Research Article

Comparative Analysis of Analog and FPGA Realizations Based on Matsuda Method for Fractional Order Integral Operator

Omer Pektas and Murat Koseoglu

***Abstract*— Realization of fractional order (FO) transfer functions is essential for real-time applications such as communication systems, video, and digital signal processing. In general, in both implementation methods, the FO transfer function including FO integral and derivative operators is transformed to an integer order approximate transfer function by one of the approximation methods such as Oustaloup, Matsuda, Continued Fraction Expansion (CFE), Modified Stability Boundary Locus (MSBL), etc. Then, the integer order approximate transfer function can be implemented using analog circuit elements such as opamps, resistors, capacitors, or digitally with field-programmable gate arrays (FPGA). In this study, integer order approximate continuous time transfer function obtained for FO integral operator by Matsuda's approximation method is converted to a discrete time function, and that function is digitally implemented by FPGA with Xilinx System Generator. The results obtained are analyzed in comparison with analog circuit implementation results presented in a former study. The study emphasizes the growing importance of fractional calculus in providing accurate models for real-world systems and the challenges posed by the long memory effect in digital implementations. Simulation and experimental results, including sinusoidal waveform, step response and impulse response analysis, reveal the pros and cons of FPGA implementation. Considering these issues, conclusions are made on the effectiveness, efficiency and potential of the FPGA implementation for real-time applications in control systems and signal processing.**

Index Terms—**Fractional Order (FO) Integral, Matsuda's Method, Field Programmable Gate Array (FPGA), Digital Circuit Implementation.**

I. Introduction

THERE HAS been a growing interest among engineers in fractional calculus due to its potential for providing more accurate representations of real-world systems. Over the past two decades, Fractional Order (FO) modeling techniques have been applied to a wide range of problems in various scientific and engineering disciplines. Researchers have implemented FO system models with the accuracy required for their specific applications as well as possible. In the field of control systems, approximate implementation methods have enabled the simulation and experimental realization of FO controllers. However, a significant challenge in the digital implementation of near-ideal FO elements is the long memory effect, which arises from the non-local nature of FO integrals and derivatives. FO operators inherently involve a backward memory, requiring consideration of all past values of a function when processing current values. This leads to a continuously increasing computational load in terms of memory and processing power when handling continuous data streams. Even for non-ideal experimental realizations of fractional elements, achieving real-time computation may necessitate high-speed digital hardware, such as FPGAs [1].

Analog realizations of FO systems also have some advantages. There are two main approaches for the analog implementation of FO systems: circuit-based implementation and implementation using Field-Programmable Analog Arrays (FPAA). The study conducted by A. Hassanein and his colleagues examines the design process of FO analog filters, focusing on both circuit-based and FPAA methods. The study demonstrates how classical integer-order filters can be generalized to FO elements. Simulation and experimental results were consistent with theoretical analysis, showing that FO systems offer greater flexibility and design freedom [2]. In another study conducted by Nako and colleagues, the design of generalized 1+α order Butterworth filters and their implementation using FPAA were presented. The FO filters were generalized from integer-order filters and made programmable in various types, such as low-pass, high-pass, band-pass, and band-stop filters. The study developed a method using MATLAB to model the filter transfer functions as integer-order and prepared these transfer functions for implementation on an FPAA device. Results showed that the proposed method allowed the filters to be easily implemented, with simulation and experimental findings aligning with theoretical expectations [3]. In addition, another study conducted by M.A. George and colleagues presented the design of complex-order (complex-order) PI/PID speed controllers and their implementation using FPAA. Due to the challenges of implementing complex-order controllers in digital environments, these controllers were made practically

Ömer Pektaş, is with Department of Vocational School of Technical Sciences, Electric and Energy, University of Karamanoglu Mehmetbey University, Karaman, Turkey,(e-mail: omerpektas@kmu.edu.tr).

https://orcid.org/0000-0002-6927-6529

Murat Köseoğlu, is with Department of Electrical & Electronics Engineering University of Inonu, Malatya, Turkey, (e-mail: murat.koseoglu@inonu.edu.tr).

https://orcid.org/0000-0003-3774-1083

Manuscript received Oct 01, 2024; accepted Dec 28, 2024.
DOI: 10.17694/bajece.1577733

implementable using FPAA. Simulation and experimental results demonstrated that complex-order controllers exhibited better stability and speed-tracking performance than traditional technologies [4]. As noted, despite the analog implementations having certain advantages, FO systems, which were transformed to high integer order systems, are difficult to implement due to their complex structures. These systems can be digitally realized using FPGAs, which provide a more efficient, faster, and cost-effective platform for digital system implementation. The study by A. Ali and colleagues is a comprehensive review of this subject. It examines the implementation of FO differentiators and integrators, exploring different digital approaches and software tools for digital system applications. Future research directions are discussed, addressing open problems and proposing potential solutions [5]. Considering the advantages of digital systems and the drawbacks of analog implementations, this study emphasizes the preference for FPGA-based digital realizations for fractional-order operators.

Various computational platforms can be utilized to design or calculate FO systems. For many years, digital signal processors (DSPs) and microprocessors have been widely used in low-speed applications where power and efficiency are not critical concerns. However, due to technological advancements in recent years, FPGAs have gained popularity as platforms for digital signal-processing applications. As known, FPGAs are integrated circuits composed of tens of thousands of programmable logic cells interconnected by programmable switches and wires. The most significant advantage of FPGAs over DSPs and microprocessors is their parallel architecture and versatility. Since an FPGA can be adapted to execute computations in maximum parallelism, it outperforms microprocessors and DSPs, which must execute computations serially [6].

FPGAs offer inherent advantages in terms of performance, flexibility, and efficiency. FPGAs enable parallel processing, which allows for high-speed data handling and low-latency operation, essential for real-time applications such as communication systems, video processing, and digital signal processing. The reconfigurability of FPGAs allows designers to customize and optimize filters for specific applications, ensuring better performance compared to fixed-function hardware [7]. Additionally, the deterministic nature of FPGA operations provides consistent and reliable performance, which is critical for applications requiring precise timing. Overall, implementing filters on FPGAs combines the benefits of high performance, adaptability, and efficiency, making them an ideal choice for complex and demanding signal processing tasks [8].

