ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

UDC 006.681.5: 004.67 doi: 10.32620/reks.2024.4.14

**Olexander SHKIL, Oleh FILIPPENKO, Dariia RAKHLIS,
Inna FILIPPENKO, Valentyn KORNIENKO**

Kharkiv National University of Radioelectronics, Kharkiv, Ukraine

ANALYSIS OF THE IMPLEMENTATION EFFICIENCY OF DIGITAL SIGNAL PROCESSING SYSTEMS ON THE TECHNOLOGICAL PLATFORM SOC ZYNQ 7000

*The **subject of this paper** is the analysis of DSP algorithm implementations based on HLS synthesis and SIMD instructions acceleration on the SoC hardware platform. The **goal** of this article is to analyze various FIR filter software and hardware implementations based on the technological platform SoC ZYNQ 7000 while obtaining metrics of hardware resource consumption, power efficiency, and execution performance. The **tasks** are as follows: determine the ways of implementing algorithms; choose the analysis criteria for multivariate experiment; implement algorithms using SIMD instructions on the ARM part of the given SoC; implement algorithms using High-Level Synthesis for the FPGA part; and measure and obtain the results for each signal topology. The used **methods**: High-Level Synthesis, optimization techniques based on vector instructions, and multivariate experiment analysis. The following **results** were obtained: for the given criteria and metrics. The FIR filter was implemented on the ZedBoard development platform with SoC ZYNQ 7000. The data were obtained from post-synthesis power analysis and dynamic SoC consumption using tools from Xilinx and Analog Devices. The corresponding IP blocks were implemented using High-Level Synthesis. The experiment was completed to obtain execution performance metrics. **Conclusions.** The scientific novelty of the obtained results is summarized as follows: the competitor analysis was performed for the set of implementations of the given algorithms deployed on the ZYNQ platform using both SIMD instructions and several HLS-based topologies for the FPGA-offload execution strategy. The analysis of the multivariate experiment was also completed for selected criteria, power consumption, filtering speed (inverse value – delay), and the amount of hardware costs as a percentage of the used resources.*

***Keywords**: digital filters; digital signal processing algorithms; audio signals; embedded systems; SoC; FPGA; programming language C; high level synthesis.*

1. Introduction

1.1. Motivation

The current development of requirements for audio and video transmission systems in real-time sets has created new challenges for developers of embedded systems where a component related to digital signal processor (DSP) algorithms and their practical implementation is present. The capabilities available in modern System-on-Chip (SoC) architectures allow developers to choose exactly where to implement the algorithm and perform all stages of its design, either on the processor part (PS) or the programmable logic (PL) side of the FPGA.

Note that the new trend – high-level synthesis (HLS) for implementation, using typical programming languages C/C++/Rust has replaced traditional approaches to algorithm design and their removal from the hardware part. This approach allows us to perform a complete cycle of algorithm implementation without considering the target hardware platform and obtain an intellectual property (IP) block that can be used in the design of a SoC-based system. The main advantages of this approach are the development speed and optimization features, which can be applied during the process settings of speech model transmission.

One of the relevant areas is the choice of the implementation approach of the DCP algorithm on the hardware platform SoC. In particular, on the ZYNQ platform, the developer has the option of complete implementation of the algorithm on the processor part of the SoC, implementation using the FPGA-offload approach when part of the data processing is carried out on the FPGA, and complete implementation in the hardware part of the programmable logic.

In this case, special attention should be paid to the selection of options for interacting PS and PL components with each other, considering the criteria of energy consumption, speed, and hardware costs of the obtained implementation.

As an option for analyzing the effectiveness of the DSP algorithms implementation, it is advisable to consider a typical algorithm of digital filter with finite impulse response (FIR) that can be implemented on the processor part of an ARM core with an elementary implementation of convolution; on the ARM part using vector instructions of the NEON coprocessor that is available in ARM A9 cores; and with FPGA offload approaches and complete hardware implementation. Several experiments with different parameters of the input audio stream are required for various filter implementations to confirm the analysis results.

Thus, determining the optimal way to implement the digital signal processing algorithm on the technological SoC platform using high-level automated synthesis is relevant.

1.2. State of the art

Implementing digital signal processing algorithms and measuring power consumption on hardware platforms like SoC is currently relevant. In particular, a specific research area focuses on the effective utilization of co-processors available on the SoC in the capacity of specialized calculation accelerators, such as the Neural Processing Unit (NPU).

In the paper [1], the authors proposed a sound effects system based on delay elements using the SoC platform. The proposed project was implemented based on the ZedBoard debugging board. Signal processing algorithms were implemented on the Programmable Logic (PL) part of ZYNQ. An Advanced RISC Machine (ARM) was used to control the parameters of the host computer's effects and interact with the programmable logic.

The features of interaction with the FFMPEG multimedia processing library were considered and analyzed in the materials of the ITOEC conference [2]. It explains that an ARM part is used to decode data in MP3 format and convert it into pure pulse-code modulation (PCM). Further, audio processing was performed on the FPGA of the ZYNQ. The AXI-DMA (Direct Memory Access) interface performs data exchange between PL and PS parts. This study also covers the debugging process of custom intellectual property (IP) cores using the built-in ILA Logic Analyzer of the Vivado development environment. In addition, an analysis of the resource utilization of the target hardware platform is presented.

This study explored the potential of real-time audio signal processing using a high-level synthesis and compilation model. This involved creating a high-level specification of the algorithm in the HLS description and generating the target IP-core for integration into the target platform [3].

