International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 2, April 2021, pp. 1709~1718
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i2.pp1709-1718

Design and implementation of DA FIR filter for bio-inspired computing architecture

B. U. V. Prashanth[1], Mohammed Riyaz Ahmed[2], Manjunath R. Kounte[3]
[1,3]School of Electronics and Communication Engineering, REVA University, India
[2]School of Multidisciplinary Studies, REVA Universiy, India

Article Info

Article history:

Received Apr 23, 2020
Revised Jul 27, 2020
Accepted Sep 16, 2020

Keywords:

Bio-inspired computing
Distributed arithmetic
Finite impulse response
MAC and parallel filters
Processor architecture
Systolic array

ABSTRACT

This paper elucidates the system construct of DA-FIR filter optimized for design of distributed arithmetic (DA) finite impulse response (FIR) filter and is based on architecture with tightly coupled co-processor based data processing units. With a series of look-up-table (LUT) accesses in order to emulate multiply and accumulate operations the constructed DA based FIR filter is implemented on FPGA. The very high speed integrated circuit hardware description language (VHDL) is used implement the proposed filter and the design is verified using simulation. This paper discusses two optimization algorithms and resulting optimizations are incorporated into LUT layer and architecture extractions. The proposed method offers an optimized design in the form of offers average miminimizations of the number of LUT, reduction in populated slices and gate minimization for DA-finite impulse response filter. This research paves a direction towards development of bio inspired computing architectures developed without logically intensive operations, obtaining the desired specifications with respect to performance, timing, and reliability.

Corresponding Author:

B. U. V. Prashanth
School of Electronics and Communication Engineering
REVA University
Rukmini Knowledge Park, Yelahanka, Bengaluru-560064, India
Email: prashanthbuv@reva.edu.in

1. INTRODUCTION

High throughput is required since the finite impulse response filters are used intensively in video, communications systems as well as bio-inspired computing systems. Essentially, digital filters are used in time and frequency domain to adjust the characteristics of the signals and are identified as the primary digital signal processing feature [1]. The DSP design techniques focus mainly on multiplier-based architectures for multiply-and-accumulate (MAC) blocks implementation which represent the FIR filters and several functions. High speed parallel filter designs are elucidated in excruciating detail. Finite impulse response (FIR) filters are prominent building blocks for several applications in the field of digital signal processing (DSP). High-speed FIR filters have been widely used to perform signal equalization on the received data in real time due to the increasing demand for video-signal processing and transmission. Therefore a structured VLSI architecture is needed for a programmable fast FIR filter [2].

The various FIR Filters were suggested in last few decades, many structures and different algorithms have been utilized for the enhamcement of the filter weights. The very common structures utilized were least mean square (LMS) derived models since their response in convergence is strong. Block processing with distributed arithmetic methods is explored to derive a design that should give high throughput [3]. The

parallelism assists in minimizing the number of clock cycles desired for partial product calculation. This increases the proposed processing speed as compared with current systems.

Distributed arithmetic (DA) is a strategy of high-speed multiplication which is a bit serial word parallel technique where the throughput rate does not depend on the data size. The DA facilitates to avoid the multipliers in the design and makes the area of the system efficient in the throughput and several DA based structures were designed in order to minimize the area and to reduce the cost of processing [4]. The primary operations necessary for DA-based processing are a series of accesses to a lookup table (LUT), preceded by the LUT output's shift-accumulation operations. The standard framework of DA used to implement the FIR filter implies that the coefficients of the impulse response are fixed and this action allows use of ROM based LUTs. However, with linear filter order the memory requirement for Distributed Arithmetic implementation of FIR filters rises exponentially is one of the hard problems to be addressed [5]. The key contributions of this research are:

- Develop systolic array architecture with tightly coupled co-processor based data processing units.
- Develop optimization algorithms with optimizations incorporated into LUT layer with architecture extractions and propose bio inspired computing architecture to compute FIR filters at high processing speeds using reconfigurable computing based on DA strategy.

2. RELATED WORK

Modular finite-impulse response (FIR) filter whose filter coefficients switch dynamically during latency, which plays a major role in architectures for software-defined radio (SDR), multi-channel filters, bi-inspired computing and digital up/down converters. However, when the filter coefficients vary dynamically, the well-known multiple constant multiplication (MCM)-based methods that are widely used to realize the FIR filters cannot be used. Addressing to the solution to the problem of such large memory requirement, systolic decomposition techniques are utilized for DA-based implementation of long-length convolutions and FIR filter of large orders. It is necessary to use rewritable RAM based LUT instead of ROM based LUT for reconfigurable DA based FIR filter whose filter coefficients alter dynamically. Another method is to store the analog domain coefficients using serial digital to analog converters, resulting in mixed-signal architectures [6].

A pipelined design for an adaptive FIR filter carry out the save accumulation technique which is used for partial inner product calculation that facilitates in enhancing the throughput with block processing is utilized in increasing the computational speed of the system. On the other hand, a particular multiplier-based structure requires a wide chip region, and thereby controls limitations on the highest allowable order of the filter that can be interpreted for high-throughput applications [7]. In recent years, distributed arithmetic (DA)-based technique has gained substantial popularity due to its high capacity for processing throughput and increased regularity, resulting in cost-effective and area-time efficient computing structures.

The primary operations required for DA-based processing are a sequence of accesses to a lookup table (LUT), followed by the LUT output's shift-accumulation operations [8]. The conventional implementation of the DA used to implement the FIR filter assumes that the coefficients of the impulse response are fixed and this behavior allows the use of ROM based LUTs. However, with the filter order the memory requirement for DA-based implementation of FIR filters increases exponentially [9].

