Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı 26, S. 122-127, Temmuz 2021
© Telif hakkı EJOSAT'a aittir
Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue 26, pp. 122-127, July 2021
Copyright © 2021 EJOSAT
Research Article

FPGA Design of a Fourth Order Elliptic IIR Band-Pass Filter Using LabVIEW

Güner Tatar [1], İhsan Çiçek[2*], Salih Bayar[3]

[1] Fatih Sultan Mehmet Vakıf University, Faculty of Engineering, Departmant of Electrical-Electronic Engineering, İstanbul, Turkey, (ORCID: 0000-0002-3664-1366), gtatar@fsm.edu.tr
[2*] İstinye University, Faculty of Engineering, Departmant of Electrical-Electronic Engineering, İstanbul, Turkey, (ORCID: 0000-0002-7881-1263), ihsan.cicek@istinye.edu.tr
[3] Marmara University, Faculty of Engineering, Departmant of Electrical-Electronic Engineering, İstanbul, Turkey, (ORCID: 0000-0002-4600-1880), salih.bayar@marmara.edu.tr

(3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications June 11-13, 2021)

(**DOI:** 10.31590/ejosat.951601)

ATIF/REFERENCE: Tatar, G., Cicek, I., & Bayar, S. (2021). FPGA Design of a Fourth Order Elliptic Band-Pass Filter Using LabVIEW. *European Journal of Science and Technology*, (26), 122-127.

Abstract

Infinite impulse response filters are often used to meet the demand of modern electrical engineering applications such as image processing, digital signal processing and telecommunications because of the high selectivity and computational efficiency. In mission-critical real-time applications, computational latency is usually intolerable. High-speed processing of data and the coefficients require a digital signal processor or an FPGA instead of a general-purpose microprocessor. In the last two decades, FPGAs have been applied to many fields of signal processing due to parallelism determined scalable performance and run-time reconfigurability. This study presents a LabVIEW driven FPGA hardware design of a fourth-order discrete-time IIR elliptic band-pass filter. The designed filter has 2 kHz low and 2.5 kHz high cut-off frequencies with 1dB pass-band ripple and 80dB stop-band attenuation. The expected behavior of the designed filter has been confirmed by the functional simulation of the developed VHDL model. LabVIEW FPGA resource estimation reports a compact footprint for the proposed design.

Keywords: Digital filter, signal processing, Infinite Impulse Response (IIR), Z-Transform, VHDL, LabVIEW.

LabVIEW Kullanarak Dördüncü Derece Eliptik IIR Bant Geçiren Filtrenin FPGA Tasarımı

Öz

Sonsuz dürtü yanıt filtreleri, yüksek seçicilik ve hesaplama verimliliği nedeniyle görüntü işleme, dijital sinyal işleme ve telekomünikasyon gibi modern elektrik mühendisliği uygulamalarının talebini karşılamak için sıklıkla kullanılır. Görev açısından kritik gerçek zamanlı uygulamalarda, hesaplama gecikmesi genellikle kabul edilemez. Verilerin ve katsayıların yüksek hızda işlenmesi, genel amaçlı bir mikroişlemci yerine bir dijital sinyal işlemcisi veya bir FPGA gerektirir. Son yirmi yılda, FPGA'lar paralellik ile belirlenen ölçeklenebilir performans ve çalışma zamanı yeniden yapılandırılabilirliği nedeniyle birçok sinyal işleme alanına uygulanmıştır. Bu çalışma, dördüncü dereceden ayrık zamanlı IIR eliptik bant geçiren filtrenin LabVIEW tabanlı FPGA donanım tasarımını sunar. Tasarlanan filtre, 1dB geçiş bandı dalgalanması ve 80dB durdurma bandı zayıflaması ile 2 kHz düşük ve 2.5 kHz yüksek kesme frekanslarına sahiptir. Tasarlanan filtrenin beklenen davranışı, geliştirilen VHDL modelinin işlevsel simülasyonu ile doğrulanmıştır. LabVIEW FPGA kaynak tahmini, önerilen tasarım için küçük bir ayak izi rapor etmiştir.

Anahtar Kelimeler: Sayısal filtre, Sinyal işleme, Sonsuz dürtü yanıtı, Z-Dönüşümü, VHDL, LabVIEW.

