EduFPGA-AI: Educational Platforms and Frameworks for Learning AI on FPGA Hardware

Mohamed Abdo
Hamm-Lippstadt University of Applied Sciences
Lippstadt, Germany
mohamed-sayed-mohamed.abdo@stud.hshl.de

***Abstract*—The intersection of artificial intelligence (AI) and field-programmable gate array (FPGA) technology has given rise to novel educational methodologies for hardware-accelerated machine learning. This paper provides a comprehensive review of educational platforms and frameworks that facilitate the implementation of AI algorithms on FPGA hardware. The architectural advantages of FPGAs for AI applications are examined, including reconfigurability, parallelism, low latency and power efficiency. The paper provides a systematic review of important deployment frameworks, such as PYNQ, Vitis AI, HLS4ML, FINN and OpenVINO, analysing their educational value and practical implementation workflows. Moreover, the discussion encompasses quantisation techniques that facilitate the effective implementation of models on FPGA platforms. These techniques have been shown to reduce model sizes by up to 75%, while maintaining or preserving inference accuracy. The survey includes recommendations for FPGA families that are suitable for educational use, ranging from edge devices such as Zynq UltraScale+ to data centre accelerators such as Alveo. This work contributes to the growing body of research on FPGA-AI education by providing educators and researchers with a structured overview of the available tools, methodologies and best practices for teaching hardware-accelerated AI in academic settings.**

***Index Terms*—Artificial Intelligence, FPGA, Hardware Acceleration, Machine Learning Education, Quantization, PYNQ, Vitis AI, Edge Computing**

I. Introduction

The rapid evolution of artificial intelligence (AI) and machine learning (ML) has created an increasing the demand for specialised hardware capabilty of executing complex neural network computations efficiently and with low latency. Furthermore the traditional processors such as central processing units (CPUs) and graphics processing units (GPUs) have dominated AI deployment, field-programmable gate arrays (FPGAs) offer unique advantages in production systems and educational environments alike too [1]. FPGAs provide hardware reconfigurability, enabling custom digital circuits to be implemented for specific AI workloads by eliminating the need for new silicon fabrication. This flexibility makes FPGAs particularly well-suited to academic AI settings, where students can experiment with various AI architectures and optimisation techniques, starting with simple models up to Data-centers FPGAs [2].

Educational platforms for FPGA-AI have emerged as critical tools for bridging the gap between theoretical machine learning concepts and practical hardware implementation [3]. These platforms reduce the barriers to entry for students and researchers by offering high-level abstractions while still providing access to low-level hardware optimisation. The importance of FPGA education in hardware-accelerated AI is growing as industry demand for engineering students with expertise in eaithr AI and hardware continues to increase. This paper surveys the current FPGA-AI educational platform landscape, providing educators and researchers with a comprehensive resource for implementing hardware-accelerated AI with FPGA.

II. FPGA Architecture for AI Applications

A. Standalone FPGA Architecture

The three most important parts of an FPGAs are these: First, there are programmable logic blocks, then programmable interconnects, and, lastly, there are hardened blocks . The PLBs are composed of thousands to millions of identical cells that are made up of basic digital components, which are LUTs, flip-flops, and logic gates. These are basically the most minimal units that are needed for any digital circuit to be built, so there is really no need to create silicon. Another important component would be the Programmable interconnects that are composed of a re-routable wiring, which links all the logic blocks together, allowing these to be interconnected any which way to allow specific pathways that are suitable for a specific algorithm, most probably an AI algorithm [1].

Contemporary FPGAs equip hardened IPs, which refer to hardwired functional blocks that enhance efficiency in tasks like mathematical calculations. They include memory, known as BRAM, that allows faster memory access, while the latest ones come with proprietary AI engines, like those available in the AMD Versal. The advantage lies in the fact that these chips are reconfigurable compared to the hardwired architecture associated with the CPU and the fixed architecture of general processors, including the GPU. It allows the hardware to stay updated with the latest AI model.

