UDC 621.121
DOI https://doi.org/10.32782/2663-5941/2023.5/28

Maidan M.V.
Lviv Polytechnic National University

IOT DEVICES PROCESSORS SYNTHESIS IN EDGE COMPUTING: APPROACH ANALISES

Modern approaches to organizing the infrastructure of the Internet of Things (IoT) network have been studied. IoT is a global network of physical devices that can interact with each other and the cloud, exchange data, and perform specific functions. To ensure efficient functioning of such a network, proper infrastructure organization is required. The effectiveness of applying the Edge Computing concept within the Internet of Things (IoT) has been demonstrated. The Edge Computing concept involves processing data on devices that are closer to the data source, rather than on remote cloud servers. This reduces the load on the cloud and reduces data transmission delays. In the context of the Internet of Things, the Edge Computing concept can be used to process data obtained from IoT devices on the devices themselves or on nearby gateways, which can significantly improve the speed and efficiency of data processing. To provide more efficient computing performance and energy consuming become popular to using reconfigurable hardware together with edge computing. Reconfigurable computing involves using devices with flexible configuration to process data. In the context of IoT and Edge Computing, devices with flexible configuration can be used to process data collected from IoT devices, which allows adapting computing resources to a specific task and reduces data processing time. The application of Reconfigurable computing concept in IoT and Edge Computing can be particularly useful in cases where it is necessary to process large volumes of data with high accuracy and speed. As a result of the research, a basic scheme for IoT architecture and Edge computing system architecture was proposed, based on which the architecture of the FGPA system was formed. The architecture provides the flexibility and scalability necessary for the efficient processing of large volumes of data in real-time, while also minimizing the workload on the central processing unit.

Key words: *Internet of Things, edge computing, reconfigurable computing, field-programmable gate arrays, high-level synthesis, register-transfer level.*

Introduction. Over the last decade, the Internet of Things (IoT) has rapidly gained popularity and is now considered one of the most important technologies. It is used across a wide range of industries, from big players such as automotive and medicine, to house building and media industries [1-3]. It brings numerous benefits, including increased efficiency, automation, optimizing supply chains, reducing downtime, improve quality control and decision-making. Despite the security risks and privacy issues, IoT is driving digital transformation across industries and enabling the creation of new business models and revenue streams. Thereby, this technology is considered as a transformative technology with far-reaching implications. As such, it is crucial to understand both the potential benefits and the challenges associated with IoT and to work towards developing effective solutions.

IoT basically can be represented as a network of interconnected devices that can exchange data among themselves and with other networks without human intervention. It can include devices such as smartphones, home appliances, cars, medical equipment, and many others. IoT network can be split into three different layers [3, 4]. The first one is IoT device itself, and this layer include device itself with different sensor, usually these are small devices with limited power consumption and performance. Second layer is Fog/Edge gateway — that is type of devices in IoT network which collect and process data from group of IoT devices. Third layer is Cloud node or server is main brain of the network and can process big amount of data. Usually all computing happens on cloud side and it create problem which is hard to resolve. One of the biggest problems is latency, IoT devices used in domain where such things as latency is very important and even can be dangers if it is big. Since device need to make decision in real time it is really important to reduce latency as much as it possible. With architecture where main decision maker is cloud, it is not possible since internet connection can be lost or server can be unavailable for different reason. Also, there is a bunch of another problem in Cloud computing conception such as: security, privacy, availability etc.

IoT devices collect and transmit massive amounts of data, and transferring this data to a centralized cloud

data processing center can be slow and resource-intensive in terms of network usage. Edge computing provides the ability to process data directly on IoT devices, which reduces delays in data transmission, reduces network congestion, and allows for faster responsiveness to changes in data. Conception of edge computing helps to resolve some issues created by Cloud computing. Instead of send data from devices to the server, Edge computing make data processing on edge gateway which is near to the device, or make this processing on the same devices. By using this manner of process data this conception brings a lot of advantages for IoT technologies and allow make decision about some action in place. Edge computing technology has proved to be a successful solution for addressing technical challenges in various practical applications. It has been particularly effective in dealing with the vast amounts of data generated by the geographically distributed network of IoT devices, which can cause significant data congestion issues [5, 6].

In the field of IoT and, in particular, Edge computing, Reconfigurable computing can be effectively utilized to enhance the performance and energy efficiency of edge devices. Reconfigurable computing enables the customization of hardware and software for specific applications, thereby optimizing the use of available resources and minimizing energy consumption. By adapting the hardware to the specific needs of the application, edge devices can achieve higher performance and lower latency, which is crucial for real-time processing and decision-making in IoT applications. Reconfigurable computing is a discipline which is includes to itself hardware and software combination for creating a new conception for high-performance computing. Reconfigurable computing is also known as configurable computing or custom computing, since many of the design techniques can be seen as customizing a computational fabric for specific applications [7]. This type of computing allows to adapt hardware for specific algorithm which can improve algorithm performance in many times since algorithm running not on CPU which is universal and might not be the best fit for it but running on specific hardware architecture which is best for that specific algorithm. For example, GPU is the one of the best hardware architectures to make image processing or making parallel computing. This type of hardware architecture proves to be much better for that type of algorithm than CPU [8]. Except speed reconfigurable hardware has many other advantages compare to microprocessor. Reconfigurable hardware help to reduce energy and power consumption. In a reconfigurable system, the circuitry is optimized for the application, such that the power consumption will tend to be much lower than that for a general-purpose processor. A recent study reports that moving critical software loops to reconfigurable hardware results in average energy savings of 35% to 70% with an average speed up of 3 to 7 times, depending on the particular device used [9].

