# Self-Improvable Computer System Model and Architecture Based on Reconfigurable Hardware, Automatic Design and Synthesis Tools and Artificial Intelligence Technologies

Anatoliy Melnyk[a,b] and Viktor Melnyk[a,b]

[a] *Lviv Polytechnic National University, Lviv, Ukraine,*
[b] *The John Paul II Catholic University of Lublin, Lublin, Poland*

**Abstract**
In the article the idea on self-improvable computer systems creation is formulated and directions of their self-improvement and the proposed approach to self-improvement are described. This approach is based on methods and tools for specialized processors high-level automatic design and synthesis, new computational models, including the one that uses principle of self-configuration, new architecture of self-configurable FPGA-based computer systems and computational intelligence technologies. An FPGA was chosen as the hardware basis of self-improvable computer systems. The directions of the FPGA-based reconfigurable computer systems development over the next years are considered. The self-improvable computer system operation model and architecture are proposed, and their components are described. Among them there are method of self-configuration of a computer system with reconfigurable logic, Artificial Intelligence technologies, and software basis of self-improvable computer systems. The core of this software are specialized processors high-level design tools, which are used as a basic mechanism for computer system self-improvement. Expected benefits of self-improvable computer systems creation are estimated at the end of the article.

**Keywords**
Self-improvable computer systems, reconfigurable computing, automatic design and synthesis tools, self-configurable computer systems, FPGA, Artificial Intelligence.

## 1. Introduction

At the junction of the development of reconfigurable computing and Artificial Intelligence a new scientific direction has emerged in recent years, and it was entitled with the term “Evolvable Hardware”. Under the term “Evolvable Hardware” is understood a hardware, that is able to permanently solely change its architecture and behavior, interacting with the environment it is placed in, possessing necessary for this embedded means for self-learning, self-optimization, self-debugging, self-improvement etc. with the aim of more effective on those or other criteria execution of the tasks assigned to them [1]. An ability to evolve in the evolvable hardware is ensured by the deployment of the evolutionary algorithms – the direction in Artificial Intelligence, which is used for evolutionary modelling. These algorithms simulate processes that reflect the basic provisions of the theory of biological evolution - the processes of selection, recombination, mutation and reproduction [2], [3].

The base of the evolvable hardware is reconfigurable logic devices, structure and performed functions of which can be changed in accordance with the task requirements. Today reconfigurable hardware means are mainly implemented in the form of semiconductor integrated circuits entitled FPGAs – Field Programmable Gate Arrays [4]. A direction of research and engineering activities on the creation and application of reconfigurable logic devices in computer technology is called Reconfigurable computing [5], [6], while computer systems based on these devices are called reconfigurable computer systems (RCCS) [7], [8].

CMIS-2021: The Fourth International Workshop on Computer Modeling and Intelligent Systems, April 27, 2021, Zaporizhzhia, Ukraine
EMAIL: anatoliy.melnyk@kul.pl (Anatoliy Melnyk); viktor.melnyk@kul.pl (Viktor Melnyk)
ORCID: 0000-0002-8981-0530 (Anatoliy Melnyk); 0000-0002-5046-8002 (Viktor Melnyk)

RCCSs are able to perform high-performance computations owing to high performance characteristics of their processing components – FPGAs, and advances in design methodology of specialized processors, which are synthesized in FPGA.

The direction of the reconfigurable computing is based on the idea of computer hardware adjustment to perform the specific computational tasks. Feasibility of the reconfigurable computing approach in the real computer systems became possible with the advent of modern FPGA with high integration degree. FPGAs are such integrated circuits that can store and, when it is needed, can change the configuration. After loading the configuration into the FPGA chip, it turns into the specialized computing module, for example, into the hardware-oriented specialized processor.

Combining in the computer system general-purpose processors with specialized processors implemented in reconfigurable logic allows one to increase its overall performance by 2-3 orders of magnitude [9]. The functional commitment of RCCS can be changed with preserving its high performance on another class of problems by implementing in reconfigurable logic of a new specialized processor with a corresponding structure and functionality.

However, along with high performance, that is the advantage of RCCSs, there are also challenges associated with their use. These challenges are the significant timing expenses for load balancing between the processing units and often unavailability of previously designed specialized processor IP cores, required for implementation in the reconfigurable logic. It forces to develop them from scratch. Also, there are high additional requirements to the qualification of RCCS operators, as they, besides modeling and programming, have to perform system analysis, design of the specialized processors architecture, their synthesis and implementation in reconfigurable logic.

In the self-configurable computer systems (SCCS), these labor-intensive and time-consuming operations are fully automated and replaced from an operator to a computer system. The method of self-configuring allows one to exploit all the potential provided by the property of reconfiguration and thus ensures one of the leading places for SCCS in high-performance computing means.

## 2. Problem Statement

The beginnings of Evolvable Hardware research date back to the mid-90s, when the first attempts were made [10], [11] to create evolvable hardware based on Xilinx FPGA. This was preceded by the publication of several theoretical works, in particular [12], [13], which described the conceptual basis of Evolvable Hardware. Subsequently, a number of scientific papers were published, including articles and monographs describing theoretical research, and some practical works in the direction of Evolvable Hardware [14-21], a number of scientific conferences and seminars, including a series of such conferences: IEEE International Conference on Evolvable Systems, IEEE International Conference on Self-Adaptive and Self-Organizing Systems, Evolvable Systems: From Biology to Hardware; NASA/DoD Conference on Evolvable Hardware.