The systolic decomposition techniques are used to get rid of the problem of such a large memory requirement. For long-length convolutions and large-order FIR filter for DA-based implementation, we must use rewritable RAM based LUT instead of ROM based LUT for reconfigurable DA-based FIR filter whose filter coefficients change dynamically. Another approach is to store the coefficients in the analog domain by using serial digital to analog converters resulting in mixed-signal architecture. We also find quite a few works on DA based implementation of adaptive filters, where the coefficients change at every cycle [10].

3. PROPOSED METHOD AND ALGORITHM DESIGN

Distributed arithmetic is a popular architecture without the use of multipliers to implement FIR filters. DA makes efficient use of LUTs, shifters, and adders to calculate the sum of products required for FIR filters. Since these operations effectively map onto an FPGA, Distributed arithmetic on these devices is a favourable architecture [11].

The Figure 1 illustrates the experimental design of the research work presented in this manuscript. Distributed Arithmetic is a prominent architecture without the use of multipliers to implement FIR filters. DA makes efficient use of LUTs, shifters, and adders to calculate the sum of multiplication factors needed for FIR filters. Though distributed arithmetic implements the FIR filter by serialization bits of inputs, a filter quantisation is required. Due to the fixed data path requirements in input analog to digital converter (ADC)

and the output digital to analog converter (DAC) widths the length of the word with 12 bit input and output with 11 fractional bits are assumed to be required to quantize the FIR filter [12].

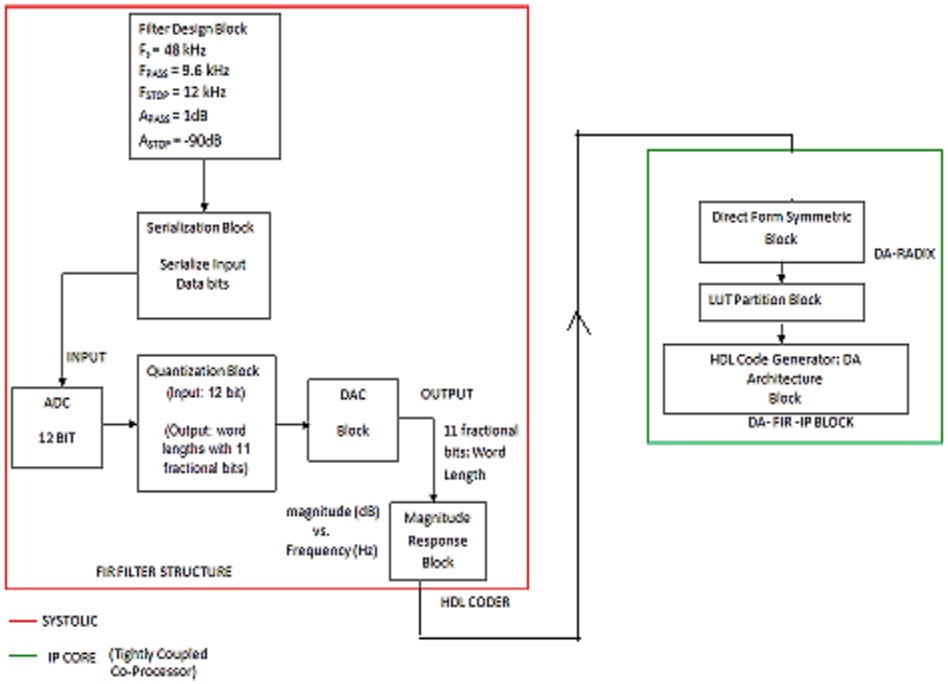


Figure 1. Block diagram of experimental design

After the quantization process the HDL Code is generated with DA architecture. The HDL code generator uses distributed arithmetic architecture, and partitions the look-up-table (LUT) into a specified number of LUT partitions with the range of taps each partition associates. It is best to divide the taps into a number of LUTs for a filter with many taps, with each LUT storing the sum of coefficients only for the taps that are associated with it.

The FIR filter structure has symmetric coefficients, and we consider converting the structure to reduce the area. Here we convert the filter structure to direct form symmetric and generate the HDL code for default radix of 2. In hardware, a symmetrical filter structure offers advantages, as it halves the number of coefficients to work with which substantially reduces the complexity of the hardware. The predefined architecture is an implementation of Radix 2 that runs on one bit of input data per clock period. Before an output is obtained, the number of clock phases elapsed is equal to the number of bits in the input data and DA may effectively limit the throughput. DA can be configured to process multiple bits in parallel, to improve the DA throughput. The processing of 12 bits at a time for a 12 bit input word length can be specified with the corresponding DA-Radix values of 2^{12}. The speed vs. area is trade off by selecting different 'DARadix' values and the amount of parallel bits illustrates the factor with the increased rate of the clock which is the number of cycles to perform an iteration [13]. The Tables 1-3 elucidate the information of DA architecture. The Table 1 depicts the 'DARadix' values with corresponding values of number of cycles to perform an iterationand multiple for LUT sets for the given filter. Further Table 2 illustrates the details of LUTs with corresponding 'DALUTPartition' values. Details of LUT indicate number of LUTs with the sizes of LUT for example (1x1024x18) implies 1 LUT of 1024 18-bit wide locations [14].