In the paper [4], the authors consider using the two-dimensional fast Fourier transform (2D-FFT) algorithm in the biometrics and security context. The objective of this study is to evaluate the impact of co-design on processing time and resource use. This paper introduces a new architecture for the two-dimensional fast Fourier transform algorithm, which was then tested on the SoC ZYNQ. This paper presents an analysis of three implementations of the algorithm on different parts of the SoC ZYNQ. The effectiveness of the proposed architecture was evaluated through high-resolution images.

The central methodology for the automated design of embedded digital signal processing systems based on the SoC platform is described in [5]. This section presents approaches that can be used when implementing FIR and comb filters on the ZYNQ 7000 platform, along with an analysis of the resulting solution performance. In particular, attention was paid to the filter simulation process and the calculation of the necessary coefficients.

The study [6] focuses on different measurement configurations were used to estimate the energy consumption of information processing systems. The power consumption measurement during several clock cycles is assigned to specific groups of instructions. This type of energy measurement provides the information required to determine the energy consumption of particular software routines. The proposed configuration provides dynamic power consumption data, especially for single-chip processing systems, by monitoring the power consumption during various operational tasks. It also considers integrating the proposed configuration into the same chip as a peripheral device, which can be used independently for self-testing and consumption monitoring by the built-in application.

The basic information and methods for effective software design considering the features of energy consumption reduction are presented in [7]. The proposed metrics can be used during the development process to analyze primary energy consumption, explore profiling options, and optimize the resulting application.

The paper [8] presents a simplified automatic energy consumption measurement platform for embedded systems. This design replaces the intermediate device with a conventional USB-TTL device, thereby optimizing communication between the host computer and the target board. This simplification improves platform reliability and simplifies the software design of the host computer by eliminating the need for intermediate device software. Despite this simplification, the proposed platform retains features similar to those of the original design and is tested to verify its capabilities and accuracy. Compared to the original design, the simplified version demonstrates consistent accuracy, ease of deployment, and reduced device commutation requirements.

The paper [9] investigates the energy efficiency of various algorithm implementations for arrhythmia detection. The proposed method uses high-level synthesis tools and co-design approaches, as well as the study of implementation based on a pure software version of the algorithm. Implementations using different sample rates of the data were compared. A bandpass filter with a peak detector algorithm and discrete wavelet transformation was used for an electrocardiogram (ECG) analysis. In addition, a classifier based on a neural network is employed for further analysis. The power consumption and performance results were obtained using Linux PMBus.

The analysis of the hardware accelerator implementations based on the SoC ZYNQ is conducted in the study [10]. This study provides information about the characteristics and possible implementation topologies based on the AXI bus. It considers several hardware accelerator implementations, such as the Monte Carlo Softcore CPU and Telco Softcore CPU, which differ in their communication methods with the central ARM core ZYNQ. Additionally, it presents an overview of the implementation and features of AXI-DMA usage for communication with the IP cores of the system.

In the paper [11], the potential of single-board computers (SBCs) in cyber-physical systems (CPS) is considered. This study emphasizes the ability of a system to adapt to various current and future applications, as well as its scalability through cluster configurations, which can lead to energy savings. This research focuses on evaluating the energy efficiency of boards based on the ZYNQ Ultrascale+, which was developed as part of the AXIOM project. This study proposes and assesses a new application of model execution called Data-Flow-Threads (DF-Threads) on the ZYNQ Ultrascale+ platform to evaluate its energy efficiency. Power consumption metrics were collected during data transfer experiments using message types such as actual message (RAW) and remote direct memory access (RDMA) over a “board-to-board” connection. This research focuses on using DF-Threads in the context of CPS (Click Per Second) for energy efficiency, with potential implications for designing and optimizing energy-efficient system architectures in the future.

The previous study [12] provides a comprehensive review and discussion of methods for power consumption analysis developed for FPGA-based designs. This study aims to optimize the design strategy analysis, focusing on low-power strategies. Against the backdrop of rapidly evolving technologies such as the Internet of Things (IoT), where energy consumption is paramount, this paper highlights the importance of developing an efficient design approach. As the trend continues toward using FPGAs, IP (hard and soft cores), and SoCs in the creation of sophisticated mobile devices capable of performing complex tasks, the need to minimize power consumption has become increasingly evident. Despite the inherent advantages of FPGAs over other digital integrated circuits, their complex architecture often leads to higher power consumption, necessitating a unique approach to power optimization. This study examines various low-power methods, covering designs (architectures) at both system and device levels. The conclusions of this paper are summarized by summarizing the ideas and results obtained during the study of these low-power methods that can be applied in FPGA-based design.

In the paper [13], a method was introduced for efficiently determining which hardware accelerators (HWaccs) should be implemented on a processor with a specialized set of instructions for a specific application (ASIP) to maximize shared resources between them. The goal of this study is to implement hardware accelerators generated from unbound behavioral descriptions during HLS. Although HLS is a synthesis method that uses a single process, the proposed approach can achieve resource sharing among HWaccs by combining their behavioral descriptions into one, considering their potential for resource sharing. These shared resources include function units (FUs), such as multipliers, adders, dividers, and registers. In particular, the proposed approach leads to area savings of up to 48%, with an average of 30%. Because an exhaustive enumeration of all possible combinations can lead to long execution times, this paper proposes a fast heuristic that yields comparable results (on average, only 6% worse) while running much faster (500 times on average).

