*electronics*

MDPI

*Article*

# Hardware Aspects of Machine Learning-Based Cardiac Fibrillation Diagnosis

**Ioannis Kouretas [1,*], Anastasios G. Skrivanos [2], Nikos C. Sagias [2] and Kostas P. Peppas [2]**

[1] Department of Electrical and Computer Engineering, University of Patras, 26500 Patras, Greece
[2] Department of Informatics and Communications, University of Peloponnese, 22131 Tripoli, Greece; a.skrivanos@go.uop.gr (A.G.S.); nsagias@uop.gr (N.C.S.); peppas@uop.gr (K.P.P.)
* Correspondence: kouretas@upatras.gr

**Abstract**

This paper presents the hardware aspects of a Hjorth-parameter deep neural network (DNN)-based cardiac fibrillation diagnosis pipeline targeting low-power Internet of Medical Things (IoMT) devices and ASIC implementations. Building on earlier edge-to-cloud studies where Hjorth activity, mobility, and complexity were computed in software on microcontrollers and classified by a floating-point DNN, we introduce a fully synthesizable fixed-point hardware module that computes these parameters in real time, together with a quantized neural network (QNN) operating directly on the resulting fixed-point features. The Hjorth block includes derivative generation, accumulator banks, variance computation, and hardware divider and square-root units. Using the Shandong Provincial Hospital Database (SPHD) as in our previous work, we evaluate the impact of end-to-end fixed-point quantization on the AF-related arrhythmia detection performance for bit widths between 6 and 16 bits. For 10–12-bit configurations, the quantized pipeline achieves accuracy of approximately 93.7% and an area under the ROC curve (AUC) above 0.97, closely matching the floating-point baseline while significantly reducing the arithmetic complexity and memory footprint. ASIC synthesis in a 28 nm CMOS standard-cell library shows that the complete atrial detector core, integrating the Hjorth extractor and the hardware fully connected QNN, occupies on the order of $2 \times 10^4$ $\mu m^2$ and dissipates about 3 mW, with a critical-path delay of 7.08 ns. For the optimal 10–12-bit operating points, the synthesized core achieves per-inference energy in the range of 18.6–19.0 nJ (18,600–19,000 pJ) per classification, confirming its suitability for integration into wearable and IoMT ECG monitoring nodes. These results demonstrate that co-designed fixed-point Hjorth hardware and quantized DNNs can deliver a favorable trade-off between diagnostic performance, area, power, latency, and per-inference energy compared with existing MCU-, FPGA-, and ASIC-based ECG classifiers.

**Keywords:** atrial fibrillation; ASIC design; deep learning



Academic Editor: John A. Kalomiros

Received: 8 July 2026
Revised: 11 August 2026
Accepted: 13 August 2026
Published: 17 August 2026



## 1. Introduction

Atrial fibrillation (AF) is the most common sustained cardiac arrhythmia and a major contributor to adverse cardiovascular outcomes, including stroke, heart failure, and increased mortality [1]. Early detection and timely intervention are therefore critical to reduce the individual and societal burden associated with AF and related rhythm disorders.

In recent years, the proliferation of wearable and portable electrocardiogram (ECG) devices and other Internet of Medical Things (IoMT) platforms has created an opportunity

for continuous, out-of-clinic rhythm monitoring, but it has also raised important challenges regarding how to process and interpret large volumes of physiological data under strict constraints on energy, memory, cost, and connectivity [2,3]. IoMT-based remote monitoring has already been demonstrated in other cardiovascular contexts—for example, in fetal heart rate monitoring systems that combine mobile wearables with ESP8266-class wireless nodes and cloud backends, as in Skrivanos et al. [4].

Machine learning (ML) and deep learning (DL) methods have significantly improved the automatic analysis of ECG signals, achieving high performance in AF detection when combined with appropriate feature representations and evaluation protocols [5,6]. Edge-AI studies have demonstrated that AF classification can be executed on low-power embedded platforms, but they also highlight stringent limitations in on-device memory, compute, and energy budgets that often necessitate compact models and carefully optimized inference pipelines [7–9]. Purely cloud-based approaches, on the other hand, can exploit abundant computational resources but may introduce latency, bandwidth overhead, and privacy concerns that are undesirable for time-sensitive healthcare applications [2,3]. These considerations motivate hybrid architectures that distribute computation between resource-constrained edge nodes and cloud services in a task-aware manner.

On the classification side, compact deep neural networks operating on low-dimensional feature vectors provide a promising compromise between accuracy and deployability for AF detection on IoMT devices. In our earlier work, we proposed a Hjorth-parameter and DNN-based AF termination predictor on the Shandong Provincial Hospital Database (SPHD), achieving high diagnostic performance with inference executed on commercial microcontrollers and GPUs [10]. A follow-up ESP8266-based edge-to-cloud system demonstrated that Hjorth activity, mobility, and complexity computed in software on a low-cost microcontroller can support AF-related arrhythmia detection, with feature vectors transmitted to a cloud-hosted DNN classifier [11]. While these studies confirmed the suitability of Hjorth parameters and compact DNNs for embedded AF monitoring, they also revealed limitations in latency, energy efficiency, and scalability when feature extraction is implemented purely in software, particularly at higher sampling rates or in multi-channel wearable configurations.

To address these constraints, this work focuses on the hardware aspects of a Hjorth-parameter and DNN-based cardiac fibrillation diagnosis pipeline. We propose a fully synthesizable, parameterizable fixed-point hardware architecture for computing Hjorth activity, mobility, and complexity over ECG windows, together with a quantized neural network (QNN) that operates directly on the resulting fixed-point features. The hardware Hjorth module is designed as a reusable building block for embedded ECG processing pipelines, while the QNN is tailored to fit within the memory and computational envelope of an ESP8266-class edge node and to map efficiently onto a standard-cell ASIC flow. Using the same SPHD dataset and windowing protocol as in our previous work, we systematically evaluate the impact of fixed-point quantization on arrhythmia detection performance and report both algorithmic metrics (accuracy, F1-score, AUC) and hardware-oriented metrics (area, power, timing, and per-classification energy) obtained from synthesis in a 28 nm CMOS technology.

The main contributions of this paper are as follows:

- We conduct a systematic word-length and scaling analysis (6 to 16 bits) to evaluate the impact of end-to-end fixed-point quantization on Hjorth parameters and QNN classification accuracy, identifying optimal 10–12-bit operating points.
- We report post-synthesis ASIC implementation results (area, delay, power, and per-classification energy) in a 28 nm CMOS technology, demonstrating that the integrated

core occupies on the order of $2 \times 10^4$ $\mu m^2$, consumes less than 3.2 mW, and achieves per-inference energy in the 18.6–19.0 nJ range for the optimal 10–12-bit configurations.
- We provide a systematic bit-width and scaling analysis for ECG signals in the millivolt range, deriving internal word lengths that prevent overflow while offering sufficient precision for subsequent classification.
- We design and implement a quantized DNN for cardiac fibrillation diagnosis that is compatible with the hardware Hjorth feature extractor.
- We present an experimental evaluation on a public AF dataset, comparing floating-point and quantized models, reporting algorithmic metrics (accuracy, F1-score, AUC, confusion matrices) as well as hardware-oriented metrics (latency, area, and power), and discussing the trade-offs between performance, complexity, and deployability.

The remainder of the paper is organized as follows. Section 2 reviews related work on AF detection, Hjorth parameters, edge and IoMT architectures, model quantization for ECG analysis, and embedded hardware implementations. Section 3 details the fixed-point Hjorth hardware architecture and word-length design, while Section 4 presents the quantized neural network and its embedded and ASIC-oriented implementation. Section 5 reports the experimental results and discusses the observed trade-offs, and Section 6 concludes the paper and outlines directions for future work.

## 2. Background and Related Work

### *2.1. Atrial Fibrillation Detection from ECG*

AF detection has long relied on the manual inspection of ECG recordings by trained clinicians, focusing on the absence of P waves, irregular RR intervals, and fibrillatory baseline activity. Recent advances in artificial intelligence (AI) have enabled the automatic analysis of both single-lead and multi-lead ECG signals, leading to improved sensitivity and specificity for AF screening in ambulatory and wearable scenarios [1,12]. In particular, AI models applied to sinus-rhythm ECGs have shown the ability to detect patients at risk of paroxysmal or incident AF, broadening AF detection beyond episodes captured during overt arrhythmia [6,12]. Comprehensive recent reviews summarize how machine learning and deep learning approaches can support AF detection, risk stratification, and management across a variety of clinical and ambulatory settings [5,6].

A wide range of feature representations has been explored for AF detection, including time-domain descriptors, frequency-domain parameters, and morphology-based metrics derived from QRS complexes and atrial activity [5,10]. Traditional approaches often rely on heart rate variability indices, P-R interval statistics, and spectral markers, whereas modern deep learning architectures can operate either on raw ECG segments, on time–frequency images, or on compact feature vectors [5,10]. Hjorth parameters constitute an attractive time-domain feature family because they capture signal variance, mobility, and complexity using low-order statistics and are therefore compatible with both classical machine learning models and lightweight embedded implementations [13].

### *2.2. Hjorth Parameters and Fixed-Point Implementations*

Hjorth activity, mobility, and complexity were originally introduced as descriptors of stochastic processes and have since been widely applied to biomedical signals, including EEG and ECG, for classification and monitoring tasks [13]. For discrete-time signals, these parameters can be expressed in terms of the variance of the signal and the variance of its first and second discrete derivatives, making them amenable to running-sum implementations over finite windows [13]. This property is particularly appealing for resource-constrained hardware, because it allows the feature extractor to be realized using adders, subtractors, and accumulators, with divisions simplified by appropriate choices of window length.