[*] Corresponding Author: ihsan.cicek@istinye.edu.tr

1. Introduction

Many analog signal processing applications have evolved into digital format because of the continuous digital evolution. In a typical signal processing chain, we use analog-to-digital converters to quantize analog variables into the digital domain. Then process the digitized information using a CPU, DSP or an FPGA, as shown in Figure 1. Finally, transform the processed signal back into the analog domain by using digital-to-analog converters. Information processing in the digital domain allows advanced algorithms that offer higher performance and more flexibility than analog counterparts fail to provide. As a result of the performance requirements in specific applications, such as real-time systems, processing speed has become an important parameter. FPGA based designs offer low-latency signal processing capability to meet this demand [1-2].

In signal processing applications, filters are used to suppress undesired signals while allowing the signal of interest to pass through. Digital signal processing filters can be implemented using both the infinite impulse response (IIR) or finite impulse response (FIR) approaches [3]. IIR approach is more preferred in practice since it requires less memory, it is easier to implement, and it is faster when compared to the FIR configuration for the same type of filter [4-7]. IIR filters are often used where a linear phase is not required. Although IIR filters have advantages such as low implementation cost, low latency, and analog equivalent compared to FIR filters, they also have some disadvantages such as nonlinear phase characteristics, more detailed analysis requirements, and numerical instability. Even though they have these disadvantages, IIR filters are successfully used in many applications with infinite impulse responses.

On the other hand, FIR filters are frequently used in audio and biomedical signal processing applications where sufficient memory space and linear phase response are required. Their stable response against any arbitrary input signal is one of the essential advantages of the FIR filters. FIR filters have advantages such as stability, linear phase response, and fixed-point performance. However, they also have disadvantages such as excessive delay, high computation and memory requirements, and lack of analog equivalence (Bilirear, matched z-transform) [8-9]. Hardware design time of digital signal processing filters can be a bottleneck for success. Alternative higher-level design flows have been developed by software vendors such as National Instruments and MATLAB to reduce hardware design time. Although conventional hardware description language (HDL) based design flows are still in use today, higher-level synthesis approaches are becoming more popular because of their advantages, such as shorter design time and ease of use.

This study has chosen a fourth order IIR type discrete-time Elliptic filter topology as an example design scenario to compare the conventional HDL-based digital filter design flow against a high-level design approach. The system-level diagram of the designed filter is provided in Figure1. The filter has 2 kHz low and 2.5 kHz high cut-off frequencies with 1 dB pass-band ripple and 80 dB stop-band attenuation. The characteristics and resource utilization of the designed filters have been evaluated for the two design flows.

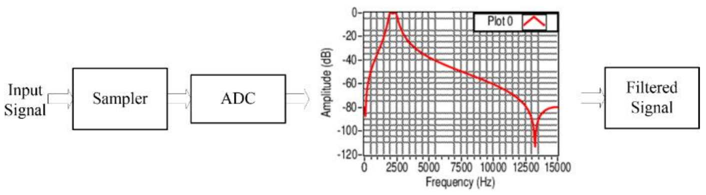


Figure 1. System diagram of the designed elliptic band-pass IIR filter

The paper is organised as follows; In Section II, we explain the design details of the discrete-time IIR elliptic band-pass filter. In Section III, we provide the details of the verification of the discrete-time IIR elliptic band-pass filter. Section IV gives the algorithmic flowchart and FPGA implementation result, and finally, we discussed results and presented future work in Section V.

2. Design of the Discrete-Time Elliptic Band-Pass IIR Filter

Band-pass filters allow signals in the band-pass region to go through and attenuate the signals outside of this region. One can adjust the passband and stopband ripple independently. If the stopband and passband ripples individually approach zero, the filter can transform into a Chebyshev type I or type II filter. Hence, to have a narrow transition band, both the passband ripple and the stopband ripple are allowed. Elliptical (also called Cauer) filters are optimum for providing a narrower transition band than other filters with the same order [9]. Discrete-time filters can be designed in IIR or FIR configurations. We preferred the IIR type elliptic filter design shown in Figure 2 due to its smaller memory requirement and ease of implementation. The IIR filter is a feedback system shown in Figure 2, and its impulse response is infinite. Mathematical difference equations that define the operation of the IIR filter are given through equations (1), (2) and (3).

