FPGA-Accelerated Fast Machine Learning for Heterogeneous Edge Systems

Mohammed Mshragi
School of Engineering
Cardiff University, UK
MshragiM@cardiff.ac.uk

Ioan Petri
School of Engineering
Cardiff University, UK
petrii@cardiff.ac.uk

Omer Rana
School of Computer Science and Informatics
Cardiff University, UK
ranaof@cardiff.ac.uk

***Abstract*—The integration of Fast Machine Learning (FastML) algorithms with edge devices for real-time building management system (BMS) poses challenges due to resource constraints and latency requirements. Addressing these challenges necessitates not only the quantization and optimization of ML models to achieve rapid inference but also their adaptation to fit within the limited resources of edge devices, thereby reducing computational overhead while maintaining predictive accuracy. These advancements are critical for enabling key functionalities for applications related to energy management, HVAC control, and fault detection in BMS applications. This study proposes an end-to-end edge-based framework utilizing **hls4ml** for the deployment of machine learning models on FPGA platforms, designed to process real-time building sensor data streams efficiently. By employing dynamic quantization and pruning techniques, the framework ensures the optimal use of FPGA resources, achieving low-latency inference without compromising model performance. The results underscore the potential of FPGA-accelerated ML systems in meeting the demands of real-time BMS applications, offering enhanced energy efficiency, operational reliability, and scalability. This work provides valuable insights into the evolving landscape of edge computing for smart building applications and highlights the broader implications for FPGA-based ML deployments in resource-constrained environments.**

***Index Terms*—FPGA, FastML, machine learning, building management systems, edge computing, acceleration, energy;**

I. Introduction

The integration of Fast Machine Learning with Building Management Systems (BMS) can accelerate decarbonisation in the built environment and improve the demand response strategies in buildings. The possibility of developing Edge supported BMS capabilities using machine learning (ML) and edge computing can facilitate the development of a (near)real-time energy optimization capability for buildings but deploying these technologies in practice using resource-constrained environments remains a significant challenge [1]. Traditional building energy management systems (BEMS) have relied heavily on rule-based control strategies, effective for simpler structures but inadequate for the complexity of modern buildings. Cloud-based solutions can process data from smart meters and IoT sensors but introduce challenges such as latency, data security concerns, high transfer costs, and operational expenses [2], [3]. These limitations are particularly problematic for real-time applications, where delays can hinder critical decision-making during demand response scenarios. Recent intelligent BMS solutions rely on ML models executed on centralized CPUs or GPUs, which achieve high prediction accuracy but can increase power consumption and introduce latency that impedes rapid responses during critical demand response (DR) scenarios.

Field Programmable Gate Arrays (FPGAs) offer a compelling alternative, excelling in power efficiency and cost-effectiveness compared to CPUs and GPUs for ML workloads. Research indicates that FPGAs can deliver comparable or superior performance while consuming less power [4]–[6]. However, deploying complex ML models on FPGAs typically requires extensive manual coding in Hardware Description Languages (HDLs).

Despite the potential of FPGA-based solutions, prior studies have attempted to address key inefficiencies. For instance, [7] implemented Model Predictive Control (MPC) using artificial neural networks (ANNs) for heating control but faced several obstacles, including high computational demands, difficulties in achieving real-time performance, and reliance on representative training datasets, which can compromise model accuracy. The absence of adaptive learning mechanisms and the challenges of deploying neural networks on FPGA hardware further complicate real-world implementation, limiting scalability.

Unlike prior hls4ml applications in physics or computer vision, this work pioneers the use of hls4ml for real-time energy prediction in building management systems, achieving 0.0022 s inference latency and 4.54×10^5 inferences/s by integrating quantized and pruned MLPs with edge-based preprocessing on a Raspberry Pi + PYNQ platform.

To address these challenges, this research introduces an FPGA-based FastML framework that leverages High-Level Synthesis for Machine Learning (hls4ml) to streamline ML deployment, achieving ultra-low-latency inference and efficient resource utilization for real-time energy management in buildings. To our knowledge, no prior studies have utilized hls4ml for BMS applications, despite the growing interest in FPGA-based solutions. Our approach significantly reduces execution time compared to traditional methods and outperforms CPU and GPU-based solutions in terms of latency and throughput. Analysis of FPGA-processed benchmark datasets demonstrates reduced execution time, enabling real-time decision-making and energy management in BMSs.

The contributions of this research are as follows:

1) *Comprehensive FPGA-Based ML Framework*: A practical pipeline for deploying ML algorithms on FPGA devices tailored to the real-time needs of building energy management.
2) *Acceleration Using hls4ml*: Utilization of the hls4ml framework to transform ML models for efficient FPGA deployment, ensuring accuracy and resource efficiency.
3) *Comparative Hardware Analysis*: Detailed comparisons of ML performance across CPUs, GPUs, and FPGAs, focusing on inference speed, resource utilization, and prediction accuracy.
4) *Real-World Validation*: Implementation in a multi-building facility demonstrating fast, accurate, and energy-efficient predictions tailored for energy management across various zones.

This research provides valuable insights into building automation and energy optimization, enabling real-time interventions that enhance building performance management. Such advancements in smart building technologies are vital for scenarios requiring rapid decision-making to ensure timely controlling and actuation in buildings.