In a former study [1], the authors present a low-cost analog circuit realization of Matsuda's approximate FO integral operators for industrial electronics using operational amplifiers (op-amps), resistors, and capacitors. The proposed method utilizes Matsuda's approximation and partial fraction expansion to decompose FO integral models into low-pass filter forms, facilitating their implementation with standard electronic components. The designed circuit's performance is validated through experimental comparisons with exact analytical solutions and alternative FO circuit designs, demonstrating satisfactory accuracy in time and frequency responses. The results highlight the practicality and efficiency of the analog realization for FO systems, emphasizing its potential for industrial applications where cost and simplicity are critical. However, in this former study, it was seen that it is difficult to readjust a capacitor or resistor value when a parameter or the degree changed in the transfer function. The increase in the degree of the transfer function may increase the complexity and the sensitivity of the analog circuit which is composed of different analog components having different tolerance values.

Considering the encountered difficulties in analog realization, this study aims to examine the advantages and disadvantages of the FPGA digital realization compared to the analog realization. In this study, an integer 4^{th} order approximate continuous time transfer function, which was obtained by the transform of FO integral operator $s^{-0.5}$ with Matsuda's approximation method in $\omega \in [0.1\ 10] rad/sec$, was considered. This function is decomposed by the partial fraction expansion method and expressed as the sum of first order low-pass filter forms in s domain [1], [9]. The sum of these filter functions can be digitally implemented with FPGA after the conversion of continuous time function to discrete time function. The obtained function is converted to a discrete time function using MATLAB's 'c2d' function for the implementation of the function in the FPGA. The FPGA realization results, obtained with the aid of Xilinx System Generator, are compared with analog realization results obtained in the former study [1]. Simulation results, including the responses for square waveform, sinusoidal waveform, 100 s step waveform and 1 s impulse waveform, demonstrate the effectiveness and efficiency of the FPGA implementation, highlighting its potential for real-time applications in control systems and signal processing. The FPGA digital realization results obtained via Xilinx System Generator were analyzed in comparison with the analog application results obtained using conventional electronic components. Simulations and experimental outcomes, encompassing sine and step response analyses, exhibit the conformity and efficiency of the FPGA implementation. These findings underscore the practicality of FPGAs for real-time control systems and signal processing applications.

II. MATERIALS AND METHODS

In the study, FO integral operator $s^{-0.5}$ was realized digitally by FPGA. Since an FO operator or an FO transfer function can not be realized directly, the FO integral operator $s^{-0.5}$ was transformed to an integer (4^{th}) order approximate transfer function by Matsuda's approximation method, which is one of the frequently used approximation methods along with Oustaloup, Continued Fraction Expansion (CFE), Modified Stability Boundary Locus (MSBL) and El-Khazali methods [1]. Matsuda's approximation technique is fundamentally grounded in the CFE method and seeks to align an integer-order approximate model with FO derivative operators. This alignment is specifically targeted at sampled frequencies that

are distributed evenly on a logarithmic scale. In other words, Matsuda's approximation method was proposed as an improvement of the CFE approximation method because the CFE method does not allow adjustment of operating frequency ranges, where the transfer function model of the CFE method behaves as a FO operator. This frequency adjustability improvement provides versatility advantages in the practical realization of FO elements for the real-world applications [1], [10]. In this study, to reduce the complexity of the realization process, the FO integral operator was transformed to a 4th integer order approximate transfer function rather than a higher order approximate transfer function by Matsuda approximation method as follows [1]:

$$T(s) = \frac{0.1132s^4+3.439s^3+5.853s^2+1.068s+0.01778}{s^4+6.007s^3+3.291s^2+0.1934s+0.0006366} \quad (1)$$

Then, by using partial fraction expansion method, Equation (1) can be expressed as [1]:

$$T(s) = \frac{2.0010}{s+5.4049} + \frac{0.4943}{s+0.5360} + \frac{0.1806}{s+0.0628} + \frac{0.0828}{s+0.0035} + 0.1132 \quad (2)$$

This transfer function was obtained by using Matsuda approximation method for the 4 th order realization of fractional order integral operator $s^{-0.5}$ in the interval of $\omega \in [0.1,10)$ rad/s. The reason why this transfer function was used is that Multisim analog responses were obtained by using this transfer function in the author's former study [1]. To make a proper comparison of analog and digital (FPGA) responses, the same transfer function was considered.

A. Digital Implementation

To implement the continuous time function with FPGA in digital form, it is needed to convert the s-domain continuous time transfer function to z-domain discrete time transfer function. For this purpose, there are many conversion methods such as zero-order hold (ZOH), first-order hold (FOH), impulse-invariant mapping, Tustin approximation (bilinear transform), zero-pole matching equivalents, least squares. Each of these methods has its own strengths and weaknesses, and the choice of method depends on the specific requirements of the system being modeled and the nature of the input signals. All the mentioned methods have been tested in MATLAB, and it has been found for this study that Tustin method with the sample time of 0.01 s gives relatively better results for the handled transfer function when error rates are considered in a defined frequency range [11], [12], [13], [14].

Using the Tustin method with a small sampling time like Ts=0.01 yields better results because it minimizes frequency warping, accurately captures high-frequency behavior, reduces discretization error, and ensures the stability and dynamic characteristics of the continuous system are well-preserved in the discrete domain. Also, any different Ts value could be chosen. Ts value of 0.01 s yields a sufficiently proper discrete time signal. On the other hand, in a former study where analog implementation was realized, the frequency range was taken in the interval of [0.1, 10) rad/s to approximate the $s^{-0.5}$ integral operator by Matsuda method. This operating frequency range was selected hypothetically, considering any frequency range that may be encountered in systems in real-life applications, another frequency range could also have been chosen.

When the continuous time transfer function given in Equation (2), which expresses the main transfer function as the sum of first order low-pass filter forms, is converted to a discrete time transfer function by using Tustin method in MATLAB, the following function is obtained in z-domain:

$$\begin{aligned} T_m(z) = {} & \frac{0.009742 + 0.009742\, z^{-1}}{1 - 0.9474\, z^{-1}} \\ & + \frac{0.002464 + 0.002465\, z^{-1}}{1 - 0.9947\, z^{-1}} \\ & + \frac{0.0009027 + 0.0009027\, z^{-1}}{1 - 0.9994 z^{-1}} \\ & + \frac{0.000414 + 0.000414\, z^{-1}}{1 - z^{-1}} \\ & + 0.1132 \end{aligned} \quad (3)$$

