iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VII **Month of publication:** July 2026

DOI: https://doi.org/10.22214/ijraset.2026.84490

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

Design and Hardware Implementation of Audio Filtering

Kandukuri Harika[1], Dr. M. Madhavi Latha[2]
[1]*M.Tech in Systems and Signal Processing, Department of Electronics and Communication Engineering,*
[2]*JNTUH University College of Engineering, Science & Technology Hyderabad, Telangana, India*

Abstract: ***In order to isolate, amplify, and analyse particular frequency components within audio signals, audio filtering is a crucial step in digital signal processing. In this study, a real-time audio filtering system that uses several Finite Impulse Response (FIR) filters on a System-on-a-Chip (SoC) platform is designed and hardware realised. Four working modes are included in the proposed system: band-pass filtering, low-pass filtering, high-pass filtering, and direct signal transmission. Using the HDL Coder methodology, the filter architectures are created in MATLAB Simulink and automatically converted into hardware descriptions that can be manufactured. Real-time switching between the different filtering modes is made possible by a changeable intellectual property (IP) core, which enables the dynamic selection of the appropriate filter response without the need for hardware redesign or reconfiguration. System performance is evaluated using stereo audio signals sampled at 44.1 kHz. In order to guarantee compatibility with common audio processing needs, the filter coefficients are simultaneously constructed based on a reference sampling frequency of 48 kHz. The outcomes of the experiment show that unwanted frequency components can be effectively attenuated while maintaining the desired audio content's quality. For real-time embedded audio processing applications on SoC platforms, the implementation validates the applicability of FIR-based filtering approaches.***
Keywords: Audio Filtering, FIR Filter, Zynq SoC, FPGA, Simulink, HDL Coder.

I. INTRODUCTION

High-quality audio transmission and processing are critical in contemporary multimedia, communication, healthcare, and entertainment systems. Unwanted noise and undesirable frequency components are common in audio communications, which deteriorate sound quality. In order to improve audio clarity and transmission efficiency, digital filtering is frequently employed to eliminate these undesirable elements while maintaining the desired information. Finite Impulse Response (FIR) filters are widely employed among digital filters due to their predictable performance, linear phase response, and intrinsic stability. Low Pass Filters (LPF), High Pass Filters (HPF), and Band Pass Filters (BPF) are three types of FIR filters that are used in many frequency-selective applications, including communication systems, speech improvement, noise reduction, and music processing. Digital FIR filters are more accurate, flexible, and simple to use than traditional analog filters. Software-only methods struggle to attain the high computational speed and low processing latency needed for real-time audio processing. Field Programmable Gate Arrays (FPGAs) use deterministic execution and parallel processing to offer an effective hardware platform. The Xilinx Zynq-7000 System-on-Chip (SoC) enables effective hardware-software co-design for real-time signal processing applications by combining programmable FPGA logic with a dual-core ARM CPU. The design and construction of a real-time audio filtering system utilizing FIR filters on the Xilinx Zynq-7000 SoC platform are shown in this project. LPF, HPF, and BPF are the three filtering modes supported by the system. Real-time filtered audio is produced by applying the appropriate FIR filter coefficients to the incoming audio stream based on the mode that has been chosen. The AXI4-Stream interface allows for continuous audio data transfer, while the AXI4-Lite interface is utilized for system configuration and filter selection. MATLAB Simulink and HDL-based methods are used to develop the entire design, which is then implemented on the Xilinx Zynq-7000 SoC. The suggested architecture offers low-latency, high-performance audio filtering and functions as a versatile platform for upcoming uses such as adaptive filtering, audio equalisation, noise cancellation, and speech enhancement, according to experimental validation.

II. LITERATURE REVIEW

Jiangchun Yu et al. [1] implemented an FPGA-based FIR filter using the Xilinx FIR Compiler IP on the Zynq platform, achieving efficient harmonic suppression with reduced design complexity and reliable real-time filtering, although the implementation primarily focused on electrical measurement applications.

Anush Anand et al. [2] developed a Zynq-7000-based real-time audio processing system combining ARM processors and FPGA logic, enabling adaptive filtering with low processing delay and efficient hardware utilization for embedded audio applications. Proakis and Manolakis [3] presented the fundamental principles of digital signal processing, including FIR filter design methods, frequency analysis, and implementation techniques, providing a strong theoretical foundation for developing FPGA-based real-time audio filtering systems.

