinite impulse response (IIR) counterparts, 2D FIR filters require no feedback, leading to guaranteed stability. Their inherent phase linearity and stability characteristics make them indispensable for tasks requiring reliable and predictable filtering operations[6].Traditionally, 2D FIR filters have been implemented using fixed-point arithmetic due to its simplicity and ease of implementation. However, fixed-point arithmetic may impose limitations on precision and dynamic range, particularly in applications demanding high accuracy and flexibility. The finite precision of fixed-point arithmetic can lead to quantization errors, especially when dealing with signals with wide dynamic ranges or when performing complex mathematical operations[7].

Moreover, fixed-point arithmetic requires careful selection of word lengths and scaling factors to ensure adequate representation of both integer and fractional parts of the data. This process can be time-consuming and may result in suboptimal performance if not done correctly[4]. Additionally, fixed-point arithmetic may suffer from overflow or underflow issues when operating on signals with large amplitudes or when performing intensive computations.

Furthermore, fixed-point arithmetic cannot represent numbe rs with fractional precision d irectly, necessitating additional scaling and rounding operations, which can introduce additional errors and computationa overhead[20]. In contrast, floating-point arithmetic offers several advantages over fixed-point arithmetic, making it well- suitec for demanding signal processing applications. One key advantage is its ability to represent a wider range o f values including higher precision, which allows for more accurate representation and manipulation of numerical data. Floating- point arithmetic also simplifies scaling and rounding operations by automatically adjusting the exponent of the floating-point number, thereby reducing the risk of overflow and underflow errors[19]. Floating-point numbers consist of a sign bit, a fixed number of exponent bits, and a variable number of fraction bits, allowing for the representation of both very large and very small numbers with greater accuracy. This increased precision and dynamic range enable floating-point arithmetic to handle a wider variety of signals and algorithms, leading to improved performance and numerical accuracy in 2D FIR filtering.

Furthermore, integrating the Residue Number System (RNS) with floating-point arithmetic improves the efficiency of 2D FIR filters. RNS offers advantages such as reduced computational complexity and faster arithmetic operations by working with residues modulo a set of coprime numbers[12]. By utilizing RNS, 2D FIR filters can achieve smaller ranges of numbers in arithmetic calculations, leading to improved resource utilization and overall performance[13].

The Kogge-Stone parallel prefix adders are used in the desi gn of floating-point adders contribute to the optimization of 2 D FIR filter implementations. Kogge-Stone adders utilize parallel computation to reduce propagation delay and improve efficiency in arithmetic operations[18]. Kogge-Stone adders in the design of floating-point arithmetic units, the delay and reso urce overhead associated with arithmetic operations can be minimized, resulting in quicker and more effective filter implementations[22].

II. LITERATURE REVIEW

Reducing the space taken up by 2D FIR filters in hardware design is crucial. The optimization aims to make 2D FIR filter designs smaller for various filter sizes. This optimization is achieved by using Radix 2r multiplication, which helps reduce the number of adders needed, thus minimizing the hardware used, this approach proposed by Ref.[5]. The data Broadcast structure of the Fir filter, this improves the area and reduces the critical path delay, this technique proposed by Ref.[17].

The floating-point multiplier employs Brickell's Algorithm, which is a specific method for performing multiplication operations with floating-point numbers. The conversion unit utilizes parallel conversion techniques for forward conversion and applies the Chinese Remainder Theorem for reverse conversion, Using the Residue Number

System (RNS) enables faster arithmetic computations by allowing parallel and carry-free operations, as proposed by Ref.[21].

In Shahana T's study, "RNS vs Traditional Digital Filters," a comparative analysis between Residue Number System (RNS) and traditional binary number system FIR filters is conducted. The paper outlines RNS fundamentals and structural differences between RNS and traditional filters, exploring hardware implementation and performance metrics. Results indicate RNS-based FIR filters achieve significantly higher processing speeds, over three times faster than traditional ones, while requiring less area, particularly as the number of filter taps increases. This highlights RNS's advantages, especially for high-speed data processing with efficient hardware usage.[11].

