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 integration of Artificial Intelligence (AI) with Field-Programmable Gate Array (FPGA) hardware represents a frontier in high-performance and low-power computing. However, the significant complexity associated with FPGA design creates a substantial barrier to entry for students and newcomers. This paper explores the emerging domain of *EduFPGA-AI*, which encompasses educational platforms and frameworks specifically designed to facilitate learning AI concepts and their implementation on FPGAs. We provide a background on FPGAs in AI deployment and survey popular, accessible educational platforms that abstract away the traditional Hardware Description Language (HDL) complexity. A detailed workflow example incorporating quantization techniques, from a high-level model to FPGA deployment, is presented to illustrate the educational process. The paper further discusses the pedagogical benefits of hands-on FPGA-AI learning, identifies persistent challenges such as toolchain maturity and quantization trade-offs, and outlines future directions for making FPGA-AI education more accessible and effective.**

***Index Terms*—FPGA, AI Education, Machine Learning, Hardware Acceleration, Educational Tools, High-Level Synthesis, Deep Learning, Quantization.**

I. Introduction

The relentless growth of Artificial Intelligence (AI), particularly in deep learning, has created an insatiable demand for computational power. While Graphics Processing Units (GPUs) currently dominate the training and inference phases, their power consumption and generic architecture are not always optimal. Field-Programmable Gate Arrays (FPGAs) offer a compelling alternative, providing the ability to create custom, application-specific hardware architectures that can achieve high performance and exceptional energy efficiency for targeted AI workloads [1].

Despite their potential, FPGAs have traditionally been inaccessible to a broad audience. The design flow typically requires expertise in Hardware Description Languages (HDLs) like VHDL or Verilog, knowledge of digital logic design, and familiarity with complex commercial toolchains. This steep learning curve is a significant impediment for students and software-centric AI practitioners who wish to explore hardware acceleration.

To bridge this gap, a new educational paradigm is emerging: *EduFPGA-AI*. This concept refers to a collection of platforms, frameworks, and methodologies whose primary goal is to demystify the process of implementing AI algorithms on FPGA hardware for learning purposes. These tools often leverage High-Level Synthesis (HLS) and advanced quantization techniques, allowing designers to describe hardware functionality using high-level programming languages like C++ or Python, thereby abstracting away the underlying HDL complexity.

This paper investigates the landscape of EduFPGA-AI. Section II reviews related work in open-source FPGA-AI tools. Section III provides necessary background on FPGAs, AI deployment, and quantization methods. Section IV surveys popular educational platforms and their quantization support. Section V walks through a concrete educational workflow with quantization experiments. Section VI and VII discuss the benefits and challenges, respectively. Finally, Section VIII and IX outline future directions and conclude the paper.

II. Related Work

The development of tools to simplify FPGA programming for AI is an active research area. Several open-source projects, while not exclusively educational, have become foundational for EduFPGA-AI due to their accessibility and free availability.

FINN [2]: The *Framework for Interactive Neural Networks* from Xilinx Research is a pivotal work for quantized neural network inference on FPGAs. FINN's significance for education lies in its end-to-end flow, which takes a trained, pre-quantized neural network from a framework like PyTorch and generates a customized streaming dataflow architecture. This allows students to explore the relationship between network precision (e.g., binary or ternary weights) and hardware resource utilization, a key concept in efficient AI deployment. Its open-source nature makes it an excellent platform for advanced projects.

hls4ml [3]: The *High-Level Synthesis for Machine Learning* project was developed within the high-energy physics community to enable fast inference for particle detectors. hls4ml translates models from TensorFlow or PyTorch into HLS code (C++). For educators, hls4ml provides a transparent bridge between a familiar software environment and hardware generation, with extensive support for quantization experiments. Students can compile small models for low-cost FPGA boards, making it an ideal tool for laboratory exercises that demonstrate the principles of latency-optimized, custom AI hardware.

PYNQ [4]: The *Python Productivity for Zynq* framework is perhaps the most directly educational of the three. PYNQ is an open-source project from AMD that allows developers

to use Python for programming Xilinx Zynq devices, which combine ARM processors with FPGA fabric. Students can interact with the FPGA using Jupyter notebooks, overlaying pre-compiled hardware libraries (bitstreams) and controlling them via Python APIs. This dramatically lowers the barrier to entry, enabling students to focus on system integration and the application of accelerated AI functions without needing to master the entire toolchain from scratch.