We have chosen 100 kHz sampling frequency (f_s) 2 KHz lower cut-off frequency (f_l), 2.5 kHz upper cut-off frequency (f_u) with 1 dB pass band ripple and 80dB stop-band attenuation for the IIR elliptic band-pass filter. Since the filter is of fourth order, it has four zero and four poles located on the unit circle in the complex plane as shown in Figure 3. Frequency response (4-a) and phase response (4-b) of the filter are presented in Figure 4. The distance of the zeros to the center is 1. An analog filter with a transfer function *H(s)* can be converted into a digital filter with transfer function *H(z)* by using the appropriate Bilinear transformation method [9]. Namely, if *s=f(z)*, a projection can be established between the s-plane and the z-plane variables. This matching is used for the band-pass filter and corresponding frequency is obtained by substituting $s=j\omega$ and $z=e^{j\omega}$, where ω is the angular frequency and $\omega=2\pi f/f_s$ The Bilinear transformation used for the band-pass filter is provided by equations (4) and (5).

$$y[n]=\sum_{k=0}^{M} b_k x[n-k]-\sum_{k=1}^{N} a_k y[n-k] \tag{1}$$

$$H(z)=\frac{\sum_{k=0}^{M} b_k z^{-k}}{\sum_{k=1}^{N} a_k z^{-k}}, a_0=1 \tag{2}$$

$$H(z) = \frac{b_0 \prod_{k=0}^{M} (1 - c_k z^{-1})}{\prod_{k=1}^{N} (1 - d_k z^{-1})} \quad (3)$$

$$s = \frac{1 - 2\cos\omega_0 z^{-1} + z^{-2}}{1 - z^{-2}} \quad (4)$$

$$\Omega = \frac{\cos\omega_0 - \cos\omega}{\sin\omega} \quad (5)$$

Where a_k and b_k are the reverse and forward filter coefficients, respectively, T_s represents the sampling period and ω_s corresponds to the center frequency of the band-pass/stop filter.

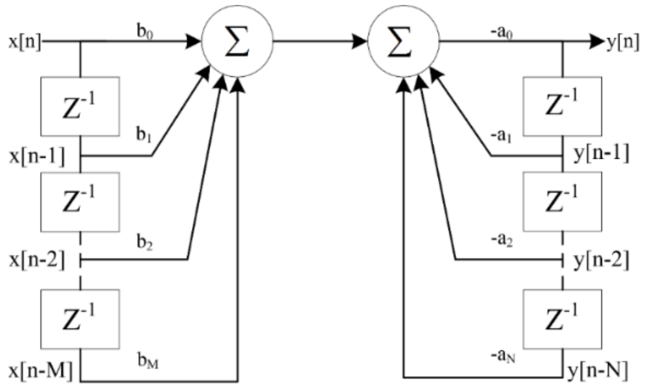


Figure 2. Block diagram of the IIR filter

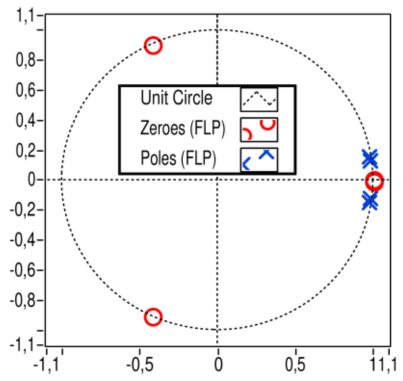
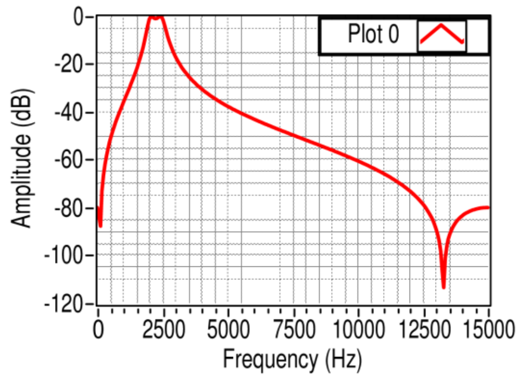
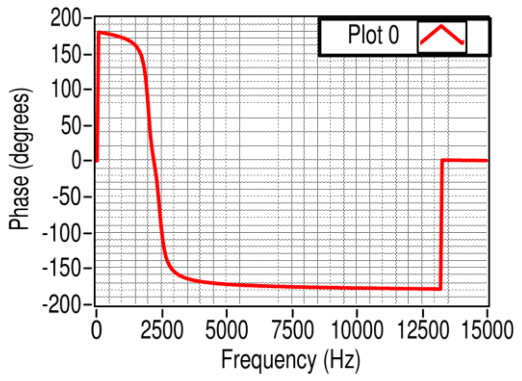


