Low-Latency FPGA-Based PCA Acceleration for Hyperspectral Image Dimensionality Reduction

Hana Ben Fredj[1]*, Ahlem kehili[2], Amani Chabbeh[3], Jamel Baili[4], Chokri Souani[5]
Laboratory of Electronics and Micro-Electronics, Faculty of Science of Monastir, Tunisia[1, 3]
Laboratory of Analysis and Processing of Electrical and Energetic Systems,
Faculty of Sciences of Tunis, Tunis El-Manar University, Tunisia[2]
College of Computer Science, King Khalid University, Abha, 61413, Saudi Arabia[4]
Higher Institute of Applied Sciences and Technology of Sousse, University of Sousse,
Tunisia, Laboratory of Electronics & Micro-electronics, Faculty of Science of Monastir, Tunisia[5]

***Abstract*—Remote sensing systems generate large volumes of high-dimensional data, which creates significant challenges for onboard processing and data transmission under computational, memory, and energy constraints. Dimensionality reduction is therefore essential for enabling efficient embedded remote sensing applications. This work presents a hardware-accelerated implementation of Principal Component Analysis (PCA) on an FPGA platform for low-latency onboard data reduction. A hardware–software co-design methodology is adopted and implemented using the Vivado design flow on an Xilinx FPGA. The proposed architecture focuses on accelerating the most computationally intensive stage of PCA while preserving the most informative components of the input data. Experimental evaluations demonstrate that the FPGA-based implementation achieves a 2.8× speedup while reducing energy consumption by 72.4% compared with an ARM-only software implementation, while maintaining efficient hardware resource utilization. These results confirm the suitability of the proposed accelerator for low-latency, energy-efficient onboard dimensionality reduction in resource-constrained remote sensing systems.**

***Keywords*—Remote sensing; hyperspectral data; dimensionality reduction; Principal Component Analysis (PCA); FPGA; hardware–software co-design**

I. Introduction

Remote sensing encompasses a wide range of techniques that enable the acquisition and analysis of information about observed scenes without direct physical contact. Among these techniques, hyperspectral imaging has gained significant attention due to its ability to capture fine-grained spectral information across hundreds of contiguous wavelength bands [1]. Unlike conventional multispectral sensors, hyperspectral systems acquire three-dimensional data cubes composed of two spatial dimensions and a highly resolved spectral dimension, where each pixel is characterized by a detailed spectral signature. This rich spectral representation enables accurate material discrimination and supports numerous applications, including environmental monitoring, precision agriculture, geological mapping, and natural resource management [2, 3].

Despite these advantages, hyperspectral imaging introduces major challenges related to data processing and analysis. The high dimensionality of hyperspectral data, combined with strong inter-band correlation, results in substantial data redundancy and increased sensitivity to noise and sensor imperfections [4]. These characteristics significantly increase computational complexity, memory requirements, and power consumption, rendering conventional software-based processing approaches inefficient for embedded applications.

These challenges are critical in Earth observation systems, where hyperspectral data must be processed onboard and transmitted under limited bandwidth, memory, and energy constraints. Dimensionality reduction is therefore essential to reduce data volume while preserving informative spectral features. Several methods, including Randomized Principal Component Analysis (RPCA) and unsupervised band selection techniques, have been widely used to reduce redundancy in hyperspectral data [5, 6]. Among them, PCA remains attractive due to its simplicity, robustness, and ability to retain most data variance with fewer components. Deep learning methods, such as hybrid 3D–2D CNNs, have also achieved strong performance and often use PCA to reduce input dimensionality and computational cost [7, 8,9] . Transformer-based models further improve spectral modeling but require high memory, computation, and energy resources [10–12]. To address these limitations, FPGA-based acceleration has been explored for PCA, hyperspectral compression, CNN inference, and anomaly detection [13–17], while in-sensor and processing-in-pixel paradigms have also been proposed to reduce data movement and energy consumption [18, 19]. Thus, PCA remains a suitable dimensionality reduction method for resource-constrained onboard hyperspectral systems.

In this context, this study proposes a real-time FPGA-based accelerator for Principal Component Analysis tailored to onboard hyperspectral data processing. The proposed solution adopts a hardware–software co-design approach and is implemented on a PYNQ-Z2 FPGA platform using the Vivado design flow. Unlike prior FPGA-based PCA accelerators that either lack configurability for varying hyperspectral dimensions or omit systematic energy analysis, the proposed architecture offers three distinctive contributions: 1) a fully pipelined, HLS-optimized covariance engine with loop unrolling (factor 8) specifically tuned for the B×B spectral covariance structure of hyperspectral cubes; 2) an AXI4-Master burst transfer strategy that aligns I/O parallelism with DDR memory bandwidth on the Zynq-7020; and 3) a hardware–software co-design that achieves an energy reduction of approximately 72% compared to ARM-only execution

*Corresponding author.