Along with significant progress in the developments in the reconfigurable computing, today there is some "slack" in the direction of Evolvable Hardware. The reason is, most of all, in the absence of significant, fundamental steps in the development of computer systems and artificial intelligence systems, such that would change approaches to their design, information processing methods they use, methods of tasks execution, their architecture. In fact, if to go a little away from the subject of Evolvable Hardware and reconfigurable computing, and analyze actual developments in computer technology in general, we can conclude, that despite the permanent technological progress (improvement of integrated technologies), methodological advances (multiple cores in a processor, multiple threads in a process), computer architecture in general, remains unchanged and continues to be based on the computing model proposed by John von Neumann more than 70 years ago.

So, it is needed to search for a new theoretical basis, which will become a platform for future developments in the Evolvable Hardware direction, on which ideas and statements of this direction will obtain an effective embodiment and will give an opportunity to make significant progress in the development of computer technology.

In this article, the authors estimate the possibilities of creating an architecture of computer systems of a fundamentally new type - self-improvable CS, which are able to independently improve their technical characteristics and functionality.

## 3. The Idea on Self-Improvable Computer Systems and Directions of Their Self-Improvement

The conception of self-improvement has so far applied only to a human as a carrier of intelligence. Moreover, self-improvement is multifaceted and covers almost all areas of human activity. In particular, people train their bodies in order to achieve certain physical results, such as running speed, swimming, jumping range or throwing various shells, and so on. In this sense, activities in the field of physical culture and sports are aimed at self-improvement. It is common knowledge that trained people in their activities significantly outnumber the untrained ones, which is the result of their self-improvement.

Naturally, the highest level of self-improvement of a person is his education in its broadest sense: at home, at school and other educational institutions, in clubs, interest groups, independently, and so on. Education allows to realize to the greatest extent in the person its potential intelligence and is a basis of all-human progress.

Applying the above considerations and the conception of self-improvement to computer technology, we can formulate a definition of self-improvable computer system (SICS) as a computer system capable of solely (independently) improving its technical characteristics and functional capabilities.

It should be noted that SICS is a CS that has already been created and is in operation. SICS as technical systems should be able to improve their technical characteristics and functional capabilities just as a human is able to improve her physical results.

When we talk about the self-improvement of the CS, we mean first and foremost, the ability of the CS to increase the efficiency of use of equipment available in it during its operation, because its amount in the already created CS is constant. Therefore, the higher is the performance, the more efficiently this equipment is operated. Moreover, to increase efficiency of use of the equipment it is possible only by increase of performance of CS on certain classes of tasks. Taking into account the complexity of modern CS and their components, we can expect a significant (in times, if not dozens or hundreds of times) self-improvement of computer systems on task classes. After all, it is common knowledge that in some classes of tasks, due to the lack of harmonization of computer architecture to the features of these tasks, only units of the percentage of peak performance of modern supercomputers are used [22].

Another important technical characteristic that the CS could improve on its own is the value of its power consumption. After all, not all tasks solved in the CS require the operation of all components of the CS at the maximum frequency, when the power consumption is greatest. Therefore, by controlling the frequency of operation, the CS can reduce power consumption. In addition, significant energy can be saved by disabling the components of the CS that are not involved in certain tasks. That is, the prospects for self-improvement in this direction are also promising.

Is the list of those that the CS can improve by itself limited to these technical characteristics of the CS? Of course not. In particular, SICS can independently, if necessary, increase the accuracy of computations and the dynamic range of processed data, using pre-created virtual bit grids.

## 4. The Main Hypothesis of the Study and the Proposed Approach to Self-Improvement of the Computer System

In order to solve arbitrary problems, computer systems must be universal and based on a universal computational model. On the other hand, the use of specialized processors or specialized parts of heterogeneous chips speeds up the operation of universal computers and increases their efficiency because their hardware can be customized to the specifics of the computations [23, 24]. This explains the widespread use of specialized processors in the latest high-performance computer systems [25]. However, not always at the design stage of computer systems are known all the features of the problems that will be solved by them in the future, which does not allow to take into account these features in their architecture in order to increase efficiency. It should also be noted that the extremely high complexity of modern computer systems does not allow to provide the highest values of their technical characteristics at the design stage.

The main hypothesis of the study is that the introduction of self-improvement mechanisms into the architecture of computer systems should provide an opportunity at the stage of their operation to improve the efficiency of equipment usage by increasing performance on classes of algorithms up to 2-3 orders of magnitude, or to decrease the power consumption the same level down.

The question of the implementation of this hypothesis and the creation of self-improvable computer systems was made possible by the creation of methods and tools for high-level automatic design of specialized processors [26], new computational models, including the one that uses principle of self-configuration [27], new architecture of self-configurable FPGA-based computer systems [28, 29] and computational intelligence technologies [30, 31].

## 5. Hardware Basis of Self-Improvable Computer Systems

As mentioned above, the hardware basis of self-improvable CS are reconfigurable components – FPGAs, the structure and functions of which can be changed according to the requirements of the problem to be solved. FPGAs have begun to be used as elements for the implementation of computer devices quite long ago. However, they have become used in a high-performance computing relatively recently. Thus, the creation of reconfigurable high-performance computer systems is relatively new scientific and technical direction. There are many reconfigurable high-performance computer systems today, some of which can be classified as supercomputers [32-34]. We believe that the trend of reconfigurable technologies implementation in the high-performance computing will continue in the coming years. Along with this, we should expect further increase of level of the application-specific computing devices design and implementation in FPGAs, and simplification of the design process.