Embedded processors in systems-on-a-chip (SoC) include hardware accelerators to improve performance and reduce power consumption. These accelerators are more efficient at tasks, performing them faster and with less power. However, they require significant hardware resources because they heavily use parallelization for computation distribution. The question addressed in [14] is whether the processor can reuse these hardware resources when executing another program. This paper proposes an integrated methodology that automatically adapts CPU architecture with a tightly integrated hardware accelerator (or several) so that any program executed on the CPU (other than the accelerator) can take advantage of the additional hardware resources available in the accelerator. A backend Very Long Instruction Word (VLIW) compiler based on shared resources to regenerate machine code is also proposed to enable new applications to take advantage of this architecture. The experimental results demonstrate that the proposed methodology achieves an average speed of up to 1.7 times. The experimental part of the study was performed using the Cadence design system.

The paper [15] considered the potential for creating multiple design variations that are functionally equivalent and based on the same behavior description but have unique trade-offs regarding area, performance, and power. The analysis is based on machine learning and Pareto spaces using high-level synthesis tools to obtain optimal designs. In addition, a dedicated multi-threaded parallel HLS design space explorer (DSE) based on transfer learning that accelerates HLS DSE for ASICs was proposed.

The paper [16] provided essential information about the technical implementation of the coprocessor ARM NEON to use SIMD instructions effectively. The list includes technical details about libraries supporting ARM NEON and features of cross-compilation for various operating systems that can run on ARM-based platforms.

In the paper [17], the authors presented an approach for implementing stencil computations using ARM NEON and proposed a data transformation model to enhance the coprocessor loading efficiency. The obtained performance results were analyzed to demonstrate the effectiveness of the proposed model. The paper also compares the results of all proposed approaches.

1.3. Objectives and the approach

The analysis of the influence of implementation technical aspects on efficiency from the point of speed, power consumption, and hardware utilization showed that the problem of optimal implementation on the technological platform SoC is highly relevant from the point of view of both hardware utilization and obtained performance results, as well as from the point of appropriate use of available resources for balancing in the case of focusing on the power consumption of the target system.

This study aims to perform a comparative analysis of different variants of the software-hardware implementation of a typical DSP algorithm. It is necessary to implement a low-frequencies FIR filter based on the ZYNQ 7000 technological platform to determine both the utilization of the available SoC resources and the main characteristics of the filter, such as power consumption and speed, for each implementation.

Comparative analysis should be performed by conducting a multivariate experiment by both changing the parameters of the audio stream and the characteristics of the filter itself. To obtain the results of the experiments, we use a computer-aided design (CAD) toolkit based on high-level synthesis that allows rapid prototyping and verification of the obtained IP-block before its inclusion into the final system architecture.

Therefore, the main objectives of this study are as follows:

– investigating possible ways to implement DSP algorithms using SoC and to design low-frequency FIR filters using Vivado/Vitis/Vitis HLS CAD tool stack with the C programming language (Section 2);

– experiments on the energy consumption, speed, and hardware use of different implementations of the low-frequency FIR filter (Section 3);

– analysis of the experimental results to identify the most effective implementation method of the low-frequency FIR filter for the ZYNQ 7000 on ZedBoard (Section 4);

– discussion of the results and directions for further research (Section 5);

– summarizing the obtained scientific results (Section 6).

2. Materials and methods of research

In general, the output of most DSP algorithms can be described as follows:

$$y[n] = T\{x[n]\}, \qquad (1)$$

where $y[n]$ – the value at the output of the system;

T – the processing function;

$x[n]$ – the input value;

For linear discrete systems, the convolution expression ($\star$) is generally used defined to obtain the output value of the system [19].

The difference equation for a linear system that doesn't depend on time (Linear Time Invariant) is defined as

$$y[n] = T\{x[n]\} = \sum_{k=-\infty}^{\infty} h[k]\cdot x[n-k] = h[n] * x[n] \qquad (2)$$

where $h[k]$ – the impulse characteristic of the filter;

$h[n]$ – the impulse response of the filter, which is convoluted with the input counts $x[n]$.

For the implementation of filters with a finite-impulse response from the family of DSP algorithms, it is necessary to implement a one-dimensional convolution with the filter's impulse response [5].

The Xilinx ZYNQ platform has two parts: PL and PS. The PS part is implemented as two fully-fledged ARM Cortex A9 cores with ARM Neon technology to execute SIMD instructions.

Thus, for the implementation of DSP algorithms, it is possible to use the ARM part without SIMD instructions. The PL part is separately allocated on which hardware specialized IP cores for user needs can be implemented, and interaction with the PS part using the AXI (Advanced eXtensible Interface) bus can be implemented. The operation with the NEON accelerator in the ARM is possible either by using the *Intrinsic* functions of the compiler or by using official libraries. One of the options is development based on the NE10 library from the official manufacturer. NE10 contains both a software implementation of typical DSP algorithms and an accelerated version, which uses NEON. Depending on the presence of NEON on the target core, appropriate implementations are selected. In addition, the required available functionality and optimization level must be specified when compiling the library. In this study, the maximum optimization level *–O3* of the compiler was specified, and *–ffast –math* was additionally added for both the library and the software part of the application. Separately, the flag of the assembly, which is specific to the ARM part ZYNQ, was added [5]. The main implementation options considered in this study are implementations of the FIR filter based on the available resources of the SoC ZYNQ.

An FIR filter was implemented for comparative analysis based on vector SIMD instructions of the ARM part of ZYNQ using the NE10 library (a). The second implementation (b) is based on the native convolution implementation without vector instructions. The third implementation (c) employs the FPGA offloading approach, which is implemented as a separate IP-core. This IP-core was developed by HLS and uses an AXI4-Lite bus to receive and transmit data to the ARM part of the SoC. The fourth implementation (d) was performed entirely on the PL part of ZYNQ using the AXI-Stream bus and the built-in Vitis HLS vectorization tools.