Crockett et al. [4] explained the Zynq-7000 SoC architecture by integrating ARM processors with FPGA programmable logic, highlighting hardware-software co-design and AXI communication for developing efficient embedded and real-time signal processing applications.

Crockett et al. [5] provided practical Zynq-7000 tutorials demonstrating custom IP creation, AXI interfacing, processor-FPGA communication, and system integration, offering a structured workflow for implementing reliable real-time embedded signal processing designs.

III. METHODOLOGY

The suggested system uses the SoC platform to construct a real-time audio filtering application. One of three FIR filters, Low-Pass Filter (LPF), Band-Pass Filter (BPF), or High-Pass Filter (HPF), can be dynamically applied to an audio signal by the system. The processed audio is examined and replayed in real time, and the filter selection is managed by a specialised control module. MATLAB/Simulink is used to construct the entire design, which is then implemented on SoC hardware via AXI4-Stream communication interfaces.

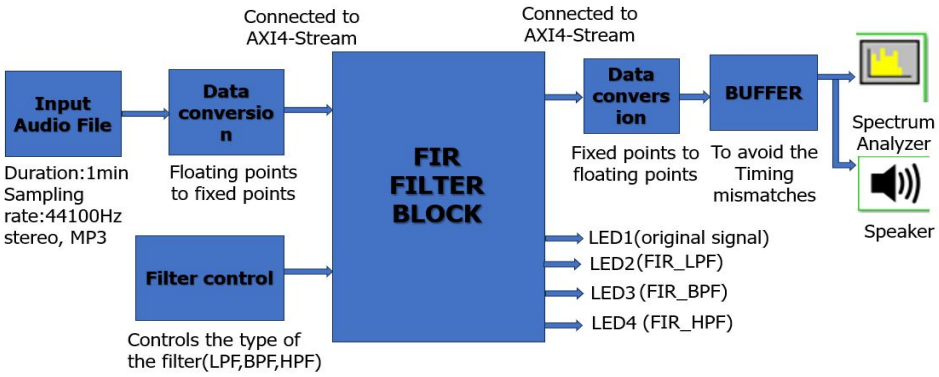


Figure 1. Block diagram of the Audio Filtering

Figure 1 illustrates The Simulink model of the suggested FIR-based audio filtering system comprises six main functional blocks: Input Audio Block, Data Conversion Block, Filter Control Block, FIR Filter Block, Output Processing Block, and Output Analysis Block. Prior to hardware implementation on the Xilinx Zynq-7000 SoC, these blocks are joined to simulate the entire audio filtering system. The model takes an input audio signal, transforms it into a format that is compatible with hardware, applies the chosen FIR filter, and then transforms the modified signal back for playback and analysis. A Spectrum Analyser and Speaker block is used to monitor the output.

A. Input Audio file

The process begins with an audio input source containing a stereo MP3 audio file sampled at 44.1 kHz with a duration of approximately one minute. This audio file serves as the test signal for evaluating the performance of different FIR filters. The input block reads the audio samples and provides them in floating-point format, which is the default numerical representation used by MATLAB during simulation.

B. Data conversion block

Since SoC hardware operates more efficiently using fixed-point arithmetic, the floating-point audio samples are converted into fixed-point format before entering the filtering stage. The conversion block transforms the double-precision data into a fixed-point representation compatible with the AXI4-Stream interface. In this design, the signal is represented using the ufix48 format, which provides sufficient precision while minimizing hardware resource utilization.

C. Filter Control block

The desired filtering action must be chosen using the filter control block. The FIR filter response that is applied to the incoming audio stream is determined by the selection signal (AXI4-Lite). The system routes the input signal through the appropriate FIR filter based on the mode that was chosen.

Accessible Modes: Original Signal, LPF or FIR Low-Pass Filter, BPF or FIR Band-Pass Filter, HPF or FIR High-Pass Filter

D. FIR filter Block

The system's primary processing unit is the FIR Filter Block. After receiving the fixed-point audio samples, it uses the chosen filter type to carry out digital filtering. The selection of Finite Impulse Response (FIR) filters is based on their predictable frequency response, linear phase properties, and intrinsic stability. The block activates the corresponding FIR filter coefficients based on the control input. Table I illustrates the Design of the digital filter specifications using the MATLAB FDA tool.

Table I: Filter design Requirements

Type of Filter	Type of Window	Normalised Cutoff Frequency	Sampling Frequency	Filter Order
LPF	Blackman	0.125	48kHz	60
HPF	Blackman	0.4792	48kHz	60
BPF	Blackman	W1=0.25, W2=0.4167	48kHz	60

E. Data Conversion block

After filtering, the audio samples are still represented in fixed-point format. However, the Spectrum Analyzer and Speaker blocks require floating-point data for proper operation. Therefore, a second conversion block transforms the fixed-point output back into floating-point format.