The remainder of the paper is organized as follows: Section 2 presents the state-of-the-art in the field of study by reviewing recent advancements in edge computing, machine learning acceleration, and their applications in BMSs, particularly in HVAC control and energy optimization. Section 3 introduces the proposed FastML-BMS approach, detailing the methodology, model architecture, and optimization techniques for FPGA deployment. Section 4 provides a comprehensive evaluation of the approach, including predictive performance, computational efficiency, and resource utilization across FPGA, GPU, and CPU platforms. Section 5 discusses the implications of the findings, trade-offs between accuracy and efficiency, and potential applications in real-time energy management in buildings. Finally, Section 6 concludes the paper by summarizing the key contributions and outlining future research directions.

II. Related Work

Recent advancements in edge computing and Fast ML have brought transformative changes to BMSs, especially in HVAC control and energy optimization. These technologies have redefined real-time building operations, addressing crucial challenges in energy efficiency and system control with remarkable precision and adaptability.

A. ML Applications in Building Management

Incorporating ML techniques into HVAC systems has proven particularly effective. [9] developed a hybrid predictive control system that combines the Single-Step Prediction Response Coefficient (SPRC) method with Convolutional Neural Networks (CNNs), achieving exceptional predictive accuracy with a mean absolute error of 0.27°C and a root mean square error of 0.24°C, while reducing HVAC runtime by approximately 19%. In a similar vein, [10] demonstrated how deep learning models, such as CNNs and Long Short-Term Memory (LSTM) networks, enhance energy management by delivering precise predictions within modern BMS frameworks.

Reinforcement learning (RL) has also emerged as a key enabler of HVAC optimization. The Phasic Policy Gradient (PPG) algorithm, for example, achieves energy reductions of 2–14% and accelerates convergence rates by 66% [11]. Further advancing RL applications, [12] introduced the MB2C framework, a model-based RL method for multi-zone HVAC control, delivering 8.23% greater energy savings while requiring significantly less training data.

B. FPGA-Based Acceleration for ML

FPGA-based solutions have demonstrated remarkable potential in enabling fast, energy-efficient operations for ML inference. A recent comprehensive analysis by [13] of 287 FPGA conference studies revealed that 81% target inference acceleration, predominantly focusing on CNNs, with increasing attention to Graph Neural Networks (GNNs). Compared to GPUs, FPGAs consistently demonstrate superior energy efficiency in inference tasks [14].

Particularly noteworthy advances have been made in accelerating Transformer architectures. [15] designed an FPGA-based accelerator for Transformers' Multi-Head Attention (MHA) and Feed-Forward Network (FFN) layers, achieving a 14.6× speedup over Nvidia V100 GPUs. Building on this, [16] proposed the SSR architecture utilizing the VCK190 FPGA, achieving 36× faster performance and 21× better energy efficiency than GPUs while maintaining optimal latency-throughput balance.

For large language models (LLMs), [17] introduced the modular DFX architecture, demonstrating a 3.8× throughput and 4× energy efficiency improvement using a multi-FPGA setup. Similarly, [18] presented the NPE architecture for language model acceleration, achieving 4× and 6× better energy efficiency compared to CPUs and GPUs, respectively.

C. Edge Computing Integration

Edge computing further enhances these advancements by enabling decentralized data processing for real-time decision-making. [19] demonstrated an open-source IoT edge-computing system that leverages existing infrastructure for secure and efficient monitoring of energy metrics. [20] applied Proximal Policy Optimization with Clipping (PPO-Clip) for HVAC energy optimization, achieving significant reductions in power consumption while maintaining occupant comfort.

While hardware innovations have boosted BMS efficiency through various implementations, such as [21] hardware acceleration in smart building access control and [22] FPGA-based Big Bang-Big Crunch (BB-BC) algorithm, these approaches do not specifically address the Fast ML implementation required for BMS applications. Traditional ML models, especially deep learning algorithms, struggle to process high-frequency sensor data and control signals quickly enough for effective building utility management [23]. While cloud-based ML/AI services are computationally efficient, they introduce substantial communication overhead, latency, and operational costs—factors that are critical for real-time systems.

