ISSN 1814-4225 (print)
Radioelectronic and Computer Systems, 2023, no. 2(106) ISSN 2663-2012 (online)

UDC 004.932:004.312.466 **doi: 10.32620/reks.2023.2.08**

Olesia BARKOVSKA[1], Inna FILIPPENKO[1], Ivan SEMENENKO[1], Valentyn KORNIIENKO[1], Peter SEDLAČEK[2]

[1] ***Kharkiv National University of Radio Electronics, Kharkiv, Ukraine***
[2] ***Department of Informatics, University of Zilina, Zilina, Slovakia***

ADAPTATION OF FPGA ARCHITECTURE FOR ACCELERATED IMAGE PREPROCESSING

The work is devoted to the topical problem at the intersection of communications theory, digital electronics and numerical analysis, namely the study of image processing methods implementation time on different architectures of computational devices, which are used for software and hardware acceleration. ***The subject*** *of this article is the investigation of reconfigurable FPGA processing systems in the image processing area.* ***The goal*** *of this work is to create a reconfigurable FPGA-based image processing system and compare it with existing processing architectures.* ***Task.*** *To fulfill the requirements of this work, it is necessary to prepare a practical experiment as well as theoretical research of the proposed architecture; to investigate the process of creating a ZYNQ SoC-based image processing system; and to develop and benchmark the speed of execution for the given set of algorithms with the specific range of the picture resolution.* ***Methods used:*** *FPGA simulation, C++ parallel programming with OpenMP, NVIDIA CUDA, performance analysis tools.* ***The result*** *of this work is the development of a resilient SoC Zynq7000–based computing system with programmable logic and the possibility to load images to FPGA RAM using the resources of ARM core for further processing and output via HDMI video interface, which enables the change of PL configuration at any time during the processing process.* ***Conclusions.*** *The efficiency of the FPGA approach was compared with a parallel image processing method implementation with OpenMP and CUDA. An overview of the ZYNQ platform with specific details related to media processing is presented. The analysis of algorithm speed testing findings based on various outputs proved the advantage (of over 60 times) of hardware acceleration of image processing over software analogs. The obtained results may be used in the development of embedded SoC-based solutions that require acceleration of big data processing. Also, the achieved findings can be used during the process of finding a suitable embedded platform for a certain image-processing task, where high data throughput is one of the most desired requirements.*

Keywords: *speedup; FPGA; performance; acceleration; parallel system; image processing.*

Introduction

The present-day development of various areas of human activity is tightly coupled with the development of information technologies, digital devices, and digital signal processing. IT development and digitization are intensively integrated in spheres that combine scientific research and engineering practice. Along with this, the mathematical tool, special-purpose technologies, input signal type (seismic vibrations, visible object images, acoustic vibrations etc.), software and hardware base as well as the demands put forward within problem definition vary depending on the application domain (radiolocation and radionavigation, oil production, space research, medical facilities, manufacturing, military hardware, science) of the digital signal processing theory [1].

Digital signal processing theory is tightly connected with disciplines such as data communication theory, numerical analysis, probability theory, statistical analysis, analog electronics, and digital electronics.

Having analyzed the abovementioned, it can be stated that the solution to the problem, which is at the intersection of various disciplines, is quite topical. An example of such task is the application of special-purpose computing devices for digital image processing algorithm acceleration.

Digital image processing is a form of information processing and conversion, in which data input is presented in the form of images (photos and video frames).

Image processing aims at producing a different image as an output or different attributes of the incoming image [2]. In addition, a necessity often emerges to process images that change in time, for instance, films and videos. The spheres of practical application of image processing methods are numerous: law, military science, medical establishments, aviation and space exploration, archives and electronic libraries, multiservice networks and the Internet [3, 4], weather services, law enforcement agencies etc.

Fig. 1 also displays digital image processing methods, which can be implemented on different software and hardware platforms. Depending on the need, processing may be done with the application of: CPU, which is enough for fulfilling the majority of everyday tasks [5]; GPU with SIMD-based architecture [6]; and FPGA,

which enables the avoidance of execution of extra instructions and gives the developer full control of the data flow [7].

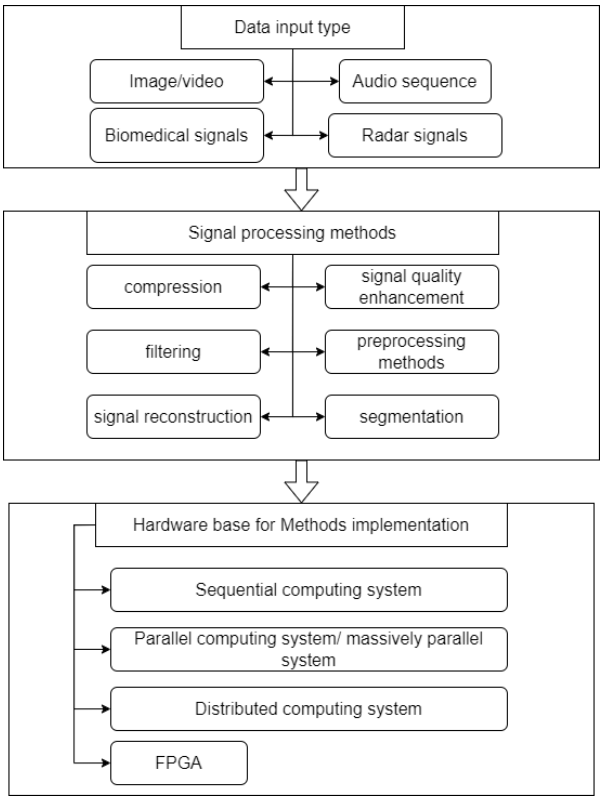


Fig. 1. Problem area analysis

One of the given boards is Xilinx Zynq 7020-based Z-turn Board (Fig. 2), which includes ARM Dual-core Cortex A9 (PS) and Kintex-7 (PL). PS may operate at a frequency of up to 866MHz and supports up to 1GHz DDR SDRAM. PL has 53200 LUTs, 106400 Flip-Flops, 140x36Kb RAM and 220 DSP blocks. The board itself fully caters for the minimum number of peripherals.

Overall, all the systems for prototyping SoC-based software and hardware solutions are similar by a series of factors. As a rule, systems on the chip combine the functional components of the whole device. They consume less energy, are less costly and work more efficiently than chipsets with the same functionality. Fewer components simplify the assembly of the finished product. One of the most popular SoC is Zynq7000 series by Xilinx (Fig. 3), which has two cores (ARM + FPGA) and provides a variety of interfaces for establishing communication (data exchange) between them.

Vendors of breadboards and software and hardware solutions debugging boards attempt to add as many peripherals as possible and provide maximum variability of their settings. The availability of HDMI, Ethernet, SD, USB HS and USB UART is the minimum for the given systems.

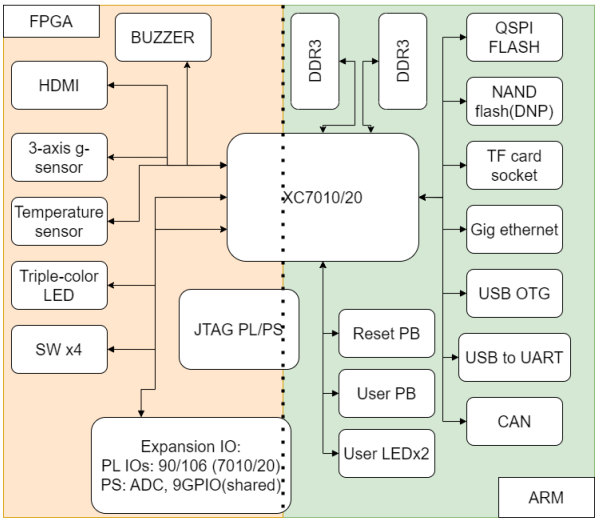


Fig. 2. Flow Diagram
for Z-turn Board Components [8]

Research task rationale. Related works

SoC-based system integration remains a practical area for both research and the platform development process.