In our view, the development of reconfigurable computer systems over the next years will take place in several directions:

1. Through the developments in integrated circuit manufacturing technologies and FPGA architecture, that will improve their technical parameters (rise the performance, increase the amount of on-chip resources, decrease power consumption ratio).
2. Design technology and tools for computer devices implementation in FPGA will further develop, and will reach a level at which rapid automatic synthesis of highly optimized specialized processors for execution of algorithms based on their high-level representation, becomes a reality, and will largely replace the use of hardware description languages in computer devices design by the high-level programming.
3. The software will be created, that run on the level of operating system, that will enable computing load automatic balancing between fixed and reconfigurable components of a computer system, automatically manage the FPGA operation, in particular, carrying out their reconfiguring during the computing processes, and use FPGAs in multitask mode.
4. Application of the reconfigurable logic in computer systems will become common, including placing the reconfigurable logic resources into the same chip with the general-purpose processors, e.g., as we can already see in Intel Acquisition of Altera.
5. Efforts will be made in order to introduce self-monitoring, self-optimization and self-improvement technics into the reconfigurable computer systems (one such approach is shown in [35]).

The processing core of self-improvable computer system are general-purpose processor and FPGA-based reconfigurable environment (RCE) connected between themselves as shown in Figure 1. Here, the universal processor is aimed, among other things, to execute programs for the initial creation of hardware configurations that are implemented in the reconfigurable environment, and their improvement in the process of CS operation.

## 6. Methodological Basis of Self-Improvable Computer Systems

As already mentioned, the question of creation of self-improvable computer systems has become possible due to the creation in recent years of new methods of high-level automatic design of specialized processors, new computational models, including the one that uses principle of self-configuration, as a basis for self-improvement, and a new architecture of self-configurable FPGA-based computer systems, as well as the creation and development of modern computational intelligence technologies, including self-learning technologies, neural networks, genetic algorithms, collective intelligence, which can be used as a basis for self-improvement.

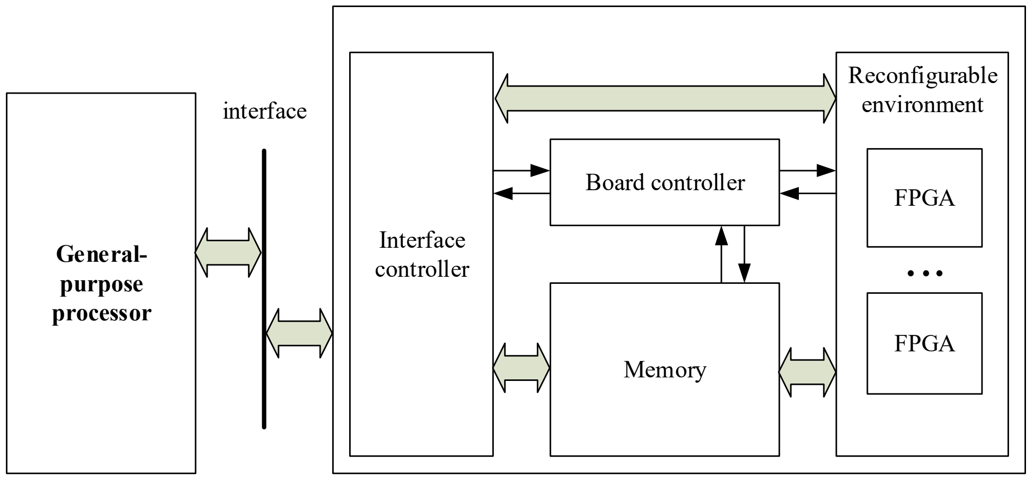


**Figure 1**: The processing core of self-improvable computer system

A description of the proposed mechanism of self-improvement of the CS together with the above-mentioned innovations used in this mechanism, are given below.

## 6.1. Model and Architecture of Self-Improvable Computer System

Given the iterative nature of self-improvement processes in general, we can determine the necessary conditions for self-improvement of the CS, namely:

- Knowledge of technical characteristics that need to be improved;
- Knowledge of ways to improve technical characteristics;
- Ability to measure the values of current technical characteristics and compare them with previous values;
- Availability of hardware, the structure and functions of which can be changed;
- Availability of methods and means of self-adjustment of the structure and functions of hardware by the computer system for self-improvement.