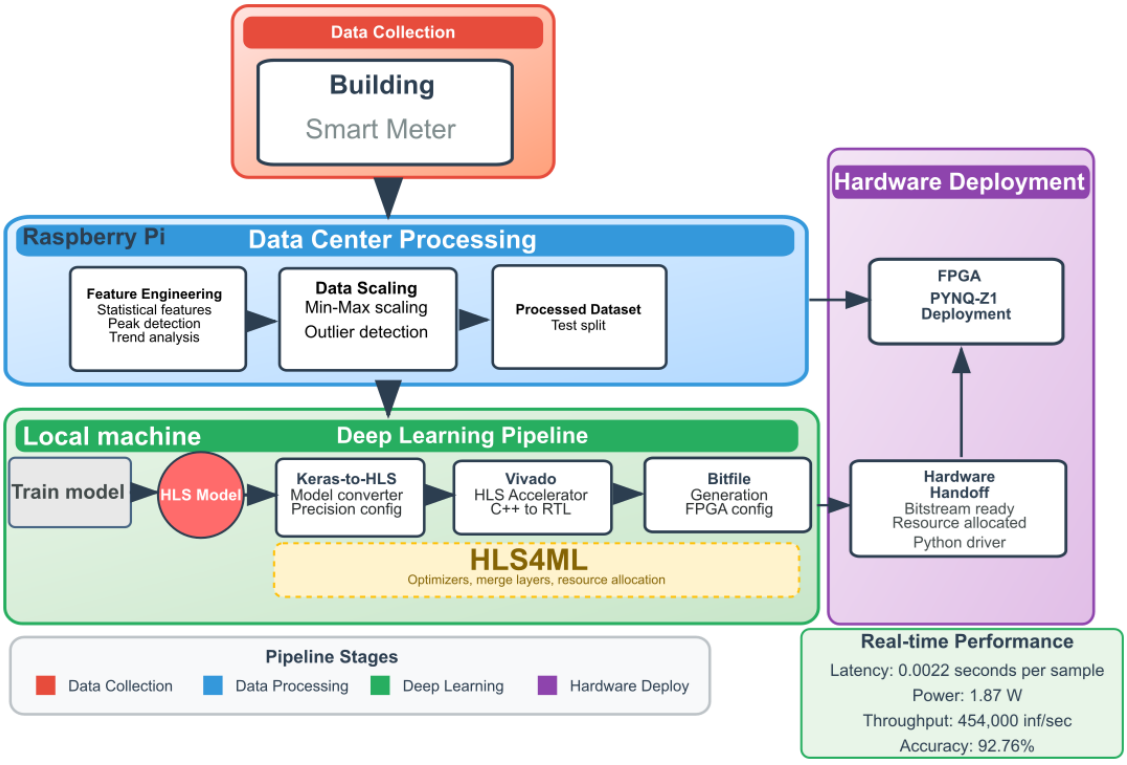

Fig. 1. End-to-end pipeline for real-time energy prediction, from smart meter data collection and preprocessing on a Raspberry Pi to MLP quantization, HLS conversion with hls4ml, and deployment on PYNQ-Z1 FPGA for fast, low-latency inference. Arrows indicate data flow between components.

III. Proposed FastML-BMS Approach

This research demonstrates the application of FastML and hls4ml for building management systems in an energy management application. The proposed FastML-BMS approach enhances operational efficiency and reduces energy consumption through a structured workflow, organized into three key stages: (1) model training and optimization on a local machine, (2) real-time data processing on a Raspberry Pi, and (3) energy forecasting model conversion and deployment on an FPGA. Fig. 1 illustrates the architecture of the proposed testbed, integrating smart meter data collection, preprocessing on a Raspberry Pi, and FPGA-accelerated machine learning predictions.

A. MLP Energy Prediction Model

The energy prediction system was developed using a comprehensive machine learning workflow implemented on a local server environment. The prediction model uses historical energy meter reading dataset comprising 32,900 entries collected over four years for an educational building (i.e. Queens Building). This dataset was used to train a Multilayer Perceptron (MLP) model for energy consumption forecasting.

After preprocessing, the dataset comprised 32,415 observations. For model input, we selected 13 features: three temporal indicators (season, day, hour) and ten lagged energy consumption values (n_{lag}). The model leverages this data across different zones within the Queen's Building case study, including key areas such as the Trevithick Building and Central Building. By aggregating energy consumption from these zones, we created a comprehensive dataset that enhances prediction accuracy through the integration of temporal features.

As illustrated in Figure 2, the MLP architecture employs a sequential design implemented using the Keras deep learning framework. The network begins with an input layer configured to accept the 13 selected features. This is followed by a hidden layer with 64 neurons and ReLU activation functions, along with a dropout layer set at 20% to prevent overfitting. A secondary hidden layer contains 32 neurons, also utilizing ReLU activation, followed by another dropout layer for consistent regularization. The final layer is a single-neuron output layer, optimized for regression tasks related to energy prediction.

The model was compiled using the Adam optimizer and Mean Squared Error (MSE) as the loss function, both standard choices for regression problems. The trained model was saved for future optimization and potential deployment on FPGA using hls4ml.

B. MLP Optimization

Following the creation of the MLP model, we incorporated QKeras layers for quantization-aware training, ensuring compatibility with edge devices like FPGAs. After training, the model was optimized and converted into FPGA-compatible code using hls4ml, facilitating seamless deployment for real-time energy forecasting applications. An educational building facility with meter data and BMS capabilities has been used

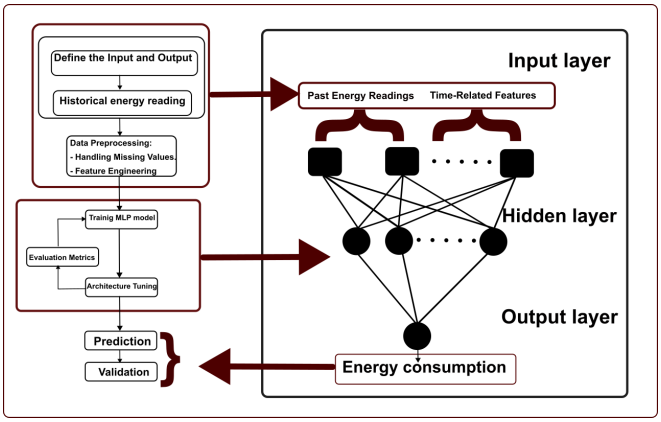

Fig. 2. Flowchart illustrating the MLP architecture for energy prediction, detailing input, processing, and output stages.

as a case study for which an energy forecasting model was developed, converted, and deployed in an FPGA-RPi hybrid computational environment.