F. Buffer block

The Buffer block is inserted between the processing stage and output stage to synchronize data transfer rates. During simulation and hardware co-simulation, different blocks may operate at slightly different execution rates. The buffer temporarily stores incoming samples and releases them at a controlled rate, preventing timing mismatches and data loss.

G. Spectrum Analyser block

The filtered audio signal's frequency-domain properties are observed using the Spectrum Analyzer. The efficacy of each FIR filter can be confirmed by looking at the spectral response. The output signal's frequency components' magnitudes are shown by the analyzer.

H. Speaker Out block

The user can hear the effect of the chosen filter in real time by sending the final filtered signal to the speaker output block. The FPGA implementation's real-time processing capacity is demonstrated by the instantaneous changes in the auditory properties of the sound as the filter selection changes.

I. HDL Generation and SoC Implementation

The HDL Workflow Advisor available in MATLAB HDL Coder provides a structured procedure for converting a verified Simulink model into hardware suitable for FPGA and SoC deployment. Rather than manually performing multiple development stages, the workflow automates code generation, hardware integration, software interface creation, and bitstream generation.

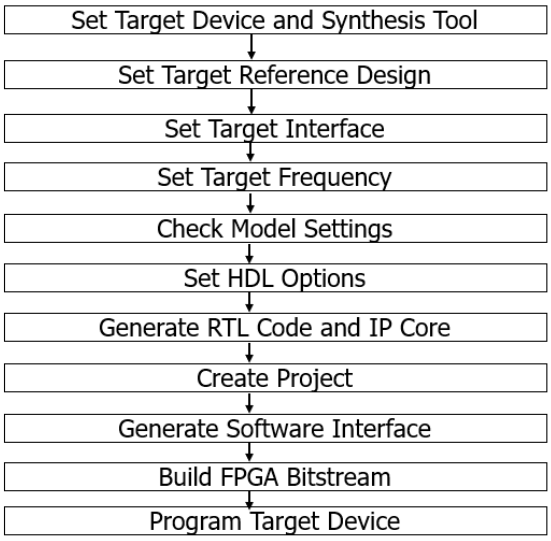


Figure 2. HDL Workflow Advisor-Based SoC Development Flow

Choosing the target hardware platform and related synthesis tools is the first step in the process. The communication between the processing system and programmable logic is then established by selecting an appropriate reference design. The model is examined to make sure it complies with HDL generating requirements after the necessary interfaces, operating frequency, and implementation characteristics have been established. Once validation is completed, HDL Coder automatically generates RTL descriptions and packages the design as an IP core. The generated hardware module is integrated into a Vivado project, where synthesis, implementation, and timing verification are performed. Software-accessible interfaces are also created to enable processor-based control of the hardware subsystem. Finally, the implementation tools generate the SoC configuration bitstream, which is transferred to the target board. This automated workflow significantly reduces development effort while ensuring compatibility between the Simulink model and the hardware platform.

The produced bitstream is programmed onto the Xilinx Zynq-7000 SoC following the successful generation of the custom FIR filter IP core by the HDL Workflow Advisor and the synthesis of the hardware design in Vivado. After that, an Ethernet cable is used to link the Xilinx Zynq-7000 SoC to the host computer, allowing MATLAB and the hardware platform to communicate. MATLAB uses the hardware support package to connect to the Zynq Processing System. MATLAB uploads the necessary settings to the board and initializes the communication interface after the connection has been confirmed. As a result, during execution, the FPGA can exchange data with the host computer and receive control orders.