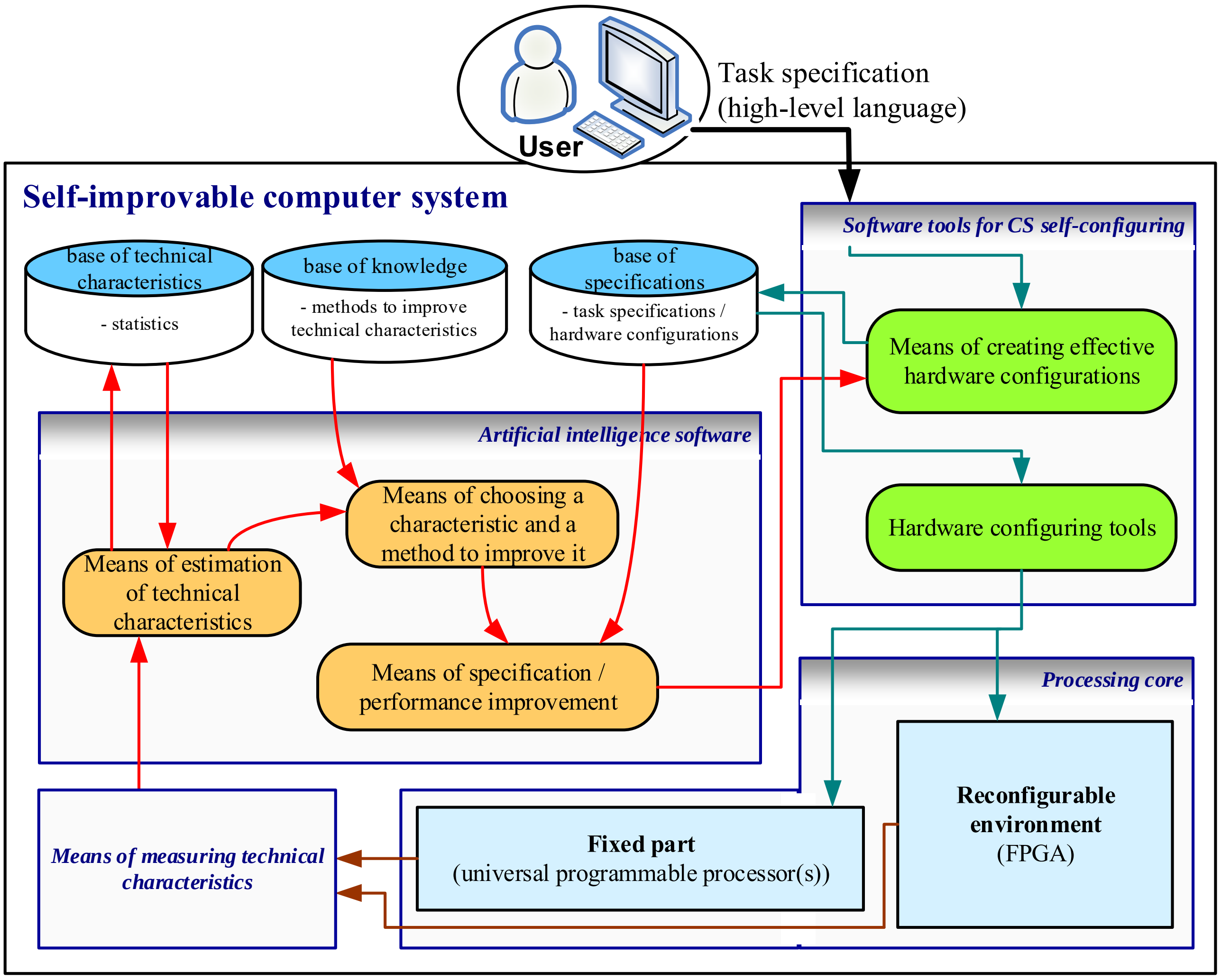
To meet these conditions in the CS at the design stage it is necessary to provide the ability to measure, store and compare its technical characteristics, which can be the subject for improvement, i.e., the efficiency of hardware use and power consumption, as well as to implement the means for the self-improvement to be performed.

Let's define the basic components of the CS, the availability of which will ensure the fulfillment of the above conditions and, accordingly, the possibility of its self-improvement.

1. the information base of technical characteristics that need to be improved, to which these characteristics are submitted and accumulated during the operation of the CS;
2. the information base of knowledge, in which algorithms and corresponding software to improve the technical characteristics of the CS should be stored;
3. the information base of specifications, in which the specifications of tasks and the corresponding descriptions of hardware configurations of CS should be stored;
4. hardware, which, in addition to a fixed part in the form of one or more universal programmable processors (UPP), contain a reconfigurable environment, the presence of which will provide the ability to specify the initial structure and functions of the CS and their subsequent changes to improve technical characteristics;
5. software tools for CS self-configuring, the goal of which is to automatically create a configuration of the reconfigurable hardware to perform user tasks;
6. means of measuring technical characteristics of CS during the execution of user tasks. These means can be hardware or software;
7. artificial intelligence software, the goal of which is to introduce changes to previously generated hardware configurations based on data from the knowledge information base, the information base of technical characteristics, and data obtained from the means of measurement of the technical characteristics, in order to improve the technical characteristics of the CS.

These means must operate in a CS in parallel with the execution of user's tasks.

High-level structure of self-improvable CS, which shows the principles of interaction of its basic components and the functions they perform, is shown in Figure 2.



User
Task specification
(high-level language)
Self-improvable computer system
base of technical characteristics
- statistics
base of knowledge
- methods to improve technical characteristics
base of specifications
- task specifications / hardware configurations
Software tools for CS self-configuring
Means of creating effective hardware configurations
Hardware configuring tools
Artificial intelligence software
Means of choosing a characteristic and a method to improve it
Means of estimation of technical characteristics
Means of specification / performance improvement
Processing core
Reconfigurable environment
(FPGA)
Fixed part
(universal programmable processor(s))
Means of measuring technical characteristics

**Figure 2**: High-level structure of self-improvable computer system

Self-Improvable computer system operation model consists in following. The user submits a task specification in the form of computer program to the SICS. This program, using software tools for CS self-configuration, is automatically compiled first into subprograms for UPP and RCE, then the second one into a software model of a specialized processor to execute RCE subprogram, and then both into executable file. This file consists of object codes for execution by the universal processor(s) and RCE configuration codes, and specifies the dedicated hardware configuration corresponding to submitted input program. The created executable file together with the corresponding input program, subprograms for UPP and RCE and the software model of the specialized processor to execute RCE subprogram, are stored in the information base of specifications.

After program initialization, from the information base of specifications the corresponding hardware configuration (executable file) is being loaded. Then, during the program execution, the technical characteristics of the CS are measured by appropriate means and submitted to the artificial intelligence software. After that, the process of self-improvement of the CS begins, which is iterative. In the first iteration, the obtained values of the characteristics are stored in the appropriate information base and simultaneously submitted for processing to the means of choosing a characteristic and a method to improve it. These means select some characteristics (according to predefined criteria) and a method to improve it. The corresponding specification is being read from the information base of specifications and a trial of its improvement is performed. The improvement can be performed at various levels: at the level of the input program, at the level of obtained subprograms of UPP and RCE, at the level of software model of specialized processor, and at the level of FPGA configuration. The obtained improved specification is submitted to software tools for CS self-configuration, which automatically create a new configuration and store it to the appropriate database. This configuration can be loaded either with the next initialization of this program, or during its current execution (which is more difficult to implement).