Following the training and optimization of the MLP, we employed hls4ml to convert the model into a format suitable for FPGA deployment, which involves three key stages: quantization, code generation, and integration with FPGA tools. First, the floating-point parameters of the neural network (weights and biases) are quantized into fixed-point representations to reduce memory requirements and enhance inference speed on FPGA hardware. hls4ml then generates optimized C/C++ code that describes the neural network architecture, including layer configurations and data flow, which is subsequently synthesized into hardware description language (HDL) to preserve the original model's performance characteristics. This process reduces model complexity while preserving accuracy, enabling efficient FPGA deployment. The generated code is compiled and synthesized using FPGA development tools such as Vivado HLS; our implementation utilized the hls4ml Vivado Accelerator backend, seamlessly integrating with the PYNQ software stack to abstract programmable logic circuits into hardware libraries (overlays) that are accessible through a Python API, improving FPGA accessibility for machine learning applications. The hardware synthesis environment was configured using Vivado HLS version 2020.1, targeting the Xilinx Zynq®-7000 SoC FPGA, with key configuration details including fixed-point representations for fully connected layers and activation functions, biases configured as `fixed<6,1>` and weights as `fixed<16,6>`, and a reuse factor set to 1 to maximize resource efficiency while preserving performance. Lastly, the hls4ml framework supports configuration through the Python API or YAML configuration files; our implementation specifically used the Python API with parameters such as target part `xc7z020clg400-1`, board `pynq-z1`, backend `VivadoAccelerator`, interface `axi_stream`, and fixed-point precision for input/output.

C. FPGA and Raspberry Pi testing

The final step involved deploying the quantized and optimized MLP inference model onto the FPGA. Leveraging the hls4ml backend, which offers multiple backend options including Vivado and Vitis, we seamlessly integrated the model into the FPGA hardware using the PYNQ framework. This deployment requires three essential files: the bitfile, hardware handoff, and Python driver for the FastMLP energy prediction model.

This approach abstracts programmable logic circuits into hardware libraries (overlays) that are accessible through a Python API, enhancing FPGA accessibility for machine learning applications. The conversion process from the quantized model to the HLS model using hls4ml and the Vivado Accelerator backend enables efficient deployment of neural networks on FPGA platforms. A comprehensive evaluation of the FPGA -Raspberry Pi performance, including latency analysis and preprocessing techniques is provided in the next section.

IV. Evaluation and Results

This section evaluates the FastML approach, focusing on the optimization and performance of the Multilayer Perceptron (MLP) model on FPGA hardware. Predictive accuracy is assessed using key metrics such as R^2, MSE, root mean squared error (RMSE), and mean absolute error (MAE). Computational efficiency is compared across FPGA, GPU, and CPU platforms, emphasizing inference speed and latency.

A. Model Development and Optimization

The MLP model was designed to manage energy consumption across multiple zones, incorporating temporal features like seasonal indicators and lagged variables for enhanced prediction accuracy. Fig. 3 illustrates the model's performance, comparing observed past values, true future values, and predictions, demonstrating effective generalization to unseen data.

Following the baseline evaluation, we optimized the MLP for hls4ml and FPGA deployment. The architecture consists of three fully connected layers, with initial optimization involving 6-bit quantization of weights and biases. The first two hidden layers use 6-bit precision, while the output layer is quantized for regression tasks, reducing computational complexity while maintaining accuracy.

Further optimization included pruning 75% of less significant weights, increasing model sparsity, and applying strip pruning to eliminate redundant connections. This resulted in a compact model, significantly reducing inference and disk sizes, suitable for resource-constrained edge devices. Fig. 4 shows the weight distributions before and after quantization, illustrating the effectiveness of this approach. Additionally, Fig. 5 compares the inference size and disk size for the baseline and quantized models, highlighting the improvements achieved through optimization.

The predictive accuracy of the FPGA-accelerated model was evaluated against the baseline using key metrics. As shown in Table I, the FPGA model achieves a R^2 score of 92.76%, slightly lower than the baseline's 97%. Despite a mean squared

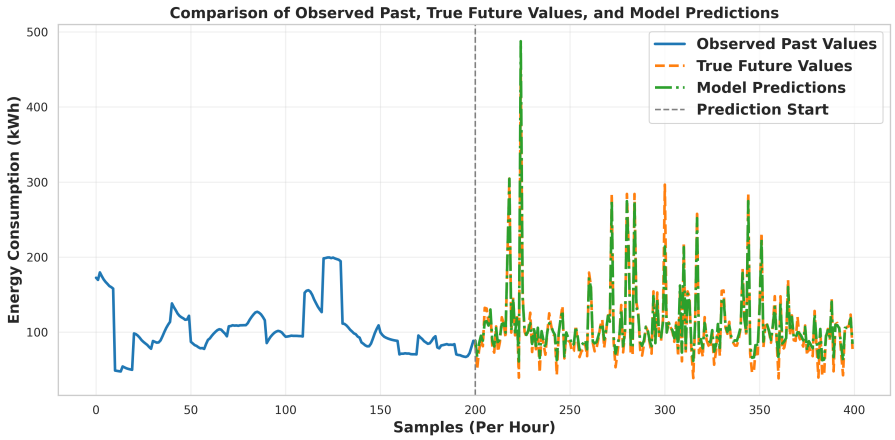

Fig. 3. Comparison of the first 200 samples of observed past values (blue), true future values (orange), and MLP model predictions (green). The vertical dashed line indicates the start of the prediction period.

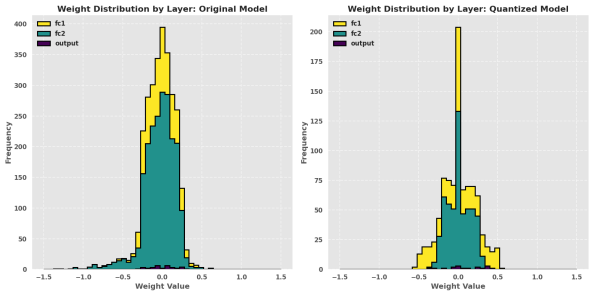

Fig. 4. Weight distributions before and after quantization.

