High-Speed DSP Pipelining and Retiming techniques for Distributed-Arithmetic RNS-based FIR Filter Design

M. BALAJI
Research Scholar, Department of ECE,
Jawaharlal Nehru Technological University Anantapur, Ananthapuramu,
Andhra Pradesh, INDIA

N. PADMAJA
Professor, Department of ECE,
Sree Vidyanikethan Engineering College,
Andhra Pradesh, INDIA

Abstract: - Digital FIR Filters plays a major role in many signal processing applications. Generally, these filters are designed with multipliers and adders to find the filter output. This paper acquaints how to reduce the complexity of higher order FIR filter by using performance optimization techniques like retiming and pipelining. The filter's throughput, energy efficiency, and latency, as well as the complexity of its technology, all need to be improved. By adopting pipelining technique, the arithmetic processes of addition and multiplication are separated. The break addition procedure is retimed. The architecture of Pipelining and Retiming with m-tap filters and n-bit word lengths were designed. The smallest delay achieved by the proposed distributed arithmetic-based FIR Filter with pipelining was 2.564ns for a 4tap implementation receiving an 8bit input, while the largest delay achieved was 56.04ns for a 64-tap implementation receiving a 32-bit word length. Delays as low as 0.68ns for a 4-tap implementation receiving an 8-bit input and as high as 4.53ns for a 64tap implementation receiving a 32bit word length have been achieved by using the suggested distributed arithmetic-based FIR Filter with retiming approach. Delay has been reduced by 73.2% for 4tap with 8bit input and by 91.9% for 64tap with 32bit word length compared to the pipelining approach.

Key-Words: - Pipelining, Retiming, Distributed Arithmetic, Area Delay Product, Power Delay Product.

Received: March 12, 2022. Revised: November 5, 2022. Accepted: December 6, 2022. Published: December 31, 2022.

1 Introduction

Finite Impulse Response (FIR) Digital filtering is one of the principal blocks in DSP systems. Multiplication processes, which result in the excessive area, delay, and power consumption, are the main issue with FIR filters, [4], [5]. Higher order filters will be difficult to construct in real-time since the number of multiply and accumulate operations needed per filter rises linearly with filter order, [6].

The majority of applications in real-time digital signal processing (DSP) systems need high throughput and energy-efficient FIR filter designs. In feed-forward control systems, FIR filters are often employed to provide an inverse response filter for the plant, [1]. To enhance the computing speed, various design techniques such as pipelining and parallel processing have been widely employed. These techniques aim to exploit the parallelism in the computing algorithms so that both the data processing rate (or throughput rate) and the total computing time can be improved.

Pipelining and retiming were used to increase the throughput rate at which input data is processed. Pipelining shortens the critical route, increasing throughput at the expense of longer delay [2], [3].

Traditionally high throughput is obtained by dividing the combinational logic into a number of stages separated by registers/latches controlled by a global clock. Various clocking styles can be used depending on the type of applications. To minimize the complexity of higher order FIR filter by using performance optimization techniques like pipelining method and retiming method.

The use of effective retiming for outsized circuits is covered in [7]. In [8] the retiming methods used with sequential circuits are covered by Jalaja et al. From the viewpoint of complexity, Jiang and Brayton, [9], synthesized the logic design. The multiplier-less multiple constant multiplication (MCM) technique has been retimed by Yagain and Vijaya, [10]. Meher, [11], has used cut-set and flexible retiming to enhance the CPD of traditional direct form filters at the expense of more sophisticated hardware. Carry increment adders (CIA) and Vedic multipliers were used by Thakral et al. [12], to apply an unfolding transformation to an FIR filter. Wallace multiplier was used by Swati and Himanshu, [13], to build the FIR filter. According to [14] discusses two variations of enhanced booth multipliers and efficient hybrid adders.