Modern embedded systems contain complex firmware as well as the peripheral interconnections between the modules. The current requirements for most applications to be developed include:

– big output data processing capacity;

– original problem solution time improvement;

– usability and understandability features of the developed application;

– ease of scalability and modification;

– competitive development cost;

– etc.

Hardware computational power may help to meet the first two requirements from the abovementioned list. Due to the fact that Moore's law has been thrown overboard, yet the necessity to increase the computational capacity still exists, it is only possible to meet the requirements for computing and big output data processing speedup via the application of parallel processing as well as targeted computer devices adapted for special tasks execution [10, 11].

The works [12, 13] show that CPU and GPU-based computing systems can be used equally effectively to speedup image preprocessing methods. However, the obtained acceleration depends on the opportunity for algorithm' decomposition for the decomposition of the algorithm.

In [14, 15], attention is paid to the approaches to data parallelism organization and techniques of output data segmentation with the definition of computation granularity regarding the type of hardware base.

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

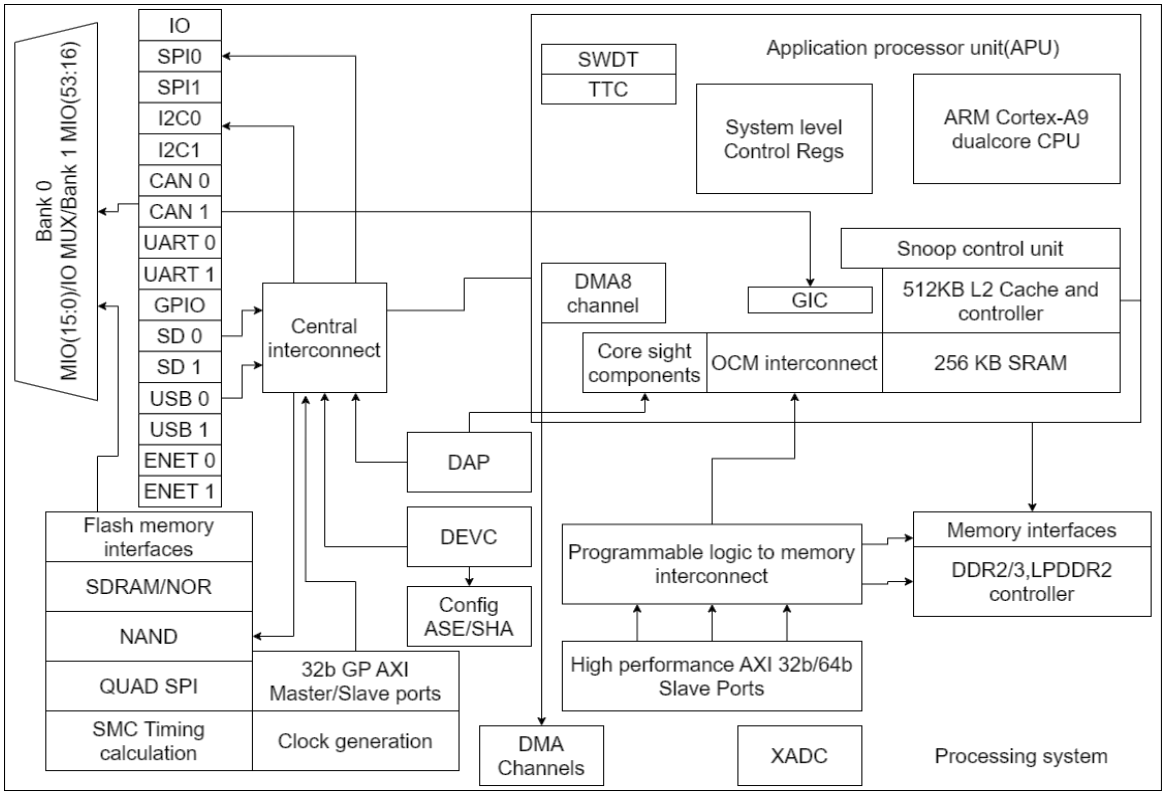


Fig. 3. Interface Flowchart for Xilinx Zynq7 [9]

Massively parallel architecture also justifies the need in GPU-based computations for general-purpose problem solutions [13, 16]. Application of CUDA technology along with the computing resource of a graphic processing unit (Nvidia Tesla K20) provides a speedup of up to 376 times during ultrasound image analysis and up to 426 times in problem solution aimed at aerial target identification based on infrared ray reflection time.

Modern FPGAs gain increased value in the application of software processors to solve several tasks, as mentioned in [17 – 19].

FPGA devices provide a series of sequential and parallel configuration interfaces [20] that can be used for loading configuration data. As a rule, the process of uploading bitstream requires the stop of chip operation; however, there are some devices that support dynamic partial reconfiguration (DPR).

FPGA-based systems use special-purpose equipment for logic processing. Unlike program solutions, which use context switching to maintain several threads, FPGA proposes real concurrent and special processing, in which tasks do not compete for access to the resources [21].

This support of multithreading must, in theory, provide for the reduction of time needed to perform naturally parallel tasks. The practical grounding for these theoretical findings is a task of high actuality and, thus, applied in practice.

All of the above gives reason to believe that the efficiency' and performance' research of a software and hardware computational complex based on FPGA [21] is an urgent task and can eliminate such problems in the field of image processing as:

– reducing of the image processing execution duration;

– computational device scalability;

– computational device portability.

Perspective studies related to special areas of image processing, such as machine vision for autonomous driving or medical image analysis, clearly shows the potential of using FPGA to accelerate processing in this domain area. In [20], attention is paid to the problem of implementation of the edge detection algorithm using Sobel operator. The corner cases of low data latency are mentioned as significant ones. Also, the implementation of Sobel algorithm is compared between general purpose CPU and Zynq-based FPGA platform. There is also the proposal of using reconfigurable facilities of the hybrid SoC, where the high-complexity image filtering was implemented via the hardware configuration [23].

The image defogging algorithms are covered in [24], where modern Vivado HLS tools are used for custom IP core generation. The aspects of HLS Synthesis for image processing are mentioned. Also, the output to a custom display with VDMA is implemented. The existing trends of using FPGA acceleration for cloud services allow us to implement the model of the top of multiple

FPGAs, which is known as FPGA-as-a-Service (FaaS). The topics related to the intercommunication delay between the nodes and reliability aspects are covered in [25]. Meanwhile, the aspects of embedded system reliability remain actual through the history of programmable logic. There are existing tools for simulation and failure detection of FPGA designs during the development process, but the practical aspects may differ from the theoretical ones. The experience in operating and analytical reliability assessment of FPGA is covered in depth in work [26], where the practical aspects are also mentioned in the summary.

Aim and tasks of the work

The aim of this study is to design a resilient Zynq7000 SoC-based hardware accelerator for further research into the processing speed of a programmable logic computing device exemplified by image preprocessing tasks.

The designed software and hardware complex should be resilient in further exploitation and development. In the case of the proposed system, resilience can be defined like the ability to continue and process the given information under adverse conditions or stress, even if debilitated state is reached.

Achieving the set aims requires the following tasks to be solved:

– study the interaction of SoC system components;

– design a Zynq7000 SoC-based hardware accelerator;

– design FPGA-based system architecture in the real-time;

– implement SoC-based image processing algorithms in Verilog on a single-processor computer and in a massively parallel system with high processing quality preservation;

– conduct the analysis of research findings.

Therefore, concerning the capacity of the selected system, it is possible to set certain requirements for hardware accelerator features, including:

– program interfaces for the interaction of a processing subsystem (PS) and programmable logic (PL);

– ability to upload images in PL RAM;

– display of the processed image via HDMI video interface;

– real-time PL configuration updatability.

