FOREX Publication
Open Access | Rapid and quality publishing

International Journal of
Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

Designing RNS-Based FIR Filter with Optimal Area, Delay, and Power *via* the use of Swift Adders and Swift Multipliers

Syed AliAsgar[1*], and A Yasmine Begum[2]

[1]*Research Scholar, Department of ECE, Mohan Babu University, Tirupati, Andhra Pradesh, India; aliasgarsyed@gmail.com*
[2]*Associate Professor, Department of ECE, Mohan Babu University, Tirupati, Andhra Pradesh, India; anwaralyyasmine@gmail.com*

***Correspondence:** Syed AliAsgar; *aliasgarsyed@gmail.com*

ABSTRACT- Based on the Residue Number System (RNS), Finite Impulse Response filters have gained prominence in digital signal processing due to their efficiency in handling complex computations. This work presents a comprehensive analysis on optimizing area, delay, and power in the FIR filter by exploring different adders and multipliers. Three prominent adders, namely Ripple Carry Adder, Kogge Stone Adder, and Proposed Adder, are evaluated for their impact on area and delay. The choice of adder influences the overall performance of the FIR filter, and a careful selection is made based on the trade-offs between area and delay. Furthermore, various multipliers, including Booth, Baugh-Wooley, Braun, and Array, are compared in terms of their efficiency in power consumption. Multipliers contribute significantly to the overall power consumption, and the analysis involves selecting the most suitable multiplier for achieving the desired power optimization. The various arrangements of swift adders and swift multipliers were suggested and executed in the Finite Impulse Response (FIR) filter and then the RNS algorithm was applied to it. The comparison of 8-tap 8-bit of the proposed adder with the array multiplier shows that the area is reduced by 52.70% with other combinations and by comparing the RCA adder with the array multiplier, the critical path delay is reduced by 38.51% and the maximum frequency is produced by the KSA adder with Braun multiplier which is increased by 29.78% with other combinations.

Keywords: FIR Filter, Residue Number System, area, Delay, Power, Baugh-Wooley multiplier.

ARTICLE INFORMATION

Author(s): Syed AliAsgar and A Yasmine Begum;
Received: 13/08/2024; **Accepted**: 20/10/2024; **Published**: 30/11/2024;
e-ISSN: 2347-470X;
Paper Id: IJEER 1308-08;
Citation: 10.37391/IJEER.120412
Webpage-link:
https://ijeer.forexjournal.co.in/archive/volume-12/ijeer-120412.html

Publisher's Note: FOREX Publication stays neutral with regard to Jurisdictional claims in Published maps and institutional affiliations.

1. INTRODUCTION

The RNS filter's area, delay, and power consumption can be reduced by using an efficient swift multiplier and swift adder architectures. There are several benefits of using optimized swift adders and swift multipliers in RNS FIR filters, including better performance, and less hardware complexity. The main function of a FIR filter is to multiply input samples by filter coefficients and then accumulate the resulting products. The total hardware complexity of the filter, as well as the circuit space and power consumption, may be minimized by using effective rapid multipliers, such as the Braun, Booth, Baugh-Wooley, and array multipliers, to reduce the number of partial products required [1]. Improved rapid adders, such as the KSA adder, proposed adder, and Ripple Carry adder, which decrease the number of logic gate levels in the adder circuit and shorten the carry signal propagation latency, may also improve the filter's performance. As a result, less power will be used and filters will operate more quickly. For effective digital signal processing applications, optimized fast adders and fast multipliers can greatly enhance FIR filter performance, power consumption, and circuit area. As such, they are an important design factor to consider [2].

Similarly, multipliers play a pivotal role in performing arithmetic operations in computer systems by digitally multiplying two binary numbers. Advanced fast multiplier architectures are tailored to meet the requirements of high-performance computing systems, offering reduced critical path delay, improved throughput, and efficient resource utilization. By utilizing efficient swift multiplier and swift adder architectures, the area, delay, and power consumption of RNS FIR filters can be significantly reduced [3]. These optimized components offer many benefits, including improved performance and reduced hardware complexity. FIR filters primarily function by multiplying input samples by filter coefficients and accumulating the resulting products. Using effective fast multipliers, such as array multipliers, Booth multipliers, Baugh-Wooley multipliers, and Braun multipliers, may speed up this process and reduce the amount of circuit space, power consumption, and hardware complexity [4].

Digital signal processing (DSP) is indispensable in various applications across diverse domains, facilitating the efficient processing, analysis, and manipulation of digital signals to extract meaningful information and enhance system performance. Integrating multiple swift adders and swift multipliers into complex DSP systems further enhances their functionality. Although the L-parallel filter design significantly increases the hardware's cost and power consumption, the parallel architecture enhances the device's performance [5]. This constraint of the parallel design led to the construction of

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

several swift multipliers by the combination of swift adders. This method reduces hardware needs by up to 50% of the total and is considerably more space-efficient than traditional parallel architecture. Without the multiplier and adder, there will be fewer FIR filters [6]. The appropriate operation of the multiplier and adder blocks is necessary for the filter to achieve the desired processing speed and power dissipation.

Residue Number System (RNS) is a non-weighted, modular number system that allows parallel processing, offering several advantages in high-speed computations and efficient hardware implementations, particularly in applications like signal processing, cryptography, and error detection. Thus, it differs greatly from weighted number systems such as binary or decimal number systems. Residue Number System (RNS) can offer several advantages when used in FIR filters. RNS is a digital arithmetic system that allows for parallel computation, which can lead to faster processing times compared to traditional binary arithmetic [7]. Additionally, RNS-based FIR filters can be implemented using simpler arithmetic operations, such as addition and subtraction, rather than multiplication, resulting in reduced hardware requirements and lower power consumption. Furthermore, RNS is well-suited for pipelining, which can further improve the throughput of FIR filters [8].

1.1 RNS operation

The RNS FIR filter is created in this work using swift-adders and swift multipliers to improve the filter's execution. The main function of a FIR filter is applying filter coefficients to the input sample multiplication and accumulation. In FIR filters, the RNS approach includes expressing filter coefficients and samples used for input as residues modulo a collection of coprime moduli. This allows for the simultaneous execution of filtering operations since each modulus's calculations are carried out individually. Initially, the input samples and coefficients are transformed into their residue representation.

In the meantime, filtering operations are carried out in the residue domain, where the output is multiplied by the respective residue of the coefficient to obtain each residue of the input sample. Ultimately, the output sample is obtained by reconstructing the filtered residues into binary form. The encoding and decoding process was done by RNS Encoder and RNS Decoder as shown in the *figure 1*. Benefits of RNS include lower complexity and higher efficiency because it carries propagation removal and parallelism. For best results, nevertheless, the moduli set must be carefully chosen, and conversion overhead must be considered**.**

