FPGA Implementation of Fast Running FIR Filters

S. Rengaprakash,[1] M. Vignesh,[2] N. Syed Anwar,[3] M. Pragadheesh,[4]
E. Senthilkumar,[5] M. Sandhya[6] and J. Manikandan[7]
[1,2,3,4] Department of EEE, Saranathan College of Engineering, Venkateswara Nagar, Trichy 620012, Tamil Nadu, INDIA
[5,6,7] Department of ECE and Crucible of Research and Innovation (CORI), PES University, 100-Feet Ring Road,
BSK Stage III, Bangalore 560085, Karnataka, INDIA
Email: [1]rsoundar555@gmail.com [7]manikandanj@pes.edu

***Abstract*—Digital filter design using Finite Impulse Response (FIR) filters are predominantly used for various applications pertaining to digital signal processing and wireless communication. Fast running FIR filters are designed using the concepts of polyphase decomposition to provide the speed benefit. In this paper, hardware implementation of fast running FIR filter on a Virtex-5 FPGA is proposed and the design is compared with two different techniques: FIR filter design using optimized transpose structure and FIR filter design using Distributed Arithmetic (DA) approach. The fast running FIR filter designed in this paper is two times faster than the usual FIR filter.**

***Index Terms*—Digital filters, Distributed Arithmetic, Fast running filters, FIR, FPGA.**

I. Introduction

Finite Impulse Response (FIR) filter design is an important topic in digital signal processing and wireless communication systems. FIR filters are used for various wireless applications such as software defined radios in [1], satellite receivers in [2], satellite payload filter in [3], pre-processing for wireless sensor networks in [4], for 3G communication systems in [5], for RADAR applications in [6] and many more. FIR filters have been implemented on various hardwares such as microcontroller in [7], DSPs in [8], [9], FPGAs in [10], [11], ASICs in [5] and using SoCs in [12], [13]. FPGAs have come a long way from mere glue-logic operations in [14], as controllers in [15], for system-on-chip based designs in [16], [17] and reconfigurable systems in [18], [19]. This motivated the implementation of proposed filter design onto an FPGA.

FIR filters are designed using various approaches and to list a few, design of FIR filters using Distributed Arithmetic (DA) is reported in [10], using residue arithmetic in [20] and using particle swarm optimization in [21]. A fast-running FIR filter was first proposed in [22] and not much work has been carried out in this area. In this paper, hardware implementation of fast-running FIR filter onto an FPGA device is proposed and the experimental results are reported followed by a comparison between proposed design and FIR filter designed using optimized transpose structure and DA.

II. FIR Filters

In this section, a comparison between mathematical expressions for FIR filter design using optimized transpose structure, distributed arithmetic and fast-running FIR filters are discussed. The impulse response for high-pass, low-pass, band-pass and band-stop filters are given in Table I.

TABLE I
Impulse Response of Various FIR Filters.

Filter type	Impulse response
LPF	$h_d\,(n) = \begin{cases} \frac{\sin[(n-M)\cdot\omega_c]}{\pi\cdot(n-M)}; & n \neq M \\ \frac{\omega_c}{\pi}; & n = M \end{cases}$
HPF	$h_d\,(n) = \begin{cases} -\frac{\sin[(n-M)\cdot\omega_c]}{\pi\cdot(n-M)}; & n \neq M \\ 1 - \frac{\omega_c}{\pi}; & n = M \end{cases}$
BPF	$h_d\,(n) = \begin{cases} \frac{\sin[(n-M)\cdot\omega_{c2}]}{\pi\cdot(n-M)} - \frac{\sin[(n-M)\cdot\omega_{c1}]}{\pi\cdot(n-M)}; & n \neq M \\ \frac{\omega_{c2}-\omega_{c1}}{\pi}; & n = M \end{cases}$
BSF	$h_d\,(n) = \begin{cases} \frac{\sin[(n-M)\cdot\omega_{c1}]}{\pi\cdot(n-M)} - \frac{\sin[(n-M)\cdot\omega_{c2}]}{\pi\cdot(n-M)}; & n \neq M \\ 1 - \frac{\omega_{c2}-\omega_{c1}}{\pi}; & n = M \end{cases}$