For the implementation of the FIR filter, Rai et al. [15], employed carry increment adders and Vedic multipliers and improved the CPD and energy efficiency at the expense of a larger filter area than the traditional direct form FIR filter. Ting et al. G. R. Hemantha et al. [16], proposed block RAM based conversion model where critical path optimization is difficult and may be implemented with retiming technique. According to Meher and Park [17] employed two-level pipelining to boost the throughput of the FIR filter used in the adaptive algorithm at the expense of a modest increase in area and latency.

Meher [18] looked at several pipelining options for DSP circuits. By pipelining to structural adders, Mathias et al. [19], enhanced the CPD of transposed form FIR filters at the price of more sophisticated hardware. Using a modified transposed form FIR filter, Pramod and Shahana, [20], created a high throughput adaptive filter design. In [21], it is addressed how the retiming transformation approach may be used to create low-power VLSI designs. Burhan et al [22], adopted Constant Matrix Vector Multiplication (CMVM) instead of conventional multiplication, where delay can be optimized by using pipelining method. Using retiming and redesigned CSLA-based adders, Pramod and Shahana [23], created two types of high throughput FIR filters.

The rest of this paper is structured as follows. The Distributed Arithmetic Residue Number System is explained in section 2. The proposed pipelining and retiming methods were discussed in section 3. The results and discussion with a comparison of the performance of filters are done in Section 4. The section 5 concludes the work proposed.

2 Distributed Arithmetic – Residue Number System

One memory-based effective multiplier-less architecture utilized in the design of FIR filters is distributed arithmetic (DA) architecture. The bit length of the input data determines how long it takes to compute a design in DA architecture. High-speed digital signal processing may be implemented utilizing filters that are realized using the residue number system (RNS). The RNS is a non-weighted number system that simultaneously divides binary values with a wider range into several smaller numbers. By employing these little integers in the math calculation, the critical route time is decreased. RNS is an addition, subtraction, and multiplication carry-free system. RNS is hence capable of operating at high speed. The following graphic displays the block diagram of an RNS-based FIR filter, [24].

In the Residue Number System, the three main components are Forward Conversion, Arithmetic manipulations, and Reverse conversion as shown in figure 1. The Chinese remainder theorem is used to map back from residue to the binary number during reverse conversion, and the binary to residue conversion using modulo operation may be carried out in the forward converter, [25].

Step– 1: Consider the RNS system to be $\{2^n-1, 2^n, 2^n+1\}$.

Let n=3, then the RNS bases become $\{2^3-1, 2^3, 2^3+1\}$

= {7, 8, 9}.

Let the moduli set = {7, 8, 9}

Let the input be 7, then the equivalent binary is '111' then the LUT value chosen from the table 1 is

A0+A1+A2 = 3+11+15 = 29.

Then the RNS Sequence generated will be

$|29|7=1$

$|29|8=5$

$|29|9=2$

i.e., the converted RNS Number is {1, 5, 2}.

Step– 2: Performing the related arithmetic operations say if it is a filter operation, then

Hence x= {1, 5, 2}, let h = {8, 4, 2, 1}

$$y[n] = \sum_{n=1}^{N} x[n]h[n]$$

Table 1. DA-LUTs-based FIR Filter with the coefficients A0=3, A1=11, A2=15

b1	b2	b3	Coefficients or Entry value	m=7 (2^n-1)	m=8 (2^n)	M=9 (2^n+1)
0	0	0	0	0	0	0
0	0	1	A2=15	$\|15\|_7=1$	$\|15\|_8=7$	$\|15\|_9=6$
0	1	0	A1=11	$\|11\|_7=4$	$\|11\|_8=3$	$\|11\|_9=2$
0	1	1	A1+A2=26	$\|26\|_7=5$	$\|26\|_8=2$	$\|26\|_9=8$
1	0	0	A0=3	$\|3\|_7=3$	$\|3\|_8=3$	$\|3\|_9=3$
1	0	1	A0+A1=14	$\|14\|_7=0$	$\|14\|_8=6$	$\|14\|_9=5$
1	1	0	A0+A2=18	$\|18\|_7=4$	$\|18\|_8=2$	$\|18\|_9=0$
1	1	1	A0+A1+A2= 29	$\|29\|_7=1$	$\|29\|_8=5$	$\|29\|_9=2$