It should be also noticed that since Edge computing algorithms require data processing near to the source of data, IoT devices itself require to be more powerful and more flexible in changing and adapting algorithm for data processing by using hardware organizing based on field-programmable gate arrays (FPGAs). Main advantages of FPGA are that it can be reprogrammable many times and it allows adapting hardware architecture for specific algorithm [10, 12]. Architecture based on FPGAs can be like a coprocessor and help main process make high performance computing on data since FPGA architecture can be adapted for this. Thus, within the scope of this study on adapting Edge computing in the organization of IoT infrastructure, methods such as Reconfigurable computing and hardware solutions based on FPGAs are proposed to be considered

Goal. The main objective of this article is to conduct a comprehensive analysis of four key technological concepts: IoT, edge computing, reconfigurable computing, and high-level synthesis. the article aims to provide a detailed description of each concept and illustrate their respective architectures to aid the reader's understandingadditionally, the article intends to suggest potential avenues for future research by exploring the potential benefits of integrating reconfigurable hardware within edge computing architectures.

Architecture of internet of things. The architecture of IoT comprises five distinct layers that collectively define the entire spectrum of functionality within an IoT system. The fundamental depiction of the corresponding structure is illustrated in Fig. 1.

Let's consider the components of the IoT architecture according to the basic architecture organizing scheme presented above:

1) *Application layer.* This layer is responsible for conducting analysis and providing data to end user. Typically, this layer is comprised of a sophisticated software solution, employing a diverse range of technology stacks. For example the following classification of components of this layer can be presented:

− Mobile and desktop applications classified into types of native applications, hybrid applications and web applications.

− Intelligent services which provide personalized and efficient solutions to users automating complex tasks, analyzing Big Data, and making predictions based on trends.

− Device monitoring software which analyses activity of various electronic devices as a IoT nodes (parental control, employee monitoring, security monitoring).

− Device control software configuring settings such as device status, permissions, and access to sensitive data.

− Cloud services accessed over the internet and hosted on remote servers maintained by third-party providers basically classified into infrastructure services, platform services, and software services.

− Machine learning solution based on artificial neural network (ANN) algorithms and statistical functions.

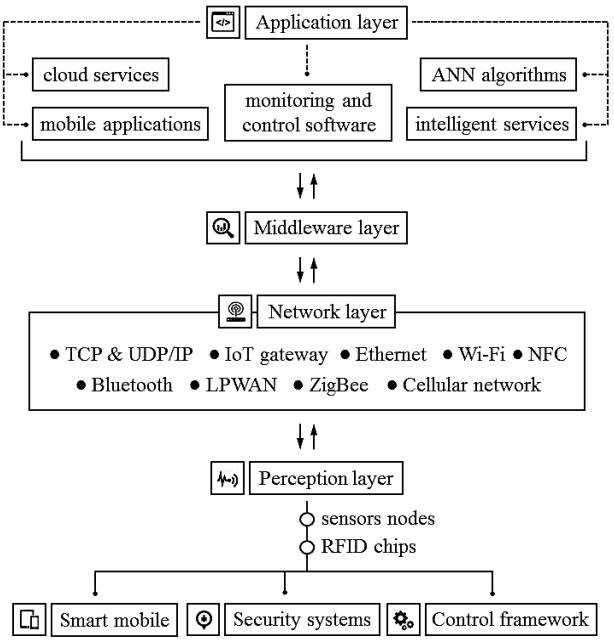

Fig. 1. Layers of IoT architecture

2) *Middleware layer.* The primary function of the layer is to receive and process data from the network layer, and based on the outcomes derived from ubiquitous computing, make informed decisions. This layer is also responsible for providing storage space for data and utilizing various algorithms to analyze it.

3) *Network layer.* The network layer, also known as the connectivity or transport layer, facilitates the transfer of data between the physical device and the middleware layer, as well as vice versa from the middleware layer to the device. The connectivity between the physical layer and the cloud is achieved in two ways:

− Connection-oriented protocol TCP/IP with high levels of reliability and basic connectionless protocol UDP/IP.

− Gateways as hardware or software modules of the IoT network performing translation between different protocols as well as encryption and decryption of IoT data.

For physical connections, there are also extensive range of technologies commonly used in IoT, including further:

− Ethernet as most basic wired computer networking technology.

− Wireless networking technology Wi-Fi which uses radio protocol for data transmitting between devices in a local network.

− NFC as wireless data transmission technology at close range that uses radio frequency identification (RFID) to secure information exchange between devices.

− Bluetooth as a wireless data transmission technology at short range that uses radio frequency for communication between mobile devices.

− LPWAN (Low-power Wide-area Network) as a wireless network technology designed for long-range communication with low power consumption.