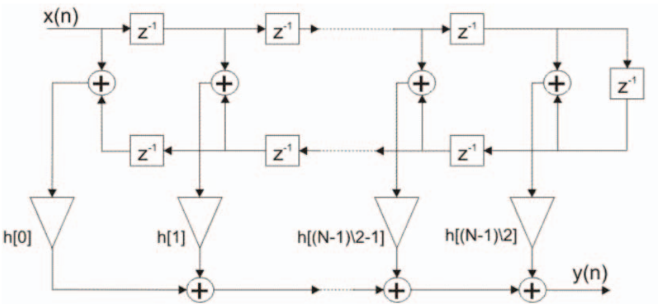


Fig. 1. Optimized transpose structure of basic FIR filter.

A. Basic FIR Filter

The block diagram for FIR filter design using optimized transpose structure is shown in Fig. 1 and the basic FIR filter equation is given as

$$y[n] = \sum_{k=0}^{N-1} h\,[k]\,x[n-k] \tag{1}$$

where N is the FIR filter length, $x[n-k]$ is the $(n-k)$th instance of input data and $h[k]$ is the kth coefficient of the filter, computed as

$$h\,[n] = w\,[n] \cdot h_d[n] \tag{2}$$

where $w\,[n]$ and $h_d[n]$ are the windowing function and impulse response of filter respectively.

B. FIR Filter Using Distributed Arithmetic

Distributed Arithmetic (DA) is an alternative and efficient technique for multiply and accumulate (MAC) operation. DA

978-1-5090-4442-9/17/$31.00 ©2017 IEEE

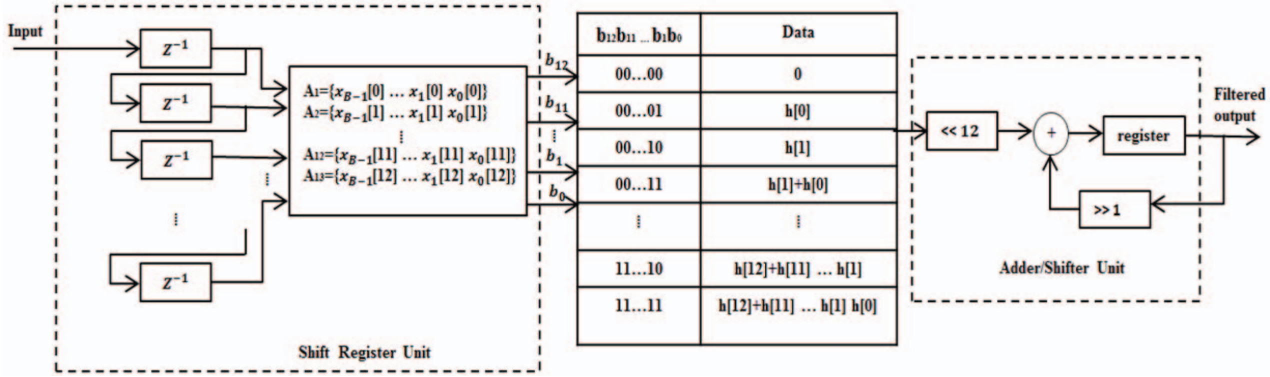


Fig. 2. Distributed arithmetic based FIR filter design.