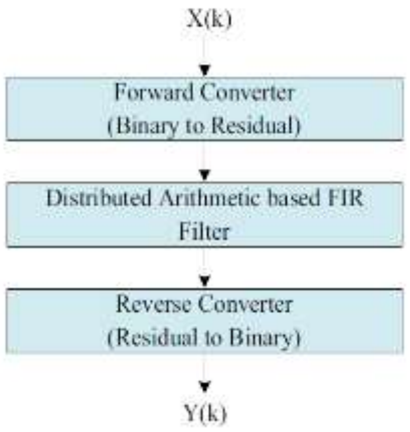
X(k)
Forward Converter (Binary to Residual)
Distributed Arithmetic based FIR Filter
Reverse Converter (Residual to Binary)
Y(k)

Fig. 1: RNS-based FIR Filter pipelined architecture

Then y = {1*8+1*4+1*2+1*1, 5*8+5*4+5*2+5*1, 2*8+2*4+2*2+2*1} = {15, 75, 30}

The reverse converter is developed based on the Chinese remainder theorem i.e., say if the input to the reverse converter is 291,
Then m1 = 7, m2 = 8, m3 = 9 then
M = 7 x 8 x 9 = 504

Table 2. Calculated values of Power Product and Inversion values for CRT

rem (remainder)	pp (power product)	inv (inversion)
15	M/m1=72	2
75	M/m2=63	7
30	M/m3=56	2

The Python code to generate the inversion value of m as shown in table 3 to satisfy $MM^{-1}=1$ is

```
i=0
If [{(pp*i) % m}]==1
out = i
else
i=i+1
```

RNS output = (Σ(rem x pp x inv)) % M
= ((15 x 72 x 2) + (75 x 63 x 7) + (30 x 56 x 2)) % 504
= (2160+33075 + 3360) % 504
= 38595 % 504
=291

3 Proposed Pipelining and Retiming Techniques

The pipelining architecture of the RNS based design is shown in figure 2. By adding extra latches between the multipliers, the pipelined approach shortens the critical path. In an M-level pipelined system, there are (M-1) more delay elements on every channel from input to output than there were in the corresponding path in the initial sequential circuit. Although it shortens the critical path, the consequence is higher latency. The longest route between any two latches, an input and a latch, an input and an output, or a latch and a latch is what determines the speed of the DSP architecture (or the clock period). By properly positioning the pipelining latches in the DSP design, the longest route, or critical path, maybe cut down.

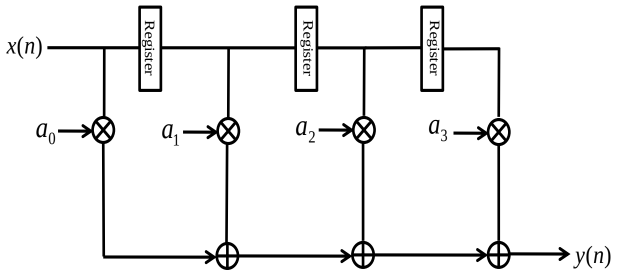

Fig. 2: Architectural view of RNS-based Pipelining Design

Retiming is a transformation method used to modify the placements of the delay elements in a circuit without changing the circuit's features for input/output as shown in figure 3. Retiming may be used to boost a circuit's clock rate by decreasing the calculation time of the critical route. Retiming is another technique for reducing the number of registers in a circuit. Retiming has several uses in synchronous circuit design. Applications for this technology include decreasing the circuit's clock period, power consumption, and clock period, as well as logic synthesis. Changing the placements of the delay elements during retiming will have no impact on the circuit's input/output characteristics.