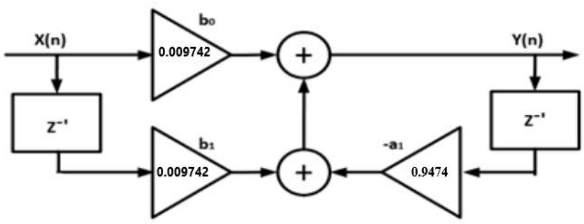
Each rational part of Equation (3) can be represented as a difference equation as shown in the following example, in which the first rational component of Equation (3) was written as a difference equation:

$$\frac{Y(z)}{X(z)} = \frac{0.009742 + 0.009742\, z^{-1}}{1 - 0.947\, z^{-1}} \quad (4)$$

$$X(z) * (0.009742 + 0.009742\, z^{-1}) = Y(z) * (1 - 0.947\, z^{-1}), \quad (5)$$

$$Y(n) = 0.009742 * X(n) + 0.009742 * X(n-1) + 0.947 * Y(n-1). \quad (6)$$

Since each rational term depends on the present and past inputs as well as the past outputs, the transfer function is realized as the sum of Infinite Impulse Response (IIR) filters. IIR filter is a digital filter with an impulse response that theoretically continues indefinitely. IIR filters are efficient regarding computational resources but can be less stable and introduce phase distortion if not designed carefully. Direct Form I structure was used to realize each first-order IIR filter as shown below, and the design example for the first rational part of the right side of Equation (3) is shown in Fig. 1.



$$\frac{Y(z)}{X(z)} = \frac{b_0+b_1z^{-1}}{1+a_1z^{-1}} \Rightarrow \frac{0.009742+0.009742z^{-1}}{1-0.9474z^{-1}}$$

Fig.1. Direct Form 1 Structure for a First Order IIR Filter

Equation (3) expresses the sum of four separate IIR filter equations and a static gain of 0.1132. The frequency response obtained for discrete time transfer function given by Equation (3), approximate continuous time transfer function given by Equation (1) and exact output are given in Fig. 2, which shows both the magnitude and phase responses in the operating frequency range of [0.1,10] rad/s. The frequency response of the discrete-time transfer function $T_m(z)$ and continuous time transfer function T(s) obtained by Matsuda approximation method are almost identical to each other, and they are very close to exact response in the defined frequency range.

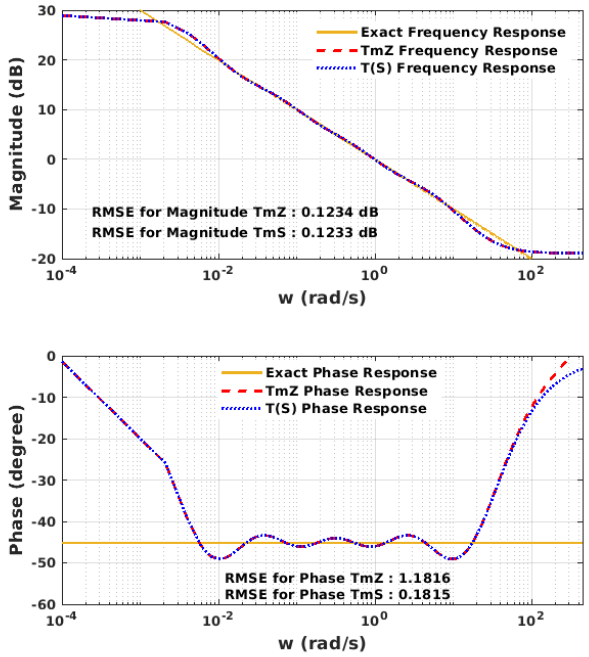


Fig.2. Bode Diagram of the Discrete Time Transfer Function

Then, Equation (3) is implemented as the sum of IIR filters on Simulink using the Xilinx System Generator tool. The Xilinx System Generator generates the IP for the function. The generated IP is used on Vivado IDE to synthesize, get Register Transfer Level (RTL) analysis, and implement it on FPGA, as seen with the block diagram in Fig. 3.

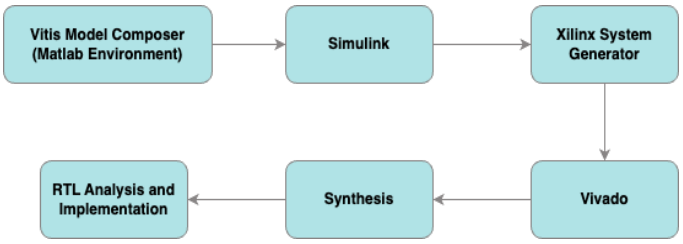


Fig.3. Simulink Block diagram based on Xilinx System Generator Tool

Xilinx System Generator design tool is provided by Xilinx, one of the leading companies producing FPGA, for implementing FPGA applications using a model-based approach in MATLAB Simulink [15]. The working principle of this tool can be explained as follows: Xilinx System Generator includes a DSP block set specific to Xilinx within Simulink. This block set allows for creating a model based on design specifications. Xilinx System Generator then uses the model designed using these block sets to generate a Register Transfer Level (RTL) netlist with the help of Xilinx components. The netlist represents the implementation or the connections of a specific logic design, which can be depicted through diagrams as a visual representation. The RTL netlist is transferred to the Xilinx Vivado design suite to generate the bitstream and application. Hardware Co-Simulation can be performed after the bitstream file is interfaced with the Xilinx System Generator and downloaded onto the FPGA device [16], [17], [18]. In this study, subsystems have been created for filters that are part of the transfer function. Pre-built add blocks were utilized to combine these systems or filters. Gateway blocks serve as interfaces between Simulink blocks and FPGA blocks [19].

III. RESULTS AND DISCUSSIONS

For FPGA simulation, IIR filters are designed using Xilinx System Generator. The discrete time transfer function consisting of a static gain block and four IIR filter blocks is implemented in two separate ways, shown in parallel with each other in Fig. 4, as FPGA (IIR multiple subsystem) and Simulink discrete filter subsystem, for comparison. The outputs of these subsystems and the input signals are compared by displaying them with a time scope.

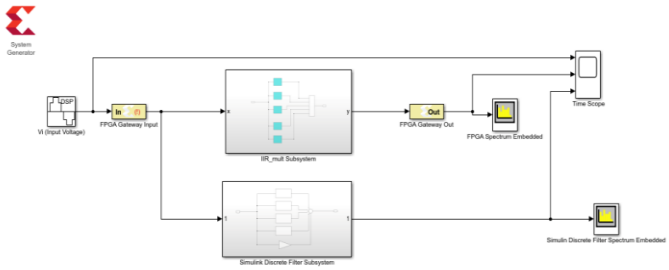


Fig.4. Implementation and comparison of filter in Xilinx System Generator at Vitis Model Composer

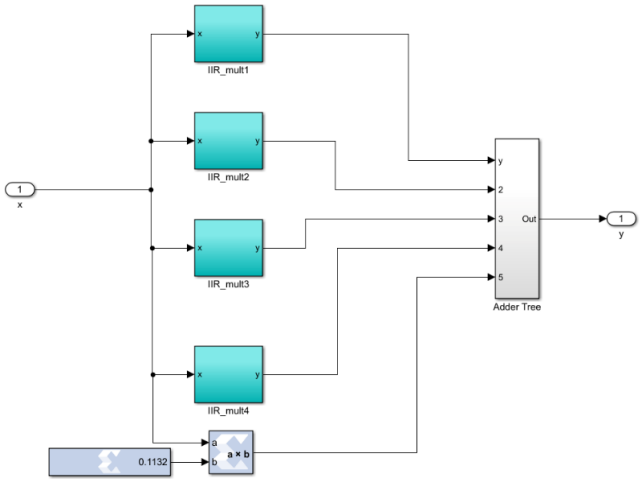


Fig.5. Implementation of IIR_mult_Subsystem Block

Each first-order IIR filter was represented by a subsystem which was called IIR_mult (see in Fig. 5). These subsystems were combined using Adder Tree from the Xilinx DSP library to obtain the output of the implemented function.