Materials and methods

The following image processing tasks were selected for the experiments: noise suppression, image transformation into the grayscale, binarization of the grayscale image, and morphological transformation [1]. A selection criterion is the natural parallelism of most studied methods and their frequent use.

Testing is performed on computing devices with varying architectures, namely:

– single-processor computing system;

– massively parallel computing system;

– field-programmable gate array.

The research of the performance of the single-processor computing system was conducted on the basis of developed software using the C ++ language and the OpenCV library (version 4.2.0). The graphical user interface was built on QtWidgets. A computer with AMD A6-7310 2.4 GHz processor (RAM 8 GB 2600 MHz) was used to test the performance of the selected digital image processing algorithms on the CPU in sequential mode.

To measure the serial algorithm time execution, the capabilities of a standard library were used. The determination of the execution duration is shown in program listing 1.

```
int
DeviceCPU::MeasureExecution(std::function<
void()> func) {
    using namespace std::chrono;
    using timestamp =
time_point<high_resolution_clock>;
    timestamp t_before =
high_resolution_clock::now();
    func(); // execute algorithm
    timestamp t_after =
high_resolution_clock::now();
    return duration_cast<milliseconds>(t_after
- t_before);
}
```

Listing 1. Sequential CPU timing

The research of the performance of the massively-parallel computing system was conducted on the basis of developed software using the CUDA technology (CUDA Toolkit 10.0 for the minimal compute capability 3.0 and the driver version 411.31) and OpenCV library with additional set for C and C ++ development.

To measure the parallel algorithm time execution, the capabilities of CUDA-events were used. The execution time determination is shown in program listing 1 and 2.

To design a FPGA-based image processing system, the “B-PDPR: Follow-behavior” pattern was used, and the “A-PDPR: Data-exchange” architectural pattern was used for interaction between program tasks.

The architectural part presents four tasks, replacing each other, depending on the current state. The data stream is formed sequentially for each item. Once the data have been processed, they are transferred to the next task (Fig. 4 and 5).

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

```
int
DeviceGPU::MeasureExecution(std::function
<void()> cuda_f){
    float elapsed=0;
    cudaEvent_t start, stop;
    HANDLE_ERROR(cudaEventCreate(&start));
    HANDLE_ERROR(cudaEventCreate(&stop));
    HANDLE_ERROR(cudaEventRecord(start,
nullptr));
    cuda_f(); // execute
    HANDLE_ERROR(cudaEventRecord(stop,
nullptr));
    HANDLE_ERROR(cudaEventSynchronize (stop));
    HANDLE_ERROR(cudaEventElapsedTime(&elapsed,
start, stop));
    HANDLE_ERROR(cudaEventDestroy(start));
    HANDLE_ERROR(cudaEventDestroy(stop));
    return static_cast<int>(elapsed);
}
```

Listing 2. Parallel GPU timing

The engineered architecture for real-time FPGA image processing allows you to perform the necessary calculations without worrying about the current state of the program and reconfiguration. The system can be easily expanded to more algorithms if needed.

Design of the programmable gate array computing device

In compliance with these requirements, the designed device has program interfaces necessary for the interaction of the microprocessor control unit with programmable logic, which facilitates uploading images to PL memory. In addition, the processed image output via HDMI video interface and real-time PL configuration is available.

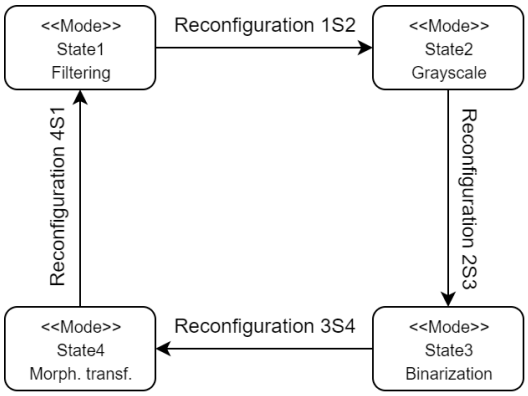


Fig. 4. "B-PDPR: Follow-behavior" template state change diagram

The dataflow chart inside the SoC system is shown in Fig.6. The following IP blocks by Xilinx were used in the design (Fig. 7):

– ZYNQ7 Processing System is responsible for PL+PS integration and peripherals configuration;

– Block Memory Generator provides an interface for accessing BRAM;

– AXI BRAM Controller and AXI SmartConnect enable PS interaction with BRAM via AXI bus line.

Processor System Reset and Clocking Wizard – generate clocking and asynchronous reset signals.

Within the implementation of the median filter, a scheme allowing input of two vectors, which represent pixel component values concerning the color of a digital image and communicate them to the outputs depending on their size, was developed for comparing every matrix element (Fig. 8).