The block diagram of the developed architecture is shown in Fig. 1.

Within the scope of this study, the implementation power consumption was measured using the available current monitor on the ZedBoard debugging board. The Vivado Power Profiler was used to obtain approximate power consumption data for each implementation. A 64-bit built-in counter was used to measure the execution time of each considered approach. It is available with the xtime-API on the hardware abstraction layer (Vitis HAL) in «bare metal» mode without an operating system. The execution time of the algorithms in the PL part was measured with the help of a logic analyzer by measuring the delay between the input and output samples of the audio stream. The experimental architecture design was carried out in the Vivado IDE environment for the final integration of the developed IP-blocks and the ARM configuration of the ZYNQ part. The software was developed using the programming language C in the Vitis IDE environment. The development of IP-blocks for the experiment with FPGA-offload and the full implementation of the algorithm on PL were performed in the Vitis HLS environment.

3. Case study: low-frequency FIR filter implementation on the SoC ZYNQ 7000

As demonstrated earlier, there are several hardware implementation methods for low-frequency digital FIR filters on the SoC ZYNQ 7000 technological platform.

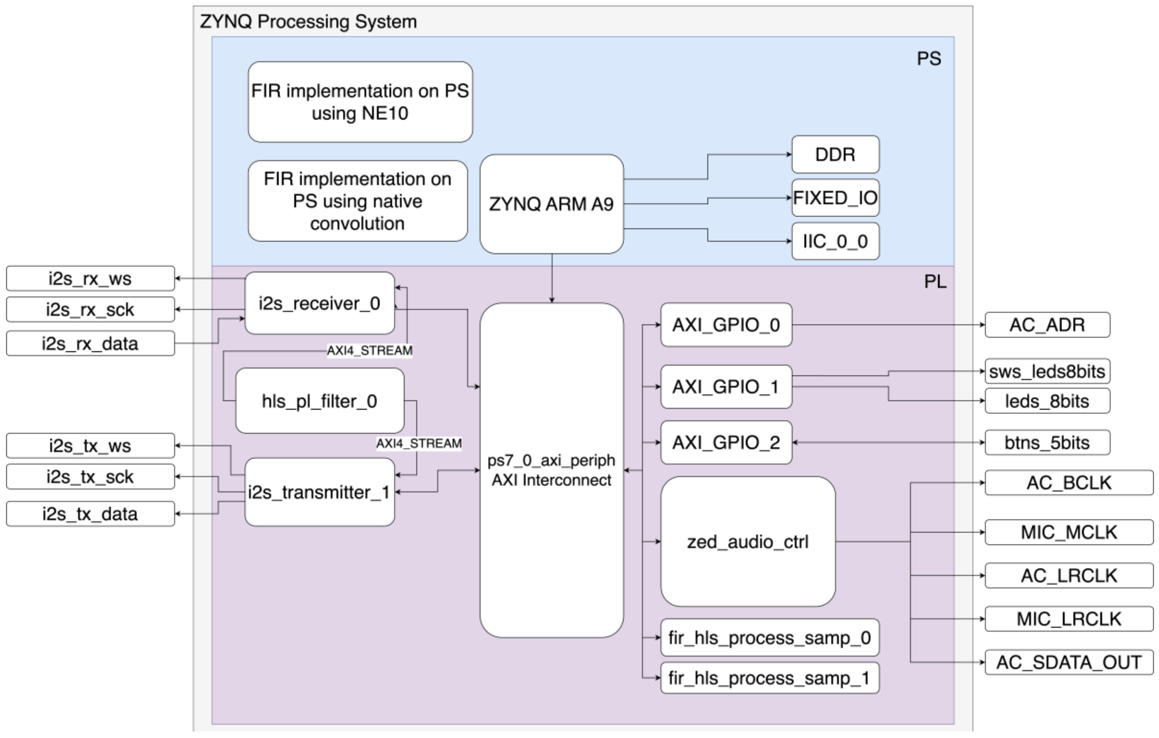

Fig. 1. Block diagram of the proposed architecture based on ZedBoard hardware platform

The main difference between the considered implementations is how the calculations are distributed between the PL and PS parts of the SoC. This, in turn, affects the filtration speed and energy consumption, which are essential for embedded systems.

3.1. Conduction methodology and results of experiments

A full-scale experiment was conducted to analyze the parameters and verify the effectiveness of the proposed technical implementation of the low-frequency FIR filter on the SoC ZYNQ 7000 technological platform using the ZedBoard debugging board.

This study focused on examining the parameters of a low-frequency FIR filter, including its power consumption, speed, and hardware use. The energy consumption was analyzed by measuring the current shunt of the ZedBoard board. Speed during the implementation in the PS part of the SoC was measured using an internal counter of the ARM core and during the implementation in the PL part using an external logic analyzer. Hardware use was analyzed according to synthesis protocols in the Vivado IDE environment.

This experiment aimed to assess how the audio stream parameters affect the implementation of a digital low-frequency FIR filter. The study used a multivariate experimental method to analyze the complex impact of various parameters on the effectiveness of technical realization [18].

Statistical analysis of the experiment results was not conducted, but the study focused on the direct influence of a factor's complex on the effectiveness of the FIR filter implementation.