These related works form the technological backbone upon which dedicated EduFPGA-AI curricula can be built, each contributing a different level of abstraction and educational focus, particularly in the realm of quantization techniques.

III. Background: FPGA and AI Deployment

A. FPGA Architecture and Advantages

An FPGA is a semiconductor device containing a matrix of configurable logic blocks (CLBs), programmable interconnects, and specialized hardware like Digital Signal Processing (DSP) slices and Block RAM (BRAM). Unlike Application-Specific Integrated Circuits (ASICs), FPGAs can be reprogrammed after manufacturing, offering a unique blend of flexibility and performance.

For AI inference, FPGAs offer several key advantages:

- **Parallelism:** FPGAs can be configured to create deeply parallel architectures tailored to the specific dataflow of a neural network, often surpassing the efficiency of fixed-instruction-set architectures like GPUs for certain models.
- **Low Latency:** The custom hardware pipelines eliminate operating system overhead and enable deterministic, sub-microsecond inference, which is critical for real-time control systems.
- **Energy Efficiency:** By implementing only the necessary operations and minimizing data movement, FPGA-based accelerators can achieve high performance per watt.

B. Neural Network Computational Foundations

The mathematical operations underlying neural networks are particularly well-suited for FPGA implementation. The fundamental operation in a fully-connected layer can be expressed as:

$$\mathbf{y} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

where $\mathbf{W}$ represents the weight matrix, $\mathbf{x}$ the input vector, $\mathbf{b}$ the bias vector, and σ the activation function. For convolutional layers, the operation becomes:

$$\mathbf{Y}(i,j,k) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\sum_{c=0}^{C-1} \mathbf{W}(m,n,c,k)\cdot\mathbf{X}(i+m,j+n,c)+\mathbf{b}(k) \tag{2}$$

where $\mathbf{W}$ is the convolution kernel, $\mathbf{X}$ the input feature map, and $\mathbf{Y}$ the output feature map. These operations exhibit massive parallelism that can be exploited through custom hardware architectures on FPGAs.

C. The Traditional FPGA Design Flow

The conventional path to implementing a design on an FPGA involves using an HDL to describe the register-transfer level (RTL) logic. This RTL description is then synthesized, mapped, placed, and routed using vendor-specific software (e.g., AMD Vivado or Intel Quartus) to generate a bitstream file that configures the FPGA. This process is notoriously time-consuming and requires specialized knowledge, making it a poor fit for rapid prototyping in AI education.

D. The Role of High-Level Synthesis (HLS)

HLS has been a game-changer. It allows designers to describe the behavior and functionality of hardware using C, C++, or SystemC. The HLS tool then automatically compiles this high-level code into an RTL description. For AI, this means that complex operations like matrix multiplication or convolution can be described algorithmically, and the HLS tool can be guided using *pragmas* to create parallel and pipelined hardware implementations. This abstraction is the cornerstone that makes EduFPGA-AI feasible.

E. Quantization Methods for Efficient FPGA AI

A critical technique enabling efficient AI deployment on FPGAs is quantization–the process of reducing the numerical precision of weights and activations in neural networks. While floating-point (FP32) representations offer high precision, they consume substantial hardware resources. Quantization to lower-precision fixed-point or integer representations enables more efficient FPGA implementations through reduced memory bandwidth, lower power consumption, and increased parallelism [2].

The quantization process can be mathematically formulated as:

$$Q(x) = \Delta \cdot \text{round}\left(\frac{x}{\Delta}\right) \tag{3}$$

where Δ represents the quantization step size, typically determined by the dynamic range of the tensor being quantized. For uniform symmetric quantization, the step size is calculated as:

$$\Delta = \frac{2 \cdot \max(|x|)}{2^b - 1} \tag{4}$$

where b is the target bit-width. Several approaches have emerged:

Post-Training Quantization (PTQ) converts pre-trained FP32 models to lower precision (typically INT8) with minimal accuracy loss through calibration with representative datasets. This approach, used by frameworks like Vitis AI [5], requires no retraining but may suffer from accuracy degradation in sensitive networks.

Quantization-Aware Training (QAT) incorporates fake quantization operations during the training process, allowing the model to learn parameters robust to precision reduction. This method typically yields better accuracy than PTQ but requires access to the training pipeline and computational resources for retraining.

