Systematic Analysis of FPGA-Based Acceleration for Deep Neural Networks

Mingyang Xu *

Department of Electronic and Electrical Engineering, University College London, London, WC1E 6BT, UK

* Corresponding Author Email: mingyang.xu.25@ucl.ac.uk

Abstract. This paper systematically reviews and evaluates the architectural characteristics, application status, and platform comparisons of FPGAs for deep neural network (DNN) acceleration. First, this paper outlines the foundation of Field-Programmable Gate Array-based (FPGA-based) reconfigurable hardware based on dataflow and parallelism. Subsequently, by combining representative designs and case studies, this paper summarizes FPGA advantages in low latency, energy efficiency, and operator customization, as well as bottlenecks. Compared to Graphics Processing Units (GPUs)/ Tensor Processing Units (TPUs), which are more suitable for large-scale training and high-throughput inference, FPGAs offer cost-effectiveness in deterministic low latency, small-batch/real-time scenarios, and specific protocol/operation customization, making them suitable for cloud-edge collaboration and industrial embedded applications. Looking ahead, technologies such as High Bandwidth Memory (HBM) and hierarchical caching, near-memory/near-computing, tensor- and dataflow-based reconfiguration coverage, model hardware co-optimization based on quantization and sparsity, automated compilation, heterogeneous Central Processing Unit (CPU)/ Artificial Intelligence (AI) engine/ FPGA System on Chips (SoCs), and chiplet/3-Dimension (3D) packaging will further lower the design barrier and improve system efficiency. This article aims to provide a reference for selecting heterogeneous computing capabilities and designing FPGA accelerators in different scenarios.

Keywords: FPGA, AI acceleration, Deep Neural Networks, low latency.

1. Introduction

FPGA is a hardware device with flexibility and high performance, positioning itself between CPU and ASIC, which can be reconfigured after production [1]. Due to its features of programmable and parallel computing, FPGA plays an increasingly important role in signal processing, embedded systems, and artificial intelligence fields. Especially in AI inference and edge computing, FPGA has become a potential core platform in deep neural networks and the deep learning acceleration field due to its low latency, high performance and customizable architecture.

However, there are still several bottlenecks of FPGAs in AI acceleration, including complex hardware programming, long development cycle, the lack of a unified software ecosystem and limited on-chip resources. These problems restrict the optimization and widespread adoption of FPGAs in large-scale AI deployments [2]. Therefore, the researchers started to explore high-level integrated tools, professional architecture design and methods of optimizing automatically, which aim to increase the efficiency of development and performance in the AI field. Continuously researching FPGA acceleration in depth has a significant meaning in achieving highly efficient, flexible and energy-saving intelligent computing systems.

This research discusses the application and development of FPGAs in the AI acceleration field. The article first makes a basic analysis of the components of an FPGA, the logic units and the interconnect structure at the architecture level. Then, the paper reviews the process of recent AI models that use FPGA acceleration and compares the characteristics and performance with the main platforms, such as GPU and TPU, evaluating FPGA on the potential and limitations of computing efficiency, energy consumption control and system flexibility. In addition, the research analyses the challenges that FPGAs face in AI acceleration, such as energy consumption optimization, limitations on algorithm portability, and excessively long compilation times. In the end, combining the

development of new-come architecture trend and open-source tools, this research has an expectation on the evolving direction and application prospects in the future intelligent computing system.

2. Architecture of FPGA

FPGA is a type of digital integrated circuit, which can be reconfigured by the users after manufacturing. Its core characteristics are programmability and the high parallelism of the hardware structure. A typical FPGA architecture basically consists of three components: Logic Elements, Programmable Interconnects and Input or Output (I/O) Blocks, which is shown in Fig. 1.