technique reduces MAC operation by distributing the inputs in bit serial fashion. The FIR filter equation given in (1) is rewritten as

$$\begin{aligned} y\left[n\right] = h\left[0\right]x\left[n\right] + h\left[1\right]x\left[n-1\right] + h\left[2\right]x\left[n-2\right] \\ + \ldots + h\left[k\right]x[n-k] \end{aligned} \tag{3}$$

and the input $x[n]$ at any instance is written as

$$x\left[n\right] = \sum_{b=0}^{B-1} x_b\left[n\right]2^b \tag{4}$$

where $x_b[n] \in [0, 1]$ is the bth bit of input $x[n]$. The FIR filter equation is rewritten as

$$y = \sum_{n=0}^{N-1} h[n]\left[\sum_{b=0}^{B-1} x_b[n]2^b\right]. \tag{5}$$

Rearranging (5) by grouping the sum of the products gives

$$\begin{aligned} y = {} & h\left[0\right]\left(x_{B-1}\left[0\right]2^{B-1} + x_{B-2}\left[0\right]2^{B-2} + \ldots + x_0\left[0\right]2^0\right) \\ & + h\left[1\right]\left(x_{B-1}\left[1\right]2^{B-1} + x_{B-2}\left[1\right]2^{B-2} + \ldots + x_0\left[1\right]2^0\right) \\ & \vdots \\ & + h\left[N-1\right]\left(x_{B-1}\left[N-1\right]2^{B-1} + \cdots + x_0\left[N-1\right]2^0\right) \end{aligned} \tag{6}$$

$$\begin{aligned} y = {} & \left(h\left[0\right]x_{B-1}\left[0\right] + \ldots + h\left[N-1\right]x_{B-1}\left[N-1\right]\right)2^{B-1} \\ & + \left(h\left[0\right]x_{B-2}\left[0\right] + \ldots + h\left[N-1\right]x_{B-2}\left[N-1\right]\right)2^{B-2} \\ & \vdots \\ & + \left(h\left[0\right]x_0\left[0\right] + h\left[1\right]x_0\left[1\right] \ldots + h\left[N-1\right]x_0\left[N-1\right]\right)2^0. \end{aligned} \tag{7}$$

Since the coefficients, are known constants and $x_b\left[n\right] \in [0, 1]$, the $h[n]x_b\left[n\right]$ multiplication in (7) is reduced to an addition of a set of $h[n]$ coefficients, based on the values of $x_b\left[n\right]$. These values are pre-computed for all the possible combinations of input bits and stored in a Look-up Table (LUT).

The block diagram of FIR filter designed using DA with filter order 12 is shown in Fig 2. The shift register unit retrieves bitwise information from every input sample $x[n]$, fed to the filter. The bit wise information obtained from shift register unit acts as LUT address and the data read from LUT are finally passed to the adder/shifter unit.

C. Fast Running FIR Filter

Fast running FIR filter design is an outcome of polyphase decomposition, wherein the input signal $x[n]$ and filter $h[n]$ are decomposed into R polyphase components. The polyphase decomposition of input signal $X(z)$ and filter $H(z)$, with $R = 2$ into even and odd polyphase components are given as

$$X\left(z\right) = \sum_n x\left[n\right]z^{-n} = X_0\left(z^2\right) + z^{-1}X_1\left(z^2\right) \tag{8}$$

$$H\left(z\right) = \sum_n h\left[n\right]z^{-n} = H_0\left(z^2\right) + z^{-1}H_1\left(z^2\right). \tag{9}$$

The basic FIR filter equation in (1) can be rewritten in time domain and z-domain as

$$y\left[n\right] = x\left[n\right] * h\left[n\right] \tag{10}$$

$$Y(z) = X(z)H(z). \tag{11}$$

Substituting (8) and (9) in (11) yields

$$\begin{aligned} Y\left(z\right) &= \left[X_0\left(z^2\right) + z^{-1}X_1\left(z^2\right)\right]\left[H_0\left(z^2\right) + z^{-1}H_1\left(z^2\right)\right] \\ &= X_0\left(z^2\right)H_0\left(z^2\right) + X_0\left(z^2\right)z^{-1}H_1\left(z^2\right) \\ &\quad + z^{-1}X_1\left(z^2\right)H_0\left(z^2\right) + z^{-2}X_1\left(z^2\right)H_1\left(z^2\right) \\ &= X_0\left(z^2\right)H_0\left(z^2\right) + z^{-2}X_1\left(z^2\right)H_1\left(z^2\right) \\ &\quad + z^{-1}X_1\left(z^2\right)H_0\left(z^2\right) + z^{-1}X_0\left(z^2\right)H_1\left(z^2\right) \\ &= Y_0\left(z^2\right) + z^{-1}Y_1\left(z^2\right) \end{aligned} \tag{12}$$