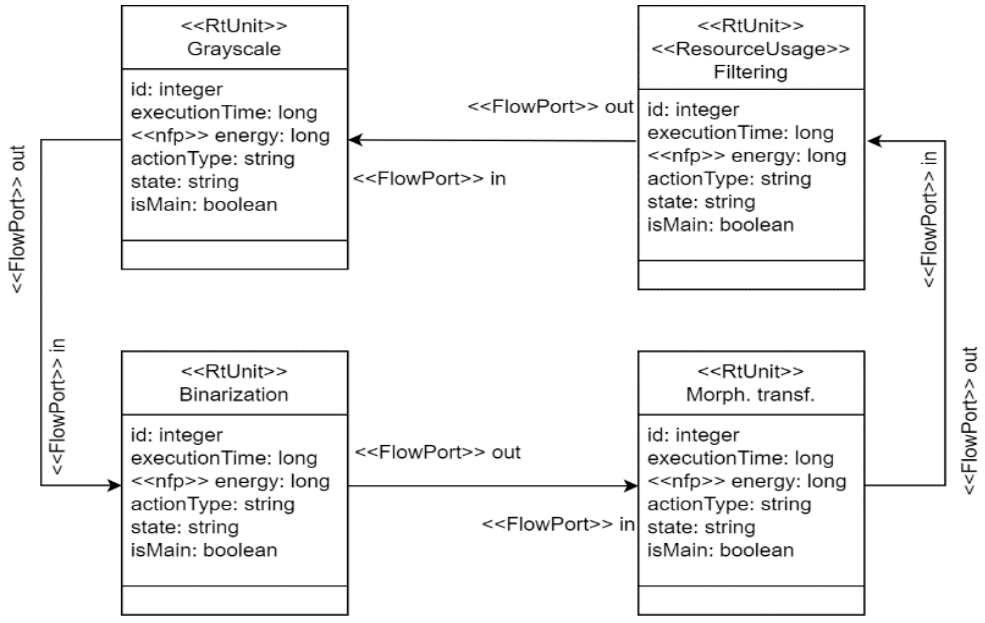


Fig. 5. "A-PDPR: Data-exchange" template state change diagram

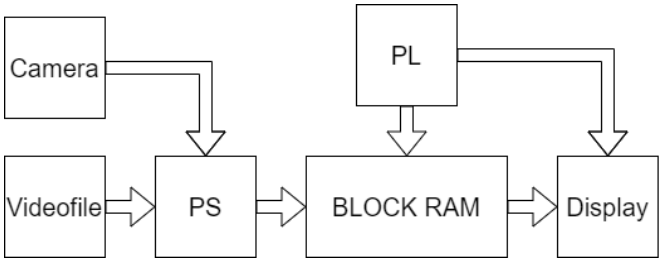


Fig. 6. Design Dataflow Chart

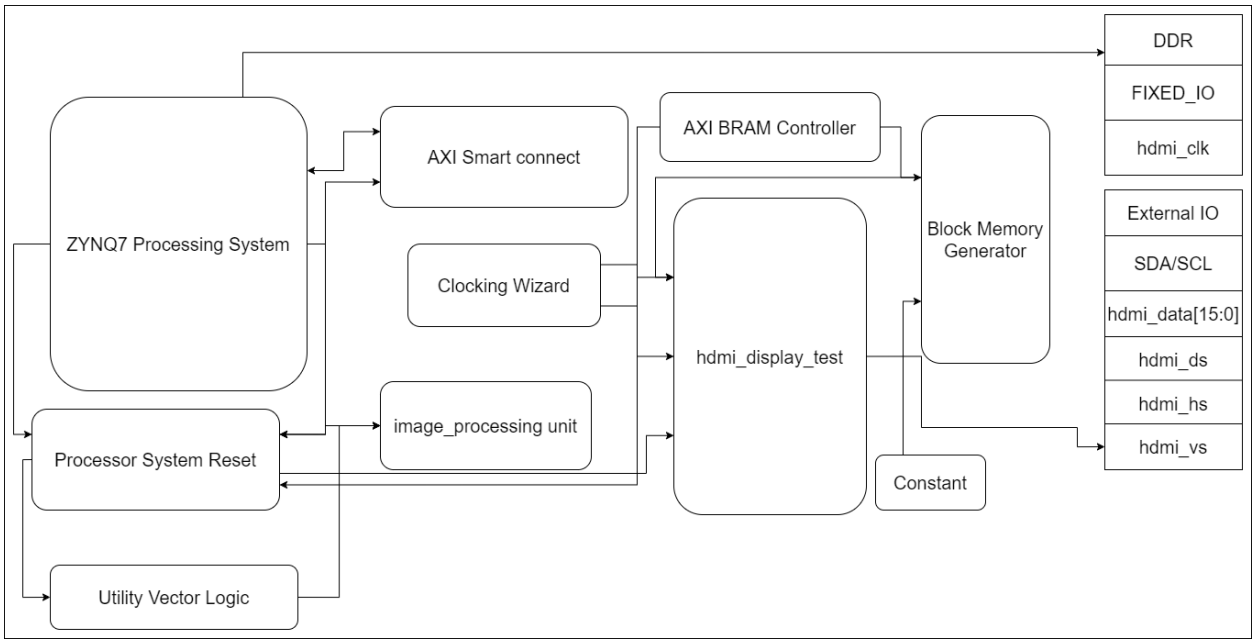


Fig. 7. General Design Structure Chart

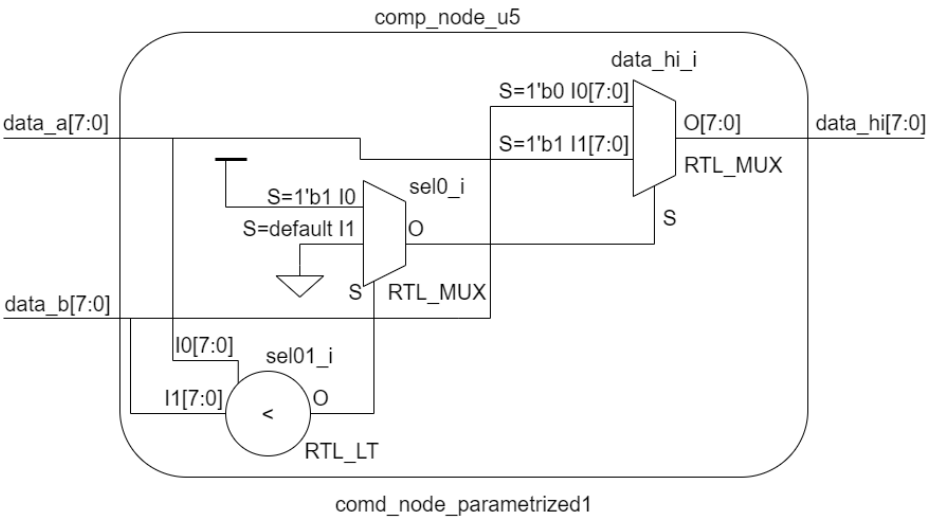


Fig. 8. RTL Scheme for Designed Element

The median filter is used in digital image and signal processing in the process of digital filtering for noise suppression. Input data are a two-dimensional matrix, the samples from which are arranged in ascending or descending order, and the value within the ordered list enters the filter output.

Nine plus one elements of this kind are needed for calculating the median value for a 3x3-sized matrix (Fig. 9).

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

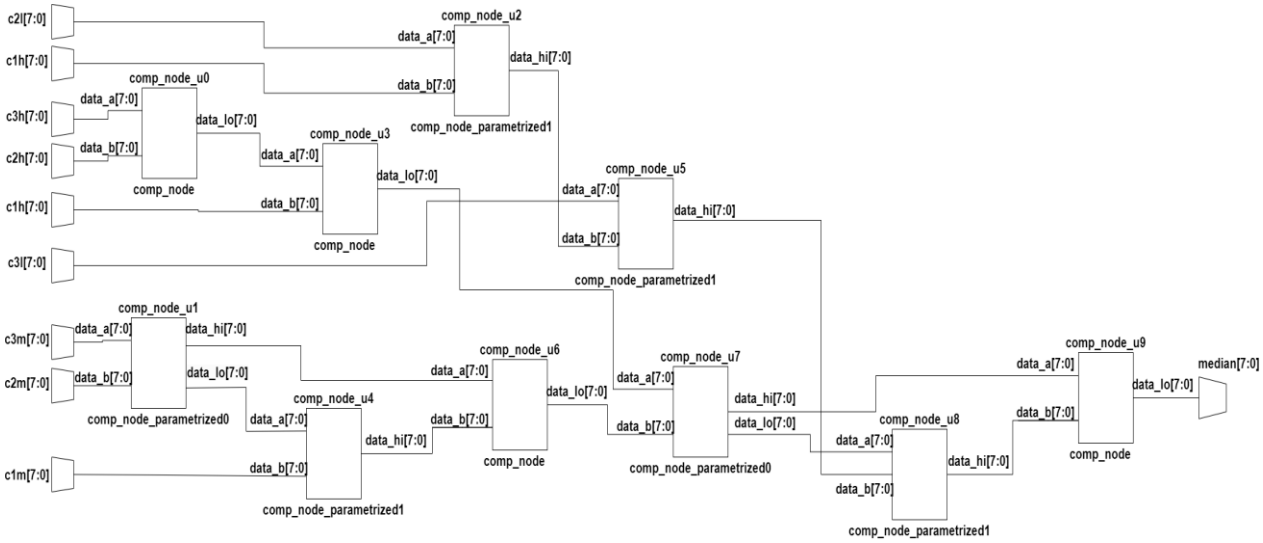


Fig. 9. RTL Scheme of Device for Median Value Calculation

Noise suppression from the color image requires the definition of the median value with the account taken of every color channel (RGB). Therefore, the whole algorithm does not require additional work cycles and is executed via a combinational circuit. The end device is shown in Fig. 10. The simulation findings prove that processing of one image pixel in the filter takes one clock signal pulse (Fig. 11).

To transform a color image into a grayscale image, finding the arithmetic mean of the channels for every pixel is enough.

After the vector division operation, the total amount of clocking signal cycles necessary for obtaining the result equals 2 (see Fig. 11).

To transform a grayscale image into a binary image, the thresholding algorithm is applied. Its main idea lies in one or several threshold values. If the input numeric color value exceeds the threshold value, the output color is white, if it does not, the output color is black (see Fig. 11, b).

Development of image output

Xilinx Zynq7020-based Z-turn Board has SiI9022A video transmitter controlled with PL (Fig. 12). The transmitter receives RGB888 (24bit) formatted data, but the board design restricts the use of data lines and is set for the work with RGB565 (16bit) format.

Interaction with the transmitter requires the use of two clock domain signals (see Fig. 12 and Fig. 13). Data and the address are generated separately and I2C protocol is used for operation with VDMA.

