www.czasopisma.pan.pl PAN www.journals.pan.pl

jet INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2022, VOL. 68, NO. 1, PP. 19–26
Manuscript received September 3, 2021; revised January, 2022. DOI: 10.24425/ijet.2022.139843

# Analysis of a Novel FPGA-based System for Filtering Audio Signals Using a Finite Impulse Response Filters

Adrian Lipowski, Paweł Majewski and Sławomir Pluta

***Abstract*—In this article, an analysis of an innovative system for filtering signals in the audible range (16 Hz - 20 kHz) on programmable logic devices using a filters with a finite impulse response, is presented. Mentioned system was neat combination of software and hardware platform, where in the program layer a multiple programming languages including VHDL, JavaScript, Matlab or HTML were used to create completely useful application. To determine the coefficients of polynomial filters the Matlab Filter Design & Analysis Tool was used. Thanks to the developed graphic layer, a user-friendly interface was created, which allows easily transfer the required coefficients from the computer to the executive system. The practical implementation made on the FPGA platform, specifically on the Altera DE2-115 development kit with the FPGA Cyclone IV, was compared with simulation realization of Matlab FIR filters. The performed research confirm the effectiveness of filtration in real time with up to 128th order of the filter for both audio channels simultaneously in FPGA-based system.**

***Keywords*—audio filtering; FIR filter; FPGA; signal processing**

## I. Introduction

A STANDARD approach to signal processing tasks is to use the DSPs [1]–[6]. This structures are characterized by the possibility to work in real-time systems and, depend on model, can implement different kinds of memories, peripherals, interfaces etc. Nowadays, they can even vary by structure of core, from single to multicore approach. Thanks to its universal possibilities they can be used in almost any scientific field. Nevertheless, sometimes the demands that are made of them can inflate part of the parameters of such a system. Therefore, in some cases, simple system may need a processor with very high clock speed, where another parameters are not so critical. Notwithstanding, instead of struggling with these parameters is better to use another alternative approach. At present, thanks to high complexity of PLDs, the FPGA chips can be a such solution [7]–[12]. The internal physical structure of such a system can perform a different specific task. Maybe these structures are not flexible as DSPs, but in some cases they can be full-fledged counterparts to their opponents. Moreover, bringing the functionality almost directly to the physical layer we significantly speeds up the computations what makes FPGAs so attractive [13]–[20]. What is more, the PLDs work same as CPUs according to clock tact, but first of them do not have as many abstract areas, so they no need to fetch code from memory, because algorithm is already implemented in hardware. This speaks in favor of FPGAs, so the authors of this article paid particular attention to them.

One of mentioned FPGAs functionality can be a real-time signal filtration process, especially with using filter with finite impulse response (FIR). Thanks to the convolutional equations describing this tool, the process of its implementation reduce to perform only a certain number of multiplication and addition operations. Through the carried research, the authors propose to realized signal processing of audio frequencies by using high order of FIR filter in FPGA-based systems. A comparative analysis of the work of a real object with its simulation counterpart, which is aim of these article, confirms the correctness of proposed approach. Therefore, the paper has following manner. After a short introduction of mentioned filters in Section 2, the tools used in the project to perform a research studies are presented in Chapter 3. Next, the discussing of the description and configuration of the authors' developed system in Section 4 precedes the results of the analysis presented in Chapter 5. The whole article is summarized in the form of a discussion in Chapter 6. before the References.

## II. Finite Impulse Response filter

Finite Impulse Response filters belong to the group of polynomials filters, which are assumed to be easily implemented in digital systems. They allow to improve the characteristics of the processed signals in a finite impulse response, therefore they are very user-friendly. Despite its numerous advantages, one is undoubtedly the most important feature - stability. If we look at the Figure (1), the current state of the output depends only on the state at the input at the present and the past. Therefore is no need to know the states of output now and in the past. This allows the FIR transfer function characteristic to include only zeros without poles, what results in the mentioned structural stability of the systems. Moreover, due to the appropriate selection of the filter coefficients, these systems can have a linear phase characteristic. While an implementation disadvantage is the memory requirement for storing the filter coefficients, but that does not bother us nowadays. Moreover, the implementation itself can be reflected in following expression

A. Lipowski, P. Majewski and S. Pluta are with Opole University Technology, Opole, Poland (e-mail: a.r.lipowski@gmail.com, p.majewski@po.edu.pl, s.pluta@po.edu.pl).