The critical path, Logic Elements, and power consumption of filter design may all be decreased by using the retiming algorithmic technique. Retiming and two-level pipelining are used in this case to reduce the length of the critical route. It is restricted to two layers since adding further levels of pipelining will add more delay. The whole filter will be divided into two portions by the two-level pipelining. It should be mentioned that the retiming won't cause delay. The pipelining delay distinguish the multiplication and addition operations and limits the critical path delay to T_{max}.

$$T_{CP} = T_{max} = \max\{T_m, T_a\} \quad (1)$$

Where T_m is the worst case combinational path delay of the multiplication section and T_a is the worst case combinational path delay of the addition section.
For lower filter lengths

$$T_{CP} = T_m \quad (2)$$

The total addition time T_a becomes greater than the multiplication time T_m as the filter length increases. That is

$$T_{CP} = T_a \quad (3)$$

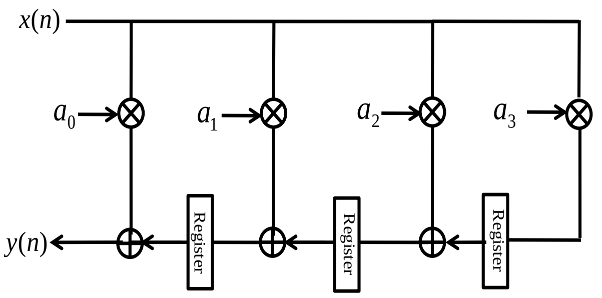

Fig. 3: Retiming circuit of FIR Filter design

Here, the addition portion has been retimed to decrease the worst-case addition latency. Here, the FIR filters have been developed using this method. The clock period and the number of registers needed to construct the circuit will be reduced in these retiming approach. In figure 4, by rearranging the registers from input to output the delay has been reduced. While pipelining reduces the critical path delay, it leads to a penalty in terms of an increase in latency. Latency essentially is the difference in the availability of the first output data in the pipelined system and the sequential system. For example, if latency is 1 clock cycle, then the kth output is available in (k+1)th clock cycle in a 1-stage pipelined system. As a result, each computation takes 2 clock cycle to propagate through the system, and a new computation is available at the output register every clock period. The insertion of register reduces the maximum delay of the combinational block, and therefore the overall system clock period can be reduced.

Retiming is a key and effective approach for improving the performance of digital circuits. It works by shifting the placement of a circuit's essential delays to equalize the combinational path delays over the whole design. Retiming is a technique that involves shifting the locations of the delay components without changing the circuit's input-output characteristics. It assists in reducing the circuit's clock period, power consumption, and logic synthesis. It is used to boost the circuit's clock rate. One of the key factors in a device's performance is speed. Digital signal processing systems, which use various methods including pipelining, parallel processing, retiming, unfolding, etc., are utilized to satisfy the requirement for high speeds. The applications of retiming are Register Minimization, Retiming for Folding, Retiming for Power Reduction, Retiming for Logic Synthesis, and Multi-Rate/Multi-Dimensional Retiming.

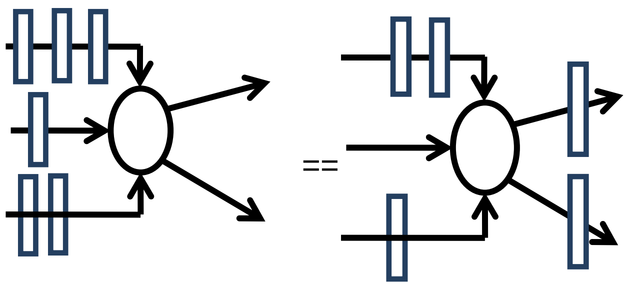

Fig. 4: Diagrammatic representation of reducing registers in retiming method

4 Results and Discussion

In general, the various parameters used for evaluation are area, delay, power, power delay product, energy-delay product, the maximum frequency of operation, order of filters, number of taps, minimum period cycle, energy per sample, etc. The area is evaluated in terms of the number of logical elements used for design i.e., Look-up tables or sliced registers.
The delay is the total combinational delay encountered when implemented on FPGA. The power consumption is evaluated as the combined static and dynamic power consumption by the design. The figure of merit of the designs is considered as the product of delay and power consumption. The area delay and power

consumption must be as minimum as possible. The energy-delay product is evaluated as the product of power and square of delay time. The order of the filter gives the desired frequency characteristics of the filter. The number of taps defines the filter length and the required memory. The energy per sample is considered as the sampling rate of the filters.