ARM core in Zynq7020 system is operated by OS Linux. Block RAM of SoC memory is accessed via a special device file, thus, data can be sent and received using a memory-mapped file region.

OpenCV library was used to simplify digital image processing in PS. The result of software to be used in PS retrieves each image from the video file (or a camera) frame by frame and transmits it to PL for further processing and display on the screen. The main indicator of the developed device quality assessment is the data processing speed compared with the program solutions.

The findings made after the application of the abovementioned methods are provided in Fig. 14.

Experiment, Results and Discussions

Research was conducted on the basis of the developed software with the use of C++ and OpenCV library (version 4.2.0).

A computer with the following configuration in order to test the speed of the selected digital image processing algorithms on the CPU: AMD A6-7310 2.4 GHz processor (8 GB 2600 MHz RAM, OC Windows 10 Home x64).

Median filter was selected as the algorithm for color image filtering with the time slot and search box size is 3x3 pixels.

Thresholding algorithm was applied to convert the filtered color image into a grayscale image and further into a binary image with the starting brightness value of 50.

Morphological transformations included opening and closing operations with 10x10 pixel cores. The obtained results for color images with various resolutions

after progressive processing by every algorithm are presented in Table 1. The table contains the execution time for the selected algorithms with the different input image size. The measurement unit is milliseconds. The table row contains the image input size and 4 algorithms runtime duration on CPU.

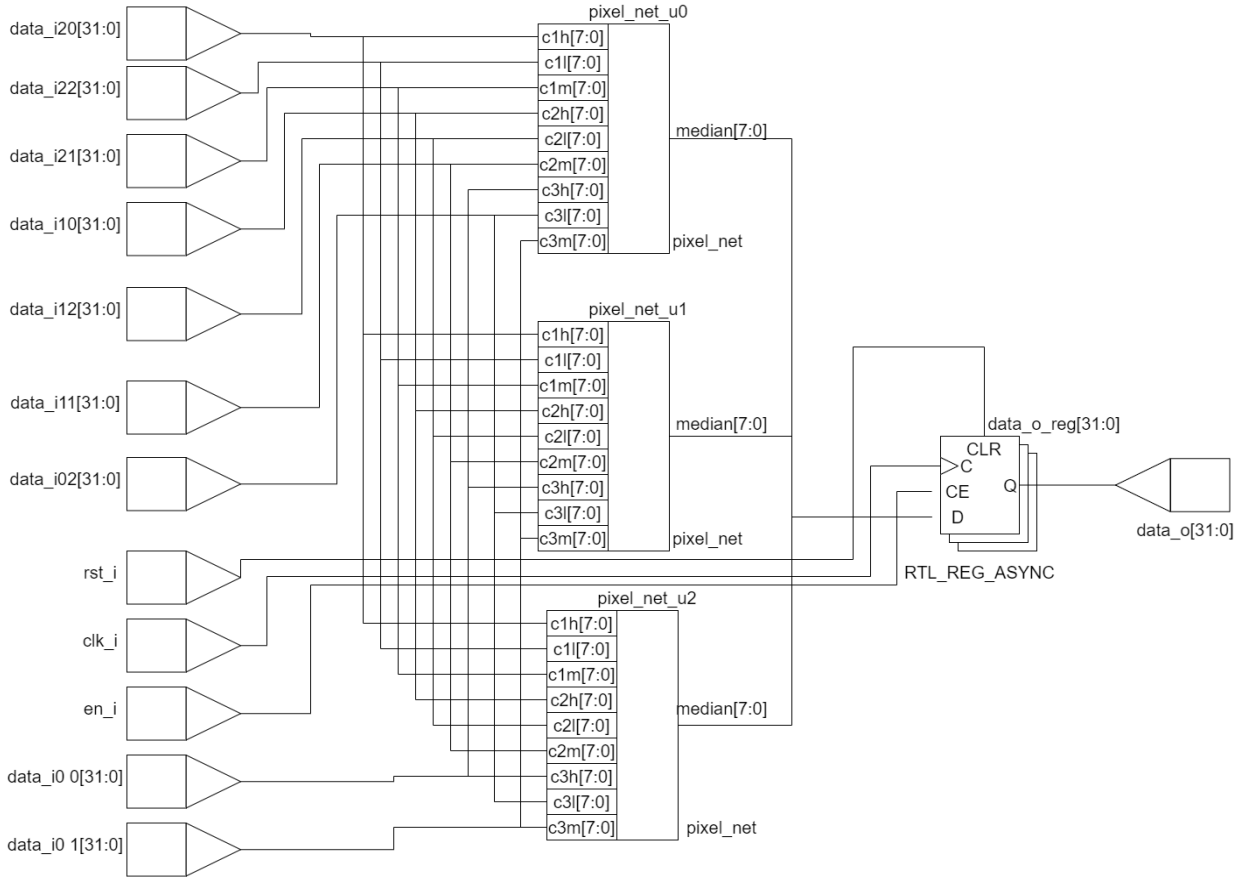
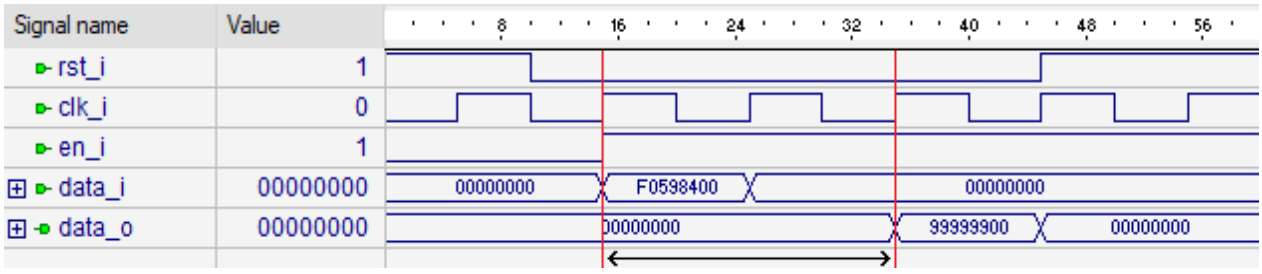
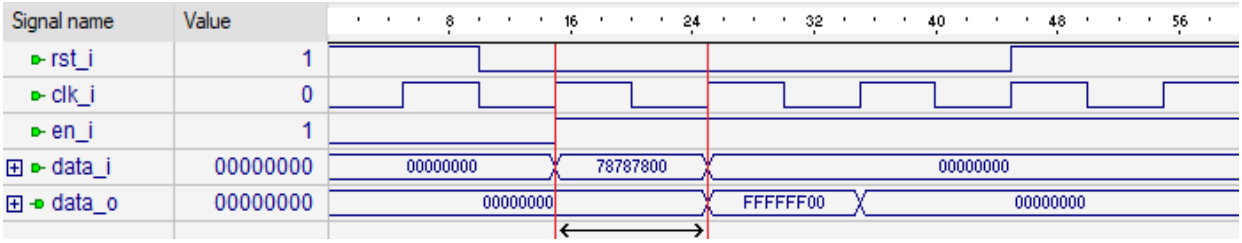


Fig. 10. RTL Scheme for 3x3 Median Filter



a)



b)

Fig. 11. Software-Based Simulation Findings:
a) for converting to grayscale image, b) for converting to binary image