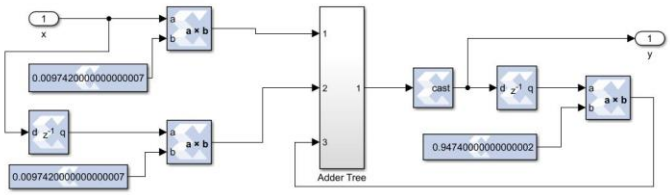


Fig.6. Implementation of IIR_mult1 Block

In these IIR_mult multiplexer subsystems seen in Fig. 6, the Direct Form I implementation of the IIR filter, which is the first rational component of $T_m(z)$, is achieved using three multipliers, two registers, one convert block and an adder tree. Although full resolution is maintained through the multipliers and the adder tree, the data path cannot expand indefinitely. Therefore, a quantization block has been placed at the output of the adder tree to reduce data width back to its input width [15].

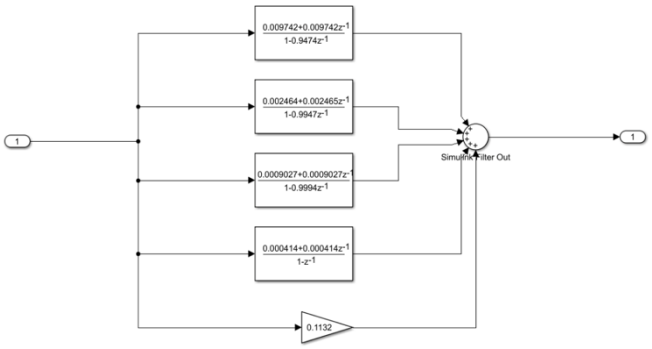


Fig.7. Design of Simulink Discrete Filter Subsystem

On the other hand, a Simulink Discrete Filter subsystem is implemented to simulate the filter function. This subsystem contains four Discrete Filter blocks and a static gain block as seen in Fig. 7, similar to IIR_mult subsystems. The outputs of the blocks are combined to obtain the final output of the filter.

Then, different waveforms were applied to the input of the system to check the accuracy, consistency and practicability for FPGA implementation. At first, sine waveform of 0.2 V and 0.5 Hz was applied as input. The obtained output waveforms and the Root Mean Square Error (RMSE) for Multisim analog and FPGA responses were presented in Fig. 8. This figure indicates that the model, when implemented in simulation, performs consistently, suggesting accurate system behavior for sine input. The relatively low RMSE values for both analog and digital realization outputs suggest that the FPGA and Multisim implementations are highly accurate and closely follow the analytical response in defined frequency range. The relatively higher error in the FPGA implementation can be attributed to the discrete time nature of the system, which includes quantization and discretization effects. However, these small errors are omittable for practical applications. On the other hand, the coherence seen at the output signals reveals negligible phase or amplitude errors between FPGA, Multisim, Simulink and analytical outputs. Thus, one can conclude that FPGA can reliably emulate the theoretical characteristic of a transfer function in the real world in a defined frequency range. This comparison shows the effectiveness of the FPGA design and its suitability for real-time signal processing applications.

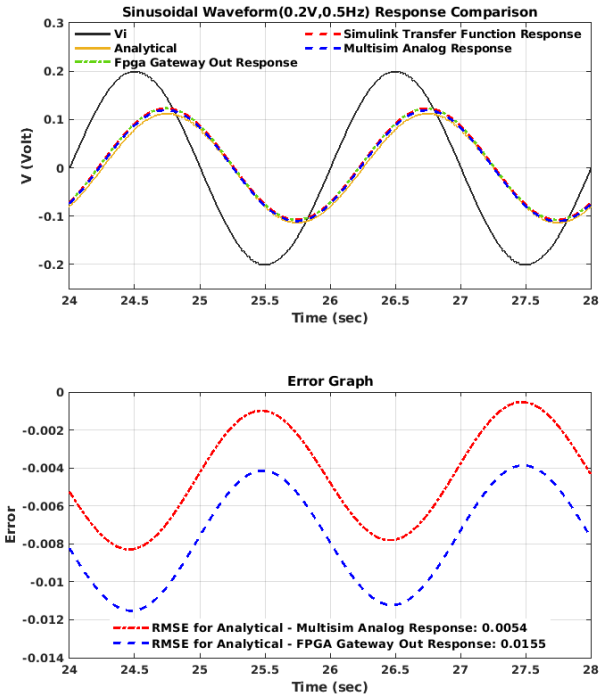


Fig.8. Sinusoidal Waveform Responses for Discrete Time Filter and Its Digital and Analog Circuit Realization and the Error Graph

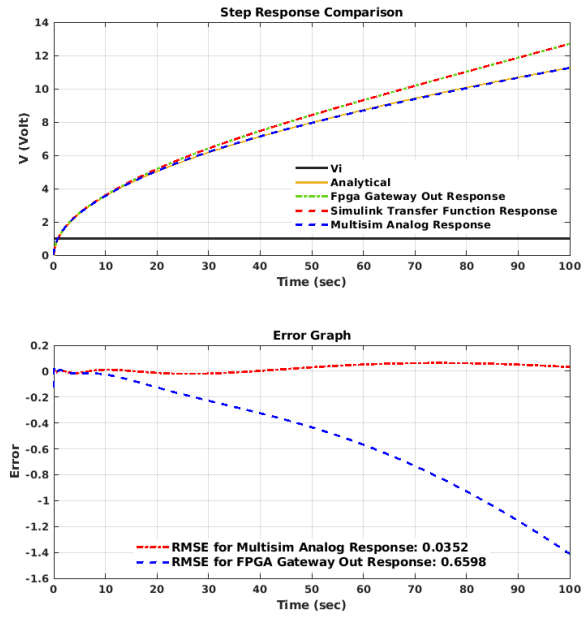


Fig.9. Step Responses for Discrete Time Filter and Its Digital and Analog Realization and the Error Graph