where $Y_0\left(z\right)$ and $Y_1\left(z\right)$ are the polyphase components with

$$Y_0\left(z\right) = X_0\left(z\right)H_0\left(z\right) + z^{-1}X_1\left(z\right)H_1\left(z\right) \tag{13}$$

and

$$Y_1\left(z\right) = X_1\left(z\right)H_0\left(z\right) + X_0\left(z\right)H_1\left(z\right). \tag{14}$$

The architecture for fast-running FIR filter design using above-mentioned equations is shown in Fig. 3.

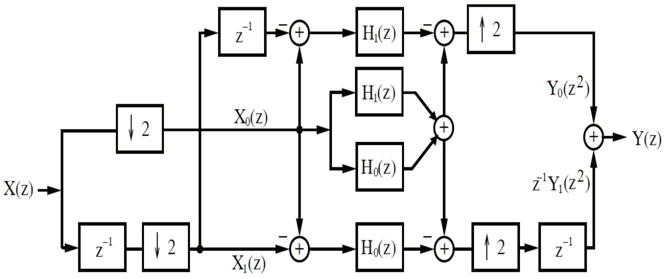
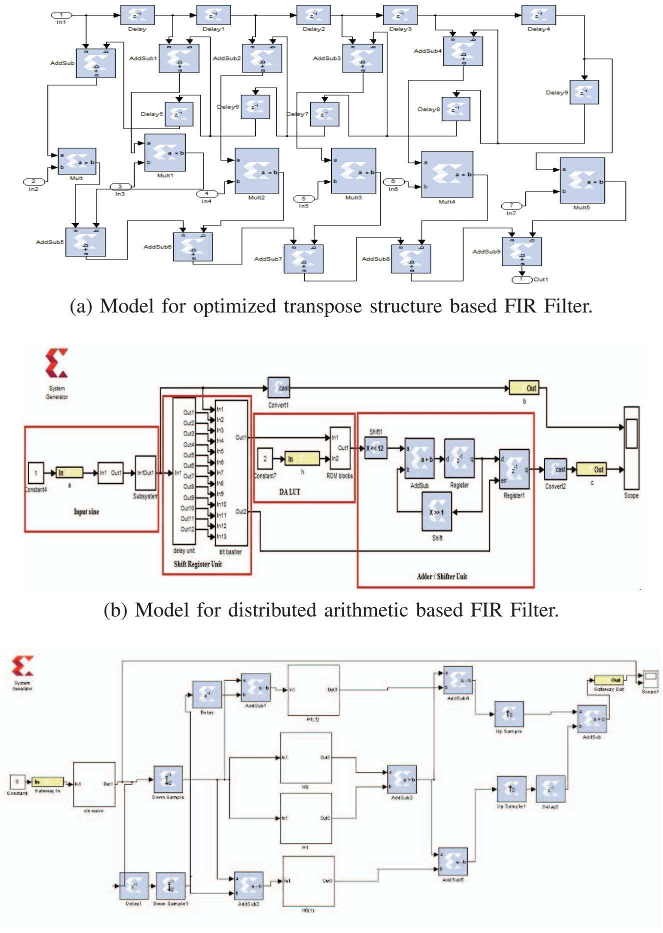


Fig. 3. Fast running FIR filter architecture.



(a) Model for optimized transpose structure based FIR Filter.

(b) Model for distributed arithmetic based FIR Filter.

(c) Model for fast-running fir filter structure.

Fig. 4. FPGA model for FIR filter designs.