Extreme Quantization, as implemented in the FINN framework [2], pushes precision to 1-2 bits (binary or ternary

networks). While offering maximum hardware efficiency and eliminating external memory dependencies, this approach requires specialized network architectures and training techniques.

The choice of quantization strategy represents a fundamental trade-off in the EduFPGA-AI context: PTQ offers simplicity for educational workflows, QAT provides better accuracy for practical applications, and extreme quantization enables research into novel efficient architectures.

IV. Popular Educational Platforms and Frameworks

Educational platforms for FPGA-AI have emerged as critical enablers for bridging the gap between theoretical machine learning concepts and practical hardware implementation. These frameworks leverage high-level synthesis and abstraction methodologies to make FPGA technology accessible to students and researchers without extensive hardware design backgrounds. The landscape comprises both commercial offerings with academic support and open-source research frameworks, each offering distinct pedagogical advantages for different educational objectives.

A. AMD PYNQ and Vitis AI Ecosystem

The PYNQ (Python Productivity for Zynq) framework represents a significant advancement in educational accessibility for FPGA-based AI development [4]. By providing a Python-based development environment integrated with Jupyter notebooks, PYNQ enables students to interact with programmable logic using familiar software paradigms. The platform supports various Zynq-based development boards that combine ARM processing systems with programmable logic, creating an ideal environment for embedded AI applications.

Complementing PYNQ, the Vitis AI development environment offers a comprehensive toolchain for deploying quantized neural networks on AMD platforms [5]. This integrated ecosystem allows students to transition seamlessly from model development in frameworks like TensorFlow or PyTorch to hardware deployment, providing practical experience with production-ready development workflows while abstracting the underlying hardware complexity. The framework's support for automated quantization and optimization pipelines enables students to focus on system-level design considerations rather than implementation details.

B. Intel OpenVINO Toolkit for Cross-Platform Deployment

Intel's OpenVINO toolkit provides another accessible pathway for FPGA-AI education, particularly for institutions utilizing Intel hardware platforms [6]. The toolkit's unified API supports deployment across heterogeneous computing environments, enabling comparative studies of AI inference performance across CPUs, GPUs, and FPGAs. This approach facilitates educational exercises in hardware-aware model optimization and performance benchmarking, teaching students to evaluate trade-offs between different acceleration platforms.

The abstraction layer allows focus on system-level integration concepts rather than low-level implementation details, making it suitable for courses emphasizing embedded systems and edge computing applications. The toolkit's model optimization capabilities and hardware-specific optimization plug-ins provide practical examples of how commercial frameworks address the challenges of cross-platform AI deployment.

C. Lattice sensAI for Low-Power Applications

For educational scenarios requiring ultra-low-power implementations, the Lattice sensAI stack offers specialized capabilities for energy-constrained applications [7]. Targeting Lattice's low-power FPGA families, this framework demonstrates practical implementations of always-on AI applications where power consumption represents a primary design constraint. The platform supports complete development workflows from model training to deployment, providing students with experience in power-efficient AI system design.

This focus on energy-aware implementation complements the performance-oriented approaches of other platforms, offering comprehensive coverage of design constraints in edge AI applications. The framework's emphasis on minimalist architectures and efficient resource utilization provides valuable lessons in optimization strategies for severely constrained environments.

D. Open-Source Research Frameworks

The open-source ecosystem contributes significantly to FPGA-AI education through frameworks like hls4ml and FINN, which emphasize transparency and customization. The hls4ml framework, developed initially for high-energy physics applications, provides direct insight into the translation of machine learning models into hardware descriptions [3]. Its open architecture allows students to examine and modify the complete toolchain, fostering deeper understanding of quantization techniques and hardware optimization strategies.

Similarly, the FINN framework explores extreme quantization approaches through binary and ternary neural networks, demonstrating the limits of neural network compression and specialized hardware architectures [2]. These research-oriented frameworks provide valuable educational resources for advanced courses and student research projects investigating novel AI acceleration techniques. The accessibility of their source code enables detailed examination of implementation methodologies that would be opaque in commercial toolchains.

E. Quantization Methodologies Across Platforms

Each platform implements distinct approaches to quantization and optimization that provide complementary educational perspectives. The commercial toolchains demonstrate production-oriented optimization strategies with automated quantization pipelines, while open-source frameworks offer explicit control over precision parameters and optimization techniques. This diversity enables educators to design learning experiences that address both practical engineering constraints and fundamental architectural principles.