B. SoC FPGA Hybrid Architecture

System-on-Chip (SoC) FPGAs combine ARM processors with FPGA fabric on a one single chip, to creat a hybrid architecture which optimizes power efficiency and processing capabilities. In such devices, the CPU handles it's own software tasks while the FPGA accelerates computationally intensive workloads. This division of labor is particularly

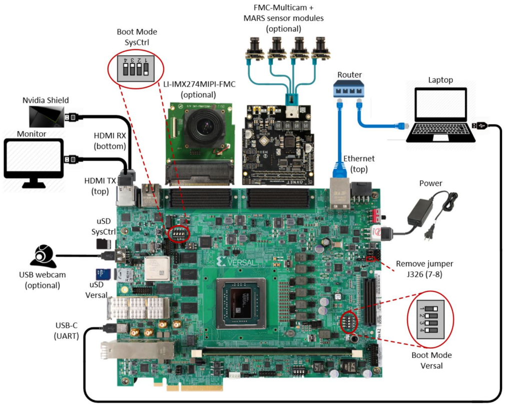

Fig. 1. AMD Versal Adaptive SoC featuring AI Engines and programmable logic for AI acceleration

effective for AI applications where certain operations benefit from hardware AI acceleration. a great example of this architecture ”The Zynq UltraScale+ MPSoC”, Which widely used in educational settings due to its balanced performance and accessibility [4].

C. High-End FPGA for Data Center Acceleration

For data center applications, high-end FPGAs such as Alveo accelerator cards provide massive computational resources. These devices feature the largest capacity FPGAs available, often equipped with high-bandwidth memory (HBM) offering 16-32GB capacity with bandwidth exceeding 460 GB/s [5]. The combination of massive parallel processing capabilities and ultra-fast memory access makes these platforms ideal for deploying large-scale AI models in cloud environments, providing valuable learning opportunities for students studying data center AI acceleration.

III. Quantization: The Mathematical Foundation

Quantization is a very important method of efficiently implementing AI models into FPGAs [6]. In this technique, floating-point parameters of 32-bit are compressed into 8-bit integer values with much less computational complexity and space requirements. Here is the mathematical equation representing this compression step is given as [7]:

$$Q(x) = \text{round}\left(\frac{x}{\Delta}\right) \times \Delta + Z \tag{1}$$

where:

- x is the floating-point value to be quantized
- Δ is the quantization step size (scale factor)
- Z is the zero-point (integer bias)
- $Q(x)$ is the quantized integer value

In the symmetric quantization implementationsyou will find that $Z = 0$, and the range is symmetric almost around zero. The scale factor, denoted by $Delta$, to be calculated as follows:

$$\Delta = \frac{\max(|x|)}{2^{b-1} - 1} \tag{2}$$

where b is the target bit-width (typically 8 bits) [7].

This quantization leads to a reduction in the size of the models by about 75% from a size of 100 MB to a maximum of 25 MB for the usual models but does not impact the accuracy of the AI model beyond acceptable limits [6]. The savings in computing power are also significant as the number of operations performed with an 8-bit integer can be performed 4 times less with access to less bandwidth using FPGA technology designed for fixed-point operations rather than the float type.

IV. Educational Frameworks and Platforms

A. Vitis AI Framework

The Vitis AI development framework offers a complete set of tools for implementing AI inference on Xilinx’s hardware solutions, including devices in the edge and datacenter PCIe cards [8]. It includes a model optimiser, an AI compiler, an AI quantiser, and an AI profiler. Thus, the students are exposed to the use of professional development tools while in academia. The Vitis AI development framework is compatible with leading neural network solutions, namely TensorFlow, PyTorch, and Caffe.