The fixed-point realization of Hjorth parameters requires careful selection of the input format, internal word lengths, and scaling factors in order to avoid overflow and maintain sufficient numerical precision. In the context of ECG, where amplitudes are typically on the order of millivolts and derivatives can amplify noise, designers must dimension squared terms and accumulators to accommodate worst-case scenarios over the chosen window size. Parameterizable architectures that expose word-length and window-length parameters at synthesis time are therefore attractive, as they enable the design-space exploration of the trade-offs between area, latency, and estimation accuracy in different application settings. Although early works (e.g., ref. [13]) demonstrated the conceptual feasibility of Hjorth parameter computation for digital signal processing, there is still a lack of fully integrated, synthesizable hardware architectures that combine fixed-point Hjorth feature extractors with multiplierless quantized neural networks in modern ASIC standard-cell flows. This work addresses this gap by co-designing the feature extractor and quantized classifier into a single monolithic core.

### *2.3. Edge and IoMT Architectures for Cardiac Monitoring*

The proliferation of Internet of Medical Things (IoMT) platforms has motivated a variety of system architectures that distribute ECG acquisition, preprocessing, and analysis between edge devices, fog nodes, and cloud services [2,3]. Simple telemetry systems stream raw or lightly compressed ECG data to a remote server for offline or near-real-time analysis, but such designs can incur high bandwidth consumption and may not scale well in large deployments [2,6]. More advanced solutions perform part of the processing pipeline on embedded microcontrollers or wearable devices, extracting compact feature vectors that are transmitted to an upstream backend, thereby reducing the communication overhead and enabling local decision support [2,3].

Several recent works have specifically targeted AF detection on low-power microcontrollers and wearable platforms, proposing energy-efficient inference pipelines and compact models tailored to resource constraints [7–9]. For example, embedded-edge AF detectors based on heart rate variability features and compact neural networks have demonstrated high accuracy while keeping the energy per inference within the millijoule range, enabling long-term battery-powered screening [7]. Other studies have investigated real-time AF detection on wearable devices using lightweight models and heartbeat interval features, confirming that clinically meaningful monitoring is feasible on severely resource-constrained hardware [8,9]. Hybrid fog–edge architectures further extend this idea by dynamically offloading computation between local and remote resources based on latency, reliability, and security requirements, offering a promising path toward scalable, resilient cardiac monitoring services in IoMT environments [3,6].

In parallel with digital edge-AI architectures, recent breakthroughs in flexible electronics and soft bio-sensing platforms have revolutionized out-of-clinic physiological monitoring [14–16]. Conformal, skin-compatible sensors enable the continuous, motion-resilient acquisition of bio-potential signals (such as single- and multi-lead ECG) with minimal contact impedance. However, streaming raw physiological data from flexible skin patches to remote servers incurs considerable wireless transmission overhead, rapidly draining low-capacity flexible batteries. Integrating compact, custom edge-AI ASIC cores directly into flexible bio-electronic patches provides a compelling solution, enabling local, on-skin feature extraction and decision-making while transmitting only low-bandwidth diagnostic flags.

### 2.4. Model Compression and Quantization for ECG Analysis

Model compression techniques, such as pruning, weight sharing, and low-rank factorization, have been extensively investigated to reduce the computational and memory footprints of deep networks, including those applied to biomedical signals [5]. Among these techniques, fixed-point quantization has emerged as a particularly effective approach for embedded inference, where weights and activations are represented with reduced bit widths while preserving most of the original model accuracy [9]. In the ECG domain, 8-bit and sub-8-bit quantized networks have been shown to maintain near-floating-point performance while significantly lowering memory usage and the compute cost, enabling deployment on microcontroller-class devices and small-form-factor ECG recorders [7,9].

For AF detection specifically, quantized neural networks (QNNs) can be combined with compact feature representations, such as Hjorth parameters or heartbeat interval statistics, to minimize the overall resource requirements of the pipeline [7,8]. However, the interaction between feature quantization, the internal word lengths in the preprocessing hardware, and the final classification performance remains an active area of research. Recent reviews emphasize that carefully co-designing signal representations, model architectures, and deployment platforms is crucial for translating AF AI models from proof-of-concept prototypes into robust, real-world monitoring systems [5,6].

### 2.5. Embedded and Hardware-Oriented Implementations

Beyond algorithmic advances, several works have examined embedded and hardware-oriented implementations of ECG and AF classifiers. Kachuee et al. [17] proposed a 1D CNN for arrhythmia classification on the MIT-BIH database, evaluated primarily on GPUs, while Isin and Ozdalili [18] explored AlexNet-based transfer learning for ECG classification, also targeting GPU platforms. Oh et al. [19] combined convolutional and LSTM layers to capture temporal dependencies in ECG signals, achieving high accuracy on MIT-BIH but focusing on software inference. Hannun et al. [20] developed a deep ResNet for AF and arrhythmia detection on a large clinical dataset, demonstrating strong diagnostic performance but again relying on data center-class hardware.

Quantization-oriented studies have investigated reduced-precision deployments. Pu et al. [21] applied quantization-aware training to binary CNNs for MIT-BIH arrhythmias, achieving near-floating-point accuracy in software. Ribeiro et al. [22] reported an INT8 post-training quantized 1D CNN running on an ARM Cortex-A55, with no accuracy loss relative to FP32. These works illustrate that low-bit-width representations can preserve the classification quality on general-purpose processors.

Closer to hardware accelerators, Chen et al. [23] realized a CNN-based classifier as an ASIC in a 180 nm CMOS technology, reporting 96.3% accuracy on MIT-BIH together with milliwatt-level power consumption. SparrowSNN [24] implemented a digital spiking neural network ASIC for arrhythmia detection, achieving 98.29% accuracy with microwatt-level power, while Rana et al. [25] mapped a related SNN architecture onto an iCE40 FPGA, reporting 98.4% accuracy and milliwatt-range power. The DARE Lab Inception-ResNeXt accelerator [26] targeted Cyclone-class FPGAs with quantized CNNs, achieving accuracies above 92% and up to 99.5% on MIT-BIH at hundreds of milliwatts of power.

Two recent works by the present authors form the immediate methodological precursors of this paper. Skrivanos et al. [10] introduced a Hjorth-parameter and DNN-based AF termination predictor on the SPHD database, achieving 94.8% accuracy and an AUC of 0.9479, with energy and latency evaluated on commercial 64-bit microcontrollers and a GPU. A follow-up PACET study [11] implemented an edge-to-cloud arrhythmia detection pipeline in which Hjorth features were computed on an ESP8266 node and classified by a cloud-hosted DNN, reaching 95% accuracy and AUC 0.95

on Shandong Provincial Hospital Database (SPHD) (the data are publicly available at https://data.mendeley.com/datasets/khxprpdt3z (accessed on 12 August 2026) while demonstrating feasibility on low-cost consumer hardware.

In addition to traditional microcontroller and GPU implementations, recent advances in flexible bio-electronics and wearable edge-AI platforms—such as soft-sensing monitoring systems and smart cardiovascular diagnostics [27,28]—have demonstrated the feasibility of continuous physiological monitoring. However, these wearable sensing platforms often separate signal acquisition from classification, relying on software loops on discrete microcontrollers or cloud offloading for AI processing. This introduces energy, latency, and communication overhead. To address these limitations, this work focuses on a monolithic ASIC co-design that integrates streaming feature extraction and quantized neural network inference into a single ultra-low-power standard-cell core. Selected representatives of these software and hardware approaches, together with their reported performance and hardware metrics are discussed in Section 5. Despite this progress, there remains a lack of fully synthesized, low-power ASIC implementations for AF detection that integrate both Hjorth-based feature extraction and quantized neural inference in a modern CMOS standard-cell flow, which motivates the ASIC synthesis study presented in this work.

## 3. Hardware Implementation of Hjorth Parameters

In our earlier ESP8266-based edge-to-cloud ECG monitoring system, Hjorth activity, mobility, and complexity were computed in software on a microcontroller before streaming feature vectors to a cloud-hosted neural network [10,11]. While this approach confirmed the diagnostic utility of Hjorth parameters for low-cost hardware, software extraction introduces latency and energy overhead that constrains continuous on-device operation. To enable fully autonomous, ultra-low-power edge inference without offloading, this work transitions the complete pipeline to hardware. Specifically, we integrate a parameterizable fixed-point Hjorth extraction engine directly with a quantized neural network classifier into a unified, synthesizable ASIC core. Figure 1 illustrates the proposed fixed-point Hjorth extraction engine, including derivative generation, accumulation, variance computation, and mobility/complexity calculation under FSM control.