Table 1. 'DARadix' values with number of cycles to perform an iterationand multiple for LUT sets

Folding Factor	LUT Sets Multiple	DA Radix
1	12	2^{12}
2	6	2^{6}
3	4	2^{4}
4	3	2^{3}
6	2	2^{2}
12	1	2^{1}

Table 2. Details of LUTs with corresponding partitions of DA-LUT

Address Width	Size (bits)	LUT	DA LUT Partition
W_{12}	b'259072	[1024x13], [13x4096], [14x4096], [15x4096], [18x4096]	(12) (12) (12) (12) (10)
W_{11}	b'147544	[2x2048x14], [1x2048x18], [2x2048x13], [1x8x11]	(11) (11) (11) (11) (11) (3)
W_{10}	b'78080	[1x1024x18], [3x1024x13], [1x1024x16], [1x256x13]	(10) (10) (10) (10) (10) (8)
W_{09}	b'43712	[1x512x12], [1x512x14], [1x512x18], [1x16x12], [2x512x13], [1x512x15]	(9) (9) (9) (9) (9) (9) (4)
W_{08}	b'25384	[1x4x10], [1x256x14], [1x256x18], [4x256x13], [1x256x15]	(8) (8) (8) (8) (8) (8) (8)(2)
W_{07}	b'14248	[1x128x14], [1x128x16], [1x4x10], [2x128x12], [3x128x13], [1x128x18]	(7) (7)(7) (7) (7) (7) (7) (2)
W_{06}	b'8000	[4x64x12], [1x16x12], [1x64x17], [2x64x14], [1x64x16], [1x64x13]	one(9,1)*(6), (4)
W_{05}	b'4696	[1x8x11], [4x32x12], [3x32x13], [1x32x15], [1x32x14], [1x32x17], [1x32x11]	one(1,11)*(5), (3)
W_{04}	b'2904	[1x16x15], [5x16x12], [2x16x13], [2x16x14], [1x16x17], [1x4x10], [3x16x11]	one(1,14)*(4),(2)
W_{03}	b'1926	[8x8x12], [2x8x14], [2x8x15], [1x8x17], [1x2x7], [5x8x11], [1x8x13]	one(1,19)*(3), 1
W_{02}	b'1412	[12x4x11], [2x4x13], [2x4x15], [1x4x17], [2x4x10], [6x4x12], [4x4x14]	one(1,29)*(2)

Table 3. Tabular column of complete twiddle factor for each LUT inputs

Folding Factor	LUT Inputs	LUT Size	LUT Details
1	LUT_4	S(34848)	(1x4x10, 5x16x12, 2x16x14, 2x16x13, 1x16x15, 1x16x17, 3x16x11)x12
2	LUT_4	S(17424)	(3x16x17, 5x16x12, 2x16x14, 2x16x13, 1x16x15, 1x16x11, 1x4x10)x6
3	LUT_4	S(11616)	(1x4x10, 5x16x12, 2x16x14, 2x16x13, 1x16x17, 1x16x15, 3x16x11)x4
4	LUT_4	S(8712)	(1x4x10, 5x16x12, 2x16x14, 2x16x13, 1x16x17, 1x16x15,3x6x11)x3
6	LUT_4	S(5808)	(1x4x10, 5x16x12, 2x16x14, 2x16x13, 1x16x17, 1x16x15,3x6x11)x2
12	LUT_4	S(2904)	(1x4x10, 5x16x12, 2x16x14, 2x16x13, 1x16x17, 1x16x15,3x6x11)x1

As depicted in Table 3, if it is required to increase the clock rate by four scales the sampling frequency and utilize six input LUTs then we can verify that the details of LUT meets the area requirements. Next a test bench is designed with a standard setup, and uses a simulator to verify the generated code for distributed arithmetic architecture [15]. The synthesis tool is utilized to compare the area and speed of the DA architecture. The Algorithm 1 illustrates the performance analysis and optimation of LUT layer. As shown in Algorithm 2, the cost function could be any arbitrary parameters delay, power or power delay multiplication (PDM) returned from optimized LUT.

Algorithm 1: Performance analysis and optimization of LUT layer

```
Result: Optimization of LUT Layer
Start
Optimize LUT (Addr bits: k, num LUTs: m)
     Delay(LUTi,1) ← dlut[j] ;        for all  j  set of  [k]
     Power(LUTi,1)← plut[j]           for all  j  set of  [k]
     Power Delay(LUTj,1)← pdlut[j];   for all  j  set of  [k]
While {Read the Input Parameters}{
                    for (i=2; i <= k; i++)
                    for (j=2; j <= m; j++)
                    else If{Perform Optimization}
{
                    Delay(LUTi,j)←minu{max{ dlut[u] +D(LUTi-u,j-1)}
};
            P(LUTi,j)←minima w{plut[w] + Power(LUTi-w,j-1)};
            PD(LUTi,j)←minima{Delay(LUTi,j.Power Utilization (LUTi,j)};
         end for
                    }{Compute:return Delay(LUTk,m), Power(LUTk,m) Power Delay(LUTk,m)}
                    Calculate the performance;
}
Stop
```