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

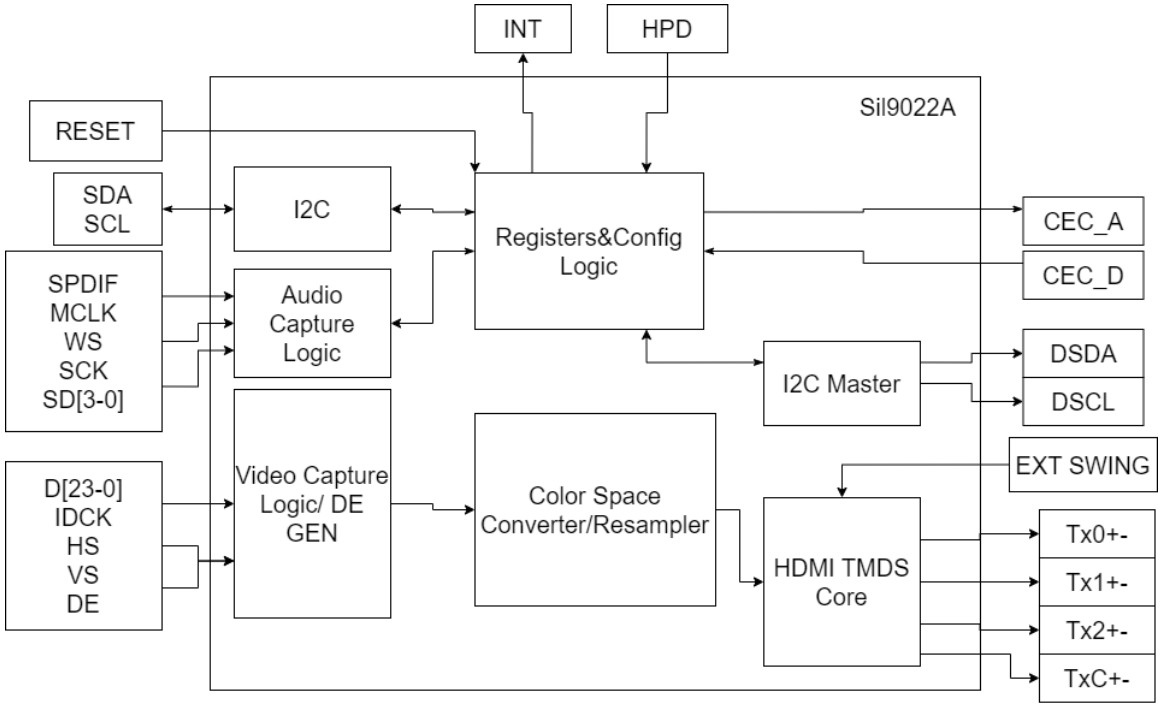
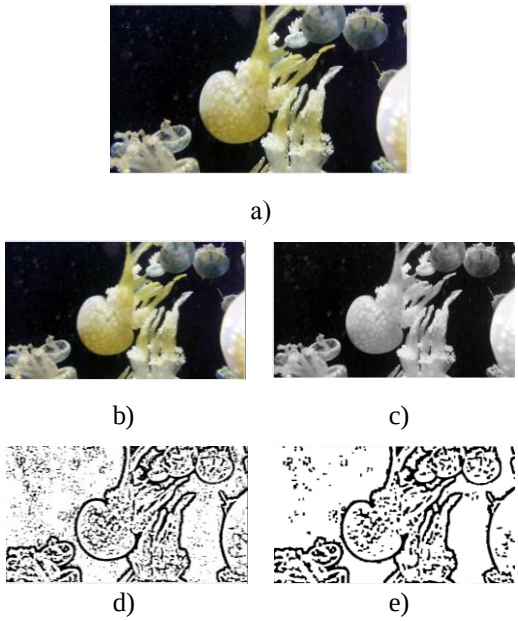


Fig. 12. Core Transmitter Structure Chart

Output Clock	Port Name	Output Freq (MHz)	
		Requested	Actual
clk_out1	clk_out1	100.000	100.000
clk_out2	clk_out2	148.500	148.438

Fig. 13. Clocking Signals Settings



a)

b) c)

d) e)

Fig. 14. Results of application of the tested digital image preprocessing methods: a) output image; b) the result of noise suppression; c) grayscale image; d) binary image; e) the result of morphological transformation

Table 1

Results of image processing algorithms on CPU

Input image size	Algorithm time (ms)			
	Image filter	Grayscale	Binarization	Morph transf.
1920x1080	723	110	23	850
1280x720	322	42	10	380
640x480	163	18	4	130
320x240	84	10	1	34

NVIDIA CUDA, which provides for the full range of applications for development in C and C++ was To ensure simplicity, an OpenCV library enhancement, which encompasses various extensions on CUDA (CUDA Toolkit 10.0), was applied.

All algorithms applied in program implementation on GPU fully comply with the analogs on CPU. All input data and parameters come in the same way and have equal values. Output data are provided in Fig. 14.

Testing was conducted on NVIDIA Quadro K2000M GPU. The device has 384 CUDA cores and 2 GB DDR3 global memory with 128-bit bus width. The shared memory size is 48Kb, the clock rate is 750 MHz.

Data allocation for the algorithms was done in global memory. The time slot for data allocation and copying into GPU was neglected in the end values. The time of the regarded methods' implementation is shown in Table 2. The table structure is similar to Table 1, but the results are shown for the same algorithms, but executed on GPU.

Table 2

Results of image processing algorithms on GPU

Input image size	Algorithm time (ms)			
	Image filter	Grayscale	Binarization	Morph transf.
1920x1080	86	3	1	23
1280x720	45	2	1	14
640x480	22	1	1	9
320x240	14	1	1	8

Obtained Results

To estimate the image processing module execution speed, start_i and done_o signals were used. Therefore, having calculated the number of clocking signal pulses between these two events, it was possible to obtain the time necessary for full processing of one image. The comparison results are presented in Table 3 and Figures 15-17.

Table 3

Execution Time Comparison Chart

1920x1080	Execution time (ms)		
	CPU	GPU	FPGA
Image filter	723	86	12
Grayscale	110	3	4
Binarization	23	1	2

As metrics for the analysis of the results obtained, we used the practical speedup of the studied image processing methods on the designed and developed software and hardware complex based on FPGAs with respect to sequential implementation on the CPU and parallel implementation on the GPU. The speedup was calculated in accordance with:

$$S_{FPGA(base)} = \frac{T_{base}}{T_{FPGA}},$$

where T_{base} is the task execution time on the CPU or GPU, T_{FPGA} is the task execution time on the developed FPGA-based computing system.

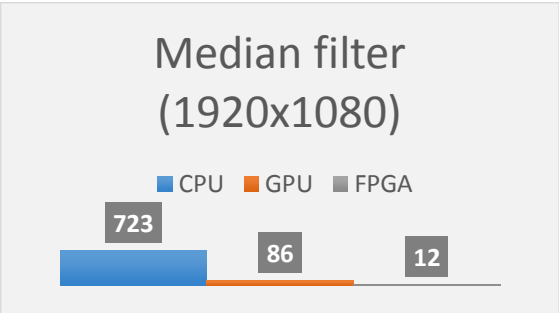


Fig. 15. Filtering Algorithm Running Speed Comparison

The implementation of the existing sequential image filtering algorithm based on the median filter takes a significant amount of time ($T_{CPU} = 723$ ms), as shown on the blue bar of the chart (Figure 15). The specialized graphics processor provides an execution time $T_{GPU} = 86$ ms. Thus, the highest obtained speedup of the developed software and hardware computing complex based on FPGA leads to $S_{FPGA(CPU)} = 60.25$ relative to the sequential execution time on the CPU.

The implementation of sequential converting an image to grayscale takes a significant amount of time ($T_{CPU} = 110$ ms), as shown on the blue bar of the chart (Figure 16).