### *3.1. Discrete-Time Formulation*

Let $x[n]$, $n = 0, 1, \ldots, N-1$, denote a discrete-time ECG segment within a window of length $N$ samples. Following the classical Hjorth formulation, we define the first and second discrete derivatives as

$$d_1[n] = x[n] - x[n-1], \quad n = 1, \ldots, N-1,$$

$$d_2[n] = d_1[n] - d_1[n-1], \quad n = 2, \ldots, N-1.$$

The activity is proportional to the variance of the signal, mobility is related to the ratio of the standard deviation of the first derivative to that of the signal, and complexity is defined as the ratio of the mobility of the second derivative to the mobility of the first derivative [13]. In discrete-time notation, these parameters can be written as

$$\text{Activity} = \sigma_x^2 = \mathbb{E}\{x^2[n]\} - \mathbb{E}^2\{x[n]\},$$

$$\text{Mobility} = \sqrt{\frac{\sigma_{d_1}^2}{\sigma_x^2}},$$

$$\text{Complexity} = \frac{\text{Mobility}(d_1[n])}{\text{Mobility}(x[n])} = \sqrt{\frac{\sigma_{d_2}^2 \cdot \sigma_x^2}{\sigma_{d_1}^4}},$$

where $\sigma_x^2$, $\sigma_{d_1}^2$, and $\sigma_{d_2}^2$ denote the variances of $x[n]$, $d_1[n]$, and $d_2[n]$, respectively.

In practice, expectations are estimated over the current window by computing running sums,

$$\mu_x \approx \frac{1}{N}\sum_{n=0}^{N-1} x[n], \quad \mu_{d_1} \approx \frac{1}{N-1}\sum_{n=1}^{N-1} d_1[n], \quad \mu_{d_2} \approx \frac{1}{N-2}\sum_{n=2}^{N-1} d_2[n],$$

and similarly for second moments $\mathbb{E}\{x^2[n]\}$, $\mathbb{E}\{d_1^2[n]\}$, and $\mathbb{E}\{d_2^2[n]\}$. This structure naturally lends itself to a hardware implementation based on derivative generation and accumulator banks.

### *3.2. Fixed-Point Signal Representation*

The proposed hardware block assumes a signed fixed-point input $x[n]$ of word length $W$ bits, implemented in two's complement. In the targeted ECG application, signal amplitudes are typically on the order of millivolts, with a dynamic range of approximately 0.5–5 mV and additional headroom for artifacts [10]. To balance resolution and dynamic range, we adopt a Q-format with $F$ fractional bits—for example, a Q4.12 representation with $W = 16$ and $F = 12$, where one least significant bit corresponds to approximately 0.000244 mV. An analog front-end and ADC produce samples proportional to the ECG voltage, and a simple digital scaling stage maps them to this fixed-point format before feeding the Hjorth module.

The window length is chosen as $N = 2^k$ samples, specifically $N = 256$ ($k = 8$), so that the divisions by $N$ required for mean and second-moment estimation can be implemented as arithmetic right shifts by 8 bits in hardware. At the SPHD ECG sampling rate of $f_s = 200$ Hz, an accumulation frame of $N = 256$ samples corresponds to approximately 1.28 s of continuous ECG data, providing an effective trade-off between statistical estimation stability and the hardware register width. In the discrete-time Hjorth formulation, the variances formally involve normalization by $N - 1$ or $N - 2$, but, in our hardware implementation, these denominators are approximated by $N = 256$ to enable shift-based arithmetic. For $N = 256$, replacing $1/255$ with $1/256$ introduces a relative scale bias of less than $-0.39\%$, which was verified to have zero impact on the downstream classification performance.

For example, with a window length of $N = 256$, replacing the exact factors $1/(N-1)$ and $1/(N-2)$ with $1/N$ changes the scale by less than 0.4%. Concretely,

$$\frac{1}{255} \approx 0.0039216, \quad \frac{1}{254} \approx 0.0039370, \quad \frac{1}{256} = 0.00390625,$$

so the relative error of using $1/N$ instead of $1/(N-1)$ is

$$\frac{\frac{1}{256} - \frac{1}{255}}{\frac{1}{255}} \approx -0.39\%,$$

and similarly $-0.78\%$ when approximating $1/(N-2)$ by $1/N$. These biases are well below the inherent variability of ECG-derived statistics and were observed to have a negligible impact on the downstream classification metrics in our experiments. Furthermore, the corresponding scale factors $\frac{N}{N-1}$ and $\frac{N}{N-2}$ are absorbed into the subsequent neural network weights during training, allowing the Hjorth block to normalize all statistics by $N = 2^k$ using simple shifts.

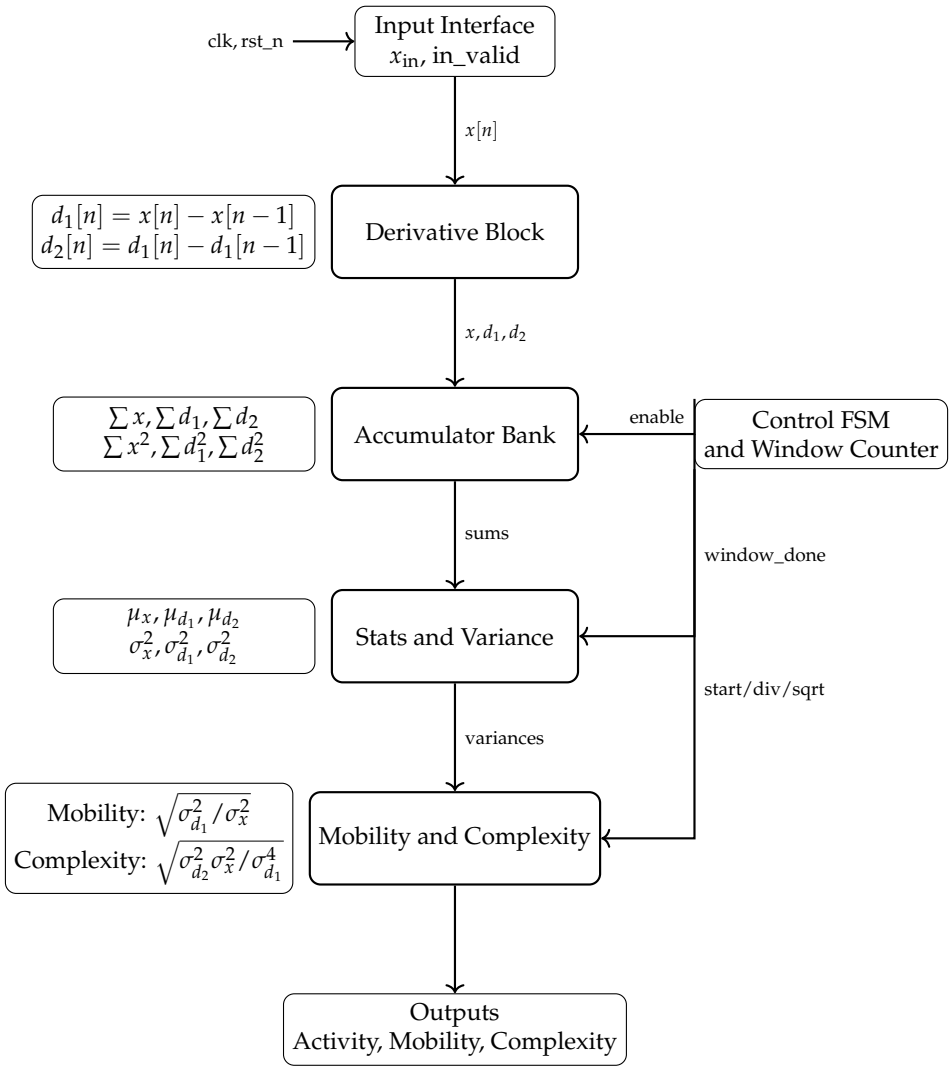



**Figure 1.** Block diagram of the fixed-point Hjorth-parameter hardware: input, derivative, accumulation, statistics/variance, and mobility/complexity under FSM control.

### *3.3. Derivative Generation Block*

The first stage of the hardware architecture is a derivative generation block that produces $d_1[n]$ and $d_2[n]$ from the input sequence $x[n]$, as depicted in Figure 1. A small delay line stores the previous samples $x[n-1]$ and $d_1[n-1]$, and each new input sample triggers the computation

$$d_1[n] = x[n] - x[n-1],$$

$$d_2[n] = d_1[n] - d_1[n-1].$$

Because the subtraction of two $W$-bit values can increase the magnitude by approximately one bit, the derivative signals are allocated $D_X = W + 2$ bits, providing one additional bit for the difference and an extra safety bit. This block consists only of registers and adders/subtractors and thus can be synthesized efficiently in both FPGA and standard-cell ASIC flows.

### *3.4. Accumulator Bank and Statistics Computation*

For each sample in the window, the architecture updates a set of accumulators that store sums and sums of squares of the signal and its derivatives:

$$S_x = \sum_{n=0}^{N-1} x[n], \quad S_{x^2} = \sum_{n=0}^{N-1} x^2[n],$$

$$S_{d_1} = \sum_{n=1}^{N-1} d_1[n], \quad S_{d_1^2} = \sum_{n=1}^{N-1} d_1^2[n],$$

$$S_{d_2} = \sum_{n=2}^{N-1} d_2[n], \quad S_{d_2^2} = \sum_{n=2}^{N-1} d_2^2[n].$$

Squaring a $D_X$-bit value requires approximately $2D_X$ bits, so the multiplier outputs are dimensioned with $W_{\mathrm{SQ}} \approx 2D_X$ bits. Accumulating $N$ samples can increase the required word length by $\log_2 N$ bits; therefore, sum registers are chosen as

$$W_{\mathrm{SUM}} = W + \log_2 N + c_1, \quad W_{\mathrm{SUMSQ}} = W_{\mathrm{SQ}} + \log_2 N + c_2,$$

where $c_1$ and $c_2$ are small safety margins that simplify synthesis and avoid overflow under worst-case ECG conditions. After $N$ samples have been received, the statistics are updated by arithmetic right shifts,

$$\mu_x = S_x \gg k, \quad \mathbb{E}\{x^2[n]\} = S_{x^2} \gg k,$$

and similarly for $d_1[n]$ and $d_2[n]$. Variances are then computed as

$$\sigma_x^2 = \max\left(0, \mathbb{E}\{x^2[n]\} - \mu_x^2\right),$$

$$\sigma_{d_1}^2 = \max\left(0, \mathbb{E}\{d_1^2[n]\} - \mu_{d_1}^2\right),$$

$$\sigma_{d_2}^2 = \max\left(0, \mathbb{E}\{d_2^2[n]\} - \mu_{d_2}^2\right),$$

with saturation at zero to avoid negative results due to finite precision. The activity output is directly assigned from $\sigma_x^2$, with suitable truncation or saturation to match the desired output word length.