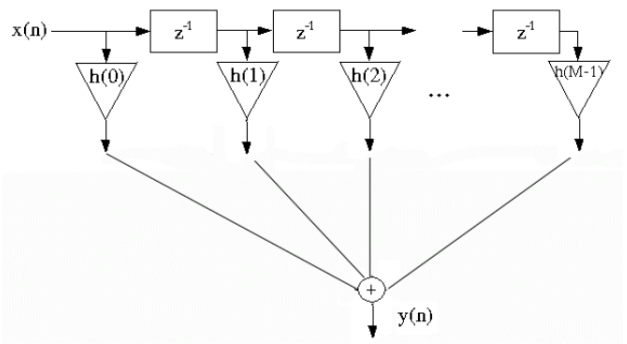



Fig. 1. Scheme of finite impulse response filter [21]

$$\mathbf{y}(n) = \sum_{k=0}^{M-1} \mathbf{h}(k)\mathbf{x}(n-k), \tag{1}$$

where $\mathbf{x}(n)$ and $\mathbf{y}(n)$ are $n$-th sample of input and output signal, respectively. While $\mathbf{h}(k)$ stands for $k$-th coefficient of the filter polynomial with $M-1$ order. Whilst the form of the filter can be written as follows

$$\mathbf{H}(z^{-1}) = \mathbf{h}_{M-1}z^{-M+1} + \mathbf{h}_{M-2}z^{-M+2} + \ldots + \mathbf{h}_1 z^{-1} + \mathbf{h}_0, \tag{2}$$

where $z^{-1}$ stands for one step backward shift operator.

As we can see in Eq. (1) implementation of the FIR require a convolution operation, hence system must perform number of multiplication and additions, what is not problem for FPGAs (even for high order $M-1$).

Therefore, if we have an outlined tool that we will use in the further part of our article, let's move on to the chapter describing the platform, specifically the hardware and software used in the research.

## III. Hardware and Software Used in the Research Studies

For the purpose of research studies on the implementation of filtration with the FIR approach, it is necessary to compose a set of appropriate tools. The whole system consists of several software and hardware key components, therefore they will be described separately.

### A. Altera DE2-115 development kit

In order to the implementation of the convolution function described by the equation Eq. (1) in hardware, the Altera DE2-115 with FPGA Cyclone IV development kit was used. This platform, apart from the chip that can work with frequencies up to 400 MHz (50 MHz clock speed was used in the research), contains numerous useful functions such as the audio codec system with 3.5 mm jack input/output or numerous peripherals from GPIO to serial communication. The block diagram of the used system is shown in Figure (2).

### B. Converter USB/UART

Although the previously presented FPGA platform can be programmed via USB, the UART communication was used for communication with a PC. Therefore, a USB/UART converter, the PL2303HX model was used in the developed system (Figure (3)). This approach guarantee, a smooth and simple connection between board and the station that displays the results i.e. PC computer.

### C. Generating and measuring equipment

The waveforms necessary to test the system were made with a Siglent SDG1025 functional generator with a base band of 25 MHz. Mentioned device allows to generate periodic signals of a selected shape (including a sine wave) and a specific amplitude. It also has a "sweep" option, which allows to automatically change the frequency of the generated signal in a given way. The generated signal was fed to the input of the audio processing line of FPGA platform.

To analyze the operation of the digital processing path the Siglent SDS 1072CML oscilloscope with a band of 70 MHz was used, which is sufficient for recording audio signals. Both the function generator output and the filtered signal audio output were connected to the oscilloscope channels. Thanks to this, it was possible to observe time-correlated signals in the time domain. It will also allow to determine the static characteristics of the processing path for the selected frequencies. The oscilloscope also has the FFT function, the use of which on the filtered signal allows to determine the effectiveness of the proposed filter.

### D. Software

For the purposes of programming the FPGA, the Quartus Prime environment was used, which, apart from the possibility of interpreting the VHDL language, allows to configure the inputs/outputs of the system and download the result code to the chip (Figure (4(a))).

The next key tools from the point of view of the research are the *Node.js* environment and an Internet browser. The first of them is responsible for managing operations in the background (backend), and together with the browser with WebSocket support, we can have a fully functional interface.The role of this system is to manage the window and FIR filter coefficients and it consists of several layers (Figure (5)). Directly to the FPGA is connected the TX line of the computer's serial connector. This connector is realized via a USB/UART converter. As part of this, the server application was launched in the Node.js environment. It handles the serial port and passes data between it and the instance of the WebSocket object. A web browser that supports the WebSocket technology can connect to the server by executing JavaScript code. Thanks to this, it is possible to transfer data between the FPGA chip and a web browser, in which the graphic layer is made up of standard HTML elements.

In the proposed approach, the filter coefficients previously determined in Matlab are read by JavaScript code and can be edited within the browser. Moreover, the user can select a specific type of filter from the list of previously prepared sets