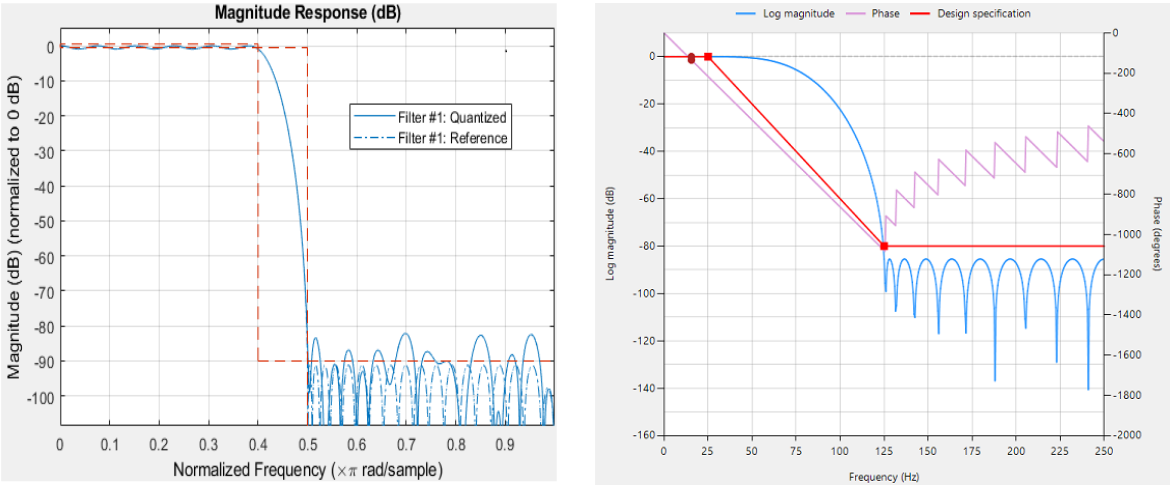
Algorithm 2: Algorithm steps to optimized architecture extractions

```
Result: Optimized Architecture Extractions
Start;
         Define parameters;
         While {Read the Input Parameters}
{
                  Architecture Optimize (N:Filter Order):
```

```
                Optimized Solution← infinity;
                Select cost from (Delay | Power | Power Delay Muliplication)
        for (i=1; i <= N; i++)
        for (j=1; j <=i; j++)
        else If{Perform Optimization}
        {
        ArchCost = cost(OptimizeLUT(i,j))
        if (Architecture Cost )
                Optmized response←Architecture Cost;
        end for
        end for
        }
Compute:
        return:
        Optimum Solution
        Calculate the performance;
Stop
}
```

4. RESULTS AND DISCUSSIONS

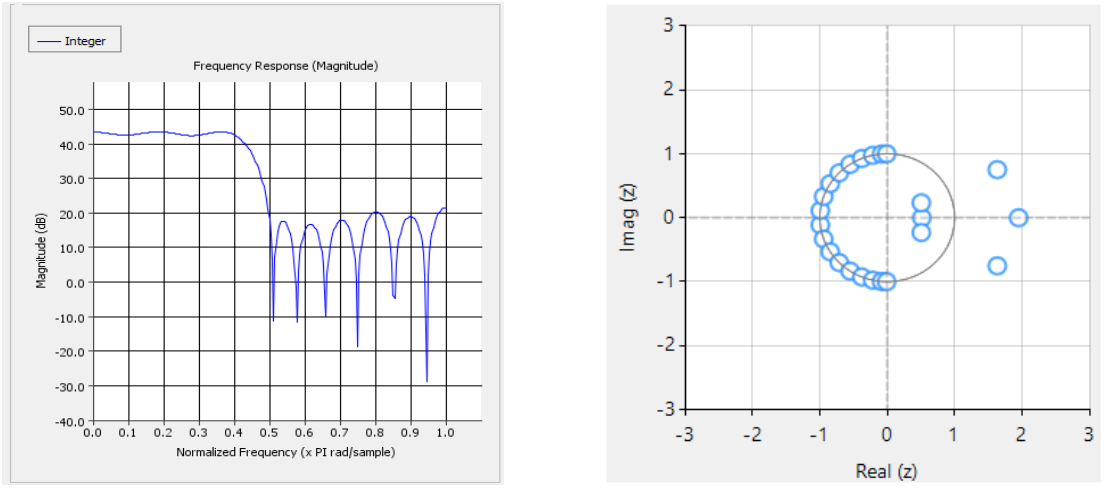
The fixed point settings are applied in order to obtain the characteristic plot of magnitude response (dB) indicating the curves between the magnitude (dB) and the normalized frequency (π radians per sample) with the comparison between reference and quantized filter as depicted in Figure 2(a). The characteristic plot representing the complete design specification of DA FIR filter along with the Log magnitude (dB) and phase (degrees) is as depicted in Figure 2(b). In this case the full precision override is not considered and custom coefficient data type is considered in the design. With the optimizations addressed by variations in architectural level enhancements using DA concept of digital filtering which improves device utilization [16, 17].



(a) (b)

Figure 2. Plot of (a) magnitude response (dB), (b) log magnitude (dB)-phase (degrees)

Here the clock rate is four times the input sample rate for this architecture and the effective filter length for serial partition value is 58 along with three samples of HDL latency, achieved with the FIR compiler and the corresponding frequency response diagram obtained in FIR compiler is as depicted in Figure 3(a) and with reference to this the pole-zero (P-Z) diagram is as depicted in Figure 3(b). Because of mid-stage pipelining, the entire architecture is split into two sections, namely the input section and the output section. Here the power consumption of the DA architecture is estimated at 20 MHz frequency and the final DA architecture is designed using the systolic rearrangement of delay elements. The preconfigured logic functions, that is the intellectual property (IP) cores optimized for FPGAs is generated using FIR compiler and Figure 4 illustrates the block design to verify the DA FIR filter responses as obtained in the Figure 2 and Figure 3.