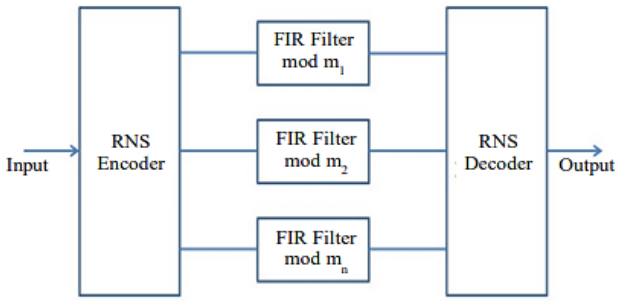

Figure 1. RNS operation

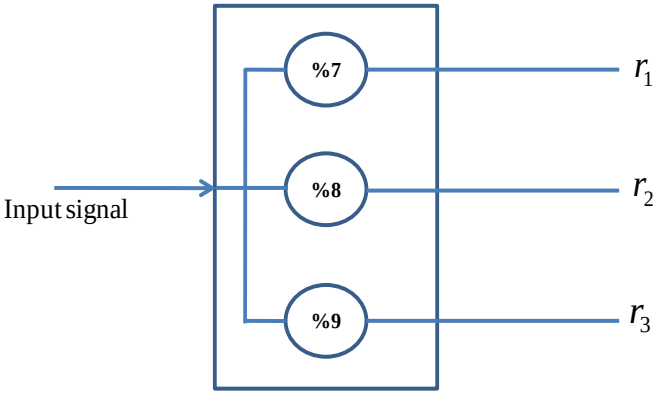

Figure 2. RNS Encoder implementation

According to the Chinese Remainder Theorem, the remainders of the Euclidean division of an integer n by multiple integers, then you may uniquely calculate the remainder of the division of n by the product of these integers, provided that the divisors are pairwise coprime, according to the Chinese remainder theorem [10]. The Chinese Remainder Theorem is a mathematical theorem that provides a way to solve a system of linear congruence's with different moduli. Alternatively, it provides a method for finding a unique solution to a set of equations where the unknowns are integers and each equation is modulo a different number. The theorem states that if we have a set of pair wise relatively prime integers, then any system of linear congruences modulo these integers has a unique solution modulo their product [11].

Let us consider the output of FIR filter $(r_1', r_2', \ldots . r_t')$ with modulo $(m_1, m_2, \ldots . m_t)$ where all m_t are mutually prime.

Let $M = (m_1 * m_2 * \ldots . m_t)$

$$M_i = {}^{M}/_{m_i} \quad (1)$$

Let K_i be the result that $(M_i * K_i)\%m_i = 1$, then we have the corresponding numbers as

$$y = (\sum_{i=1}^{t}(M_i K_i r_i'))\%M \quad (2)$$

RNS output will be obtained by

$$= (\sum (M_i \times K_i \times r_i))\%M \quad (3)$$

The following sections comprise the division of the paper: A review of the methods used in earlier designs may be found in *Section 2*. *Section 3* describes the suggested work that was executed in making the Residue Number System based Finite Impulse Response filter. *Section 4* shows the filter's performance analysis and outcomes. The conclusion is presented in *Section 6.*

2. LITERATURE REVIEW

The Partial Product lowering stages in multiplication tasks are energy-intensive and take up an immense amount of silicon space. Therefore, three approximate multipliers have been designed using fundamental methodologies. The first method involves making use of approximations to produce partial

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

products. The Partial product tree is then subjected to truncation. The last technique approximates adders and compressors and accumulates partial derivatives. Thus, the concept of approximate computation was created to lower power usage. This employs an approximation methodology proposed by [12].

In [13], an AND and OR gate-based error-compensated approximation multiplier with a reordered 2:1 compressor was introduced for low energy design. The compressor can be made simpler and require fewer gates if it can operate with only two of the four inputs by rearranging the information order. The proposed method delivers 99.3% efficiency requiring only 44.7% energy and 31.7% use of space compared to current standard practices. According to [14], 8x8 approximation multipliers that use high-order similar compressors were developed. Using various compressors for different weights allows for the error-free accumulation of product terms and the reduction of energy usage. The intermediate significance weights are computed using higher-order approximation compressors, like 8-to-2 compressors, which optimize the logic of the "carry cycle."

To design a system that is error-efficient, [15] introduced the approximation multiplier based on the rounding approach by rounding the input parameters to the next power of 2. The input data is transformed and compiled using a mathematical section that includes subtractor, adder, and shifter units. The data operands can range in size from 8 to 32 bits. The simulation findings show that there is a considerable improvement over similar approximation multipliers, with an estimated 22% delay and 57% power usage. To fix this issue, [16] proposed a rounding technique that can be quickly modified to act as an approximation multiplier. Compared to filters employing existing multipliers, the suggested technique has a 50.8% smaller footprint, uses 32.5% less energy, and has 54.7% less latency. To generate an approximate multiplier, [17] proposed including a single-gate, approximate compressor. Compared to current methods, the suggested technique uses 52% less area and 61% less energy.

Because of its excellent configurability, [18] proposed an approximation multiplier for unsigned numbers, which aims to minimize all electronic measurements while maintaining superior precision. It may be employed in a variety of situations without breaking the budget because it offers several alternatives for lowering energy use by 35–85%. By using the truncating method, [19] presented an approximation multiplier. The approximation multiplier computed the outcome using the binary format of the operands and truncated outcomes. This multiplier is 89.2 percent more energy-efficient than the previous one, taking up just 74.9 percent of the space.

To prevent "carry propagation" an error vector and an approximate total are produced using the suggested approximation adder. Two designs for approximate 8X8 multipliers, M1 and M2, are provided by the OR gates and error reduction techniques [20]. It is shown that compared to an exact Wallace multiplier with speed optimization, the suggested approximation multipliers use less power. The proposed multipliers achieve superior accuracy due to their minimal error margins. Additionally, simulations have demonstrated that M2 is more precise than M1, although requiring more power and having a longer latency. Compared to earlier approximation models, the multipliers for the suggested approximation are more accurate [21].

To maintain a high level of precision in delay and area, [22] proposed an alternative methodology which includes the RNS algorithm. For certain tasks in digital signal processing, the suggested approach offers a mechanism to execute modular arithmetic on the residues in parallel Applying modular operations to the filter coefficients and samples entering the filter results in the generation of residues. Subsequently, the forward converter's output is then sent to the RNS decoder after passing through a series of FIR filters. The demonstration of area and total thermal power dissipation for various combinations of multipliers, adders, and combinations of different input bit sizes shows that the difference in power usage was minimal as compared to bit by bit separately, and reduction of logic elements by 14.05% for 8tap-8 bit along with 77.15% reduction of logic elements when compared with proposed methods. In contrast to previous designs, delay, and energy savings were given priority, although accuracy was uneven. The suggested alternatives offer significant savings reductions without sacrificing precision [23].

3. PROPOSED WORK

3.1. Baugh-Wooley Multiplier

The Baugh-Wooley multiplier implements binary multiplication in hardware. It works especially well with signed binary integers. The technique is optimized for binary operations and works similarly to manual multiplication where the numbers are multiplied one by one of their respective digits, and the sum of the results is then determined. The length of the partial products and their quantity will both be quite large in signed multiplication. Accordingly, the Baugh Wooley algorithm- a method for multiplying signs- was introduced [24]. Mux has control over which bit multiplies. Assume that when we multiply +6 by -6 in decimal, we get '0'. Upon expressing these integers in the form of a two's complement, we get +6 as 0110 and -6 as 1010. We get 10000 by adding these two binary values. If carries are discarded, the number is shown as '0'. The signed multiplication method offered by the Baugh-Wooley Multiplier is quite quick. It maximizes the consistency of the multiplication array by adjusting the partial products and using identical outcomes to complement multiplication.

The Baugh-Wooley multiplier embeds the number's sign when it is expressed in the two's complement form. When operands are more than or equal to 32 bits, Baugh-Wooley techniques acquire a strong area. The Baugh-Wooley multiplier has a lot of benefits. In comparison to other methods such as Booth's

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

algorithm, it provides decreased hardware difficulty, resulting in quicker multiplication speed. Its regular structure makes pipelining and parallelism simple and efficient when implemented in hardware. It can also support operands of different sizes and is expandable without requiring a large increase in hardware complexity.

3.2 Braun Multiplier

Thea Braun multiplier is a hardware-based binary multiplication technique designed to handle unsigned binary integers efficiently. This method, named for its developer, Franz Josef Braun, is favored for hardware implementations due to its efficiency, regularity, and simplicity. The Braun multiplier accepts two binary values as input: the multiplicand and the multiplier. It makes partial products efficiently by moving the multiplicand to the left and aligning it with each set bit of the multiplier. After that, these partial products are methodically added together to get the outcome.

