UDC 004.8:004.312.466 **doi: 10.32620/reks.2023.3.03**

Artem PEREPELITSYN

National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine

METHOD OF CREATION OF FPGA BASED IMPLEMENTATION OF ARTIFICIAL INTELLIGENCE AS A SERVICE

The subject *of study in this article is the technologies of Field Programmable Gate Array (FPGA), methods, and tools for prototyping of hardware accelerators of Artificial Intelligence (AI) and providing it as a service. The* ***goal*** *is to reduce the efforts of creation and modification of FPGA implementation of Artificial Intelligent projects and provide such solutions as a service.* ***Task****: to analyze the possibilities of heterogeneous computing for the implementation of AI projects; analyze advanced FPGA technologies and accelerator cards that allow the organization of a service; analyze the languages, frameworks, and integrated environments for the creation of Artificial Intelligence projects for FPGA implementation; propose a technique for modifiable FPGA project prototyping to ensure a long period of compatibility with integrated environments and target devices; propose a technique for the prototyping of FPGA services with high performance to improve the efficiency of FPGA based AI projects; propose a sequence of optimization of neural networks for FPGA implementation; and provide an example of the practical implementation of the research results. According to the tasks, the following* ***results*** *were obtained. Analysis of the biggest companies and vendors of FPGA technology is performed. Existing heterogeneous technologies and potential non-electronic mediums for AI computations are discussed. FPGA accelerator cards with a large amount of High Bandwidth Memory (HBM) on the same chip package for implementation of AI projects are analyzed and compared. Languages, frameworks, and technologies as well as the capabilities of libraries and integrated environments for prototyping of FPGA projects for the AI applications are analyzed in detail. The sequence of prototyping of FPGA projects that are stable to changes in the environment is proposed. The sequence of prototyping of highly efficient pipelined projects for data processing is proposed. The steps of optimization of neural networks for FPGA implementation of AI applications are provided. An example of practical use of the results of research, including the use of sequences is provided.* ***Conclusions.*** *One of the main contributions of this research is the proposed method of creation of FPGA based implementation of AI projects in the form of services. Proposed sequence of neural network optimization for FPGA allows the reduction of the complexity of the initial program model by more than five times for hardware implementation depending on the required accuracy. The described solutions allow the construction of completely scalable and modifiable FPGA implementations of AI projects to provide it as a service.*

Keywords: *FPGA; FPGA-as-a-Service; FaaS; Artificial Intelligence; Artificial Intelligence-as-a-Service; AIaaS; hardware implementation of AI; heterogeneous computing for AI; neural network optimization; DPU.*

Introduction

The rapid evolution of the technologies for the creation of embedded solutions, processors, and hardware implementations allows the possibility of more complicated computations for enhanced algorithms. It enables improvements in the performance and complexity of algorithms in applications where the elements of AI are implemented.

The computations of such AI projects are typically based on the use of Central Processing Unit (CPU) or Graphics Processing Unit (GPU). To improve the efficiency of AI computations, it is possible to use dedicated hardware implementations.

FPGA technology assumes the possibility of implementation of both fast centralized computations and small low-power onboard solutions. The use of the newest accelerator cards during prototyping of FPGA projects simplifies the fast check of concepts and allows to prove the workability of algorithms [1]. At the same time, such solutions allow the acceleration of AI computations and can be installed at the datacenter.

The organization of such AI applications with FPGA implementations as a service makes possible the hardware acceleration of AI computations and remote access for the end user of the service [2].

However, this possibility requires improvements in the process of project creation, considering the problems of achieving high frequency for big projects and neuron networks optimization the implementation of for FPGA.

It is important to know how all such AI solutions are implemented and how to find the implementation of such accelerations of AI solutions.

The main purpose of this research is to reduce the efforts required for creation and modification of FPGA implementation of Artificial Intelligent projects and providing such solutions as a service. To reach this goal, it is necessary to perform the following tasks:

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

1) to analyze the possibilities of heterogeneous computing for Artificial Intelligence projects;

2) to analyze advanced FPGA technologies and accelerator cards that allow the organization of a service;

3) to analyze languages, frameworks, and integrated environments for the creation of Artificial Intelligence projects for FPGA implementation;

4) propose a technique for modifiable FPGA project prototyping to ensure a long period of compatibility with integrated environments and target devices;

5) propose a technique for prototyping FPGA services with high performance to improve the efficiency of FPGA-based AI projects;

6) propose a sequence of optimization of neural networks for FPGA implementation;

7) provide an example of the practical implementation of the research results.

1. Analysis of possibilities of heterogeneous computing for Artificial Intelligence

The projects involving the realization of neural networks and elements of AI are resource intensive and computationally intensive. However, the process of creation of such systems typically starts with the use of one platform that is then replaced with another during deployment or scaling the service.

It means that from CPU-only computations this service moves to a heterogeneous model involving acceleration with GPU [3] or even the combination with dedicated architectures, including FPGA and ASIC accelerators [4].