− ZigBee which operates on low power and uses routing protocols to increase network coverage providing low power consumption and high reliability for data transmission between devices.

− Cellular networks widely used in IoT applications due to their wide coverage area, high reliability, and secure communication.

Furthermore it should be mentioned that to enable rapid and secure data exchange in IoT, the following message protocols are employed:

− MQTT (Message Queue Telemetry Transport) as a lightweight publish-subscribe messaging protocol that enables efficient communication between devices in IoT networks.

− LWM2M (Light way machine two machine) as a a device management protocol for IoT that defines a set of standards and interfaces for managing and monitoring devices in a secure and efficient manner.

− CoAP (the Constrained Application Protocol) as a lightweight application layer protocol for to enable communication between devices with limited processing power and memory using uses UDP as the underlying transport protocol and supports RESTful interfaces for resource discovery and manipulation.

– AMQP (the Advanced Message Queuing Protocol) as an open standard messaging protocol enabling communication between applications and services in a distributed environment providing

a reliable and secure messages exchanging, with message queuing, routing, and flow control.

4) *Perception layer.* The perception layer is comprised of IoT network nodes based on sensors, RFID chips, etc. It includes mobile devices (smartphones, tablets, notebooks), security systems and control framework components (traffic network, industrial objects, healthcare). IoT nodes of these devices collect information in order to deliver it to the network layer.

In addition to the essential layers outlined above, the basic architecture of an IoT system can be augmented with several additional layers at the discretion of the system designer. These supplementary layers, which have gained popularity in recent years, include:

− The layer of edge or fog computing is responsible for processing data in close proximity to its source (IoT node). As part of the corresponding architecture, it could either be an edge gateway or the IoT device itself.

− The security layer is responsible for providing encryption and decryption of data, as well as managing the authentication and authorization flow.

− The business layer is where decisions are made based on data, enabling informed business choices. This layer facilitates the integration of the IoT system with the organizations existing business processes, allowing the system to provide insights that can inform business strategies and improve operational efficiency.

Edge computing architecture. Edge computing refers to a form of computing that occurs in close proximity to the device or even on the device itself. This approach is not limited to certain types of devices, but rather encompasses the use of any device capable of performing data processing. In essence, edge computing is a broad concept that extends beyond computing on specific devices.

Edge computing facilitates the integration of networking and computing, resulting in high-performance data computing and exchange architecture. The processing of data takes place in close proximity to the device, thereby minimizing the amount of data that needs to be transmitted to the central server. Moreover, most operations are carried out in real-time at the location where the data is generated, leading to the following benefits:

− improved response times;
− better bandwidth availability;
− more reliable system insights;
− comprehensive and expeditious analysis of data;
− high availability.

Edge computing is well-suited for scenarios that require the processing of time-sensitive data to facilitate prompt decision-making. Additionally, it outperforms cloud solutions for operations carried out in remote regions with limited or no internet connectivity. However, it is important to note that edge computing should not be viewed as a substitute for cloud computing. These technologies serve distinct purposes, and cannot be used interchangeably. Instead, edge computing should be seen as a complement to cloud computing, as the two technologies working in tandem can offer superior performance for specific use cases.

The typical architectural framework for edge computing [5, 6] comprises three distinct layers, as illustrated at a basic level in Figure 2:

− Terminal layer. The terminal layer constitutes a crucial aspect of the edge network, comprising a diverse array of devices, including mobile terminals, as well as a multitude of (IoT) devices, such as sensors, smartphones, smart cars, cameras, and numerous others. It is worth noting that this layer holds immense potential, as the number of devices capable of capturing and relaying data in this layer can reach into the billions. As such, the terminal layer plays a pivotal role in enabling the seamless functioning of the edge network, facilitating the collection and processing of data at the edge.

− Edge layer. The edge layer represents a crucial element in the architecture of edge computing, serving as a pivotal foundation for the operation of the overall network. This layer is composed of a diverse range of nodes that are distributed among edge devices, operating in close proximity to these devices to enable high-performance computations. These nodes undertake a variety of critical tasks, including data processing, storage, and transmission, among others, thereby facilitating the seamless functioning of the edge network.

− Cloud layer. Federated cloud-edge computing services have emerged as a promising paradigm for enabling efficient and scalable data processing in a distributed environment. Among the various layers of this architecture, the cloud layer serves as a dominant data processing center, offering a wide range of capabilities to handle complex computations. Comprising an extensive collection of high-performance servers and storage devices, the cloud layer holds immense potential to cater to the ever-increasing demand for data analysis, particularly in domains such as routine maintenance and business decision support.

To evaluate the effectiveness of a network based on IoT network edge computing architecture, it is

necessary to formalize the process of its operation by defining key components:

− *Edge device.* An edge device refers to a specialized device with restricted computing capacity that functions as the terminal layer in a standard three-layer architecture. In an Edge Computing Architecture, edge devices play a crucial role in gathering and processing data at the network's edge, thereby reducing latency and bandwidth usage.

− *Edge Gateway.* The Edge Gateway is a server whose primary function is to perform network routing, tunneling, firewall management, and other related tasks. Additionally, this device can facilitate certain data processing as necessary. It is a fundamental component of the Edge layer in a typical three-layer architecture.