Table 3. Performance analysis of FIR filter with a pipelining technique for different taps with different bit lengths

Parameter		LE's used	Delay (ns)	Power (mW)	Fmax (MHz)
4 tap	4 bit	562	2.631	65.29	275.41
	8 bit	798	2.564	66.43	280.58
	16 bit	1046	2.639	68.69	274.80
	32 bit	1584	2.646	73.21	274.27
8 tap	4 bit	613	5.444	65.30	155.18
	8 bit	1071	6.795	66.44	128.29
	16 bit	1319	6.811	68.70	128.02
	32 bit	1863	6.852	73.22	127.36
16 tap	4 bit	613	5.444	65.30	155.18
	8 bit	1446	14.301	66.45	65.36
	16 bit	1721	15.598	68.71	60.25
	32 bit	2293	15.685	73.23	59.93
32 tap	4 bit	613	5.444	65.30	155.18
	8 bit	1446	14.301	66.45	65.36
	16 bit	2065	31.390	68.72	30.87
	32 bit	3412	33.251	73.27	29.20
64 tap	4 bit	2299	8.918	65.33	100.83
	8 bit	3313	16.087	66.49	58.52
	16 bit	3563	16.231	68.75	58.03
	32 bit	9879	56.040	73.50	17.53

A delay of 2.564ns was achieved for a 4tap, 8bit input using the suggested distributed arithmetic-based FIR Filter with pipelining approach, while a delay of 56.04ns was achieved for a 64tap, 32bit word length. The delay produced in the circuit is directly proportional to the number of taps and input word length. The area (LEs) utilized by the 4 taps with 4bit input is 562, and for the 64 taps with 32bit input word length is 9879. The maximum frequency produced by a 4 tap with 4 bit is 275.41MHz, and for 64 taps with a 32bit word, the length is 17.53MHz.The maximum power consumed for different taps with different bits is varied from 65.29mW to 73.50mW. The performance analysis for 4 tap to 64 tap with 4 bit to 32 bit was compared in Table 3.

Delays as low as 0.687ns for a 4tap implementation receiving an 8-bit input and as high as 4.535ns for a 64tap implementation receiving a 32bit word length have been achieved by using the suggested distributed arithmetic-based FIR Filter with retiming approach. The delay produced in the circuit is directly proportional to the number of taps and input word length. The area (LE) utilized by the 4 tap with 4-bit input is 555, and for the 64 tap with 32-bit input word length is 12237. The maximum frequency produced by a 4 tap with 4 bit is 550.36MHz, and for 64 tap with 32-bit word, the length is 180.67MHz. The maximum power consumed for different taps with different bits is varied from 65.29mW to 73.54mW. The performance analysis for 4tap to 64tap with 4-bit to 32-bit was compared in Table 4.

Delays have been reduced by 73.2% for 4tap with 8bit input and 91.9% for 64tap with 32-bit word length as compared to the pipelining approach. There is a slight variation in power consumed for all the 64tap, 32tap, 16tap, 8tap, and 4tap with 32-bit, 16-bit, 8-bit, and 4-bit word lengths when the pipelining method is implemented.

Table 4. Performance analysis of FIR filters with a retiming technique for different taps with different bit lengths.