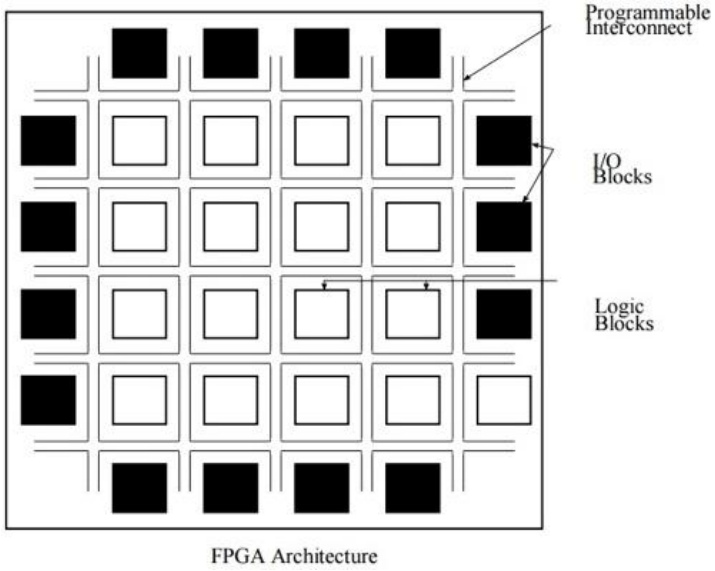

Figure 1. The three components of an FPGA Architecture [3]

Logic Elements generally include a Look-up Table (LUT), a Flip-Flop (FF), and a small quantity of algorithm logic resources, which are used to achieve basic logic computing and time-order control shown in Fig. 2. This programmable interconnect structure flexibly connects each logic unit through a multi-layer switch matrix to form a customizable circuit path, thereby enabling different circuit functions. I/O Blocks are used to interconnect with external devices' signals, aiming to realize multiple communication interfaces at a systematic level for the FPGA.

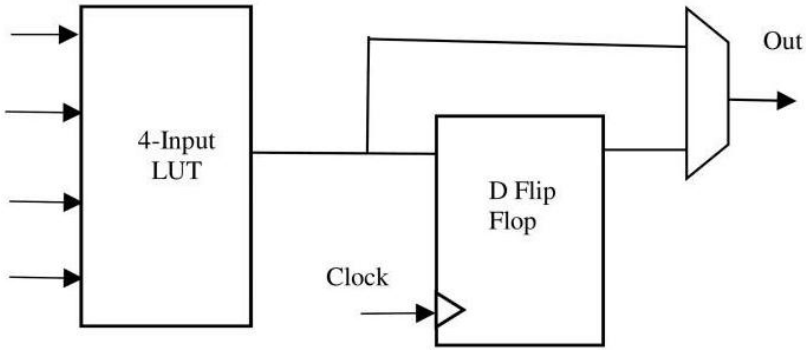

Figure 2. The structure of a logic element [4]

In addition, modern FPGA chips integrated specialized hardware resources such as Block RAM (BRAM), digital signal processing Slice (DSP Slice) and an on-chip clock management module, etc., to increase the computing density and ability of processing data. This flexible and reconfigured architecture takes advantage of FPGA on performance, energy efficiency and adaptability, which have become significant components of AI accelerators and embedded computing platforms.

3. Analysis and Comparison of FPGA, GPU and TPU

3.1. FPGA

It makes an expectation in [5] of the future of the data center and the FPGA accelerator used in the Cloud. This research gives an evaluation of applications such as deep learning, network and storage, and the sort of architecture. It demonstrates that the systematic structure facing to cloud and data center, and the visualization of FPGA. It refers to the sort of details of the architecture of the deployment form, such as bump-in-the-wire and co-processor configuration. Specialization and reconfigurability are advancing simultaneously, which implies the need for stronger abstraction, reliable virtualization, a robust tool ecosystem, as well as deep integration of high-bandwidth memory/high-speed interconnects.

Reference [6] reviews the choice of target detection on GPU/ASIC/FPGA, and focuses on the design target, methods and challenges of FPGA. It demonstrates the reason for using an FPGA as the accelerator, which is FPGA can customize parallelism, low latency, and better energy efficiency. However, as the convenience FPGA can bring to the accelerator, it also has some drawbacks, such as the complexity of design and the threshold of the tool chain. The purpose of designing the accelerator is to create a chip with high handling capacity, low latency, low energy consumption, small area, and easy to transplant. The typical method to design an accelerator chip is to reuse the on-chip data, task/data parallel partitioning and row-stationary. The main challenges are the bandwidth bottleneck, algorithm-hardware collaboration and load balancing of the detection head branch. This design is friendly to juniors and selecting solutions, and is proper to serve as a roadmap for accelerating testing and a comparison baseline.