Fig. 9 shows 100 seconds step response and error graph for FPGA gateway out, Simulink transfer function, analytical response and Multisim analog output. The error graph was obtained by calculating the difference between analytical and realization responses. Also, calculated RMSE values for both Multisim Analog response and FPGA gateway out were added to the graph. The FPGA and Simulink responses closely follow the Multisim analog circuit response and analytical response for the first 30-35 seconds. After 35 seconds a deviation between FPGA response and analytical or Multisim analog response is observed. This deviation may arise from a variety of factors resulting from the nature of digital implementation besides the characteristics of the system being simulated. This case can be explained by numerical precision and quantization errors, sampling and discretization issues, accumulated errors from numerical computations and the closeness of the poles of the discrete-time transfer function to the unit circle. More comprehensively, the differentiations observed especially in time domain responses can be explained as follows: The divergence over time can be attributed to fundamental differences in how analog and digital systems process signals. Analog systems operate on continuous signals with theoretically infinite precision, while digital systems rely on sampled and quantized representations, introducing small errors that accumulate over time. In digital systems, numerical integration, frequency warping during discretization, and finite precision arithmetic can lead to inaccuracies, especially in long-duration responses. Feedback delays and numerical limitations in digital computations further contribute to the observed discrepancies. These factors together cause more deviations in digital response compared to the analog responses as time progresses. Nonetheless, the FPGA may be used particularly for real-time applications requiring flexibility and high-speed processing.

Fig. 10 shows continuous time analytical response, discrete time response obtained from FPGA Gateway out and Simulink for 1 second pulse input voltage. Also, the calculated RMSE value for FPGA Gateway out was added to the graph. As seen, the responses obtained for MATLAB Xilinx system generator realization and Simulink discrete model agree with each other. The low RMSE value confirms the consistency, the reliability and the effectiveness of the FPGA implementation in accurately modeling the dynamic behavior of the system for a 1-second pulse input.

Fig. 11 shows the 1 V square wave responses for the FPGA, Simulink Transfer Function, and Multisim Analog Circuit. A remarkable consistency is observed between the FPGA, Simulink, and Multisim responses, indicating that digital simulations can be as accurate as analog implementations.

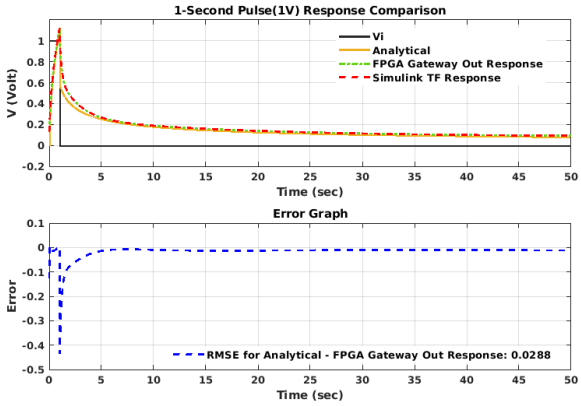


Fig.10. Responses of the Discrete Time Function Obtained for 1-Second Pulse (1 V) and Analytical-FPGA Error Graph

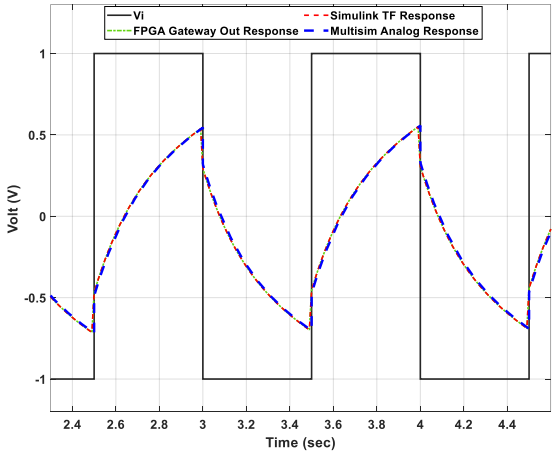


Fig.11. Square Wave Response for FPGA, Simulink Transfer Function and Multisim Analog Circuit

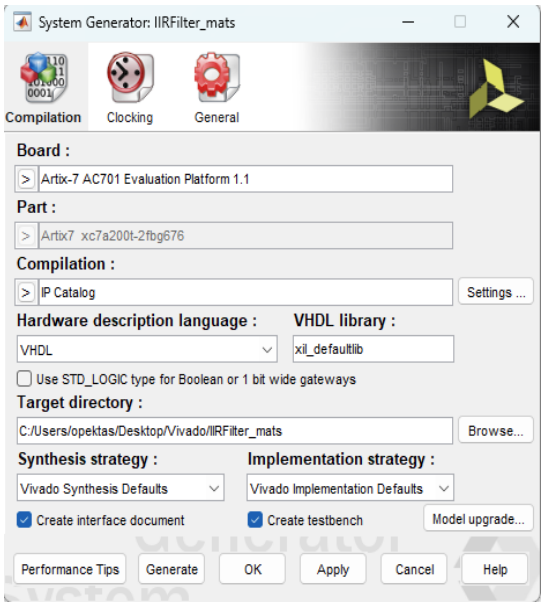


Fig.12. Xilinx System Generator Settings

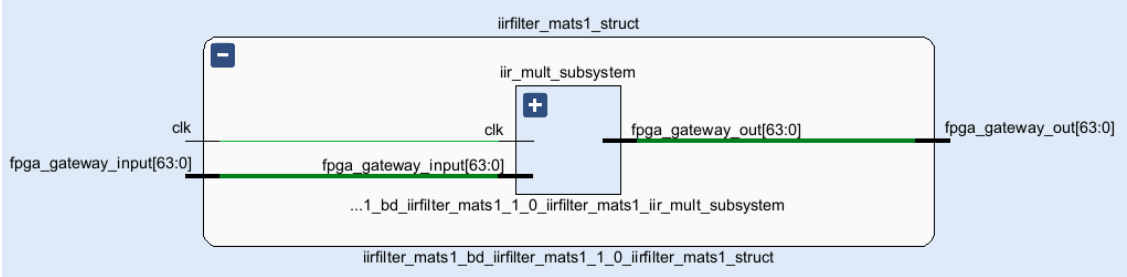


Fig.13. Filter's Elaborated Design Structure obtained with RTL Analysis at Vivado