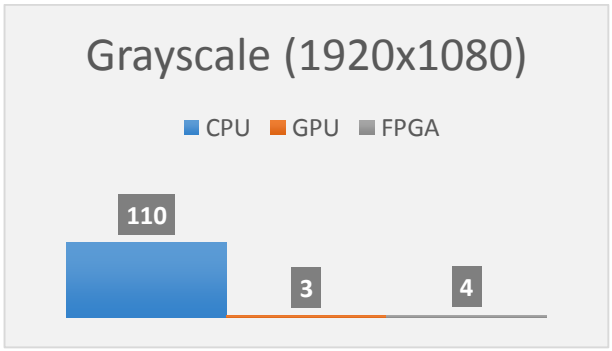


Fig. 16. Color Conversion Algorithm Running Speed Comparison

The specialized graphics processor provides an execution time $T_{GPU} = 3$ ms. The developed software and hardware computing complex based on FPGA provides an execution time $T_{FPGA} = 4$ ms and does not give the highest speedup for color conversion algorithm running. This is due to the architectural features of the NVidia company graphics processors.

The situation is similar to binarization algorithm running (Fig.17). The time of the binarization algorithm execution on the GPU and on the developed computer complex differs by 1ms and show an acceleration of almost 20 times.

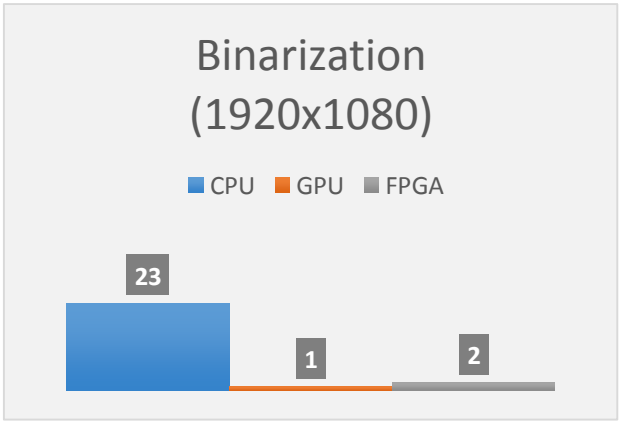


Fig.17. Binarization Algorithm Running Speed Comparison

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

A comparative analysis was conducted for the results of algorithm running speed testing on various input data, which demonstrated the significant advantage of hardware acceleration of image processing over their software analogs.

Conclusions

The result of the work is the research conducted into the capabilities of a SoC Zynq7000-based hardware accelerator on the example of image preprocessing compared with massively parallel systems and parallel shared memory computing systems.

The advantage of the designed software and hardware complex with programmable logic is the ability to load images to FPGA RAM using the resource of ARM core for further processing and display via HDMI video interface, in which PL configuration can be changed at any time within the processing process, and this makes the system resilient in further exploitation and development. For the designed software and hardware, complex resilience can be defined as the ability to continue and process the given information under adverse conditions or stress, even if a debilitated state is reached. The results of the execution are well defined among the system launches, and essential operation capabilities are maintained. The system is extensible by using partial reconfiguration facilities, which add an extra isolation layer from the architectural and development perspective.

A comparative analysis was conducted for the results of algorithm running speed testing on various input data, which demonstrates the considerable superiority (by over 60 times) of hardware image processing acceleration over their software analogs.

In conclusion, it should be noted that the advantage of the developed software and hardware computing complex based on FPGAs is that it is easily expandable and reconfigurable for most image processing methods. However, just for the problem of filtering based on the median filter, the proposed and developed system gives a high acceleration, reaching up 60 times. This confirms the need for hybrid computing to achieve maximum efficiency in computer graphics problem execution.

The obtained results can be applied in the development of embedded SoC-based solutions that require big data processing acceleration.

Thus, the work shows that the solution of the problem posed on the basis of FPGA is an alternative to special-purpose integrated circuits and can reduce image processing time execution. This is important in many fields of science and technology where the time of filtering, transformation and binarization of the image stream is critical (for example, processing information from satellites when compiling maps of the area, identifying objects in computer vision systems, analyzing the kinematics of human body movement in medical systems, and so on). Compared to other hard-coded logic solutions, FPGAs have the advantage of being flexible and programmable in nature. Also, most modern FPGA devices support dynamic partial reconfiguration, which enables the possibility of performance increasing in comparison to computing on a conventional computer.

Acknowledgement

This work was supported by the Slovak Research and Development Agency under the grant ‘‘New methods development for reliability analysis of complex system’’ reg.no. APVV-18–0027

Contribution of authors: formulation of the problem – **Olesia Barkovska, Inna Filippenko, Peter Sedlaček;** hardware platform investigation and development tools selection – **Valentyn Korniienko, Ivan Semenenko;** C++ software algorithms realization – **Valentyn Korniienko;** FPGA algorithms realization – **Ivan Semenenko;** analysis and processing of the obtained results – **Olesia Barkovska, Inna Filippenko;** finalization of the draft article version – **Inna Filippenko, Peter Sedlaček;** corrections and postediting **– Olesia Barkovska, Ivan Semenenko, Peter Sedlaček.**

All authors have read and agreed to the published version of the manuscript.

References

1. Barkovskaya, O. & Axak, N. Contrastive Analysis of the Parallel Version of the Binary Image Skeletonization Algorithms on Basis of Binary Matrix and Structural Elements. *9th International Conference - The Experience of Designing and Applications of CAD Systems in Microelectronics*, Lviv, Ukraine, 2007, pp. 435-436. DOI: 10.1109/CADSM.2007.4297609.

2. Gonzalez, C. & Woods, R. E. *Digital Image Processing. Instructor's Manual.* 3rd Edition. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2007. 976 p.

3. Barkovska, O., Axak, N., Rosinskiy, D. & Liashenko, S. Application of mydriasis identification methods in parental control systems. *IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, 2018, pp. 459-463. DOI: 10.1109/DESSERT.2018.8409177.

4. Barkovska, O., Movsesian., I., Yeromina, N., Liashenko, O. & Tkachenko, D. System of individual multidimensional biometric authentication. *International Journal of Emerging Trends in Engineering Research*, 2019, vol. 7, iss. 12, pp. 812–817. DOI: 10.30534/ijeter/2019/147122019.

5. Dagum, L. & Menon, R. OpenMP: an industry standard API for shared-memory programming. *IEEE*

Computational Science and Engineering, vol. 5, iss. 1, pp. 46-55, Jan.-March 1998. DOI: 10.1109/99.660313.

6. Oden, L. Lessons learned from comparing C-CUDA and Python-Numba for GPU-Computing. *28th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, Västerås, Sweden, 2020, pp. 216-223. DOI: 10.1109/PDP50117.2020.00041.

7. Monmasson, E., Idkhajine, L. & Naouar, M. W. FPGA-based Controllers. *IEEE Industrial Electronics Magazine*, vol. 5, no. 1, pp. 14-26, March 2011. DOI: 10.1109/MIE.2011.940250.

8. *Z-turn Board V2 (with Zynq-7020)*. MYIR Tech Limited. Available at: https://www.xilinx.com/products/boards-and-kits/1-571ww1.html (accessed 12.12.2022).

9. Taylor, A. *The Zynq PS/PL, Part One: Adam Taylor's MicroZed Chronicles Part 21*. Available at: https://support.xilinx.com/s/article/418935?language=en_US (accessed 12.12.2022).

10. Flynn, M. J. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*, Sept. 1972, vol. C-21, iss. 9, pp. 948-960. DOI: 10.1109/TC.1972.5009071.

11. Park, I. K., Singhal, N., Lee, M. H., Cho, S. & Kim, C. Design and Performance Evaluation of Image Processing Algorithms on GPUs. *IEEE Transactions on Parallel and Distributed Systems*, Jan. 2011, vol. 22, no. 1, pp. 91-104. DOI: 10.1109/TPDS.2010.115.

12. Usha, R., Pandey, P. & Mangala, N. A Comprehensive Comparison and Analysis of OpenACC and OpenMP 4.5 for NVIDIA GPUs. *IEEE High Performance Extreme Computing Conference (HPEC)*, 2020, pp. 1-6. DOI: 10.1109/HPEC43674.2020.9286203.