(from 4.39 J to 1.21 J per PCA run), alongside a 2.8× speedup. These characteristics address key gaps in the existing literature and confirm the suitability of the proposed solution for high-performance onboard hyperspectral processing under energy constraints.

II. Related Work

Hyperspectral image analysis has motivated extensive research on dimensionality reduction methods designed to reduce spectral redundancy while preserving the most informative features. Techniques such as Principal Component Analysis (PCA) [20], Singular Value Decomposition (SVD) [21], and band selection [22] have been widely used for dimensionality reduction in hyperspectral image analysis. Among them, PCA remains one of the most commonly adopted approaches because of its simplicity, robustness, and ability to transform highly correlated spectral bands into a reduced set of uncorrelated components. Unlike band selection methods, which preserve the original bands but may discard useful information, PCA provides a compact representation that concentrates most of the data variance in fewer components.

Nonlinear dimensionality reduction methods, including manifold learning approaches such as Locally Linear Embedding (LLE) [23] and Laplacian Eigenmaps (LE) [24], have also been explored to better capture complex spectral structures in hyperspectral data. However, these methods generally require higher computational resources and are sensitive to parameter settings [25], which limit their applicability in practice. For this reason, PCA remains widely adopted as an efficient preprocessing step in hyperspectral classification pipelines [26].

With the development of deep learning, several spatial–spectral models have been proposed for hyperspectral image classification. Hybrid CNN architectures have shown strong performance by extracting both spectral and spatial features. In many of these works, PCA is applied before classification to reduce the number of input bands and decrease the computational cost of 3D convolution operations. For example, Ghaderizadeh et al. [7] demonstrated that PCA-reduced hyperspectral data can be effectively used with hybrid 3D–2D CNN models to achieve competitive classification accuracy with reduced complexity. Similarly, Datta et al. [19] combined superpixelwise PCA with dense CNN-based spatial–spectral feature learning, further confirming the usefulness of PCA in deep learning-based hyperspectral analysis.

More recently, transformer-based architectures have been introduced to model long-range spectral dependencies through self-attention mechanisms. Models such as SpectralFormer [12] and MassFormer [11] have demonstrated promising classification performance by exploiting full spectral information. Nevertheless, these approaches usually involve high memory usage, significant computational complexity, and increased energy consumption. As a result, their deployment on resource-constrained onboard platforms remains challenging, particularly in applications where latency, memory, and power consumption are critical constraints.

To address these limitations, hardware acceleration has become a key direction for onboard hyperspectral processing. FPGAs are particularly suitable for this purpose due to their parallelism, deterministic execution, and higher energy efficiency compared with general-purpose processors. Fernández et al. [13] proposed an FPGA-based architecture for hyperspectral dimensionality reduction, demonstrating the potential of reconfigurable hardware for accelerating spectral projection operations. He et al. [14] developed a configurable FPGA accelerator for 2D–3D CNN-based hyperspectral classification, achieving efficient resource utilization and reduced inference latency through an HLS-based design flow.

Despite these advances, many existing FPGA-based implementations mainly focus on latency, throughput, or resource utilization separately. Few works provide a combined analysis of execution time, energy consumption, and configurability for different hyperspectral data dimensions. In addition, several prior designs do not provide sufficient details about memory transfer mechanisms or AXI-based communication strategies, which limits reproducibility and practical deployment.

Therefore, the present work addresses these limitations by proposing a hardware–software co-designed PCA accelerator implemented on the Zynq-7020 PYNQ-Z2 platform. The proposed architecture aims to jointly improve latency, energy efficiency, and configurability, while providing a detailed characterization of fixed-point arithmetic, AXI burst transfer, and energy consumption per PCA run.

III. Materials and Methods

A. PCA Algorithm for Hyperspectral Images

The conventional Principal Component Analysis (PCA) algorithm operates on two-dimensional data matrices. However, hyperspectral images are inherently multivariate datasets represented as three-dimensional data cubes, composed of two spatial dimensions $(D_{x'} D_y)$and one spectral dimension (D_λ). Therefore, the first step in applying PCA to hyperspectral data consists of unfolding (reshaping) the hyperspectral cube into a two-dimensional matrix.

$$X \in \mathbb{R}^{N \times B},$$

where, $N = D_x \times D_y$denotes the total number of pixels and $B = D_\lambda$ represents the number of spectral bands. In this representation, each row corresponds to the spectral signature of a single pixel, while each column represents a specific spectral band across the entire image. The unfolding process is illustrated in Fig. 1, where a hyperspectral cube composed of three rows, three columns, and B spectral bands is reshaped into a two-dimensional matrix suitable for PCA computation.