In each subsequent iteration, the obtained values of the technical characteristics are compared with their previous ones, read from the information base, and depending on the results, the means of artificial intelligence decide to further improve this characteristic, choose another characteristic, or return back to the previous configuration. As a result, the CS improves itself.

Next, we consider the method of self-configuration of CS with reconfigurable logic, which is realized by software tools for CS self-configuration. An overview of these software tools will be given in the next section.

## 6.2. Method of Self-Configuration of Computer System with Reconfigurable Logic

The principle of self-configuration, which is a basis of self-configurable computer systems design concept, provides operation of a computer system with reconfigurable logic by independent and automatic determining and creating of hardware configuration effective for the submitted task – its architecture and behavior of its hardware (in the form of a data structure - a file specifying the configuration) and automatic and immediate acquisition of this configuration when the task is executed.

To implement the principle of self-configuration in a computer system with reconfigurable logic, a method of self-configuring was developed, as well as a method of information processing based on it, which are described in detail in [27-29].

The method of self-configuring of a computer system with reconfigurable logic, which contains general-purpose processors and reconfigurable environment built with FPGA or other types of reconfigurable integrated circuits, is that a software is introduced into computer system, that during compilation of user programs solely and independently extract from it such fragments, the execution of which in the reconfigurable environment accelerates the computer system, and distribute this program to the subprograms for GPP and for RCE formed from the selected fragments. Next, the GPP subprogram is compiled into object code, and the RCE subprogram into its configuration file so that first from this subprogram a software model of a specialized processor for its execution is generated, then its logic synthesis is performed and thus an executable file from these two components is created.

At the stage of program loading after its initialization the executable code of the GPP subprogram is being loaded into a main memory and, at the same time, the configuration code is being loaded into a RCE and thus creates an ASP in there. Then program execution is being performed.

This ensures automatic and transparent to the user and efficient (in terms of performance and hardware use) use of reconfigurable logic in the computer system. The method of self-configuring given above is one of the basic mechanisms for the CS self-improvement.

## 6.3. Artificial Intelligence Technologies

CS self-improvement tools will also be based on the use of artificial intelligence technologies. The first in this context is the use of self-learning technologies, which allow to significantly improve the services of CS through the accumulation and application of historically acquired knowledge. Secondly, it is the use of mechanisms of expert systems that allow for in-depth analysis of the possibility of improving the service programs of CS. Third, it is the use of computational intelligence technologies: neural networks, genetic algorithms, fuzzy logic, etc., for the fast execution of optimization procedures [31]. And fourth, it is the use of technologies of collective intelligence: multi-agent systems, ant intelligence, swarm intelligence, etc., to quickly find the best solutions, in particular, described in [36].

The integration of the above technologies in order to obtain a synergistic effect from their joint use is a factor that will ensure the development of the architecture of self-improvable computer systems.

## 7. Software Basis of Self-Improvable Computer Systems

Below we will consider the basic software tools of self-improvable computer systems. First of all, we consider the tools for generating software models of specialized processors from description of their operation in a high-level programming language. Next, we provide a brief overview of other software tools used to implement self-configuration of the CS.

### 7.1. Specialized Processors High-Level Design Tools as a Basis of the Mechanism of Computer Systems Self-Improvement

The basis of the mechanism of computer systems self-improvement, which provides the ability to solely and independently improve the efficiency of hardware use, are specialized processors high-level design tools. These tools, in contrast to the conventional design on the register transfer level, suppose a description of the algorithm executed by the specialized processor in high-level language, a separate description of the specialized processor interface and its technical characteristics, and, based on this information, further automatic generation of the spectrum of possible VHDL-models of specialized processor, their synthesis, estimation of characteristics and selection of the most effective one according to the set criteria.

That is, the use of high-level design tools allows one to design specialized processors and choose among the designed the best in terms of technical characteristics. As an example, Figure 3 depicts the dependency of the operation frequency to the ALU number with the preserved level of performance of FFT processor, VHDL models of which have been generated automatically with use of the Chameleon C2HDL design tool developed by Intron Ltd [26], [37], [38], [39], to the development and implementation of which the authors have made significant efforts.

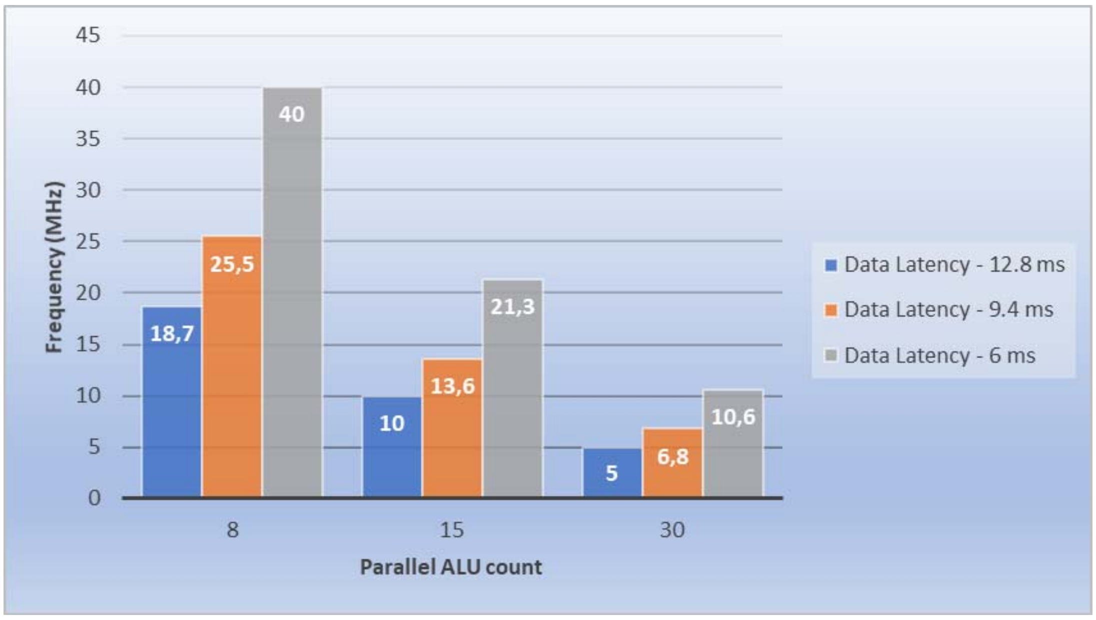


**Figure 3**: Dependency of the FFT processor operation frequency to the ALU number with the preserved level of performance