− *Edge server.* An Edge server is a type of server that executes data processing tasks originating from edge devices. Typically, such servers are situated in close proximity to the edge devices that generate the data. This constituent is an essential element of the Edge layer in a typical three-layer architecture.

− *Cloud server.* A Cloud server is the primary server that may be located remotely from the edge devices. This category of server can execute computational tasks that are not time-critical in nature. There may be multiple servers employed in this regard. This component is an integral aspect of the Cloud layer in a conventional three-layer architecture.

FPGA Design. Reconfigurable computing systems are a class of computing systems that utilize FPGAs to enhance the execution speed of computationally-intensive algorithms by mapping them onto the reconfigurable substrate. FPGAs are particularly well-suited for such tasks as they can be reprogrammed multiple times to suit the requirements of a particular computation and thereby offer significant flexibility. These reconfigurable hardware resources are often coupled with a general-purpose microprocessor that is responsible for controlling the reconfigurable logic and executing program code that cannot be efficiently accelerated. This combination of a reconfigurable hardware substrate and a general-purpose processor is commonly referred to as hybrid architecture. The microprocessor operates as a host controller that communicates with the reconfigurable logic and directs it to execute the accelerated computations. This type of system is particularly beneficial for IoT applications where high performance is crucial, but the computations required are too complex or too diverse to be efficiently executed on traditional general-purpose processors [13-16]. FPGAs are integrated circuits that comprise an array of programmable logic

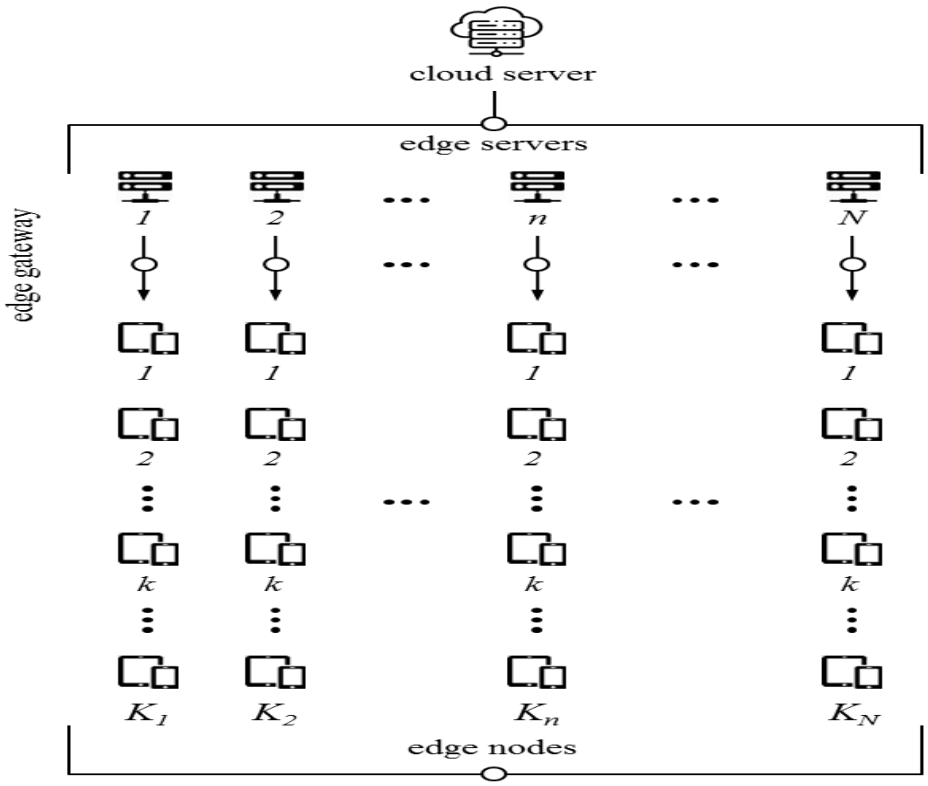

Fig. 2. Edge computing system architecture with key components

blocks, which are interconnected by a hierarchy of reconfigurable interconnects. These logic blocks can be configured to perform complex combinational functions or serve as simple logic gates, such as AND/XOR gates. Moreover, most FPGAs also incorporate memory elements, which can take the form of simple flip-flops or more advanced blocks of memory. These memory elements enhance the versatility of the FPGAs [15-17].

Figure 3 displays the internal architecture of a standard FPGA system, which is composed of three primary components:

− Configurable Logic Blocks (CLB) which implement logic functions.

− Programmable Interconnects (PI) which implement routing.

− Programmable I/O Blocks (PIOB) which connect with external components.