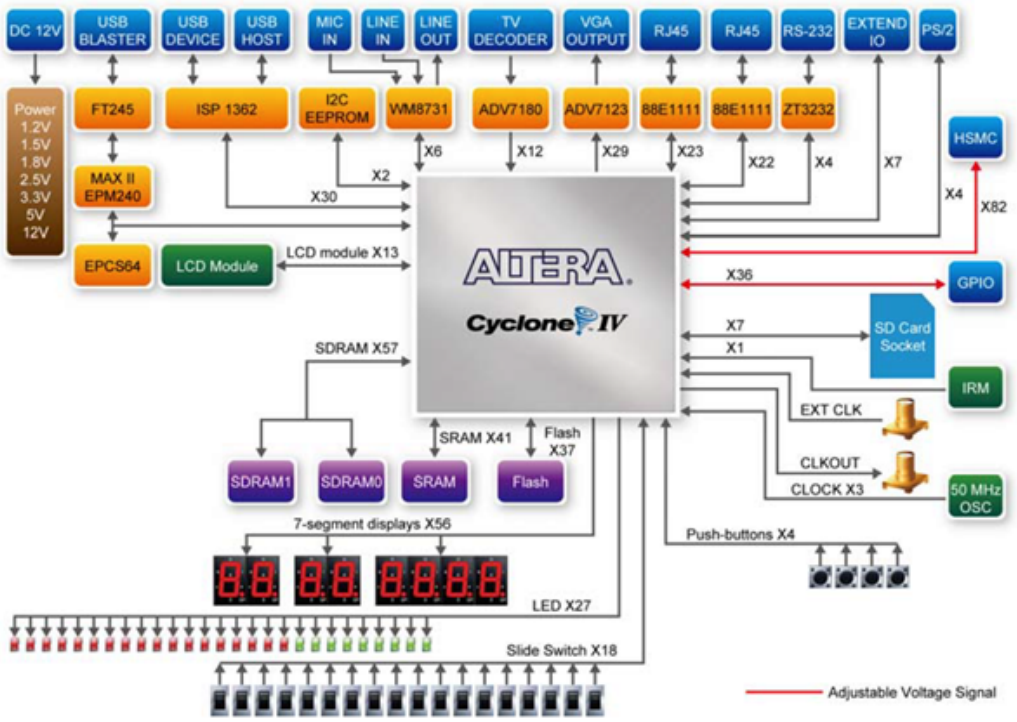



Fig. 2. Block scheme of DE2-115 platform [22]

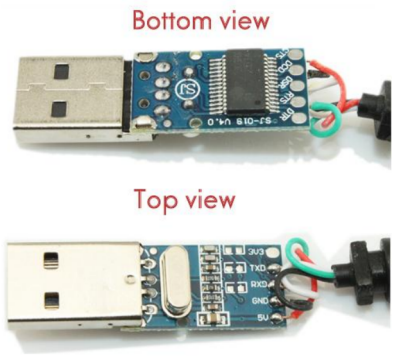



Fig. 3. Converter USB/UART PL2303HX [23]

(Figure (6(a))), and can also edit it graphically with the use of active sliders (Figure (6(b))). The set values can be sent to the layout as window or filter coefficients.

Last but not least is Matlab environment. Thanks to this software, especially thanks to the Filter Designer & Analysis Tool, it is possible to design and verify the proposed filtering algorithm and of course receive the required filter coefficients (Figure (4(b)).

## IV. Description and Configuration of the Research System

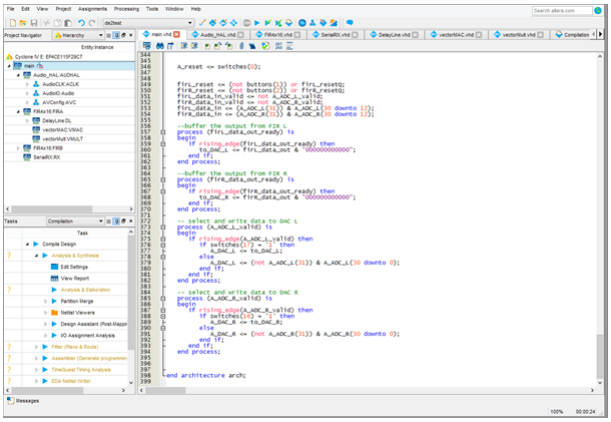
In order to perform the research studies, the laboratory system was prepared from earlier mentioned hardware and software components. However, before the research was made, some setup remarks should be considered. Due to its complexity each of them will be taking into account separately.

### A. Audio interface operating module

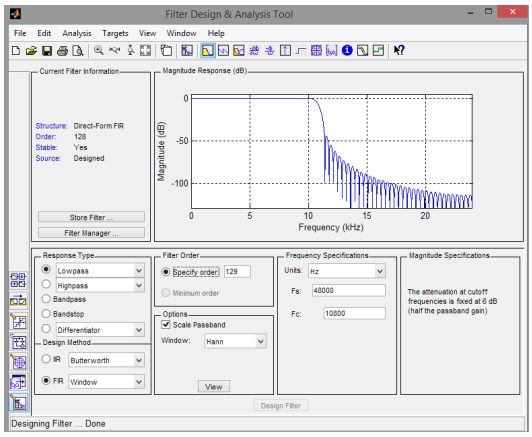
In the hardware part of the whole system we used audio codec of the platform which has a microphone input and line input/output, as well as communication lines and buses. The connections between the codec and FPGA, was made permanently in the DE2-115 development kit. But from the point of view of the software, it can be seen that thanks to the IP modules, many configuration tasks can be performed automatically, but not all of them. Thanks to the mentioned modules provided by the manufacturer of the development kit, the codec system could be configured by automatically generating the VHDL code. For example block named *Audio and Video Config* [24] provides communication with the codec module necessary for its configuration.

The audio input path has been configured to send the signal from the Line In to the ADC. The ”Enable DAC Output” option was also selected to allow the samples to be written to the DAC and the audio signal generated to the Audio Out, simultaneously. The audio samples (both from the ADC and to the DAC) were recorded on 32 bits with left justification (MSB at position 31).