The Chameleon C2HDL automatic design tool is also the basis of self-configurable computer systems and the method of self-configuration. Note that besides to the Chameleon, there are a number of other high-level design software tools that are designed to generate HDL code of computer devices from the description of the algorithm of their operation in high-level programming languages. These, for example, are: C-to-Verilog [40], System-C [41] and Catapult-C [42] from Calypto Design Systems, SPARK [43] from University of California. These tools can be used with various efficiency for the implementation of self-improvable CS.

VHDL-model of specialized processor can be further implemented in FPGA using CAD tools provided by their manufacturers (Intel Quartus, Xilinx ISE, etc.).

### 7.2. Other Software Tools of SICS, with which Self-Configuration is Realized

To the other software tools of SICS, with the use of which self-configuration is implemented, belong computational load balancing system, compiler for GPP subprograms compilation, and specialized processor logic synthesis tools.

The software means are used for computational load balancing between GPP and RCE. This software has to automatically extract fragments from the input (user) program, execution of which in RCE reduces this program execution time, and divide this program into the GPP subprogram, and the RCE subprogram, formed with the extracted fragments. An example of such system implementation is shown in [44]. The RCE subprogram here is created in the x86 assembly language, and therefore it must be supplemented by some software for the assembly language code translation into a high-level programming language to be used in SICS. The mentioned software is also available on the market, e.g., Relogix Assembler-to-C Translator [45] from MicroAPL.

A compiler is used for compilation of the GPP subprogram from the language that it is represented in into the object codes that can be directly executed by the general-purpose processors.

Logic synthesis tools and FPGA configuring tools are used for the specialized processor HDL-models logic synthesis during the program compilation stage and FPGA configuring during the program loading stage. These tools are available from the FPGA vendors, for example, Vivado Design Suite and ISE from Xilinx; Quartus II from Intel.

## 8. Expected benefits brought by creation of SICS

Computers have become a mandatory component of our life. They are used for solving of science and engineering problems including the design of automobiles, buildings, electronic devices, aircrafts, medications, and robots. Their use makes automobiles safer, more aerodynamic, more comfortable, and more energy-efficient. The more powerful computer is used the more considerable effect of it is gained. For example, exascale computing is used to understand economics, national security, and setting public policy. Billions of processor hours are devoted to understanding and predicting climate change. With faster computer systems, scientists and engineers could simulate critical details - such as clouds in a climate model or mechanics, chemistry, and fluid dynamics in the human body.

Development of the new computer systems architecture has gained significant attention mainly because of its core influence on the computer system characteristics improvement, especially during the past decade when using microelectronic technology improvements for it became very low.

Further practical implementation of self-improvable computer systems based on new computing models will be a fundamental innovation which has no analogues in the field of computer technology. According to preliminary estimates, self-improvable computer systems will be able to increase their technical characteristics by 2-3 orders of magnitude. This assessment is based on the fact that self-improvable CS will be able to adapt their architecture to the executed tasks by its specialization, and according to many years of experience in the production and operation of computer systems, confirmed by data thoroughly covered in [9], the value of improving computer systems performance due to their specialization reaches 2-3 orders of magnitude. A similar indicator is observed in the reduction of power consumption of computer systems. This will dramatically increase the efficiency of their operation.

The following example shows what benefits can be obtained as a result. In 2018, the global electricity consumption was 23 809 TWh out of which 2 476 TWh or 10% were consumed by information and communication technology. The trend suggests that in 2030, the global consumption will be around 40 000 TWh out of which around 8 000 TWh or 20% will be information and communication technology [46], [47]. It is obvious that the reduction of energy consumption by at least 1 order, which is quite possible to achieve by self-improvement of computers, in 2030 will bring, based on this estimate, savings of 7200 TWh! This figure is impressive!

Thus, the creation of such systems will have a positive impact on the further development of society and civilization in general, taking into account the use of computer systems in all spheres of today's human activity.

## 9. Conclusions

1. SICS is a new type of computer systems, and today there is already available all methodological and technological base necessary for their practical implementation, namely - high-performance hardware of universal processors and reconfigurable logic devices, technologies and software of self-configuration of computer systems with reconfigurable logic and artificial intelligence technologies. Therefore, the creation of SICS is likely to be one of the significant stages of further development of the computer technology in the near future.
2. Self-improvement can be carried out at different levels of the CS specification. At the highest - the level of the input program that specify the algorithm of the CS operation, and at the lower - at the level of UPP and RCE subprograms, at the level of software model of a specialized processor, and at the level of FPGA configuration.
3. An important software basis for the mechanism of self-improvement of computer systems are specialized processors high-level automatic design tools, which allow one to automatically obtain a number of software models of a specialized processor from description of the algorithm of its operation in high-level language, and then choose the model with best characteristics.
4. Supplementing the architecture of computer systems with the self-improvement mechanisms should provide an opportunity at the stage of their operation to solely and independently increase the efficiency of hardware use by increasing performance in the classes of algorithms up to 2-3 orders of magnitude, or decreasing power consumption by the same level. Based on the fact that the share of information and communication technology in total global energy consumption today is more than 10%, in the global dimension, as the result of such innovations the energy savings can have a huge impact!
5. SICS are multifaceted, which requires a wide range of research.

