DOI: 10.21917/ijme.2023.0248

FIR FILTER DESIGN FOR FPGA AND ASIC IMPLEMENTATION USING SIMULATED ANNEALING

Y.J. Pavitra[1], S. Jamuna[2] and J. Manikandan[3]

[1,3]Department of Electronics and Communication Engineering, PES University, India

[2]Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, India

Abstract

Finite Impulse Response (FIR) filters are widely used in various fields of engineering like image processing, signal processing, communication, consumer electronics etc. Power, area, and delay are primary performance parameters and design of optimized FIR filters meeting these requirements is desired in all fields of application. Simulated annealing algorithm is used for solving discrete optimization problems. In this paper, discrete FIR filters are designed using simulated annealing for FPGA and ASIC implementation. The proposed FIR filters are designed using 2-input logic gates and universal logic module 2:1 multiplexer. The proposed FIR filters designed using simulated annealing are compared with FIR filters designed using Wallace tree multipliers and Dadda multipliers. Experiments reveal that proposed FIR filter with logic gates are suitable for both FPGA and ASIC implementations and MUX based FIR filters are promising for FPGA implementation. It is observed that proposed FIR filters saved maximum of 11.11% of area and 33.45% of power over conventional designs.

Keywords:

Discrete FIR Filter, Simulated Annealing, Optimization, Performance

1. INTRODUCTION

FIR filters are widely used in applications like image processing, signal processing, communication, instrumentation, consumer electronics etc. Reducing the implementation complexity of FIR filters on FPGA has gained significant importance in recent years [1]. A typical digital filtering application reads the input samples from an analog to digital converter, performs the prescribed mathematical operations on the required filter type and outputs the processed result onto digital to analog converter. There are many filter types among which low pass filter, high pass filter, band pass filter, band stop filter are the most used. The drawback of the conventional FIR filter architectures is that it contains many multiplication operations, which consumes a large computation time and leads to large area and increased power consumption for real time applications [2]. Optimized FIR filter architectures with reduced area, power and delay have gained importance in the recent research work [3], [4]. The hardware cost of the FIR architecture can be reduced by encoding constant coefficients in canonical digit representation [5], [6]. The proposed method can be applied in an architecture where a set of constant multipliers are used to multiply a common variable which is a transposed direct form FIR filter as in [2].

Multipliers are one of the important digital functional blocks that contributes for the evaluation of FIR filters. Traditional methods used for the design of functional blocks requires design expertise and time consuming. Metaheuristics are alternatives that produce competitive results when compared to traditional methods like Karnaugh maps and Quine-McCluskey algorithm [7]. Metaheuristic algorithms has been widely used for discrete designs because of their ability in converging to the global optima [8]. Simulated annealing is a metaheuristic algorithm that avoids a candidate being trapped in local minima by using random selection of sample and a probability parameter [9]. Simulated annealing is used to find optimal coefficients for filter design in [10], particle swarm optimization and simulated annealing is used for the design of two-dimensional recursive digital filters in [11], adaptive simulated annealing for digital IIR filter design was carried out in [12], review on application of evolutionary techniques for the design of digital FIR filters is proposed in [13]. Literature confirms that application of metaheuristics for filter design was carried out specific to the filter type or coefficient extraction. In this paper, an attempt is made to design FIR filters using simulated annealing for FPGA and ASIC implementation for various size of the sample and order of the filter suitable for all filter types.

2. FINITE IMPULSE RESPONSE FILTER

In signal processing, a finite impulse response (FIR) filter is one whose impulse response is of finite duration due to the property that it settles down to zero within finite time [14]. The input-output relationship for an FIR filter is defined by the convolution operation given as

$$y(n) = \sum_{k=0}^{N-1} b_k x(n-k) \tag{1}$$

where $x(n)$ and $y(n)$ represent the input and output of an N length FIR filter respectively and b_k represents the filter coefficients. The conventional building block of an FIR filter is as shown in Fig.1. It can be observed from Fig.1 that multipliers and adders form the important functional blocks for FIR filter design and the proposed work focuses on designing FIR filters using simulated annealing.