The Braun multiplier's intrinsic simplicity is one of its greatest benefits. Its simplified functioning makes it easier to understand and implement in hardware circuits. Furthermore, because of its regular form, hardware design is made more efficient and allows for smooth integration utilizing standardized cells or building blocks. Although not the quickest multiplication approach available, the Braun multiplier excels in resource efficiency [25]. When compared to more complex algorithms, it usually requires fewer logic gates, which makes it perfect for applications where power efficiency or hardware resources are limited.

3.3. Booth Multiplier

The Booth multiplier uses Booth's algorithm, a systematic multiplication technique that significantly lowers the number of partial products created and added throughout the multiplication procedure. Initially, bitwise shifts and conditional addition based on Booth encoding are used to divide the multiplicand (M) and multiplier (Q) into partial products. The number of additions needed is then reduced by effectively aggregating these partial products and utilizing patterns in the multiplier's binary representation. This method produces an implementation that is hardware-efficient, using fewer partial product generators and adders. As a result, Booth multipliers provide quicker multiplication than traditional techniques, which is especially advantageous for applications like digital signal processing and encryption that need high-speed computing. Input signals for four Booth encoders/selectors consist of an 8-bit multiplicand and an 8-bit multiplier. The final output is obtained by simple addition following the application of Booth's algorithm to the inputs. To reduce the number of partial products that must be combined, one of the multiplier data inputs is often encoded using the Booth method, which encodes a binary number one bit-pair at a time to the signed-digit set [26].

3.4. Array Multiplier

An array multiplier is a hardware version of the multiplication function that is widely utilized in digital circuits. It operates by dividing the multiplication process into smaller, easier-to-manage parts and using a variety of functional units or logic gates to carry out the required calculations. The main idea is to multiply each multiplier digit by each multiplicand digit, adding the results, and then dividing the result to get the answer. This procedure is often performed in parallel, making array multipliers ideal for high-speed multiplication tasks. The multiplier circuit employs the add shift algorithm. The array multiplier's primary benefit is its regular form and straightforward construction. The drawbacks of an array multiplier include a long latency as well as high power usage. When multiplying an array, the number of partial products added must equal the number of multiplier bits [27].

3.5. Ripple Carry Adder

A sequence of complete adders equal to the number of bits makes up the ripple carry adder. The first complete adder will be given input carry (let's say Cin) and the initial bits of the two numbers (let's say A (0) and B (0)). The first full adder's output will be the first carryout and piece of the total, and this will be rippled to the next full adder and so on. Thus, the Ripple Carry Adder was created. Despite having a lower area usage, RCAs have a large circuit latency. Compared to the other adders that have been described, the RCA combines low area usage, high delay time, and higher power consumption.

3.6. Kogge Stone Adder

The Kogge Stone Adder (KSA) is known for its fast arithmetic operations. KSA employs a network of interconnected carry-look ahead units to effectively calculate binary addition by operating in parallel across many stages. KSA dramatically lowers the critical path delay as compared to conventional adders like the Ripple Carry Adder (RCA) by utilizing parallelism and a parallel prefix computing approach. This delay reduction improves processing performance and allows for effective carry propagation. Because of its organized layout, KSA can be implemented into contemporary integrated circuits with ease, allowing for effective use of available space. Its greater complexity, however, could make design verification and debugging more difficult, and in some situations, it might use more power than simpler adder designs. However, KSA is an attractive option for high-speed computing systems due to its remarkable performance, scalability, and appropriateness for Very Large-Scale Integration (VLSI).

High-speed operations are performed by the Kogge Stone Adder, which is also regarded as a parallel prefix version of the Carry Look Ahead Adder. Circuits and industries requiring high-speed operations heavily rely on this adder because it eliminates all time delays in the circuit required to create carry signals [28].

3.7. Proposed Adder

The proposed half-adder circuit makes use of simple gates and performs operations based on the inputs that are provided to it.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

The conventional half adder utilizes 3-AND gates with an OR gate, whereas for the proposed half adder requires 2-AND gates, 1-OR gate with a NOT gate as shown in the *figure 3* which utilizes less number of logic elements, less area and low power consumption.

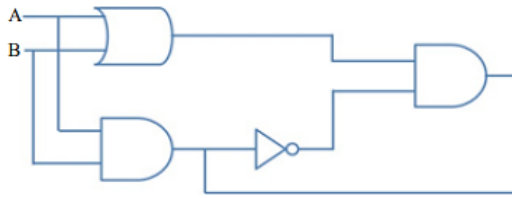

Figure 3. Proposed Half Adder

Two proposed half adders are used in the complete adder circuit together with an OR gate, and functionality was created. These suggested adders are used to create a RCA. The full adder is designed by using the above proposed Half adder as shown in *figure 4*.

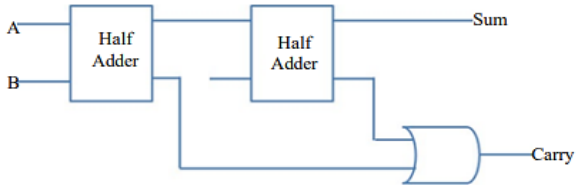

Figure 4. Proposed Full Adder

4. RESULTS & DISCUSSIONS

Since the FIR filter's low-area, high-speed, and low-power design is crucial, it is suggested that Braun, Baugh-Wooley, Booth, and Array multipliers be used in combination with Ripple Carry, Kogge Stone, and Proposed adders to design the FIR filter. This filter will be simulated and analyzed in Quartus and MATLAB tools. *Table 1* compares the Logic Elements and total thermal power dissipation of swift adders mentioned below with 4x4, 8x8, 16x16, 32x32, and 64x64 bit sizes. It is observed that the logic elements are decreased by 14.28% and 25% respectively, when comparing the proposed adder with ripple carry and kogge stone adders.

Table 1. Comparison of Logic Elements (LEs) and power dissipation for adders of various bit sizes

Swift-Adders	Specifications	4-Bit	8-Bit	16-Bit	32-Bit	64-Bit
RCA Adder	LEs	7	16	**32**	64	128
	Pd(mW)	65.21	66.34	**68.56**	73.11	82.13
Kogge Stone Adder	LEs	8	23	69	189	459
	Pd(mW)	65.31	66.43	68.69	73.21	82.24
Proposed Adder	LEs	6	8	16	32	64
	Pd(mW)	107.96	109.09	111.34	115.85	124.86