Before applying PCA, data standardization is carried out to ensure that all spectral bands contribute equally to the analysis. Each spectral variable is normalized by subtracting its mean and dividing by its standard deviation, as expressed by:

$$x'_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \quad (1)$$

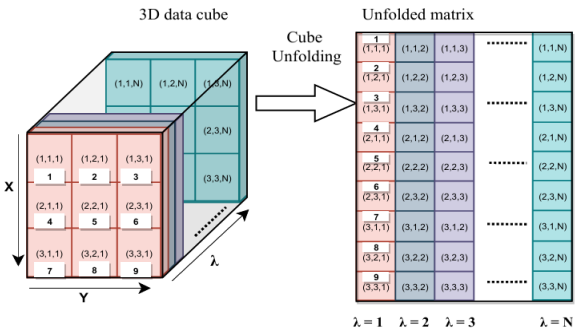

Fig. 1. Unfolding a hypercube.

where, x_{ij} denotes the reflectance value of the j-th spectral band at the i-th pixel, and μ_j and σ_j are the mean and standard deviation of the j-th band, respectively. This step enhances numerical stability and prevents spectral bands with larger dynamic ranges from dominating the PCA process.

After standardization, the covariance matrix is computed from the standardized data matrix X_s as:

$$C = \frac{1}{N-1} X_s^T X_s \tag{2}$$

The covariance matrix captures the inter-band spectral correlations and forms the foundation for dimensionality reduction.

PCA is then performed by applying eigenvalue decomposition to the covariance matrix:

$$C v_k = \lambda_k v_k \tag{3}$$

where, λ_k and v_k represent the eigenvalues and their corresponding eigenvectors, respectively. Each eigenvector defines a principal component, and its associated eigenvalue indicates the amount of variance explained by that component.

The eigenvalues are sorted in descending order, and the first K principal components are selected according to the cumulative explained variance criterion. The dimensionality-reduced data are then obtained by projecting the standardized data onto the selected eigenvectors:

$$Y = X_s V_k \tag{4}$$

where, V_k contains the K selected eigenvectors. The resulting matrix Y represents the reduced hyperspectral representation, corresponding to the PCA sub-bands that preserve the most informative spectral content of the original hyperspectral image while significantly reducing data dimensionality.

Hyperspectral images are characterized by a very high number of spectral bands, resulting in extremely high-dimensional data volumes and substantial computational complexity during onboard or embedded processing. The simultaneous handling of hundreds of spectral bands renders conventional software-based implementations inadequate for embedded and low-latency applications due to excessive execution latency, high memory bandwidth demands, and increased power consumption. In this context, Principal Component Analysis (PCA) is widely adopted as an effective and well-established dimensionality reduction technique, enabling hyperspectral data to be projected onto a lower-dimensional subspace while preserving most of the original data variance. However, to fully exploit the benefits of PCA in embedded sensing applications, an efficient hardware implementation becomes necessary to satisfy stringent latency and energy constraints while ensuring reliable and continuous data processing.

B. Proposed Hardware–Software Co-Design Architecture for PCA Acceleration

The proposed PCA acceleration architecture follows a hardware–software co-design approach. The covariance matrix computation, characterized by intensive numerical operations, is implemented as a custom FPGA IP core, while the remaining PCA stages are executed on the ARM-based processing system. These software-executed stages include data centering, eigenvalue decomposition, and principal component selection, which benefit from software flexibility and control.

Fig. 3 illustrates the proposed hardware–software co-design architecture of the FPGA-based PCA accelerator. The covariance IP is fully integrated into the FPGA and operates in a highly parallel and pipelined manner, leading to substantial performance gains compared to a pure software implementation. The reconfigurable nature of the FPGA allows the architecture to be tailored to different hyperspectral data dimensions by adjusting parameters such as the number of spectral bands and accumulation depth. By combining hardware-accelerated covariance computation with software-based PCA control, the proposed system achieves reduced latency, increased throughput, efficient hardware resource usage, and improved energy efficiency. These characteristics make the architecture well suited for high-performance hyperspectral data processing in embedded remote sensing applications.