Receiving and storing samples over physically available buses require the creation of additional coding/decoding modules and appropriate buffers. But in the Quartus environment, we can find an IP block that is an interface to the audio codec (Figure (7)). Using this module allows to gain access to the ADC and DAC in a relatively simple, high-level way. The module is physically connected to the audio codec via a data bus and clock lines. Thanks to the use of the *AudioIO* module, we obtain 2 input (ADC) and 2 output (DAC) buses with *valid*



(a) Main window of the Quartus Prime




(b) Filter designer in Matlab

Fig. 4. Software used in research

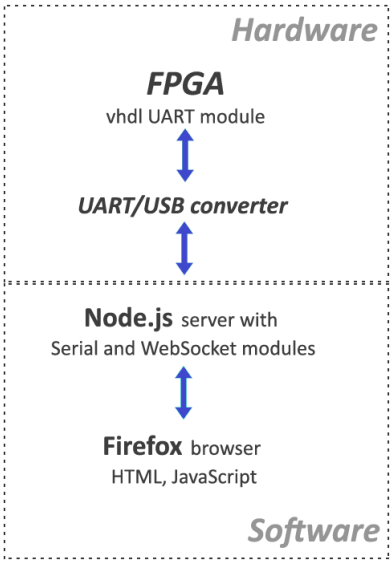



Fig. 5. System interface diagram

and *ready* signals, which allows for simple, both synchronous and asynchronous, data exchange with the codec.

The last element needed both to configure the codec and to work with the *AudioIO* module is the appropriate clock signal. According to the information included in the catalog note, a clock frequency of $256 * f_s$ should be used for the sampling frequency $f_s$. This approach provides the IP *AudioCLK* module which can generate clock signals at the frequencies needed for the codec system operation.

The code of the above-mentioned components was used to develop a authors' single module *Audio HAL* constituting an abstraction layer for the audio codec. It allows to easily configure the codec, read and write the audio signal directly with a single component, what can be see in the Figure (8).

### B. FIR module

The filter module has been designed to have as less inputs/outputs as possible and to be easy to use. The parallel execution of 128 multiplication operations on 20-bit numbers was divided into eight steps. In each of them, the 16 consecutive sample values are calculated. The subtotal is kept, and after all steps have been completed, the total result represents the filter output. The block diagram of the module (Figure (9)) presents the data flow between the components.

The filter module logic was implemented as a state machine [25]. At the beginning the machine first applies the window function to the audio samples. Then the convolution operation is performed, which takes eight iterations. Then, data from the respective input bus portions is entered into the computing components. After the result is obtained, it is accumulated in the special register.

In *Delay line* component which can be seen in (Figure (9)) shift register function is performed. The *Shift cells* create a buffer for storing audio samples and only two clock cycles are needed to perform the shift operation. In the tests, the data bus width was assumed to be 20 bits.

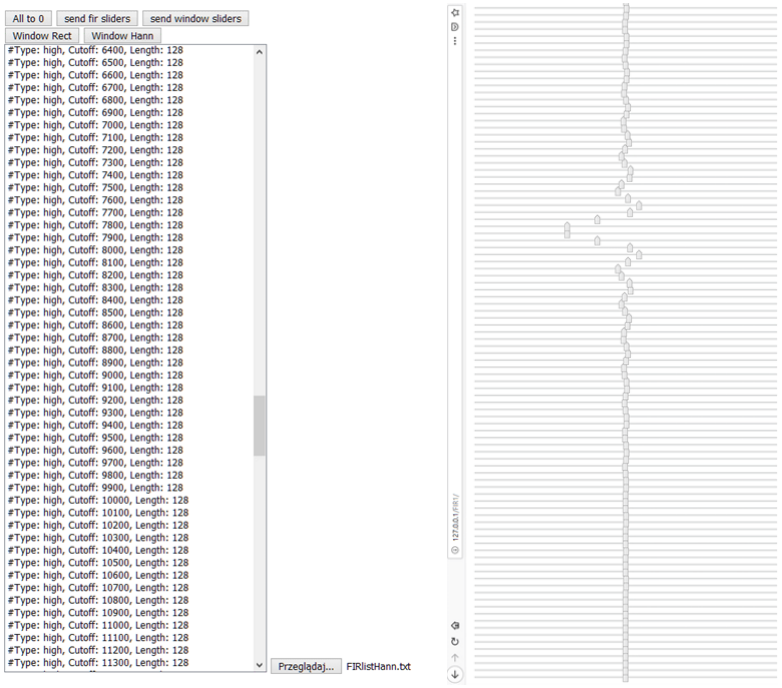
Thanks to the modular structure of the *Shift cell* elements, it is possible to freely configure the length of the Delay line block by adding the mentioned blocks. Moreover, regardless of the delay block length adopted, each time the action associated with the shift of the audio samples in the memory takes place in only two steps of the clock cycle.

### C. Serial interface module

In order to implement the UART link, modules supporting serial transmission were created, where the RX and TX lines were connected directly to the FPGA. For this approach, communication was handled with special values of the *bitTicks* argument (the generic variable defines the number of master clock cycles per bit transmission time), so that the signal value could be correctly read at appropriate link mode. For the project, a bandwidth of 115 200 bps was selected, which means a transmission of 14,4 KB/s, through which 3-byte packets of filter coefficients were sent. This means that the parameter package for both channels was 128*2*3 = 768 bytes.