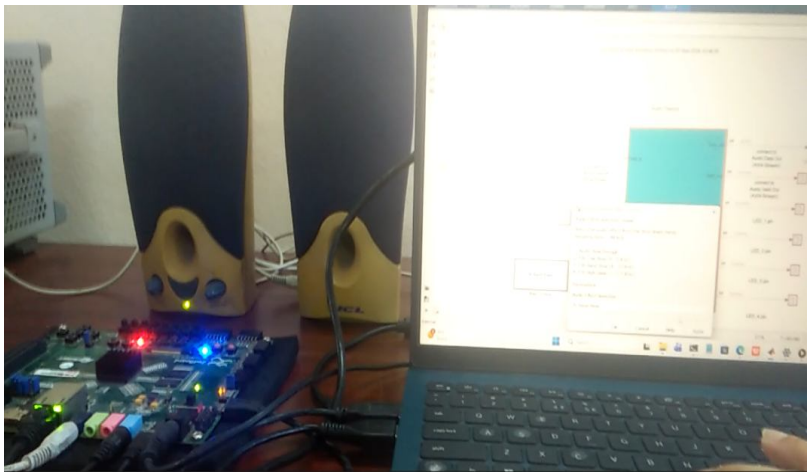


Figure 3. System to Hardware Connection Setup

IV. RESULTS AND DISCUSSION

Using MATLAB/Simulink simulation and hardware implementation on the Xilinx Zynq-7000 SoC, the audio filtering performance was assessed. Real-time audio filtering on the Xilinx Zynq-7000 SoC platform was accomplished by the developed solution. Low-pass, high-pass, and band-pass filters were developed and included in the hardware design using the AXI-Lite interface. Simulation data showed that each FIR filter exhibited the expected frequency response characteristics.

.

A. Low-pass filtered signal

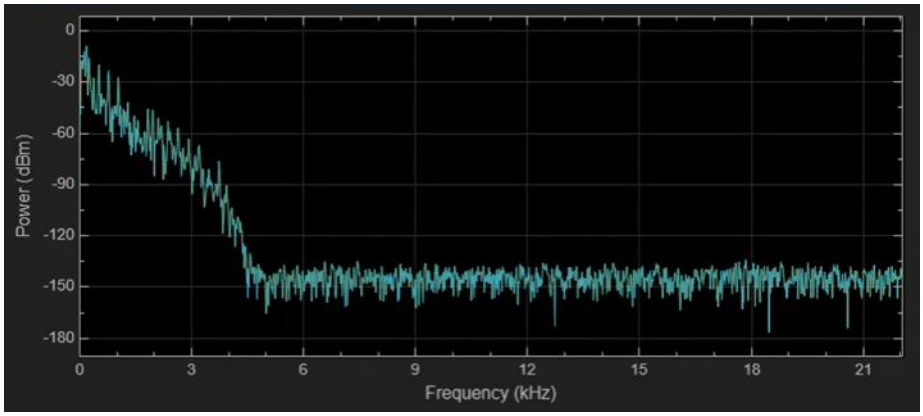


Figure 4. Low-pass filtered output (Power vs Frequency)

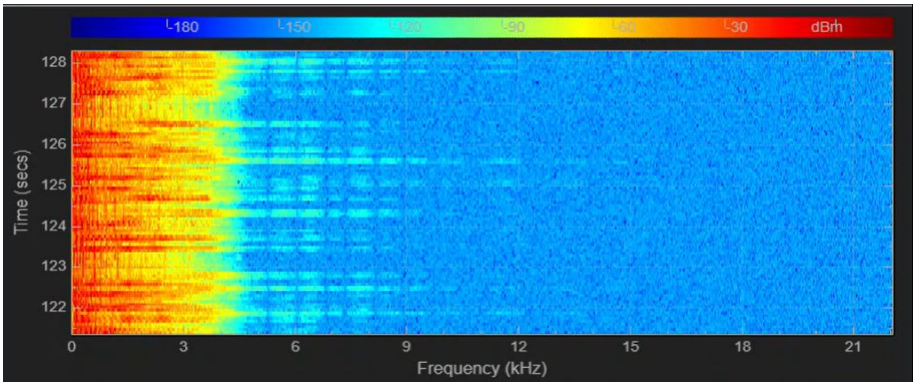


Figure 5. Spectrogram of the FIR lowpass filter

The simulation results of the intended FIR low-pass filter are displayed in Figures 4 and 5. The power spectrum shows that while frequencies outside of the cutoff range are greatly reduced, low-frequency components are kept. Additionally, the spectrogram verifies that there is very little high-frequency content and that the majority of the signal energy is centred in the low-frequency region. These findings show that the filter operates as intended and meets the design requirements found using the FDA Tool.