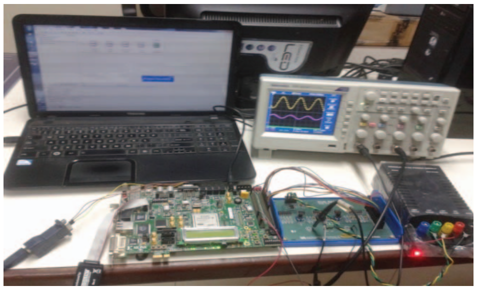


Fig. 5. Hardware setup for evaluating proposed work.

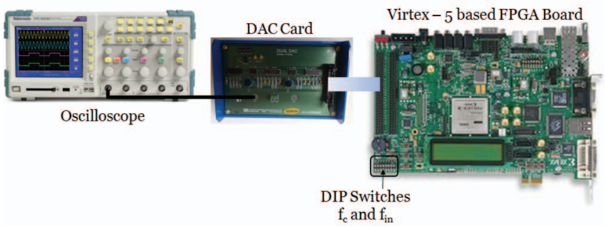


Fig. 6. Block diagram of the hardware setup established.

TABLE II
SIMULATION RESULTS OF FAST RUNNING FIR FILTER.

Filter type →	LPF ($f_c = 2$ kHz)		HPF ($f_c = 3$ kHz)		BPF ($f_{c1} = 2$ kHz, $f_{c2} = 6$ kHz)		BSF ($f_{c1} = 3$ kHz, $f_{c2} = 6$ kHz)	
f_{in} ↓	Sim.	Xil.	Sim.	Xil.	Sim.	Xil.	Sim.	Xil.
1 kHz	4.57	4.51	0.71	0.55	0.82	0.77	4.00	4.13
2 kHz	3.19	3.04	1.65	1.37	2.4	2.42	2.90	2.75
3 kHz	1.38	1.32	2.86	2.75	3.87	3.85	1.62	1.82
4 kHz	0.17	0.17	4.67	4.81	4.26	4.40	0.40	1.65
5 kHz	0.50	0.44	5.50	5.50	3.60	3.58	0.90	2.20
6 kHz	0.44	0.39	5.72	5.50	1.90	2.20	2.30	3.63
7 kHz	0.33	0.28	5.50	5.50	0.97	0.99	4.00	5.50
8 kHz	0.28	0.28	5.33	5.22	0.53	0.49	5.40	6.38
9 kHz	0.50	0.07	5.28	5.28	0.20	0.28	6.30	6.20
10 kHz	0.33	0.39	5.39	5.39	0.88	1.10	6.40	5.90

TABLE III
HARDWARE RESULTS OF FAST RUNNING FIR FILTER.

f_{in} (kHz)	LPF	HPF	BPF	BSF
1	4.56	0.80	1.04	4.40
2	3.20	1.68	2.60	3.60
3	1.52	5.60	4.24	2.00
4	0.48	5.36	4.60	2.30
5	0.72	5.20	4.08	2.30
6	0.80	5.76	2.48	4.10
7	0.64	5.60	1.28	5.52
8	0.64	5.44	1.04	5.60
9	0.48	5.92	0.64	5.52
10	0.72	5.84	1.28	5.44

III. FPGA IMPLEMENTATION OF THE FILTERS

The FIR filter design using various approaches discussed in Section II are implemented on Virtex-5 FPGA based ML507 evaluation board. An FPGA can be programmed using VHDL, Verilog, SystemVerilog, SystemC, High Level Synthesis (HLS) or System Generator blocksets using MATLAB Simulink. Hardware implementation of proposed work is carried out using Xilinx System Generator blocksets for FPGA, which is a model based programming approach. The Xilinx model designed for FIR filter using optimized transpose structure, FIR filter using DA technique and fast running FIR architecture are shown in Fig. 4(a)–(c).

IV. EXPERIMENTAL RESULTS

The hardware setup established and its block diagram for testing proposed work are shown in Figs. 5 and 6 respectively. DIP switches on FPGA board are used to assess the

TABLE IV
Comparison of Hardware Resources utilized by Various FIR Filter Design Approaches.