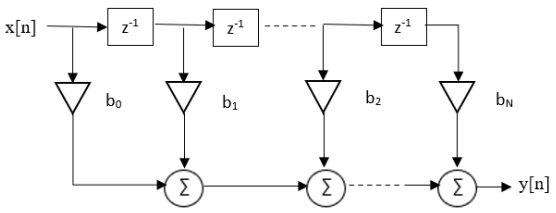

Fig.1. Block diagram of an FIR filter

3. SIMULATED ANNEALING

Metaheuristics are classified as solo search algorithms that starts search process with single initial sample (simulated annealing, tabu search, hill climbing etc.) and population-based search algorithms with several initial samples in the sample space (population based simulated annealing, genetic algorithms, ant

colony optimization, particle swarm optimization etc). Population based searches have proven to perform well over solo search methods for several real-world problems [15] [16]. Initial samples for population-based metaheuristics can be either randomly seeded or can have a feasible solution generated from domain specialist for circuit optimization. In this paper, population based simulated annealing with randomly seeded initial population is used to realize and optimize FIR filters.

Annealing is a process in metallurgy where metals are slowly cooled to make them reach a state of low energy where they are very strong. Simulated annealing is an analogous method for optimization, typically described in terms of thermodynamics. Simulated annealing is a process where the temperature is reduced slowly, starting from a random search at high temperature eventually approaching zero. The random movement corresponds to high temperature and there is little randomness at low temperature. The randomness should tend to jump out of local minima and find regions that have a low heuristic value.

Population based SA commences with the initialization of m initial samples, given as

$$x = \{x_1, x_2,..,x_m\} \quad (2)$$

where x_i is a circuit of fixed length with r predefined resources. Illustration of a sample x_i, with sample size four is shown in Fig.2. In Fig.2, first row lists the type of resource used and second row lists the probable inputs (logic 1(1), logic 0(0), primary inputs) and output(O) from these resources. The resources used for the circuit realization can also have outputs from any previous stage (O^I) as their inputs. This is illustrated in Fig.2 where an OR gate has primary input 'b' and output from AND gate as its other input. Similarly, XOR gate has logic 0 and primary input 'a' as its input.

Type	AND			OR			XOR			XNOR		
I/O	a	b	O	b	O^I	O	0	a	O	a	1	O

Fig.2. Representation of randomly generated initial sample before Perturbation

SA requires an annealing schedule, which specifies how the temperature is reduced as the search progresses. For the proposed work, each x_i is assigned with an initial temperature of $T_i = 20$ and is cooled by a factor of 4 at the end of present generations.

Simulated annealing comprises of two stages: perturbation and evaluation. Perturbation is analogous to mutation in genetic algorithm. The proposed work uses standard point mutation operator. The resource type and/or inputs are chosen for perturbation using the probability function. Illustration of perturbed sample x_i for sample given in Fig.2 is shown in Fig.3 with the perturbed resource type or input highlighted in yellow colour. It can be noticed from Fig.3 that second input field of NOT gate is not considered as it has only one input.

Type	AND			OR			NOT			XNOR		
I/O	a	1	O	b	O^I	O	0	-	O	b	1	O

Fig.3. Representation of perturbed sample

The perturbed population obtained for initial population X is given as:

$$X_p = \{x_{1p}, x_{2p},\ldots x_{mp}\} \quad (3)$$

The evaluation stage of simulated annealing includes the computation of fitness score (FS) for every individual sample x_i in X and x_{ip} in X_p using:

$$FS = 2^n - \sum_{i=0}^{2^n-1} \left(Z_i \oplus Y_i\right) \quad (4)$$

For every output and n denotes the number of primary inputs representing the circuit, Z_i is actual output generated by circuit designed using SA and Y_i is the corresponding output from truth table. FS is equal to 2^n if the output of the circuit matches the truth table output completely. Let FS_{old} and FS_{new} represent the fitness score obtained for X and X_p respectively. The sample x_i in initial population is replaced with perturbed sample x_{ip} if $FS_{new} > \mathrm{FS_{old}}$. For samples with $FS_{new} \leq FS_{old,}$ the condition to replace initial sample with perturbed sample is given as:

$$e^{(FS_{new}-FS_{old})/T} \geq s \quad (5)$$

where T is the present temperature of the sample and s is a random value generated between 0 and 1.