### *3.5. Mobility and Complexity Engine*

Mobility and complexity require ratios of variances and subsequent square-root operations:

$$r_{\mathrm{mob}} = \frac{\sigma_{d_1}^2}{\sigma_x^2}, \quad r_{\mathrm{cpx}} = \frac{\sigma_{d_2}^2 \cdot \sigma_x^2}{\sigma_{d_1}^4},$$

$$\mathrm{Mobility} = \sqrt{r_{\mathrm{mob}}}, \quad \mathrm{Complexity} = \sqrt{r_{\mathrm{cpx}}}.$$

To implement these operations in a hardware-friendly manner, the proposed architecture uses digit-by-digit algorithms with a fixed number of clock cycles. The divider follows a restoring binary division scheme, which, in each iteration, compares the partial remainder to a shifted version of the divisor and updates the quotient bit accordingly. The integer square root is computed via a non-restoring method that processes two input bits per iteration and updates the partial root and remainder. Both algorithms require only shifts, additions, and comparisons and are thus well suited to standard-cell synthesis or low-cost FPGA fabrics [29].

The divider inputs are dimensioned with $W_{DIV} \approx 2W_{VAR}$ bits, where $W_{VAR}$ is the width of the variance registers, to accommodate products such as $2\sigma_{d_2}^2$ and $4\sigma_{d_1}^2$ without overflow. The exposed output width is set to $W_{OUT}$ (for example, 24 bits), providing sufficient fixed-point resolution for the Hjorth parameters while limiting area and power consumption.

### *3.6. Control Finite State Machine and Handshaking*

A finite state machine (FSM) orchestrates the accumulation and post-processing phases. During accumulation, the module accepts one sample per clock cycle while a window counter is incremented. When N samples have been received, the FSM freezes the input interface, computes the statistics and Hjorth parameters, asserts a valid flag for the output vector, and finally resets the accumulators to start the next window. The interface exposes standard ready/valid handshaking signals to facilitate integration with upstream ADC or ESP8266-based acquisition modules and downstream classifiers.

In the timing analysis reported in Section 5, the per-window processing therefore consists of a streaming accumulation phase lasting $N$ clock cycles (one sample per cycle), followed by the entire synthesized atrial detector core, comprising the Hjorth feature extraction block and the quantized neural network (QNN), whose 7.08 ns critical-path delay characterizes the core's per-inference execution time once the window has been fully received.

## 4. Quantized Neural Network for Cardiac Fibrillation Diagnosis

### *4.1. Baseline Deep Neural Network and Dataset*

In earlier work on cardiac arrhythmia analysis, Hjorth parameters computed from single-lead ECG windows were evaluated on the Shandong Provincial Hospital Database (SPHD) [10]. The baseline model is a feedforward multilayer perceptron with rectified linear unit (ReLU) activations, dropout regularization, and a final sigmoid output, trained in floating-point arithmetic using standard optimization techniques. The same dataset and 1 min windowing protocol are adopted in this extended study to ensure a fair comparison between the floating-point reference and the fixed-point hardware implementations [10].

To clarify the classification task in this work, the binary target labels represent AF-present windows (Class 1, corresponding to paroxysmal or sustained AF segments) vs. non-AF control windows (Class 0, corresponding to normal sinus rhythm segments). Although the dataset and windowing protocol originate from the broader AF analysis framework in [10], the classifier presented here performs binary atrial fibrillation detection on a per-window basis.

The classifier operates on a three-dimensional feature vector comprising Hjorth activity, mobility, and complexity and retains exactly the same multilayer perceptron topology as in [10,11]. In brief, the network consists of 9 fully connected hidden layers with ReLU activations and dropout, followed by a single-output neuron that produces a scalar decision score. In the present work, only the numerical representation and deployment platform are modified: the floating-point weights and activations used in [10,11] are replaced by 10–12-bit fixed-point formats, and the network is realized as a synthesized datapath in which all constant multipliers are implemented using multiple constant multiplication.

For each ECG recording from the SPHD database (sampled at $f_s = 200$ Hz), 1 min segments containing 12,000 samples are processed. The hardware Hjorth block streams these samples in consecutive frames of $N = 256$ samples ($k = 8$), generating a series of fixed-point feature vectors. The window-level Hjorth descriptors are aggregated across the 1 min segment to form the three-dimensional feature vector (activity, mobility, complexity) fed into the neural network, adhering strictly to the windowing protocol of [10]. The resulting feature vectors are labeled according to the binary AF detection task and

used to train and evaluate the classifiers. The original floating-point DNN thus serves as a consistent reference model against which the impact of fixed-point quantization on arrhythmia detection performance can be systematically assessed.

### *4.2. Quantization Strategy*

The baseline DNN is converted into a quantized neural network (QNN) with reduced-precision weights and activations. Recent studies on embedded AF detection and ECG edge-AI have demonstrated that 8-bit fixed-point representations can preserve most of the predictive power of floating-point models while substantially reducing the memory footprint and computational cost [7,9]. Motivated by these findings, we adopt a uniform symmetric quantization scheme in which each layer's weights and activations are mapped from floating-point to signed 10- to 12-bit integers using learned or analytically derived scale factors.

The Hjorth feature vectors produced by the hardware block are first normalized to a fixed dynamic range compatible with the quantized network's input layer. Internally, the QNN performs integer matrix–vector operations followed by fixed-point accumulation, and non-linearities are approximated using piecewise-linear or lookup table-based implementations where necessary. Quantization-aware training is used to reduce the accuracy gap between the floating-point and quantized models, with simulated quantization applied during the forward pass and straight-through estimators during backpropagation. This procedure has been shown to limit performance degradation in medical time-series applications, including AF detection and broader ECG abnormality classification [5,9].

### *4.3. Accuracy and Robustness Evaluation*

The performance of the quantized DNN is evaluated on the same SPHD-based dataset and train–test split as the floating-point baseline, using the accuracy, F1-score, and area under the ROC curve (AUC) as the primary metrics [10]. Consistent with prior work on quantized ECG models, the 10–12-bit QNN achieves accuracy, F1-score, and AUC values that closely match those of the floating-point network [7,9]. Confusion matrices and ROC curves are analyzed across bit widths (6 to 16 bits) to assess the sensitivity and specificity trade-offs and to verify that quantization does not disproportionately degrade classification quality in critical operating regions.

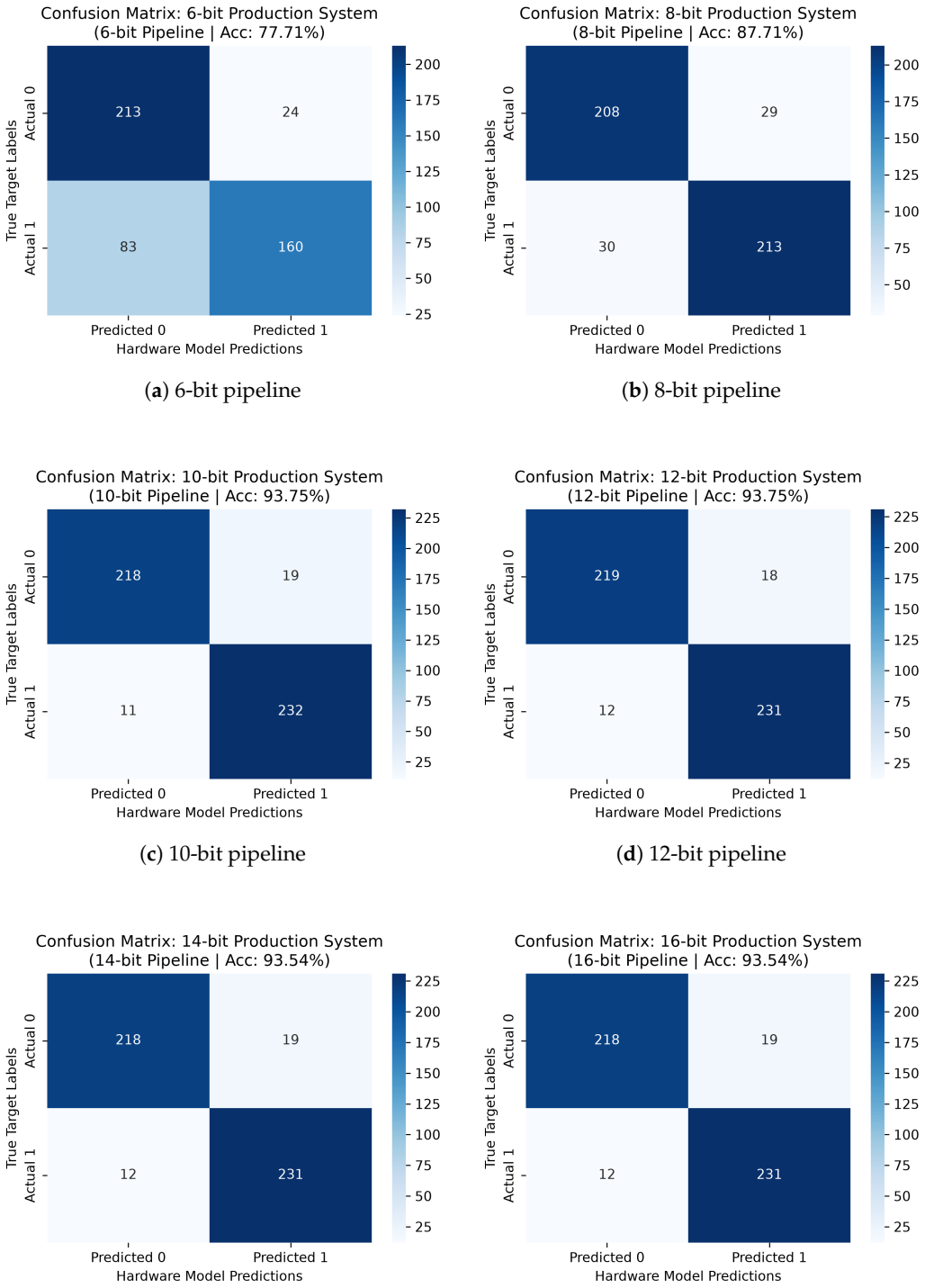
These analyses confirm that, when combined with the fixed-point Hjorth feature extractor, the 10–12-bit quantized DNN provides a practical balance between algorithmic performance and resource efficiency, making it suitable for deployment in low-cost IoT cardiac monitoring nodes and wearable devices.