Parameter		LE's used	Delay (ns)	Power (mW)	Fmax (MHz)
4 tap	4 bit	555	0.817	65.29	550.36
	8 bit	782	0.687	66.43	592.77
	16 bit	1030	0.767	68.69	565.93
	32 bit	1568	0.760	73.21	568.18
8 tap	4 bit	647	1.341	65.30	427.17
	8 bit	1152	0.912	66.44	523.01
	16 bit	1400	1.044	68.70	489.24
	32 bit	1946	0.952	73.22	512.30
16 tap	4 bit	676	1.512	65.30	398.09
	8 bit	1696	1.044	66.45	489.24
	16 bit	1999	1.100	68.71	476.19
	32 bit	2611	1.020	73.23	495.05
32 tap	4 bit	671	1.513	65.30	397.93
	8 bit	1716	1.513	51.77	398.09
	16 bit	3234	1.513	68.73	397.93
	32 bit	4798	1.219	73.28	450.65
64 tap	4 bit	2281	1.513	65.33	397.93
	8 bit	4163	2.177	66.50	314.76
	16 bit	6989	2.648	68.85	274.27
	32 bit	12237	4.535	73.54	180.67

Distributed Arithmetic decreases the delay and power consumed as in Modified Binary DA, but the size of LUT is large. Partitioned LUT, decreases the size of LUT thereby reducing the power consumed and delay. RNS system reduces the input signal bit width and with parallel computation of filters the

delay is reduced a lot but due to the addition of an encoder and a decoder, it consumes slightly higher power compared to circuits without RNS.
The logic elements are reduced by 66.79% and 74.67% and Fmax is increased by 3.25 times and 1.37 times in 4-tap and 16-tap filter lengths respectively when compared with the RNS DA-based computation. When compared with the previous works in Table 5, pipelined DA RNS-based FIR filter is decreased by 57.01% in the area (LE's), decreased by delay by 22.55%, decreased by 66.69% in ADP, PDP was decreased by 22.39% and increased by 19.47% in maximum frequency.

When compared with the previous works in Table 6, the critical path delay in the retiming method was 0.687ns whereas for pipelining method it is 5.39ns. The area occupied by the pipelining method is less when compared with the retiming method. The retiming method gives the best performance parameters when compared with the pipelining method in terms of delay decreased by 87.25%, whereas in terms of the area, decreased by 27.74%, and decreased by 83.33% in ADP.

Table 5. Comparison of Retiming technique with already existing methods

FIR model used	LE’s used	Fmax (MHz)	Power (mW)	Delay (ns)	ADP (um^2 ns)	PDP (pJ)
Modified Binary DA [9]	1644	121.58	65.22	7.22	11869.68	470.88
Partitioned DA [9]	1363	123.11	64.09	7.12	9704.56	456.32
Memory Less DA I [10]	1557	112.28	65.38	7.90	12300.30	516.50
Memory Less DA II [10]	1314	125.58	65.16	6.96	9145.44	453.51
Pipelined DA RNS FIR	565	156.47	65.30	5.39	3045.91	351.96
Retiming DA RNS FIR	782	512.77	66.43	0.687	537.23	45.63

Table 6. Overall comparison of proposed method

Design/ Parameters	Critical Path Delay (ns)	Area (μm^2)	ADP (mm^{2}ns)
G N Jyothi et.al [1]	15.30	6271.6	0.096
Shaheen Khan et.al [2]	6.001	2084.0	0.012
G. Reddy Hemantha et.al [3]	14.00	5432.0	0.076
Burhan Khurshid et. al [4]	13.28	6825.8	0.091
Chitra, E at. al [5]	16.84	7943.1	0.134
Patronik et. al [6]	17.48	5353.9	0.093
Patronik et. al [6]	4.672	9800.3	0.046
Patronik et. al [6]	7.072	8987.0	0.063
Patronik et. al [6]	7.296	6525.0	0.048
Patronik et. al [6]	7.280	6478.9	0.047
R Kamal et. al [7]	5.235	9632.0	0.050
T. K. Shahana et. al [8]	2.552	7795.0	0.020
Modified Binary DA [9]	7.22	6945.9	0.011
Partitioned DA [9]	7.12	5758.6	0.009
Memory Less DA I [10]	7.90	6578.3	0.012
Memory Less DA II [11]	6.96	5551.6	0.009
Pipelined DA RNS FIR	5.39	2387.1	0.003
Retiming DA RNS FIR	0.687	3303.9	0.0005