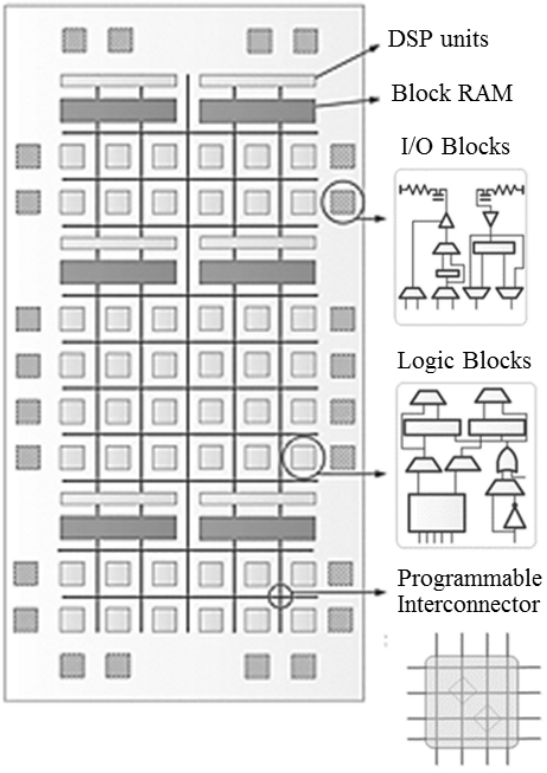

Fig. 3. Architecture of FGPA system

The logical functions are comprised of multiple components, and are essential for the design implemented by the logic block. To interface the configurable logic blocks and routing architecture with external components, input/output blocks are utilized.

High-level synthesis for FPGA. High-level synthesis (HLS) is a design process that transforms a high-level, functional description of a design into a Register Transfer Level (RTL) implementation, which satisfies user-specified design constraints. HLS is also known as behavioral synthesis and algorithmic synthesis. The HLS design description is considered "high-level" in two ways: design abstraction and specification language [18]. Design abstraction refers to the degree of detail at which the design is described. In HLS, the design is described at a higher level of abstraction than RTL. At the high-level of abstraction, the design is described in terms of its functionality and behavior, whereas, at the RTL level, it is described in terms of low-level hardware components such as registers, flip-flops, and gates. Specification language refers to the language used to describe the design. In HLS, the design is described using high-level languages which prove to be are closer to natural language and are easier to understand for designers who do not have a background in digital design.

Compared to RTL, the HLS design description exhibits two aspects of being “high level”: design abstraction and specification language.

− High level of abstraction. The high level of abstraction in HLS can be attributed to the fact that its input consists of an untimed, or partially timed, dataflow or computation specification of the design. This level of abstraction surpasses that of RTL, as it does not delineate a definitive cycle-by-cycle behavior, thereby providing HLS tools with the autonomy to make decisions regarding each clock cycle.

− The specification language employed in HLS is deemed high level, as its input is articulated using languages such as C, C++, System C, or even Matlab. This feature enables the utilization of advanced language attributes such as loops, arrays, classes, pointers, inheritance, overloading, templates, polymorphism, and other similar features. This level of specification surpasses that of the synthesizable subset of RTL description languages, as it permits design descriptions that are concise, reusable, and readable.

Quartus Prime software. The Quartus Prime software from Intel Corporation is a powerful tool for designing digital devices based on programmable logic devices (PLDs) and systems-on-chip (SoCs). Its versatile applications include distributed computing performed on edge devices that are in close proximity to the data source. With Quartus Prime, it is possible to create high-performance and energy-efficient PLDs and SoCs that can process data on-site with minimal latency and high speed. This capability empowers developers to design devices that not only perform data processing, but also manage it. Consequently, Quartus Prime facilitates the creation of edge devices that can not only process data, but

also make decisions on-site, without the need to send data to a central server. The software's ability to create software for SoCs opens up new possibilities for developing smarter devices that can perform advanced data processing tasks [19].

This subsection delineates the implementation of Quartus Prime software using the suggested approach on the FPGA. It is supposed that Verilog HDL as high-level hardware description language could be used to design and simulate digital circuits and systems to script the modules, while Quartus Prime software was used to perform synthesis, converting the Verilog register-transfer level design to bit streams, which could then be programmed onto the FPGA. FPGA devices are equipped with specialized on-chip memory blocks known as block random access memories (BRAMs) which come in different sizes depending on the particular FPGA in use. BRAMs could be leveraged as the internal memory resource for storing the external memory addresses.