(a) (b)

Figure 3. Plot of (a) frequency response (dB), (b) pole-zero (P-Z) diagram

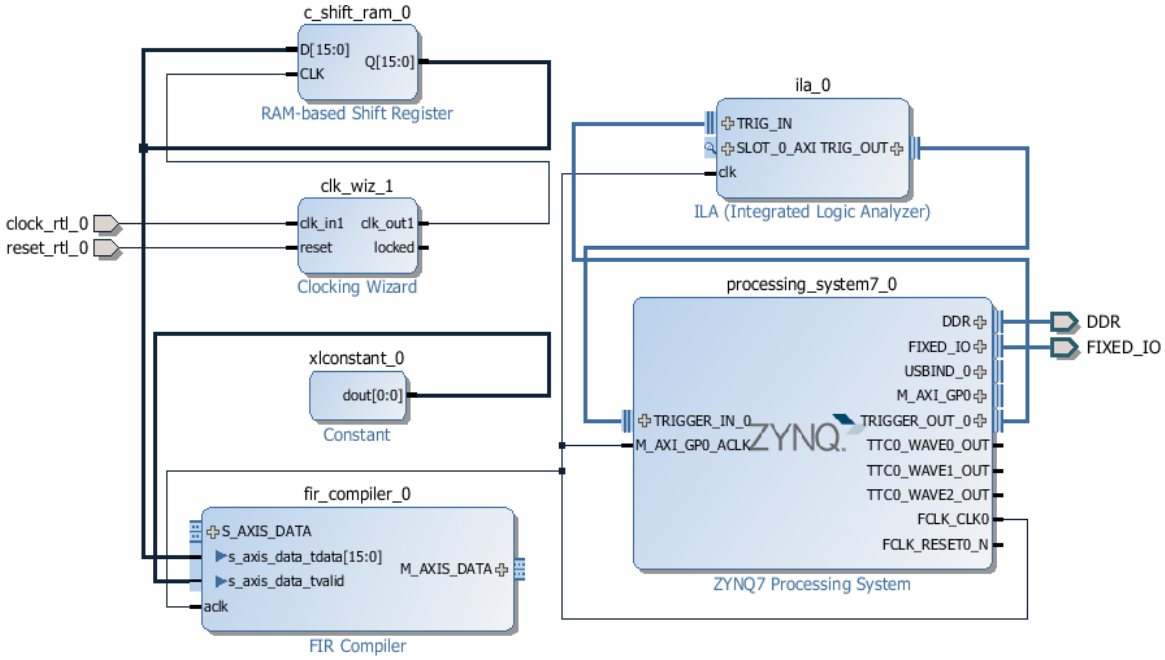


Figure 4. A system construct of DA-FIR filter optimized for ZYNQ FPGA

As depicted in Figure 4 the RAM based shift register is having 16 bit width and 16 bit depth is configured as a circular buffer and it is initialized with memory initialization radix and memory initialization vector of 16-bits as arbitary waveform generator and on every cycle of 100 MHz clock, the shift RAM outputs the last sample first and proceeds towards the initial sequence and loops back. Further the complete DA FIR filter is processed using the ZYNQ FPGA as a special purpose tightly coupled processor. The Figure 5 illustrates the performance evaluation of the design with behavioral simulation of DA FIR Filter obtained in Xilinx ISE environment with phase (phase 0, 3) and serial (serial out 1, 2, 3) and the Figure 6 depicts the performance evaluation with analysis of filter coefficient values.

The Figure 7 compares the proposed DA FIR filter design with the previous designs available in [18-21] in terms of number of multipliers versus the filter order as depicted in Figure 7(a). In Figure 7(b) the number of adders versus order of Filter is illustrated along with the LUT optimization with number of LUTs versus order of filter in Figure 7(c), the Figure 7(d) represents the number of registers versus the filter order. The estimated delay based on (Gate delay-DG) for Distributed Arithmetic unit of LUT, LUT-less and

proposed architecture implementation [22, 23] is as shown in the Figure 8. Here the delay of the proposed architecture is 14 % (for 8-order filter) and 64.7% (for 140-order filter) less delay in comparison of LUT-less architecture [24, 25].

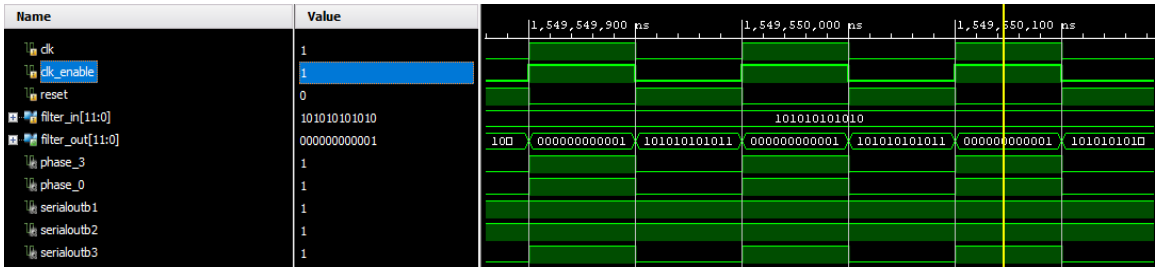


Figure 5. Performance evaluation with simulation of DA-FIR Filter with phase, and serial outputs