### D. Main module

The task of the main module, the highest in the hierarchy, is to collect and connect the sub-segments among themselves,




(a) List of filters in the user interface

(b) Sliders for changing the filter coefficients

Fig. 6. Partial view of the user interface

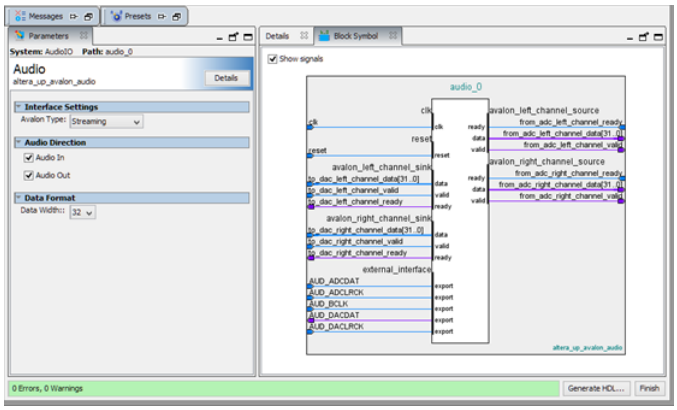



Fig. 7. *AudioIO* IP block

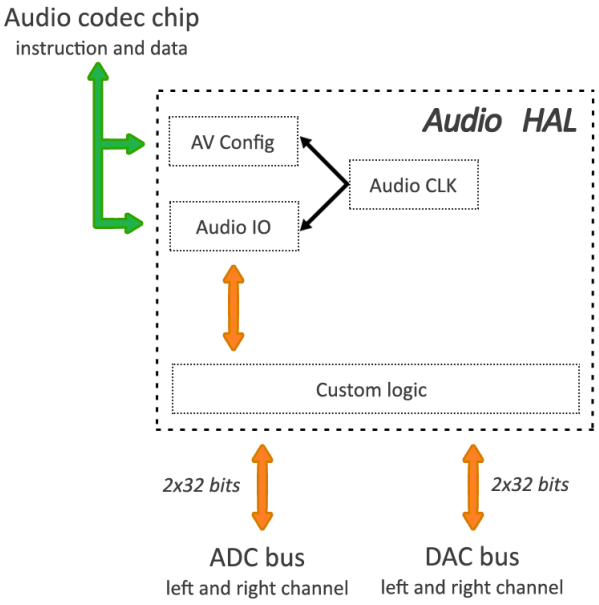



Fig. 8. Block diagram of the VHDL author's *Audio HAL* module

as well as to describe the signals that can be led out. Only from this unit, signals can be assigned to the physical pins of the FPGA. All the previously described components have been integrated into the main module. In the mentioned system there is also a logic which realizes buffering of coefficients sent via UART link. The bytes received over the serial link are sequentially folded into 20-bit coefficients and written to the window buffer or FIR filter.

### *E. Generation of FIR coefficients in Matlab*

The practical system verification required the generation of correct coefficients for the FIR filter. For this purpose, a script was developed in the Matlab environment, which, using the default function "fir1(.)", generated a text file containing a number of filters with given parameters. The arguments of the mentioned function are filter type, order and bandwidth, as well as additional metadata containing basic information about filters.