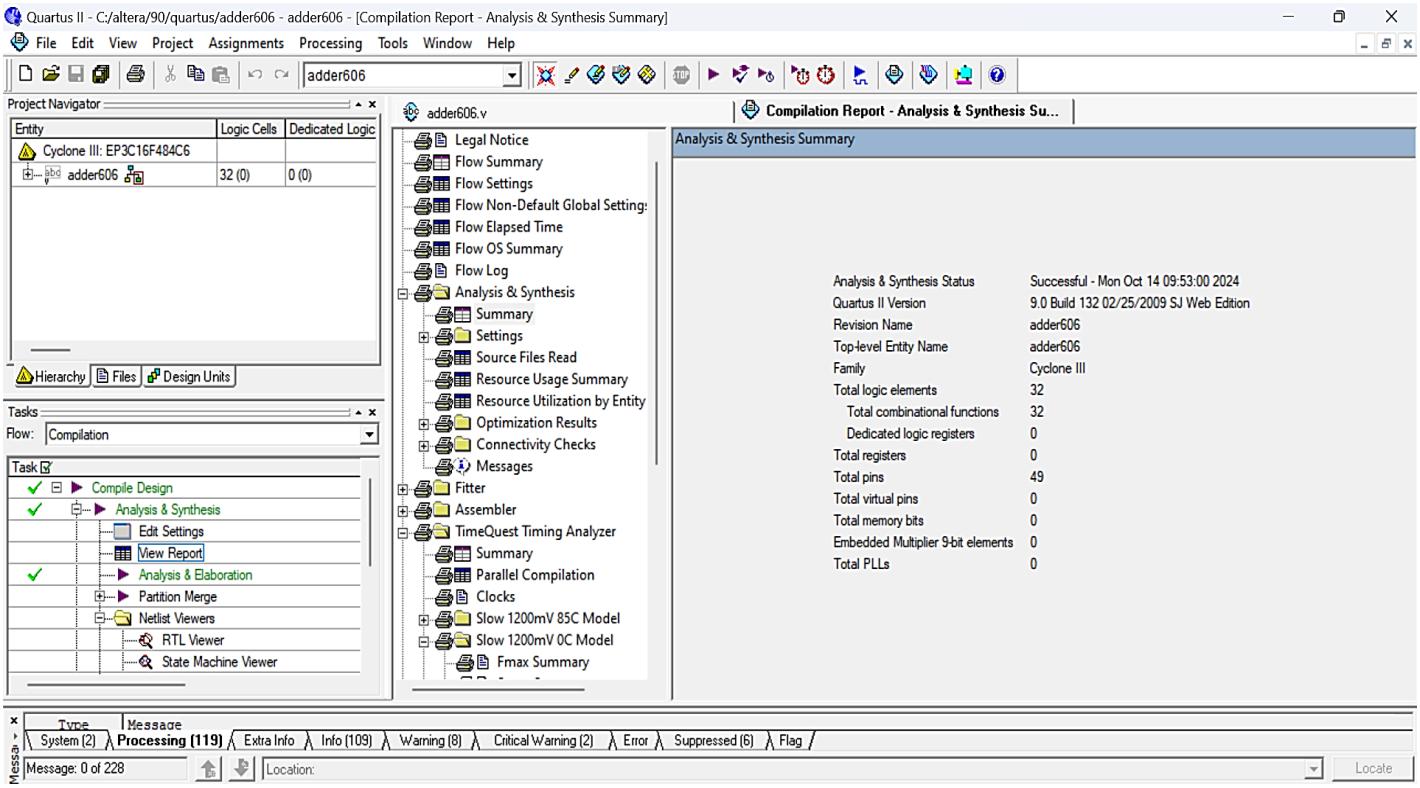

Figure 5. Logic Elements for 16-bit RCA adder

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

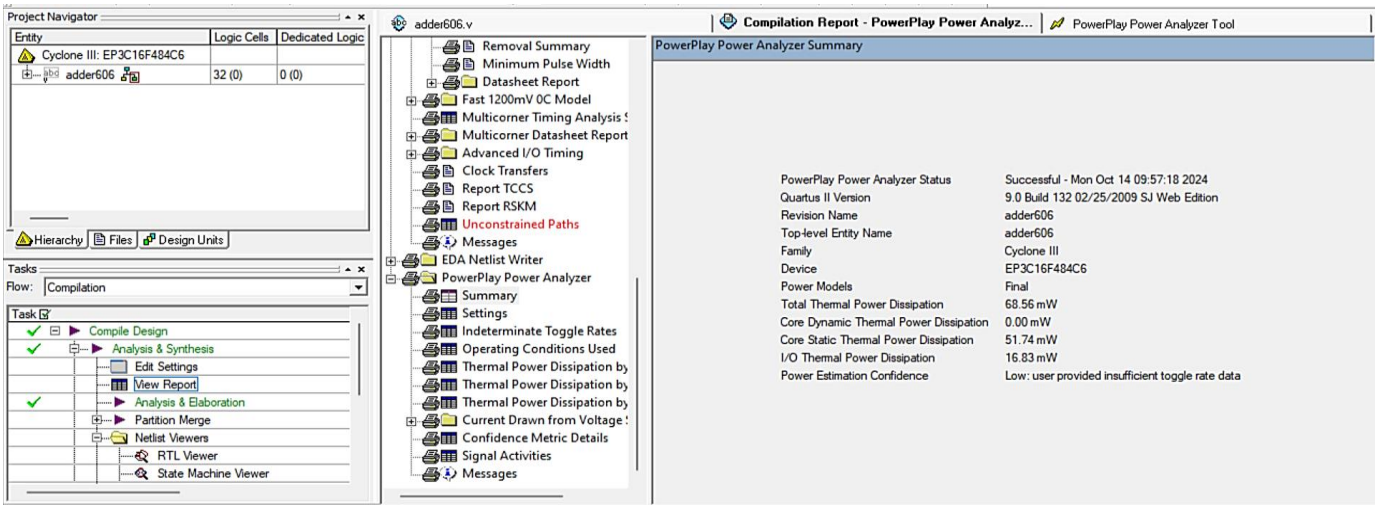

Figure 6. Power dissipation for 16-bit RCA adder

Table 2. Comparison of Logic Elements (LEs) and power dissipation for multipliers of various bit sizes

Swift-Multipliers	Specifications	4-Bit	8-Bit	16-Bit	32-Bit	64-Bit
Braun Multiplier	LEs	33	160	695	2871	11596
	Pd(mW)	65.50	67.00	70.03	76.10	88.43
Baugh-Wooley Multiplier	LEs	33	158	679	2808	11377
	Pd(mW)	65.50	67.00	70.02	76.10	88.41
Booth Multiplier	LEs	65	299	**1232**	5001	20094
	Pd(mW)	65.50	67.01	**70.01**	76.17	132.11
Array Multiplier	LEs	35	168	723	2975	12046
	Pd(mW)	65.50	67.00	70.03	76.11	88.44

In *table 2*, Braun, Baugh-Wooley, Booth, and Array multipliers were compared with 4, 8, 16, 32, and 64 bits each. The Logic Elements for 4-bit Braun, Baugh-Wooley, Booth, and Array multipliers are 33, 33, 65, and 35 respectively, whereas for 32-bit, they are 11596, 11377, 20094, and 12046. The power dissipation for all the multipliers of 4 bits is the same and for the remaining bits, they are nearly the same. In the suggested work, the Braun, Baugh-Wooley, Booth, and Array multipliers in different combinations with RCA, KSA, and Proposed adders were used to build the RNS FIR filter. For various tapping combinations (4, 8, 16, 32, 64) with varied bit sizes (4, 8, 16, and 32), the performance analysis parameters, such as area (LEs), power consumption, Fmax, and delay were examined.