The design methodology adopted in this work is based on Vivado High-Level Synthesis (HLS) and the Vivado Design Suite, as illustrated in Fig. 2, and is implemented on the PYNQ-Z2 development platform. The PCA algorithm is first described using high-level C/C++ code and validated through a dedicated C-based testbench, which serves as a golden reference model to ensure functional correctness. C-level simulation is then performed to verify the behavior of the algorithm prior to hardware synthesis. Vivado HLS is employed to translate the high-level PCA description into register-transfer level (RTL) code, while enabling the insertion of synthesis directives and optimization constraints to improve latency, throughput, and hardware resource utilization. After C/RTL co-simulation, the generated RTL is exported and integrated into a Zynq-7020–based PYNQ-Z2 system using Vivado. This workflow enables a seamless transition from algorithmic modeling to hardware implementation while preserving functional correctness and providing fine-grained control over performance optimization.

Among the different stages of the PCA algorithm, the covariance matrix computation represents one of the most computationally intensive operations. For hyperspectral images from the Pavia University dataset [22], this corresponds to

approximately N=207,400 samples and B=103 spectral bands, resulting in a 103×103 covariance matrix. This computation requires a large number of multiply–accumulate operations, leading to high execution time in software-based implementations.

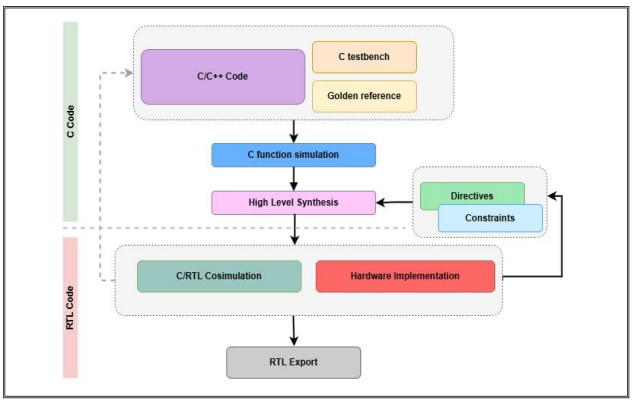

Fig. 2. Design and verification framework for custom hardware accelerator.

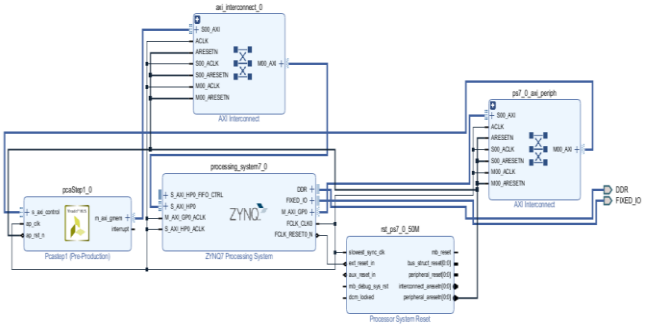

Fig. 3. Block diagram of the FPGA-based PCA acceleration architecture on Zynq SoC.

To reduce latency and meet stringent performance constraints, the covariance computation is offloaded to a dedicated FPGA IP core implemented in the programmable logic of the PYNQ-Z2 platform. The proposed IP exploits fine-grained parallelism through loop unrolling and loop pipelining directives in the HLS design, allowing multiple accumulation operations across spectral bands to be executed concurrently within each clock cycle. This significantly reduces the number of clock cycles required for covariance matrix generation and improves throughput while maintaining efficient hardware resource utilization.

To further enhance the performance of the covariance matrix computation, several HLS-level optimization strategies are applied. Loop unrolling with a factor of 8 (#pragma HLS unroll factor=8) enables parallel accumulation operations across spectral bands, while loop pipelining allows continuous processing of hyperspectral samples with a low initiation interval. Array partitioning is used to increase parallel memory access to spectral data buffers, and on-chip memory buffering is employed to locally store intermediate accumulation results, thereby reducing external DDR memory transactions.

In addition, fixed-point arithmetic is adopted using a 32-bit fixed-point representation with 16 integer bits and 16 fractional bits to provide a balance between numerical precision and hardware efficiency. Optimized AXI burst transfers are also employed to improve memory throughput and reduce power consumption. The covariance IP accesses hyperspectral data stored in external DDR memory via an AXI4-Master interface, while configuration parameters such as data dimensions and memory addresses are managed through an AXI-Lite control interface. An interrupt mechanism is used to notify the processing system upon completion of the covariance computation.

IV. Results

This section reports the experimental results of the proposed PCA accelerator implemented on the PYNQ-Z2 FPGA. The evaluation considers hyperspectral data from the Pavia University dataset and analyzes both the quality of the extracted principal components and the performance of the hardware-accelerated implementation in terms of execution time, power consumption, and resource usage. A comparison with a software-based PCA execution on the ARM Cortex-A9 is also presented to demonstrate the advantages of the proposed hardware solution.