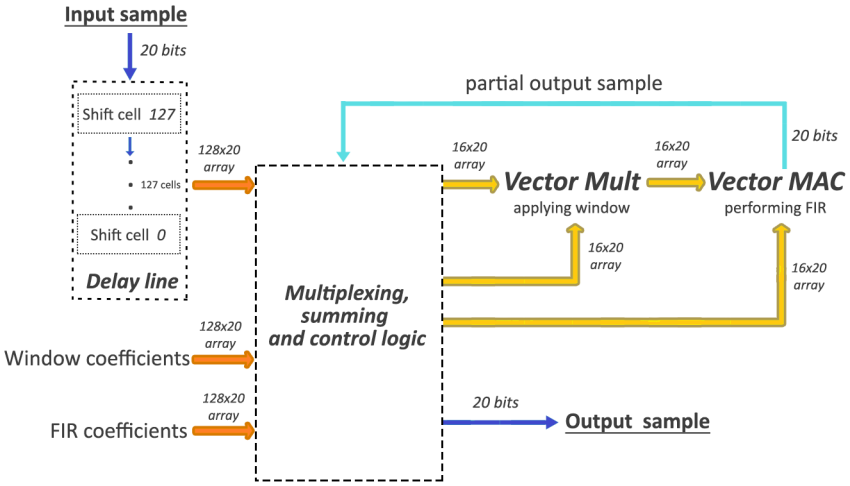



Fig. 9. Block diagram of the VHDL FIR module

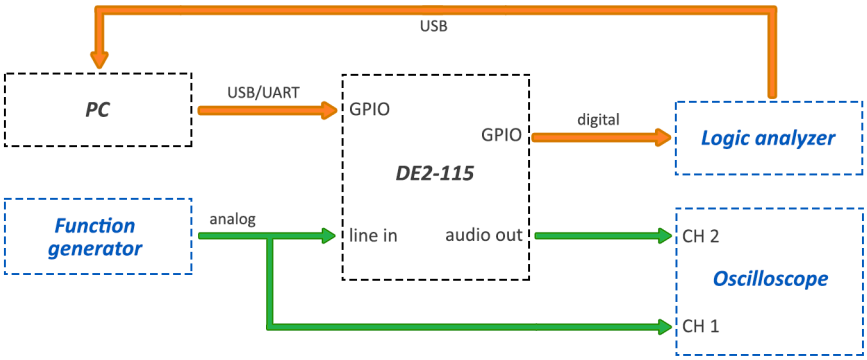



Fig. 10. Block diagram of the test system

## V. Experimental results and analysis

The performance of the practical system was verify by the devices/tools mentioned in the previous sections. The measuring set was connected as shown in Figure (10), where the output of the generator was connected to both the audio input of the DE2-115 set and the input of the oscilloscope. Logic analyzer channels was also connected to the appropriate pins of the GPIO port of the FPGA. Therefore, the analysis of the authors' FPGA-based system comes down to comparing the physically obtained results with those received from the simulation of appropriate FIR filters.

The results of the analysis of the authors' system in terms of frequency responses are presented in the further part of this section.

### A. Frequency analysis

To perform the frequency analysis of the developed system the functional generator and a digital oscilloscope, described in previous section, were used. Both the signal fed to the input of the processing board and the signal after the filtration process were observed on the oscilloscope. By displaying both of them at the same time, it was possible to measure their amplitudes and calculate the phase shift. In addition, the Fast Fourier Transform (FFT) was also performed, which made it easy to observe how the filter's frequency response changed due to the changes of it's coefficients. In the presented studies, dBVrms was adopted as the measurement unit (Figure (11)).

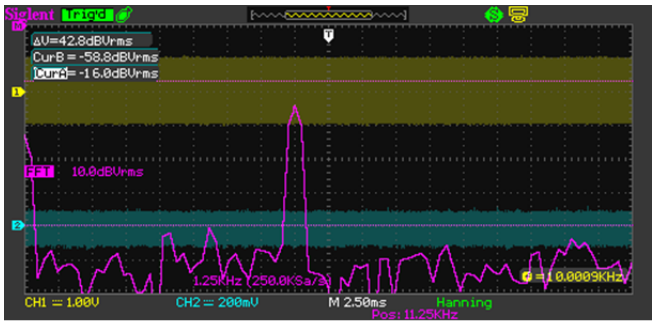
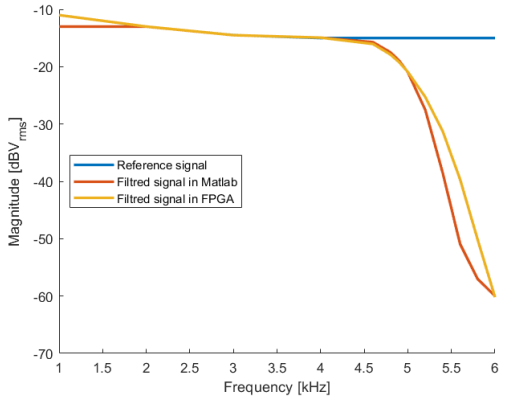
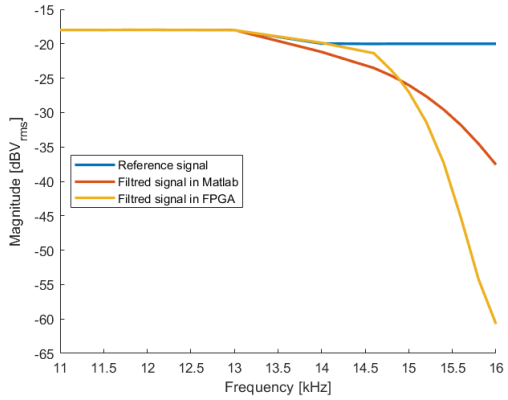



Fig. 11. FFT analysis

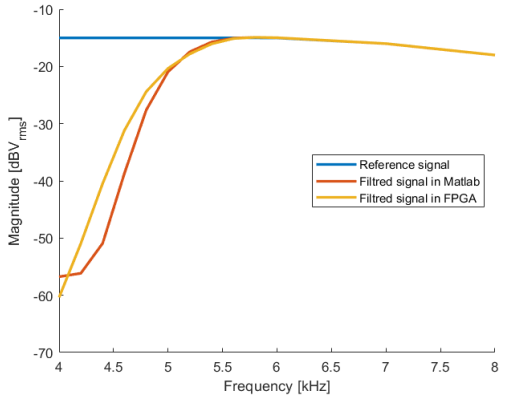
During the measurement studies, it was found that the noise level was on ca. -60 dBVrms. Moreover, the tested physical low- and high-pass filters reduced signal to the mentioned noise level at a distance of 1 kHz and further from the cut-off frequency (please see Figures (12(a)-12(d))). What is more, in case of these filters for high frequency (here was 15 kHz)