A flexible and highly efficient FPGA accelerator is referred to in [7], which aims to support standard convolution, depth wise and pointwise convolution (DWCV/PWCV), transposed convolution (TPVC) and dilated convolution (DLCV) in the same reconfigurable hardware. This method can solve the low reuse of the tight-weight model data and the imbalance load due to the large quantity of zero values of TPCV/DLCV. The key mechanism is that Zero Transfer & Skipping (ZTS) rearranges the computing of TPCV, avoiding the ineffective sum or multiple computing of zero. Another key point is that the Sparsity-like Processing (SAP) processes the rule sparsity of DLCV/TPCV using "weight-oriented data flow". The platform used is Intel Arria-10 Soc, up to 339.9 FPS. It only adapts the embedded platform, which requires altering different CNN missions, such as segmentation, classification, and super-resolution. However, it still needs optimization for the extreme model.

Reference [8] refers to the first coalesced Tsetlin Machine self-contained FPGA accelerator, supporting 10-class classification for 28×28 binary images, enforcing low power consumption and interpretability. The core design is that the parallelism of 128 logical clauses, the realization of the 34 BRAMs of Tsetlin Automata (TAs); clause weights are serialized with action signals to accelerate inference; "Random patch selection" during convolution is stored in BRAM + reservoir sampling; the impact of the bit width of the training random number LFSR on the accuracy was studied. The platform used is Xilinx Zynq. The attractions are that the Tsetlin Machine mainly adopts logic and sum operations, which have low algorithms and high energy efficiency, and has interpretability compared to DNN/BinaryNN. It also realized on-chip training, which is friendly to the edge nodes of the supply batteries. The technology referred to in [6] is suitable for edge vision with limited resources and the need for online adaptive capabilities; the accuracy on more complex datasets (such as CIFAR-10) is still limited, and a composite TM or a larger configuration is required.

3.2. TPU/GPU

3.2.1. Performance analysis of TPU and FPGA

The research in [9] discussed matrix arrays and high-bandwidth memory design of TPU and cloud rent mode. The research used TPUv3 as an example, which has 4 chips, and each chip has 2 cores. Every core consists of scalar/vector units and 128*128 MXU, with 16 GB HBM at the same time.

Every MXU can realize 16000 multiply or sum action every time. For the mode of using a TPU, it is mainly used by cloud rental, instead of being purchased by enterprises. Therefore, the physical card is not sold to the public. By contrast, an FPGA can be reconfigured using HDL after manufacturing, featuring local/on-chip storage and supporting various data types such as integers and floating-point numbers. However, FPGA often relies on vendor-specific toolchains and model conversion (such as Open VINO), which impose limitations on micro-benchmark/operation-level experiments; the counting metrics of monitoring counters vary across different vendors, posing a risk of vendor lock-in [10].

In the research [11], it sorts TPU into a type of NPU (parallel to Huawei Ascend and Cambrian), and refers to the heterogeneous architecture of CPU/FPGA/NPU to manage the collaboration of two types of systems, CPU/FPGA and CPU/NPU. The platforms used are Intel Cyclone 10GX FPGA and Huawei Ascend 310p NPU.

3.2.2. Performance analysis of GPU and FPGA

The research [12] discusses the three types of deep learning processors (DLPs) used in CNN, which are GPU, FPGA, and ASIC, and gives the typical architecture and the statistics of the experiment. The GPU accelerator is mainly used in parallel linear algebra; however, the RAM (such as 12-16 GB for P100) is restricted, and the SSD (such as 128GB) is much larger than the RAM. The deep-level model will bring high pressure to the RAM storage. The software ecosystem includes CUDA and the acceleration library. The representative hardware includes V100, which has 16/32 GB RAM and a speed of up to 900 GB/s; Tesla T4, which has 16 GB RAM and a speed of 300 300GB/s; and A100, which has 40 GB RAM and a speed of up to 1555 GB/s. The result tested in reality [12] indicates that the convolution time of the GPU is shorter than the CPU under multiple frameworks.