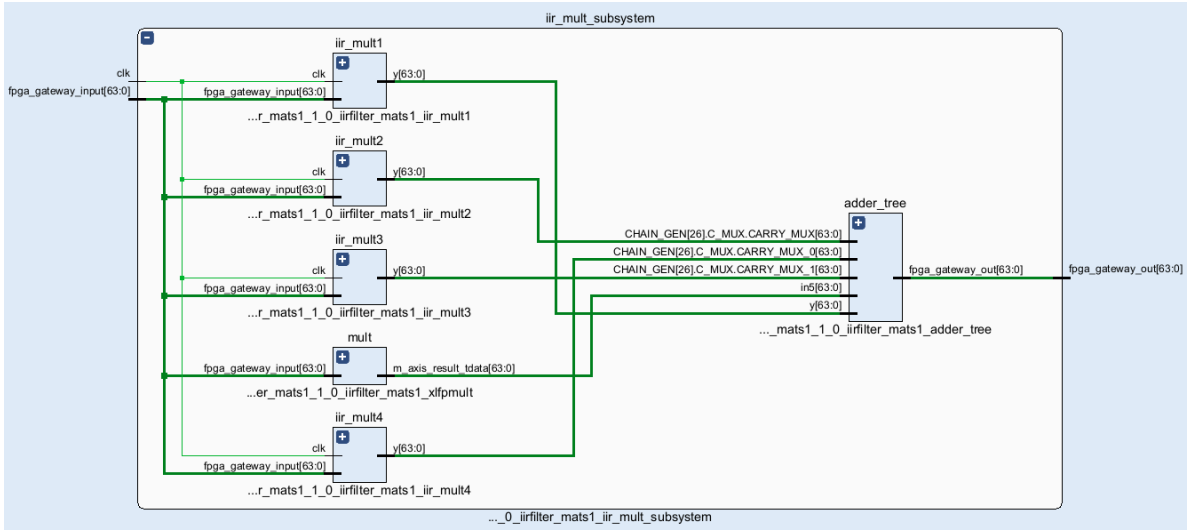


Fig.14. IIR_mult_subsystem Structure at Filter's Elaborated Design

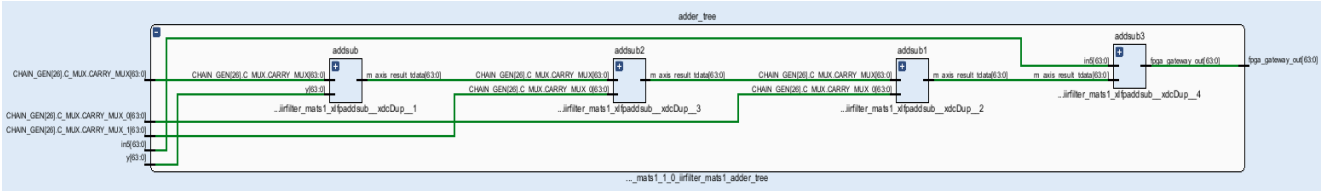


Fig.15. The Adder Tree Structure at the Design of IIR_mult_subsystem

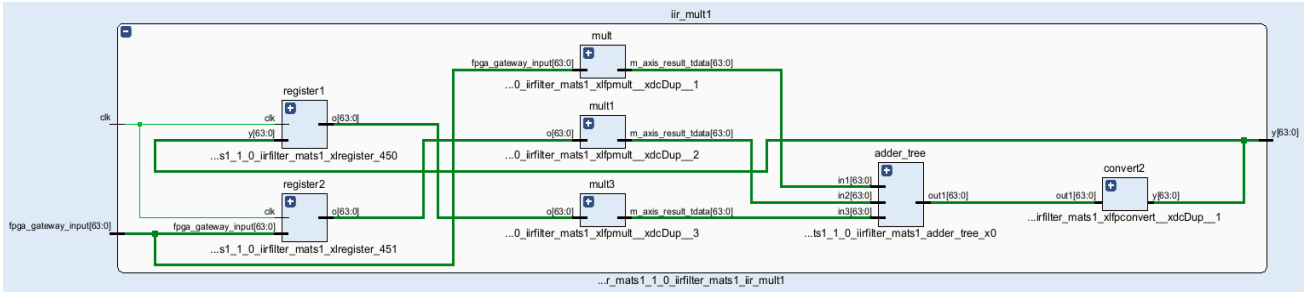


Fig.16. IIR_mult1 multiplexer Structure at the Design

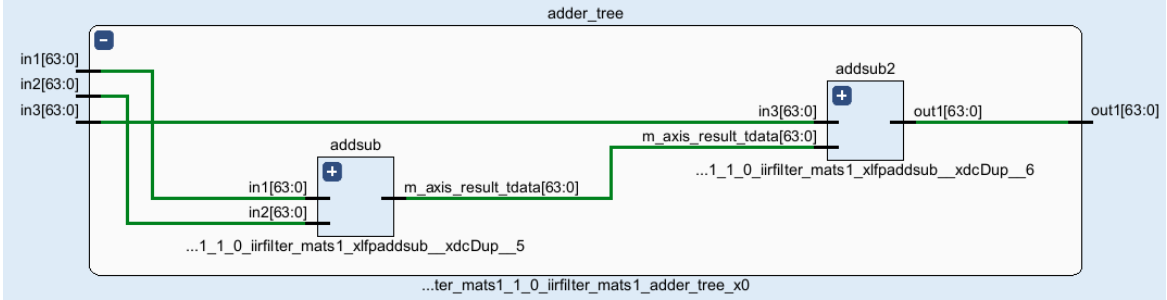


Fig.17. The Adder Tree Structure at the Design of IIR_mult1

After obtaining the simulation result graph, we generate the filter design and obtain power data using the Xilinx System Generator block shown in Fig. 12. Artix-7 AC701 board is used to synthesis and RTL analysis. Since the synthesis will be performed via Vivado, we select the default Vivado Synthesis option in the Synthesis Strategy section. Then, we choose the target directory for saving the output and click the generate button to complete the system generation.

To open the generated system, launch the Vivado application and select the .xpr file from the saved location via the Open Project option. Once the project is open, access the Elaborated Design to view the detailed design of the filter. This will display the design shown in the image below. The iirfilter_mats_bd block in the image represents our filter design, which includes the clk and FPGA_gateway_input inputs and the FPGA_gateway_output output. By double-clicking on this block, we can access the subsystems that comprise it. The iir_mult_subsystem in the design is similar to our design in MATLAB Simulink. As shown in the image, each iir_multx system consists of three multiplexers, three registers, and one adder tree, just like the MATLAB Simulink design. Since our filter is of fourth order, a total of four multiplexers were used, and their results were combined using adders to obtain the filter output (see in Fig. 13,14,15,16 and 17).