5 Conclusion
The RNS system aids in the high-speed design of FIR Filters with various moduli sets, and forward and reverse converters. The design is enhanced in speed by using pipelining and retiming techniques and evaluated for various parameters like area (number of Logic Elements), delay, power, ADP, PDP, the maximum frequency of operation, order of filters, and the number of taps. For an 8-bit input and 4 taps with retiming mechanism of the

distributed arithmetic-based FIR filter, the resulting delay is 0.687ns, whereas for 64 taps and 32-bit word length, it is 4.535ns. DSP systems benefit from the RNS's increased efficiency because of its capacity for performing parallel and rapid calculation operations. Verilog HDL was used to define all the design modules, and ModelSim was used to test and validate their operation. To meet the needs of the ALTERA CYCLONE III logic family devices in 65nm technology, the RNS design requirements, and FIR filter designs were generated in the FPGA QUARTUS II 9.0 online version. This paper compares various RNS FIR filters and the design with retiming proves to be a better option with improvement in performance in terms of delay. For an 8-bit input and a 4-tap FIR filter, the suggested distributed arithmetic-based FIR filter with pipelining approach yields a delay of 2.56ns, while for 64 taps and 32 bits, the delay increases to 56.04ns. Delays as low as 0.687ns for a 4-tap implementation receiving an 8-bit input and as high as 4.535ns for a 64tap implementation receiving a 32-bit word length have been achieved by using the suggested distributed arithmetic-based FIR filter with retiming approach. Delay has been reduced by 73.2% for 4tap with 8-bit input and by 91.90% for 64-tap with 32-bit word length compared to the pipelining approach.

Future Recommendations

Low power design techniques can be applied for retimed and pipelined RNS FIR filters. Also reconfigurability to the filter can be added.

References:

[1] G. N. Jyothi, K. Sanapala, and A. Vijayalakshmi, "ASIC implementation of distributed arithmetic based FIR filter using RNS for high-speed DSP systems," *International Journal Speech Technology*., Vol. 23, No. 2, 2020, pp. 259–264.

[2] 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, 2020, pp.554-570.

[3] G. Reddy Hemantha, S. Varadarajan, and M. N. Giri Prasad, "FPGA Implementation of Speculative Prefix Accumulation-Driven RNS for High-Performance FIR Filter", *Springer, Singapore*, Vol. 65. 2019,

[4] Burhan Khurshid, RoohieNaaz Mir, "An Efficient FIR Filter Structure Based on Technology-Optimized Multiply-Adder Unit Targeting LUT-Based FPGAs", *Circuits System and Signal Processing*, 2016.

[5] Chitra, E., T. Vigneswaran, and S. Malarvizhi. "Analysis and implementation of high performance reconfigurable finite impulse response filter using distributed arithmetic." *Wireless Personal Communications*, Vol. 102, No. 4, 2018, pp. 3413-3425.

[6] Patronik, Piotr, and Stanisław J. Piestrak. "Hardware/Software Approach to Designing Low-Power RNS-Enhanced Arithmetic Units." *IEEE Transactions on Circuits and Systems I*: *Regular Papers* Vol. 64, No. 5, 2017, pp. 1031-1039.

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

[8] T. K. Shahana, R. K. James, B. R. Jose, K. Poulose Jacob, and S. Sasi, "Performance analysis of FIR digital filter design: RNS versus traditional," International Symposium in Communication Information Technology Proceedings, November, 2007, pp. 1–5.

[9] M Balaji, and N. Padmaja, "High-Speed Pipelined Architecture-Based Residue Number System For FIR Filter Design" *Gongcheng Kexue Yu Jishu/Advanced Engineering Science*, Vol. 54, Issue. 6, 2022, pp. 2056-2066.