B. High-pass filtered signal

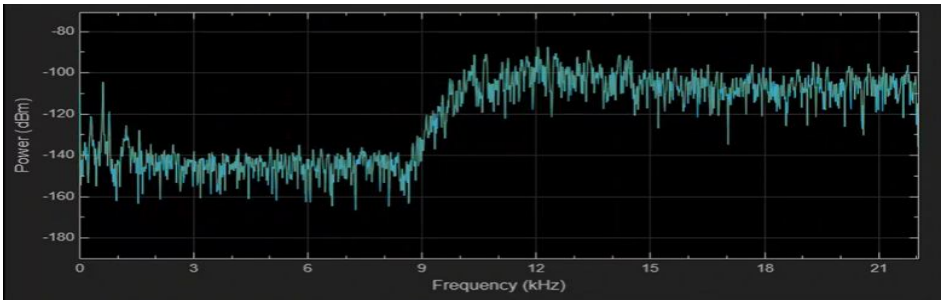


Figure 6. High-pass filtered Output (Power vs Frequency)

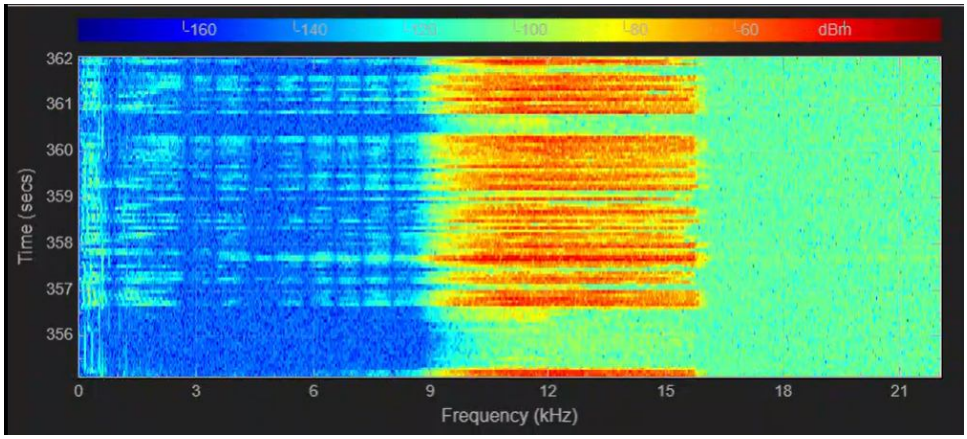


Figure 7. Spectrogram of the FIR High-pass filter

The simulation results of the intended FIR high-pass filter are displayed in Figures 6 and 7. Higher-frequency components are retained inside the passband while low-frequency components are successfully muted, according to the power spectrum. The spectrogram also demonstrates that there is very little energy at lower frequencies and that the majority of the signal energy is concentrated in the high-frequency range. These findings confirm that the filter satisfies the design requirements acquired with the FDA Tool and demonstrates the anticipated high-pass filtering properties.

C. Band-pass filtered signal

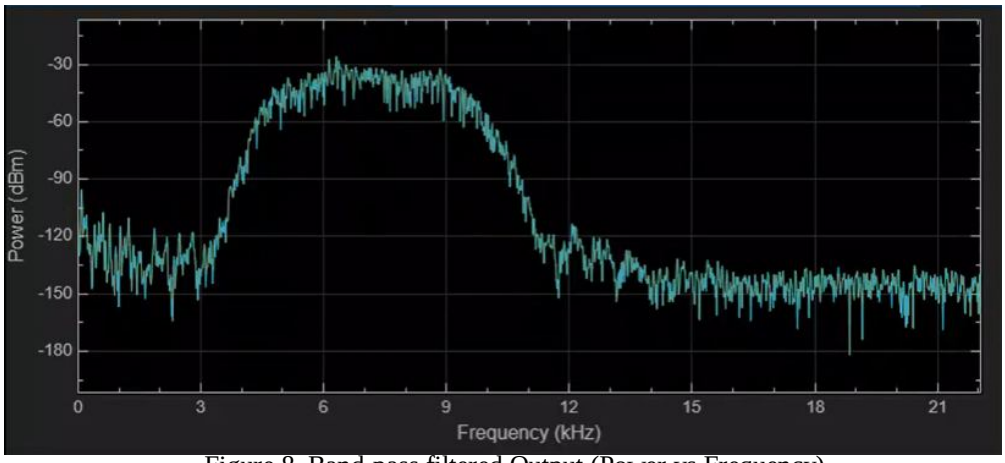


Figure 8. Band-pass filtered Output (Power vs Frequency)