As an example, the Let's Enhance service was initially CPU-based service, but then the service was moved to dedicated implementations. This project allows to perform the graphical processing and the improvement of the quality of images based on the provided input file with the use of trained neural networks [5]. Based on the implemented algorithm, it is possible to improve the quality of the original picture using elements of AI.

This is an interesting service representing the best example of such services where AI solutions are implemented and available to the end user. They started with GPU implementation, but sometimes it is not easy to rent GPU facilities for processing. This is an example of how a graphic processing service that is available online can be implemented on a dedicated logic.

All the services that will attempt to move to dedicated solutions can be implemented in the same way and use the same approach.

Existing heterogeneous computing options for the acceleration of services with elements of AI include CPU, GPU, FPGA, and ASIC implementations. This is a possible implementation not only for AI but also for projects with their own architecture of computing.

For AI solutions, Vitis from Xilinx allows the combination of all these execution platforms for execution. It allows not only FPGA implementations of data processing but also the types of implementation of AI. It is already a form of heterogeneous computations for AI.

The search showed that there are many implementations of such computations. Some of them are elements of AI in the form of neural networks. However, it is interesting to find the potential form of wider ways of implementation of neural networks.

Water-based computations are really interesting from a practical point of view. This is the way of implementation of programming at the physical level with the use of liquid instead of electricity. It potentially may be useful in conditions where electronic implementations cannot be easily applied.

Different modifications of water droplet processors are the only possible forms of such a way of computing.

There are projects that consider the dynamic part of water instead of volume. Water waves are also a possible medium for computing [6].

The water integrator is also a well-known example of water-based calculations. This device was developed a century ago and implemented with a set of tubes with water that allows to simulate the complicated process of spreading temperature inside the concrete.

These examples show that it is possible to replace the normal computations with water as the medium. For AI projects is only a theoretical possibility for the moment, but in the case of use in combination with other mechanical ideas, it is an interesting perspective.

Photonic neural network implementation is one of the most interesting ways of performing heterogeneous computing for Artificial Intelligence [7]. The implementation includes neural network implementation inside the nanophotonic medium that can perform artificial neural computing.

The elements of AI are implemented directly inside this medium. This means that there is a modified structure of a solid medium where the set of modifications of the internal structure is performed. The neural network is implemented and can receive a set of decimal represented numbers. On the output after voting, it is possible to obtain the area with maximum probability at indicators with the exact number.

This means that it is possible to perform such implementations not only based on classical architecture. However, it is possible to involve such implementations using another kind of medium with solid, liquid, and other implementations of heterogeneous computing.

It is not a part of existing services, but it is important to take it into account to allow next steps and the future of AI services and AI as a whole.

2. Analysis of advanced FPGA technologies and accelerator cards to organize a service

The current status of FPGA technology allows the organization of fast communication with the host computer. The use of programming of the chip from the same program that controls the transfer of data allows the use of FPGA for the implementation of services.

Each service is based on a chips where the task can be executed. However, FPGA allows the acceleration of some computationally intensive tasks especially for AI.

The server that is based on the existing architecture uses the CPU or is based on the GPU accelerator. This server can run the service, and there are many services that use such acceleration.

However, at the same time, there are advantages in the dedicated hardware architecture. Because of this, only the dedicated solutions provide enough acceleration for specific tasks. The problem is that a dedicated task is developed for the exact project. However, the FPGA-based service allows the development of a dedicated architecture for the exact task.

It is important that only the highest level of integration of the chip that is measured by the technology process of a few nanometers allows the economical and power efficient implementation of such tasks, including the execution of AI solutions or AI projects.

The biggest FPGA vendors are also the CPU manufacturers. It is interesting that all modern vendors of FPGA chips are now vendors of CPUs. It is important to know that Intel company bought Altera. Then Microchip company that previously bought Atmel also bought Microsemi. Facilities of Actel were a part of Microsemi after 2010. Currently, this means that this set of families of FPGA chips is provided by Microchip [2].

The same process of integration is happening for AMD and Xilinx. Xilinx was smoothly integrated by AMD 3 years ago.

At the moment, the largest part of FPGAs chips is provided at the same or near the same technological process as CPUs. The percentage of total production of FPGA for a company is provided in Table 1.

Table 1

Part of production of leading FPGA vendors with total contribution above or near one percent

Percentage	Company (FPGA manufacturer)
52	AMD (Xilinx)
35	Intel (Altera)
5	Microchip (Microsemi)
5	Lattice
>1	Achronix
>1	QuickLogic
>1	Efinix

It is important to consider the set of such companies with contributions above or near one percent. It is interesting that the first four places are well known and the products of all those companies are frequently used for the development of all FPGA solutions including AI.

The first two are widely used in the ranges. The Microchip and Lattice are also used for such tasks. But first of all, they are used in embedded solutions.

However, for FPGA-based AI projects, it is important to have many resources, including fast memory and PCI express interfaces for fast shares between the system memory at the host computer and the FPGA chip with dedicated implementation of some neural network accelerator. Because of that, for the moment only the first two lines are interesting for the possibility of implementation of a service.