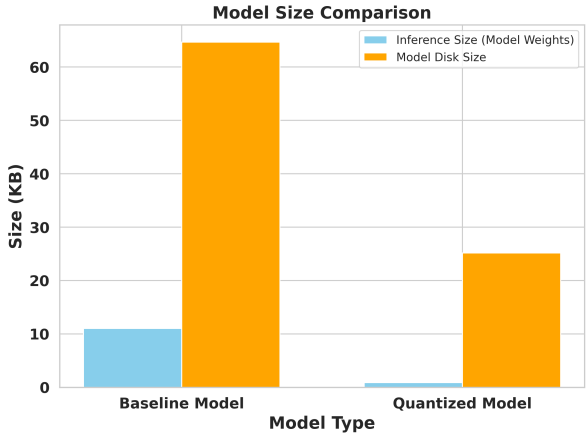

Fig. 5. Comparison of inference size and model disk size for baseline and quantized models.

error (MSE) of 305.8 compared to the baseline's 87.94, the FPGA model remains practically useful, with RMSE and MAE values of 17.48 and 13.4, respectively.

Fig. 6 compares the first 100 predictions from the FPGA, HLS, and quantized models against true values. Both the FPGA and HLS models achieve an R^2 score of 92.76%, while the quantized model scores 94.26%. Although FPGA predictions show slightly larger deviations, they effectively capture overall trends, confirming their suitability for deployment in hardware-efficient scenarios.

TABLE I
PERFORMANCE METRICS COMPARISON

Metric	Baseline Model	FPGA Model
R^2 Score	0.97	0.92
MSE	87.94	305.8
RMSE	9.37	17.48
MAE	5.6	13.4
95% CI	[101.39, 104.22]	[105.67, 108.78]

B. Inference Speed, latency and timing

The FPGA implementation significantly outperforms GPU and CPU platforms in latency and throughput Fig. 7. Processing 6,481 samples in 0.01427 seconds (454k inferences/second), the FPGA achieves 42× faster throughput than the CPU (10.5k inferences/second) and 27× faster than the GPU (16.7k inferences/second).

Table II highlights the FPGA's superior consistency, with a standard deviation of 0.68 ms versus 26.73 ms (CPU) and 4.22 ms (GPU). This reliability stems from hardware parallelism and fixed-pipeline execution.

TABLE II
LATENCY STATISTICS FOR PYNQ, CPU, AND GPU INFERENCE

Metric	FPGA	CPU	GPU
Entries	6481	6481	6481
Mean (ms)	8.36	64.42	22.98
StdDev (ms)	0.68	26.73	4.22

The FPGA design achieved a 4.367 ns clock period (229 MHz), surpassing the 5 ns target (200 MHz) Table III. This corresponds to a 12.66% timing improvement, ensuring robustness against voltage and temperature variations.

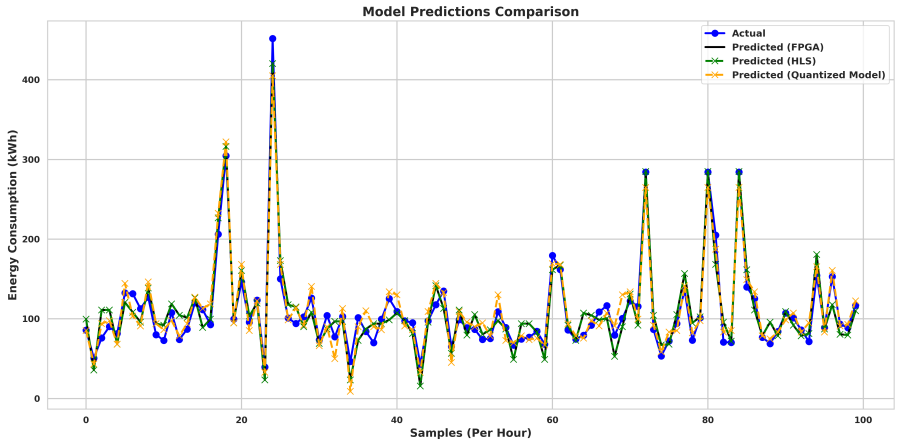

Fig. 6. Comparison of true values and predictions for the first 100 samples: blue for actual values, black for FPGA predictions, green for HLS predictions, and orange for the quantized model.

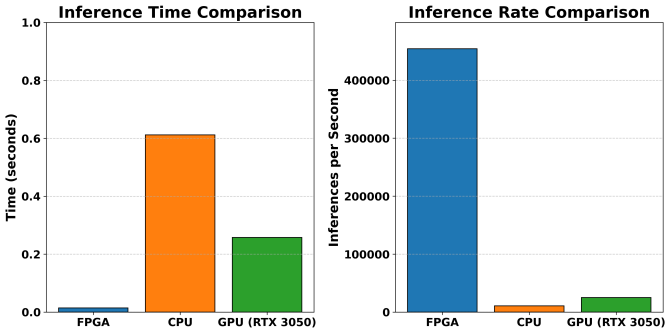

Fig. 7. Performance comparison across FPGA, GPU, and CPU platforms. The left plot presents inference latency (seconds), while the right plot illustrates inference throughput (inferences per second), demonstrating the FPGA's superior performance.

The target clock period of 5 ns was set considering FPGA capabilities and model complexity. However, post-synthesis analysis showed an achieved 4.367 ns clock period, exceeding the target due to hls4ml optimizations, such as pipelining, dataflow, and parallelization. These techniques minimized the critical path delay, allowing faster execution while maintaining timing constraints. The 0.633 ns margin ensures stability under variations.