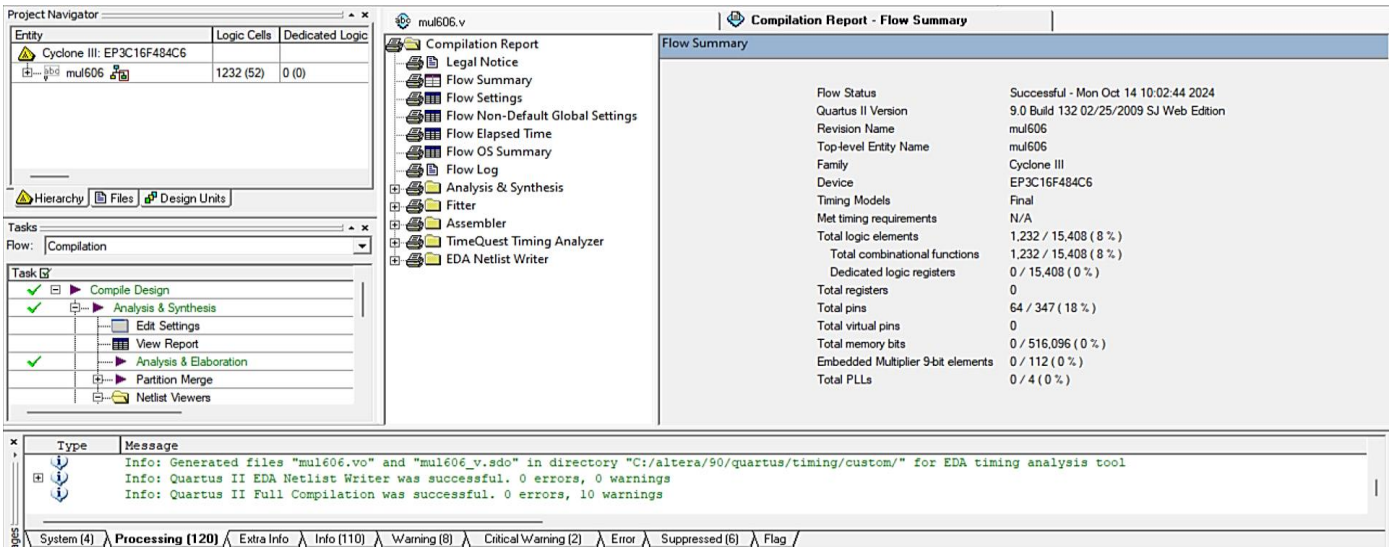

Figure 7. Logic Elements for 16-bit Booth multiplier

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

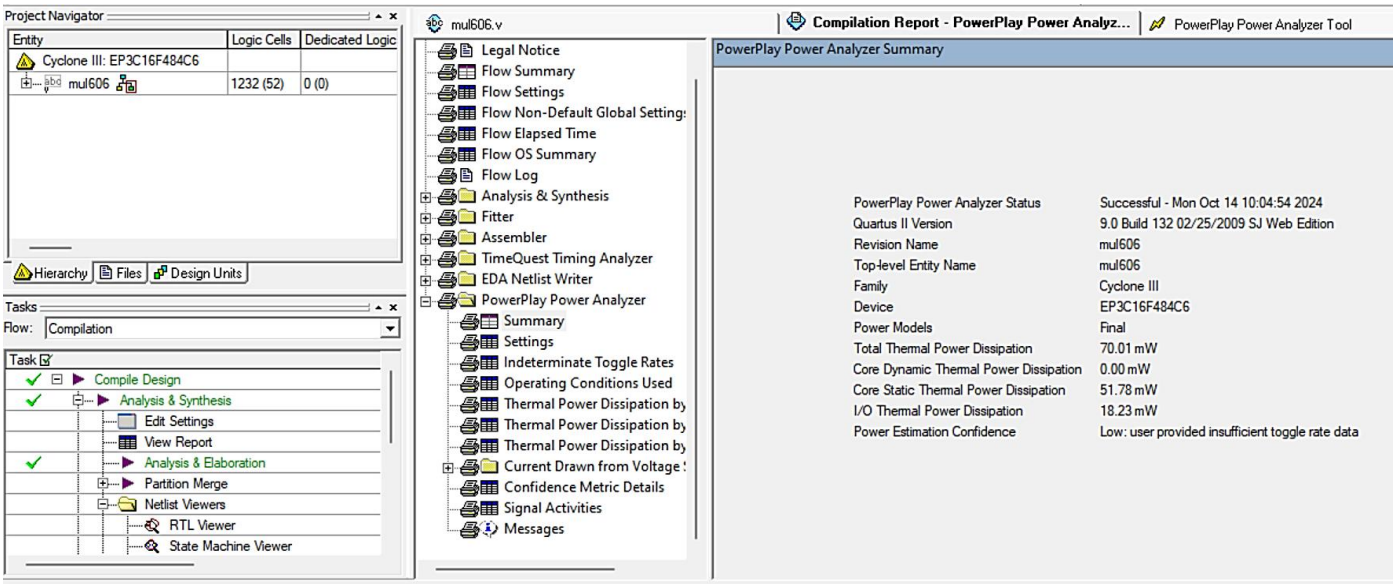

Figure 8. Power dissipation for 16-bit Booth multiplier

Table 3 compares the logic components for various combinations of swift adders and swift multipliers of an RNS-based FIR filter, ranging in word length from 4-bit to 32-bit and from 4-tap to 64-tap. When compared to various adder and multiplier combinations, the proposed adder with an array multiplier offers superior efficiency regarding less area. 11.76% of logic elements are saved for a 4bit, 52.70% for 8bit, 62.30% for 16bit, and 75.64% for a 32bit proposed adder with an array multiplier of 8-tap Residue Number System based Finite Impulse Response filter.

Table 3. Logic Elements (LEs) comparison for various combinations of swift adders and swift-multipliers

Parameters		Proposed Adder-Array Mul	Proposed Adder-Braun Mul	RCA Adder-Array Mul.	RCA Adder-Braun Mul	KSA Adder-Array Mul	Proposed Adder–Booth Mul	RCA Adder-Booth Mul	KSA Adder-Braun Mul	KSA Adder-Booth Mul	Proposed Adder–Baugh-Wooley Mul	RCA Adder-Baugh-Wooley Mul	KSA Adder-Baugh-Wooley Mul
4 Tap	4 Bit	548	536	547	554	594	539	542	584	580	560	560	592
	8 Bit	778	832	796	840	842	979	964	907	1119	1270	1285	1473
	16 Bit	1025	1079	1044	1087	1100	1233	1214	1153	1361	1708	1723	2136
	32 Bit	1518	1575	1538	1583	1580	1724	1708	1647	1854	3430	3441	4576
8 Tap	4 Bit	645	610	650	657	722	624	630	722	687	679	648	731
	8 Bit	**966**	**1016**	**1049**	**1051**	**1170**	**1193**	**1246**	**1247**	**1501**	**1619**	**1689**	**2044**
	16 Bit	1215	1264	1298	1301	1421	1441	1494	1495	1747	2243	2352	3223
	32 Bit	1705	1760	1791	1795	1911	1937	2243	1988	1979	4429	4641	7001
16 Tap	4 Bit	1010	1031	1000	1019	1206	1049	992	1223	1209	1102	1011	1277
	8 Bit	2100	2099	2299	2235	2971	2522	2583	2964	3528	3086	3241	4363
	16 Bit	2099	2148	2273	2228	2812	2492	2533	2876	3441	4090	4182	6749
	32 Bit	2845	2826	3038	2994	3773	3300	3393	3741	4729	8353	8516	16156
32 Tap	4 Bit	1614	1648	1615	1652	2019	1812	1656	2073	2114	1843	1640	2176
	8 Bit	4089	4256	4372	4490	5992	4769	4847	6215	7033	5559	5942	8261
	16 Bit	4574	4776	4942	5087	7372	5633	5779	7696	9227	8786	9203	15759
	32 Bit	5272	5504	5669	5846	8202	6474	6648	8497	10129	15322	15747	33898
64 Tap	4 Bit	2875	2904	2825	2902	3676	3376	2976	3744	3938	3285	2905	3929
	8 Bit	8515	8664	8824	8951	12776	9416	9414	12988	14086	10343	11165	16006
	16 Bit	10846	1142	11779	11943	16949	12448	12893	17375	19874	17131	18184	31762
	32 Bit	13383	13553	13871	14072	18142	15193	15522	18523	21138	30618	31818	70235