FPGA platforms for running AI services are produced in the form of dedicated accelerator cards. It is known that many services are based on such accelerators. Normally, they are installed in datacenters where many accelerators are placed together to perform hardware-based acceleration of AI tasks.

To accelerate AI systems, it is necessary to combine many individual FPGA accelerator cards into a single set at the datacenter. One host computer can handle all of these cards, and each card can be organized to perform individual AI processing or can work together for a single task for AI processing.

All implementations are run based on the FPGA accelerator cards. AMD together with Alveo provides many U-cards with different resources [8]. There are many Alveo FPGA accelerator cards with PCI express interface. They dedicated for the specific firmware that was preinstalled inside the circuit with FPGA.

Such cards are based on the modern FPGA chip with an efficiency comparable to or higher than that of modern powerful GPU chips. As a result, such cards can be used for accelerating dedicated tasks to implement the service with intensive computations for AI projects.

While U25 is first for network tasks, U280 and U50 are cards with 8 GB of second generation of High Bandwidth Memory. They may face a problem with normal performance if the passive version is used outside the datacenter conditions. Passive versions are developed for datacenters with centralized cooling. They can also be installed on a normal personal computer (PC).

One of the most powerful cards for such tasks is U280. The Xilinx company considers the new card U55C as the most powerful in this format [9]. However, a detailed comparison of this card with U280 card shows that the only advantage of this card is the increased on-chip size of HBM2E memory. There is 16 GB memory in the new card instead of 8 GB. This is the only significant difference.

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

During this comparison of the two cards, it is possible to find that there is no significant difference between them except for the size of the memory.

In fact, for modern AI solutions, a large amount of memory is required because many algorithms require memory on the chip. It is faster than using external memory because it is necessary to connect to some dynamic memory over the general bus. However, instead of that, it is possible to store the data inside the same package without transferring the data outside the chip. Especially because there are 32 independent wide channels with a width of each channel of 256 bits.

However, the disadvantage of this new U55C card is that the number of independent channels that allow connecting the part of the project to the exact memory block is the same as that in U280 and U50. However, at the same time, there are no useful resources in this card, including the external memory, like it was in U280.

External dynamic memory is dramatically slower than HBM, but for some processing algorithms, it is necessary to store the data between the processing operations, and 32GB of DDR4 memory is a good option because it is easy to access it from the on card chip.

VHK158 board also contains HBM2E memory of 32 GB, but this evaluation kit is designed in a bit another form [10]. It is considered a dedicated accelerator for machine learning.

UL3524 accelerator cards are targeted for custom algorithms and AI-enabled trading strategies [11]. This is one of the newest boards without HBM. Ultra-low latency in communications is the main priority for this board.

3. Analysis of languages and frameworks and technologies for creation of Artificial Intelligence projects for FPGA

Integrated Environments simplifies the creation of FPGA projects to perform computations for AI tasks. Program tools from Xilinx have a history of changes in versions.

Previously there was ISE tool that was then replaced by Vivado. It can be considered as a separate tool. But for now, after all changes and some specific tools like SDAccel, for usage is available only pair like Vivado and Vitis.

These tools focus on different tasks. For products of this company, Vivado is the main development environment. It is used for designing typical FPGA projects.

However, the Vitis tool is a specific tool that allows some steps of fast prototyping. It is really impotent for the development flow where some designers try to develop AI solutions without enough knowledge about the FPGA design flow or hardware description languages such as System Verilog, Verilog and VHDL.

The current process of development for FPGA includes the block diagram or circuitry manner of development of systems. Of course, it is possible to compose the big system based on the block design. It is great because it allows a visual check of how this system is implemented. And sometimes it is really useful.

However, it is also possible to use popular languages, including high-level languages, or to use HLS flow that supports the creation of a system directly in C and C++ languages or in OpenCL. This kind of description is called HLS. It helps developers without the experience of direct creation of systems for FPGA [12].

The advantage of using such technologies, languages, and frameworks is the possibility to implement AI solutions using OpenCL or even C and C++ language, and even with the use of Python, TensorFlow, and Caffe technologies. Actually, the last three technologies were added only in recent versions of Vitis. The last two are not the languages but frameworks.

Register Transfer Level (RTL) design flow is supported. In modern tools, RTL is the name of the set of hardware decryption languages, including System Verilog, Verilog, and VHDL. This is the default design flow for FPGA.

The most efficient FPGA projects are implemented using this set of technologies. Because with these languages, it is possible to achieve the highest accuracy of predictable implementation of data processing with the highest possible accuracy and efficiency.

On the other hand, the efficient use of these languages requires many years of education, practice, and experience in the use of these languages. It is possible to achieve the highest possible efficiency.

Python, TensorFlow, and Caffe are also supported. These technologies are used for prototyping of AI primitives and allow the process of building systems and AI components using the FPGA development flow.

Using this type of framework, it is very comfortable performing the description of the AI solutions.

The development environment, like Vitis, includes not only the ability to work with the kernels over the Xilinx Runtime (XRT) framework and communicate over PCI express interface, but also a lot of libraries to support AI development. The parts of these libraries are well known.