Siva Kumar's paper presents the Kogge Stone adder (KSA) as a faster and more efficient option than the Carry Skip adder (CSKA) for adding numbers in VLSI circuits of various bit lengths (8, 16, 32, and 64 bits). Derived from carry look-ahead adders, KSA minimizes latency to O(log2 n) using a tree network, resulting in reduced energy consumption, compact area, and faster operation compared to CSKA. The study emphasizes enhancing adder performance in digital systems, underscoring the significance of speed and energy efficiency in arithmetic units like ALUs, showing that KSA outperforms CSKA in modern electronic devices[22].

III. RESIDUE NUMBER SYSTEM

The Residue Number System (RNS) depicts numbers by their remainders modulo several pairwise coprime integers known as moduli. Residue Number System (RNS) is a nonpositional number system which has drawn a lot of attention due to its capacity for high-speed arithmetic operations, especially in the context of modular arithmetic principles where it provides benefits in terms of parallelism, speed, and fault tolerance compared to conventional positional number systems like decimal or binary.

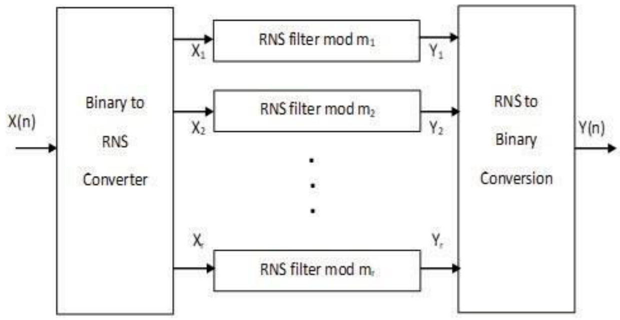

Fig. 1. FIR Filter Implementation in RNS

At the core of the Residue Number System lies the concept of residue classes. By selecting a set of prime numbers with pairwise coprimality, RNS enables a unique representation of integers within a defined range. The Chinese Remainder Theorem (CRT) plays a pivotal role in RNS, allowing for reconstructing a unique solution from its residues.

This section explains the basic mathematical principles of RNS and its implications for arithmetic operations with an example.

Since moduli are positive integers, let's say that k_{1}, k_{2},..., k_{n} are the greatest common divisor for a couple

(k_{i}, k_{j}) equals '1'. y =x1 (mod k_{1}); y =x2 (mod k_{2}) .

Then there is a single simultaneous solution for y =xn (mod p_{n}) modulo k_{1}, k_{2},...,k_{n}. Equation P(Dynamic range) = k_{1}*k_{2}*k_{3} yields 5 * 7 * 9 = 315.

Undoubtedly, we are able to express any integer within the range of 0 to 314 (dynamic range). C = 100 = 0 (mod 5)

Let's C=100, hence C = 100 =2(mod 7)

In the RNS representation, C = (0, 2, 1) and C = 100 = 1(mod 9)

K = k_{1}*k_{2}*k_{3} = 5 * 7 * 9 = 315

C + D = 100 + 13 = P

C + D = 100 + 13 = P

1) C = (0, 2, 1) D = (3, 6, 4)

2) C + D = ((0 + 3) (mod 5), (2+6) (mod 7), (1+4) (mod 9))= (3 (mod 5), 1 (mod 7), 5 (mod 9)) = (3, 1, 5)

3) Reconstruction from corresponding residues

P = P_{1}*Z_{1} + Z_{2}*Z_{2} + P_{3}*Y_{3} - r*K
Z_{i} = (K/k_{i}) * h_{i} & Z_{i}/k_{i}
= 1 (mod k_{i});
r* K<= P_{1}*Z_{1} + P_{2}*Z_{2} + P_{3}*Z_{3} < (r+1) * K

a) $Z_{1} = (315/5) * h_{1} = 63h_{1}$ and $(63h_{1})/5 = 1 \pmod 5$,
then $h_{1} = 2$ and $Z_{1} = 126$