The main energy consumption of the FPGA accelerator mainly comes from the data interconnection between the processor and DRAM. Therefore, we need to balance between the computing speed and the energy consumption. A typical architecture is DLAU, which consists of TMMU/PSAU/AFAU, by using tiles, FIFO and pipelines to reduce external memory access and reuse computing units, supporting user-adjustable trade-off between energy consumption and speed. Tests show that the power consumption of DLAU is 234 mW, which is lower than 485 mW of ASIC, but the total power consumption with the host computer amounts to 1814 mW. GPU is the most mature but not flexible enough, and has higher energy consumption and cost; GPU has obvious advantages in mobile applications and matrix calculations; the tensor cores of NVIDIA Volta/Turing have been well modelled (with an IPC correlation of 99.6% compared to Titan V). FPGA is flexible and has low energy consumption but relatively inferior performance; there is a lack of a similar mature toolchain for FPGA, like CUDA, but it can evolve through the OpenCL path; promoting co-processing of storage and computation is a potential direction; it is recommended for applications such as aerospace in-orbit processing, energy-saving cloud computing, and rapid iteration in laboratories.

Under all test benchmarks, the energy efficiency of the FPGA is all higher than GPU (GTX 1080) and CPU (i7-7700K), for example, the energy efficiency of the FPGA in the MM benchmark is approximately 6.3 times that of the GPU [13]. Zhang et al. argue that GPUs are more suitable for high-throughput tasks involving "massive parallel data processing", while FPGAs have an advantage in terms of energy consumption.

The research done by Wang [14] is about the energy consumption and cost of FPGA and GPU. The energy consumption is extremely high of GPU due to a large quantity of transistors rotating in each clock. There are two examples in the research of Wang. The energy consumption of Tesla V100 is approximately 250W; however, the energy consumption of Smart SSD (which is an FPGA chip) is only approximately 30W. In the CSV parsing experiment, the performance and power consumption of Smart SSD were approximately 25 times better than that of V100. By analyzing the works, we can summarize that the GPU has higher performance, higher energy consumption and medium cost, while the FPGA has medium CPU performance, medium energy consumption and low cost.

The key points given by Sano et al. [15] are GPU has a high peak value and high bandwidth, but has a weak performance for applications that have bad parallelism and irregular computation or

frequent interconnection between nodes. In this case, an FPGA can construct flow hardware that specialized for the application and storage system to compensate.

Alshemi et al. make an argument on the performance of the GPU and FPGA in lane detection. They use NVIDIA Tesla K80 as a GPU sample, while using ZYNQ-7 ZC706 as an FPGA sample [16].

Table 1. Comparison of GPU and FPGA property

	GPU	FPGA
Latency	4.874 ms	2.62 ms
Energy Consumption	74.22 W	1.619 W
Internal Storage Usage	6.5 MB	2.88 MB

By comparing the comparison of GPU and FPGA, we can figure out that FPGA has lower latency than GPU. The GPU has better internal storage resource scalability at higher resolutions. If a high resolution is desired, the GPU might be more suitable. If performance and delay are critical while resolution is limited, the FPGA is better. In this 512*512 scenario, the FPGA performs better.

3.2.3 Analysis of Deep Neural Networks and FPGA

Thang put forward a customizable hardware architecture that can realize feedforward Deep Neural Networks (DNN) [17]. It reused a single physical computing layer and control units on the multilayer computing of the networks. It goes through the time order and multiplexing to alter at each layer to increase the rate of use of the area. The internal structure of the physical layer consists of S parallel "neurons", each of which includes a dot-product unit, an activation function, and a hierarchically selected weight storage.