The above-mentioned process of perturbation and evaluation of fitness score is repeated for G_i iterations. At the end of G_i iterations, the temperature of sample with best score is reset to 0. The cooling process is initiated for rest of the samples and their temperature is reduced by applying a cooling factor of 4. The process is repeated for maximum preset G generations. Once the fitness score $FS=2^n$ is matched, the algorithm proceeds to satisfy the objective function by removing redundant or unused resources that do not contribute for circuit design. The new design which matches the fitness score is compared with old design to check if it is better than the previous circuit. If so, the newer solution replaces the older. The best circuit obtained at the end of G generations is an optimal circuit of interest.

The objective function considered for the proposed work is given as:

$$\text{Minimise } F(X) \text{ s.t. } FS = 2^n * q \quad (6)$$

where q is number of outputs.

4. EXPERIMENTAL RESULTS

The efficiency of FIR filter depends on the design of multiplier and adder blocks. Multipliers have complex structures compared to that of adders. Design of adder and multiplier blocks is carried out using proposed simulated annealing algorithm. Two input logic gates are used for the design of proposed multipliers and adders. Multiplexers (MUX) are universal logic modules and forms the fundamental building blocks on FPGA. Repeated use of same element reduces the manufacturing cost and is a primary criterion in VLSI design [17]. Hence, design of multipliers and adders is carried out using 2:1 MUX using proposed simulated annealing algorithm.

Wallace tree multipliers and Dadda multipliers are the most used multipliers in various fields of engineering [18], [19]. A Wallace tree multiplier is a hardware implementation of a binary multiplier using full adders and half adders. An efficient high speed Wallace tree multiplier architecture has the advantage of reduced latency. Dadda multipliers are also binary multipliers that use different reduction techniques to minimize the number of gates and make it faster. To evaluate the efficiency of proposed work, FIR filters designed using proposed simulated annealing

algorithm is compared with the most used FIR filters designed with Wallace tree multipliers and Dadda multipliers.

The proposed work is used to design FIR filters for varying order and size of the input sample or filter coefficients for the evaluation of FIR filter design using simulated annealing. Higher order filters can roll off gain after the bandwidth at a sharper rate than lower order filters. As the order of the filter increases, the actual stopband response of the filter approaches its ideal stopband characteristics. To evaluate the efficiency of proposed FIR filter design using simulated annealing with the FIR filters designed with most commonly used Wallace tree multipliers and Dadda multipliers, FIR filters for various orders are designed and the details of the resource utilization is reported in Table 1 and Table 2 for FPGA and ASIC implementation respectively.

Artix-7 family of FPGAs has redefined cost-sensitive solutions by reducing power consumption and provides advanced functionality for high-performance applications. They are built on 28nm low power process making them suitable for various high-performance applications. xc7a35tcpg236 with speed grade of -1 and clock frequency of 100MHz is used to implement the designs for various orders of FIR filters reported in Table.1. It can be observed from Table.1 that the proposed designs (PD1_gates, PD2_mux) are in par with the existing designs irrespective of the order of the FIR filter. It can also be noted from Table 1 that FIR filters with Dadda multipliers consumes more LUTs compared to other FIR filters for FPGA implementation. Also, it can be noted from Table.1 that with just one additional flip-flop, proposed MUX based FIR filter designs (PD2_mux) has same resource utilization and power consumption as other conventional FIR filters making it suitable for FPGA implementation by reducing manufacturing cost.