Deep Learning Processor Unit (DPU) provides the ability to fast prototype AI solutions with the use of FPGA. This library is already prepared for development and is available for beginners. This parametrizable IP-core was designed for convolutional neural networks and is described in PG338 from Xilinx [13].

In this description, it is possible to find the exact required parameters for this block. It can be used in both RTL and block design. Any kind of design can use instances of this block.

This is a parameterized IT core that is designed for such tasks and supports many existing and popular neural networks.

DPU is already prepared and is a default library. Ii is provided, created, and tested by the Xilinx company. It allows the implementation of popular algorithms that are frequently used in AI solutions.

The entire process of the description includes a few phases of convolution from the initial representation of the task and data to the final view of this representation. The parallelism of computations can be customized for the task. The supported convolutional neural networks are presented in Table 2.

Table 2

Popular convolutional neural networks supported by DPU

Algorithm	Description
VGG	To highlight image features
ResNet	Residual image classification network
GoogLeNet	Mage recognition neural network
YOLO	Neural network for object recognition
SSD	Detection of objects Single Shot MultiBox Detector
MobileNet	Optimized network for recognition
FPN	Multilevel object characteristics

Data Processing Unit is also called DPU because this abbreviation means not only deep learning processing unit, but also any type of DPU such as some independent computational module in system.

DPU is a frequently used name in such projects, and in each project, it means an independent unit that can perform AI or data processing tasks.

The entire hierarchy of AI task execution in the Xilinx environment includes a few elements starting from FPGA chip itself. The modifications of Ultrascale FPGA chip are used as the basis. For communication with this chip, the XRT framework is used. It helps to reduce the required efforts for the developer and the complexity of the implementation of communication with the environment.

A significant set of libraries prepared for the transformation of the system description into the internal representation allows the transformation of some well-known frameworks to the hardware descriptions that can run on this platform.

This is a significant result because it is possible to perform prototyping very fast and use the popular technologies that are used for AI development directly for FPGA projects.

In this case, the time to market is reduced by using the same technologies. At the same time, it is possible to use it directly for building optimized FPGA implementations as a platform for acceleration.

The disadvantage of this way of project creation is the duration of compilation.

The compilation process of such projects takes time. This process includes many steps that can be controlled in an integrated environment or performed separately from the command line.

After implementation, the modules of the AI project can be parameterized and connected to inputs and outputs. After parameterization and selection of the required algorithm the instances of DPU or another library look like the normal unit. It is a node with many inputs that can be connected to other inputs inside the project using the selected design and the selected manner of the development flow.

The process of compilation is based on the compilation of the project using Vivado. Normally, if the project describes the use of any technology or high-level languages, it will be represented internally like some big file in Verilog, VHDL, or System Verilog.

This description file may be up to 300-400 MB and contains the text of the entire project with the direct implementation of all components.

Then, it will be implemented or synthesized using Vivado tool as a part of a normal process that is hidden from the end user and the developer that uses Vitis flow. During the description of AI solutions with such languages and technologies, it is not necessary to know the details of implementation. However, it is impossible to optimize it without this knowledge.

Then, it can be built to obtain the binary file for such accelerator cards. The entire compilation takes from 5 to 10 hours of compilation using a very powerful hardware station or PC for compilation. It requires no less than 64 GB of dynamic memory. There are a lot of other requirements for the operating system and the program environment. However, after the compilation, the binaries will be obtained.

These binaries can be loaded with XRT framework on the FPGA chip. Thus, it is possible to obtain runnable versions of AI projects. Then, it will be available like a kernel and accessible from the host PC over the PCI express interface. The end user can run it in the form of calling a function in C-style language.

Communication with FPGA in case of use of such accelerator cards occurs over PCI express interface. To make it possible, the chip vendor introduced XRT framework that simplifies the communication with such FPGAs provided or implemented in the form of an evaluation kit or accelerator card to perform the communication over PCI express interface to achieve the highest possible throughput of the communication.

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

This framework reduces the efforts required for implementation of such development of communication, and for the end user, it looks like a simple call of function in C language or C-style function [14].

This means that at the host PC, the developer just calls some function inside the FPGA. This function is the kernel with the name inside the precompiled binary file loaded on FPGA. This kernel is an accelerator, a project that was built during a lot of hours or even days. It depends on the size of AI project. Then, it can be run really fast and enquired using the single line of code at the host PC.

The binary representation of the kernel should be provided from the host PC using a specific function that looks like OpenCL function to FPGA. Then, this kernel can be implemented inside the FPGA. This is the FPGA programming process. This is the process of configuration of SRAM-based FPGA and all the hardware modules inside the chip. Normally, it requires less than a second. With JTAG interface, it may take a few minutes.

Running the kernel includes the process of programming FPGA by loading the kernel and providing the required parameters and input data for processing. Then, the host PC provides the required task to the specific kernel. After the end of the computations inside the chip and at the end of the running of the kernel, the result of the computations can be returned to the host.