TABLE I. FPGA Resource Utilization After Vivado Implementation of the PCA Accelerator System on PYNQ-Z2 (XC7Z020)

Resource (PL)	Used	Available	Utilization (%)
LUT	16,881	53,200	31.7%
FF	14,673	106,400	13.8%
BRAM	1	140	0.7%
DSP48	174	220	79.1%

Table I presents the FPGA resource utilization obtained after Vivado post-implementation of the PCA accelerator system on the PYNQ-Z2 platform based on the Xilinx Zynq-7020 (XC7Z020) SoC. The results show a moderate consumption of general logic resources, with 16,881 LUTs (31.7%) and 14,673 flip-flops (13.8%), indicating that the proposed architecture maintains a balanced use of combinational and sequential logic for the complete programmable-logic design, including the system interconnect, control and interface logic, together with the PCA accelerator.

In contrast, the utilization of DSP48 blocks reaches 174 units (79.1%), which is consistent with the arithmetic-intensive nature of the covariance matrix computation, dominated by multiply–accumulate operations and matrix-based numerical processing. This confirms that the design efficiently maps computationally demanding operations onto dedicated DSP resources, while preserving LUT resources mainly for control and interfacing purposes. The BRAM usage is very limited, with only one BRAM block used (0.7%), reflecting a lightweight on-chip memory footprint within the programmable logic.

Overall, these results demonstrate an efficient hardware implementation of the PCA accelerator on PYNQ-Z2, achieving high computational density while maintaining moderate logic and memory utilization, making it suitable for embedded dimensionality reduction applications with stringent performance requirements.

TABLE II. PERFORMANCE COMPARISON OF THE PCA IMPLEMENTATION ON ARM CORTEX-A9 AND ZYNQ XC7Z020.

Platform	Execution Time (s)	Power (W)
Dual-Core ARM Cortex-A9	1.83	2.4
Zynq XC7Z020	0.65	1.86

To evaluate the performance benefits of hardware acceleration for the Principal Component Analysis (PCA) algorithm, the proposed implementation was evaluated on two different platforms: a software-only execution on the dual-core ARM Cortex-A9 processor and a hardware-accelerated implementation on the Zynq XC7Z020 FPGA.

Table II compares the execution time and power consumption of the PCA implementation when executed entirely on the ARM Cortex-A9 processor and when accelerated using the FPGA resources of the Zynq XC7Z020 platform. The software-only execution on the ARM Cortex-A9 requires 1.83 s, reflecting the sequential nature of the PCA covariance computation and the limited parallelism available on a general-purpose processor.

The FPGA-accelerated implementation significantly reduces the execution time to 0.65 s, corresponding to a speed-up of approximately 2.8×. This improvement is achieved by offloading the computationally intensive stages of the PCA to dedicated hardware, enabling parallel processing and efficient pipelining in the programmable logic.

From a power perspective, the FPGA-based solution also demonstrates an advantage, consuming 1.86 W compared to 2.4 W for the ARM-only execution. Although both platforms operate within acceptable power limits, the combination of lower execution time and reduced power consumption highlights the superior energy efficiency of the FPGA-accelerated approach. Overall, these results confirm that integrating a hardware accelerator for PCA on the Zynq XC7Z020 platform provides a favorable trade-off between performance and power, making it well suited for embedded and real-time signal processing applications.

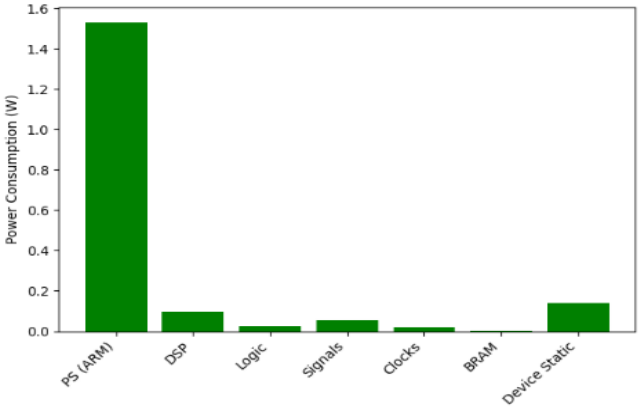

Fig. 4. Power consumption breakdown of the PCA system on the Zynq XC7Z020 platform.