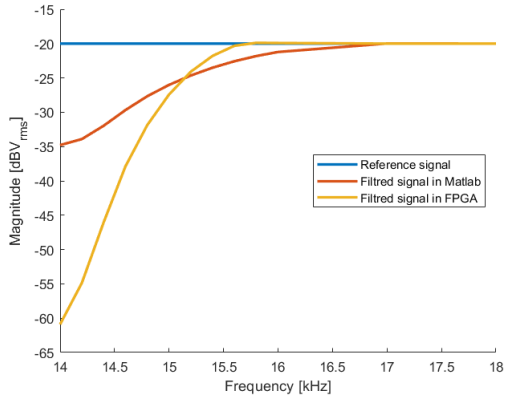

(a) Characteristic of the 5 kHz low-pass filter



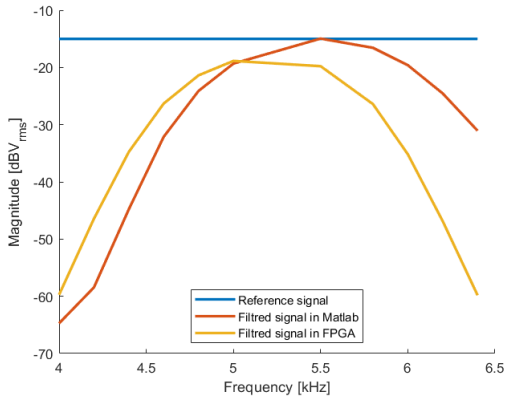

(b) Characteristic of the 15 kHz low-pass filter



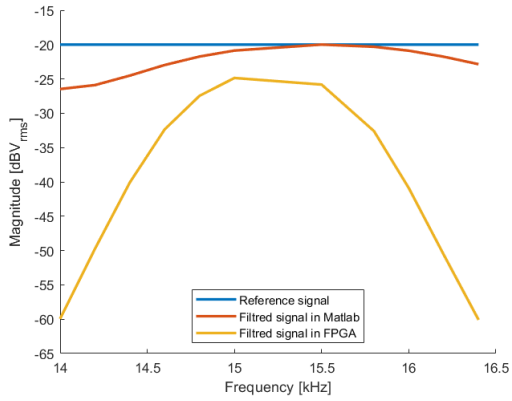

(c) Characteristic of the 5 kHz high-pass filter




(d) Characteristic of the 15 kHz high-pass filter




(e) Characteristic of the 5-6 kHz band-pass filter




(f) Characteristic of the 15-16 kHz band-pass filter

Fig. 12. Amplitude characteristics of selected ilters

the attenuation of signal in the stop band is much better in authors' develop system than in reference simulation studies (Figures (12(b)-12(d))).

While for the physical band-pass filters the response was obtained with a slightly higher slope on the high-frequency side and a smaller slope on the lower-frequency direction (Figure (12(c)-12(f))). Also in this case, the attenuation to the noise level in the stop band, was obtained not more than 1 kHz from the cut-off frequencies. For authors' 15-16 kHz band-pass filter the fading results in stop band are far more better than Matlab simulation studies. Nevertheless, we can see a slightly higher signal attenuation in the pass band (please see Figure (12(f))).

## VI. Discussion

The managed studies and analysis confirm the effectiveness of the developed system that met the assumed requirements. Authors' digital audio signal processing module was created

using the FPGA Altera DE2-115 development kit. The frequency response of the system, resulting from the adopted sampling frequency of 48kHz, was 23kHz (the requirement of the Nyquist criterion), therefore it is sufficient for filtering audio signals with FIR type filters. Practical analysis confirm the effectiveness of filtration process in real time with up to 128th order of the filters. The developed authors' user interface allows not only to select from a list and send the previously generated filter coefficients to the FPGA, but also to edit them. This approach makes it possible to generate individual filter configurations. Based on the time analysis, it can be seen that the signal sample processing time is less than half the period of feeding subsequent samples to the processing system. This means that the amount of computation can be at least doubled while maintaining real-time processing capability. The surplus time can also be used to increase the filter order, but this involves minor modifications to the VHDL code. As part of the work, a two-channel filter was implemented, so it is possible to modify the interface so that the FIR coefficients sent to the FPGA can be different for individual channels.

## REFERENCES


[1] E. Salgado-Plasencia, R.V. Carrillo-Serrano, M. Toledano-Ayala, "Development of a DSP Microcontroller-Based Fuzzy Logic Controller for Heliostat Orientation Control", *Applied Sciences* 10 (5), 1598 (2020). https://doi.org/10.3390/app10051598

[2] X. Gong, Z. Le, H. Wang, Y. Wu, "Study on the Moving Target Tracking Based on Vision DSP", *Sensors* 20 (11), 6494 (2020). https://doi.org/10.3390/s20226494

[3] Q. Guo, Z. Dong, H. Liu, X. You, "Nonlinear Characteristics Compensation of Inverter for Low-Voltage Delta-Connected Induction Motor", *Energies* 13 (3), 590 (2020). https://doi.org/10.3390/en13030590

[4] Ch.T. Ma, Z.H. Gu, "Design and Implementation of a GaN-Based Three-Phase Active Power Filter", *Micromachines* 11 (2), 134 (2020). https://doi.org/10.3390/mi11020134