This iteration can be repeated many times. This technology allows the ability of the developer or user to perform the enquiring of many kernels during a second. It is possible to perform the reprogramming FPGA multiple times in seconds and perform practical computations.

The normal duration of such computations takes no longer than 10 s, because it is specific requirements of this framework.

The distribution of FPGA projects can be dome in the form of prebuilt binary files with kernels. From the point of view of the programmer this is really easy to run it because it is not necessary to know technical details how is it implemented. The implementation of AI solutions is important because typical AI projects require a set of specific operations that can be implemented in hardware.

The third party or the user can run this kernel, and it is not necessary to provide a lot of information about the implementation.

In addition, this data processing kernel considered as AI kernel can be implemented and provided as an FPGA library or protected IP-core. It is also possible to share the ready project with the intellectual properties and Artificial Intelligence inside. This is a possible way of running and distributing such solutions.

4. Proposed technique of prototyping of modifiable FPGA projects

Based on the results of the research and practical experience, it is possible to propose the following steps for prototyping the systems to avoid the problem of movement to the new platform.

To reduce the required efforts of porting FPGA project to new versions of integrated environments, it is necessary during the development flow to consider the following:

1) it is proposed to use best practices during the usage of the development environment and during the preparation of the project to maximize comparability;

2) to ensure the possibility of reconfiguration and modification of the project components that allows the transfer of the components to new versions.

3) it is recommended to use command line methods for automation of project compilation because this is the best practice for such processes and it will help to avoid problems with the graphical user interface during the movement to new versions of integrated environments;

4) it is possible to use direct communication with the company to check all the features of existing tools, frameworks, and libraries and for new versions that are planned for publication.

It is important to perform these steps to avoid the problem of porting these projects from one chip to another chip.

5. Proposed technique for prototyping of FPGA services with high performance

During the process of implementation using RTL languages, it is necessary to consider the specifics of the implementation of the chip. Some large FPGA chips include many independent silicon parts in the same package of the integrated circuit (IC). To perform the normal placement of the project inside FPGA it is necessary to know how this project can be divided between these parts or Super Logic Regions (SLRs) [15].

For the user, it is a single chip, but the developer needs to consider the total number of available connectors in each part of the chip. A typical powerful accelerator card is based on chips with 3 independent parts connected together over an internal interface. Depending on the package of the FPGA chip used for the exact AI project, a different number of such connectors is available for the developer [16].

Actually, some chips include a modification of really fast memory called High Bandwidth Memory even if HBM2 is used. In this case, SLR placement must ensure the use of the entire range of memory ports.

This approach reduces the complexity of routing of FPGA implementation of such AI solutions.

In addition, the implementation of AI solutions in FPGA requires the type of pipelining. This is required because the normal processing of the tasks is based on the out-of-order manner of operation, which is typical for algorithm implementations. Normally, the process of calculation of some data or information in FPGA-based AI project is a set of a few millions of complicated operations. However, it is possible to perform the pipelining of these operations.

It is possible to propose the following sequence of the optimization of the projects to improve the performance.

1. It is recommended to use the documents, including user guides for a specific chip from the vendor, to improve the efficiency of the implementation and routing of the chip.

2. It is proposed to use 2 or 3 registers in a chain to improve the frequency and simplify the placement of the project. This is especially relevant for kernels with communication using XRT framework.

3. It is proposed to use a short tag during the implementation of pipelined RTL-based projects to obtain identification data inside the chip. In this case, it is possible to perform some asynchronous processing and out-of-order operations in complicated computing systems.

4. It is proposed to use the mapping of a specific part of the project to the nearest super logic region inside the FPGA chip. It will help minimize cross-connections between super logic regions.

This sequence takes into account the number of elements and the different sizes of the connected blocks inside each element in the case of the description in RTL languages. It is necessary to perform the parameterization of such solutions to simplify the control of the elements.

6. Proposed sequence for neural network optimization for FPGA implementations

The optimization of neural networks for FPGA implementation is an important step in the acceleration of AI computations. The popular data types cannot be directly used during the creation of the dedicated version.

It is interesting that normally for the implementation of neural networks, it is easy to use a typical CPU and to perform such program implementation of the application. However, to allow efficient implementation using FPGA, it is necessary to perform the optimization.

It depends on the exact solution, but it is technically possible. For example, normally, the value can be represented in such AI project implementation like the value from 0 to 1 like a real number. However, during optimization for FPGA, it is necessary to transform it to some quantized number of states. It is possible to represent the number of possible states. For example, only 100 states are possible. This means that only 7 bits is enough to represent this model instead of 32 bits required for implementation of the 4-byte float point value. Based on this value, it is possible to replace the float point-based implementation of AI solution with a fixed point. Therefore, it is important to perform the usage only fixed point implementation.

It is proposed to use the following steps of neural networks optimization for FPGA implementation.

1. In the first step, it is proposed to reduce the number of implemented nodes. In addition, at these steps, it is necessary to determinate the possible number of levels of representation of possible states of such neural network. It is possible to prune unnecessary connections and unused lines with less priority. It is necessary to do this to make it possible to reduce the complexity of the neural network.