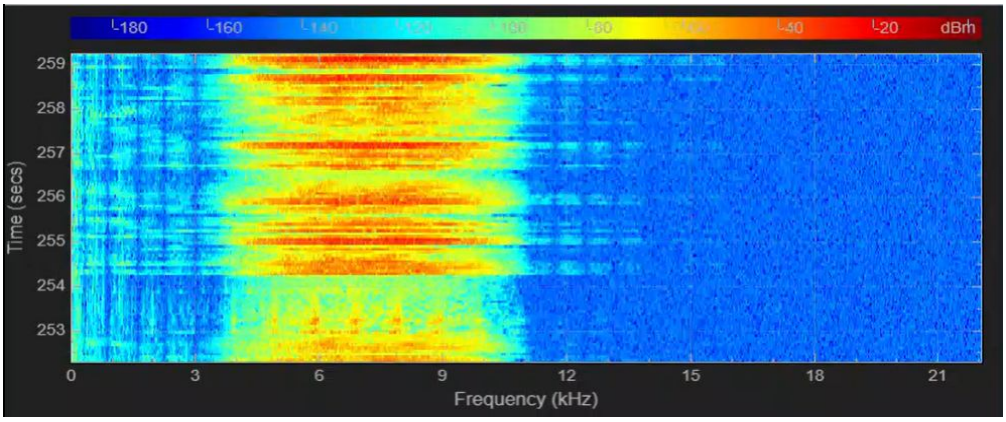


Figure 9. Spectrogram of the FIR Band-pass filter

The simulation results of the intended FIR band-pass filter are displayed in Figures 8 and 9. The power spectrum shows that while components below and above the cutoff frequencies are successfully suppressed, components within the designated passband are retained. The spectrogram further confirms that the signal energy is concentrated within the desired frequency range, with reduced energy outside the passband. These results demonstrate that the filter satisfies the design specifications obtained using the FDA Tool and exhibits the expected band-pass filtering characteristics.

The audio signal was processed continuously in real time without any discernible stoppage, according to hardware tests. The output audio successfully reflected the filter choices made using the check box, indicating that the CPU and programmable logic elements of the Xilinx Zynq-7000 SoC were communicating properly. Reliable control over the filter parameters and selection procedure was offered by the AXI-based interface.