The timing improvement is calculated as:

$$\text{Improvement (\%)} = \frac{T_{\text{target}} - T_{\text{achieved}}}{T_{\text{target}}} \times 100, \qquad (1)$$

where $T_{\text{target}} = 5.00$ ns and $T_{\text{achieved}} = 4.367$ ns. Substituting these values yields a 12.66% improvement, further enhancing resilience against voltage and temperature variations.

TABLE III
CLOCK TIMING CHARACTERISTICS

Parameter	Clock	Target	Achieved	Uncertainty
Timing	`ap_clk`	5.00 ns	4.367 ns	0.62 ns

Three key *optimization strategies* contributed to this improvement:

- *Dataflow*: Overlapping tasks to reduce latency and enhance resource utilization.
- *Pipelining*: Partitioning tasks into sequential stages, improving throughput.
- *Parallelization*: Executing multiple iterations concurrently, reducing loop trip counts but increasing resource usage.

These optimizations resulted in a well-structured design with a 4.367 ns clock period, ensuring high performance for real-time applications.

C. Real-Time Processing via Raspberry Pi

To evaluate the system's real-time capabilities, we integrated the Raspberry Pi for data collection, preprocessing, and transfer to the FPGA, ensuring efficient handling of real-time energy meter data with low-latency communication.

The Raspberry Pi collects real-time energy meter readings and preprocesses the data for inference. Preprocessing techniques include imputation for missing values, outlier detection filtering values beyond three standard deviations, and smoothing using the Exponential Moving Average (EMA):

$$S_t = \alpha X_t + (1-\alpha) S_{t-1}, \quad \alpha = \frac{2}{n+1}, \qquad (2)$$

where S_t is the smoothed value, X_t is the raw value, and α is the smoothing factor. Normalization scales input features, and temporal aggregation computes hourly and seasonal averages to identify patterns in energy consumption.

The processed data is transmitted to the FPGA via Universal Asynchronous Receiver/Transmitter (UART) over USB, with batch processing optimizing transfer efficiency. The average latency is 304.96 ms, stabilizing beyond 8000 samples, suitable for real-time applications like dynamic HVAC control. Fig. 8 illustrates this latency during data transfer.

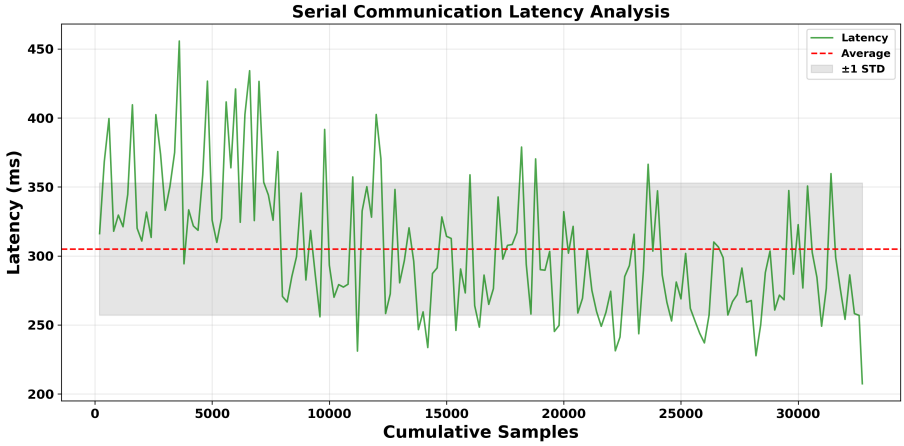

Fig. 8. Latency in milliseconds during real-time data transfer from the Raspberry Pi to the PYNQ Z1 via USB.

D. FPGA Resource Optimization via Quantization and Pruning

Quantization reduced DSP usage by 58% and LUTs by 41% compared to the baseline Fig. 9. This enables deployment on low-cost FPGAs (e.g., Zynq-7020) while maintaining 92% of the baseline accuracy.

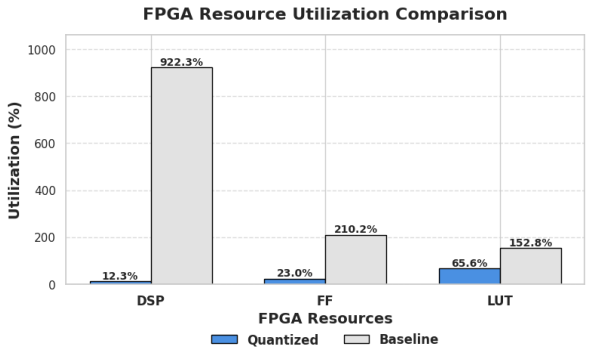

Fig. 9. Comparison of FPGA resource utilization (DSP, FF, LUT) between Quantized and Baseline Models.

We also evaluated the MLP model using the `qkeras` library with 4-bit, 6-bit, and 8-bit quantization, combined with structured weight pruning (75% sparsity), and deployed the resulting models on FPGA using `hls4ml`. We report the R^2 score and the corresponding FPGA resource utilization, including LUTs, DSPs, and FFs.

As shown in Table IV, increasing the quantization bit-width improves model accuracy but also raises resource consumption. The 8-bit model achieves the highest R^2 score (0.9611) at the cost of increased LUT (80%) and DSP (55%) usage. The 6-bit model offers a good balance between accuracy (0.9276) and efficiency. The 4-bit model consumes the least hardware resources but with lower prediction accuracy (R^2 = 0.8127). These results highlight the trade-off between model accuracy and hardware footprint under combined quantization and pruning constraints.