2. It is necessary to perform the switching to a fixed point representation of voting conditions. It is necessary to use a fixed-point data type or an unsigned binary representation instead of a float data type. It allows the reduction of the complexity of FPGA implementation at least in 4 or 5 times depending on the required accuracy.

3. After all, these steps, it is possible to take this model and represent it as a single unit that performs operations. Then, it is necessary to combine the obtained blocks and represent such blocks as DPU block. Then, it is necessary to combine the set of commands, the data for processing, and the DPU to execute such AI data processing. Inside a single chip, it is possible to place many such units.

It is important to consider the advantages of optimizing the project based on the proposed steps.

7. Example of practical implementation of research results

To create a service that performs AI computations in FPGA, it is necessary to formalize the task and follow the proposed steps. Selection of the required neural network model is required.

A prototype image analysis service using a convolutional neural network was created [17]. To improve the performance after the creation of the first prototype, the RTL flow was selected as the method of description.

The proposed sequences of neural network optimization, modifiable project prototyping, and FPGA project optimization were used during the creation of the service prototype. Two target devices, U50 and U280 accelerator cards were selected.

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

Optimization of the SLR placement was also performed as the key point of the final steps of performance optimization. The created project uses the HBM to transfer data with the host application.

This part of the project was moved to SLR0, which is connected directly to High Bandwidth Memory inside the chip. Using this modification, the final frequency for the AI project was improved by reducing the number of connections between different super logic regions in the chip.

After the optimization and placement, it is possible to use the floor plan viewer to check the efficiency of such modification of parameters and to check how they are implemented inside the FPGA chip (Figure 1).

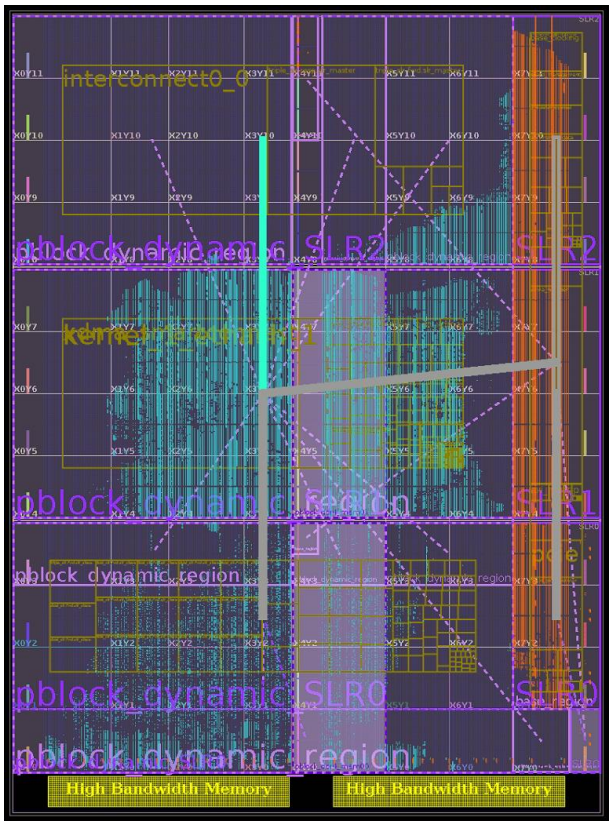

Fig. 1. Results of optimization of placement of the project part with the kernel interface to SLR0

This view of the chip is obtained in Vivado with U280 as the target device. In the figure, it is possible to see the placement of parts of the project inside the FPGA chip that consists of 3 separate silicon super logic regions connected together using interconnects with a limited number of lines. Therefore, this is the view of 3 chips inside the single package.

This optimization allowed to achieve higher values of frequency for the project after compilation. To assign the required frequency the XOCC for SDAccel and then v++ parameters for Vitis were used. The parameter kernel_frequency specifies the frequency. To reduce the duration of compilation no_ip_cache parameter is used.

8. Discussion

This study considers the possibilities of FPGA technology for creating AI applications and services. The analysis shows that at present, all chips and FPGAs from the biggest companies are provided at the same or near the same technological process as CPUs and GPUs.

Results of the research focused on the implementation of Artificial Intelligent services to provide such projects as a service. This idea is based on the possibility of using the FPGA for the implementation of such Artificial Intelligent components.

For the end user, it is possible to represent some service or AI-based service like the end product. However, this service can be implemented using hardware accelerators based on FPGA technology.

It is possible to create and use the basic elements of AI algorithms and support computationally intensive algorithms using FPGA technologies. The tools considered by the Xilinx company allow the preparation of such basic components for the implementation of neural networks and even for the high-level synthesis of AI prototypes. The DPU block can be considered as one of the possible ways to implement AI in such FPGAs.

The integrated environments like Vitis include not only the ability to work with the kernels over XRT framework and communicate over PCI express interface, but also a lot of well-known libraries to support AI development. In addition, it is proposed to use memory-intensive AI algorithms using such chips because from 8 to 16 GB or even 32 GB of fast dynamic memory is available with direct connection to the chip for such implementations.