## 10.References

[1] Yao X., Higuchi T., Promises and challenges of Evolvable hardware. In: Higuchi T., Iwata M., Liu W. (eds) Evolvable Systems: From Biology to Hardware, ICES 1996, volume 1259 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 1997. doi: 10.1007/3-540-63173-9_38.

[2] E. Cantu-Paz, A survey of parallel genetic algorithms, Calculateurs Paralleles, Reseaux et Systems Repartis, 10(2), 141–171, 1998.

[3] D. Goldberg, Genetic Algorithms in search, optimization, and machine learning, Addison– Wesley, 1989. ISBN:978-0-201-15767-3.

[4] S. Trimberger, Field-programmable gate array technology, Kluwer Academic Publishers, Boston, 1994. doi: 10.1007/978-1-4615-2742-8.

[5] S. Hauck, A. DeHon, Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation, Morgan Kaufmann, 2007. ISBN: 9780123705228.

[6] C. Bobda, Introduction to Reconfigurable Computing: Architectures, Algorithms, and Applications, Springer, 2007. doi: 10.1007/978-1-4020-6100-4.

[7] T. Todman, G. Constantinides, S. Wilton, O. Mencer, W. Luk and P. Cheung, Reconfigurable Computing: Architectures, Design Methods, and Applications, in: IEE Proceedings on Computers and Digital Techniques, 152 (2), 2005, pp.193-207. doi: 10.1049/ip-cdt:20045086.

[8] V. Melnyk. “Principles of Design and Operation of Reconfigurable Computer Systems.” Scientific journal "Bulletin of Khmelnytskyi National University", series "Technical sciences", №6, (2012): 212 - 217.

[9] Shao, Y.S., Reagen, B., Wei, G.Y., and Brooks, D. Aladdin, A pre-RTL, power-performance accelerator simulator enabling large design space exploration of customized architectures, in: Proceedings of the Int. Symp. Comput. Archit., 2014, pp. 97–108. doi:10.1109/ISCA.2014.6853196.

[10] Upegui A., Sanchez E., Evolving Hardware by Dynamically Reconfiguring Xilinx FPGAs, in: Moreno J.M., Madrenas J., Cosp J. (eds), Evolvable Systems: From Biology to Hardware, ICES 2005, volume 3637 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2005. doi:10.1007/11549703_6.

[11] A. Thompson, An evolved circuit, intrinsic in silicon, entwined with physics, in: First International Conference on Evolvable Systems (ICES-96), Lecture Notes in Computer Science, Springer-Verlag, 1996, pp. 390- 405. doi: 10.1007/3-540-63173-9_61.

[12] T. Higuchi et al., Evolvable hardware: A first step towards building a Darwin machine, in Proceedings of the 2nd International Conference on Simulated Behaviour, MIT Press, 1993, pp. 417–424.

[13] J.R. Koza et al., The importance of reuse and development in evolvable hardware, in: J. Lohn et al. (Eds.), Proceedings of the 2003 NASA/DoD Conference on Evolvable Hardware, Chicago, IL, USA, 2003, pp. 33–42. doi: 10.1109/EH.2003.1217640.

[14] M. Sipper, E. Sanchez, D. Mange, M. Tomassini, A. Perez-Uribe, and A. Stauffer. "A Phylogenetic, Ontogenetic, and Epigenetic View of Bio-Inspired Hardware Systems." IEEE Transactions on Evolutionary Computation, vol. 1, 1997: 83-97.

[15] J. F. Miller, Digital filter design at gate-level using evolutionary algorithms, in: W. Banzhaf et al. (Eds.), Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99), Morgan Kaufmann, 1999, pp. 1127–1134.

[16] M. Murakawa et al. "The grd chip: Genetic reconfiguration of dsps for neural network processing." IEEE Transactions on Computers, 48(6), June 1999: 628–638.

[17] X. Yao, T. Higuchi. "Promises and challenges of evolvable hardware." IEEE Transactions on Systems, Man, and Cybernetics, Part C, vol. 29, February 1999: 87 - 97.

[18] M. Murakawa, S. Yoshizawa, I. Kajitani, T. Furuya, M. Iwata, and T. Higuchi, Hardware evolution at function level, in: Proceedings of Parallel Problem Solving from Nature IV (PPSN IV), volume 1141 of Lecture Notes in Computer Science, Springer-Verlag, 1996, pp. 62–71.

[19] Sakanashi et al., Evolvable hardware chip for high precision printer image compression, in Proceedings of 15th National Conference on Artificial Intelligence (AAAI-98), 1998.

[20] R. Porter, K. McCabe and N. Bergmann, An applications approach to evolvable hardware, in: Proceedings of the First NASA/DoD Workshop on Evolvable Hardware, Pasadena, CA, USA, 1999, pp. 170-174. doi: 10.1109/EH.1999.785449.

[21] L. Sekanina, Evolvable Components: From theory to Hardware Implementations, SpringerVerlag, 2004. doi: 10.1007/978-3-642-18609-7.

[22] C. Hsu, W. Feng and J. S. Archuleta, Towards efficient supercomputing: a quest for the right metric, in: 19th IEEE International Parallel and Distributed Processing Symposium, Denver, CO, USA, 2005, p. 8. doi: 10.1109/IPDPS.2005.440.

[23] Małgorzata Płaza, Stanisław Deniziak, Mirosław Płaza, Radosław Belka, Paweł Pięta, Analysis of parallel computational models for clustering, in: Proceedings SPIE 10808, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2018, 108081O (1 October 2018). doi: 10.1117/12.2500795.