Yufei et al. analyzed the acceleration of CNN on FPGA [18]. The method they took is to establish three storage levels: External DRAM, On-chip buffer, Register and PE, then abstract the four cycles of convolution: unrolling, tiling, and interchange, to design variables. Inferring computing period, partial sum storage, and data reuse, it can choose the maximum reuse and minimum communication. Yufei et al. also gave the unified and reusable convolutional data flow and architecture, including an irregular data router for handling strides or fillings. The key points made in the research are that the FPGA has low latency and approximately 10-50 GOP/s/W energy efficiency advantages in deep neural networks. At the same time, it also referred that an FPGA need to optimize the parallelism, chunking and reordering due to the limitation of on-chip resources.

3.3. The Comparison of GPU, TPU and FPGA used in Deep Neural Networks

By a comprehensive comparison of GPU, TPU and FPGA, I can get the result that: GPUs are suitable for large-scale parallel SIMT processors and can perform matrix operations with tensor cores, making them the main choice in the training field. TPUs are designed for dedicated ASICs for ML, with their core being large pulse arrays and cloud interconnection. They excel in large-scale training and high-throughput inference. FPGAs have reconfigurable logic and can build data flow pipelines on demand. They are typically known for ultra-low latency and custom operators (mostly used for real-time inference) [12].

Table 2. The Comparison of GPU, TPU and FPGA in 6 dimensions

Dimension	GPU	TPU	FPGA
Compute style	SIMT and Tensor Cores, mixed precision	Matrix multiplies units in large systolic arrays; compiler/XLA feeds batches efficiently	Custom spatial/dataflow pipelines synthesized to logic & DSPs
Scale-out fabric	NVSwitch multi-node GPU fabrics	Pods interconnect with reconfigurable high-speed links	App-specific; can network cards but not turnkey like GPU/TPU
Typical use	SOTA training, general inference	Large-scale training & high-throughput inference in GCP	Deterministic, very low-latency inference; custom operators/protocols
Dev stack	CUDA/cuDNN; PyTorch/JAX/TF ecosystems mature	TensorFlow/JAX via XLA on Cloud TPU	Vitis AI / HLS / RTL; higher bring-up effort
Flexibility	High	Medium	Very high, but design time is higher
Latency	Good	Good at scale	Excellent

4. Conclusion

FPGAs for deep neural networks still face several challenges, including a mismatch between memory and bandwidth, irregular data and control flow due to the diversity of operators, and an immature toolchain. Especially when processing extremely large models, on-chip memory cannot accommodate weight and activation data, and external memory access becomes a performance and energy efficiency bottleneck. Ultra-low bit quantization and mixed precision can improve throughput but are sensitive to accuracy and hardware control overhead. Compared to the general-purpose GPU ecosystem, FPGAs are still catching up in terms of portable cores and end-to-end automated compilation. The cost of development and prototyping iterations raises the engineering barrier. Overall, FPGAs are more suitable for low-latency inference and specific training phases but struggle to handle ultra-large-scale end-to-end training tasks.

Looking ahead, HBM and on-chip hierarchical buffering will alleviate memory access bottlenecks. Near-memory/near-computing and on-chip network optimizations are expected to further reduce energy consumption and latency. Reconfigurable overlays based on programmable arrays for tensor operations and dataflow will enable rapid reuse and migration between different model families. Quantization-aware training, structural sparsity, and compression/reduction co-design will make "hardware-aware" models the norm. Furthermore, graph-level automatic partitioning, pipelining, and timing-aware compilation based on MLIR/TVM will lower the design barrier. Heterogeneous SoCs combining FPGAs with CPUs, DSPs, and AI engines, along with chiplets and 3D packaging, will also achieve higher system efficiency in edge and industrial scenarios.

In summary, the core value of FPGAs lies in their reconfigurability and deterministic latency. In applications that prioritize low latency, customization, verification, and long lifecycles (industrial, automotive, medical, and private network edge), FPGAs can achieve excellent cost-effectiveness through co-optimization of the "model-dataflow-hardware" architecture. However, GPUs/ASICs still hold advantages in general high-throughput and large-scale model training. With the continuous evolution of storage systems, low bitrate/sparseness, automated compilation, and the advancement of heterogeneous integration technologies, FPGAs are expected to transform from "hand-optimized single-purpose network accelerators" to "platforms for rapid migration of model families and scenarios," securing a long-term and stable position in cloud-edge collaboration.The meaning of this research is to summarize the advantages and flaws of the FPGA used in the Deep Neural Networks, which can help researchers identify the features of each chip so that they can choose proper types of chips in specific fields.