Thus, FPGA chips and technologies of design significantly simplify the possibility of implementing AI applications with supporting popular languages and frameworks. However, ensuring the required performance of the project is a rather difficult task.

Conclusions

The main result of this research is the consideration of technologies, languages, and tools for the creation of AI applications with FPGA implementation as a service.

The main advantage of the use of such FPGA elements is the significant improvement in acceleration with this technology. It is also possible that the implementation and prototyping of such elements will allow the acceleration of AI systems.

In addition, it is possible to use the modifications of High Bandwidth Memory in the same package with the chip for the implementation of resource- and memory-intensive AI algorithms and implementations of such algorithms.

The proposed sequence of prototyping of easily modifiable projects reduces the required efforts during the change of version of the integrated environment or porting of the project to another platform.

The proposed sequence of prototyping of scalable, highly efficient pipelined FPGA solutions allows to ensure an out-of-order manner of operation while considering the specifics of placement of the project inside Super Logic Regions.

The proposed sequence of optimization of neural network for FPGA implementation allows to reduce the complexity of the initial program model by 5 times or more. With two previous sequences, it allows the creation of FPGA-based implementations of AI as a service.

The prototype AI service was developed, trained, and tested. The main result of the next step in this area can be based on this research and consists of 3 main elements. AI as a service can be considered some Artificial Intelligent functions or functionality as a final cost product for the end user. In addition, FPGA accelerators can be used to improve the performance, reliability, and security of such solutions. On the other hand, both kinds of implementations, cloud-based or some edge-based AI prototypes, can be used as a basis for the implementation of FPGA and AI as a service. This is the main result and the main direction of the research.

Further research would be based on consideration of work of such AI services with hardware implementation at the detailed model level to allow efficient mechanisms of theoretical estimation and prediction. It is also interesting to consider the stability of such solutions using approaches acceptable for FPGA systems and cloud services.

Contribution of author: tasks formulation, analysis of heterogeneous computations, theoretical, practical and experimental research of Xilinx tools, frameworks and accelerator cards, formulating of three sequences, creation and optimization of the prototype of FPGA based service, manual assignment of location of the project part inside the chip, analysis of publications and user guides, writing the text – **Artem Perepelitsyn**.

The author has read and agreed to the published version of the manuscript.

References

1. 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.

2. Perepelitsyn, A., & Kulanov, V. Technologies of FPGA-based projects Development Under Ever-changing Conditions, Platform Constraints, and Time-to-Market Pressure. *Proceedings 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies, DESSERT 2022*, 2022, pp. 1-5, DOI: 10.1109/DESSERT58054.2022.10018828.

3. Hu, N., Wang, C., & Zhou, X. FLIA: Architecture of Collaborated Mobile GPU and FPGA Heterogeneous Computing. *Electronics,* 2022, vol. 11, iss. 22, no. 3756. pp. 1-14. DOI: 10.3390/electronics11223756.

4. Liang, B., Wang, S., Huang, Y., Liu, Y., & Ma, L. F-LSTM: FPGA-Based Heterogeneous Computing Framework for Deploying LSTM-Based Algorithms. *Electronics,* 2023, vol. 12, iss. 5, no. 1139. pp. 1-15. DOI: 10.3390/electronics12051139.

5. *How does Let's Enhance increases image resolution?* Available at: https://help.letsenhance.io/en/article/how-does-lets-enhance-increases-image-resolution-3p5z0p/ (accessed February 28, 2023).

6. Maksymov, I. Analogue and Physical Reservoir Computing Using Water Waves: Applications in Power Engineering and Beyond. *Energies*, 2023, vol. 16, iss. 14, no. 5366. pp. 1-26. DOI: 10.3390/en16145366

7. Khoram, E., Chen, A., Liu, D., Ying, L., Wang, Q., Yuan, M., & Yu, Z. Nanophotonic media for artificial neural inference. *Photon. Res.* 2019, vol. 7, iss. 8, pp. 823-827. DOI: 10.1364/PRJ.7.000823.

8. *Alveo Product Selection Guide, Data Center Accelerator Cards, Xilinx.* Available at: https://www.xilinx.com/content/dam/xilinx/support/documents/selection-guides/alveo-product-selection-guide.pdf. (accessed February 28, 2023).

9. *Alveo U55C Data Center Accelerator Cards Data Sheet, AMD, DS978 (v1.3).* Available at: https://docs.xilinx.com/r/en-US/ds978-u55c. (accessed June 23, 2023).

10. *VHK158 Evaluation Board User Guide, AMD, UG1611 (v1.0).* Available at: https://docs.xilinx.com/r/en-US/ug1611-vhk158-eval-bd. (accessed July 18, 2023).

11. *Alveo UL3524 Ultra Low Latency Trading Data Sheet, AMD, DS1009 (v1.1).* Available at: https://docs.xilinx.com/r/en-US/ds1009-ul3524. (accessed September 27, 2023).