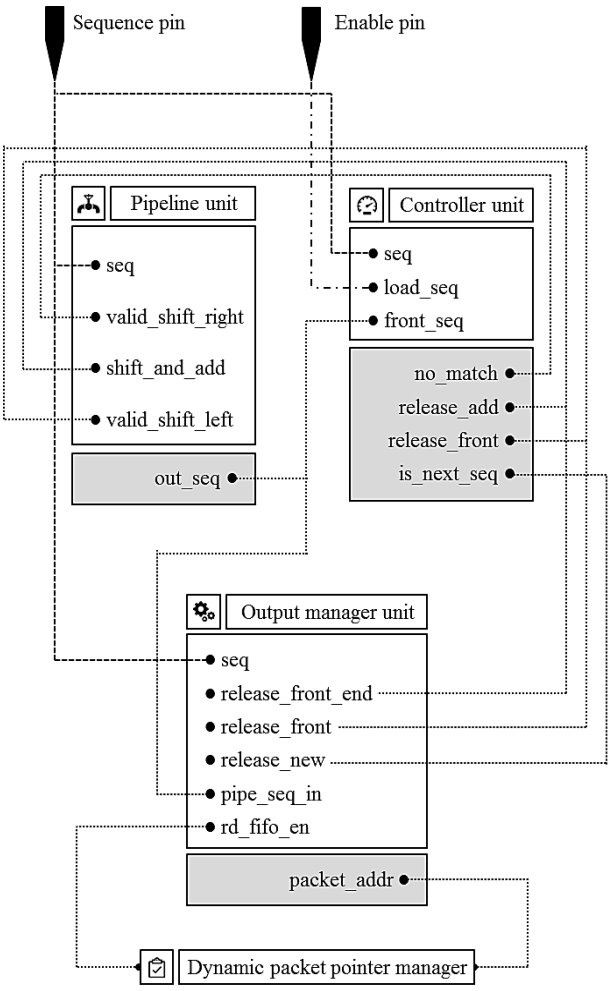

Fig. 4. Verilog HDL module architecture for FPGA prototyping with Quartus Prime

The standard BRAM capacity ranges up to 32 Kbits, with 32 bits of memory width and 1 Kbits of depth. Thereby for edge computing reconfigurable devices systems application, a single BRAM is adequate without the need for concatenation.

For the sake of simplicity in explanation, we have omitted the input manager module and external memory connections from our discussion. Once the input is received, the sequence number extracted from it is directed to the pipeline controller for sorting. The counter, which serves as an internal register within the controller, is responsible for keeping track of the current in-order value. The controller compares this counter with the incoming sequence and the first register of the pipeline.

To facilitate these functions, the controller is implemented using a finite state machine, which generates the appropriate command signals to direct the pipeline or the output manager. By means of this design, the pipeline can effectively sort the incoming sequence and relay the sorted data to the output manager for further processing or output. In order to facilitate shifting, the shift registers within the pipeline are interconnected with their left and right neighbors. This method of shifting all data bits at once in a single clock cycle is a commonly used technique in digital circuit design.

Comparator blocks1could be utilized within the pipeline as indicators. The output manager incorporates a multiplexer to choose between the sequence released from the pipeline or a by-passed value directly from the input manager, depending on the current operation. The output selected by the multiplexer is then directed to the FIFO queue. At this level, only the BRAM address from the pipeline's reference pointer is of significance. The queue identifies the corresponding BRAM memory location, matches it with the external memory address, and retrieves the entire packet.

SLX Silexica. The SLX Software Tool, developed by Silexica and currently integrated within Xilinx Company, serves as an additional platform for the creation of highly optimized FPGA-based architectures for computing purposes. The SLX FPGA platform functions as an overlay above the High-Level Synthesis (HLS) compiler, providing a suite of tools for optimizing code written in the high-level programming languages (C or C++). The structure of the SLX tool is depicted in Fig. 5.

Formalization of the SLX Software Tool work process allows building a model that includes the following components:

− Synthesizability Refactoring. At the Synthesizability Refactoring stage, the tool analyzes

C++ code and performs refactoring to adhere to SLX's own code standard.

− Parallelism Detection. At the Parallelism Detection stage, SLX analyzes code to identify the potential for parallelizing certain parts of the code. This stage determines the number of parts that can be parallelized and optimizes them for FPGA.

− Hardware Optimization. The Hardware Optimization layer is responsible for optimizing algorithms for specific hardware. Based on the characteristics of different hardware, SLX can optimize FPGA characteristics such as clock frequency, among others.

− Pragma Insertion. At the Pragma Insertion stage, SLX prepares sets of pragmas for the source code based on the previous stage. These pragma sets are instructions for the HSL compiler and can vary depending on the specific hardware platform.

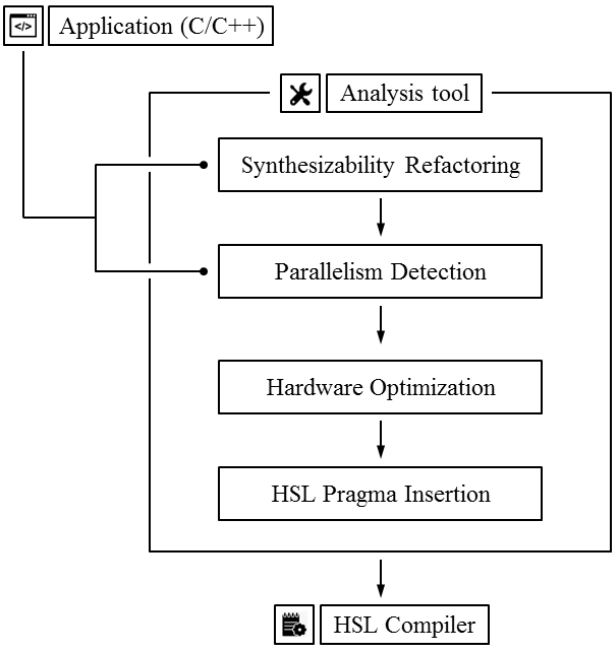

Fig. 5. Structure of SLX platform

Chameleon C2HDL. The Chameleon C2HDL design tool has been developed by the ASP to facilitate the automatic generation of HDL models from algorithms described in the ANSI C language [20]. By specifying an algorithm for data processing in ANSI C, a developer can obtain a thoroughly debugged and synthesizable VHDL RTL model of the device that implements the algorithm. In addition to the algorithm of the data processing, input information required by the Chameleon© C2HDL design tool includes the ASP's interface specification and technical characteristics, such as desired performance boundaries [21]. The platform for the ASP synthesis is configurable processor architecture configured according to the following input parameters:

− desired performance (the number of parallel Functional Units);

− the width of data structure;

− the minimal percentage of commands that should load each parallel Functional Unit;

− the communication network structure.