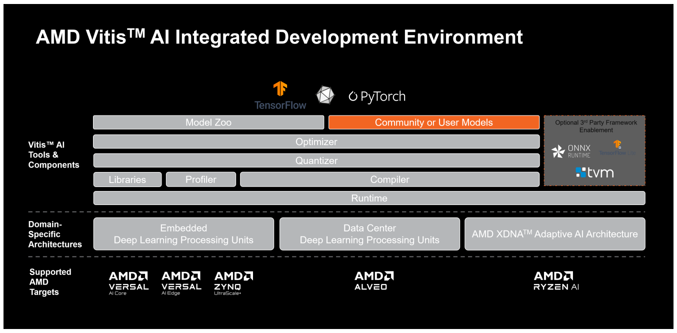
AMD Vitis™ AI Integrated Development Environment
TensorFlow
PyTorch
Model Zoo
Community or User Models
Optional 3rd Party Framework Enablement
ONNX RUNTIME
TensorFlow Lite
tvm
Vitis™ AI Tools & Components
Optimizer
Quantizer
Libraries
Profiler
Compiler
Runtime
Domain-Specific Architectures
Embedded Deep Learning Processing Units
Data Center Deep Learning Processing Units
AMD XDNA™ Adaptive AI Architecture
Supported AMD Targets
AMD VERSAL AI Core
AMD VERSAL AI Edge
AMD ZYNQ UltraScale+
AMD ALVEO
AMD RYZEN AI

Fig. 2. Vitis AI development workflow from model to deployment

Recommended Hardware: Vitis AI has been demonstrated to provide support for a wide range of AMD Xilinx platforms including Zynq UltraScale+ MPSoCs for edge applications, Alveo accelerator cards for data centres, and Versal Adaptive SoCs for AI-optimised workloads [8]. In the context of educational pursuits, the Zynq UltraScale+ ZU3EG. This may be regarded as an excellent combination of performance, cost-effectiveness and comprehensive documentation.

B. hls4ml Framework

The HLS4ML (High-Level Synthesis for Machine Learning) framework makes it easier to migrate machine learning models to FPGA-based firmwares using the technique of high-level synthesis [9]. The framework has proved to be highly capable in fields involving ultra-low-latency processing, including those involving high-energy physics data. The open-source model of this program has made it an ideal choice for use in research and educational institutions . The framework described in this paper supports training with quantization awareness and several optimization techniques. This allows university students to analyze all factors related to model resources, latency, and accuracy.

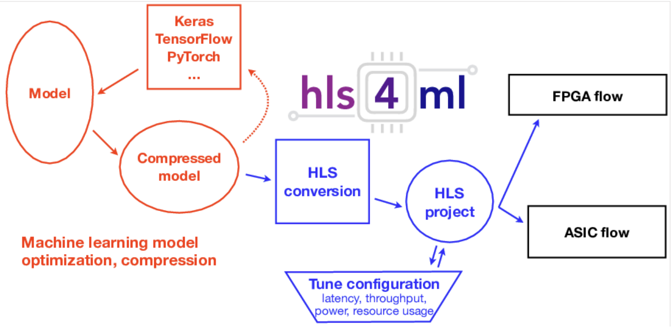

Fig. 3. hls4ml conversion pipeline from ML models to FPGA firmware

Recommended Hardware: HLS4ML is compatible with a variety of FPGA platforms; however, it is especially efficacious when utilised with the Zynq-7000 and Zynq UltraScale+ devices. The PYNQ-Z2 board is an optimal initiation into the field, offering both cost-effectiveness and comprehensive PYNQ assistance for accelerated prototyping [4]. This is achieved by uploading a bitstream file, with no need even for VHDL or Verilog.

C. PYNQ Framework and DPU Integration

The PYNQ (Python Productivity for Zynq) framework signifies a substantial enhancement in the accessibility of FPGA for the purposes of AI education [4]. PYNQ facilitates the utilisation of Python programming for Zynq devices through the medium of Jupyter notebooks. This enables students to implement AI applications without the prerequisite knowledge of hardware description language (HDL). This approach has been demonstrated to significantly reduce the learning curve, while ensuring that hardware acceleration remains available [2].