Filter Type →		LPF			HPF			BPF			BSF		
Technique → Resources ↓	Available	Opt. Tr.	DA	Fast	Opt. Tr.	DA	Fast	Opt. Tr.	DA	Fast	Opt. Tr.	DA	Fast
Slice registers	44800	4147	322	3013	4147	322	2386	8152	322	4571	8352	322	4545
Slice LUTs	44800	4739	1005	2571	4754	1005	3332	9250	1005	5668	9524	1005	5837
Block memory	148	0	98	10	0	98	16	0	98	16	0	98	16
DSP48Es	128	39	2	64	44	2	53	77	2	71	81	2	77
I/Os	640	23	24	21	23	24	21	27	24	21	27	24	21

TABLE V
Power Consumed by Various FIR Filter Designs.

Design → Filter Type ↓	Optimized Transpose	Distributed Arithmetic	Fast Running
LPF	1.444 W	1.635 W	1.432 W
HPF	1.446 W	1.631 W	1.432 W
BPF	1.473 W	1.632 W	1.432 W
BSF	1.464 W	1.637 W	1.432 W

performance of the filter designed by varying the filter cut-off frequencies and input signal frequency.

The simulation results of proposed fast-running FIR filter designed using Simulink blocksets and Xilinx System generator blocksets are reported in Table II for all the four types of filters. The hardware results of fast-running FIR filter implemented on FPGA for the same cut-off frequencies used for simulation are reported in Table III. It is observed from Tables II and III that the hardware results of fast running FIR filter are in par with the simulation results. Satisfactory performance is observed on testing the filter for different cut-off frequencies too.

The hardware resources utilized to design fast-running FIR filters is compared with the resources utilized to design FIR filters using optimized transpose structure and Distributed Arithmetic based approach in Table IV. It is observed that the DA based approach consumes least resources followed by fast running FIR and optimized transpose structure based FIR filter. Details about the power consumption for each design are obtained using Xilinx XPower Analyzer and the results are reported in Table V.

V. Conclusion

Design and implementation of fast running FIR filter on Xilinx Virtex-5 FPGA is reported and its performance is evaluated with optimized transpose structure based FIR filter and Distributed Arithmetic based FIR filter approaches. The proposed filter is two times faster than usual FIR filter. The proposed system can be easily employed for various wireless communication applications for swift processing.

References

[1] F. Harris, "Fixed length FIR filters with continuously variable bandwidth," in *2009 1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology*, Aalborg, 2009, pp. 931–935.

[2] J. J. Patel, K. R. Parmar, and H. N. Mewada, "Design of FIR filter for burst mode demodulator of satellite Receiver," in *2016 International Conference on Communication and Signal Processing (ICCSP)*, Melmaruvathur, 2016, pp. 0686–0690.

[3] A. Goel and A. Gupta, "Design of satellite payload filter emulator using hamming window," in *2014 International Conference on Medical Imaging, m-Health and Emerging Communication Systems (MedCom)*, Greater Noida, 2014, pp. 202–205.

[4] Cheng Xu, Su Yin, Yunchuan Qin, and Hanzheng Zou, "A novel hardware efficient FIR filter for wireless sensor networks," in *2013 Fifth International Conference on Ubiquitous and Future Networks (ICUFN)*, Da Nang, 2013, pp. 197–201.

[5] M. Ojail, S. Chevobbe, R. David, and D. Demigny, "A frequency domain FIR filter implementation method for 3G communication systems," in *2008 The Third International Conference on Digital Telecommunications (icdt 2008)*, Bucharest, 2008, pp. 1–5.

[6] M. Lavanya and A. Kalaiselvi, "High speed FIR adaptive filter for RADAR applications," in *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, Chennai, 2016, pp. 2118–2122.