Figure 6 depicts the fundamental operational framework of the chameleon C2HDL design tool.

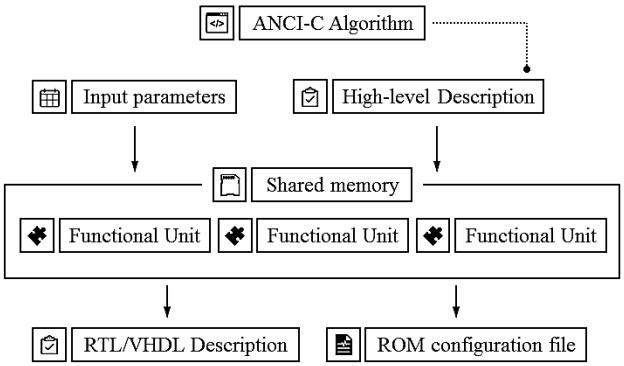

Fig. 6. Basic Scheme of Chameleon C2HDL Design Tool Operation

Future research. The primary research focus moving forward will center on the concept of combining edge computing architecture for IoT devices with reconfigurable hardware. The incorporation of reconfigurable hardware promises to enhance the flexibility of IoT devices and enable them to process data locally. The potential advantages of this approach include the ability to adapt data processing algorithms to specific environmental conditions and varying data loads, which in turn will result in IoT devices that are more energy-efficient and operate with maximal performance.

Conclusion. This article has examined several key concepts, including the Internet of Things (IoT), edge computing, and reconfigurable hardware. The analysis included an assessment of the contemporary toolset available for synthesizing hardware solutions utilizing Field Programmable Gate Array technology. Furthermore, this study introduced a novel avenue of research focused on the integration of reconfigurable processors for both edge computing and IoT devices.

Bibliography:

1. Sabourin V., Jabo J. T. IOT and the future of the Telecom Industry. *IoT Benefits and Growth Opportunities for the Telecom Industry*. Boca Raton, Florida : CRC Press, 2022. P. 57–59. DOI: 10.1201/9781003294412-6.

2. Mattihalli C., Gared F., Getnet L. Automatic plant leaf disease detection and auto-medicine using IOT Technology. *IoT-Based Intelligent Modelling for Environmental and Ecological Engineering* / eds.: P. Krause, F. Xhafa. Cham : Springer, 2021. P. 257–273. DOI: 10.1007/978-3-030-71172-6_11.

3. Kutseva M. Adaptation of seven-layered IOT architecture for energy efficiency management in Smart House. *2022 10th International Scientific Conference on Computer Science (COMSCI)*, 30 May 2022 - 02 June 2022. Bulgaria, Sofia : IEEE, 2022. DOI: 10.1109/comsci55378.2022.9912604.

4. Karpan A. *The Internet of Things.* New York : Greenhaven Publishing, 2022.

5. Minu R. I., Nagarajan G. Bridging the IOT gap through Edge Computing. *Edge Computing and Computational Intelligence Paradigms for the IoT*. Hershey, PA : IGI Global, 2019. P. 1–9. DOI: 10.4018/978-1-5225-8555-8.ch001.

6. Glance D. G., Cardell-Oliver R. Privacy of edge computing and IOT. *Secure Edge Computing* / eds.: M. Ahmed, P. Haskell-Dowland. Boca Raton, Florida : CRC Press, 2021. P. 83–98. DOI: 10.1201/9781003028635-7.

7. Kavitha V., Malathi V. A smart solar PV monitoring system using IOT. *First International Conference on Secure Reconfigurable Architectures & Intelligent Computing (SRAIC 2019)*, November 28-30, 2019. Tiruchirappalli : National Institute of Technology, 2019. P. 19-33. DOI: 10.5121/csit.2019.91502.

8. Sano Y., Kobayashi R., Fujita N., Boku T. Performance evaluation on GPU-FPGA accelerated computing considering interconnections between accelerators. *HEART '22* : Proceedings of the 12th International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies, 09-10 June 2022. New York : Association for Computing Machinery, 2022. P. 10-16. DOI: 10.1145/3535044.3535046.

9. Automatic energy-minimized HW/SW partitioning for FPGA-accelerated MPSoCs / Fuhr G. et al. *IEEE Embedded Systems Letters.* 2019. Vol. 11, No. 3. P. 93–96. DOI: 10.1109/les.2019.2901224.

10. Kim J.-H. FPGA design of an NB-IOT downlink transmitter using a Simulink model. *The Journal of Korean Institute of Communications and Information Sciences*. 2022. Vol. 47, No. 12. P. 2179–2191. DOI: 10.7840/kics.2022.47.12.2179.

11. Hong S., Park Y. A FPGA-based neural accelerator for small IOT devices. *2017 International SoC Design Conference (ISOCC)*, 05-08 November 2017. Korea (South), Seoul : IEEE, 2017. DOI: 10.1109/isocc.2017.8368903.