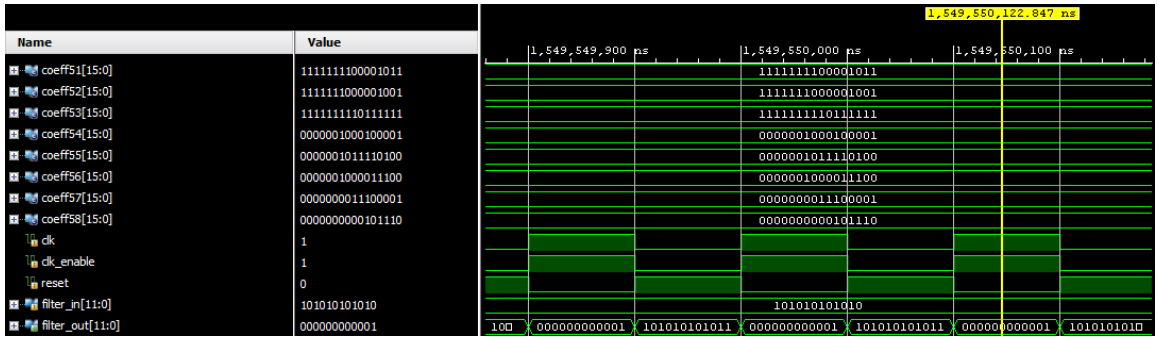


Figure 6. Performance evaluation with simulation of DA-FIR Filter with filter coefficients

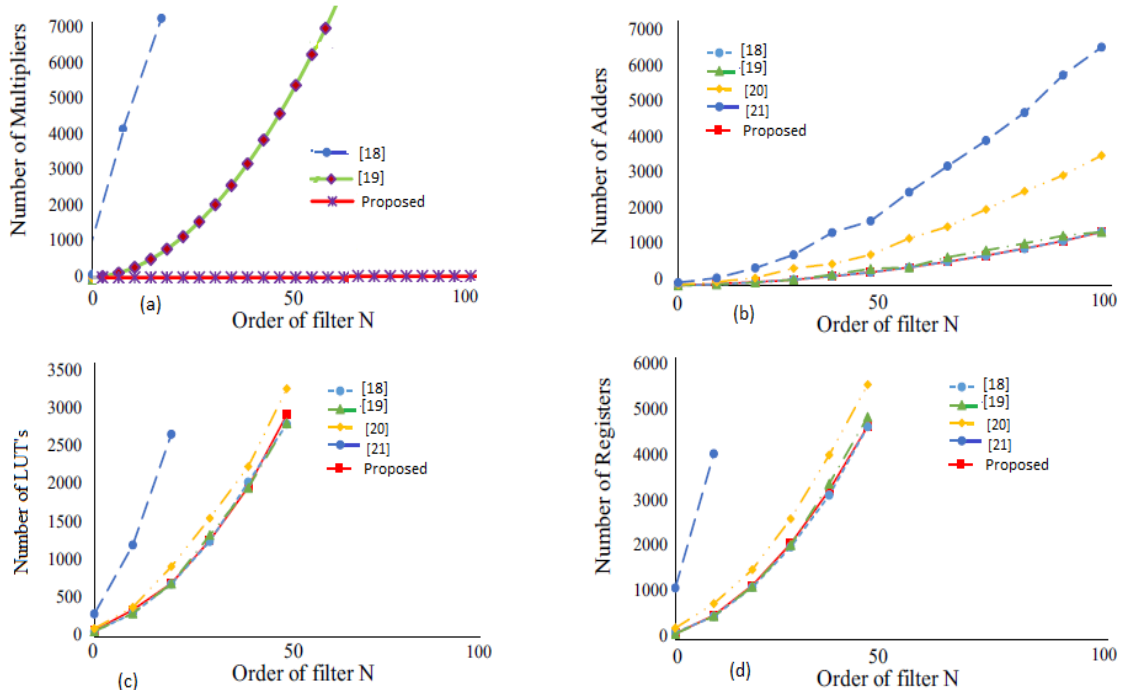


Figure 7. Comparisons of previous research with proposed research with filter order versus (a) Number of multipliers, (b) Number of adders, (c) Number of LUT's, (d) Number of registers

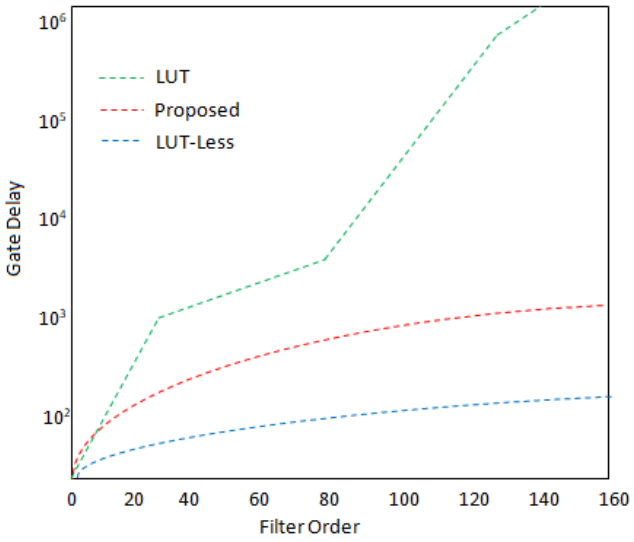


Figure 8. Estimated gate delay of FIR Filters for DA based architectures

The Table 4 illustrates the comparison of the obtained synthesis results of the proposed research work with the previously published literature. Here the factor B is denoted as bit width of the filter coefficients. Further extending this concept in the following mathematical formulae compares the hardware and time complexities of our proposed DA-FIR structures with other filter structures.