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

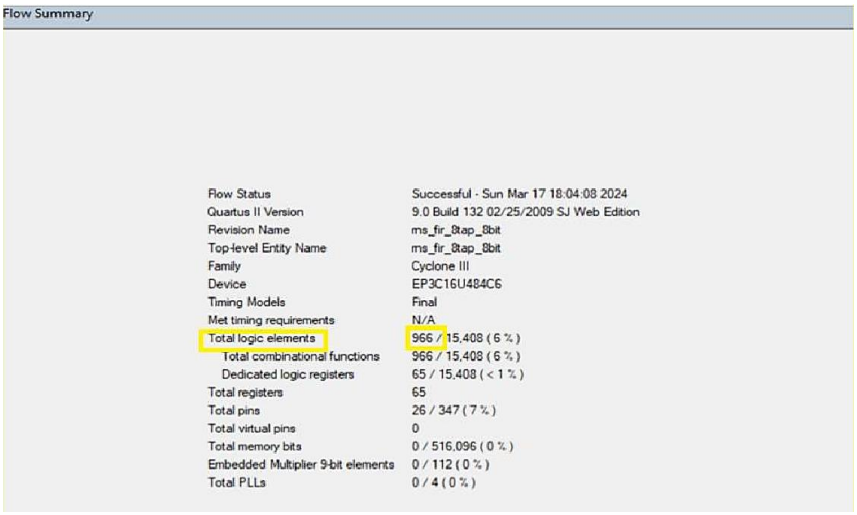

Figure 9. Logic Elements for the proposed adder with Array multiplier for 8-tap with 8-bit combination

Table 4. Area (Logic Elements usage) comparison of suggested outcomes with current findings for the 8-tap with an 8-bit word length combination

Design	Area (Logic Elements)
Proposed Adder - Array Multiplier	966
M. Balaji [12]	978
Proposed Adder-Braun Multiplier	1016
RCA Adder - Array Multiplier	1049
RCA Adder - Braun Multiplier	1051
KSA Adder - Array Multiplier	1170
Proposed Adder – Booth Multiplier	1193
RCA Adder – Booth Multiplier	1246
KSA Adder – Braun Multiplier	1247
Pavel Lyakhov [16]	1388
KSA Adder – Booth Multiplier	1501
Proposed Adder – Baugh-Wooley Multiplier	1619
RCA Adder – Baugh-Wooley Multiplier	1689
KSA Adder – Baugh-Wooley Multiplier	2044
D. Kaplun [17]	2456
C.W. Tung [18]	2637
Burhan Khurshid [14]	2789
G. Reddy Hemantha [13]	4281

The results of the area (LEs) used by the suggested works in contrast to the earlier works are shown in *table 4*. It is observed that the area utilized is reduced by 77.43% when the proposed adder with an array multiplier is compared with Reference [13] and with the latest proposed work of Reference [12], the area is reduced to 1.22%.

Table 5. Comparison of Delay for various combinations of swift adders and swift-multipliers

Parameters		RCA Adder-Array Mul	KSA Adder-Baugh-Wooley Mul	Proposed Adder-Array Mul	Proposed Adder-Braun Mul	RCA Adder-Baugh-Wooley Mul.	Proposed Adder-Booth Mul.	Proposed Adder-Baugh-Wooley Mul.	RCA Adder-Braun Mul	KSA Adder-Array Mul.	KSA Adder-Booth Mul.	RCA Adder-Booth Mul.	KSA Adder-Braun Mul.
4 Tap	4 Bit	0.500	0.563	0.499	0.541	0.512	0.492	0.559	0.500	0.512	0.561	0.520	0.537
	8 Bit	0.503	0.492	0.513	0.513	0.514	0.538	0.513	0.540	0.551	0.541	0.546	0.538
	16 Bit	0.492	0.494	0.506	0.493	0.510	0.543	0.514	0.543	0.508	0.536	0.545	0.514
	32 Bit	0.556	0.491	0.554	0.512	0.487	0.534	0.483	0.543	0.549	0.531	0.502	0.532
8 Tap	4 Bit	0.513	0.516	0.515	0.512	0.482	0.463	0.507	0.490	0.457	0.484	0.502	0.493
	8 Bit	**0.340**	**0.358**	**0.474**	**0.480**	**0.484**	**0.486**	**0.487**	**0.502**	**0.510**	**0.511**	**0.513**	**0.553**
	16 Bit	0.507	0.496	0.495	0.478	0.475	0.501	0.493	0.509	0.488	0.540	0.510	0.496
	32 Bit	0.483	0.490	0.294	0.473	0.497	0.500	0.472	0.509	0.493	0.543	0.515	0.526
16 Tap	4 Bit	0.501	0.491	0.500	0.503	0.504	0.499	0.503	0.503	0.497	0.501	0.503	0.500
	8 Bit	0.493	0.502	0.500	0.498	0.503	0.495	0.505	0.497	0.499	0.499	0.496	0.498
	16 Bit	0.500	0.333	0.507	0.500	0.376	0.502	0.311	0.499	0.495	0.500	0.500	0.500
	32 Bit	0.499	0.218	0.499	0.498	0.294	0.498	0.327	0.496	0.501	0.499	0.503	0.499
32 Tap	4 Bit	0.216	0.495	0.414	0.416	0.043	0.310	0.297	0.416	0.192	0.417	0.258	0.017
	8 Bit	0.041	0.052	0.193	0.419	0.022	0.416	0.046	0.013	0.103	0.411	0.414	0.146
	16 Bit	0.144	0.320	0.366	0.417	0.028	0.116	0.171	0.092	0.189	0.113	0.160	0.414
	32 Bit	0.416	0.313	0.179	0.119	0.208	0.418	0.080	0.189	0.089	0.298	0.092	0.189
64 Tap	4 Bit	0.041	0.100	0.027	0.099	0.072	0.124	0.024	0.065	0.181	0.062	0.009	0.008
	8 Bit	0.229	0.251	0.174	0.237	0.167	0.290	0.362	0.183	0.377	0.293	0.366	0.242
	16 Bit	0.274	0.513	0.194	0.197	0.408	0.219	0.347	0.204	0.335	0.317	0.290	0.285
	32 Bit	0.263	0.350	0.245	0.028	0.329	0.307	0.407	0.176	0.219	0.488	0.226	0.327

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

Table 5 compares delay for various combinations of swift adders and swift multipliers of an RNS-based FIR filter, ranging in word length from 4-bit to 32-bit and from 4-tap to 64-tap. When compared to the various adder and multiplier combinations, the RCA adder with an array multiplier offers better performance in terms of less delay. 11.43% of logic elements are saved for a 4bit, 38.51% for 8bit, 12.03% for a 16bit, and 45.58% for a 32bit RCA adder with an array multiplier of 8 tap Residue Number System based Finite Impulse Response filter.

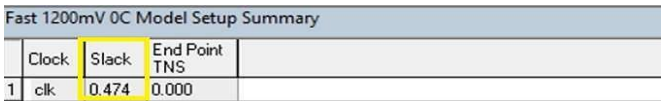

Figure 10. Delay for the proposed adder with Array multiplier for 8-tap with 8-bit combination

Table 6. Delay comparison of suggested outcomes with current findings for the 8-tap with an 8-bit word length combination

Design	Delay (ns)
RCA Adder - Array Multiplier	0.340
KSA Adder - Baugh-Wooley Multiplier	0.358
Proposed Adder - Array Multiplier	0.474
Proposed Adder-Braun Multiplier	0.480
RCA Adder - Baugh-Wooley Multiplier	0.484
Proposed Adder - Booth Multiplier	0.486
Proposed Add - Baugh-Wooley Multiplier	0.487
M. Balaji [12]	0.487
RCA Adder - Braun Multiplier	0.502
KSA Adder - Array Multiplier	0.510
KSA Adder - Booth Multiplier	0.511
RCA Adder - Booth Multiplier	0.513
KSA Adder - Braun Multiplier	0.553
T. K. Shahana [11]	2.55
Patronik [3]	4.67
R Kamal [5]	5.23
Shaheen Khan [12]	6.00
Patronik [3]	7.29

The results of the delay used by the suggested works in contrast to the earlier works are shown in *table 6*. It is observed that the delay produced is reduced by 95.33% when the RCA adder with an array multiplier is compared with Reference [3] and with the latest proposed work of Reference [12], the delay is reduced to 30.18%. *Table 7* compares the maximum frequency for various combinations of swift-adders and swift multipliers of an RNS-based FIR filter, ranging in word length from 4-bit to 32-bit and from 4-tap to 64-tap. When compared to the various adder and multiplier combinations, the KSA adder with braun multiplier offers superior efficiency regarding more maximum frequency. The Fmax of 8-tap 8-bit Kogge stone adder with braun multiplier is increased by 29.78%.