A fundamental element of PYNQ's AI capabilities is the Deep Learning Processing Unit (DPU), Xilinx's configurable AI accelerator IP that can be integrated into FPGA designs. The DPU provides:

- **Massive parallelism** for convolutional layers, pooling, batch normalization, and activation functions
- **Customizable architecture** adaptable to different AI workloads
- **Real-time performance** with 20×–100× faster inference compared to pure ARM CPU implementations
- **Lower latency and power consumption** ideal for edge AI devices

The integration with PYNQ is particularly elegant: educators can load a bitstream containing the DPU and then use pre-built Python APIs to load AI models (in .xmodel format), feed input tensors, and receive inference outputs—all without writing HDL or Vitis AI C++ code. This makes complex AI acceleration accessible to students with primarily software backgrounds [4].

Recommended Hardware: PYNQ officially supports numerous boards including PYNQ-Z1/Z2, Ultra96-V2, ZCU104, and Kria KV260 [4]. For classroom settings, the PYNQ-Z2 offers the best combination of cost, documentation, and community support. However as startup and average price for students Ultra96-V2 is very good to starts with in AI application.

D. FINN Framework

The FINN (Fast Inference on Neural Networks) framework from Xilinx Research Labs is a dataflow compiler specifically designed for quantized neural network inference on FPGAs [10]. The present framework is distinguished from traditional frameworks in that it does not map neural networks to existing hardware resources. Rather, it generates custom streaming architectures that are optimised for each model. This approach has been demonstrated to provide exceptional throughput and efficiency, particularly for binary and ternary neural networks [10].

Recommended Hardware: FINN has been developed for Xilinx UltraScale+ devices and is particularly well-suited for research-oriented courses. The Alveo U250 accelerator card provides the necessary resources for the exploration of large-scale quantized networks [5], while Zynq UltraScale+ devices, such as the ZCU102, are well-suited for edge-focused studies.

E. OpenVINO Framework

Intel's OpenVINO (Open Visual Inference and Neural Network Optimization) toolkit facilitates the implementation of artificial intelligence (AI) across a diverse range of heterogeneous platforms, encompassing central processing units (CPUs), graphics processing units (GPUs), supporting all Intel's (FPGAs). The cross-platform capabilities of the software render it a valuable resource for the instruction of comparative analysis of acceleration technologies. The 2025.4 release has continued to expand its support for (FPGAs), with a particular emphasis on Intel Agilex and Stratix devices.

Recommended Hardware: While OpenVINO supports multiple platforms, its FPGA capabilities are best demonstrated on Intel Development Kits featuring Agilex SoC FPGAs with integrated AI capabilities.

F. SensiML and RISC-V AI Ecosystem

SensiML's Picolo AI OS offers a framerok that focuses on edge AI development, providing a lightweight environment compatible with RISC-V and ARM Cortex-M processors.and its support for emerging RISC-V ecosystem makes SensiML particularly interesting for education, which represents an open-source alternative to proprietary processor architectures.

The RISC-V AI ecosystem is gaining traction as an educational platform due to its provision of comprehensive architectural transparency, enabling students to examine processor designs from RTL to software. The open-source nature of the software eliminates licensing barriers for academic institutions, thereby facilitating its utilization for hands-on learning. Furthermore, the RISC-V Vector (V) extension and other custom instruction set extensions enable the addition of custom extensions for AI-specific instructions, thereby facilitating specialized experimentation. The platform also fosters understanding of hardware-software co-design principles, while its modular architecture allows for specialized AI accelerators to be integrated alongside processor cores. As per the industry study, the growing adoption of the RISC-V open standard instruction set architecture in the domain of artificial intelligence and machine learning within embedded systems is being noticed. This is because the architecture has the qualities that make it suitable for learning environments where the optimization of AI algorithms or the design of the processor can be explored.