12. Barkovska, O., Filippenko, I., Semenenko, I., Korniienko, V., & Sedlaček, P. Adaptation of FPGA architecture for accelerated image preprocessing. *Radioelectronic and Computer Systems*, 2023, no. 2, pp. 94-106. DOI: 10.32620/reks.2023.2.08

13. *Zynq DPU Product Guide, Xilinx, PG338 (v3.3).* Available at: https://docs.xilinx.com/r/3.3-English/pg338-dpu. (accessed February 28, 2023).

14. Perepelitsyn, A., Fesenko, H., Kasapien, Y., & Kharchenko, V. Technological Stack for Implementation of AI as a Service based on Hardware Accelerators. *Proceedings 2022 IEEE 12th International Conference on Dependable Systems, Services and Technologies,*

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

DESSERT 2022, 2022, pp. 1-5. DOI: 10.1109/DESSERT58054.2022.10018615.

15. *UltraFast Design Methodology Guide for Xilinx FPGAs and SoCs, Xilinx, UG949 (v2021.2).* Available at: https://docs.xilinx.com/r/2021.2-English/ug949-vivado-design-methodology/SLR-Utilization-Considerations. (accessed February 28, 2023).

16. *Vivado Design Suite Properties Reference Guide, Xilinx, UG912 (v2022.1).* Available at: https://docs.xilinx.com/r/2022.1-English/ug912-vivado-properties/USER_CROSSING_SLR. (accessed February 28, 2023).

17. Perepelitsyn, A., Kasapien, Y., Fesenko, H., & Kharchenko, V. Technologies for Implementing of Artificial Intelligence as a Service based on Hardware Accelerators. *Aerospace Technic and Technology*, 2022, no. 6, pp. 57-65. DOI: 10.32620/aktt.2022.6.07.

Received 31.07.2023, Accepted 20.09.2023

МЕТОД СТВОРЕННЯ ШТУЧНОГО ІНТЕЛЕКТУ ЯК СЕРВІСУ НА ОСНОВІ FPGA РЕАЛІЗАЦІЇ

Артем Перепелицин

Предметом вивчення в даній статті є технології FPGA, методи та інструментальні засоби для проєктування апаратних прискорювачів для реалізації штучного інтелекту (ШІ) і надання його в якості сервісу. **Метою** роботи є зменшення трудовитрат на створення і модифікацію FPGA реалізації проєктів штучного інтелекту та надання таких рішень як сервісу. **Завдання:** проаналізувати можливості використання гетерогенних обчислень для реалізації проєктів штучного інтелекту; проаналізувати передові технології FPGA і карти прискорювачів, що роблять можливою організацію сервісу; проаналізувати мови, фреймворки та інтегровані середовища розробки для створення FPGA реалізації проєктів штучного інтелекту; запропонувати послідовність створення проєктів FPGA, які передбачають можливість легкої модифікації, для забезпечення тривалого періоду сумісності з інтегрованими середовищами та цільовими пристроями і картами прискорювачів; запропонувати послідовність створення FPGA сервісів з високою продуктивністю для підвищення ефективності проєктів AI на їх основі; запропонувати послідовність оптимізації нейронних мереж для реалізації в FPGA; навести приклад практичної реалізації результатів дослідження. Відповідно до поставлених завдань, були отримані наступні **результати.** Проведено аналіз найбільших компаній і виробників FPGA. Обговорюються існуючі гетерогенні технології обчислення та потенційні неелектронні середовища для виконання ШІ обчислень. Виконано аналіз та порівняння карт FPGA прискорювачів з великим об'ємом динамічної пам'яті всередині корпусу чіпа для реалізації обчислень проєктів ШІ. Детально аналізуються мови, фреймворки та технології, а також можливості бібліотек для побудови проєктів ШІ. Запропоновано послідовність прототипування FPGA проєктів, стійких до змін середовища. Запропоновано послідовність прототипування FPGA проєктів для високоефективної конвеєрної обробки даних. Наведено етапи оптимізації нейронних мереж для реалізації в FPGA. Наведено приклад практичного використання результатів дослідження. **Висновки:** Один із головних внесків цього дослідження полягає в запропонованому методі створення реалізації проєктів штучного інтелекту на основі FPGA у формі сервісів. Запропонована послідовність оптимізації нейронної мережі для FPGA дозволяє знизити складність вихідної програмної моделі більш ніж у 5 разів для апаратної реалізації в залежності від необхідної точності. Описані рішення дозволяють будувати повністю масштабовні та модифіковні реалізації FPGA проєктів штучного інтелекту, щоб надавати їх як сервіс.

Ключові слова: FPGA; FPGA як сервіс; FaaS; штучний інтелект; штучний інтелект як сервіс; AIaaS; апаратна реалізація ШІ; гетерогенні обчислення для ШІ; оптимізація нейронної мережі; DPU.

Перепелицин Артем Євгенович – канд. техн. наук, доц., доц. каф. комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Artem Perepelitsyn – PhD, Associate Professor of Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.perepelitsyn@csn.khai.edu, ORCID: 0000-0002-5463-7889, Scopus Author ID: 56332607800.