Figure 3. Zero/Pole location in the complex z-plane

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\left(\frac{2\pi nk}{N}\right)} \quad (6)$$



a)



b)

Figure 4. Frequency (a) and phase (b) response of the designed elliptic IIR band-pass filter

3. Verification of the Discrete-Time IIR Elliptic Band-Pass Filter

We have verified the discrete-time IIR elliptic band-pass filter operation using the simulations in the NI LabVIEW environment. We obtained the inverse and forward coefficients of the filter from the program written in the LabVIEW environment. We used the sum of any two test signals with 2.2 kHz and 3.5 kHz frequency and 100 kHz sampling frequency to obtain the filter coefficients. The amplitudes of these two test signals are given as 0.8V, and 1.2V phases 15 and 30 degrees, respectively. The test signal was applied to the filter according to 80dB stopband attenuation, 1dB passband ripple, and the filter coefficients obtained. The sampling frequency of the filter is 100 kHz, the lower cut frequency is 2 kHz, and the uppercut frequency is 2.5 kHz. The filter will allow signals in these two cut-off frequency ranges to pass while stopping signals outside the upper and lower cut-off band. LabVIEW program block diagram, which includes filter coefficients, was given in Figure 5. The filter coefficients obtained are given in Figure 6. In IIR type filters, the coefficients that multiply the inputs are forward coefficients, and the coefficients that multiply the outputs are the inverse coefficients. LabVIEW Elliptic Coefficients VI (Virtual Instrumentation) gives second and fourth-order filter coefficients. If a low-pass filter or a high-pass filter is to be designed, VI must be set in second-order, if band-pass or band-stop is designed, VI must be set in fourth-order. IIR digital filters in LabVIEW are designed by the bilinear transformation of the analogue filter's zeros and poles. LabVIEW design environment consists of two windows. One window keeps the system block diagram where the filter model is written, and the other one has the front panel where the outputs are shown. The written IIR filter model outputs are shown with the front panel in Figure 6. Filter coefficients obtained in Figure 6 were used in real-time FPGA application.

The sum of the 2.2 kHz signal of interest and a 3.5 kHz interference signal (7-a), and the filtered signal output (7-b) were presented in Figure 7. We can observe that the frequency of the output signal is 2.2 kHz with an amplitude of around 0.8V. The signal of interest is passed to the output since it lies within the passband of the designed filter. And the interfering 3.5 kHz signal has been filtered out as it is out of the passband. We calculated the filtered signal output frequency spectrum using Fast Fourier Transform (FFT), as shown in Figure 8. FFT is a fast algorithm used for calculating Discrete Fourier Transform (DFT) [11], which is defined by the equation (6). Applying DFT directly to N data instances is time-consuming since it requires computationally complex calculations (~N^2). For this reason, we used the LabVIEW built-in FFT algorithm.

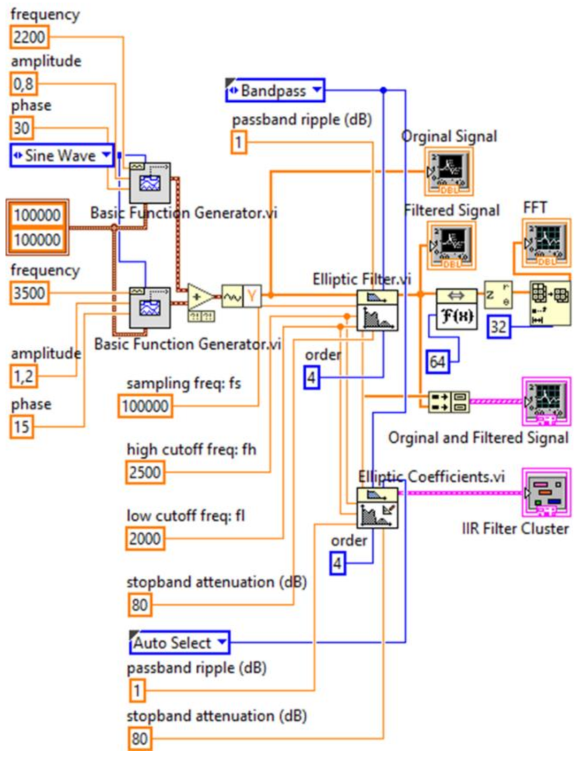
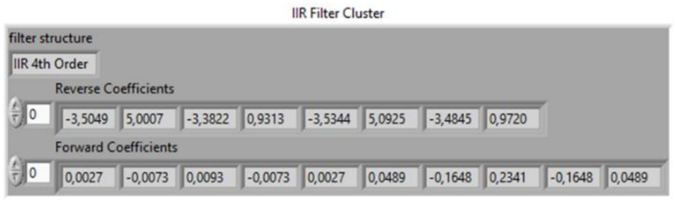