During the research, the following hardware-software implementations of the FIR algorithm on the SoC ZYNQ 7000 were considered:

– implementation of the algorithm on the PS part using NE10 for SIMD operations (a);

– native implementation without the use of vector instructions (b);

– partial transfer of algorithm components to the PL part using AXI (c);

– full transfer of the algorithm to the PL part (d).

The following variables and parameters of the input audio stream were used as factors in the experiment:

– N (signal/noise ratio: 5dB – 15 dB – 25dB);

– P (resolution, bits per sample: 24/16 bit);

– F (sample rate: 48 kHz).

During the experiment, the following initial parameters were measured:

– E (energy consumption in W);

– S (filtering speed, inverse value – delay in μs);

– A (amount of hardware use as a percentage of resource use).

Hardware use for the PS part (experiments a and b) was measured as a percentage of the used RAM out of 512 MB of available memory. Hardware utilization for the PL part (experiments c and d) was measured as a percentage of the used synchronous flip-flops (total 106400 FF) and look-up-tables (total 53200 LUTs). Table 1 lists the results of the five most significant experiments.

Table 1

Results of the multivariate experiments

	E(W)	S(μs)	A (Resource utilization)
Experiment 1: N1(5dB), P1(24 bits), F1(48kHz)			
a	1.850	1.85	248 bytes, $4.84*10^{-5}$ %
b	1.890	3.51	188 bytes, $3.67*10^{-5}$ %
c	1.734	2.65	FF 1187, 1.12%, LUT 1744, 3.28%
d	1.699	0.283	FF 1814, 1.70%, LUT 4741, 8.91%
Experiment 2: N2(15dB), P1(24 bits), F1(48kHz)			
a	1.843	1.83	248 bytes, $4.84*10^{-5}$ %
b	1.848	3.61	188 bytes, $3.67*10^{-5}$ %
c	1.836	2.68	FF 1187, 1.12%, LUT 1744, 3.28%
d	1.699	0.235	FF 1814, 1.70%, LUT 4741, 8.91%
Experiment 3: N3(25dB), P1(24 bits), F1(48kHz)			
a	1.840	1.82	248 bytes, $4.84*10^{-5}$ %
b	1.884	3.49	188 bytes, $3.67*10^{-5}$ %
c	1.731	2.63	FF 1187, 1.12%, LUT 1744, 3.28%
d	1.699	0.28	FF 1814, 1.70%, LUT 4741, 8.91%
Experiment 4: N1(5dB), P2(16 bits), F1(48kHz)			
a	2.16	1.87	184 bytes, $3.59*10^{-5}$ %
b	2.196	3.31	436 bytes, $8.51*10^{-5}$ %
c	1.740	2.85	FF 1146, 1.08%, LUT 1472, 2.77%
d	1.699	0.22	FF 1806, 1.7%, LUT 4717, 8.87%
Experiment 5: N2(15dB), P2(16 bits), F1(48kHz)			
a	2,064	1.87	184 bytes, $3.59*10^{-5}$ %
b	2.10	3.35	436 bytes, $8.51*10^{-5}$ %
c	1.740	2.84	FF 1146, 1.08%, LUT 1472, 2.77%
d	1.699	0.218	FF 1806, 1.7%, LUT 4717, 8.87%

3.2. Analysis of the obtained results

The experimental results showed that the proposed low-frequency FIR filter implementation methods are workable and can be implemented on the SoC ZYNQ 7000 hardware platform.

Based on the fact that the hardware costs of the FIR filter on the SoC ZYNQ 7000 do not exceed 10%

for any technical implementation, the hardware costs as a third measurement parameter can be neglected.

The experimental results can be plotted on the two-dimensional plane E – S (fig. 2).

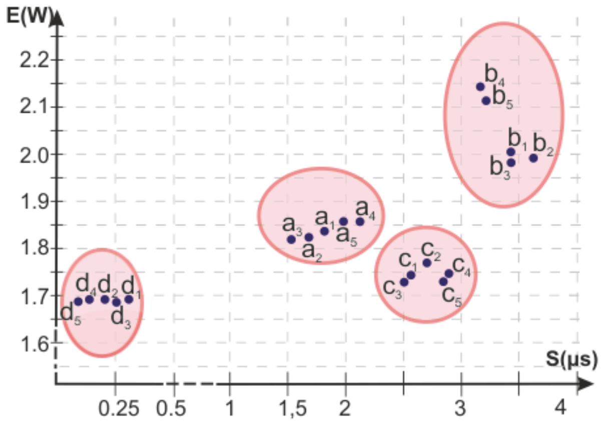

Fig. 2. Distribution of multivariate experiment results

Upon analyzing the distribution, it can be concluded that the values of the output measuring quantities (E and S) don't depend on the input parameters of the audio stream but rather on the technical implementation method of the FIR filter on the SoC ZYNQ 7000.

The best results (in terms of E and S parameters) were obtained for implementation options d and a. This proves that using the AXI4-Lite/AXi4HP-bus during the technical implementation of the FIR filter on the SoC ZYNQ 7000 is impractical. At the same time, using the AXI4-Stream interface with Master-Slave topology is appropriate. It is important to note that if data need to be collected from the PS part after partial preprocessing on the PL, then implementations (options a and d) using AXI-FIFO and AXI Stream are appropriate. Another option to consider is AXI-DMA with AXI-FIFO+AXI Stream bus.

The best results (in terms of E and S parameters) were achieved by implementing the FIR filter exclusively on the PL part (FPGA) of the SoC ZYNQ 7000. Regarding the speed parameter, the NE10-based PS implementation using vector SIMD instructions was the most effective.