b) $Z_{2} = (315/7) * h_{2} = 45h_{2}$ and $(45h_{2})/7 = 1 \pmod 7$.
then $h_{2} = 5$ and $Z_{2} = 225$

c) $Z_{3} = (315/9) * h_{3} = 35h_{3}$ and $(35h_{3})/9 = 1 \pmod 9$,
then $h_{3} = 8$ and $Z_{3} = 280$

d) $r * 315 <= 3 * 126 + 8 * 225 + 5 * 280 < (r + 1) * 315$ then $r = 6$

$P = 3 * 126 + 8 * 225 + 5 * 280 - 6 * 315 = 2003-1890=113$

Consider above simple example where the sum of two numbers, A and B, needs to be computed within the RNS framework. Given A = (0, 2, 1) and B = (3, 6, 4) in their respective residue representations, the sum S = A + B is determined through modular arithmetic operations modulo 5, 7, and 9. By applying the principles of RNS, the result S = (3, 1, 5) is obtained, showcasing the simplicity and effectiveness of RNS in handling arithmetic operations..

IV. PROPOSED METHOD

The FIR Filter Implementation in RNS mainly consists of three steps. They are Binary to RNS Converter, FIR filter, RNS to Binary Converter as shown in fig 1.

Binary to RNS Converter: Here it converts binary input number to Residue Number and the sends the number to the FIR filter.

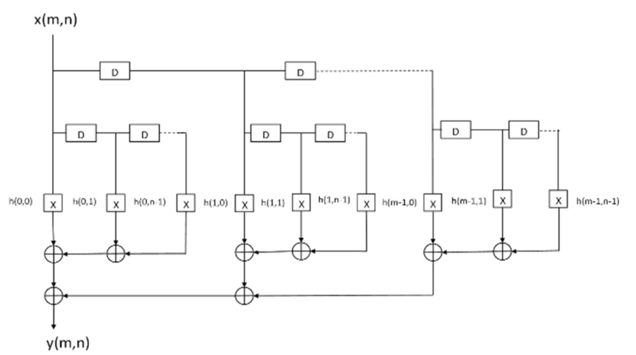

Fig. 2. Two-dimensional FIR filter

The 2D FIR filter architecture employs fixed-point adders and multipliers. In 2D FIR filter each input sample undergoes multiplication with a corresponding filter coefficient, and the results are summed up to produce the filtered output. The fixed-point arithmetic involves representing numbers with a fixed number of integer and fractional bits. One of the primary disadvantages of using fixed-point adders and multipliers is the limited dynamic range and precision. Fixed-point arithmetic requires careful selection of word lengths and scaling factors to prevent overflow and quantization errors. Additionally, fixed-point implementations often suffer from increased complexity in designing and optimizing algorithms due to the need for manual scaling and quantization.

Moreover, floating-point arithmetic simplifies algorithm design and implementation by eliminating the need for manual scaling and quantization. The flexibility of floating point representation allows for easier adjustment of precision requirements and facilitates faster prototyping and development of complex signal processing algorithms.

In summary, while the conventional fixed-point 2D FIR filter architecture is limited by its precision and range constraints, the use of single-precision floating-point adders and multipliers in the 2D FIR filter architecture offers improved accuracy, flexibility, and ease of implementation. Floating-point arithmetic presents a viable alternative for researchers and engineers seeking to optimize signal processing systems for performance and accuracy.

Single precision floating point represents the standardized format for representing real numbers in computer systems. Because it can represent a large range of numbers with a respectable degree of precision, it is commonly employed. The exponent component allows for a wide range of magnitudes, from 2^{-126} to 2^{127}, which corresponds to approximately 1.18×10^{-38} to 3.4×10^{38} in decimal notation. This large range offered by the single-precision floating-point arithmetic and Residue Number system allows for the processing of binary data across various bit lengths, including 32, 48, and 64 bits with the same proposed filter by simply changing the RNS moduli. One significant advantage of single-precision floating-point representation over fixed-point arithmetic is its ability to handle a dynamic range of values without sacrificing precision. The decimal point's location is set in fixed-point arithmetic, limiting the range of representable values and potentially leading to loss of precision or overflow/underflow errors. However, with floating point representation, the position of the decimal point can float within a wide range determined by the exponent, allowing for both very large and very small numbers to be accurately represented without loss of precision. While the absolute precision decreases for very large or very small numbers due to the limited number of significant digits available in the fraction component, the relative precision remains constant