TABLE IV
SENSITIVITY ANALYSIS OF QUANTIZATION AND PRUNING

Quantization	R^2 Score	LUTs (%)	DSPs (%)	FFs (%)
4-bit	0.8127	53	0	15
6-bit	0.9276	65	12	22
8-bit	0.9611	80	55	32

V. DISCUSSION

This study investigates the implementation of the FastML solutions for Building Management Systems (BMS) within a case study application at Queen's Building, integrating data collection, preprocessing, hls4ml conversion, and FPGA-accelerated inference. The architecture demonstrates the synergy between software-based model development and FPGA-accelerated hardware deployment, enhancing computational performance and energy efficiency in real-time applications.

Transitioning from a locally trained machine learning model to an FPGA-accelerated implementation via hls4ml achieved two critical objectives: (1) preservation of predictive performance, with a 92% R^2 score, and (2) significant computational efficiency gains, delivering 4.54×10^5 inferences per second—a $42.7\times$ speedup over CPU and $27.2\times$ over GPU architectures. This throughput-power balance addresses edge-computing constraints in BMSs, where low-latency inference is essential for dynamic HVAC control and fault detection.

A key insight from this study is the trade-off between accuracy and efficiency of the machine learning models. Although the FPGA model's R^2 score is 5.4% lower than the baseline Table I, this decrease can be attributed to quantization effects. While quantization increases the MSE by $3.5\times$, it results in a significant $42.7\times$ improvement in inference speed. This speed gain justifies the trade-off for latency-critical applications, particularly in scenarios requiring sub-20 ms response times. The FPGA's high throughput enables real-time performance, which is crucial for effective energy management in smart buildings, making the trade-off acceptable in this context.

The findings emphasize several advancements for BMSs. First, the FPGA implementation outperforms CPU and GPU counterparts in high-speed inference, positioning FPGAs as frontrunners for real-time applications. Second, low power consumption aligns with sustainability goals, ensuring efficient computation without excessive energy use. Third, hls4ml facilitates FPGA-compatible code conversion for localized decision-making. Finally, FPGA systems are inherently reconfigurable, enhancing their long-term viability. Moreover, While this study did not measure direct HVAC energy savings, the FPGA's 1.87 W power consumption and 0.0022 s prediction latency enable frequent, low-latency energy predictions, laying the groundwork for adaptive BMS control to reduce energy use in future deployments. The framework's hls4ml-based pipeline is also adaptable to domains like autonomous vehicles, where [27] achieved 4.9 ms latency for semantic segmentation on a Xilinx ZCU102 FPGA, demonstrating versatility for latency-critical applications.

Future research will explore advanced quantization techniques, such as weight pruning and fixed-point optimization, to minimize accuracy degradation. The FPGA-based approach can be extended to HVAC control, occupancy monitoring, and renewable energy integration, with a focus on adaptive models that dynamically adjust to real-time conditions.

VI. Conclusion

This paper presents an edge-based FastML model for BMS applications, utilizing FPGA acceleration and hls4ml for energy management in buildings. The successful deployment of a deep learning model on FPGA demonstrates the feasibility of real-time, low-latency predictions, offering significant advantages in speed and computational efficiency compared to traditional CPU-based systems.

The FPGA achieves a throughput of 4.54×10^5 inferences per second, which is $42.7\times$ faster than the CPU and $27.2\times$ faster than the GPU. These results highlight the potential of FPGA-accelerated machine learning—enabled by `hls4ml`—to enhance BMS operations by improving energy efficiency, operational responsiveness, and adaptability. It is worth noting that the hls4ml framework supports a limited range of architectures (e.g., MLPs, CNNs, and RNNs), which precludes the introduction of entirely new algorithms. Instead, our methodological contributions focus on model compression, edge preprocessing, and platform-specific deployment strategies tailored for real-time BMS energy prediction.

Looking ahead, the integration of AI-augmented BMS represents a significant advancement in smart building technologies. Future work will explore extending fast machine learning models to optimize HVAC systems and other infrastructure components, incorporating algorithmic innovations using HLS and Vivado for hardware-aware tuning. This could lead to adaptive and predictive control strategies that dynamically adjust to real-time environmental conditions and occupant needs—ultimately improving building sustainability, reducing operational costs, and enhancing urban resilience.

References

[1] S. A. Nabavi, A. Aslani, M. A. Zaidan, M. Zandi, S. Mohammadi, and N. Hossein Motlagh, "Machine learning modeling for energy consumption of residential and commercial sectors," Energies, vol. 13, no. 19, pp. 5171, 2020.

[2] S. I. Khan, C. Kaur, M. S. Al Ansari, I. Muda, R. F. C. Borda, and B. K. Bala, "Implementation of cloud-based IoT technology in manufacturing industry for smart control of manufacturing process," International Journal on Interactive Design and Manufacturing (IJIDeM), pp. 1–13, 2023.

[3] A. A. Laghari, K. Wu, R. A. Laghari, M. Ali, and A. A. Khan, "A review and state of art of Internet of Things (IoT)," Archives of Computational Methods in Engineering, pp. 1–19, 2021.

[4] C. Murphy and Y. Fu, "Xilinx all programmable devices: A superior platform for compute-intensive systems," in *Xilinx White Paper*, 2017.

[5] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, and G. Boudoukh, "Can FPGAs beat GPUs in accelerating next-generation deep neural networks?," in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2017, pp. 5-14.

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

[7] A. Agouzoul, E. Simeu, and M. Tabaa, "Synthesis of model predictive control based on neural network for energy consumption enhancement in building," in *AEU-International Journal of Electronics and Communications*, vol. 173, Elsevier, 2024, p. 155021.