All technical implementations of the FIR filter on the SoC ZYNQ 7000 had approximately the same power consumption (ranging from 1.7 to 2.2 W). However, the proposed PL implementation significantly outperforms other methods in terms of filtering speed (almost 10 times faster). This makes PL implementation the preferred choice when there is no need for data processing at the PS part.

4. Discussion and recommendations

Within the framework of this study, an effectiveness analysis of the digital FIR filter implementation was performed using the high-level synthesis tool Vitis HLS and the ARM part. Due to the actual problem of calculation distribution in SoCs during the implementation of digital signal processing systems and the power consumption problem of embedded systems, an analysis of several possible topologies of the ZYNQ platform was carried out. During the effectiveness analysis of the implementation of digital signal processing systems on the technological platform SoC ZYNQ 7000, we investigated two aspects: the effective implementation of digital filtering algorithms using the NEON coprocessor, as well as the technical characteristics of different options for distributing calculations between the PS and PL parts of the SoC ZYNQ 7000 using the AXI interface for data transfer.

FIR filters are widely used in digital signal processing. The main advantages and capabilities of such systems make them critical elements for various tasks in digital systems. The main benefits of this method are accuracy and stability because they do not require feedback. This is important for real-time computing at the hardware level, such as in SoC, where instability in signal processing must be avoided. FIR filters allow fine-tuning of frequency characteristics, such as bandwidth or attenuation, in specific frequency ranges. We can use them to construct high-precision filters (e.g., low-pass, high-pass, or band-pass). In modern processor cores, which often use accelerators for signal processing or embedded digital signal processors, FIR filters can be implemented using hardware blocks to accelerate calculations.

The topology with the implementation of the FIR filter based on NE10, which is optimized for ARM Cortex-A processors, such as those used in the ZYNQ (Cortex-A9) platforms, showed its advantage due to the need to implement the filter on the PS part and higher speed compared to the topology with partial export to the PL part and native implementation. The topology's main disadvantage is its limited support for algorithms implemented in NE10, which can be accelerated by SIMD instructions for NEON.

The most straightforward option from the development side is topology with a native implementation of the FIR filter without vector instructions; however, it is the slowest among the considered options.

The topology with partial removal of the FIR filter components to the PL part and the use of AXI4-Lite offers a balance between performance and power consumption. Using the AXI4-Lite allows us to reload the filter coefficients into the PL part, if necessary.

The topology with a complete implementation of the FIR filter in the PL part of the SoC is the best solution for speed and power consumption. The disadvantages of this topology include higher hardware costs for implementation and limited options for interaction with the PS part using only the AXI FIFO or AXI DMA.

We need to conduct further research on algorithmic optimizations of filtering algorithms and their implementation in the programming language C considering the capabilities of the high-level synthesis environment Vitis HLS on the platform ZYNQ 7000 SoC, including the distribution of calculations between PS and PL parts.

5. Conclusion

The main contribution of this study is to analyze the effectiveness of different implementation methods for digital FIR low-pass filters on the technological platform SoC ZYNQ 7000 in terms of calculation distribution between the PL and PS parts of the SoC. FIR low-pass filter models were created and implemented based on the stack of CAD tools Vivado/Vitis/Vitis HLS using the C programming language.

The models were implemented based on the ZedBoard hardware platform using the Xilinx Vivado toolkit to design and integrate IP blocks. The IP blocks in the PS/PL and PL parts were designed using Vitis HLS. Depending on the type of AXI bus, an appropriate set of pragma operators was used to determine the interface type that should be between PL-PL blocks and PL-PS parts.

A multivariate analysis experiment was conducted to verify the effectiveness of the proposed models by examining both the audio stream parameters and filter characteristics. The parameters of the low-pass FIR filter, including power consumption, speed, and hardware costs, were investigated.

The performance of the PL implementation was measured using a logic analyzer, and the audio frame delay was analyzed through the I2S interface. During the measurement, the I2S-bypass was implemented, and the IP core of the FIR filter was added. Data exchange between blocks in the PL part was built based on the AXI Stream bus with a master-slave topology. The analysis of the multivariate experiment results showed that implementing the FIR low-pass filter on the PL part of the SoC ZYNQ 7000 is optimal in terms of speed with approximately the same power consumption and hardware costs.

Further research may include several relevant options for future research.

Specifically, this study investigates a more complex object that uses multiple algorithms during processing, such as RNNoise, for adaptive noise reduction.

Another research direction involves manipulating the software-hardware implementation code and balancing the distribution of data processing between hardware and software parts.

Further research into power consumption includes studying and using the power profiling tools available in the Embedded Linux user space/kernel space part.

The current research directions should also consider open direction-of-arrival (DOA) algorithms and their potential implementation using the FPGA-offload approach.

Contributions of authors: conceptualization, methodology – **Olexander Shkil;** the review and analysis of sources – **Valentyn Kornienko**, **Dariia Rakhlis**; formulation of tasks, analysis **– Olexander Shkil**, **Oleh Filippenko**; DSP algorithm development – **Inna Filippenko**; software implementation, evaluation of the method effectiveness **– Valentyn Kornienko;** analysis of results, visualization **– Olexander Shkil**, **Inna Filippenko**, **Valentyn Kornienko**; writing – **Olexander Shkil**, **Valentyn Kornienko**; review, translation and editing **– Dariia Rakhlis.**

Conflict of Interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

This study was conducted without financial support.

Data Availability

The work has no associated data.