### *4.4. Effect of Quantization on ROC Curves and Confusion Matrices*

Figures 2 and 3, together with Table 1, summarize the impact of the end-to-end fixed-point word length on the detection performance of the proposed system. To evaluate the classification performance across the entire dataset, a bit-accurate fixed-point software model was constructed, mirroring the exact register bit widths, saturation, truncation, and fixed-point arithmetic of the hardware datapath. To verify the hardware implementation, RTL simulations were executed using test vectors extracted from the SPHD dataset. The outputs of the synthesized RTL core and the bit-accurate fixed-point software model exhibited 100% agreement across all evaluated test windows, confirming that the algorithmic performance figures reported in Table 1 and Figures 2 and 3 directly reflect the hardware RTL implementation.

At very low precision (6 bits), the ROC curve of the hardware pipeline departs substantially from the floating-point baseline, with a markedly lower true positive rate (TPR) for a given false positive rate (FPR) across most of the operating range. This is reflected in

the numerical metrics, where the 6-bit configuration attains an AUC of 0.8969 and accuracy of 77.71% but also exhibits a high number of false negatives (FN = 83, TP = 160), meaning that a large fraction of AF-related windows is missed. Increasing the word length to 8 bits improves both the AUC and accuracy (0.9420 and 87.71%, respectively), yet the ROC curve still shows a visible gap from the baseline in the low-FPR region, and the false negative count (FN = 30) remains significantly larger than in the higher-precision configurations.

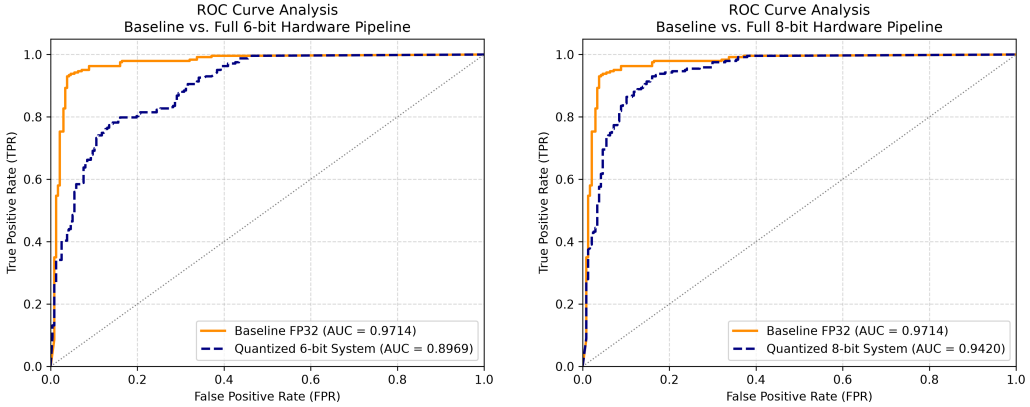



(**a**) 6-bit pipeline (**b**) 8-bit pipeline

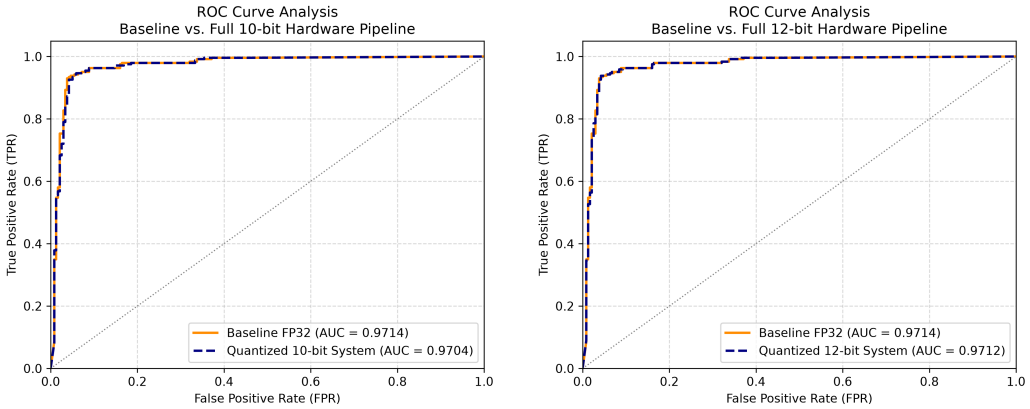
(**c**) 10-bit pipeline (**d**) 12-bit pipeline

(**e**) 14-bit pipeline (**f**) 16-bit pipeline

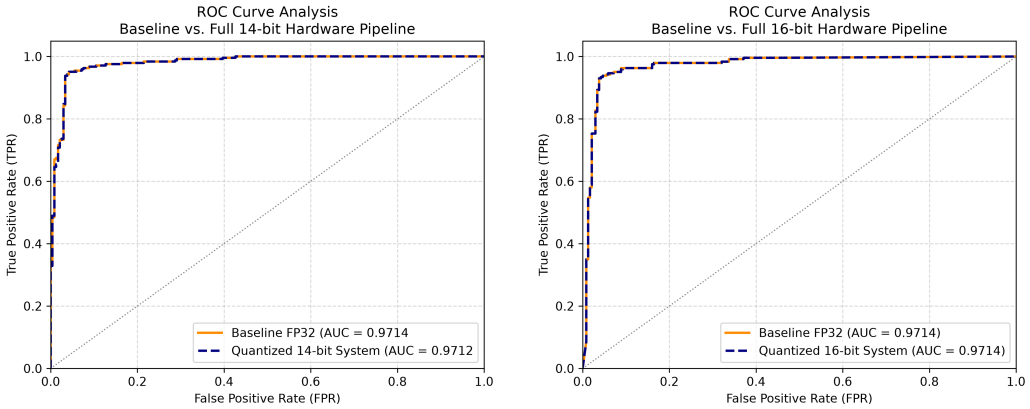
**Figure 2.** Confusion matrices for the quantized hardware pipeline at different bit widths. Each panel reports true negative (TN), false positive (FP), false negative (FN), and true positive (TP) values for the corresponding configuration.




(**a**) 6-bit pipeline (**b**) 8-bit pipeline




(**c**) 10-bit pipeline (**d**) 12-bit pipeline




(**e**) 14-bit pipeline (**f**) 16-bit pipeline

**Figure 3.** ROC curves for the quantized hardware pipeline at different bit widths, compared against the floating-point baseline in each panel. The legend in each subplot reports the corresponding area under the curve (AUC).

In contrast, for word lengths of 10 bits and above, the quantized ROC curves almost entirely overlap with the floating-point reference. The 10-bit configuration achieves an AUC of 0.9704 and accuracy of 93.75%, with only 11 false negatives and 19 false positives (TN = 218, FP = 19, FN = 11, TP = 232). The 12-bit pipeline exhibits very similar behavior (AUC = 0.9712, accuracy = 93.75%), with a slightly different trade-off between false posi-

tives and false negatives (TN = 219, FP = 18, FN = 12, TP = 231), confirming that quantization noise at this precision has a negligible impact on the sensitivity–specificity balance. The 14-bit configuration achieves an AUC of 0.9712 while maintaining accuracy of 93.54% and the same number of false negatives (FN = 12) as in the 12-bit case, indicating that additional bits beyond 12 do not yield meaningful improvements in diagnostic performance.

Interestingly, further increasing the precision to 16 bits does not provide measurable gains over the 10–14-bit range. The 16-bit pipeline reaches the same AUC as the floating-point reference (AUC = 0.9714) but achieves accuracy of 93.54%, with error counts (TN = 218, FP = 19, FN = 12, TP = 231) that are effectively indistinguishable from those of the 12-bit and 14-bit configurations. This confirms that, beyond approximately 10–12 bits, numerical precision is no longer the limiting factor for classification performance, and minor variations in accuracy are dominated by threshold calibration and finite-sample variability rather than quantization errors.