Fig. 13, 14, 15, 16 and 17 show the Elaborated design created on Vivado as a result of the RTL analysis of the filters. As can be seen, a design similar to the filter design we designed on MATLAB Simulink has been created. In this design, the filter was designed using the collectors and multipliers in Vivado. Thus, the filter was synthesized, and the necessary working data was obtained by performing RTL analysis.

TABLE I
SYNTHESIS REPORT RESULTS OF THE FILTER AT VIVADO

Resource	Utilization	Available	Utilization
LUT	11589	134600	8.61%
FF	1348	269200	0.50%
DSP	165	740	22.30%
IO	129	400	32.25%

Table 1 shows the synthesis report results which are obtained on Vivado. The utilization of LUTs is quite low, indicating that the FPGA still has a significant number of available resources for further implementation or additional functionality. The 8.61% utilization shows that the design is efficient in terms of LUT usage. Flip-Flop (FF) utilization is minimal, similar to LUTs, with only 0.5% of the available FFs being used. This suggests that the design does not heavily rely on sequential logic elements, leaving ample room for further expansion. DSP utilization is higher than LUTs and FFs but still reasonable at 22.3%. This level of usage indicates that the design includes some memory-intensive components but remains within a comfortable range for resource availability. The IO utilization is relatively higher at 32.25%, which is expected for designs requiring substantial external interfacing. However, it remains within a manageable range, suggesting that the design is efficient regarding IO resource allocation. Global buffers (BUFG) are minimal, indicating that the design's clock distribution network is not heavily taxing the available resources. This is a good sign of efficient clock management.

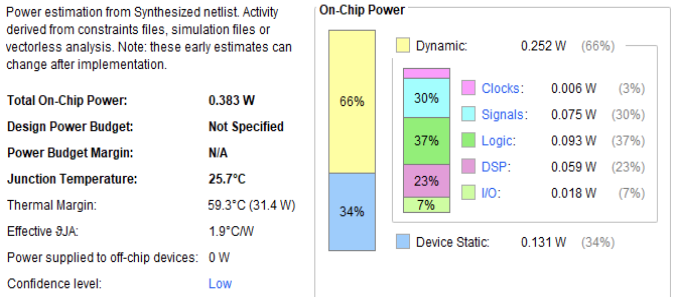


Fig.18. Power Analysis of Filter

Figure 18 shows the Power analysis obtained by implementing the filter to FPGA. As is seen, the total power consumption of the FPGA is moderate at 0.383 W, which suggests an efficient design with manageable thermal characteristics. The junction temperature is well within safe operating limits, with a substantial thermal margin indicating that the FPGA runs cool and efficiently. On the other hand, the dynamic power analysis shows the usage of clocks, signals, logic, DSP, and input-output(I/O) percentages on-chip power. The clock distribution is minimal, indicating efficient clock gating and distribution mechanisms. The most considerable portion is consumed by signal switching, which is expected in designs with significant data movement and processing. The logic power consumption is moderate, reflecting the computational activities within the FPGA. The DSP blocks consume substantial power, indicating the design's reliance on digital signal processing capabilities. Lastly, the I/O power consumption is relatively low, which is advantageous for minimizing overall power consumption. Also, the static power analysis highlights the importance of leakage power in the overall power budget. This is typical for modern FPGAs.

This study shows that FPGAs offer significant advantages for implementing FO systems in digital signal processing, particularly in performance, flexibility, and energy efficiency. However, these benefits come with trade-offs in design complexity, development cost, and resource constraints. As FPGA technology continues to evolve, some of these challenges will likely be mitigated, further enhancing the suitability of FPGAs for fractional-order DSP applications [20], [21]. In summary, FPGA implementation shows efficient resource utilization and power consumption. The low LUT and FF usage percentages indicate that the FPGA has ample room for additional logic or future expansion. Power consumption is low, with dynamic power dominating due to active signal processing and logic operations. The design runs cool with a significant thermal margin, suggesting good thermal management. Overall, the FPGA design is resource-efficient and power-efficient, with room for further optimization and expansion.

IV. CONCLUSION

In this study, 4th integer order approximate continuous time transfer function was obtained for FO integral operator $s^{-0.5}$ in the operating frequency range of [0.1,10] rad/s by Matsuda's approximation method. Then this function was converted to a discrete time function with Tustin method, and that function was digitally implemented by FPGA with Xilinx System Generator. The results obtained were analyzed in comparison with analog circuit implementation results presented in a former study [1]. The results obtained from FPGA design, executed with Xilinx System Generator, have demonstrated a superior coherence with the analog realization results and exact results when frequency response is considered.

For the sinusoidal waveform, the FPGA system exhibited a slightly higher RMSE value (0.0155) compared to that of Multisim analog response (0.0054), highlighting minor discrepancies due to digital quantization. For the 1-second pulse response, the FPGA system achieved an RMSE of 0.0288, demonstrating a strong agreement with the analytical model. Also, an excellent agreement between the Multisim analog response and FPGA Gateway out for the square wave input was observed. When the responses of FPGA realization, analog realization and exact analysis for sinusoidal input, square wave input and 1-second pulse input were compared and calculated RMSE values were considered, it was seen that the FPGA results mostly yielded a good agreement with exact and analog results, demonstrating a considerable accuracy and reliability in replicating the dynamic behavior of fractional-order systems especially in the predefined frequency ranges.

When a 100 second step time response was considered, a time dependent increasing deviation was observed especially after 35 seconds between analytical and FPGA Gateway out responses, resulting in the RMSE of 0.6598 for FPGA Gateway out and RMSE of 0.0352 for Multisim analog output. This deviation is supposed to be resulted from the discrete time nature of digital systems, which causes numerical precision and quantization errors, sampling and discretization issues, accumulated errors from numerical computations. This issue is predicted to be substantially solved by increasing precision in the FPGA realization, monitoring and mitigating accumulated numerical errors, validating sampling time and stability.

Compared to the analog realization, the FPGA design demonstrated significant advantages, including reconfigurability, low power consumption (0.383 W), and efficient resource utilization, with LUT usage at 8.61% and DSP utilization at 22.3%. These findings underscore the practicality of FPGA implementations for real-time applications, offering high-speed processing, adaptability, and precision while overcoming the limitations of analog circuits, such as inflexibility and sensitivity to component tolerances. However, it should be noticed that analog implementation, which includes many design and configuration difficulties, has given relatively better time response. This comparative analysis highlights the suitability of FPGA-based implementations for FO systems in control and signal processing applications when frequency response is considered.