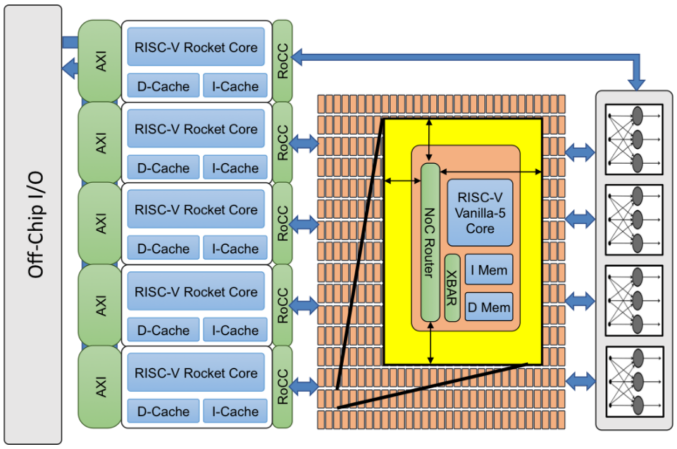

Fig. 4. RISC-V processor core with AI acceleration extensions

Recommended Hardware: The Microchip PolarFire SoC FPGA is suitable for learning with RISC-V cores. For larger AI tasks, the Xilinx Zynq UltraScale+ and Intel DE10-Nano provide more processing power and memory, enabling on-device inference and hardware–software co-design experiments.

V. Datasets and Educational Projects

The success of FPGA-AI learning is dependent on the use of appropriate datasets, in addition to project implementa- tion, which helps to demonstrate the concept of implementation. The datasets used include MNIST classification datasets for basic image classification [11], CIFAR-10 and CIFAR-100 datasets that are used in more complex image classification, such as computer vision, and TensorFlow datasets to be used in various projects. The datasets are well documented and publicly available, with appropriate scales for educational FPGA projects.

Learning projects typically begin with simple examples, like MNIST digit classification on PYNQ boards [4], and move on to increasingly complex examples, such as real-time object detection on Kria system-on-modules [12]. Other learning projects might involve the comparison of various quantization strategies via hls4ml [9] or the development of personalized neural layers in the context of FINN [10]. By defining the learning goals in terms of established datasets paired with increasingly complex learning objectives, learning paths are created that foster theoretical as well as practical mastery.

VI. Case Studies and Implementation Projects Examples for Edu FPGA-AI

1) Introduction to MNIST Digit Classification: One traditional starting project is completing an implementation for digit classification on MNIST on a PYNQ-Z2 board [4], [11]. The traditional case study here is to first train a simple CNN example written in Python via TensorFlow or PyTorch to reach around 98-99 percent accuracy on the test set. The neuronal weights are then converted to 8-bit storage as an integer, which shrinks the size to around 500-750 KB, down from 2-3 MB [7].

The deployment process also involves the transformation of the quantized model into a form compatible with DPU with the use of the Vitis AI tools [8]. The students then transfer the created .xmodel file to the PYNQ-Z2 platform and run the inference process using the Python APIs from the PYNQ platform [4].

This project helps to achieve the understanding of the entire range of software training to hardware deployment with an emphasis on the trade-offs involved in the accuracy of the model, the size of the model, and the inference time involved for the entire process of the project. Fast inference times of 2-5 ms are registered with a speedup of up to 20-50 times when compared to the inference time taken when performed using the ARM processor only [2].

A. Real-time Object Detection on Kria KV260

For an intermediate-level course, working with real-time object detection on the Kria KV260 board is an enriching learning experience regarding computer vision applications [12]. The students are able to train and deploy models such as MobileNet-SSD and YOLOv3-tiny on FPGAs. The pre-configured vision pipelines on the Kria board make it easier to implement.