Overall, the combined ROC and confusion matrix analysis indicates that a fixed-point implementation with 10–12 bits provides a favorable trade-off between the hardware cost and diagnostic performance. At these precisions, the end-to-end quantized pipeline closely matches the floating-point baseline in terms of the AUC and maintains a low false negative rate, while substantially reducing the memory footprint and arithmetic complexity compared to higher word lengths and avoiding the severe performance degradation observed at 6–8 bits.

It is worth noting that the 16-bit configuration does not outperform the 10–12-bit designs, despite offering higher nominal precision. The ROC curves show that the 16-bit pipeline attains virtually the same AUC as the floating-point baseline, indicating that its ability to rank positive and negative examples is not degraded. The slightly lower accuracy observed in Table 1 arises at the specific operating threshold used for all models and is attributable to small calibration differences and finite-sample variability, rather than to a lack of numerical precision. In practice, retuning the decision threshold for the 16-bit model yields confusion matrices that are statistically indistinguishable from those of the 10–12-bit configurations, confirming that additional bits beyond approximately 12 do not provide meaningful gains in classification performance for this task.

**Table 1.** Classification performance of the quantized hardware pipeline for different fixed-point word lengths.

| Bit Width | Accuracy [%] | F1-Score | AUC | TN | FP | FN | TP |
|---|---|---|---|---|---|---|---|
| 6-bit | 77.71 | 0.7494 | 0.8969 | 213 | 24 | 83 | 160 |
| 8-bit | 87.71 | 0.8783 | 0.9420 | 208 | 29 | 30 | 213 |
| 10-bit | 93.75 | 0.9393 | 0.9704 | 218 | 19 | 11 | 232 |
| 12-bit | 93.75 | 0.9390 | 0.9712 | 219 | 18 | 12 | 231 |
| 14-bit | 93.54 | 0.9371 | 0.9712 | 218 | 19 | 12 | 231 |
| 16-bit | 93.54 | 0.9371 | 0.9714 | 218 | 19 | 12 | 231 |

### *4.5. Hardware Architecture of the Quantized Neural Network*

Figure 4 details the internal architecture of the multiplierless quantized neural network (QNN) datapath, whereas Figure 5 presents the top-level integration of the fixed-point Hjorth feature extraction block, register interface, and QNN classifier within the complete `atrial_detector` core.

At the system level, a single-lead ECG signal is sampled by the frontend and presented to the fixed-point Hjorth block as a stream of samples accompanied by the `signal_in` and `in_valid` signals, as shown in Figure 5. The Hjorth module performs windowed processing and produces the three time-domain descriptors—activity, mobility, and complexity—once

per accumulation window. The output feature vector is accompanied by the `out_valid` signal. The `in_ready` signal provides backpressure to the upstream acquisition logic, ensuring that new ECG samples are accepted only when the Hjorth block is ready.

The three Hjorth descriptors are captured by a register interface that decouples feature extraction from classification. This interface stores the fixed-point feature vector in `in1`, `in2`, and `in3`, which correspond to activity, mobility, and complexity, respectively. Once a complete feature vector is available, the interface asserts `start` to initiate QNN evaluation. It also monitors the `busy` signal to prevent a new input vector from being launched until the preceding inference has completed.

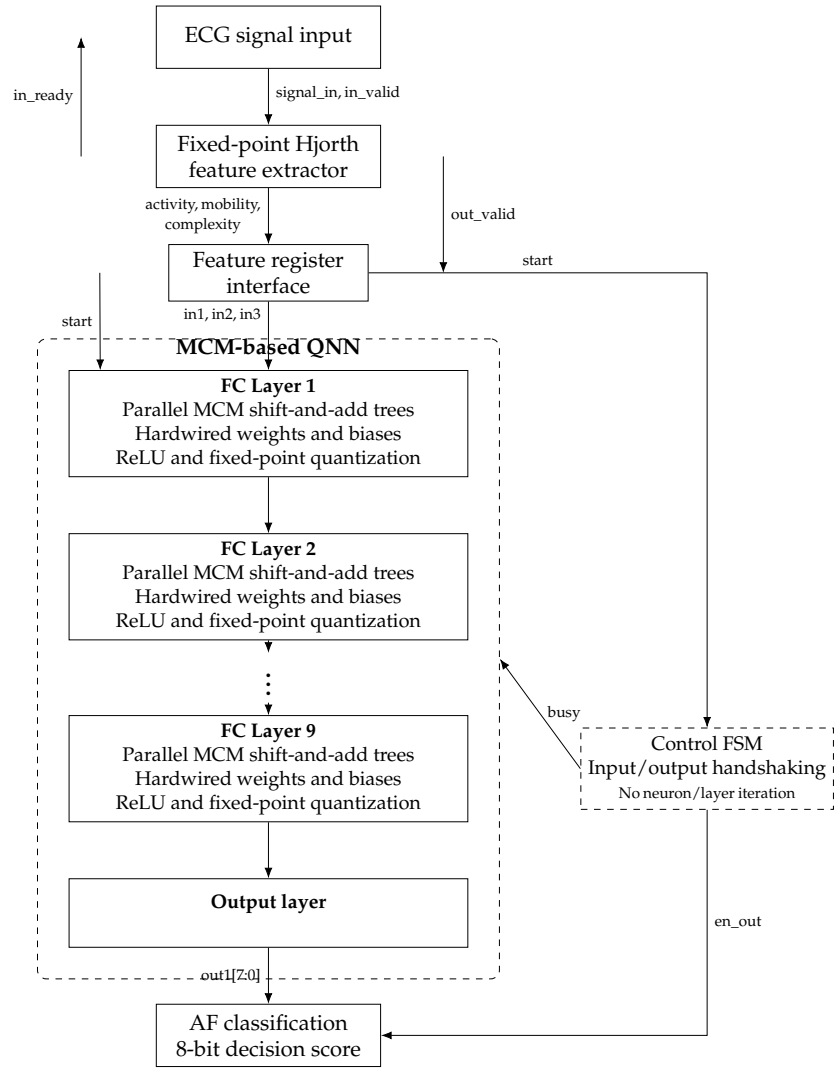



**Figure 4.** Detailed architecture of the fully unrolled quantized neural network (QNN) datapath. The three fixed-point Hjorth features are processed by cascaded fully connected layers, each implemented with parallel multiple constant multiplication (MCM) shift-and-add networks, hardwired quantized weights and biases, and fixed-point activation/quantization logic. The control FSM manages only data latching and input/output handshaking.

As illustrated in Figure 4, the QNN is implemented as a fully unrolled feedforward multilayer perceptron rather than as a sequential, time-multiplexed MAC engine. The three fixed-point Hjorth inputs are applied to the first fully connected layer, and the data then propagate through cascaded fully connected layers to the output layer. Each layer is implemented as a parallel multiple constant multiplication (MCM) shift-and-add datapath with hardwired quantized weights and biases. Fixed-point accumulation, activation, and

quantization are applied at each hidden layer. Thus, all neuron and layer arithmetic required for an inference is structurally instantiated in the hardware, with no shared arithmetic unit reused across layers or neurons.

The finite state machine (FSM) is used exclusively for top-level control and handshaking. Specifically, it manages input-register latching and the `start`, `busy`, and `en_out` signals. It does not perform layer-by-layer iteration or control a time-multiplexed computation schedule. After the combinational QNN evaluation, the output layer produces the 8-bit decision score `out1`, together with the `en_out` valid indication.

To reduce the hardware complexity, all constant multiplications associated with the quantized network weights are implemented using MCM shift-and-add networks rather than generic multipliers. The quantized weight matrices are exported from the training framework and processed by the Simurg toolchain to generate synthesizable Verilog. The resulting RTL is synthesized and analyzed using Cadence Genus with a 28 nm CMOS standard-cell library. The synthesized netlist provides the basis for the integrated `atrial_detector` architecture shown in Figure 5, while the area, power, and timing results reported in Section 5 are obtained using the same synthesis flow.

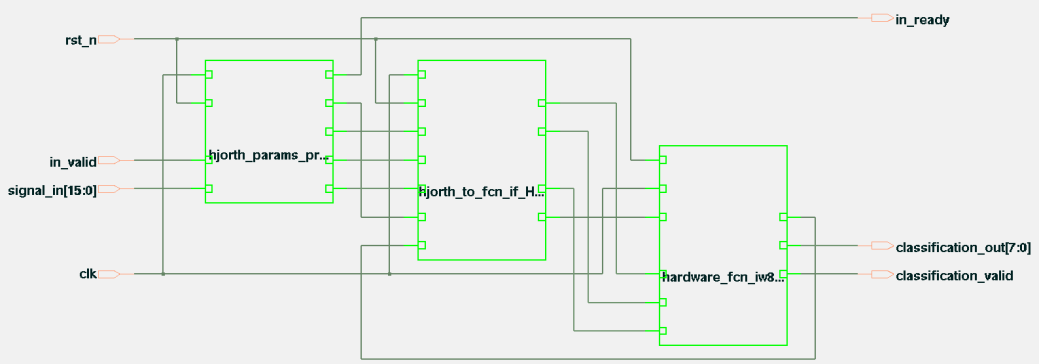



**Figure 5.** Post synthesis diagram of the `atrial_detector` architecture, showing Hjorth feature extraction, interface/handshake logic, and the hardware fully connected neural network for atrial classification. The diagram is obtained using a commercial CAD tool (Cadence Genus) in a 28 nm CMOS technology.