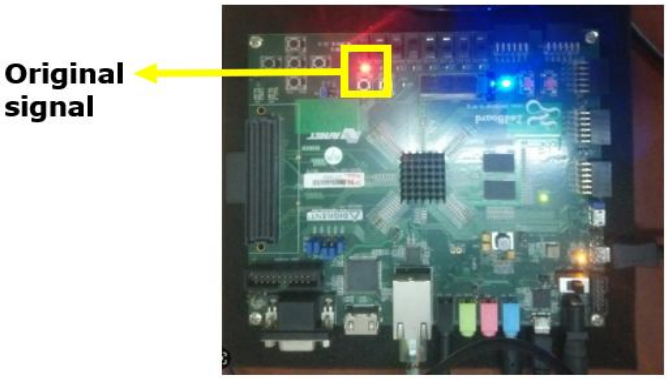


Figure 10. Original Signal Mode LED representation on board

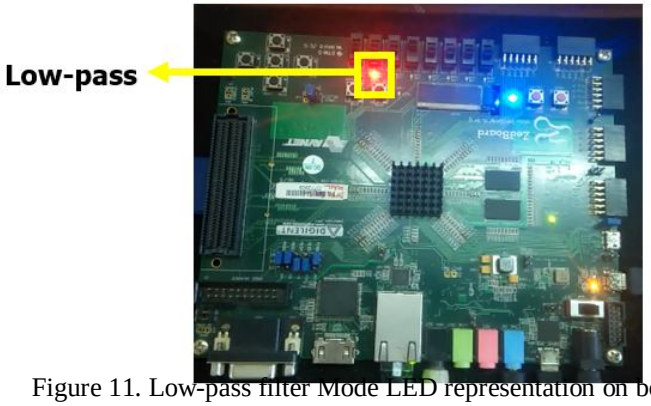


Figure 11. Low-pass filter Mode LED representation on board

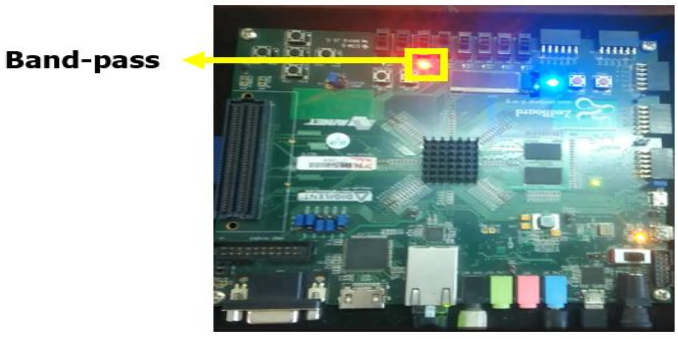


Figure 12. Band-pass filter Mode LED representation on board

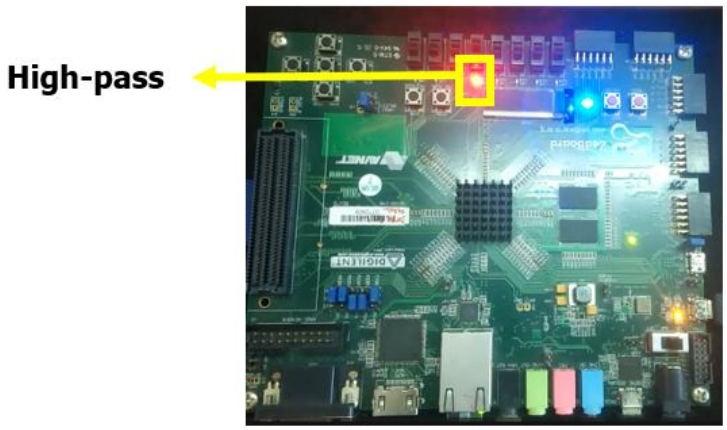


Figure 13. High-pass filter Mode LED representation on board

The developed audio filtering system's operation modes and the related LED indications are shown in Figures 10-13. The original signal mode, represented by LED 1 (Red LED), shows that the input audio is routed through the system unfiltered. When the Low-Pass Filter (LPF) mode is chosen, which attenuates higher frequencies while permitting low-frequency components to pass, LED 2 is turned on. The High-Pass Filter (HPF) mode, which preserves higher-frequency components while suppressing low-frequency ones, is shown by LED 3. Only frequencies falling inside the designated range are permitted to flow through the system while LED 4 is in the Band-Pass Filter (BPF) mode. The LED indicators offer a straightforward visual way to confirm the real-time audio filtering system's proper operation and active operating mode. The audio of each mode can be heard through the speaker.

V. CONCLUSION

The design and hardware implementation of FIR-based audio filters utilizing the SoC platform was the main emphasis of the work reported in this project. Using the MATLAB FDA Tool, low-pass, high-pass, and band-pass filters were created and incorporated into the hardware design. The developed filters met the necessary frequency-domain parameters, as confirmed by both simulation and experimental tests. Effective filter selection was made possible via the AXI-Lite interface, and the active working mode was visually confirmed by LED indications. In real-time operation, the system showed consistent performance and precise audio filtering. In general, the experiment confirmed that SoC-based FIR filters work well for digital audio signal processing applications.

VI. FUTURE WORK

By adding sophisticated signal processing methods like adaptive filtering and noise reduction, the created audio filtering system can be expanded. A wider variety of audio applications can be supported by implementing additional filter types. Design effort and development time can be decreased by reusing the FIR filter IP core produced in this research in other SoC-based systems. The MATLAB interface is used in the current design to select filters and control LEDs. The SoC's integrated push buttons and switches can be used to accomplish these control capabilities as a future improvement. This change would increase the system's versatility and allow for standalone operation.