To evaluate the power efficiency of the proposed PCA accelerator and to illustrate the distribution of power consumption between the processing system and the programmable logic, Fig. 4 presents the power consumption breakdown of the PCA system implemented on the Zynq XC7Z020 platform. The results show a total on-chip power consumption of 1.867 W, with 1.726 W (92.4%) corresponding to dynamic power and 0.141 W (7.6%) to static power, indicating that the system consumption is largely dominated by active computation. As shown in Fig. 4, the Processing System (PS7) represents the main contributor, consuming 1.530 W, which corresponds to 88.6% of the dynamic power and 81.9% of the total power. In contrast, the programmable logic (PL), including the PCA hardware accelerator, accounts for only 0.196 W of dynamic power (11.4% of dynamic power and 10.5% of total power), demonstrating the low energy overhead introduced by hardware acceleration. Within the PL, the power consumption is mainly dominated by the DSP blocks (0.094 W, 5.0%), which is consistent with the arithmetic-intensive nature of the PCA covariance computation, while the contributions of logic (0.025 W, 1.3%), clocks (0.020 W, 1.1%), signals (0.055 W, 2.9%), and BRAM (0.002 W, 0.1%) remain negligible. Although the PS dominates instantaneous power consumption, the reduced execution time achieved by the FPGA accelerator leads to an energy-efficient PCA implementation suitable for embedded and performance-constrained applications.

To fully characterize the efficiency of the proposed accelerator, energy consumption per PCA run is quantified as $E = P \times T$ and further normalized to the dataset dimensions ($N = 207{,}400$ pixels, $B = 103$ spectral bands) to derive the energy per pixel per band. The resulting values are reported in Table III.

TABLE III. ENERGY EFFICIENCY COMPARISON PER PCA RUN

Platform	Energy per Run (J)	Energy per pixel/band (nJ)
Dual-Core ARM Cortex-A9	4.39	205
Zynq XC7Z020 (FPGA)	1.21	56.6

The results show that the FPGA-accelerated implementation consumes only 1.21 J per PCA run, compared to 4.39 J for the ARM-only execution, achieving an energy reduction of approximately 72.4%. At the pixel–band level, the energy footprint decreases from approximately 205 nJ/pixel/band to 56.6 nJ/pixel/band, showing that the efficiency gain is reflected across the processed hyperspectral data volume. This reduction results from the simultaneous decrease in both execution time and average power consumption. Such an improvement is particularly important for embedded remote sensing platforms, where energy consumption, memory usage, and execution time must be carefully optimized. These results demonstrate that the proposed PCA accelerator provides not only a performance advantage but also a substantial improvement in energy efficiency, supporting its potential for energy-constrained onboard remote sensing applications.

To illustrate the effect of Principal Component Analysis (PCA) on hyperspectral data, PCA was applied to hyperspectral images from the Pavia University dataset, and the resulting principal components were visualized. This representation enables an initial qualitative assessment of how the spectral information contained in the Pavia University

dataset is redistributed across the principal components. Fig. 5 illustrates the first eight principal components obtained after applying Principal Component Analysis (PCA) to the hyperspectral dataset. It can be observed that the first principal components (Bands 1–3) preserve the most significant spatial structures of the scene, with clear edges, textures, and man-made objects being distinctly visible. These components concentrate the majority of the data variance and therefore carry the most relevant spectral–spatial information. The intermediate components (Bands 4 and 5) still contain meaningful information but with reduced contrast and fewer discernible details, indicating that they mainly capture secondary variations in the data. In contrast, the higher-order components (Bands 6–8) exhibit low spatial structure and appear increasingly uniform, suggesting that they are dominated by noise and minor spectral variations rather than useful information.

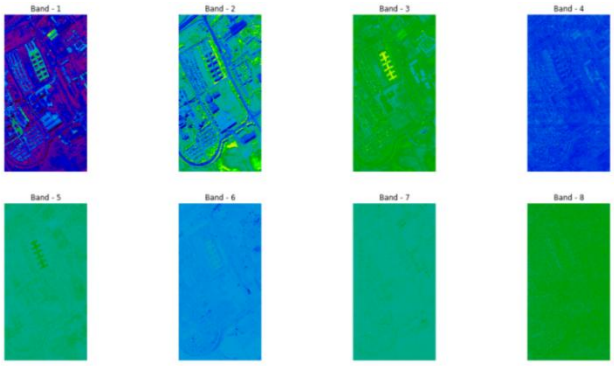

Fig. 5. The first principal components obtained from hyperspectral images.

V. Discussion

The proposed PCA acceleration architecture adopts a hardware–software co-design approach that effectively balances performance and flexibility for hyperspectral data processing. By selectively offloading the covariance matrix computation to the programmable logic, the system accelerates the most computationally intensive stage of PCA while maintaining software-level control for the remaining processing steps.