The availability of multiple platforms also facilitates comparative studies that help students understand the relationship between design choices, implementation constraints, and application requirements in FPGA-based AI systems. Students can investigate how different quantization schemes affect both model accuracy and hardware efficiency across various platform implementations.

F. Integration in Academic Curricula

The integration of these platforms into academic curricula is supported by widespread availability of development boards through university programs and the accessibility of open-source toolchains. This ecosystem development has been crucial for enabling hands-on learning experiences that bridge theoretical concepts and practical implementation. The continued evolution of these platforms, driven by both commercial investment and research community development, ensures that FPGA-AI education remains aligned with both industrial practices and research frontiers in efficient AI computing.

Educational institutions can leverage these resources to create progressive learning pathways that introduce fundamental concepts through high-level abstractions before advancing to detailed implementation studies using open-source frameworks.

V. Workflow Example: From Model to FPGA Deployment for Education

This section outlines a simplified educational workflow implementing a quantized neural network for digit classification on the MNIST dataset using **hls4ml** and a **PYNQ-Z2** board. The workflow demonstrates how mathematical operations are translated into hardware implementations while maintaining educational accessibility.

1) **Model Training and Quantization:** Students first train a small neural network using Keras/TensorFlow. Critical to the educational process is experimenting with quantization–students apply post-training quantization to reduce precision from FP32 to INT8 or INT4. This hands-on experience demonstrates the accuracy-efficiency tradeoff fundamental to edge AI. The quantization process follows the mathematical formulation:

$$Q(w) = \text{clamp}\left(\left\lfloor \frac{w}{s} \right\rceil, -2^{b-1}, 2^{b-1} - 1\right) \qquad (5)$$

where s is the scaling factor and b is the target bit-width.
2) **Precision Configuration with hls4ml:** Using hls4ml's Python API, students configure the quantization parameters for each layer type. They explore how different precision settings (e.g., 16-bit for first layer, 8-bit for hidden layers, 4-bit for final layer) affect both accuracy and hardware resource utilization.
3) **Hardware-Aware Optimization:** Students adjust the *reuse factor* parameter, which controls the parallelism-resource tradeoff. Lower reuse factors increase parallelism but consume more FPGA resources, providing practical insight into hardware constraints. The quantization choices directly impact viable reuse factor settings.
4) **Model Conversion and HLS Generation:** The quantized model is translated into optimized HLS code. Students can inspect the generated C++ to see how quantization manifests in hardware–reduced bit-width operations, smaller buffers, and simplified arithmetic units.
5) **Deployment and Comparative Analysis:** After generating and loading the bitstream, students execute the quantized model and compare its performance and accuracy against the original floating-point version. This concrete demonstration reinforces why quantization is essential for FPGA-based AI.

This workflow encapsulates the core EduFPGA-AI process: starting from a software-defined AI model, navigating hardware-oriented optimization and quantization steps, and culminating in a functional, accelerated application on real hardware.

VI. Benefits of EduFPGA-AI Platforms

The adoption of EduFPGA-AI platforms in curricula offers multifaceted benefits:

- **Democratization of Hardware Knowledge:** They lower the barrier to entry, allowing computer science and AI students with little hardware background to engage with FPGA technology.
- **Deepened Conceptual Understanding:** Hands-on experience with quantization, parallelism, and resource constraints moves students beyond abstract AI theory. They learn firsthand how algorithmic choices directly impact hardware efficiency.
- **Bridging the Software-Hardware Divide:** These platforms foster a systems-thinking mindset, encouraging students to consider the entire stack from algorithm to silicon, a highly valuable skill in the industry.
- **Stimulation of Innovation:** By making FPGA-AI accessible, educators can empower students to prototype novel accelerator architectures and explore research ideas that would otherwise be infeasible.

VII. Challenges and Limitations

Despite the progress, several challenges remain for widespread adoption in education:

- **Toolchain Complexity and Instability:** Even with abstractions, the underlying vendor tools (Vivado, Quartus) are complex, can have long runtimes, and may present installation and licensing hurdles in an academic IT environment.
- **Resource Constraints of Educational Hardware:** Low-cost boards have limited logic elements, DSP slices, and memory, restricting the size and complexity of models that can be implemented. This often forces the use of overly simplified networks.
- **Abstraction Leakage:** When things go wrong, debugging can be very difficult. Students may encounter cryptic

HLS synthesis warnings or timing closure failures, requiring them to delve into the lower-level RTL or architectural details, which can be frustrating.