The critical learning outcomes include knowledge of model optimization methods, such as channel pruning, layer merging, and scaling. The student applies various methods of quantization (post-training versus quantization-aware training) and evaluates their influence on accuracy as well as frame rate [7]. The project delivers 15 to 30 frames per second at 640x480 resolution, with a moderate level of accuracy (mAP = 0.6 to 0.7 on COCO classes), establishing the usefulness of FPGA acceleration for real-time processing.

1) Ultra96-V2 Board Overview: Edge AI applications benefit significantly from FPGA acceleration. In a case study on audio classification using the Ultra96-V2 board and the Google Speech Commands dataset [**?**], students implement keyword spotting with lightweight networks such as depth-wise separable or temporal convolutions. The project focuses on power efficiency, comparing FPGA performance against microcontrollers and GPUs. Experimental results show FPGA inference energy of 1–2 mJ—10–20 times lower than software implementations (10–30 mJ). This exercise illustrates hardware-aware algorithm design and teaches power profiling and optimization for edge environments [13].

B. Research Project: Accelerating GAN Inference on Alveo

For higher-level courses, such as graduate-level courses, the idea is to demonstrate the capability to carry out accelerated inference using Generative Adversarial Networks in Alveo cards [5]. The objective here is to optimize the throughput of models such as StyleGAN and CycleGAN, and so on. This project teaches concepts of pipeline parallelism, memory hierarchy optimization, and the use of HBM. Students learn how to split the graph into multiple compute units. Finally, performance analysis considers the amount of images per second, power efficiency of images per Joule, and comparison to the performance of a GPU. The results indicate a 2-3 fold improvement in images per second per watt over high-end GPUs for particular GAN networks.

VII. Hardware Selection Guide for Educators

Choosing suitable hardware platforms for FPGA-AI can be a trade-off among technical chops, learning goals, or resource constraints [2]. Various platforms are suitable for different aspects of learning. A table I is created for mapping the platforms with the goals and aspects of the FPGA-AI learning.

TABLE I
Hardware Platform Selection Guide for FPGA-AI Education

Platform	Educational Use Cases and Strengths
PYNQ-Z2	• **Ideal for:** Introductory courses, first exposure to FPGA-AI concepts • **Strengths:** Beginner-friendly, extensive documentation, Python-based workflow • **Features:** Large user community, abundant pre-built examples, good for basic prototyping
Ultra96-V2	• **Ideal for:** IoT-focused courses, edge AI projects, portable applications • **Strengths:** Compact form factor with wireless connectivity • **Features:** Mobile and edge computing demonstrations, power-efficient design
Kria KV260	• **Ideal for:** Computer vision curricula, robotics applications • **Strengths:** Vision-optimized with pre-built applications, system-on-module design • **Features:** Embedded vision systems, ready-to-use vision pipelines
ZCU104	• **Ideal for:** Advanced undergraduate courses, complex model deployment • **Strengths:** High-performance capabilities, professional development features • **Features:** Supports complex models, industry-standard interfaces
Alveo U50	• **Ideal for:** Graduate research, data center AI studies • **Strengths:** High-bandwidth memory (HBM), cloud integration capabilities • **Features:** Data center acceleration, high-performance computing applications

Such a course on foundational FPGA-AI concepts can start most easily with the Python-centric approach of the PYNQ-Z2, supported by a strong community [4]. Then, for intermediate courses focused on particular specialized applications, students may move to platforms such as the Ultra96-V2 for edge computing, or the Kria KV260 for computer vision [12]. Finally, advanced curricula requiring professional grade capabilities may utilize ZCU104 boards; research-oriented programs will make use of Alveo accelerator cards to explore data center AI [5]. In this tiered way, educators may introduce hardware complexity in steps that maintain pedagogical clarity.

VIII. Development Education FPGA-AI Guidelines

Designing effective curricula on Educational AI using an FPGA-based platform involves pre-planning of prerequisites, learning objectives, and available resources [2], [3]. A suggested flow would be a final-year progression starting with foundational knowledge, including digital logic, basic neural network concepts, and Python programming to ensure students have a wider engagement in both hardware and AI software. The next of the sequence will involve modules on FPGA architecture, HDLs or HLS tools, and on quantization and model optimization for embedded AI [6].