Figure 5. LabVIEW design of the IIR band-pass filter with filter coefficients.



IIR Filter Cluster

filter structure

IIR 4th Order

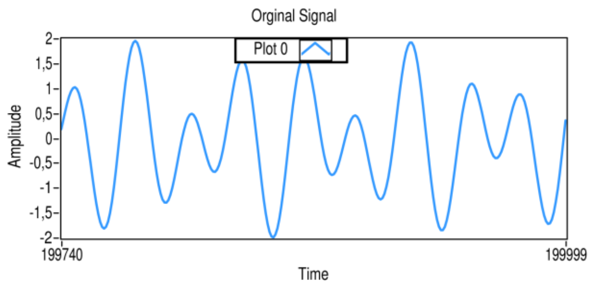
Reverse Coefficients

| 0 | -3,5049 | 5,0007 | -3,3822 | 0,9313 | -3,5344 | 5,0925 | -3,4845 | 0,9720 |

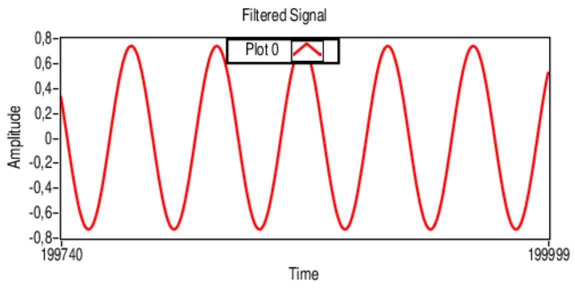
Forward Coefficients

| 0 | 0,0027 | -0,0073 | 0,0093 | -0,0073 | 0,0027 | 0,0489 | -0,1648 | 0,2341 | -0,1648 | 0,0489 |

Figure 6. Filter coefficients of the elliptic IIR band-pass filter.



a)



b)

Figure 7. Test signal at the input (a), and filtered signal at the output (b) of the elliptc IIR band-pass filter

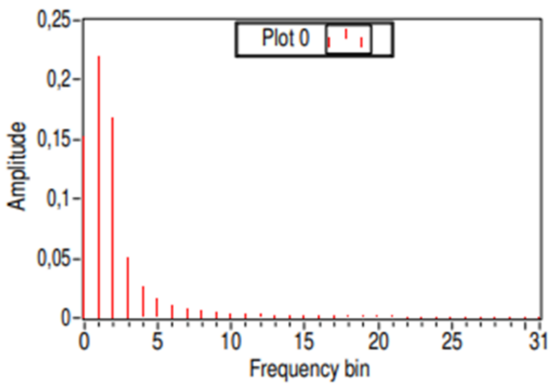


Figure 8. FFT of the filtered signal output

4. VHDL Implementation

We used the flowchart shown in Figure 9 for VHDL implementation of the elliptic IIR band-pass filter. The algorithm consists of a loop and multiple processes within the loop. Forward and reverse filter coefficients are needed for implementing the discrete-time IIR elliptic band-pass filter. The filter's reverse coefficients are due to the IIR filter's feedback. We created the VHDL model of the 4th order elliptic IIR band-pass filter based on the filter design approach [12]. We used the coefficients obtained from the LabVIEW program in the FPGA implementation. The filter can be designed by writing these filter coefficients into the difference equation in (1). Figure 9 shows the top-level VHDL design block. The code block consists of the entity block where we made input and output port assignments, the architecture in which we assigned constants and signals, and processes where we implemented all transactions. We used the same frequency coefficients used by the LabVIEW design in the VHDL implementation for a fair comparison. Since FPGAs can operate in parallel and the filter has a pipelined architecture, designers must keep the data in memory until the end of each calculation [12-14]. We shifted the data previously calculated and stored in the memory by one unit by considering the shift register and inserted the new value into the computation process. The input signal is multiplied with the filter coefficients and accumulated to create the output signal.