- **Quantization-Related Challenges:** Students must grapple with the accuracy-efficiency tradeoffs inherent in precision reduction. Debugging quantization-related issues, such as numerical overflow or significant accuracy drops, requires understanding both the neural network behavior and hardware limitations.
- **Curriculum Development Effort:** Creating effective laboratory exercises and course material that balance conceptual depth with practical feasibility requires significant time and expertise from educators.

VIII. Future Directions

The field of EduFPGA-AI is poised for growth. Future directions should focus on:

- **Standardized and Cloud-Based Labs:** Developing preconfigured, containerized toolchains and offering cloud-based access to FPGA hardware can eliminate local installation issues and provide equitable access to all students.
- **Enhanced Debugging and Visualization Tools:** Creating educational tools that visually illustrate the mapping of the AI model onto the FPGA fabric and provide intuitive debugging at the algorithm level would greatly enhance the learning experience.
- **Advanced Quantization Pedagogical Tools:** Developing interactive visualization tools that show how different quantization schemes affect both model accuracy and hardware resource usage in real-time.
- **Integration with Broader Embedded AI Curricula:** EduFPGA-AI should be positioned as a component within larger courses on embedded systems and edge AI, comparing and contrasting FPGAs with microcontrollers, GPUs, and other AI accelerators.
- **Focus on End-to-End Edge Systems:** Future platforms could emphasize full system integration, including sensor input processing and actuator control, showcasing FPGAs as the core of a complete intelligent edge device.

IX. Conclusion

The emergence of EduFPGA-AI platforms represents a critical step towards nurturing a new generation of engineers who are fluent in both AI algorithms and efficient hardware implementation. By leveraging HLS, quantization techniques, open-source frameworks like FINN, hls4ml, and PYNQ, and low-cost hardware, educators can now provide hands-on experience that was previously out of reach. While challenges related to toolchain complexity, resource limitations, and quantization trade-offs persist, the pedagogical benefits are profound. These platforms demystify hardware acceleration, bridge the software-hardware gap, and empower students to explore the intricate trade-offs at the heart of efficient AI computing. As these tools mature and become more integrated into academic curricula, they will play a vital role in preparing the workforce for the demands of the intelligent edge computing era.

X. Declaration of Originality

I am Mohamed Abdo, herewith declare that I have composed the present paper and work by myself and without the use of any other than the cited sources and aids. Sentences or parts of sentences quoted literally are marked as such; other references with regard to the statement and scope are indicated by full details of the publications concerned. The paper and work in the same or similar form have not been submitted to any examination body and have not been published. This paper was not yet, even in part, used in another examination or as a course performance. I agree that my work may be checked by a plagiarism checker

---

03/04/2025 Lippstadt Mohamed Abdo

References

[1] E. Nurvitadhi *et al.*, "Can fpgas beat gpus in accelerating next-generation deep neural networks?" *IEEE Micro*, vol. 37, no. 4, pp. 18–27, 2017.

[2] M. Blott, T. B. Preußer, N. J. Fraser, G. Gambardella, K. O'brien, Y. Umuroglu, M. Leeser, and K. Vissers, "Finn-r: An end-to-end deep-learning framework for fast exploration of quantized neural networks," *IEEE Micro*, vol. 38, no. 3, pp. 60–65, 2018. [Online]. Available: https://arxiv.org/abs/1809.04570

[3] J. Duarte *et al.*, "Fast inference of deep neural networks in fpgas for particle physics," *Journal of Instrumentation*, vol. 13, no. 07, p. P07027, 2018. [Online]. Available: https://arxiv.org/abs/1804.06913

[4] T. Courtney, J. Anderson, and S. Chin, "Pynq: A productive system for agile embedded system development," in *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE, 2018, pp. 269–269. [Online]. Available: https://ieeexplore.ieee.org/document/8344604

[5] AMD Corporation, "Vitis ai development environment," https://www.xilinx.com/products/design-tools/vitis/vitis-ai.html, 2023, accessed: 2024.

[6] Intel Corporation, "Intel distribution of openvino toolkit," https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html, 2023, accessed: 2024.

[7] Lattice Semiconductor, "Lattice sensai stack," https://www.latticesemi.com/sensai, 2023, accessed: 2024.