[7] S. Pujari, A. Yeotkar, V. Shingare, S. Momin, and B. Kokare, "Performance analysis of microcontroller and FPGA based Signal Processing a case study on FIR filter design and implementation," in *2015 International Conference on Industrial Instrumentation and Control (ICIC)*, Pune, 2015, pp. 252–257.

[8] S. Vityazev, A. Kharin, A. Kalinkin, and V. Vityazev, "Parallel form of FIR filter for implementation on multicore DSP," in *2014 3rd Mediterranean Conference on Embedded Computing (MECO)*, Budva, Montenegro, 2014, pp. 177–179.

[9] C. L. Hu, "Design and verification of FIR filter based on Matlab and DSP," in *2012 International Conference on Image Analysis and Signal Processing*, Hangzhou, 2012, pp. 1–4.

[10] P. Longa and A. Miri, "Area-efficient FIR filter design on FPGAs using distributed arithmetic," in *2006 IEEE International Symposium on Signal Processing and Information Technology*, Vancouver, BC, 2006, pp. 248–252.

[11] E. Ozpolat, B. Karakaya, T. Kaya, and A. Gulten, "FPGA-based digital Filter Design for Biomedical Signal," in *2016 XII International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH)*, Lviv, 2016, pp. 70–73.

[12] I. Steiner and G. A. Jullien, "A Fault-Tolerant Complex FIR Filter for SoC Communication Technologies," in *2007 Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, 2007, pp. 2009–2013.

[13] A. T. Erdogan and T. Arslan, "High throughput FIR filter design for low power SoC applications," in *Proceedings of 13th Annual IEEE International ASIC/SOC Conference (Cat. No.00TH8541)*, Arlington, VA, 2000, pp. 374–378.

[14] K. J. Raut and S. S. Shiriramwar, "MP3 Portable Player System-Level Glue Logic on FPGA," in *2007 IEEE International Conference on Microelectronic Systems Education (MSE'07)*, San Diego, CA, 2007, pp. 97–98.

[15] J. Manikandan, M. Jayaraman, and M. Jayachandran, "Design of an FPGA-based electronics flow regulator for spacecraft propulsion system," *Advances in Space Research*, vol. 47, no. 3, pp. 488–495, Feb. 2011.

[16] A. Fridman and S. Semenov, "System-on-Chip FPGA-based GNSS receiver," in *Design & Test Symposium, 2013 East-West*, Rostov-on-Don, 2013, pp. 1–7.

[17] J. Manikandan and B. Venkataramani, “System-on-programmable-chip implementation of diminishing learning based pattern recognition system,” *International Journal of Machine Learning and Cybernetics*, vol. 4, no. 4, pp. 347–363, Aug. 2013.

[18] N. Montealegre, D. Merodio, A. Fernández, and P. Armbruster, “In-flight reconfigurable FPGA-based space systems,” in *Adaptive Hardware and Systems (AHS), 2015 NASA/ESA Conference on*, Montreal, QC, 2015, pp. 1–8.

[19] J. Manikandan, S. Shruthi, S. J. Mangala, and V. K. Agrawal, “Design and implementation of reconfigurable coders for communication systems,” in *Proc. Int. Conf. on VLSI Systems, Architectures, Technology and Applications (VLSI-SATA)*, Bengaluru, India, Jan. 2016, pp. 1–5.

[20] G. Loonawat and R. E. Siferd, “FPGA implementation of a FIR filter using residue arithmetic,” in *Proceedings of the IEEE 1996 National Aerospace and Electronics Conference NAECON 1996*, Dayton, OH, 1996, vol. 1, pp. 286–290.

[21] S. Mukherjee, R. Kar, D. Mandal, S. Mondal, and S. P. Ghoshal, “Linear phase low pass FIR filter design using Improved Particle Swarm Optimization,” in *2011 IEEE Student Conference on Research and Development*, Cyberjaya, 2011, pp. 358–363.

[22] U. Meyer-Baese, *Digital signal processing with field programmable gate arrays*, 2nd ed., Berlin Heidelberg: Springer-Verlag, 2004.