12. Kelati A., Gaber H. IOT for Home Energy Management (HEM) using FPGA. *2021 IEEE 9th International Conference on Smart Energy Grid Engineering (SEGE)*, 11-13 August 2021. Canada, ON, Oshawa : IEEE, 2021. DOI: 10.1109/sege52446.2021.9534986.

13. Harsha V. Implementation of low cost IOT based home automation system on SPARTAN FPGA. *International Journal of Recent Trends in Engineering and Research*. 2018. P. 513–516. DOI: 10.23883/ijrter.conf.20171225.078.eq2op.

14. Kim J.-H. FPGA design of an NB-IOT downlink transmitter using a Simulink model. *The Journal of Korean Institute of Communications and Information Sciences*. 2022. Vol. 47, No. 12, P. 2179–2191. DOI: 10.7840/kics.2022.47.12.2179.

15. Magyari A., Chen Y. *FPGA remote laboratory using IOT approaches. Electronics*. 2021. Vol. 10, No. 18. P. 2229. DOI: 10.3390/electronics10182229.

16. Daisy A. Neuroscience in FPGA and application in IOT. *Advances in Systems Analysis, Software Engineering, and High Performance Computing* / eds.: P. Sharma, & R. Nair. Hershey, PA : IGI Global, 2020. P. 97–107. DOI: 10.4018/978-1-5225-9806-0.ch005.

17. Kelati A., Gaber H. IOT for Home Energy Management (HEM) using FPGA. *2021 IEEE 9th International Conference on Smart Energy Grid Engineering (SEGE)*, 11-13 August 2021. Canada, ON, Oshawa : IEEE, 2021. DOI: 10.1109/sege52446.2021.9534986.

18. Oshana R. Overview of embedded systems development lifecycle using DSP. *DSP for Embedded and Real-Time Systems*. London : Newnes, 2012. P. 29–61. DOI: 10.1016/b978-0-12-386535-9.00003-2.

19. Hoang V. Q., Chen Y. Cost-effective network reordering using FPGA. *Sensors.* 2023. Vol. 23, No. 2, P. 819. DOI: 10.3390/s23020819.

20. Melnyk A., Melnyk V. Self-improvable computer system model and architecture based on reconfigurable hardware, automatic design and synthesis tools and artificial intelligence technologies. *Proceedings of the Fourth International Workshop on Computer Modeling and Intelligent Systems (CMIS-2021)*, April 27, 2021. Zaporizhzhia, 2021. P. 356-367.

21. Melnyk A., Melnyk V. Specialized Processors Automatic Design Tools-the Basis of Self-Configurable Computer and Cyber-Physical Systems. *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT*), 18-20 December 2019. Kyiv : IEEE, 2019. P. 326-335. DOI: 10.1109/ATIT49449.2019.9030481.

Майдан М.В. СИНТЕЗ ПРОЦЕСОРІВ ІОТ ПРИСТРОЇВ В ПЕРЕДЖЕВИХ ОБЧИСЛЕННЯХ: АНАЛІЗ ПІДХОДІВ

Досліджено сучасні підходи до організації інфраструктури мережі Інтернет речей (IoT). IoT – це глобальна мережа фізичних пристроїв, які можуть взаємодіяти один з одним і хмарою, обмінюватися даними та виконувати певні функції. Для ефективного функціонування такої мережі необхідна належна організація інфраструктури. Продемонстровано ефективність застосування концепції Edge Computing в Інтернеті речей (IoT). Концепція Edge Computing передбачає обробку даних на пристроях, розташованих ближче до джерела даних, а не на віддалених хмарних серверах. Це знижує навантаження на хмару та зменшує затримки передачі даних. У контексті Інтернету речей концепцію Edge Computing можна використовувати для обробки даних, отриманих від пристроїв IoT, на самих пристроях або на найближчих шлюзах, що може значно підвищити швидкість і ефективність обробки даних. Для забезпечення більш ефективної обчислювальної продуктивності та енергоспоживання стає популярним використання реконфігурованого апаратного забезпечення разом із периферійними обчисленнями. Реконфігурація обчислень передбачає використання пристроїв із гнучкою конфігурацією для обробки даних. У контексті IoT та Edge Computing пристрої з гнучкою конфігурацією можуть використовуватися для обробки даних, зібраних з пристроїв IoT, що дозволяє адаптувати обчислювальні ресурси до конкретного завдання та скорочує час обробки даних. Застосування концепції Reconfigurable computing в IoT і Edge Computing може бути особливо корисним у випадках, коли необхідно обробляти великі обсяги даних з високою точністю та швидкістю. В результаті дослідження запропоновано базову схему архітектури IoT та архітектури Edge computing системи, на основі якої сформовано архітектуру системи FGPA. Архітектура забезпечує гнучкість і масштабованість, необхідні для ефективної обробки великих обсягів даних у режимі реального часу, а також мінімізує навантаження на центральний процесор.

Ключові слова: *Інтернет речей, периферійні обчислення, реконфігуровані обчислення, програмовані вентильні матриці, високорівневий синтез, рівень регістрової передачі.*