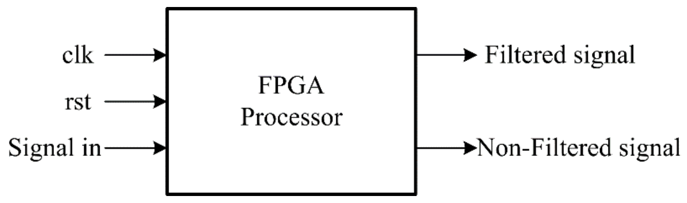
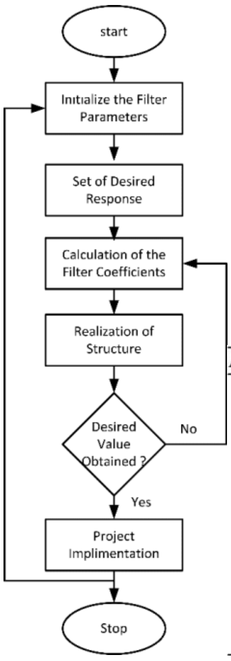


Figure 9. VHDL implementation code block

Algorithm 1 provides the pseudo-code of the VHDL implementation of the elliptic IIR band-pass filter. First, the upper and lower cutoff frequencies of the filter and the sampling frequency are determined to put the algorithm simply. Then when we run the simulation, the filter coefficients from LabVIEW are activated and multiplied by the input signal. Since the filter structure is in pipeline architecture, we keep the previous value of the filter output for each cycle in memory. Then, if the obtained value is equal to the desired value, the output signal is calculated. For the filter coefficients obtained from LabVIEW to coincide smoothly in the VHDL environment, the designers should pay attention to the parameter values and filter order.



Algorithm 1 Discrete-time elliptic IIR band-pass filter

```
1: Initialize the input values
2: Define clock frequency
3: Define the frequency ranges
4: Calculate filter coefficients
5: for iteration = 1,2,...... do
6:         Run program until desired value obtained
7:         if the value == desired value
8:         then
9:         Compute the output signal
10: end for
11: Implement as hardware
```

Figure 10. Flowchart [15] and algorithm of digital IIR filter design

Figure 11 shows the unfiltered and filtered result of 4th order IIR elliptic band-pass filter implementation written in VHDL code. ALDEC Active-HDL Student Edition has been used to simulate the VHDL model. As can be observed in the figure, we have obtained similar results compared to the LabVIEW design's simulation results. Table 1 presents the FPGA resource utilization estimated by LabVIEW.

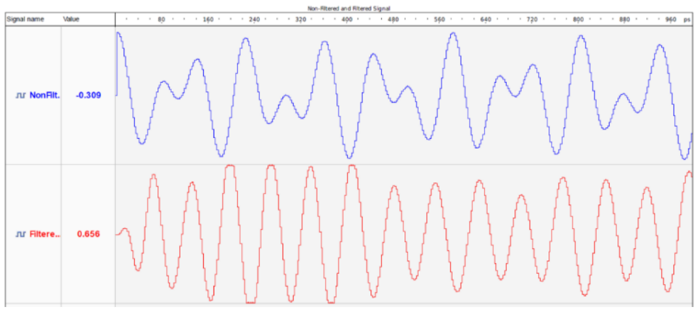


Figure 11. Functional simulation results of the VHDL model

Table 1. LabVIEW Estimated FPGA Resource Utilization

Data type	**FF**	**LUT**	**BRAM**	**MUX**
Elliptic filter	381	967	0	4
Function generator	94	156	1	0
Elliptic filter coefficients	381	967	0	4
FFT	697	2368	6	16
Total usage	1553	4458	0	24

4. Conclusions and Recommendations

In this study, we designed a 4th order discrete-time IIR elliptic band-pass filter using an FPGA. We obtained the coefficients of the discrete-time digital filter using the LabVIEW software. The designed filter can pass the signals between the 2.2 kHz and 3.5 kHz frequency range and suppress the signals outside this range. We chose the sampling frequency of the filter as 100 kHz. We developed a VHDL model of the proposed filter and performed simulations to verify the filter's functionality. We confirmed the LabVIEW estimated behaviour with the functional simulation of the developed VHDL model for the 4th order discrete-time IIR elliptic band-pass filter. LabVIEW FPGA resource estimation shows that the designed filter has a compact footprint, making it a convenient choice for integration.