REFERENCES

[1] M. Koseoglu, F. N. Deniz, B. B. Alagoz, A. Yuce, and N. Tan, "An experimental analog circuit realization of Matsuda's approximate fractional-order integral operators for industrial electronics," *Eng. Res. Express*, vol. 3, no. 4, p. 045041, Dec. 2021, doi: 10.1088/2631-8695/ac3e11.

[2] A. M. Hassanein, A. H. Madian, A. G. G. Radwan, and L. A. Said, "On the Design Flow of the Fractional-Order Analog Filters Between FPAA Implementation and Circuit Realization," *IEEE Access*, vol. 11, pp. 29199–29214, 2023, doi: 10.1109/ACCESS.2023.3260093.

[3] J. Nako, C. Psychalinos, and A. S. Elwakil, "A 1 + α Order Generalized Butterworth Filter Structure and Its Field Programmable Analog Array Implementation," *Electronics*, vol. 12, no. 5, p. 1225, Mar. 2023, doi: 10.3390/electronics12051225.

[4] M. A. George, A. S. Elwakil, A. Allagui, and C. Psychalinos, "Design of Complex-Order PI/PID Speed Controllers and its FPAA Realization," *IEEE Access*, vol. 11, pp. 118606–118614, 2023, doi: 10.1109/ACCESS.2023.3326446.

[5] A. Ali, K. Bingi, R. Ibrahim, P. A. M. Devan, and K. B. Devika, "A review on FPGA implementation of fractional-order systems and PID controllers," *AEU - Int. J. Electron. Commun.*, vol. 177, p. 155218, Apr. 2024, doi: 10.1016/j.aeue.2024.155218.

[6] C. X. Jiang, J. E. Carletta, and T. T. Hartley, "Implementation of Fractional-order Operators on Field Programmable Gate Arrays," in *Advances in Fractional Calculus: Theoretical Developments and Applications in Physics and Engineering*, J. Sabatier, O. P. Agrawal, and J. A. T. Machado, Eds., Dordrecht: Springer Netherlands, 2007, pp. 333–346. doi: 10.1007/978-1-4020-6042-7_23.

[7] W. Wolf, *FPGA-based system design*. Pearson education, 2004.

[8] U. Meyer-Baese and U. Meyer-Baese, *Digital signal processing with field programmable gate arrays*, vol. 65. Springer, 2007.

[9] M. Koseoglu, "Time response optimal rational approximation: Improvement of time responses of MSBL based approximate fractional order derivative operators by using gradient descent optimization," Eng. Sci. Technol. Int. J., vol. 35, p. 101167, Nov. 2022, doi: 10.1016/j.jestch.2022.101167.

[10] F. N. Deniz, B. B. Alagoz, N. Tan, and M. Koseoglu, "Revisiting four approximation methods for fractional order transfer function implementations: Stability preservation, time and frequency response matching analyses," *Annu. Rev. Control*, vol. 49, pp. 239–257, 2020, doi: 10.1016/j.arcontrol.2020.03.003.

[11] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital control of dynamic systems*, vol. 3. Addison-wesley Menlo Park, 1998.

[12] "Matlab Documentation 'c2d,'" c2d. [Online]. Available: https://www.mathworks.com/help/control/ref/dynamicsystem.c2d.html?s_tid=doc_ta

[13] K. Ogata, *Discrete-time control systems*. Prentice-Hall, Inc., 1995.

[14] J. G. Proakis, *Digital signal processing: principles, algorithms, and applications, 4/E*. Pearson Education India, 2007.

[15] "Xilinx System Generator for DSP Getting Started Guide." AMD. Accessed: Aug. 16, 2024. [Online]. Available: https://docs.amd.com/v/u/en-US/sysgen_gs

[16] A. Sharma and T. K. Rawat, "Design and FPGA implementation of lattice wave fractional order digital differentiator," *Microelectron. J.*, vol. 88, pp. 67–78, Jun. 2019, doi: 10.1016/j.mejo.2019.04.013.

[17] D. Datta and H. S. Dutta, "High performance IIR filter implementation on FPGA," *J. Electr. Syst. Inf. Technol.*, vol. 8, no. 1, p. 2, Dec. 2021, doi: 10.1186/s43067-020-00025-4.

[18] V. Dhillon, S. Nair, A. Pabarekar, M. Kumbhare, K. Thakur, and R. Krishnan, "Implementation of FIR Digital Filter on FPGA," in *2021 4th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*, NaviMumbai, India: IEEE, Jan. 2021, pp. 1–5. doi: 10.1109/ICNTE51185.2021.9487744.

[19] H. V. Dixit and D. V. Gupta, "IIR filters using Xilinx System Generator for FPGA implementation," *Int. J. Eng. Res. Appl.*, vol. 2, no. 5, pp. 303–307, 2012.

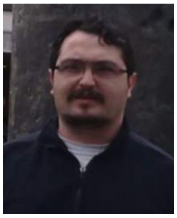
[20] P. Paz and M. Garrido, "Efficient Implementation of Complex Multipliers on FPGAs Using DSP Slices," *J. Signal Process. Syst.*, vol. 95, no. 4, pp. 543–550, Apr. 2023, doi: 10.1007/s11265-023-01867-7.

[21] P. Kwiatkowski, "Digital-to-time converter for test equipment implemented using FPGA DSP blocks," *Measurement*, vol. 177, p. 109267, Jun. 2021, doi: 10.1016/j.measurement.2021.109267.

BIOGRAPHIES

Ömer Pektaş Malatya, Turkey, in 1991. He received the B.S. and M.S. degrees from the Department of Electrical & Electronics Engineering, Inonu University, Malatya, Turkey in 2015 and 2018, respectively. He is a Ph.D. candidate in electrical-electronics engineering at Inonu University, Malatya.

He is working as a lecturer at Department of Vocational School of Technical Sciences, Electric and Energy, University of Karamanoglu Mehmetbey, Karaman, Turkey. His research interests include digital circuit implementation, FPGA design and programming, wearable sensors, object-oriented programming and mobile application development.



Murat Köseoğlu was born in Giresun, Turkey in 1978. He received the B.Sc., the M. Sc. and the Ph.D. degrees from the Department of Electrical & Electronics Engineering, Inonu University, Malatya, Turkey in 2000, 2003 and 2010, respectively. He worked as engineer in several power plants between 2000 and 2002. He worked as Research Assistant in the Department of Electrical & Electronics Engineering, Inonu University, Malatya, Turkey between 2002 and 2011. He worked as an Assistant Professor in the same department between 2011 and 2024. He has been working as an Associate Professor in the same department since 2024. His research interests include electronics, dielectrics, semiconductors and discharge phenomena.