Table 7. Maximum Frequency comparison of various combinations of swift adders and swift-multipliers

Parameter		KSA Adder-Braun Mul	RCA Adder-Booth Mul	KSA Adder-Booth Mul.	KSA Adder-Array Mul.	RCA Adder-Braun Mul.	RCA Adder-Baugh-Wooley Mul.	Proposed Adder-Braun Mul	Proposed Adder-Array Mul.	Proposed Adder-Baugh-Wooley Mul.	Proposed Adder-Booth Mul.	KSA Adder-Baugh-Wooley Mul	RCA Adder-Array Mul
4 Tap	4 Bit	1340.48	1307.19	1414.43	1290.32	1251.56	1270.65	1355.01	1253.13	1406.47	1215.07	1430.62	1231.53
	8 Bit	1360.54	1381.22	1353.18	1388.89	1362.40	1256.28	1256.28	1254.71	1253.13	1344.09	1203.37	1254.71
	16 Bit	1287.00	1373.63	1345.9	1240.69	1366.12	1251.56	1201.92	1237.62	1256.28	1367.99	1236.09	1239.09
	32 Bit	1338.69	1222.49	1328.02	1386.96	1369.86	1197.60	1253.13	1390.82	1189.06	1355.01	1201.92	1398.60
8 Tap	4 Bit	1248.44	1231.53	1212.12	1177.86	1225.49	1196.17	1254.71	1254.71	1239.16	1186.24	1287.00	1256.28
	8 Bit	**1390.82**	**1254.71**	**1253.13**	**1250.00**	**1239.16**	**1221.00**	**1219.51**	**1209.19**	**1197.60**	**1193.32**	**1003.01**	**976.56**
	16 Bit	1223.99	1251.56	1356.85	1236.09	1250.00	1179.25	1203.37	1228.50	1245.33	1225.49	1246.88	1237.62
	32 Bit	1319.26	1259.45	1366.12	1213.59	1251.56	1245.33	1194.74	915.75	1175.09	1225.49	1293.16	1191.90
16 Tap	4 Bit	1230.01	1237.62	1230.01	1222.49	1237.62	1237.62	1230.01	1233.05	1237.62	1219.51	1207.73	1223.99
	8 Bit	1228.5	1226.99	1222.49	1226.99	1221.00	1236.09	1225.49	1225.49	1242.24	1225.49	1228.5	1212.12
	16 Bit	1226.99	1226.99	1223.99	1212.12	1222.49	1017.29	1225.49	1237.62	946.97	1228.50	962.46	1225.49
	32 Bit	1225.49	1239.16	1225.49	1225.49	1231.53	931.1	1225.49	1223.99	968.99	1225.49	818.33	1223.99
32 Tap	4 Bit	676.13	885.74	1066.10	809.72	1067.24	686.34	1063.83	1062.70	919.12	937.21	1226.99	841.04
	8 Bit	761.61	1062.7	1060.45	606.43	659.63	645.58	1072.96	816.99	707.21	1064.96	653.17	688.71
	16 Bit	1062.7	803.86	602.77	809.72	725.69	654.02	1070.66	1008.06	794.91	596.30	954.20	775.80
	32 Bit	812.35	600.96	521.65	728.33	801.92	827.81	754.72	808.41	704.23	1067.24	512.30	1067.24
64 Tap	4 Bit	663.13	671.59	630.91	564.33	687.92	628.93	601.68	671.59	651.89	771.60	607.53	634.52
	8 Bit	543.77	493.34	524.66	499.75	567.21	569.15	543.77	573.07	502.77	515.20	543.18	546.15
	16 Bit	529.38	513.35	504.29	509.42	555.86	475.29	563.06	562.11	507.61	561.80	458.09	525.76
	32 Bit	504.54	545.55	454.75	537.06	585.48	513.08	646.41	536.77	480.77	520.83	493.10	535.91

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

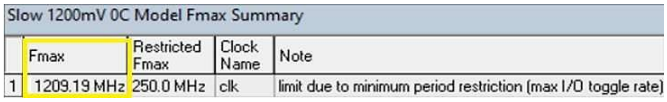

Figure 11. FMax for the proposed adder with Array multiplier for 8-tap with 8-bit combination

Table 8. Fmax comparison of suggested outcomes with current findings for the 8-tap with an 8-bit word length combination

Design	Maximum Frequency (MHz)
KSA Adder – Braun Multiplier	1390.82
RCA Adder – Booth Multiplier	1254.71
KSA Adder – Booth Multiplier	1253.13
KSA Adder - Array Multiplier	1250.00
RCA Adder - Braun Multiplier	1239.16
RCA Adder – Baugh-Wooley Multiplier	1221.00
Proposed Adder-Braun Multiplier	1219.51
CLA Adder, LUT Multiplier [12]	1219.51
Proposed Adder - Array Multiplier	1209.19
Proposed Adder – Baugh-Wooley Multiplier	1197.60
Proposed Adder – Booth Multiplier	1193.32
KSA Adder – Baugh-Wooley Multiplier	1003.01
RCA Adder - Array Multiplier	976.56
Burhan Khurshid (Direct Form) [14]	568.04
H.M. Kamboh [15]	535.00
Burhan Khurshid (Transposed form) [14]	525.88
Pavel Lyakhov [16]	285.00
D. Kaplun [17]	180.00

The results of the Fmax used by the suggested works in contrast to the earlier works are shown in *table 8*. It is observed that the Fmax produced is reduced by 87.05% when the KSA adder with braun multiplier is compared with Reference [17] and with the latest proposed work of Reference [12], the maximum frequency is increased to 12.31%.

5. CONCLUSION

The Residue Number System, with its built-in parallelism and effective arithmetic operations, speeds up the building of Finite Impulse Response (FIR) filters. The Finite Impulse Response based on the Residue Number System considers a variety of design factors, such as the quantity of taps, Fmax, critical path delay, power dissipation, and area. The Hardware description language called Verilog was used to execute the design sections, and ModelSim was used to verify their functioning. To create the Finite Impulse response architecture and define Residue Number System design criteria for the 65nm ALTERA CYCLONE III logic family, the Field Programmable Gate Array Quartus II 9 edition was used. When compared to various adder and multiplier combinations, the proposed adder with an array multiplier offers superior efficiency regarding less area. 11.76% of logic elements are saved for a 4bit, 52.70% for 8bit, 62.30% for 16bit, and 75.64% for a 32bit proposed adder with an array multiplier of 8 tap. The RCA adder with an array multiplier offers superior efficiency regarding less delay. 11.43% of logic elements are saved for a 4bit, 38.51% for 8bit, 12.03% for a 16bit, and 45.58% for a 32bit RCA adder with an array multiplier of 8 tap Residue Number System based Finite Impulse Response filter. The KSA adder with braun multiplier offers superior efficiency regarding maximum frequency. The Fmax of 8-tap 8-bit Kogge stone adder with braun multiplier is increased by 29.78%.