Use of Artificial Intelligence

The authors confirm that they did not use artificial intelligence methods while creating the presented work.

All the authors have read and agreed to the publication of the final version of this manuscript.

References

1. Cannon, D., Fang, T., & Saniie, J. Modular delay audio effect system on FPGA. *IEEE International Conference on Electro Information Technology (*EIT'22*)*, Mankato, USA 19-21 May, 2022 , pp. 248–251. DOI: 10.1109/eIT53891.2022.9813875.

2. Xie, W., & Yang, F. Design and implementation of audio stream processing based on ZYNQ. *IEEE 6th information technology and mechatronics engineering conference (ITOEC)*, Chongqing, China, 4-6 March 2022, pp. 589–592. DOI: 10.1109/ITOEC53115.2022. 9734347.

3. Popoff, M., Michon, R., Risset, T., Cochard, P., Letz, S., Orlarey, Y., & Dinechim, de F. Audio DSP to FPGA Compilation: The Syfala Toolchain Approach, *Research report №9507*, Grame, Emeraude: Inria, 2023. 18 p. Available at: https://hal-lara.archives-ouvertes.fr/hal-04099135 (accessed 10.05.24).

4. Kortli, Y., Gabsi, S., Jridi, M., Alfalou, A., & Atri, M. HW/SW co-design technique for 2D fast fourier transform algorithm on Zynq SoC. *Integration*, 2021, vol. 82, pp. 78–88. DOI: 10.1016/j.vlsi.2021.09.005.

5. Shkil', O. S., Rakhlis, D. Yu., Filipenko, I. V., Korniyenko, V. R., & Rozhnova, T. H. Avtomatyzovane proyektuvannya vbudovanykh system tsyfrovoho obroblennya syhnaliv na plat-formi SoC [Automated design of embedded digital signal processing systems on SoC platform]. *Suchasnyy stan naukovykh doslidzhen' ta tekhnolohiy v promyslovosti – Innovative technologies and scientific solutions for industries*, 2024, vol. 1 (27), pp. 72–83. DOI: 10.30837/ITSSI.2024.27.192.

6. Konstantakos, V., Kosmatopoulos, K., Nikolaidis, S., & Laopoulos, T. Measurement of power consumption in digital systems. *IEEE Transactions on Instrumentation and Measurement,* 2006, vol. 55, no. 5, pp. 1662–1670. DOI: 10.1109/tim.2006.880311.

7. Kruglov, A., & Succi, G. *Developing sustainable and energy-efficient software systems*. Springer International Publ., Cham, 2023. 77 p. DOI: 10.1007/978-3-031-11658-2.

8. Wu, H., Chen, C., & Weng, K. A Simplified design of automatic energy consumption measuring platform for embedded systems. *Journal of Physics: Conference Series,* 2020, vol. 1575, pp. 1–6. DOI: 10.1088/1742-6596/1575/1/012073.

9. Railis, K., Tsoutsouras, V., Xydis, S., & Soudris, D. Energy profile analysis of Zynq-7000 programmable SoC for embedded medical processing: Study on ECG arrhythmia detection. *26th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS),* Bremen, Germany, 21-23 September 2016, pp. 275–282. DOI: 10.1109/patmos.2016.7833699.

10. Toft, J. K., & Nannarelli, A. Implementation of hardware accelerators on Zynq. DTU Compute Technical Report. *Technical University of Denmark*, 2016, No. 7. 26 p. Available at: https:// orbit.dtu.dk/files/125849853/tr16_07_Nannarelli_A.pdf (accessed 10.04.24).

11. Giorgi, R., Khalili, F., & Procaccini, M. Energy Efficiency Exploration on the ZYNQ Ultrascale+. *30th International Conference on Microelectronics (ICM'2018)*, Sousse, Tunisia, 2018, pp. 48–54. DOI: 10.1109/icm.2018.8704092.

12. Ibro, M., & Marinova, G. Review on Low-Power Consumption Techniques for FPGA-based designs in IoT technology. *16th International Conference on Telecommunications (ConTEL'2021)*, Zagreb, Croatia, 30 June-2 July 2021, pp. 110–114. DOI: 10.23919/ConTEL 52528.2021.9495970.

13. Si, Q., & Schafer, B. C. MOSAIC: maximizing resource sharing in behavioral application specific processors. *Microprocessors and Microsystems*, 2024, vol. 106, pp. 1–9. DOI: 10.1016/j.micpro.2024.105039.

14. Si, Q., & Schafer, B. C. PEPA: performance enhancement of embedded processors through HW accelerator resource sharing. *Great Lakes Symposium on VLSI (GLSVLSI'23),* New York, 2023, pp. 23–28. DOI: 10.1145/3583781.3590277.

15. Rashid, Md. I., & Schafer, B. C. Fast and inexpensive high-level synthesis design space exploration: Machine learning to the rescue. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* 2023, vol. 42 (11), pp. 3939–3950. DOI: 10.1109/TCAD.2023.3258341.

16. *Architecture Neon*. Arm Developer. Available at: https://developer.arm.com/ Architectures/%20Neon (accessed 02.04.2024).

17. Zhang, K., Su, H., Zhang, P., & Dou, Y. Data layout transformation for stencil computations using ARM NEON extension. *IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, Yanuca Island, Cuvu, Fiji, 2020, pp. 180–188. DOI: 10.1109/hpcc-smartcity-dss50907.2020.00023.