References

[1] Farooq U, Marrakchi Z, Mehrez H. FPGA architectures: An overview. Tree-Based Heterogeneous FPGA Architectures: Application Specific Exploration and Optimization, 2012: 7 - 48.

[2] Tang Q, Mehrez H, Tuna M. Multi-FPGA prototyping board issue: the FPGA I/O bottleneck. 2014 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV). IEEE, 2014: 207 - 214.

[3] Bolton W. Programmable logic controllers. Newnes, 2015.

[4] Mohammed S Q. Implementation of Simplified Data Encryption Standard on FPGA using VHDL. Science, 2022, 2022: 2.

[5] Bobda C, Mbongue J M, Chow P, et al. The future of FPGA acceleration in datacenters and the cloud. ACM Transactions on Reconfigurable Technology and Systems (TRETS), 2022, 15 (3): 1 - 42.

[6] Zeng K, Ma Q, Wu J W, et al. FPGA-based accelerator for object detection: a comprehensive survey. Journal of Supercomputing, 2022, 78 (12).

[7] Wu X, Ma Y, Wang M, et al. A flexible and efficient FPGA accelerator for various large-scale and lightweight CNNs. IEEE Transactions on Circuits and Systems I: Regular Papers, 2021, 69 (3): 1185 - 1198.

[8] Tunheim S A, Jiao L, Shafik R, et al. Tsetlin machine-based image classification FPGA accelerator with on-device training. IEEE Transactions on Circuits and Systems I: Regular Papers, 2024.

[9] Hesse C N. Analysis and comparison of performance and power consumption of neural networks on CPU, GPU, TPU and FPGA. Master's thesis, University of Hildesheim, 2021.

[10] Posso J, Kieffer H, Menga N, et al. Real-Time Semantic Segmentation of Aerial Images Using an Embedded U-Net: A Comparison of CPU, GPU, and FPGA Workflows. arXiv preprint arXiv: 2503.08700, 2025.

[11] Liu X, Xu W, Wang Q, et al. Energy-efficient computing acceleration of unmanned aerial vehicles based on a cpu/fpga/npu heterogeneous system. IEEE Internet of Things Journal, 2024, 11 (16): 27126 - 27138.

[12] Hu Y, Liu Y, Liu Z. A survey on convolutional neural network accelerators: GPU, FPGA and ASIC. 2022 14th International Conference on Computer Research and Development (ICCRD). IEEE, 2022: 100 - 107.

[13] Zhang C, Yu H, Zhou Y, et al. High-performance and energy-efficient fpga-gpu-cpu heterogeneous system implementation. Advances in Parallel & Distributed Processing, and Applications: Proceedings from PDPTA'20, CSC'20, MSV'20, and GCC'20. Cham: Springer International Publishing, 2021: 477 - 492.

[14] Wang Y. Artificial-intelligence integrated circuits: Comparison of gpu fpga and asic. Applied and Computational Engineering, 2023, 4 (1): 99 - 104.

[15] Sano Y, Kobayashi R, Fujita N, et al. Performance evaluation on GPU-FPGA accelerated computing considering interconnections between accelerators. Proceedings of the 12th International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies. 2022: 10 - 16.

[16] Alshemi M, Saif S, Taher M. Hardware acceleration of lane detection algorithm: A GPU versus FPGA comparison. arXiv preprint arXiv: 2212.09460, 2022.

[17] Huynh T V. Deep neural network accelerator based on FPGA. 2017 4th NAFOSTED Conference on Information and Computer Science. IEEE, 2017: 254 - 257.

[18] Ma Y, Cao Y, Vrudhula S, et al. Optimizing the convolution operation to accelerate deep neural networks on FPGA. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 2018, 26 (7): 1354 - 1367.