Here if: B: = bit width of filter coefficients,
M: = image width,
T_M: = delay of multiplier,
TPA: =delay of parallel adder)
Then: $T_1 = T_{PA} + 2.T_{FA} (\log_2 N - 1)$,
$T_{DAHLUT} = 2T_{PA}$,
$T_{PU} = T_{MUX} + T_{SAB}$,
$T_{SAB} = T_{FA} + T_{XOR} + T_D$

Table 4. Synthesis results comparison of proposed structure with past works

Design	Minimum Sampling Period(ns)	Area(μm^2)	Power (mw)	Area Delay Product (μm^2 ns)	Energy per output	Throughput (MHz)
[26]	B = 8, 11.79	1,720,962.1471	50.0106	2,537,558	312.56	678.19
	B = 16, 13.3	12,661,783.52	181.165	10,550,431	566.14	1200
[27]	B = 8, 13.01	356,293.0216	9.3807	4,636,084	1319.07	76.80
	B = 16, 14.65	1,154,123.0880	26.9941	16,907,903	312.56	68.25
[28]	B = 3, 2.11	98,266.83	2.47	207,343.01	1235	
	B = 7, 2.34	412,267.34	8.91	964,704.78	4555	-
	B= 15, 2.41	1,734,743.62	36.48	4,180,732.12	18,240	
This Work	B = 8, 6.73	651,615.4709	20.1069	531,880	175.66	1325.11
	B = 16, 6.92	4,936,081.6759	102.3432	2,131,770	319.82	2325.48

The T_{MUX}, T_{FA}, T_{XOR} and T_D are the delay of MUX, full adder, XOR gate and D flip-flop, respectively. This comparison of time complexities and hardware of proposed DA-FIR designs with other filter designs is as depicted in Table 5. The Table 4 illustrates our best solution and compares the obtained parameters of our synthesis results with previous works in terms of numerical values of (MSP)-Minimum Sampling Period(ns), Area(μm^2), Power(mw) ,(ADP)-Area Delay Product(μm^2 ns), Energy per output, Throughput(MHz) [29, 30]. Further the Table 5 compares the obtained results in our work with previous works with numerically addressing with mathematical formulas of various parameters such as Throughput, multipliers, adders and registers [31, 32]. The implementation of multi-core computing system is done on the ZYNQ platform with the use of VERILOG language to program and compile the framework [33].

Table 5. Time complexities and hardware of proposed DA-FIR designs with other designs

Design	Throughput	Multipliers	Adders	Registers
[34]	$L/(T_M + T_I)$	M^2	M^2-1	$(M+N)(M-1)$
[35]	$1/(T_M+2T_{PA})$	MN^2	$L([NxN]-1)$	$(M+N)(M-1)$
This Work	$N/(3T_{PA} + 2.B.T_{PU})$	-	$N(7(L+N)-26)+L((3M^2/5)-1)$	$N(M-1)+N.Lc$

5. CONCLUSIONS

The VHDL is used implement the proposed DA finite implse response filter and the design is verified using simulation. The calculated theoretical values of the design match with obtained practical values in the real time simulation environment. Two optimization algorithms are proposed and the resulting optimizations are incorporated into LUT layer and architecture extractions of designed block. The proposed work offers an optimized design in the form of average reductions of number of LUT, reduction in populated slices and reduction in the number of gates for DA-finite impulse response filter implementation. This research paves a way for bio inspired computing architecture with reconfigurable computing strategies designed to avoid computationally intensive operations, achieving the desired specifications with respect to flexibility, timing, and performance.

REFERENCES

[1] C. S. Vinitha and R. K. Sharma, "New approach to low-area, low-latency memory-based systolic architecture for FIR filters," *Journal of Information and Optimization Sciences*, vol. 40, no. 2, pp. 247-262, 2019.

[2] A. Agarwal and L. Bopanna, "Low Latency Area-Efficient Distributed Arithmetic Based Multi-Rate Filter Architecture for SDR Receivers," *Journal of Circuits Systems and Computers*, vol. 27, no. 8, pp. 1-21, 2018.

[3] T. Xu, et al., "Efficient real-time digital subcarrier cross-connect based on distributed arithmetic DSP algorithm," in *Journal of Lightwave Technology*, vol. 38, no. 13, pp. 3495-3505, 2020.

[4] D. Datta, et al., "FPGA implementation of high performance digital down converter for software defined radio," *Microsystem Technologies*, 2019.

[5] B. K. Mohanty and Pramod K. M., "An Efficient Parallel DA-Based Fixed-Width Design Approx Inner-Product Comp," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 28, no. 5, pp. 1221-1229, 2020.

[6] P. Kumar, et al., "ASIC implementation of area efficient highthroughput 2-D IIR filter using distributed arithmetic," *Circuits Syst. Signal Process*, vol. 37, no. 7, pp. 2934-2957, 2018.

[7] G. N. Jyothi, et al., "ASIC implementation of distributed arithmetic based FIR filter using RNS for high speed DSP systems," *International Journal of Speech Technology*, vol. 23, pp. 259-264, 2020.

[8] G. N. Jyothi and S. Sridevi, "High speed and low area decision feed-back equalizer with novel memory less distributed arithmetic filter," *Multimedia Tools and Applications*, vol. 78, no. 23, pp. 79-93, 2019.