Table.1. Resource Utilization of FIR filters with varying order of filter for FPGA implementation

Type of filter	Order of the filter	# LUTs	# Flip-Flops	Power consumption (mw)
FIR_Wallace	4	17	27	93
FIR_Dadda		20	27	93
PD1_gates		17	27	93
PD2_mux		17	28	93
FIR_Wallace	8	37	59	95
FIR_Dadda		41	60	95
PD1_gates		37	59	95
PD2_mux		37	59	95
FIR_Wallace	16	86	123	97
FIR_Dadda		95	123	97
PD1_gates		86	123	97
PD2_mux		86	124	97
FIR_Wallace	32	178	251	99
FIR_Dadda		195	251	99
PD1_gates		178	251	99
PD2_mux		178	252	99

Table.2. Resource Utilization of FIR filters with varying order of filter for ASIC implementation

Type of filter	Order of the filter	Area (μm^2)	Power (ηw)	CPD (ps)
FIR_Wallace	4	1565.798	17611.521	761
FIR_Dadda		1399.910	15244.563	730
PD1_gates		1391.616	12321.019	810
PD2_mux		4716.749	53313.517	1679
FIR_Wallace	8	3009.946	36059.474	768
FIR_Dadda		2859.725	31754.025	731
PD1_gates		2816.410	31648.349	873
PD2_mux		7960.781	92597.423	1865
FIR_Wallace	16	6019.891	90889.472	830
FIR_Dadda		5851.238	80944.660	788
PD1_gates		5731.430	64046.406	921
PD2_mux		9505.382	135794.809	1789
FIR_Wallace	32	11958.681	175634.400	899
FIR_Dadda		11786.342	169177.505	876
PD1_gates		11537.510	143658.781	1066
PD2_mux		15890.227	234818.128	1924

Power, area, and delay are the parameters defining the efficiency of any design [20]-[23]. Hence, FIR filters were implemented on Cadence virtuoso with 90nm standard cell library to evaluate the performance of FIR filter designs for ASIC implementation. It can be observed from Table 2 that the proposed FIR filter, designed using logic gates (PD1_gates) have reduced area and power over conventional designs. The critical path delay (CPD) of the proposed design is slightly high over conventional designs because of gate level implementation. Conventional designs are implemented using functional blocks like half adders and full adders. The proposed gate level implementation has provided the flexibility of choosing the resources to generate design alternatives for FIR filter. The proposed design PD1_gates have saved a maximum of 11.11% of area and 30.03% of power over conventional designs.

Filter coefficients are represented by a finite number of bits, commonly in the range of 8 to 16 bits [21] and the designer must ensure that the error introduced by finite precision would not violate filter specification. Hence, the design of FIR filters for varying size of sampled data and filter co-efficient is reported in Table.3 and Table.4 for FPGA and ASIC implementation using similar experimental setup as mentioned above.

Table.3. Resource Utilization of FIR filters with varying sample size and filter coefficients for FPGA implementation

Type of filter	Order of the filter	# LUTs	# Flip-Flops	Power consumption (mw)
FIR_Wallace	4	17	27	93
FIR_Dadda		20	27	93
PD1_gates		17	27	93

PD2_mux		17	28	93
FIR_Wallace	8	45	48	109
FIR_Dadda		40	51	111
PD1_gates		36	49	110
PD2_mux		45	48	109
FIR_Wallace	16	115	79	134
FIR_Dadda		91	80	135
PD1_gates		105	81	134
PD2_mux		112	79	135
FIR_Wallace	32	203	126	195
FIR_Dadda		197	124	192
PD1_gates		200	127	195
PD2_mux		200	127	195

From Table.3 it can be observed that proposed designs are suitable for FPGA implementations. However, it can be noticed that FIR filters designed with Dadda multipliers performed better in terms of resource utilization as the size of the input sample or filter coefficients increased. From Table.4 it can be observed that proposed FIR filters with gates are area and power efficient by saving a maximum of 11.11% area and 33.45% of power over conventional designs with slight increase in the delay making it suitable for both FPGA and ASIC implementation. However, the proposed FIR filters designed using MUX consume larger resources over conventional design for ASIC implementation with an advantage of use of single type of resource to realize FIR filter reducing implementation cost.

Details of percentage of resources saved for ASIC implementation of proposed design with gates over conventional design is reported in Table 5. It can be noted from Table 5 that PD1_gates saved area between 3.5-11.11% and power between 9.64-30.03% for varying sample/filter coefficient size and order of the filter respectively.