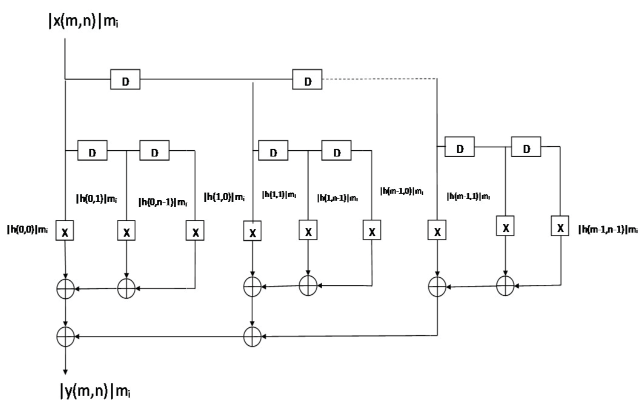

Fig. 3. I th modulo Two dimensional FIR filter

Designing a single floating-point multiplier and adder for a 2D FIR (Finite Impulse Response) filter involves carefully selecting efficient architectures to optimize performance while minimizing resource utilization. One common approach is to utilize Kogge-Stone adders for addition which offer high-speed operation and efficient use of hardware resources.

A parallel prefix adder that excels at adding multi-bit values quickly is the Kogge-Stone adder. It operates by performing addition in a hierarchical manner, where each stage generates partial sums and carries that propagate through the subsequent stages. This architecture is particularly suitable for the addition stage of the FIR filter, as it allows for fast computation of the filter's output by efficiently summing the products of the filter coefficients and input samples.

The proposed filter takes coefficients as floating point residues of actual coefficients obtained from corresponding floating point RNS Ith modulo[7] and processes the data in floating point format. The proposed filter operates by first accepting integer inputs, which are then converted into corresponding residues through a binary to Residue Number System (RNS) converter. These residues are subsequently transformed into single-precision floating-point data. Following this conversion, a modulo filter is applied to process the residues effectively. The processed data is then reverted back to fixed-point values, and these values are ultimately converted to the desired result through a RNS to binary converter. This process facilitates efficient manipulation and transformation of the input data, ensuring compatibility with the desired output format. The above architecture consumes less power compared to other existing filters, giving it the perfect option for putting this FIR filter into practice.

When creating a single floating-point adder and multiplier for a two-dimensional FIR filter, the use of KoggeStone adders for addition offers several advantages. Firstly, these architectures enable high-speed computation of filter outputs, which is essential for real-time signal processing applications. Additionally, they efficiently utilize hardware resources, allowing for the implementation of complex filter designs on resource-constrained platforms.

The design process involves mapping the filter coefficients and input samples to the appropriate multiplier and adder units, ensuring that the computation is performed with the desired precision and accuracy. This may involve scaling the inputs and coefficients to fit within the dynamic range of the floating-point representation and handling overflow and underflow conditions to prevent numerical errors.

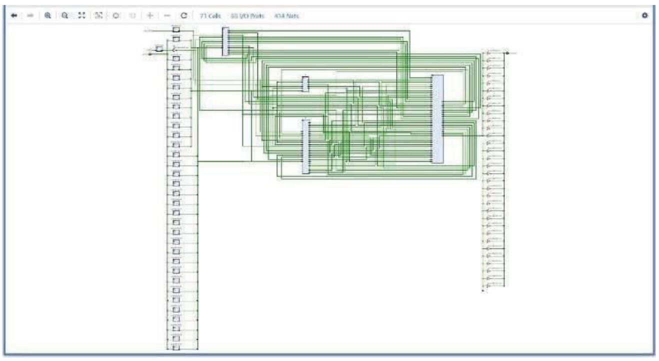

Fig. 4. RTL Schematic

Furthermore, the design must consider factors such as power consumption, area utilization, and timing constraints to ensure that the resulting implementation meets the performance requirements of the target application. This may involve optimizing the architecture and algorithm to minimize resource usage while maintaining the desired level of accuracy and efficiency.