REFERENCES

[1] Shaheen Khan, and Zainul Abdin Jaffery, “Modified High-Speed FIR Filter Using DA-RNS Architecture,” International Journal of Advanced Science and Technology, vol. 29, no. 4, pp. 554-570, 2020.B. M, P. N, G. P, S. A. Shaik, S. K. P and S. G. R, "Design of FIR filter with Fast Adders and Fast Multipliers using RNS Algorithm," 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 2023, pp. 1-6.

[2] Balaji. M, Padmaja. N, Gitanjali. P, S. A. Shaik, Siva Kumar. P and Sai Geetesh. R, "Design of FIR filter with Fast Adders and Fast Multipliers using RNS Algorithm," 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 2023, pp. 1-6, doi: 10.1109/INCET57972.2023.10170321.

[3] P. Patronik and S. J. Piestrak, "Hardware/Software Approach to Designing Low-Power RNS-Enhanced Arithmetic Units," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 64, no. 5, pp. 1031-1039, May 2017.

[4] Y. Xu, Y. Guo and S. Kimura, "Approximate Multiplier Using Reordered 4–2 Compressor with OR-based Error Compensation," 2019 IEEE 13th International Conference on ASIC (ASICON), Chongqing, China, 2019, pp. 1-4.

[5] R. Kamal, P. Chandravanshi, N. Jain and Rajkumar, "Efficient VLSI architecture for FIR filter using DA-RNS," 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE), Hosur, India, 2014, pp. 184-187.

[6] Rao, E.J., Samundiswary, P. Error-Efficient Approximate Multiplier Design using Rounding Based Approach for Image Smoothing Application. J Electron Test 37, 623–631 (2021).

[7] K. Neelima, C. Padma, C. Nalini and M. Balaji, "FIR Filter design using Urdhva Triyagbhyam based on Truncated Wallace and Dadda Multiplier as Basic Multiplication Unit," 2023 IEEE 12th International Conference on Communication Systems and Network Technologies (CSNT), Bhopal, India, 2023, pp. 434-438, doi: 10.1109/ CSNT57126.2023.10134709.

[8] Garg, B., Patel, S. Reconfigurable Rounding Based Approximate Multiplier for Energy Efficient Multimedia Applications. Wireless Pers Commun 118, 919–931 (2021).

[9] Seyed Amir Hossein Ejtahed, and Somayeh Timarch, “Efficient Approximate Multiplier Based on a New 1-Gate Approximate Compressor,” Circuits Systems and Signal Processing, vol. 41, pp. 2699-2718, 2022.

[10] Mostafa Abbasmollaei et al., “A Power Constrained Approximate Multiplier with a High Level of Configurability,” Microprocessors and Microsystems, vol. 90, 2022.

[11] Shaghayegh Vahdat et al., “LETAM: A Low Energy Truncation-Based Approximate Multiplier,” Computers & Electrical Engineering, vol. 63, pp. 1-17, 2017.

[12] M. Balaji, N. Padmaja, "Area and Delay Efficient RNS-Based FIR Filter Design Using Fast Adders and Multipliers," SSRG International Journal of Electrical and Electronics Engineering, vol. 10, no. 10, pp. 151-164, 2023.

International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 4 | Pages 1211-1221 | e-ISSN: 2347-470X

[13] G. Reddy Hemantha, S. Varadarajan, and M.N. Giri Prasad, "FPGA Implementation of Speculative Prefix Accumulation-Driven RNS for High-Performance FIR Filter," Innovations in Electronics and Communication Engineering, pp. 365-375, 2019.

[14] Burhan Khurshid, and Roohie Naaz Mir, "An Efficient FIR Filter Structure Based on Technology-Optimized Multiply-Adder Unit Targeting LUT-Based FPGAs," Circuits System and Signal Processing, vol. 36, pp. 600-639, 2017.

[15] Hamid M. Kamboh, and Shoab A. Khan, "An Algorithmic Transformation for FPGA Implementation of High Throughput Filters," 2011 7th International Conference on Emerging Technologies, Islamabad, Pakistan, pp. 1-6, 2011.

[16] Pavel Lyakhov et al., "High-Performance Digital Filtering on Truncated Multiply-Accumulate Units in the Residue Number System," IEEE Access, vol. 8, pp. 209181-209190, 2020.

[17] Kaplun, D.; Aryashev, S.; Veligosha, A.; Doynikova, E.; Lyakhov, P.; Butusov, D. Improving Calculation Accuracy of Digital Filters Based on Finite Field Algebra. Appl. Sci. 2020, 10, 45. https://doi.org/10.3390/app10010045.

[18] Che-Wei Tung, and Shih- Hsu Huang, "A High-Performance Multiply-Accumulate Unit by Integrating Additions and Accumulations into Partial Product Reduction Process," IEEE Access, vol. 8, pp. 87367-87377, 2020.

[19] Morasa, Balaji, and Padmaja Nimmagadda. 2022. "Low Power Residue Number System Using Lookup Table Decomposition and Finite State Machine Based Post Computation." Indonesian Journal of Electrical Engineering and Computer Science 6(1): 127–34.

[20] P. Kishore, R. Sirimalla, K. S. Sushma and R. S. Reddy, "Implementation of Braun and Baugh-Wooley Multipliers Using QCA," 2023 2nd International Conference for Innovation in Technology (INOCON), Bangalore, India, 2023, pp. 1-4.

[21] M. Saha and A. Dandapat, "Modified Baugh Wooley Multiplier using Low Power Compressors," 2021 2nd International Conference for Emerging Technology (INCET), Belagavi, India, 2021, pp. 1-6.

[22] M. Balaji, N. Padmaja, A Low Power Transistor Level FIR Filter Implementation using CMOS 45nm Technology, International Journal of Nanotechnology, Vol. 20, No. 1-4, pp. 390-409, 2023.

[23] K. Yugandhar, V. G. Raja, M. Tejkumar and D. Siva, "High Performance Array Multiplier using Reversible Logic Structure," 2018 International Conference on Current Trends towards Converging Technologies (ICCTCT), Coimbatore, India, 2018, pp. 1-5.

[24] D. Govekar and A. Amonkar, "Design and implementation of high speed modified booth multiplier using hybrid adder," 2017 International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2017, pp. 138-143.

[25] M. Balaji, and N. Padmaja, "High-Speed DSP Pipelining and Retiming techniques for Distributed-Arithmetic RNS-based FIR Filter Design," WSEAS Transactions on Systems and Control, vol. 17, pp. 549-556, 2022.

[26] M. Sarkar, G. S. Taki, Prerna, R. Sengupta and S. N. Ray, "Design of ripple carry adder using CMOS output wired logic-based majority gate," 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), Bangkok, Thailand, 2017, pp. 328-33.

[27] J. Pathak and S. L. Tripathi, "Novel Obfuscated Secure Architecture for Baugh Wooley Multiplier," 2023 International Conference on Artificial Intelligence and Applications (ICAIA) Alliance Technology Conference (ATCON-1), Bangalore, India, 2023, pp. 1-4.

[28] Lavanya Maddisetti, Ranjan K. Senapati, and J.V.R. Ravindra, Image Multiplication with a Power-Efficient Approximate Multiplier Using a 4:2 Compressor, Advances in Image and Data Processing Using VLSI Design, Smart Vision Systems, 13th ed., IOP Publishing Ltd, pp. 13-15, 2021.

[29] Nikolay N. Kucherov et al., "A High-Speed Residue-to-Binary Converter Based on Approximate Chinese Remainder Theorem," 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), Moscow and St. Petersburg, Russia, pp. 325-328, 2018.