[9] P. V. P. Sundar, et al., "Low power area efficient adaptive FIR filter for hearing aids using distributed arithmetic architecture," *International Journal of Speech Technology*, vol. 2, pp. 1-10, 2020.

[10] M. R. Ahmed and B. K. Sujatha, "A review on methods, issues and challenges in neuromorphic engineering," *2015 International Conference on Communications and Signal Processing (ICCSP)*, 2015, pp. 0899-0903.

[11] K. Vijetha and B. R. Naik, "High performance area efficient DA based FIR filter for concurrent decision feedback equalizer," *International Journal of Speech Technology*, vol. 3, pp. 1-7, 2020.

[12] X. Lou, et al., "Lower bound analysis perturbation critical path for area-time efficient multiple constant multiplications," *IEEE Trans.Comput. Aided Des. Integr. Circuits Syst*, vol. 36, no. 2, pp. 313-324, 2016.

[13] M. D'Arco, et al., "Digital Circuit for Seamless Resampling ADC Output Streams," *Sensors*, vol. 20, no. 6, p. 1619, 2020.

[14] M. R. Ahmed and B. K. Sujatha, "A review of reinforcement learning in neuromorphic VLSI chips using computational cognitive neuroscience," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 8, pp. 3315-3320, 2013.

[15] L. Cao, et al., "Hardware-efficient implementation of digital FIR filter using fast first-order moment algorithm," *MIPPR2017: Parallel Process of Images,Optimization Techniques; and Medical Imaging*, vol. 10610, 2018.

[16] H. Yao, et al., "Experimental demonstration of 4-PAM for high-speed indoor free-space OW communication based on cascade FIR-LMS adaptive equalizer," *Optics Communications*, vol. 426, pp. 490-496, 2018.

[17] E. Chitra, et al., "Analysis and implementation of high performance reconfigurable FIR filter using distributed arithmetic," *Wireless Personal Communications*, vol. 102, no. 4, 2018.

[18] H. Jiang, et al., "A review, classification, and comparative evaluation of approximate arithmetic circuits," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 13, no. 4, 2017.

[19] G. Sanchez, et al., "A highly scalable parallel spike-based digital neuromorphic architecture for high-order fir filters using LMS adaptive algorithm," *Neurocomputing*, vol. 330, 2019.

[20] B. K. Mohanty, et al., "LUT Optimization for D.A-Based Block Least Mean Square Adaptive Filter," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 24, no. 5, pp. 1926-1935, 2016.

[21] M. Sharma and S. K. Singh, "New Technique Based Peasant Multiplication for Effcient Signal Processing Applications," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 8, no. 3, pp. 726-729, 2017.

[22] M. T. Khan and R. A. Shaik, "Optimal complexity architectures for pipelined D.A-based LMS adaptive filter," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 2, pp. 630-642, 2018.
[23] H. Jiang, et al., "A high-performance and energy-efficient FIR adaptive filter using approximate D.A circuits," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 1, pp. 313-326, 2018.
[24] M. Sumalatha, et al., "Low power and low area VLSI implementation of vedic design FIR filter for ECG signal de-noising," *Microprocessors and Microsystems*, vol. 71, pp. 1-13, 2019.
[25] R. K. Sarma, et al., "A novel time-shared and lut-less pipelined architecture for lms adaptive filter," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 28, no. 1, pp. 188-197, 2019.
[26] B. U. V. Prashanth and M. R. Ahmed, "Design and Performance Analysis of Artificial Neural Network Based Artificial Synapse for Bio-inspired Computing," *International Conference on Computational Vision and Bio Inspired Computing*, Springer, Cham, 2019, pp. 1294-1302.
[27] B. U. V. Prashanth and M. R. Ahmed, "FPGA Implementation of Bio-inspired Computing Based D.L Model," *Advances in Distributed Computing and Machine Learning*. Springer, Singapore, pp. 237-245, 2020.
[28] M. Z. Alom, et al., "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, 2019.
[29] A. Mahabub, "Design and implementation of cost-effective simple FIR filter for EEG signal on FPGA," *World Scientific News*, vol. 125, pp. 1-17, 2019.
[30] A. Mahabub, "Design and Implementation of a Novel Complete Filter for EEG Application on FPGA," *International Journal of Image, Graphics and Signal Processing*, vol. 10, no. 6, pp. 22-30, 2018.
[31] B. Khurshid and R. N. Mir, "An efficient FIR filter structure based on technology-optimized multiply-adder unit targeting LUT-based FPGAs," *Circuits, Systems, and Signal Processing*, vol. 36, no. 2, pp. 600-639, 2017.
[32] K. S. Reddy and H. Suresh, "A Low-Power VLSI Implementation of RFIR Filter Design using Radix-2 Algorithm with LCSLA," *IETE Journal of Research*, pp. 1-10, 2019.
[33] B. Srikanth, et al., "The enhancement of security measures in advanced encryption standard using double precision floating point multiplication model," *Wiley-Transactions on Emerging Telecommunications Technologies*, pp. 1-13, 2020.
[34] W. Zhao, et al., "A division-free and variable-regularized LMS-based generalized sidelobe canceller for adaptive beamforming and its efficient hardware realization," *IEEE Access*, vol. 6 , pp. 64470-64485, 2018.
[35] S. Dixit and D. Nagaria, "LMS Adaptive Filters for Noise Cancellation: A Review," *International Journal of Electrical and Computer Engineering*, vol. 7, no. 5, pp. 2520-2529, 2017.