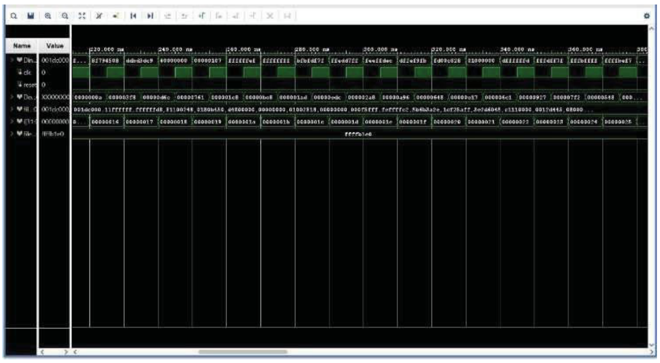

Fig. 5. Simulation Result

TABLE I. POWER DISSIPATION(MW) OF VARIOUS 2D FIR FILTERS.

Kernel size / Multiplier	3x3	5 5	7x7
Basic Multiplier	207.01	337.91	347.40
Booth Multiplier	300.41	369.98	394.29
Dadda multiplier	207.01	358.65	385.98
Floating Point multiplier	175.18	191.83	260.83

The data presented in the above table indicates that the proposed 2D FIR Filter demonstrates the lower power consumption compared to other existing filters.

V. CONCLUSION

Finally, this study presents a new method for creating 2D Finite Impulse Response (FIR) filters using floating-point arithmetic operations and the Residue Number System (RNS). By leveraging benefits with RNS, which offers smaller ranges of numbers in arithmetic calculations, and floating-point arithmetic, which provides increased precision and dynamic range, our proposed filter demonstrates enhanced performance in various applications such as image processing, medical diagnosis, pattern recognition, and signal processing. The utilization of a Kogge-Stone parallel prefix adder in the design of floating-point adder effectively improves addition speed, enhancing the overall efficiency of the filter. Additionally, the flexibility to process different data sizes (32, 48, 64 bits) by simply adjusting RNS moduli selection further highlights the versatility of our proposed approach. The implementation of the filter in Verilog HDL using Altera's Quartus software for FPGA Board Cyclone III (EP3C16F484C6) allows for evaluation of design parameters, ensuring its feasibility and effectiveness in practical applications.

REFERENCES

[1] Omondi, A. R., & Premkumar, A. B. (2007). Residue number systems: theory and implementation (Vol. 2). World Scientific.

[2] Najim, Mohamed, ed. Digital filters design for signal and image processing. John Wiley & Sons, 2013. [3]. Yadav, Akarsha, Vijaya Prakash, and Vidya Saraswathi.

[3] "Design of Modified RNS-PPA Based FIR Filter for HighSpeed Application." European Journal of Engineering and Technology Research 7, no. 5 (2022): 37-44.

[4] Samhitha, Naamatheertham R., Neethu Acha Cherian, Pretty Mariam Jacob, and P. Jayakrishnan. "Implementation of 16-bit floating point multiplier using residue number system." In 2013 International Conference on Green Computing, Communication and Conservation of Energy (ICGCE), pp. 195-198. IEEE, 2013.

[5] Cheemalakonda, Soumica, Sudeeksha Chagarlamudi, Bhasita Dasari, Shaik Sameera Sulthana, and Kishore Kumar Gundugonti. "Area efficient 2D FIR filter architecture for image processing applications." In 2022 6th International Conference on Devices, Circuits and Systems (ICDCS), pp. 337-341. IEEE, 2022.

[6] Mohanty, Basant Kumar, and Pramod Kumar Meher. "A high-performance FIR filter architecture for fixed and reconfigurable applications." IEEE transactions on Very Large Scale Integration (VLSI) systems 24, no. 2 (2015): 444-452.