[5] S. Cuoghi, R. Mandrioli, L. Ntogramatzidis, G. Gabriele, "Multileg Interleaved Buck Converter for EV Charging: Discrete-Time Model and Direct Control Design", *Energies* 13 (2), 466 (2020). https://doi.org/10.3390/en13020466

[6] W. Yao, J. Cui, W. Yao, "Single-Phase Inverter Deadbeat Control with One-Carrier-Period Lag", *Electronics* 9 (1), 154 (2020). https://doi.org/10.3390/electronics9010154

[7] G. La Tona, M. Luna, M.C. Piazza, M. Pucci, A. Accetta, "Development of a High-Performance, FPGA-Based Virtual Anemometer for Model-Based MPPT of Wind Generators", *Electronics* 9 (1), 83Li (2020). https://doi.org/10.3390/electronics9010083

[8] X. Li, N. Wang, G. San, X. Guo, "Current Source AC-Side Clamped Inverter for Leakage Current Reduction in Grid-Connected PV System", *Electronics* 8 (11), 1296 (2019). https://doi.org/10.3390/electronics8111296

[9] B. Wang, W. Tang, "A Novel Three-Switch Z-Source SEPIC Inverter", *Electronics* 8 (2), 247 (2019). https://doi.org/10.3390/electronics8020247

[10] X. Sun, Ch.J. Xue, J. Yu, T.W. Kuo, X. Liu, "Accelerating data filtering for database using FPGA", *Journal of Systems Architecture* 114, 101908 (2021). https://doi.org/10.1016/j.sysarc.2020.101908

[11] R. Guo, "Strength Fitness Control System and Motor balance Based on FPGA and Wireless Sensors", *Microprocessors and Microsystems* 81, 103684 (2021). https://doi.org/10.1016/j.micpro.2020.103684

[12] S. Kim, U. Yun, J. Jang, G. Seo, J. Kang, H.N. Lee, M. Lee, "Reduced Computational Complexity Orthogonal Matching Pursuit Using a Novel Partitioned Inversion Technique for Compressive Sensing", *Electronics* 7 (9), 206 (2018). https://doi.org/10.3390/electronics7090206

[13] A. Lipowski, "Developing DSP techniques in FPGA systems (in Polish)", *Bachelor's Thesis, Opole University of Technology* (2019).

[14] M. Skiwski, "Cyfrowa filtracja sygnałów z wykorzystaniem układów FPGA", *Pomiary Automatyka Kontrola* 59 (6), 503–506 (2013).

[15] C.J. Kikkert, "A Phasor Measurement Unit Algorithm Using IIR Filters for FPGA Implementation", *Electronics* 8 (12), 1523 (2019). https://doi.org/10.3390/electronics8121523

[16] F. Nekoei, Y.S. Kavian, O. Strobel, "Some schemes of realization digital FIR filters on FPGA for communication applications", *In the proceedings of 20th International Crimean Conference "Microwave Telecommunication Technology"*, 616–619 (2010). https://doi.org/10.1109/CRMICO.2010.5632348

[17] R.R. Sudharsan, "Synthesis of FIR Filter using ADC-DAC: A FPGA Implementation", *In the proceedings of IEEE International Conference on Clean Energy and Energy Efficient Electronics Circuit for Sustainable Development (INCCES)*, 1–3 (2019). https://doi.org/10.1109/INCCES47820.2019.9167696

[18] H.S.O. Migdadi, R,A. Abd-Alhameed, H.A. Obeidat, J.M. Noras, E.A.A. Qaralleh, M.J. Ngala, "FIR implementation on FPGA: Investigate the FIR order on SDA and PDA algorithms", *In the proceedings of Internet Technologies and Applications (ITA)*, 417–421 (2015). https://doi.org/10.1109/ITechA.2015.7317439

[19] D. Datta, S. Akhtar, H.S. Dutta, "FPGA Implementation of Symmetric Systolic FIR Filter using Multi-channel Technique", *In the proceedings of IEEE VLSI DEVICE CIRCUIT AND SYSTEM (VLSI DCS)*, 225–228 (2020). https://doi.org/10.1109/VLSIDCS47293.2020.9179926

[20] M.M. Shahbaz, A. Wakeel, Junaid-ur-Rehman, B. Khan, "FPGA Based Implementation of FIR Filter for FOFDM Waveform", *In the proceedings of 2nd International Conference on Communication, Computing and Digital systems (C-CODE)*, 226–230 (2019). https://doi.org/10.1109/C-CODE.2019.868100510.1109/VLSIDCS47293.2020.9179926

[21] M. Krzysiek, "Digital filters realizations for TMS320 signal processors", *Master Thesis, Wrocław University of Science and Technology*, (2006).

[22] Intel/Altera, "Terasic DE2-115 User manual", *Technical data sheet*, (2013).

[23] Handson Technology, "Intel Audio Core for Intel DE Series Boards", *User guide*, (2020).

[24] Intel, "Intel Audio Core for Intel DE Series Boards", *Technical data sheet*, (2020).

[25] T. Łuba, "Synthesis of digital circuits (in Polish)", Wydawnictwa Komunikacji i Lacznosci, (2003).