The lab sessions should be more practical, scaffolded from guided examples on any of the PYNQ, PolarFire SoC FPGA, or Ultra96-V2 [4]. In this regard, the students will be able to independently modify state-of-the-art trained neural network architectures for FPGA deployment, quantization, and experimentation with optimized inference. Some advanced projects may include developing a model end-to-end, deploying and evaluating the performance for some real-world AI application. The assessment strategy needs to combine theoretical understanding with practical implementation based on code reviews, project demonstrations, and reports on metrics analysis like inference latency, accuracy, and power efficiency.

It is necessary that any Educational AI program be integrated with already existing courses in computer science, electrical engineering, and AI to guarantee its sustainability. FPGA-AI topics will complement studies in computer architecture, embedded systems, machine learning, and parallel computing [1]. Cross-disciplinary projects involving software-hardware interaction in the design of an AI system would allow students to acquire a holistic understanding of edge intelligence and prepare them for their research or application in an educational or embedded AI context.

IX. Challenges and Solutions in EduFPGA-AI

The initiation of efficient EduFPGA-AI systems faces various challenges concerning technology, pedagogy, and other factors [2]. The mentioned challenges should be addressed in order to design learning systems and concepts that enable successful learning of AI by using FPGA hardware.

A. Technical Challenges

Technical difficulties in the EduFPGA-AI curriculum include the availability of hardware, the high learning barrier, or the complexity of FPGA tools. Expensive FPGA boards exist, though expensive ones such as Alveo or Versal can be overcome by the usage of cloud FPGA platforms like AWS

F1 [5]. High learning barriers can begin with the learning of high-level frameworks such as PYNQ, which enable Python programming with easier hardware handling [4], after which Vitis AI or hls4ml can be introduced [8], [9]. Additionally, complex FPGA toolchains can be avoided through the usage of Docker environments, scripts, or learning guides which enable the student to concentrate on programming AI.

B. Pedagogical Challenges

EduFPGA-AI needs interdisciplinary learning, which integrates topics in AI, computer architecture, and embedded systems [3]. Courses should be designed in modular fashion with independent topics with well-sequenced prerequisites to enable interdisciplinary learning. Conventional written assessments cannot be used to determine FPGA-AI dexterity, while portfolio assessments with measures such as inference delay, power efficiency, and code reviews are more informative on student learning achieved. The fast-paced evolution of algorithms in AI and FPGA technology can be overcome through emphasis on fundamental principles, courses designed in modular fashion, and industry partnerships [1].

C. Institutional and Resource Challenges

Expertise of faculty, lab facilities, and integration into curricula are major challenges in institutions [2]. Professional development opportunities in areas like team teaching and industry expert guest lectures can fill faculty expertise gaps.Lab facilities may be incrementally expanded with cloud FPGA services in addition to other possible means like grant acquisition or industry collaborations. FPGA-AI course content may be delivered in the form of special electives, integrals to other courses, and capstone projects to offer a variety of entry points while coping with resource limitations.

X. Conclusion and Future Directions

This Paper has reviewed the current trends in educational frameworks aimed at FPGA-based hardware education of AI. The distinctive traits of the FPGA architecture of reconfiguration capabilities, high concurrency, low latency, low power consumption—make these chips the most appropriate environment to educate students about machine learning acceleration. PYNQ platform tools like Vitis AI, the hls4ml framework, and FINN libraries keep making education much simpler and discard the need for students to understand the intricacies of deep learning hardware architecture. Future trends will be aimed at a closer interlink between cloud development tools and the education of AI-accelerating hardware. More focus on the newest forms of deep learning models and closer scrutiny of the ethics of AI-based hardware systems will also shape the future of AI education on hardware systems. The inclusion of the latest developments involving the availability of the RISC-V intellectual properties within the architecture of the FPGA chips will provide the advantage of the convergence of the latest developments of the hardware designing of the chips and the education of AI acceleration on these chips.