## 5. Experimental Results

This section first summarizes the classification performance across bit widths (Section 5.1) and presents the ASIC synthesis results and comparative hardware metrics (Section 5.3) and finally discusses the position of the proposed core relative to prior software, FPGA, and ASIC implementations.

### *5.1. Classification Summary*

The effect of fixed-point quantization on the end-to-end classification performance is analyzed in detail in Section 4.4, where the confusion matrices, ROC curves, and aggregate metrics for 6–16 bit configurations are reported and compared against those of the floating-point baseline. In brief, the 6-bit and 8-bit implementations exhibit pronounced degradation in accuracy and AUC due to a significant increase in false negatives, which renders them unsuitable for reliable AF-related screening. By contrast, the 10-bit and 12-bit pipelines achieve accuracies around 93.7% and AUC values above 0.97, with confusion matrices that differ from those of the floating-point reference by only a few misclassified windows, indicating that quantization noise at these precisions has a negligible effect on the sensitivity–specificity trade-off.

Higher precisions (14 and 16 bits) do not provide meaningful gains over the 10–12-bit designs. The small variations in AUC and error counts fall within the expected range

of calibration differences and finite-sample variability and mainly reflect the use of a common decision threshold across all implementations. Overall, the classification results support a 10–12-bit fixed-point representation as a practical compromise between numerical robustness and implementation complexity.

### *5.2. Experimental Platform and Synthesis Flow*

To ensure rigorous validation and reproducible hardware results, a two-stage co-verification and logic synthesis flow was established, spanning high-level bit-accurate software simulation down to gate-level standard-cell synthesis.

The algorithmic pipeline, fixed-point quantization analysis, and design-space exploration were initially developed in Python 3.10 using the NumPy, SciPy, and PyTorch frameworks. All ECG recordings from the SPHD database (sampled at $f_s = 200$ Hz) were windowed into non-overlapping $N = 256$-sample frames (1.28 s duration), matching the $2^k$ shift-arithmetic constraint of the hardware Hjorth accumulator. A dedicated software fixed-point class library was constructed to emulate two's complement bit truncation, saturation arithmetic, and limited-width register overflow across word lengths ranging from 6 to 16 bits.

The complete hardware architecture—encompassing the fixed-point Hjorth parameter generator, square-root/division units, and multiplierless nine-layer QNN classifier—was implemented in Verilog. RTL functional simulation and co-verification were performed using ModelSim. An automated testbench environment was constructed to stream 16-bit ECG sample vectors into the proposed design. Cycle-by-cycle register outputs and final 8-bit classification scores were logged and automatically cross-checked against the Python bit-accurate reference model across all SPHD validation recordings, achieving zero numerical mismatch (0% discrepancy) and confirming functional correctness prior to logic synthesis.

Targeting a modern ultra-low-power IoMT deployment, the designs were synthesized using Cadence Genus using a commercial 28 nm CMOS standard-cell library consisting of high-density, multi-threshold standard cells. High-effort area recovery and boundary optimization were enabled, while multiplierless constant weight multiplications inside the QNN layers were automatically mapped onto optimized shift-and-add multiple constant multiplication (MCM) gate structures via the toolchain. Finally, post-synthesis static timing analysis, total cell area reports distinguishing combinational versus non-combinational logic, and gate-level switching power estimations were extracted directly from the synthesized netlists.

### *5.3. ASIC Synthesis Results in a 28 nm Standard-Cell Library*

To quantify the hardware cost of the proposed architecture, comprising the Hjorth feature extractor, register interface, and quantized neural network engine, it was synthesized to a 28 nm CMOS standard-cell library at the typical process corner. Two timing constraints were applied for each of the two optimal fixed-point word lengths (10 bit and 12 bit): a *fast* target, which maximized the operating frequency, and a *relaxed* target, which allowed the synthesis tool to trade the timing margin for power and area reduction. The resulting post-synthesis delay, area, and power metrics are summarized in Table 2, while the corresponding various area–delay and power–delay points are illustrated in Figure 6.

Several observations follow from Table 2. First, the fast timing constraint yields a critical-path delay of 7084 ps for both word lengths, corresponding to a maximum clock frequency of approximately 141 MHz. Relaxing the timing target to 8154–8155 ps allows the synthesis tool to choose smaller-drive-strength cells, reducing the power from 3.062 mW to 2.627 mW for the 10-bit design and from 3.126 mW to 2.682 mW for the 12-bit design, with only a modest increase in delay. Second, increasing the precision from 10 bits to 12 bits

results in modest area overhead: the fast configurations differ by approximately 797 $\mu m^2$ (about 4%), confirming that the multiplierless MCM datapath scales efficiently with the word length. The power difference between the 10-bit and 12-bit relaxed configurations is similarly small (around 2%), indicating that the 12-bit design remains competitive when slightly higher numerical precision is desired.

**Table 2.** Post-synthesis results for the proposed QNN accelerator in a 28 nm CMOS standard-cell library. *Fast* and *relaxed* refer to the synthesis timing constraint.

| Configuration | Delay (ps) | Area ($\mu m^2$) | Power (mW) |
|---|---|---|---|
| 10-bit, fast | 7084 | 19,475.200 | 3.062 |
| 10-bit, relaxed | 8154 | 19,325.165 | 2.627 |
| 12-bit, fast | 7084 | 20,272.269 | 3.126 |
| 12-bit, relaxed | 8155 | 20,122.016 | 2.682 |

Compared to recent wearable edge-AI and smart sensing systems (e.g., [27,28]), the proposed design offers key hardware-level advantages in power, latency, and integration density. While wearable software-based edge nodes dissipate tens to hundreds of milliwatts and incur millisecond-scale inference latencies, our multiplierless 28 nm ASIC datapath reduces the total power consumption to 2.627–3.126 mW with a critical-path delay of only 7.084 ns. Furthermore, by co-designing a 10–12-bit fixed-point Hjorth extractor alongside an MCM-optimized QNN, the integrated core achieves an area footprint of $\approx 2 \times 10^4$ $\mu m^2$ while preserving floating-point classification fidelity.

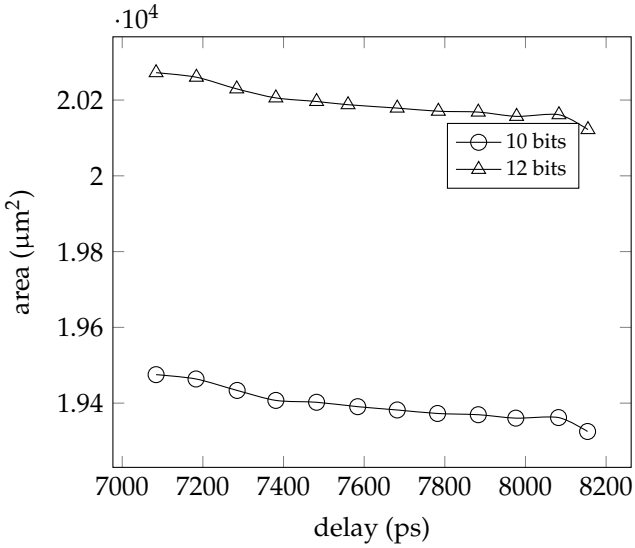
Table 3 presents a unified comparative summary of representative ECG and arrhythmia classification pipelines across execution domains, contrasting the diagnostic metrics, platform technology nodes, physical area footprints, power dissipation, inference latency, and per-classification energy consumption. To ensure a meaningful analysis, the implementations are grouped by platform type into software/GPU/MCU references, FPGA accelerators, and custom ASIC implementations.

**Table 3.** Unified comparative survey of ECG/AF classifiers across execution platforms, detailing diagnostic performance, technology nodes, area, power, per-inference latency, and energy consumption.