[8] FastML Team, "fastmachinelearning/hls4ml," 2024. [Online]. Available: *https://github.com/fastmachinelearning/hls4ml*. [Accessed: Date]. doi: 10.5281/zenodo.1201549.

[9] G. Liu, J. Gao, Z. Han, and Y. Yuan, "Hybrid model-based predictive HVAC control through fast prediction of transient indoor temperature fields," in *Building and Environment*, vol. 267, 2025, p. 112253.

[10] A. Kristian, T. S. Goh, A. Ramadan, A. Erica, and S. V. Sihotang, "Application of AI in optimizing energy and resource management: Effectiveness of deep learning models," in *International Transactions on Artificial Intelligence*, vol. 2, no. 2, 2024, pp. 99–105.

[11] A. T. Nguyen, D. H. Pham, B. L. Oo, M. Santamouris, Y. Ahn, and B. T. Lim, "Modeling building HVAC control strategies using a deep reinforcement learning approach," in *Energy and Buildings*, vol. 310, 2024, p. 114065.

[12] X. Ding, A. Cerpa, and W. Du, "Multi-zone HVAC control with model-based deep reinforcement learning," in *IEEE Transactions on Automation Science and Engineering*, 2024.

[13] F. Yan, A. Koch, and O. Sinnen, "Accelerating machine learning algorithms using FPGA-based hardware accelerators: A comprehensive survey of 287 papers," in *Proceedings of Top FPGA Conferences*, 2022.

[14] A. Boutros, E. Nurvitadhi, R. Ma, S. Gribok, Z. Zhao, J. C. Hoe, V. Betz, and M. Langhammer, "Beyond peak performance: Comparing the real performance of AI-optimized FPGAs and GPUs," in *2020 International Conference on Field-Programmable Technology (ICFPT)*, IEEE, 2020, pp. 10–19.

[15] S. Lu, M. Wang, S. Liang, J. Lin, and Z. Wang, "Hardware accelerator for multi-head attention and position-wise feed-forward in the transformer," in *2020 IEEE 33rd International System-on-Chip Conference (SOCC)*, 2020, pp. 84–89.

[16] J. Zhuang, Z. Yang, S. Ji, H. Huang, A. K. Jones, J. Hu, Y. Shi, and P. Zhou, "SSR: Spatial sequential hybrid architecture for latency-throughput tradeoff in transformer acceleration," in *Proceedings of the 2024 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ACM, 2024.

[17] S. Hong, S. Moon, J. Kim, S. Lee, M. Kim, D. Lee, and J.-Y. Kim, "DFX: A low-latency multi-FPGA appliance for accelerating transformer-based text generation," in *Proceedings of IEEE International Conference on FPGA Design*, 2022.

[18] H. Khan, A. Khan, Z. Khan, L. B. Huang, K. Wang, and L. He, "NPE: An FPGA-based overlay processor for natural language processing," in *Proceedings of the 2021 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, ACM, 2021, pp. 227.

[19] D. A. V. Romero, E. V. Laureano, R. O. J. Betancourt, and E. N. Álvarez, "An open-source IoT edge-computing system for monitoring energy consumption in buildings," in *Results in Engineering*, vol. 21, 2024, p. 101875.

[20] A. Bereketeab, L. Yuan, X. Zhou, and Y. Wang, "HVAC energy optimization using Proximal Policy Optimization with Clipping (PPO-Clip)," in *IEEE Transactions on Automation Science and Engineering*, 2024.

[21] T. Sakthi Ram, L. Yogesh, S. Vetriashwath, G. Nishanth, and O. G. Swathika, "FPGA-based smart building access control," in *Smart Grids as Cyber Physical Systems: Artificial Intelligence, Cybersecurity, and Clean Energy for Next Generation Smart Grids*, vol. 1, 2024, pp. 137–144.

[22] A. Almabrok, M. Psarakis, and A. Dounis, "Fast tuning of the PID controller in an HVAC system using the big bang–big crunch algorithm and FPGA technology," in *Algorithms*, vol. 11, no. 10, 2018, p. 146.

[23] Petri, Ioan, Ioan Chirila, Heitor Murilo Gomes, Albert Bifet, and Omer F. Rana. "Resource-aware edge-based stream analytics." IEEE Internet Computing 26, no. 4 (2022): 79-88.

[24] R. Chen, T. Cheng, N. Lin, T. Liang, and V. Dinavahi, "Hardware-in-the-loop real-time transient emulation of large-scale renewable energy installations based on hybrid machine learning modeling," in *IEEE Journal of Emerging and Selected Topics in Industrial Electronics*, 2024.

[25] A. Pötsch and F. Hammer, "Towards End-to-End Latency of LoRaWAN: Experimental Analysis and IIoT Applicability," in *2019 15th IEEE International Workshop on Factory Communication Systems (WFCS)*, Sundsvall, Sweden, 2019, pp. 1-4, doi: 10.1109/WFCS.2019.8758033.

[26] F. Fahim, B. Hawks, C. Herwig, J. Hirschauer, S. Jindariani, N. Tran, and Z. Wu, "hls4ml: An open-source codesign workflow to empower scientific low-power machine learning devices," in *arXiv preprint arXiv:2103.05579*, 2021.

[27] N. Ghielmetti, V. Loncar, M. Pierini, M. Roed, S. Summers, T. Aarrestad, C. Petersson, H. Linander, J. Ngadiuba, K. Lin, and others, "Real-time semantic segmentation on FPGAs for autonomous vehicles with hls4ml," Machine Learning: Science and Technology, vol. 3, no. 4, pp. 045011, 2022.