Table.4. Resource Utilization of FIR filters with varying sample size and filter coefficients for ASIC implementation

Type of filter	Sample/ Filter coefficient size	Area (μm^2)	Power (ηw)	CPD (ps)
FIR_Wallace	4	1565.798	17611.521	761
FIR_Dadda		1399.910	15244.563	730
PD1_gates		1391.616	12321.019	810
PD2_mux		4716.749	53313.517	1679
FIR_Wallace	8	3045.824	54429.420	1481
FIR_Dadda		2967.552	40612.328	1245
PD1_gates		2856.030	37424.996	2006
PD2_mux		14140.109	155847.947	3009
FIR_Wallace	16	5952.614	123234.416	2724
FIR_Dadda		5708.656	95630.116	2050
PD1_gates		5580.288	82003.429	3266
PD2_mux		37119.283	481610.890	5190
FIR_Wallace	32	13479.322	223153.750	5816
FIR_Dadda		12488.602	213635.070	5629
PD1_gates		11991.859	201636.210	5852
PD2_mux		60747.264	1084002.640	12397

Table.5. Percentage resource saved for PD1_gates over conventional designs for ASIC implementation

Sample/Filter Coefficient Size	% Area Saved for PD1_gates over		% Power Saved for PD1_gates over	
	Wallace	Dadda	Wallace	Dadda
4	11.11	0.05	30.03	19.17
8	6.4	1.5	11.93	0.33
16	4.7	2.05	29.53	20.87
32	3.5	2.10	18.20	15.08
Order of Filter	**% Area Saved for PD1_gates over**		**% Power Saved for PD1_gates over**	
	Wallace	**Dadda**	**Wallace**	**Dadda**
4	11.11	0.05	30.03	19.17
8	6.20	3.74	25.38	7.84
16	6.25	2.24	33.45	14.24
32	11.03	3.97	9.64	5.61

5. CONCLUSION

Design of FIR filters were carried out using simulated annealing algorithm for varying order of the filter and varying sample or filter coefficients size. Two designs were proposed-FIR filters using 2-input logic gates and universal logic module 2:1 MUX for both FPGA and ASIC implementation. Experiments carried revealed that PD1_gates were suitable for both FPGA and ASIC implementation saving a maximum of 11.11% area and 33.45% of power over conventional designs whereas PD2_mux are suitable for FPGA implementation and can save implementation cost. Experiments carried confirm that metaheuristics can generate optimal or near optimal solutions.

REFERENCES

[1] F. Ahmed and Md. F. Islam, "Energy Efficient FIR Filter for Biomedical Application using FPGA", *Computer Reviews Journal*, Vol. 3, pp. 236-244, 2019.

[2] K.K. Parhi, "VLSI Digital Signal Processing Systems: Design and Implementation", Wiley, 1999.

[3] S.M. Cho, P.K. Meher, L.T.N. Trung, H.J. Cho and S.Y. Park, "Design of Very High-speed Pipeline FIR filter through precise Critical path Analysis", *IEEE Access*, Vol. 9, pp. 34722-34735, 2021.

[4] P. Hemamalini and K.V.L. Bhavani, "Realization of FIR Filter Design for Low power Efficient Digital Signal Processing Applications", *IOSR Journal of VLSI and Signal Processing*, Vol. 5, No. 5, pp. 1-8, 2015.

[5] S. Roy and A. Chandra, "A Triangular Common Subexpression Elimination Algorithm with Reduced Logic Operators in FIR Filter", *IEEE Transactions on Circuits and*

Systems II: Express Briefs, Vol. 67, No. 12, pp. 3527-3531, 2020.

[6] S. Bose, Arijit De, and I. Chakrabarti, "Area-Delay-Power Efficient VLSI Architecture of FIR Filter for Processing Seismic Signal", *IEEE Transactions on Circuits and Systems II: Express Briefs*, Vol. 68, No.11, pp. 3451-3455, 2021

[7] L.A. Muller de Souza, J.E. Henriques Da Silva, L.J. Chaves and H.S. Bernardino, "A Benchmark Suite for Designing Combinational Logic Circuits via Metaheuristics", *Applied Soft Computing*, Vol.91, pp. 1-12, 2020.