The covariance computation is implemented as a custom FPGA IP core generated using Vivado High-Level Synthesis and integrated into the Zynq-7020–based PYNQ-Z2 platform. This IP exploits fine-grained parallelism through advanced HLS optimization techniques, including loop pipelining and loop unrolling, enabling concurrent accumulation operations across multiple spectral bands. As a result, the number of clock cycles required for covariance matrix generation is significantly reduced compared to a pure software implementation on the ARM Cortex-A9 processor.

Overall, the combination of HLS-based design, pipelined execution, and parallel accumulation allows the proposed architecture to achieve reduced latency, increased throughput, and efficient FPGA resource utilization. These characteristics demonstrate the suitability of the proposed solution for onboard processing in embedded remote sensing systems operating under strict power and computational constraints.

From a system-level perspective, the proposed architecture is designed to accommodate the statistical variability inherent to hyperspectral data acquired under different sensing conditions. The hardware–software co-design paradigm provides sufficient flexibility to adapt the processing flow through software control and parameter configuration, reinforcing the robustness and applicability of the approach in practical remote sensing scenarios.

Future work will focus on further enhancing the flexibility and scalability of the proposed PCA accelerator. One important direction is the development of more configurable architectures capable of adapting to varying input dimensions and data characteristics while preserving real-time performance.

In addition, extending the current design to support more advanced dimensionality reduction techniques or hybrid FPGA–AI approaches represents a promising direction to further improve performance and enable more complex onboard remote sensing applications.

VI. Conclusion

This work presented an FPGA-based hardware acceleration framework for Principal Component Analysis (PCA) applied to hyperspectral image processing, following a hardware–software co-design methodology. Through a detailed analysis of the PCA processing pipeline, the covariance matrix computation was identified as the most computationally demanding stage due to its intensive arithmetic operations and high data volume.

To address this challenge, a dedicated PCA covariance hardware intellectual property (IP) core was designed using Vivado High-Level Synthesis and integrated into a Zynq-7020–based PYNQ-Z2 platform. By exploiting fine-grained parallelism through loop pipelining and unrolling, the proposed accelerator significantly reduced execution latency compared to a software implementation on a dual-core ARM Cortex-A9 processor (from approximately 1.83 s to 0.65 s), while also lowering power consumption (from 2.4 W to 1.86 W).

These results confirm the effectiveness of the proposed hardware acceleration strategy in enabling hyperspectral data processing under strict computational and energy constraints, while maintaining efficient FPGA resource utilization.

Future work will focus on extending hardware acceleration to additional PCA stages and exploring further architectural optimizations to enhance scalability and throughput on more advanced FPGA platforms.

References

[1] D. Cozzolino, P. J. Williams, and L. C. Hoffman, “An overview of pre-processing methods available for hyperspectral imaging applications,” Microchem. J., vol. 193, p. 109129, 2023.

[2] R. Vaddi, B. L. N. Phaneendra Kumar, P. Manoharan, L. Agilandeeswari, and V. Sangeetha, “Strategies for dimensionality reduction in hyperspectral remote sensing: A comprehensive overview,” Egypt. J. Remote Sens. Space Sci., vol. 27, no. 1, pp. 82–92, 2024.

[3] R. Qureshi, M. Uzair, K. Khurshid, and H. Yan, "Hyperspectral document image processing: Applications, challenges and future prospects," Pattern Recognition, vol. 90, pp. 12–22, 2019.

[4] L. Zhu, Y. Chen, P. Ghamisi, J. A. Benediktsson, M. Li, and R. Tao, “Generative adversarial networks for hyperspectral image

classification," IEEE Trans. Geosci. Remote Sens., vol. 56, no. 9, pp. 5046–5063, Sep. 2018.

[5] M. Ustuner, "Randomized principal component analysis for hyperspectral image classification," in 2024 IEEE Mediterranean and Middle-East Geoscience and Remote Sensing Symposium, pp. 26–30, 2024.

[6] R. Vaddi and P. Manoharan, "Hyperspectral remote sensing image classification using combinatorial optimisation based un-supervised band selection and CNN," IET Image Process., vol. 14, no. 15, pp. 3909–3919, 2020.

[7] S. Ghaderizadeh, D. Abbasi-Moghadam, A. Sharifi, N. Zhao, and A. Tariq, "Hyperspectral image classification using a hybrid 3D–2D convolutional neural network," IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 15, pp. 2379–2393, 2022

[8] H. Firat, M. E. Asker, and D. Hanbay, "Classification of hyperspectral remote sensing images using different dimension reduction methods with 3D/2D CNN," Remote Sens. Appl.: Soc. Environ., vol. 25, p. 100694, 2022.

[9] D. Datta, P. K. Mallick, D. Gupta, et al., "Hyperspectral image classification based on novel hybridization of spatial-spectral-superpixelwise principal component analysis and dense 2D–3D convolutional neural network fusion architecture," Can. J. Remote Sens., vol. 48, no. 5, pp. 663–680, 2022.

[10] L. Scheibenreif, M. Mommert, and D. Borth, "Masked vision transformers for hyperspectral image classification," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), pp. 2166–2176, 2023..

[11] L. Sun, H. Zhang, Y. Zheng, et al., "Massformer: Memory-augmented spectral-spatial transformer for hyperspectral image classification," IEEE Trans. Geosci. Remote Sens., vol. 62, pp. 1–15, 2024.

[12] D. Hong, Z. Han, J. Yao, et al., "SpectralFormer: Rethinking hyperspectral image classification with transformers," IEEE Trans. Geosci. Remote Sens., vol. 60, pp. 1–15, 2021.

[13] D. Fernández, C. González, D. Mozos, and S. López, "FPGA implementation of the principal component analysis algorithm for dimensionality reduction of hyperspectral images," Journal of Real-Time Image Processing, vol. 16, no. 5, pp. 1395–1406, 2019.

[14] W. He, Y. Yang, S. Mei, J. Hu, W. Xu, and S. Hao, "Configurable 2D–3D CNN accelerator for FPGA-based hyperspectral imagery classification," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 16, pp. 9406–9421, 2023.

[15] K. Avagian and M. Orlandić, "An efficient FPGA implementation of Richardson–Lucy deconvolution algorithm for hyperspectral images," Electronics, vol. 10, no. 4, p. 504, 2021.

[16] J. Caba, M. Díaz, J. Barba, A. Sánchez, and R. Sarmiento, "FPGA-based on-board hyperspectral imaging compression: Benchmarking performance and energy efficiency against GPU implementations," Remote Sens., p. 3741, 2020.

[17] J. Caba, M. Díaz, J. Barba, A. Sánchez, and R. Sarmiento, "Low-power hyperspectral anomaly detector implementation in cost-optimized FPGA devices," IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 15, pp. 2379–2393, 2022.

[18] G. Datta, Z. Yin, A. Jacob, A. Majumdar, and A. Veeraraghavan, "Toward efficient hyperspectral image processing inside camera pixels," arXiv preprint arXiv:2203.05696, 2022.

[19] G. Datta and C. Zhou, "Analog vs. Digital In-Sensor Computing: A Tale of Two Paradigms," in Proc. Great Lakes Symp. VLSI (GLSVLSI), pp. 829–834, 2025.

[20] K. M. Lim, C. P. Lee, Z. Zahisham, J. Y. Lim, and J. N. Mogan, "PCA-ViT: Hyperspectral image classification using principal component analysis and vision transformer," in Proc. IEEE 12th Conf. Syst., Process Control (ICSPC), pp. 30–34, 2024.

[21] M. M. Rahman, M. R. Islam, M. I. Afjal, M. A. H. Akhand, and M. A. H. Khan, "Segmentation-based truncated-SVD for effective feature extraction in hyperspectral image classification," Int. J. Remote Sens., vol. 46, no. 2, pp. 538–574, 2025.

[22] W. Kurz, K. Wang, F. Bektaş, M. M. Rahman, and M. M. Islam, "Dimensionality reduction in hyperspectral imaging using standard deviation-based band selection for efficient classification," Sci. Rep., vol. 15, no. 1, p. 34478, 2025.

[23] H. Ji and Z. Zuo, "Multiview locally linear embedding for spectral-spatial dimensionality reduction of hyperspectral imagery," IEEE/CAA Journal of Automatica Sinica, vol. 9, no. 6, pp. 1091–1094, 2022..

[24] S.-E. Qian and G. Chen, "A new nonlinear dimensionality reduction method with application to hyperspectral image analysis," in 2007 IEEE International Geoscience and Remote Sensing Symposium, pp. 270–273, 2007.

[25] D. Lunga, S. Prasad, M. M. Crawford, and O. Ersoy, "Manifold-learning-based feature extraction for classification of hyperspectral data: A review of advances in manifold learning," IEEE Signal Processing Magazine, vol. 31, no. 1, pp. 55–66, 2014.

[26] W. Song, X. Zhang, G. Yang, Y. Liu, and X. Li, "A study on dimensionality reduction and parameters for hyperspectral imagery based on manifold learning," Sensors, vol. 24, no. 7, p. 2089, 2024.