XI. Declaration of Originality

I hereby declare that I, Mohamed Abdo, have authored this paper independently, using only the cited sources for reference. All quoted material has been properly indicated, and all references have been fully acknowledged. This work has not been previously submitted for any examination or published in any form, in whole or in part.

17/12/2025 Lippstadt Mohamed Abdo

References

[1] A. Shawahna, S. M. Sait, and A. El-Maleh, "FPGA-Based Accelerators of Deep Learning Networks: A Review," *IEEE Access*, 2019, survey of FPGA DL architectures and optimization techniques. [Online]. Available: https://ieeexplore.ieee.org/document/8594633

[2] S. Afifi, H. GholamHosseini, and R. Sinha, "Introducing FPGA-based Machine Learning on the Edge to Undergraduate Students," 2020, structured curriculum for teaching edge ML with HLS, AXI bus, and hands-on projects. [Online]. Available: https://ieeexplore.ieee.org/document/9274007

[3] J. Hackett, A. Mishra, T. Krishna *et al.*, "Open-Source Educational Platform for FPGA Accelerated AI in Robotics," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 8939–8944, a Framework for designing neural networks targeting edge computing platforms. [Online]. Available: https://ieeexplore.ieee.org/document/9734102

[4] Xilinx Inc., "PYNQ: Python Productivity for Zynq," 2024, python framework with Jupyter notebooks for rapid AI prototyping on Zynq FPGAs. Accessed: Month Day, Year. [Online]. Available: http://www.pynq.io/

[5] AMD Xilinx, "Alveo Data Center Accelerator Cards," 2025, high-end FPGAs with 16-32GB HBM and 460+ GB/s bandwidth. [Online]. Available: https://www.xilinx.com/products/boards-and-kits/alveo.html

[6] S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks," achieves 35-49× compression via pruning, quantization, and Huffman coding. [Online]. Available: https://arxiv.org/abs/1510.00149

[7] B. Jacob *et al.*, "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, quantization-aware training for 8-bit integer inference with minimal accuracy loss. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2018/html/Jacob_Quantization_and_Training_CVPR_2018_paper.html

[8] AMD Xilinx, "Vitis AI Development Stack," 2024, complete AI toolkit with quantizer, compiler, and DPU support for TensorFlow/PyTorch. Accessed: Month Day, Year. [Online]. Available: https://github.com/Xilinx/Vitis-AI

[9] J. Duarte *et al.*, "Fast inference of deep neural networks in FPGAs for particle physics," *Journal of Instrumentation*, vol. 13, p. P07027, 2018, [OPEN ACCESS] Original hls4ml paper on ultra-low-latency ML inference via HLS. [Online]. Available: https://iopscience.iop.org/article/10.1088/1748-0221/13/07/P07027

[10] Y. Umuroglu, N. J. Fraser, M. Blott *et al.*, "FINN: A Framework for Fast, Scalable Binarized Neural Network Inference," in *Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, dataflow compiler for quantized/binary NN on FPGAs. [Online]. Available: https://dl.acm.org/doi/10.1145/3020078.3021744

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based Learning Applied to Document Recognition," 1998, mNIST dataset with 70,000 handwritten digits for classification tasks. [Online]. Available: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

[12] AMD Xilinx, "Kria KV260 Vision AI Starter Kit," 2024, [FREE DOCS] Vision-optimized SOM with pre-built AI applications and Ubuntu support. Accessed: Month Day, Year. [Online]. Available: https://www.xilinx.com/products/som/kria/kv260-vision-starter-kit.html

[13] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017, depthwise separable convolutions for efficient embedded deployment. [Online]. Available: https://arxiv.org/abs/1704.04861