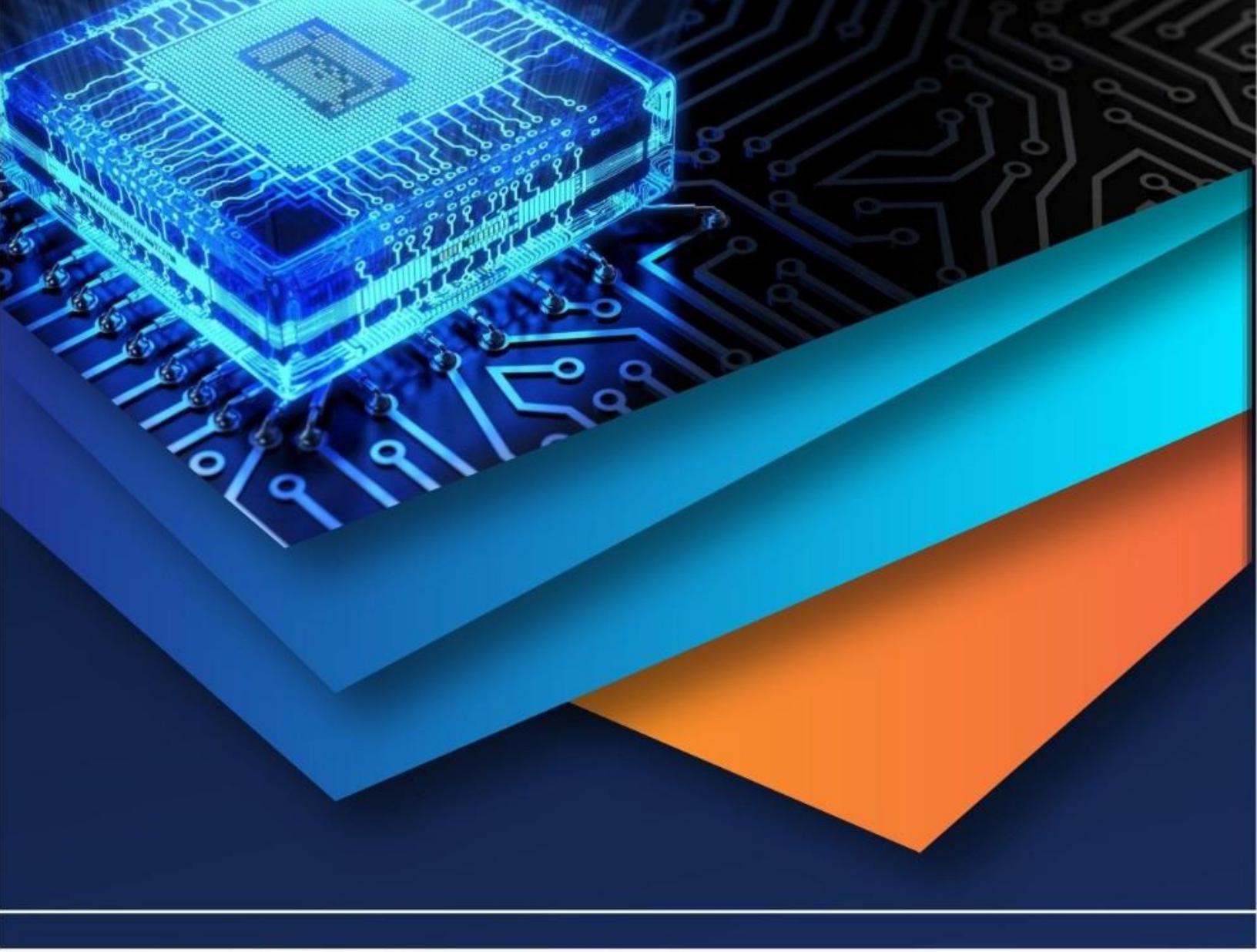
VII. ACKNOWLEDGMENT

The authors would like to thank the Department of Electronics and Communication Engineering of Jawaharlal Nehru Technological University Hyderabad (JNTUH) which provided them the resources and academic help to carry out the study. The authors are heartily thankful to Dr. M. Madhavi Latha for her expert guidance, insightful suggestions and steadfast support while developing this book. Additionally, the authors thank all of the faculty members and colleagues who helped to successfully complete this study, whether directly or indirectly.

REFERENCES

[1] J. Yu, H. Zhong and P. Yan, "Implementation of FIR filter based on Xilinx IP core," *2020 2nd International Conference on Information Technology and Computer Application (ITCA)*, Guangzhou, China, 2020, pp. 79-85, doi: 10.1109/ITCA52113.2020.00024.

[2] A. Anand and R. B, "Design and Implementation of Real-Time Audio Processing on Zynq-7000," *2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India, 2023, pp. 286-291, doi: 10.1109/ICCCIS60361.2023.10425313.
[3] Proakis, J. G. (2007). *Digital signal processing: principles, algorithms, and applications, 4/E*. Pearson Education India.
[4] Crockett, L. H., Elliot, R. A., Enderwitz, M. A., & Stewart, R. W. (2014). *The Zynq book: embedded processing with the ARM Cortex-A9 on the Xilinx Zynq-7000 all programmable SoC*. Strathclyde Academic Media.
[5] Crockett, L. H., Elliot, R. A., & Enderwitz, M. A. (2015). *The zynq book tutorials for zybo and zedboard*. Strathclyde Academic Media.
[6] The MathWorks, "Implement an Audio Filter on a Zynq Board," in *HDL Coder™ Documentation*, The MathWorks, Inc., Natick, MA, USA, 2026.
[7] The MathWorks, “HDL Coder Documentation,” The MathWorks, Inc., Natick, MA, USA.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY