At the Edge of the Heart: ULP FPGA-Based CNN for On-Device Cardiac Feature Extraction in Smart Health Sensors for Astronauts

Kazi Mohammad Abidur Rahman[1], Davis Rakhshan[1], Philipp Lütke[1], Laura Harms[3,2], Ulf Kulau[1]
[1]*Smart Sensors Group, Hamburg University of Technology*, Hamburg, Germany
[2]*Networked Cyber-Physical Systems, Hamburg University of Technology*, Hamburg, Germany
[3]*Distributed Systems, Kiel University*, Kiel, Germany
kazi.rahman@tuhh.de, davis.rakhshan@tuhh.de, philipp.luetke@tuhh.de, laura.harms@cs.uni-kiel.de, ulf.kulau@tuhh.de

***Abstract*—The convergence of accelerating human spaceflight ambitions and critical terrestrial health monitoring demands is driving the unprecedented requirements for reliable and real-time feature extraction capabilities on extremely resource-constraint wearable health sensors. We present a ULP FPGA-based solution for real-time Seismocardiography (SCG) feature classification using Convolutional Neural Networks (CNNs). Our approach combines quantization-aware training with a systolic-array accelerator to enable efficient integer-only inference on the Lattice iCE40UP5K FPGA, which offers an ideal platform for battery-powered deployments – particularly in space environments – for their power efficiency and radiation resilience. The implementation achieves a validation accuracy of 98% while consuming only 8.55 mW of power, completing inference in 95.5 ms with minimal hardware resources (2,861 LUTs and 7 DSP blocks). These results demonstrate that fully on-device SCG-based cardiac feature extraction is feasible on resource-constrained hardware, enabling energy-efficient, autonomous health monitoring for astronauts in long-duration space missions.**

***Index Terms*—FPGA, Ultra-Low-Power, CNN, Systolic Array, Time-Series-Classification, SCG, Space Wearables**

I. Introduction

As humanity pushes toward sustained exploration of the Moon and Mars, safeguarding astronaut health, one of the most critical priorities in space missions, demands a new generation of intelligent, wearable health monitoring systems capable of operating far beyond Low Earth Orbit (LEO). Unlike terrestrial or near-Earth deployments, deep-space missions impose severe communication constraints, where latency to ground stations can extend to minutes, rendering continuous remote supervision infeasible. Consequently, physiological monitoring systems must achieve unprecedented levels of autonomy, reliability, and real-time decision-making to enable timely health assessment and intervention.

Building upon insights from previous cardiovascular monitoring experiments conducted aboard the International Space Station (ISS) [1]–[3], this work fosters the development of smart, space-grade wearable sensors specifically designed for continuous cardiac monitoring. Seismocardiography (SCG), which captures the mechanical vibrations of the heart using inertial sensors, offers a non-invasive and compact sensing modality well suited for long-duration missions. However, robust extraction of clinically meaningful cardiac features from SCG signals, particularly systolic and diastolic intervals, remains challenging due to motion artifacts, physiological variability, and altered cardiac mechanics in microgravity.

Recent advances in deep learning have demonstrated that CNNs can robustly extract cardiac features from SCG signals by learning hierarchical temporal-spectral representations directly from raw data [4], [5]. While CNNs enable accurate systolic-diastolic segmentation and feature extraction, their computational complexity presents a significant challenge for deployment on wearable, spaceborne platforms. Deep-space health monitoring systems must operate under extreme constraints: they must be highly miniaturized to meet spacecraft mass and volume limitations and exhibit exceptional energy efficiency to support prolonged autonomous operation. In addition the radiation environment limits available processing components that could be used within this harsh environment. These combined requirements preclude the use of conventional radiation-hardened components, which are typically bulky and power-hungry.

To address these constraints, this work adopts a carefully selected Commercial Off-The-Shelf (COTS) design strategy. A key enabler of this approach is the Lattice iCE40UP-series ultra-low-power (ULP) Field Programmable Gate Array (FPGA), which offers milliwatt-level power consumption alongside demonstrated resilience to radiation-induced effects [6]. This ULP FPGA provides a unique balance between energy efficiency, flexibility, and reliability, making it well suited for continuous, real-time physiological monitoring in space. However, deploying CNN inference on such resource-constrained hardware requires algorithm-architecture co-design to overcome limitations in logic resources, memory capacity, and arithmetic precision.

This paper demonstrates that ULP FPGAs using co-designed CNN models, quantization-aware training, and a systolic-array

The biomedical data used in this study were collected with approval from the Ethics Committee of Hamburg University of Technology.

accelerator are capable of classifying systolic and diastolic phases in wearable SCG signals accurately and in real-time.

The outline of this paper is as follows: Section II reviews related work, followed by short introduction to the space-grade SCG health sensor hardware architecture in Section III. Section IV describes SCG data acquisition and labeling, while Section V presents the CNN design and its optimization for resource-constrained devices. Section VI details the ULP FPGA inference implementation[1]. Section VII evaluates accuracy and performance, and Section VIII concludes the paper.

II. Related Works

Continuous cardiovascular monitoring is essential for astronaut health, as microgravity induces cardiac deconditioning and altered hemodynamics [7]. Wearable SCG offers a non-invasive, lightweight modality for monitoring cardiac mechanical activity in space; however, reliable analysis beyond heart rate (HR) and heart rate variability (HRV) remains challenging due to motion artifacts, inter-cycle variability, sensor placement sensitivity, and microgravity-induced morphological changes [8], [9]. Accurate systolic-diastolic segmentation is therefore critical, as key mechanical events such as mitral and aortic valve motions occur in distinct cardiac phases and enable extraction of physiologically meaningful metrics including the systole-diastole ratio (SDR), which reflects ventricular ejection-filling balance and is sensitive to myocardial relaxation, coronary perfusion, and hemodynamic load [10], [11]. Traditional SCG analysis based on handcrafted features and classical classifiers such as Support Vector Machine (SVM) and K-Nearest Neighbors (KNN) degrades under physiological variability, whereas recent studies show that CNNs consistently outperform these approaches by learning hierarchical temporal-spectral representations directly from raw SCG signals [4], [5].

Despite their effectiveness, CNNs are computationally intensive, motivating hardware acceleration for real-time, on-device inference. Existing FPGA-based CNN accelerators primarily target high-end platforms such as Virtex-7, Zynq, Stratix-V, and Cyclone-V [12], [13]. These devices provide extensive digital signal processor (DSP) resources, large on-chip memories, and power budgets ranging from 18 W to 50 W [14], [15]. While suitable for high-throughput inference, such platforms are incompatible with wearable and ULP health monitoring systems. In contrast, resource-constrained FPGAs such as the Lattice iCE40 family operate in the milliwatt range but offer limited logic resources, minimal DSP support, and small memory capacities, rendering conventional CNN mapping strategies impractical.

For CNN inference on ultra-constrained devices, hardware capabilities and on-chip storage must be carefully considered, as memory is primarily allocated to model parameters, including weights and biases. Floating-point inference is generally infeasible on ULP FPGAs; therefore, quantization is essential. While post-training quantization can reduce model size, Quantization-Aware Training (QAT) explicitly models reduced-precision arithmetic during training, enabling accurate integer-only inference at low bit widths. This approach significantly reduces memory footprint and aligns computation with resource-constrained hardware datapaths [16], [17], making it particularly suitable for ULP FPGA-based CNN deployment.

[1]Implementation code available at: https://github.com/Smart-Sensors-Group/NN_SysArray.git

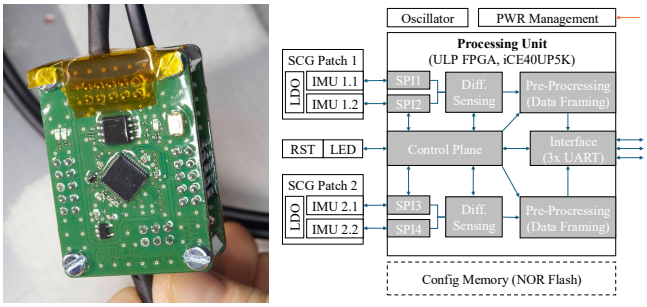

Fig. 1: The processing unit of the wearable health sensor used by astronauts during ESA's Cosmic Kiss mission on the International Space Station (ISS) [1]

Systolic array architectures, introduced by H.T. Kung [18], offer an efficient solution for CNN inference on resource-constrained FPGAs. Composed of two-dimensional grids of processing elements (PEs) with local registers, they enable pipelined data movement and efficient data reuse [14], [19]. By propagating intermediate results between adjacent PEs, systolic arrays reduce on-chip and off-chip memory accesses, lowering energy consumption, memory bandwidth requirements, and I/O bottlenecks while maintaining balanced workloads [14], [20].

Hence, systolic arrays are well matched to CNNs because convolution operations rely heavily on matrix and tensor multiplications that map naturally onto their pipelined structure [14]. The predictable computation patterns of CNNs further mitigate issues related to irregular parallelism. Consequently, systolic arrays form the core of modern Tensor Processing Units (TPUs), Neural Processing Units (NPUs), and Graphics Processing Units (GPUs) and are widely used in signal and image processing applications [21]. These characteristics enable real-time, on-device SCG-based cardiac feature extraction on ULP FPGA platforms.

III. Space-grade SCG Health Sensor

Before presenting the proposed CNN model and inference architecture, we introduce the underlying hardware platform and its suitability for ULP wearable health sensing in extreme and radiation-exposed environments. For space applications, FPGAs are typically preferred over microcontroller units (MCUs) due to their higher radiation tolerance, computational throughput, deterministic timing, and fine-grained architectural controllability. The Lattice iCE40UP5K ULP FPGA [22] aligns well with these requirements and has demonstrated suitability through prior flight heritage [1] and radiation characterization studies [6].

Figure 1 illustrates the iCE40UP5K-based processing unit used in ESA's *Cosmic Kiss* mission on the ISS [3], which also

serves as the base sensor platform for this work. This platform enables two-channel SCG acquisition using high-precision Inertial Measurement Units (IMUs). Each SCG channel (Patch 1, Patch 2) uses differential sensing [1] to facilitate ultra-low-noise signal acquisition. The system samples at 1 kHz, and FPGA-based acquisition ensures precise clock synchronization and cycle-accurate inter-channel alignment.

From a CNN acceleration perspective, the iCE40UP5K is highly resource constrained but provides a set of ultra-low-power hardware primitives, integrating 128 kB of Single-Port RAM (SPRAM), organized as four 32 kB blocks (16K×16-bit), for storing convolution weights and intermediate activation buffers, alongside 30 Block RAM (BRAM) instances totaling 4 kB that support dual-port access and synthesis-time initialization and are therefore used for precision-critical parameters such as biases. Computation is supported by eight embedded 16×16-bit DSP blocks capable of single-cycle multiply-accumulate (MAC) operations, which can be organized into a systolic MAC structure and time-multiplexed across kernel taps and channel dimensions to efficiently execute fixed-point convolutional and fully connected layers under tight resource constraints.

IV. Data Acquisition and Labelling

For data acquisition, we used the ground reference system of the space-grade SCG health sensor described in Section III and extended this platform with a single-lead Electrocardiography (ECG) for reference purposes [23].

For each experiment, an SCG Sensor Patch was placed on the subject's chest, in particular on the sternum, which is well suited regardless of the subject's gender. Both SCG and ECG data were collected under ethical approval from six subjects (age range 13-40 years), including four male and two female participants. Each subject was recorded continuously for 60 minutes. To increase signal variability and capture physiological changes due to higher heart and breathing rates, subjects performed a brief running activity after every 15 minutes of recording.

During labeling, Systolic and Diastolic cardiac phases were annotated using a semi-automated labeling pipeline. Initial pre-labels were generated using the wavelet-driven automated SCG Systolic/Diastolic window extraction approach described in [24]. These predictions were imported into Label Studio [25] and subsequently refined through manual correction to ensure high labeling accuracy.

V. Neural Network Design

To classify the cardiac phases, we build a neural network composed of a sequence of one-dimensional convolutional blocks that extract hierarchical temporal features from fixed-length SCG signal windows. Each block consists of a 1D convolution followed by batch normalization, a ReLU activation, and max pooling. This structure enables progressive abstraction of temporal patterns while reducing the temporal resolution of the signal.

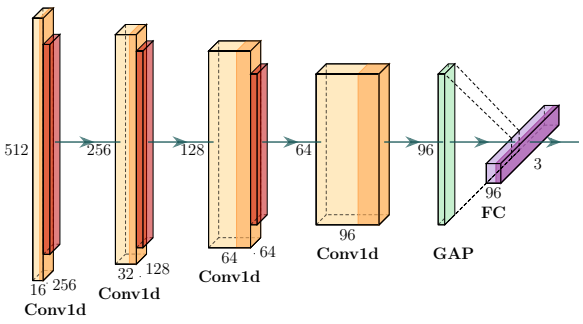

Fig. 2: Neural network architecture for SCG feature extraction. Yellow: 1D convolution, red: batch normalization, green: global average pooling (GAP), violet: fully-connected (FC).

We chose a one-dimensional convolutional neural network for its ability to learn temporal representations directly from raw signals while remaining compact and hardware-efficient. The proposed model operates on fixed-length SCG windows and is designed to balance classification performance with the power, memory, and latency constraints required for on-device processing in smart health sensors.

The input to the network is a single-channel SCG signal window, which is classified into one of three classes: (1) *Systolic*, (2) *Diastolic*, and (3) *Background*. The input signal is normalized using per-window z-score normalization to reduce amplitude variations across recordings. During training, the target label corresponds to the class of the sample located at the center of the input window.

We illustrate the network architecture in Figure 2, which shows the layer-wise structure and corresponding dimensionality changes.

All convolutional layers use a kernel size of 9, which provides sufficient temporal context to capture characteristic cardiac signal patterns while maintaining computational efficiency and a limited parameter budget. A padding of 4 is applied to preserve temporal resolution across these layers. In the final convolutional layer, the kernel size is reduced to 5 with a padding of 2, reflecting the increased level of abstraction in deeper layers. Three convolutional blocks are stacked with channel dimensions $16 \rightarrow 32 \rightarrow 64$, followed by a final convolutional layer of size 96 and global average pooling.

The model is trained using a weighted cross-entropy loss to address class imbalance between background and event samples. The loss weights emphasize systolic and diastolic events relative to background. Optimization is performed using the AdamW optimizer [26] with weight decay for regularization, and dropout is applied in the classification head to reduce overfitting.

The pooled features are processed by a lightweight fully connected classification head that outputs class probabilities for the three target classes.

TABLE I: Confusion matrix for the FP32 model on the test set. Rows denote ground truth, columns denote predictions.

True \ Pred	Background	Systolic	Diastolic
Background	9469	39	383
Systolic	55	9914	125
Diastolic	44	45	9926

TABLE II: Classification performance and model size comparison between FP32 and INT8 (QAT) models.

Metric	FP32	INT8 (QAT)
Validation accuracy [%]	98.68	98.32
Test accuracy [%]	97.70	97.68
Model size [KB]	226.5	68.2

A. Optimization

Since the target application focuses on near-sensor inference, Quantization-Aware Training (QAT) is employed to reduce the model weights and activations to 8-bit integer precision while preserving most of the classification accuracy. Compared to post-training quantization, QAT generally results in a lower accuracy degradation at the cost of retraining the network. Since the trained model is deployed on the target device exclusively for inference, the additional training cost is negligible. During inference, the intermediate results of the multiply-accumulate operations are accumulated in higher precision and must be rescaled to match the fixed 8-bit activation format. This re-quantization step is necessary because applying the scaling multiplier to the 32-bit accumulation produces a 64-bit intermediate value, which must then be reduced back to 8-bit. This precision was chosen to meet strict memory constraints, since higher precision would exceed the limit or require removing model structures, leading to reduced accuracy.

Table I presents the confusion matrix of the FP32 model evaluated on the test set. The results indicate strong class separability, with the majority of misclassifications occurring between background and event classes, while confusion between systolic and diastolic events remains limited. This behavior is consistent with the temporal proximity of cardiac events and validates the robustness of the learned feature representations.

As shown in Table II, quantization results in only a marginal degradation in classification performance, with a decrease of 0.36 percentage points in validation accuracy and 0.02 percentage points in test accuracy, while reducing the model size by approximately 70 %.

At inference time, several operations can be fused to further improve efficiency. Since batch normalization parameters are fixed after training, the batch normalization operation can be absorbed into the preceding convolution by folding the scale and bias terms into the convolution weights and bias. Additionally, the ReLU activation is implicitly fused during quantized inference, as it is performed through clamping the re-quantized outputs to the non-negative range of 8-bit integers. Furthermore, the multiplicators used to scale the intermediate values at re-quantization can be fused in one multiplicator for inference.

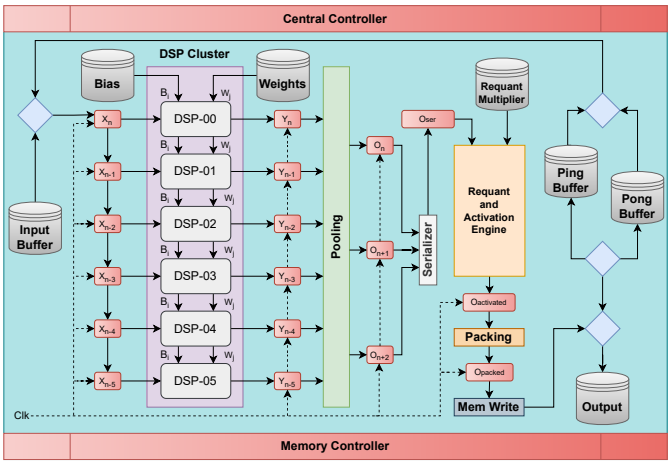

Fig. 3: Proposed FPGA inference architecture. Components: Memory Subsystem (Weight Memory, Input Buffer, Ping-Pong Buffers, Bias/Scale ROMs), Compute Subsystem (DSP Systolic Cluster, Pooling Unit, Requant & Activation Engine, Result Packer), Control Subsystem.

VI. ULP FPGA Inference

This section describes the implementation of a dedicated CNN accelerator for one-dimensional convolutions in biomedical signals on ULP FPGA (iCE40UP5K), organized around a systolic-array dataflow for efficient, low-latency computation as illustrated in Figure 3.

A. Memory Subsystem

The proposed architecture exploits the iCE40UP5K's heterogeneous memory, combining SPRAM for high-density storage and dual-port BRAM for parallel access. The memory subsystem is organized hierarchically into four functional blocks-Weight Memory, Input Buffer, Ping-Pong Buffers, and ROMs for biases and scaling factors-maximizing bandwidth while avoiding compute-control conflicts.

The Weight Memory stores quantized INT8 model parameters and is implemented using two cascaded 16K×16-bit SPRAM primitives, forming a continuous 32K-word (64 KB) address space. An address-decoding scheme uses the most significant bit (MSB) of the 15-bit address bus to select between the Lower Bank (`0x0000-0x3FFF`) and Upper Bank (`0x4000-0x7FFF`). Two 8-bit weights are packed per 16-bit word, doubling effective read bandwidth and enabling a single fetch cycle to supply operands for two sequential multiply-accumulate operations in the compute cluster. Weights can be dynamically loaded at runtime via the Universal Asynchronous Receiver Transmitter (UART) interface using the packet-based loading mechanism described in Section VI-C.

Bias and Scale ROMs store precision-critical constants for inference. Bias values for up to 512 output channels are

held in four SB_RAM40_4K block RAMs, each storing 32-bit values. Scale multipliers are implemented using D flip-flops (DFFs), with one 32-bit scale value per layer. Both Bias and Scale ROMs are statically initialized during synthesis, ensuring immediate availability at system boot without external configuration or runtime loading.

The Input Buffer accommodates incoming 1D time-series input vectors using two SB_RAM40_4K blocks reconfigured in native x16 mode, forming a dual-bank system with 1 kB total capacity (2 × 256 × 16-bit). This allows packing of two 8-bit inputs per address, while a lightweight Read Mux in the controller supports byte-level access.

Finally, Ping-Pong Buffers manage inter-layer data transfer as zero-copy intermediate scratchpads. Two 16k × 16-bit SPRAM blocks operate as 'Ping' and 'Pong' buffers in a double-buffering scheme: Layer N reads execution data from the 'Ping' buffer while the preceding Layer $N-1$ (or the Requantizer) simultaneously writes results into the 'Pong' buffer. A global toggle signal swaps buffer ownership upon layer completion, eliminating memcpy overhead and ensuring continuous pipeline operation.

B. Compute Subsystem

1) DSP Systolic Cluster: The core of the accelerator is a one-dimensional systolic array consisting of six iCE40 SB_MAC16 DSP blocks arranged in a pipelined sequence. In this design, the input feature map (X) enters the first DSP unit and propagates through the array via six pipeline registers (X[0..5]), establishing a systolic data flow that reduces register fanout and balances routing delays across the FPGA fabric. In contrast, the filter weights and biases are simultaneously broadcast to all units. This architecture allows the cluster to perform six MAC operations per clock cycle, achieving a theoretical peak throughput of 144 million MACs (MMACs) per second (at 24 MHz).

The architecture is layer-agnostic and supports variable kernel sizes (K) as well as input and output channel depths (C_{in}, C_{out}) through runtime configuration registers. For scaling, large layers are time-multiplexed across the fixed-size array. The control finite state machine (FSM) iterates over the kernel weights ($k = 0..K$), input($c_in = 0..C_{in}$) and output ($c_out = 0..C_{out}$) channels, accumulating partial sums in the DSP's 32-bit registers before producing the final output. On fully connected layers, the final classification layer (L5) is implemented as a special case of a 1×1 convolution. Because the preceding global average pooling (GAP) reduces the spatial dimension to one, the fully connected operation is reduced to a vector-matrix multiplication. The systolic array executes this operation efficiently, eliminating the need for a separate dense hardware unit. The current implementation is strictly optimized for stride-one convolutions. This approach simplifies the systolic dataflow, as the pipeline registers shift by one position per cycle, precisely matching the sliding-window requirement. Supporting stride values greater than one would necessitate complex skip-connection routing or stall cycles to align the data.

2) Pooling Unit: Positioned after the DSP cluster, the Pooling Unit performs dimensionality reduction with layer-specific behaviour. For standard convolutional layers (L0-L2), it applies 2×1 Max Pooling by comparing adjacent systolic outputs (i.e. Y_n and Y_{n-1}) to retain dominant features. In the final CNN layer (L3), it applies GAP, where the input data is first scaled using arithmetic right shift (as input feature depth is fixed 64) and acts as a persistent accumulator, summing the entire feature map channel into a single 32-bit value. Once the entire input channel is processed, it flushes the accumulator. For the fully connected layer (L4), the Bypass Mode disables all reduction logic, allowing the raw accumulation result from the systolic array to flow directly to the Requantizer. This is essential for the final classification operation, where 1×1 convolution results must be preserved exactly.

3) Requant and Activation Engine: As discussed in Section V-A, for QAT models, the pooled output (32-bit) must be scaled with a multiplier (32-bit) to retain precision before activation, producing a 64-bit intermediate value. However, iCE40 SB_MAC16 DSP blocks natively support only 16×16 multiplication. To enable 32×32 signed multiplication, the proposed architecture employs a dedicated mul64signed unit based on a sum-of-products decomposition. Each 32-bit operand is split into 16-bit upper (H) and lower (L) halves, and the full 64-bit product is accumulated over four clock cycles, producing one result every four cycles.

Following multiplication, the result is accumulated with a zero scale and passed through an activation unit. For convolution layers (L0-L3), a ReLU-style activation is implemented via unsigned saturation to the range $[0, 255]$, which re-quantizes the output to 8-bit. For the final fully connected layer (L4), the activation stage is bypassed, allowing signed 32-bit rounded logits to be forwarded directly to the host processor.

4) Result Packer: The final stage is the Result Packer, which adapts the 8-bit compute stream to the 16-bit memory interface. Incoming bytes are buffered in a temporary register and two consecutive results are concatenated into a single 16-bit word. This packing enables efficient utilization of the single-port SPRAMs (ping-pong buffers), reducing write operations by half and maximizing available memory bandwidth for subsequent layer readout.

C. Control Subsystem

The control subsystem, shown in Figure 4, is implemented as a hierarchical FSM that partitions system control across three levels: a top-level arbitration controller, a layer-level inference sequencer, and a memory-compute controller. This hierarchy decouples external communication and system orchestration from layer scheduling and arithmetic dataflow, allowing each component to be optimized independently. As a result, the arithmetic core remains agnostic to the user interface protocol while maintaining deterministic execution and low control overhead. At the highest level, the arbitration controller manages system operation and UART-based interaction, coordinating weight loading, verification, and inference by dynamically arbitrating access to shared memory and compute resources.

TABLE III: Layer-wise Cycle Breakdown, Compute Efficiency, and Primary Bottlenecks

Layer	Type	In (depth)	Out (depth)	$C_{\text{in}} \rightarrow C_{\text{out}}$	K	Prime	Compute	Requant / Pack	Array Eff.	Sys. Eff.	Primary Bottleneck
L0	Conv1D	512	256	$1 \rightarrow 16$	9	9,632	12,384	24,768	56%	26%	Requantization
L1	Conv1D	256	128	$16 \rightarrow 32$	9	154,112	198,144	24,768	56%	52%	Balanced
L2	Conv1D	128	64	$32 \rightarrow 64$	9	315,392	405,504	25,344	56%	53%	Balanced
L3	Conv1D	64	1	$64 \rightarrow 128$	5	630,784	450,560	768	41%	41%	Priming
L4	FC	1	1	$128 \rightarrow 3$	1	2,688	384	18	12.5%	12%	Priming
Total						**1,112,608**	**1,066,976**	**75,666**	-	-	-

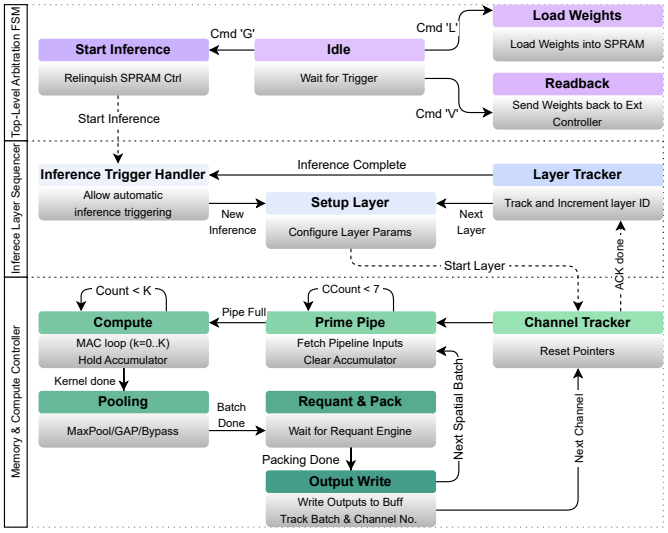

Fig. 4: Top-level arbitration and control flow. The controller decodes UART commands to switch between weight loading, memory verification, and inference modes. During loading and verification, it arbitrates exclusive access to the Weight Memory, while a run command transfers control to the inference layer sequencer for autonomous CNN execution.

Once execution is initiated, control is transferred to the *Layer Sequencer*, a dedicated FSM that encodes the neural network topology directly in hardware. Rather than relying on an instruction fetch mechanism, the sequencer traverses the network layers using a fixed FSM schedule, eliminating instruction memory overhead. For each layer, it programs the memory and compute controller with layer-specific parameters, asserts a start signal to launch computation, and waits for a completion flag. Upon layer completion, the sequencer automatically advances to the next stage and reconfigures the controller as needed, enabling fully autonomous end-to-end inference once triggered.

At the lowest level of the hierarchy, the *Memory and Compute Controller* governs fine-grained data movement and computation for a single layer. It implements a four-level nested loop structure in hardware to maximize data reuse and sustain high utilization of the systolic compute array. Output channels are processed in an outer loop, while the input signal is consumed in fixed six-element spatial batches. Each batch begins with a prime phase that pre-fetches input samples and fills the DSP pipeline, hiding cascade latency, as detailed in Figure 4. This is followed by a compute phase that iterates exclusively over the kernel depth, broadcasting weights and accumulating partial sums across all output channels. Upon completion of the compute phase, pooling, re-quantization, and result packing are applied, after which the controller advances to the next spatial batch. Valid outputs are packed into 16-bit words for efficient memory writes.

D. Computational Cycle Analysis

To evaluate the performance of the accelerator, we analyze the total cycle count per layer and identify the primary sources of overhead. The execution time of a layer can be decomposed into three major components:

$$T_{\text{Total}} = T_{\text{Prime}} + T_{\text{Compute}} + T_{\text{Requant}}, \tag{1}$$

where T_{Prime} represents the pipeline priming overhead incurred when switching between input channels, T_{Compute} corresponds to the convolution computation, and T_{Requant} accounts for the cycles spent on re-quantization and packing. Specifically, the timing contributions are given by:

$$\begin{cases} T_{\text{Prime}} = C_{\text{out}} \cdot N_{\text{batches}} \cdot C_{\text{in}} \cdot 7 \\ T_{\text{Compute}} = C_{\text{out}} \cdot N_{\text{batches}} \cdot C_{\text{in}} \cdot K \\ T_{\text{Requant}} = N_{\text{outputs}} \cdot 9 \end{cases} \tag{2}$$

where C_{in} and C_{out} denote the number of input and output channels, K is the kernel size, $N_{\text{batches}} = \lceil W_{\text{in}}/6 \rceil$ is the number of spatial batches, and N_{outputs} is the total number of outputs per layer. The 7-cycle term in T_{Prime} captures the fixed latency to fill the systolic pipeline, while the 9-cycle term in T_{Requant} includes six cycles for the serial multiplier and three cycles for FSM overhead and packing.

We define two efficiency metrics to quantify hardware utilization. The array efficiency measures the fraction of cycles performing arithmetic within the systolic array, ignoring re-quantization and packing:

$$\eta_{\text{array}} = \frac{T_{\text{Compute}}}{T_{\text{Prime}} + T_{\text{Compute}}} = \frac{K}{K+7} \tag{3}$$

whereas the system efficiency captures useful compute relative to the total execution time including re-quantization:

$$\eta_{\text{sys}} = \frac{T_{\text{Compute}}}{T_{\text{Total}}} \tag{4}$$

A layer-wise analysis reveals that early convolutional layers with large kernels achieve moderate array efficiency (about $56\,\%$) but exhibit varying system efficiency due to the relative cost of re-quantization. For instance, Layer 0, with a 1D convolution producing 4,096 outputs, is dominated by the re-quantization stage, resulting in a system efficiency of approximately $26\,\%$, despite a pipeline utilization of $56\,\%$. In subsequent layers (L1 and L2), the computational workload increases and the re-quantization cost becomes negligible, leading to higher system efficiency (about $52\,\%$ to $53\,\%$). Later layers, including L3 with a smaller kernel and wider channels, are dominated by pipeline priming, while the fully connected Layer L4 represents the worst-case scenario for the array, where $K = 1$ limits pipeline utilization to only $12.5\,\%$, yielding similarly low system efficiency.

Table III summarizes the cycle breakdown and efficiency per layer. The analysis highlights two primary performance bottlenecks: output-bound layers (e.g., L0), limited by the serial multiplier in the re-quantization stage, and latency-bound layers (e.g., L3 and L4), limited by pipeline priming in the systolic array. Early convolutional layers are compute-efficient, while deeper and pointwise layers are increasingly dominated by pipeline overhead, emphasizing the impact of kernel size and channel width on array utilization and overall system performance.

Overall, the network requires approximately 2.26M cycles per inference, corresponding to about $94\,\mathrm{ms}$ at a $24\,\mathrm{MHz}$ clock frequency, yielding an effective throughput of $\sim$10.6 FPS. This dual-bottleneck behavior provides clear guidance for optimization: either increasing the parallelism in the re-quantization stage to mitigate output-bound layers or reducing pipeline depth and kernel imbalance to improve efficiency in latency-bound layers.

VII. Evaluation

This section presents a comprehensive evaluation of the proposed system, covering both model-level classification performance and hardware-level inference efficiency. The analysis first examines the accuracy, robustness, and calibration of the CNN on SCG data using qualitative examples and quantitative window-level metrics. Inference performance of the FPGA implementation is then evaluated with respect to resource utilization, latency, power consumption, and energy efficiency, with comparisons to a low-power MCU baseline.

A. Classification Performance

The proposed model demonstrates strong window-level discrimination for rare SCG events. Figure 5 shows a qualitative comparison between ground-truth and CNN-predicted labels on a representative SCG segment, illustrating good temporal alignment of systolic and diastolic events.

As shown in Figure 6, quantitative performance is maintained under severe class imbalance, with both systolic and diastolic events exhibiting high separability in precision–recall space. Per-class metrics further confirm consistent behavior across all labels, indicating that performance is not dominated by the background class.

To reduce ambiguity near event boundaries, background windows were sampled using a temporal exclusion constraint, enforcing a minimum gap of $0.05\,\mathrm{s}$ from any annotated systolic or diastolic event.

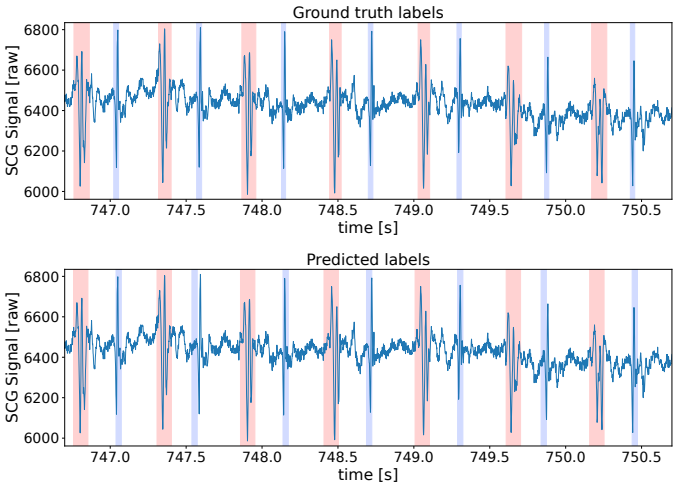

Fig. 5: Comparison of ground-truth (top) and CNN-predicted (bottom) labels for the same SCG segment. Shaded regions indicate systolic (red) and diastolic (blue) intervals over the raw signal (small excerpt of 7 heartbeats).

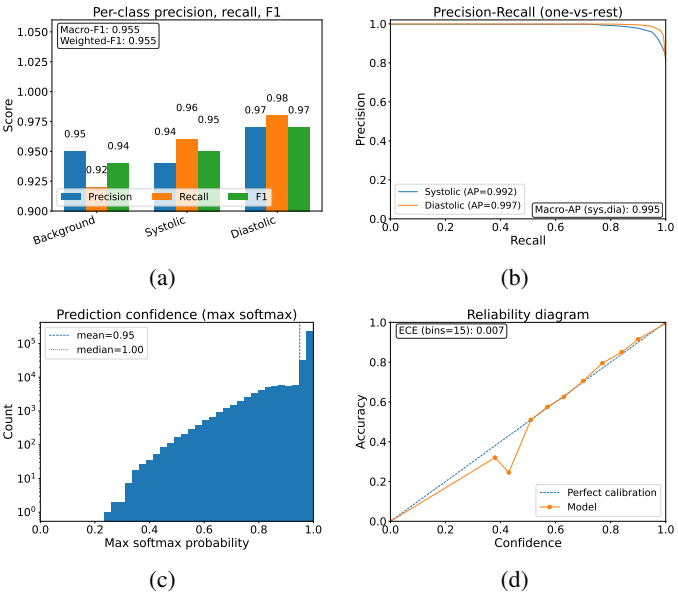

Fig. 6: Combined window-level evaluation of the SCG CNN: (a) per-class precision, recall, and F1 with macro and weighted F1; (b) one-vs-rest precision-recall curves for systolic and diastolic events with average precision (AP); (c) histogram of maximum softmax confidence; (d) reliability diagram with expected calibration error (ECE).

TABLE IV: iCE40UP5K Resource Utilization

Resource	Used	Total	Util.	Description
LUTs	∼2861	5280	54%	FSMs, address calculation, multiplexers
SPRAM[a]	4	4	100%	Weight storage and feature maps
BRAM[b]	6	30	33%	Input buffers and bias ROM
DSP Blocks	7	8	87%	Compute DSP Cluster and Requant logic
IO Pins	14	96	14%	UART interface, debug bus, LEDs

[a] (256 kbit) [b] (4 kbit)

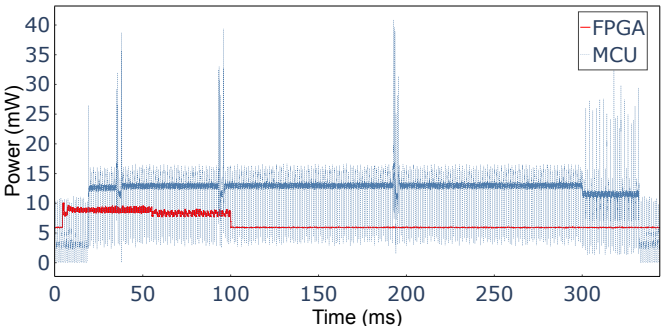

Fig. 7: Inference Power Consumption comparison between iCE40UP5K FPGA and nRF52840 MCU

B. Inference Performance Evaluation

The FPGA implementation demonstrates efficient logic utilization, requiring only 2,861 Look-up tables (LUTs) and 7 DSPs, as shown in Table IV. These results indicate that the neural network maps effectively onto the FPGA fabric and preserves substantial capacity for additional features or future scaling. In comparison to the energy-efficient nRF52840 MCU, which operates at 64 MHz and uses CMSIS-NN to accelerate the LiteRT neural network inference with ARM DSP cores, inference on the iCE40UP5K FPGA (operating at 24 MHz) consumes only 819 µJ per inference. This energy consumption is more than four times lower than that of the MCU, and inference completes in 95.5 ms, making it approximately three times faster (see Table V). The architecture achieves a throughput of 134 MOps/s while maintaining a low average power consumption of 8.55 mW, as shown in Figure 7. These findings suggest that the architecture is suitable for ultra-low-power, resource-constrained FPGAs and is scalable to larger devices.

VIII. Conclusion

In this paper, we demonstrate a complete pipeline for real-time cardiac feature extraction (systolic-diastolic segmentation) for a space-grade SCG wearable, encompassing data acquisition, high-fidelity semi-automated labeling, CNN training, and energy-efficient inference on ULP FPGAs. Quantization-aware training enables CNN models to achieve over 98 % validation accuracy, while the systolic-array FPGA implementation allows efficient on-device inference, completing classification in about 95.5 ms using only 2,861 LUTs and 7 DSPs (∼10.6 FPS) at 8.55 mW, which is over four times more energy-efficient than state-of-the-art low-power MCUs. Layerwise inference-cycle profiling identifies optimization opportunities in re-quantization and pipeline balance. This full-cycle system is robust to sensor variability and enables fully autonomous, energy-efficient cardiovascular monitoring for astronauts, with broader implications for wearable and smart health technologies.

TABLE V: Inference Efficiency and Performance Comparison Between FPGA and MCU Implementations

Metric	iCE40UP5K (FPGA)	nRF52840 (MCU)
Inference Time [ms]	95.5	314.9
Avg. Inference Power [mW]	8.55	11.4
Inference Energy [μJ]	819.1	3589.9
Throughput [MMAC/s]	67.0	–
Throughput [MOps/s]	134.0	–

Acknowledgement

This work has received funding from the Federal Ministry of Research, Technology and Space (BMFTR) supported by German Aerospace Center (DLR) in the projects 'AuRelia' and 'SArES' under the grant numbers 50RP2350 and 50WB2421A, respectively. Also partially funded by the German Research Foundation (DFG) in the project 'KORVEKSiS' under the grant number 542151450. The authors would like to express their sincere gratitude to Prof. Dr. Maximilian Kiener for facilitating the ethical approval of this study.

References

[1] U. Kulau, J. Rust, D. Szafranski, M. Drobczyk, and U.-V. Albrecht, "A differential BCG sensor system for long term health monitoring experiment on the ISS," in *2022 18th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2022, pp. 85–92.

[2] M. Drobczyk, A. Lübken, C. Strowik, U. Kulau, J. Rust, J. Beringer, and U.-V. Albrecht, "Wireless Compose-2: A wireless communication network with a ballistocardiography smart-shirt experiment in the ISS Columbus module," in *2021 IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*. IEEE, 2021, pp. 103–108.

[3] M. Drobczyk, A. Lübken, C. Strowik, U.-V. Albrecht, J. Rust, J. Beringer, and U. Kulau, "A wireless communication network with a ballistocardiography experiment on the ISS: Scenario, components and preflight demonstration," *IEEE Journal of Radio Frequency Identification*, vol. 6, pp. 258–268, 2022.

[4] J. Wang, S. M. Nouraie, N. J. Kelly, and S. Y. Chan, "Deep learning predicts cardiac output from seismocardiographic signals in heart failure," *The American journal of cardiology*, 2025.

[5] N. Konnova, M. Basarab, and V. Khaperskaya, "Application of machine learning algorithms for SCG signal classification," in *2020 International Conference on Image, Video Processing and Artificial Intelligence*, vol. 11584. SPIE, 2020, pp. 374–379.

[6] K. M. A. Rahman, T. Dirkes, B. Delfs, V. Wyrwoll, and U. Kulau, "ISFD: Efficient and fault-tolerant in-system-failure-detection for LP FPGA-based smart-sensors in space expeditions," in *2024 20th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*. IEEE, 2024, pp. 74–83.

[7] F. E. Garrett-Bakelman, M. Darshi, S. J. Green, R. C. Gur, L. Lin, B. R. Macias, M. J. McKenna, C. Meydan, T. Mishra, J. Nasrini *et al.*, "The NASA twins study: A multidimensional analysis of a year-long human spaceflight," *Science*, vol. 364, no. 6436, p. eaau8650, 2019.

[8] O. T. Inan, P.-F. Migeotte, K.-S. Park, M. Etemadi, K. Tavakolian, R. Casanella, J. Zanetti, J. Tank, I. Funtova, G. K. Prisk *et al.*, "Ballistocardiography and seismocardiography: A review of recent advances," *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 4, pp. 1414–1427, 2015.

[9] A. Taebi, B. E. Solar, A. J. Bomar, R. H. Sandler, and H. A. Mansy, "Recent advances in seismocardiography," *Journal of Biomedical Engineering and Medical Imaging*, vol. 6, no. 1, pp. 1–12, 2019.

[10] A. M. Weissler, W. S. Harris, and C. D. Schoenfeld, "Systolic time intervals in heart failure in man," *Circulation*, vol. 37, no. 2, pp. 149–159, 1968.

[11] T. Bombardini, V. Gemignani, E. Bianchini, L. Furlan, L. Pratali, and E. Picano, "Diastolic time – stress for the heart: A new index of myocardial perfusion," *European Heart Journal Supplements*, vol. 11, no. Supplement E, pp. E12–E18, 2009.

[12] J. Qiu, J. Wang, S. Yao, K. Guo, B. Li, E. Zhou, J. Yu, T. Tang, N. Xu, S. Song *et al.*, "Going deeper with embedded FPGA platform for convolutional neural network," in *Proceedings of the 2016 ACM/SIGDA international symposium on field-programmable gate arrays*, 2016, pp. 26–35.

[13] X. Zhang, J. Wang, C. Zhu, Y. Lin, J. Xiong, W.-m. Hwu, and D. Chen, "DNNBuilder: An automated tool for building high-performance DNN hardware accelerators for FPGAs," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2018, pp. 1–8.

[14] R. Xu, S. Ma, Y. Guo, and D. Li, "A survey of design and optimization for systolic array-based DNN accelerators," *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–37, 2023.

[15] X. Wei, C. H. Yu, P. Zhang, Y. Chen, Y. Wang, H. Hu, Y. Liang, and J. Cong, "Automated systolic array architecture synthesis for high throughput CNN inference on FPGAs," in *Proceedings of the 54th Annual Design Automation Conference 2017*, 2017, pp. 1–6.

[16] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.

[17] D. Khudia, J. Huang, P. Basu, S. Deng, H. Liu, J. Park, and M. Smelyanskiy, "FBGEMM: Enabling high-performance low-precision deep learning inference," *arXiv preprint arXiv:2101.05615*, 2021.

[18] H.-T. Kung, *Why systolic architecture?* Design Research Center, Carnegie-Mellon University Pittsburgh, PA, USA, 1982.

[19] A. Samajdar, J. M. Joseph, Y. Zhu, P. Whatmough, M. Mattina, and T. Krishna, "A systematic methodology for characterizing scalability of DNN accelerators using SCALE-Sim," in *2020 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 2020, pp. 58–68.

[20] H. Genc, S. Kim, A. Amid, A. Haj-Ali, V. Iyer, P. Prakash, J. Zhao, D. Grubb, H. Liew, H. Mao *et al.*, "Gemmini: Enabling systematic deep-learning architecture evaluation via full-stack integration," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 769–774.

[21] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.

[22] Lattice Semiconductor Corporation, *iCE40 UltraPlus Family Data Sheet*, Lattice Semiconductor, Hillsboro, OR, USA, 2025, document Number: DS-02008, Version 2.4. [Online]. Available: https://www.latticesemi.com/

[23] U.-V. Albrecht, A. Mielitz, K. M. A. Rahman, U. Kulau *et al.*, "Identifying gravity-related artifacts on ballistocardiography signals by comparing weightlessness and normal gravity recordings (ARTIFACTS): Protocol for an observational study," *JMIR Research Protocols*, vol. 13, no. 1, p. e63306, 2024.

[24] K. Rahman, U.-V. Albrecht, and U. Kulau, "Wavelet-driven denoising and cross-axis fusion for automated SCG systolic/diastolic window extraction," in *Proceedings of the 22nd GI/ITG KuVS Fachgespräch Sensornetze (FGSN)*, 2025.

[25] M. Tkachenko, M. Malyuk, A. Holmanyuk, and N. Liubimov, "Label Studio: Data labeling software," 2020-2025, open source software available from https://github.com/HumanSignal/label-studio. [Online]. Available: https://github.com/HumanSignal/label-studio

[26] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019. [Online]. Available: https://arxiv.org/abs/1711.05101