[8] A.K. Dwivedi, S. Ghosh and N.D. Londhe, "Low Power FIR Filter Design using Modified Multi-Objective Artificial Bee Colony Algorithm", *Journal of Engineering Applications of Artificial Intelligence*, Vol. 55, pp. 58-69, 2016.

[9] X. Yang, "Engineering Optimization- An Introduction with Metaheuristic Applications", Wiley Publications, 2010.

[10] R.S. Chauhan and S.K. Arya, "Determine Optimal Coefficients of IIR Digital Filters using Simulated Annealing", *International Journal of Computer Applications*, Vol. 43, No. 10, pp. 36-40, 2012.

[11] S. Dhabal and P. Venkateswaran, "Two-Dimensional IIR Filter Design using Simulated Annealing based Particle Swarm Optimization", *Journal of Optimization*, Vol. 2014, pp. 1-10, 2014.

[12] S. Chen, R. Istepanian and B.L. Luk, "Digital IIR Filter Design using Adaptive Simulated Annealing", *Journal of Digital signal Processing*, Vol. 11, pp. 241-251, 2001.

[13] A.K. Dwivedi, S. Ghosh and N.D. Londhe, "Review and Analysis of Evolutionary Optimization-based Techniques for FIR Filter Design", *Journal of Circuits, Systems, and Signal Processing*, Vol. 37, pp. 4409-4430, 2018.

[14] S. Roy and A. Chandra, "A Survey of FIR Filter Design Techniques: Low-Complexity, Narrow Transition-band and Variable Bandwidth", *Integration, the VLSI Journal*, Vol. 77, pp. 193-204, 2021.

[15] D.S. Johnson, C.R. Aragon, L.A. McGeoch and C. Schevon, "Optimization by Simulated Annealing: An Experimental Evaluation; Part 1, Graph Partitioning", *Journal of Operations Research*, Vol. 37, No. 6, pp. 865-892, 1989.

[16] A. Prugel-Bennett, "Benefits of a Population: Five Mechanisms that Advantage Population Based Algorithms", *IEEE Transactions on Evolutionary Computation*, Vol. 14, No. 4, pp. 500-517,2010.

[17] C.K. Vijayakumari, P. Mythili, K.J. Rekha and C.V. Anil kumar, "Genetic Algorithm Based Design of Combinational Logic Circuits using Universal Logic Modules", *Proceedings of International Conference on Information and Communication Technologies*, pp. 1246-1253, 2015.

[18] Y.D. Ykuntam, K. Pavani and K. Saladi, "Design and Analysis of High speed Wallace Tree Multiplier using Parallel Prefix Adders for VLSI Circuit Designs", *Proceedings of International Conference on Computing, Communication and Networking Technologies*, pp. 1-6,2020

[19] K.K. Senthil Kumar, S. Yuvaraj and R. Seshasayanan, "High-Performance Wallace Tree Multiplier", *International Journal of Computer Techniques*, Vol. 7, No. 1, pp. 1-8, 2020.

[20] K.C. Pathak, "An Efficient Dadda Multiplier using Approximate Adder", *Proceedings of International Conference on Tencon*, pp.176-181, 2020.

[21] W.K.Jenkins and B.J. Hunsinger, "*Discrete-Time Signal Processing*", Reference Data for Engineers, 2002.

[22] S. Park, D. Shin, K. Koh and S. Ramnan, "A Low Power, High -Linearity Wideband 3.25GS/s Fourth-order Programmable Analog FIR Filter using Split-CDAC Coefficient Multipliers", *IEEE Transactions on Microwave Theory and Techniques*, Vol. 68, No. 4, pp.1576-1590,2020.

[23] V. Kaushik and H Saini, "The Proposed Full-Dadda Multipliers", *International Journal for Innovative Research in Science and Technology*, Vol. 4, No. 11, pp. 104-111, 2018.