[10] C. H. Vun, A. B. Premkumar, and W. Zhang, "A new RNS based DA approach for inner product computation," IEEE Transactions on Circuits and Systems, Regular Paper, vol. 60, no. 8, pp. 2139–2152, 2013.

[11] 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* Vol. 6, No. 1, 2022, 127–34.

[12] S. Jalaja and V.A.M. Prakash, "Different retiming transformation technique to design optimized low power VLSI architecture," *AIMS Electronics and Electrical Engineering,* Vol. 2, No. 4, 2018, pp. 117-130.

[13] J.R. Jiang and R.K. Brayton, "Retiming and resynthesis: a complexity perspective, " *IEEE Trans. Computer-Aided Design of Integration Circuits Systems,* Vol. 25, No. 12, 2006, pp. 2674–86.

[14] D. Yagain and K.A. Vijaya, "Design of synthesizable, retimed digital filters using FPGA based path solvers with MCM approach: comparison and CAD tool,' *VLSI Design, Hindawi Pub. Corp*., 2014, Article ID 280701.

[15] P.K. Meher, "On efficient retiming of fixed-point circuits," *IEEE Transaction on VLSI Systems,* Vol. 4, No. 4, 2016.

[16] S. Thakral, D. Goswami and R.S. Sharma, "Design and implementation of a high-speed digital FIR filter using unfolding," *Proc. IEEE 7th Int. Con., Bikaner,*2016, India.

[17] C. Swati and J. Himanshu, "FIR filter designing using wallace multiplier," *International Journal of Engineering and Technical Research*, Vol. 3, No. 6, 2015, pp. 276-78.

[18] P. Pramod and T.K. Shahana, "Efficient modular hybrid adders and Radix-4 booth multipliers for DSP applications', *Microelectronics Journal*, 2020, Print ISSN 0026-2692.

[19] Chakrabarty, R, S Roy, T Pathak, and N Kumar Mandal. "Design of Area Efficient Single Bit Comparator Circuit Using Quantum Dot Cellular Automata and Its Digital Logic Gates Realization." *International Journal of Engineering*, Vol. 34, No. 12, 2021, pp. 2672–78.

[20] P.K. Meher and S.Y.Park, "Critical-path analysis and low-complexity implementation of the LMS adaptive algorithm," *IEEE Trans. Circuits Syst.-I*: Reg. Papers, Vol. 61, No. 3, 2014, pp.778-788.

[21] F. Mathias, K. Martin and Chip-Hong, "Efficient structural adder pipelining in transposed form FIR filters," *Proc. IEEE Int. Conf. Digital Signal Processing (DSP),* Singapore, 2015, pp. 21-24.

[22] P. Pramod and T.K. Shahana,(2018) "High throughput adaptive filter architecture using modified transpose form FIR filters," *Journal of Advanced Research in Dynamical and Control Systems*, Vol. 10, No. 15, 2018, pp. 68-82.

[23] R Kamal, Piyush Chandravanshi, Neha Jain, Rajkumar, "Efficient VLSI architecture for FIR filter using DA-RNS", *2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE),* pp. 184-187, 2014.

[24] Burhan Khurshid, Roohie Naaz Mir, "An Efficient FIR Filter Structure Based on Technology-Optimized Multiply-Adder Unit Targeting LUT-Based FPGAs", *Circuits System and Signal Process*, 2016.

[25] S.N. Rai, B.S.P. Shree and Y.P. Meghana, "Design and implementation of 16 tap FIR filter for DSP applications," *Proc. Sec. Int. Conf. Adv. in Electron. Comp. and Comm.*, Bengaluru, India, 2018, pp. 9-10.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

M Balaji carried out the simulation, optimization and implementation of FIR filter in Quartus tool. **N Padmaja** contributed by supervising the proposed works and statistical analysis was also done.