[7] Chiang, Jen-Shiun, and Mi Lu. "Floating-point numbers in residue number systems." Computers & Mathematics with Applications 22, no. 10 (1991): 127-140.

[8] Mohan, PV Ananda, and PV Ananda Mohan. Residue Number Systems. Cham, Switzerland: Birkhäuser, 2016.

[9] Phalguna, P. S., Dattaguru V. Kamat, and PV Ananda Mohan. "New RNS-To-Binary Converters for Three-Moduli Set $\{2^{n+ 1}-1,\2^{n}-1,\2^{n+ k}\} $." In 2019 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), pp. 1-6. IEEE, 2019.

[10] Patronik, Piotr, and Stanisław J. Piestrak. "Design of reverse converters for the general RNS 3-moduli set {2 k, 2 n− 1, 2 n+ 1}." EURASIP Journal on Advances in Signal Processing 2023, no. 1 (2023): 92.

[11] Shahana, T. K., Rekha K. James, Babita R. Jose, K. Poulouse Jacob, and Sreela Sasi. "Performance analysis of FIR digital filter design: RNS versus traditional." In 2007 International Symposium on Communications and Information Technologies, pp. 1-5. IEEE, 2007.

[12] Balaji, M., and N. Padmaja. "Area and delay efficient RNS-based FIR filter design using fast multipliers." Measurement: Sensors 31 (2024): 101014.

[13] Kotha, Srinivasa Reddy, Akshit Singhvi, and S. K. Sahoo. "An efficient RNS-FIR filter implementation using the moduli set {2 k− 1, 2 k, 2 k− 1− 1}." In 2013 IEEE Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia), pp. 191-195. IEEE, 2013.

[14] Kucherov, Nikolay N., Viktor A. Kuchukov, Nataliya N. Kuchukova, and Anastasia E. Shangina. "A high-speed residue-to-binary converter based on approximate Chinese Remainder Theorem." In 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp. 325-328. IEEE, 2018.

[15] Cetin, A. E., and Rashid Ansari. "Iterative procedure for designing two-dimensional FIR filters." Electronics Letters 3, no. 23 (1987): 131-133.

[16] Hu J, Rabiner L. Design techniques for two-dimensional digital filters. IEEE Transactions on Audio and Electroacoustics. 1972 Oct;20(4):249-57.

[17] Singh, Thingujam Churchil, and Manoj Kumar. "Design and implementation of two-dimensional digital finite impulse response filter using very high speed integrated circuit hardware description language." International Journal of Electrical and Computer Engineering 12, no. 4 (2022): 3684.

[18] Nikhil, S., & Lakshmi, P. V. (2016, August). Implementation of a high speed multiplier desired for highperformance applications using kogge stone adder. In 2016 International Conference on Inventive Computation Technologies (ICICT) (Vol. 1, pp. 1-4). IEEE.

[19] Singh, N. and Dhanabal, R., 2018, February. Design of Single Precision Floating Point Arithmetic Logic Unit. In 2018 4th International Conference on Electrical Energy Systems (ICEES) (pp. 133-137). IEEE.

[20] Dhanabal, R., V. Barathi, Sarat Kumar Sahoo, Naamatheertham R. Samhitha, Neethu Acha Cherian, and Pretty Mariam Jacob. "Implementation of floating point MAC using residue number system." In 2014 International Conference on Reliability Optimization and Information Technology (ICROIT), pp. 461-465. IEEE, 2014.

[21] Balaji, M., and N. Padmaja. "Area and delay efficient RNS-based FIR filter design using fast multipliers." Measurement: Sensors 31 (2024): 101014.

[22] Penchalaiah, U., and Siva Kumar VG. "Design of highspeed and energy-efficient parallel prefix kogge stone adder." In 2018 IEEE International Conference on System, Computation, Automation and Networking (ICSCA), pp. 17. IEEE, 2018.

[23] Daphni, S., and KS Vijula Grace. "Design an area efficient kogge stone adder using pass transistor logic." In 2021 Third international conference on intelligent communication technologies and virtual mobile networks (ICICV), pp. 614-618. IEEE, 2021.