18. Box, E. P. G., Hunter, W. G., & Hunter, J. S. *Statistics for experimenters: design, innovation, and discovery.* 2nd ed. Hoboken, New Jersey, John Wiley & Sons Publ., 2005. 672 p. URL: https://pages.stat.wisc.edu/~yxu/Teaching/16%20spring%20Stat602/%5BGeorge_E._P._Box,_J._Stuart_Hunter,_William_G._Hu(BookZZ.org).pdf (accessed 10.04.24).

19. Smith, S. W. *The scientist and engineer's guide to digital signal processing*. 2nd ed. San Diego, California, California Technical Publ., 1999. 650 p. URL: https://www.dspguide.com/ pdfbook.htm (accessed 10.04.24).

Received 28.05.2024, Accepted 18.11.2024

АНАЛІЗ ЕФЕКТИВНОСТІ РЕАЛІЗАЦІЇ СИСТЕМ ОБРОБКИ ЦИФРОВОГО СИГНАЛУ НА ТЕХНОЛОГІЧНІЙ ПЛАТФОРМІ SOC ZYNQ 7000

О. С. Шкіль, О. І. Філіпенко, Д. Ю. Рахліс,
І. В. Філіпенко, В. Р. Корнієнко

Предметом вивчення в статті є аналіз ефективності реалізацій алгоритмів цифрової обробки сигналів з використанням інструментів високорівневого синтезу та векторних інструкцій обробки на технологічній платформі SoC. **Метою** є аналіз різновидів програмно-апаратної реалізації КІХ-фільтру на технологічній платформі SoC ZYNQ 7000 з метою визначення апаратних витрат, енергоспоживання та швидкодії. **Завдання:** визначити основні варіанти реалізації алгоритмів; обрати критерії оцінки для багатофакторного експерименту; виконати реалізацію з використанням векторних інструкцій ARM ядра; навести реалізацію з використанням високорівневого синтезу для FPGA частини; отримати результати виконання кожної з топологій. Використані **методи**: високорівневий синтез, оптимізаційні техніки на базі векторних інструкцій, аналіз даних багатофакторного експерименту. Отримані такі **результати**: згідно до обраних критеріїв та метрик виконано реалізацію алгоритму цифрової обробки сигналу на базі технологічної платформи ZYNQ 7000 та плати налагодження ZedBoard. Для реалізації використано набір інструментів від Xilinx для сімейства SoC ZYNQ 7000. Враховано дані, отримані в ході аналізу synthesis power report та вимірювання споживання за допомогою інструментарію від Analog Devices. Блоки обробки на FPGA частині розроблено за допомогою високорівневого синтезу. Виконано експериментальне дослідження та аналіз швидкодії кожної з реалізацій з підрахунком відповідних апаратних витрат. **Висновки.** Наукова новизна отриманих результатів полягає в наступному: було виконано порівняльний аналіз ефективності різних варіантів реалізації алгоритму цифрової обробки сигналів на технологічній платформі ZYNQ з урахуванням використання векторних інструкцій ARM NEON на частині PS та з використанням підходу FPGA-offload для двох топологій. Також було проаналізовано результати багатофакторного експерименту з урахуванням обраних критеріїв, таких як споживання енергії, швидкодія фільтрації (зворотна величина – затримка) та кількість апаратних витрат в відсотках використаних ресурсів.

Ключові слова: цифрові фільтри; алгоритми цифрової обробки сигналів; аудіосигнали; вбудовані системи; системи на кристалі; FPGA; мова програмування C; високорівневий синтез.

Шкіль Олександр Сергійович – канд.техн. наук, доц., доц. каф. автоматизації проєктування обчислювальної техніки, Харківський національний університет радіоелектроніки, Харків, Україна.

Філіпенко Олег Ігорович – канд. техн. наук, доц., доцент кафедри інфокомунікаційної інженерії ім. В.В. Поповського, Харківський національний університет радіоелектроніки, Харків, Україна.

Рахліс Дарія Юхимівна – канд. техн. наук, доц., доц. каф. автоматизації проєктування обчислювальної техніки, Харківський національний університет радіоелектроніки, Харків, Україна.

Філіпенко Інна Вікторівна – канд. техн. наук, доц., доц. каф. автоматизації проєктування обчислювальної техніки, Харківський національний університет радіоелектроніки, Харків, Україна.

Корнієнко Валентин Русланович – асп. каф. автоматизації проєктування обчислювальної техніки, Харківський національний університет радіоелектроніки, Харків, Україна.

Olexander Shkil – PhD, Associate Professor, Associate Professor at the Design Automation Department, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: oleksandr.shkil@nure.ua, ORCID: 0000-0003-1071-3445, Scopus Author ID: 6506160916.

Oleh Filippenko – PhD, Associate Professor, Associate Professor at the Infocommunication Engineering Department Named by V. V. Popovsky, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine.
e-mail: oleh.filippenko@nure.ua, ORCID: 0000-0003-4616-250X, Scopus Author ID: 57194036787.

Dariia Rakhlis – PhD, Associate Professor, Associate Professor at the Design Automation Department, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: dariia.rakhlis@nure.ua, ORCID: 0000-0002-6652-1840, Scopus Author ID: 27867781600.

Inna Filippenko – PhD, Associate Professor, Associate Professor at the Design Automation Department, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: inna.filippenko@nure.ua, ORCID: 0000-0002-3584-2107, Scopus Author ID: 24483080100.

Valentyn Korniienko – PhD Student of the Design Automation Department, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: valentyn.korniienko1@nure.ua, ORCID: 0000-0001-7070-5127; Scopus Author ID: 57352374900.