[24] Leiserson CE, Thompson NC, Emer JS, et al., There's plenty of room at the Top: What will drive computer performance after Moore's law?, Science, New York, N.Y., 2020 Jun, 368(6495). doi: 10.1126/science.aam9744.

[25] Neil C. Thompson, Svenja Spanuth, The Decline of Computers as a General Purpose Technology, Communications of the ACM, Vol. 64 No. 3, (2021) 64-72. doi: 10.1145/3430936.

[26] Melnyk, A., Salo, A., Automatic generation of ASICs, in: Proceedings of 2007 NASA/ESA Conference on Adaptive Hardware and Systems, AHS-2007, 2007, pp. 311–317.

[27] Melnyk, A., Melnyk, V., Self-Configurable FPGA-Based Computer Systems, Advances in Electrical and Computer Engineering, vol. 13, no. 2 (2013) 33-38. doi: 10.4316/AECE.2013.02005.

[28] V. Melnyk. "Conceptual bases of the self-configurable computer system components design." Lviv Polytechnic National University Journal "Computer Systems and Networks" №773, 2013: 73-81.

[29] Anatoliy Melnyk, Viktor Melnyk, Heterogeneous computing: from the ASIC-based hardware accelerators to the reconfigurable and self-configurable computer systems, in: Proceedings of the 6-th International Conference on Advanced Computer Systems and Networks, ACSN-2013, Lviv, 2013, pp. 9-12.

[30] R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, M. Steinbrecher, P. Held, Computational Intelligence: A Methodological Introduction, Springer-Verlag London, 2013. doi: 10.1007/978-1-4471-5013-8.

[31] L. Rutkowski, Computational Intelligence: Methods and Techniques, Springer, 2008. doi: 10.1007/978-3-540-76288-1.

[32] J. Leijten, G. Burns, J. Huisken, E. Waterlander, A. Van Wel, AVISPA: a massively parallel reconfigurable accelerator, in: Proceedings of International Symposium on System-on-Chip (IEEE Cat. No.03EX748), Tampere, Finland, 2003, pp. 165-168. doi: 10.1109/ISSOC.2003.1267747.

[33] T. El-Ghazawi, Reconfigurable supercomputing, in: Proceedings of the IEEE/ACS International Conference on Pervasive Services, ICPS-2004, Beirut, Lebanon, 2004, p. 163. doi: 10.1109/PERSER.2004.1356790.

[34] Sandeep Kumar, Christof Paar, Jan Pelzl, Gerd Pfeiffer, Manfred Schimmler, COPACOBANA: A Cost-Optimized Special-Purpose Hardware for Code-Breaking, in: Proceedings of the 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, FCCM'06, 2006, pp. 311-312. https://doi.org/10.1109/FCCM.2006.34.

[35] O. Drozd, K. Zashcholkin, O. Martynyuk, O. Ivanova, J. Drozd, Development of Checkability in FPGA Components of Safety-Related Systems, in: CEUR Workshop Proceedings, vol. 2762, pp. 30-42 (2020). URL: http://ceur-ws.org/Vol-2762/paper1.pdf.

[36] Melnyk A., Golembo V., Bochkaryov A., Multiagent approach to the distributed autonomous explorations, in: Proceedings of NASA/ESA Conference on Adaptive Hardware and Systems, AHS-2007, Edinburgh, UK, 2007, pp. 606-610.

[37] A. Melnyk, V. Melnyk., Personal Supercomputers: Architecture, Design, Application, Lviv Politechnic National University Publishing, 2013.

[38] Chameleon – the System-Level Design Solution, 2008. URL: http://intron-innovations.com/?p=sld_chame.

[39] Melnyk, A., Salo, A., Klymenko, V., Tsyhylyk, L. "Chameleon – system for specialized processors high-level synthesis." Scientific-technical magazine of National Aerospace University "KhAI", Kharkiv, N5 2009: 189-195.

[40] Giang Nguyen Thi Huong, Seon Wook Kim, GCC2Verilog Compiler Toolset for Complete Translation of C Programming Language into Verilog HDL, ETRI Journal, 2011, 33(5), pp. 731-740. https://doi.org/10.4218/etrij.11.0110.0654.

[41] IEEE Standard for Standard SystemC Language Reference Manual, IEEE Std 1666-201, 9 January 2012, 638 p.

[42] Catapult High-Level Synthesis and Verification, Siemens EDA Software, 2021. URL: https://eda.sw.siemens.com/en-US/ic/catapult-high-level-synthesis/.

[43] S. Gupta, N.D. Dutt, R.K. Gupta and A. Nicolau, SPARK: a high-level synthesis framework for applying parallelizing compiler transformations, in: Proceedings of the 16th International Conference on VLSI Design, New Delhi, India, 2003, pp. 461-466, doi: 10.1109/ICVD.2003.1183177.

[44] V. Melnyk, V. Stepanov, Z. Saraireh. "Computational Load Balancing System Between the Host-Computer and Self-Configurable Accelerator." Scientific Bulletin of Chernivtsi National University "Computer systems and components", Chernivtsi: Yuriy Fedkovych Chernivtsi National University, Vol.3, Issue 1, 2012: 6-16.

[45] Relogix Assembler-to-C translator, 2017. URL: http://www.microapl.co.uk/asm2c/.

[46] Anders S. G. Andrae, Tomas Edler, On Global Electricity Usage of Communication Technology: Trends to 2030, Challenges 2015, 6(1), 117-157. doi:10.3390/challe6010117.

[47] Between 10 and 20% of electricity consumption from the ICT sector in 2030? 9 Aug 2018. URL: https://www.enerdata.net/publications/executive-briefing/between-10-and-20-electricity-consumption-ict-sector-2030.html.