13. Miroshnikov, A. S., Berko, I. A. & Berko, A. A. Optimization Method for Parallel Algorithm for Face Recognition in Graphic Images. *International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, 2021, pp. 729-735. DOI: 10.1109/ICIEAM51226.2021.9446397.

14. Rakhimov, M., Mamadjanov, D. & Mukhiddinov, A. A High-Performance Parallel Approach to Image Processing in Distributed Computing. *IEEE 14th International Conference on Application of Information and Communication Technologies (AICT)*, 2020, pp. 1-5. DOI: 10.1109/AICT50176.2020.9368840.

15. Idzenga, T., Gaburov, E., Vermin, W., Menssen, J. & De Korte, C. L. Fast 2-D ultrasound strain imaging: the benefits of using a GPU. *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 61, no. 1, pp. 207-213, January 2014. DOI: 10.1109/TUFFC.2014.2893.

16. Yokota, T., Nagafuchi, M., Mekada, Y., Yoshinaga, T., Ootsu, K. & Baba, T. A scalable FPGA-based custom computing machine for a medical image processing. *10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, Napa, CA, USA, 2002, pp. 307-308. DOI: 10.1109/FPGA.2002.1106695.

17. Mamatha, G., Sumalatha, V. & Lakshmaiah, M. V. FPGA implementation of satellite image fusion using wavelet substitution method. *Science and Information Conference (SAI)*, London, UK, 2015, pp. 1155-1159. DOI: 10.1109/SAI.2015.7237290.

18. Shandilya, R. & Sharma, R. K. FPGA implementation of image enhancement technique for Automatic Vehicles Number Plate detection. *International Conference on Trends in Electronics and Informatics (ICEI)*, Tirunelveli, India, 2017, pp. 1010-1017. DOI: 10.1109/ICOEI.2017.8300860.

19. Dhaussy, P., Filloque, J., Pottier, B. & Rubini, S. Global control synthesis for an MIMD/FPGA machine. *Proceedings of IEEE Workshop on FPGA's for Custom Computing Machines*, Napa Valley, CA, USA, 1994, pp. 72-81. DOI: 10.1109/FPGA.1994.315603.

20. Fan, R. & Yamaguchi, Y. A study of FPGA-based cluster computing by high-speed serial-link communication. *Eighth International Symposium on Computing and Networking Workshops (CANDARW)*, 2020, pp. 401-405. DOI: 10.1109/CANDARW51189.2020.00082.

21. Li, B., Chen, J., Zhang, X., Xu, X., Wei, Y. & Kong, D. A design of zynq-based medical image edge detection accelerator. *6th international conference on biomedical signal and image processing (ICBIP '21), August 20 - 22, 2021*, Suzhou China, New York, NY, USA: ACM, pp. 59-64. DOI: 10.1145/3484424.3484434.

22. Liu, J. & Feng, J. Design of embedded digital image processing system based on ZYNQ. *Microprocessors and microsystems*, 2021, vol. 83, article no. 104005. DOI: 10.1016/j.micpro.2021.104005

23. Zhang, C., Bi, S., Jiang, T., Wang, J. & Mao, W. Implementation of ZYNQ for image defogging. *IEEE 9th joint international information technology and artificial intelligence conference (ITAIC)*, 11-13 December 2020, Chongqing, China, 2020, pp. 1971-1977. DOI: 10.1109/itaic49862.2020.9339196.

24. Perepelitsyn, A., Kulanov, V. & Zarizenko, I. Method of QoS evaluation of FPGA as a service. *Radioelectronic and Computer Systems*, 2022, no. 4, pp. 153-160. DOI: 10.32620/reks.2022.4.12.

25. Babeshko, E., Kharchenko, V., Leontiiev, K. & Ruchkov, E. Practical aspects of operating and analytical reliability assessment of FPGA-based I&C systems. *Radioelectronic and computer systems*, 2020, no. 3, pp. 75-83. DOI: 10.32620/reks.2020.3.08.

Received 21.03.2023, Accepted 20.05.2023

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

АДАПТАЦІЯ FPGA АРХІТЕКТУРИ ДЛЯ ПРИСКОРЕННЯ АЛГОРИТМІВ ОБРОБКИ ЗОБРАЖЕНЬ

Олеся Барковська, Інна Філіппенко , Іван Семененко, Валентин Корнієнко, Петер Седлачек

Ця робота присвячена актуальній проблемі на перетині теорії зв'язку, цифрової електроніки та чисельного аналізу, а саме вимірюванню часу виконання методів обробки зображень на різних архітектурах обчислювальних пристроїв, які забезпечують програмне та апаратне прискорення. **Предметом** вивчення статті є дослідження реконфігурованих FPGA систем в області обробки зображень. **Метою** цієї роботи є створення перероблювальної системи обробки зображень на базі FPGA та порівняння її з існуючими архітектурами обробки. **Завдання:** Для виконання вимог роботи необхідно підготувати практичний експеримент, а також теоретичне дослідження запропонованої архітектури; дослідити процес створення системи обробки зображень на основі ZYNQ SoC; розробити та порівняти швидкість виконання для заданого набору алгоритмів із певним діапазоном роздільної здатності зображень. **Використані методи:** симуляція FPGA, паралельне програмування на C++, технологія NVIDIA CUDA, інструментарій для аналізу продуктивності виконання програм. **Результатом** цієї роботи є розробка стійкої обчислювальної системи на базі SoC Zynq7000 з програмованою логікою та можливістю завантаження зображень в RAM FPGA з використанням ресурсів ядра ARM для подальшої обробки та виводу через інтерфейс відео HDMI, що дозволяє змінювати конфігурацію PL у будь-який момент під час обробки. **Висновки.** Ефективність підходу FPGA порівнювали з реалізацією методів паралельної обробки зображень за допомогою OpenMP та CUDA. Представлено огляд платформи ZYNQ з конкретними деталями, пов’язаними з обробкою медіа. Аналіз результатів тестування швидкості алгоритму на основі різних виходів довів перевагу (у понад 60 разів) апаратного прискорення обробки зображень перед програмними аналогами. Отримані результати можуть бути використані при розробці вбудованих рішень на базі SoC, які вимагають прискорення обробки великих даних. Крім того, отримані результати можна використовувати під час пошуку відповідної вбудованої платформи для певного завдання обробки зображень, де висока пропускна здатність системи є однією з найбільш пріоритетних вимог.

Ключові слова: прискорення обробки даних; FPGA; висока продуктивність; паралельні системи; обробка зображень.

Барковська Олеся Юріївна – канд. техн. наук, доц., доц. каф. Електронних обчислювальних машин, Харківський національний університет радіоелектроніки, Харків, Україна

Філіппенко Інна Вікторівна – канд. техн. наук, доц., доц. каф. Автоматизації проектування обчислювальної техніки, Харківський національний університет радіоелектроніки, Харків, Україна

Семененко Іван Георгійович – магістрант каф. Автоматизації проектування обчислювальної техніки, Харківський національний університет радіоелектроніки, Харків, Україна

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

Петер Седлачек – д-р філос. з комп'ютерних наук, доц. каф. інформатики Жилінського університету, Жиліна, Словаччина.

Olesia Barkovska – PhD (Computer Sciences), Associate Professor of the Department of Electronic Computers, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: olesia.barkovska@nure.ua, ORCID: 0000-0001-7496-4353, Scopus Author ID: 24482907700.

Inna Filippenko – PhD (Computer Sciences), Associate Professor of 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.

Ivan Semenenko – master of the Design Automation Department, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine,
e-mail: ivan.semenenko@nure.ua, ORCID: 0000-0002-6498-2440, Scopus Author ID: 57352374800.

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.

Peter Sedlaček – PhD in Computer Sciences, Assistant Professor of the Department of Informatics, University of Zilina, Zilina, Slovakia,
e-mail: peter.sedlacek@fri.uniza.sk, ORCID: 0000-0002-7481-6905, Scopus Author ID: 57210661864