As a result, when we consider LabVIEW and VHDL designs, prototyping and getting results in LabVIEW is easier than VHDL design. It is well-known that arithmetic optimization is required for a code written for simulation in VHDL language to implement a hardware unit. In other words, designers cannot use any VHDL code implementation that works in every simulation environment. They should give efforts to design fixed and floating-point representations, which is a time-consuming process. It makes more sense to use LabVIEW in terms of fast prototyping. LabVIEW-FPGA software module is all that is required for real-time hardware implementation of a LabVIEW-FPGA board of code written in a LabVIEW environment. VHDL language is challenging to learn by everyone, but it offers an advantage because it is readable. On the other hand, LabVIEW is easy to design because it provides graphical programming, but the code content cannot be very descriptive.

References

[1] Venieris, S. I., & Bouganis, C. S. (2017, September). Latency-driven design for FPGA-based convolutional neural networks. In 2017 27th International Conference on Field Programmable Logic and Applications (FPL) (pp. 1-8). IEEE.
[2] Tatar, G., Kılıç, O., & Bayar, S. (2019, November). FPGA Based Fault Distance Detection and Positioning of Underground Energy Cable by Using GSM/GPRS. In 2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT) (pp. 1-6). IEEE.
[3] Pal, R. (2017, December). Comparison of the design of FIR and IIR filters for a given specification and removal of phase distortion from IIR filters. In 2017 International Conference on Advances in Computing, Communication and Control (ICAC3) (pp. 1-3). IEEE.
[4] Paul, A., Khan, T. Z., Podder, P., Hasan, M. M., & Ahmed, T. (2015, February). Reconfigurable architecture design of FIR and IIR in FPGA. In 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 958-963). IEEE.
[5] Shukl, P., & Singh, B. (2020). Combined IIR and FIR filter for improved power quality of PV interfaced utility grid. IEEE Transactions on Industry Applications, 57(1), 774-783.
[6] Seshadri, R., & Ramakrishnan, S. (2021). FPGA implementation of fast digital FIR and IIR filters. Concurrency and Computation: Practice and Experience, 33(3), e5246.
[7] (2021, March 11). Difference between IIR and FIR filters: a practical design guide. https://www.advsolned.com/difference-between-iir-and-fir-filters-a-practical-design-guide/.
[8] (2021, March 11). Know all About FIR Filters in Digital Signal Processing. https://www.elprocus.com/fir-filter-for-digital-signal-processing/.

[9] (2021, February 2). Digital Elliptic Filter Design 13.3 Digital Elliptic Filter Design. https://community.ptc.com/sejnu66972/attachments/sejnu66972/PTCMathcad/176202/1/13.3 Digital Elliptic Filter Design.pdf.

[10] Getu, B. N. (2020, November). Digital IIR Filter Design using Bilinear Transformation in MATLAB. In *2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)* (pp. 1-6). IEEE.

[11] Zhang, D., Chen, L., & Wu, Y. (2020, November). Research on High Precision FFT Algorithm Based on FPGA. In *2020 5th International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)* (pp. 42-46). IEEE.

[12] Savran, I. (2017). Donanım Tanımlama Dili VHDL ve FPGA Uygulamaları. İstanbul: PapatyaBilim.

[13] Owen, J., & Henry, M. (2018, October). 384 TMAC/s FIR filtering on an Artix-7 FPGA using Prism signal processing. In *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society* (pp. 2659-2664). IEEE.

[14] Samanta, S., & Chakraborty, M. (2014, March). FPGA based implementation of high speed tunable notch filter using pipelining and unfolding. In *2014 Twentieth National Conference on Communications (NCC)* (pp. 1-6). IEEE.

[15] Singh, R., & Arya, S. K. (2012). Genetic algorithm for the design of Optimal IIR Digital filters.