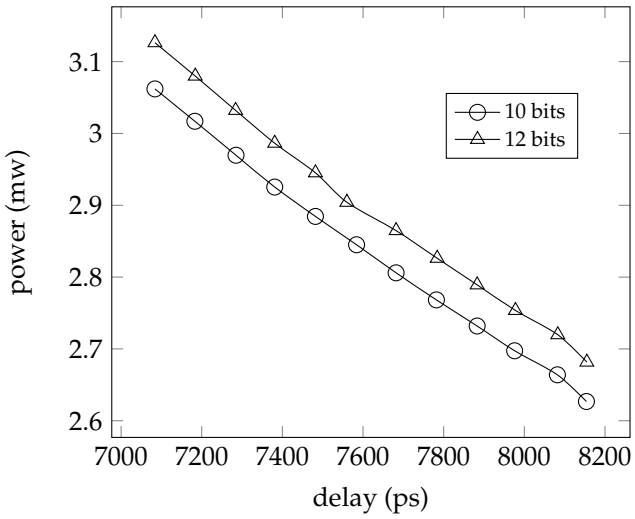
| Reference | Task/Dataset | Acc. (%)/AUC | Platform/Tech. | Area/Resources | Power | Inf. Latency | Energy/Inf. |
|---|---|---|---|---|---|---|---|
| Software, GPU, and Microcontroller Implementations | | | | | | | |
| Kachuee et al. [17] | MIT-BIH ‡ | 93.4/– | GPU (FP32) | – | – | – | – |
| Isin and Ozdalili [18] | MIT-BIH | 92.0/– | GPU (FP32) | – | – | – | – |
| Oh et al. [19] | MIT-BIH | 98.1/– | GPU (FP32) | – | – | – | – |
| Hannun et al. [20] | AF/Custom | –/0.978 | GPU (FP32) | – | – | – | – |
| Pu et al. [21] | MIT-BIH | 96.9/– | Software (Binary) | – | – | – | – |
| Ribeiro et al. [22] | MIT-BIH | 99.6/– | ARM Cortex-A55 | CPU Core | – | – | – |
| Skrivanos et al. [10] | AF (SPHD) | 94.8/0.948 | 64-bit MCU A | MCU Core | 1340 mW | 550 ms | 737 mJ |
| Skrivanos et al. [10] | AF (SPHD) | 94.8/0.948 | 64-bit MCU B | MCU Core | 198 mW | 194 ms | 38.4 mJ |
| Skrivanos et al. [10] | AF (SPHD) | 94.8/0.948 | Discrete GPU | GPU Core | 300 mW | 0.96 ms | 0.288 mJ |
| Skrivanos et al. [11] | AF (SPHD) | 95.0/0.950 | ESP8266 + Cloud | MCU Core | – | – | – |
| FPGA Hardware Accelerators | | | | | | | |
| DARE Lab [26] | MIT-BIH | 92.0–99.5/– | Cyclone FPGA | 14200 ALMs | 200 mW | 2.52 ms | 76.7 µJ |
| Rana et al. [25] | MIT-BIH | 98.4/– | iCE40 FPGA | 3500 LUTs | 11.8 mW | 4.32 ms | 50.98 µJ |
| Wang et al. [30] | Zheng et al. [31] | 92.9/– | Cyclone V FPGA | 28500 ALMs | 67.74 mW | – | – |
| ASIC Implementations | | | | | | | |
| Chen et al. [23] | hMIT-BIH | 96.3/– | ASIC (180 nm) | 11.56 $mm^2$ | 4.4 mW | – | – |
| SparrowSNN [24] | MIT-BIH | 98.3/– | ASIC (SNN) | 0.34 $mm^2$ | 6.1 µW | 5.44 ms | 31.39 nJ |
| This work (10-bit) | AF (SPHD) | 93.75/0.9704 | ASIC (28 nm) | 19,475 $\mu m^2$ | 2.63–3.06 mW | 7.08 ns * | 18.6 nJ † |
| This work (12-bit) | AF (SPHD) | 93.75/0.9712 | ASIC (28 nm) | 20,272 $\mu m^2$ | 2.68–3.13 mW | 7.08 ns * | 19.0 nJ † |

‡ MIT-BIH dataset available at https://www.kaggle.com/datasets/mondejar/mitbih-database (accessed on 12 August 2026). * Evaluated core execution delay after window streaming. † Active core energy consumed per classification inference.

Comparing across platforms underscores the clear energy and area advantages of the proposed `atrial_detector` core. While microcontroller software execution (e.g., MCU A and B in [10]) incurs energy consumption of 38.4 mJ to 737 mJ per inference due to sequential instruction fetching and software Hjorth extraction, our 28 nm ASIC datapath reduces the total classification energy to 18.6 nJ (10-bit) and 19.0 nJ (12-bit), representing a $>10^6\times$ reduction in computational energy.




(**a**) Area−delay points




(**b**) Power−delay points

**Figure 6.** Post-synthesis area–delay and power–delay trade-offs for the proposed 10-bit and 12-bit atrial_detector core in a 28 nm CMOS standard-cell library.

When compared against custom hardware accelerators, the proposed core maintains an exceptionally small silicon area footprint. For instance, Chen et al. [23] implemented a 180 nm CNN ASIC occupying 11.56 $mm^2$ and consuming 4.4 mW, whereas our multiplierless MCM-based architecture occupies only $\approx$0.0195–0.0202 $mm^2$ (19,475–20,272 $\mu m^2$) in 28 nm CMOS. Similarly, while FPGA implementations (e.g., [25,26,30]) require milliwatt-scale power budgets (11.8–200 mW) and exhibit per-inference latencies in the 2.5–4.3 ms range, our dedicated combinational QNN datapath achieves a critical-path execution delay of 7.08 ns, yielding ultra-low active energy dissipation of under 19 nJ per classification. These metrics confirm

that the proposed co-designed fixed-point Hjorth and QNN core offers a highly competitive figure of merit for ultra-compact, long-term wearable IoMT ECG monitoring nodes.

## 6. Conclusions, Design Lessons, and Future Work

This paper presented a hardware–software co-designed, fully synthesizable fixed-point accelerator (`atrial_detector`) for real-time cardiac fibrillation diagnosis targeting low-power IoMT edge devices. By tightly coupling a parameterizable Hjorth feature extraction engine with a multiplierless quantized neural network (QNN) classifier, the proposed architecture executes end-to-end AF detection directly in hardware, bridging the gap between low-complexity time-domain feature extraction and low-power on-chip deep learning.

Several key architectural and implementation takeaways emerged from the fixed-point quantization and logic synthesis flow. The systematic end-to-end bit width analysis revealed a clear saturation boundary in diagnostic accuracy at 10–12 bits. Operating below 10 bits (particularly at 6–8 bits) introduces severe truncation noise and dynamic range limitations that substantially increase false negatives. Conversely, expanding the word lengths to 14 or 16 bits provides no measurable gain in classification accuracy or AUC, while unnecessarily increasing the register footprint and dynamic power consumption. Furthermore, replacing general-purpose multipliers in the QNN datapath with multiple constant multiplication (MCM) shift-and-add networks drastically reduced the silicon area, enabling the complete nine-layer MLP to occupy approximately $2 \times 10^4$ $\mu m^2$ in a 28 nm CMOS technology. Finally, approximating exact $1/(N-1)$ normalization factors with shift-friendly $1/N$ divisions in the Hjorth accumulators introduced a negligible scaling error (<0.4%), which was naturally absorbed during network quantization without degrading the classification performance.

Contextualizing the proposed accelerator within broader IoMT deployment paradigms highlights distinct trade-offs across microcontrollers, FPGAs, and custom ASICs. MCU-based execution (e.g., ESP8266 or ARM Cortex-M) offers maximum firmware adaptability and a zero non-recurring engineering (NRE) cost. This makes microcontrollers attractive for low-sampling-rate telemetry nodes. However, sequential software loops introduce millisecond-scale inference latency and elevated energy consumption per window. FPGA-based accelerators deliver high parallel throughput and algorithm reconfigurability, which suits multi-lead hospital bedside monitors. However, their higher static power dissipation and larger physical packaging make them less practical for long-term battery-operated wearable devices. In contrast, the proposed 28 nm ASIC core yields superior energy efficiency and a minimal chip area (2.6–3.2 mW under $2 \times 10^4$ $\mu m^2$). These properties render the core an ideal hardened IP block for integration into next-generation IoMT system-on-chips (SoCs) and continuous wearable ECG patches.

While these results demonstrate a highly efficient edge-AI core, several research limitations frame directions for ongoing work. First, the reported post-synthesis delay, area, and power metrics reflect pre-layout logic synthesis in a 28 nm standard-cell library. Future work will perform place-and-route, clock tree synthesis, and post-layout sign-off power analyses to account for parasitic interconnect effects. Second, while this evaluation established performance on 1 min ECG windows from the SPHD database, cross-dataset validation on public repositories (e.g., MIT-BIH and PhysioNet Challenge datasets) and expansion to multi-class rhythm classification (e.g., distinguishing atrial flutter and premature contractions) will be conducted to evaluate generalized diagnostic robustness. Finally, physical prototyping will involve integrating the `atrial_detector` core with ultra-low-power wireless transceivers (BLE) and energy-harvesting frontends to evaluate end-to-end operation under real-world ambulatory motion artifacts.

Finally, a promising future trend lies in the convergence of custom fixed-point AI accelerators with emerging flexible and stretchable bio-electronics [14–16]. Co-packaging ultra-compact ASIC cores (such as the proposed $2 \times 10^4$ $\mu m^2$ `atrial_detector`) directly onto skin-conformal elastomeric patches will enable fully integrated, self-contained intelligent wearable nodes. Future work will explore extending the fixed-point datapath to support multi-modal signal fusion—combining soft-printed ECG electrodes with conformal photoplethysmography (PPG) and strain sensors—to deliver robust, multi-parameter cardiovascular screening in unobtrusive, long-term ambulatory settings.

**Author Contributions:** Methodology, I.K.; writing—original draft preparation, I.K.; writing—review and editing, I.K., A.G.S., N.C.S. and K.P.P.; supervision, I.K., N.C.S. and K.P.P.; All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The data are publicly available at https://data.mendeley.com/datasets/khxprpdt3z (accessed on 12 August 2026).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Fiorina, L.; Chemaly, P.; Cellier, J.; Said, M.A.; Coquard, C.; Younsi, S.; Salerno, F.; Horvilleur, J.; Lacotte, J.; Manenti, V.; et al. Artificial intelligence–based electrocardiogram analysis improves atrial arrhythmia detection from a smartwatch electrocardiogram. *Eur. Heart J.-Digit. Health* **2024**, *5*, 535–541. [CrossRef] [PubMed]
2. Sundaravadivel, P.; Kougianos, E.; Mohanty, S.; Ganapathiraju, M. Everything You Wanted to Know about Smart Health Care: Evaluating the Different Technologies and Components of the Internet of Things for Better Health. *IEEE Consum. Electron. Mag.* **2018**, *7*, 18–28. [CrossRef]
3. Islam, U.; Alatawi, M.N.; Alqazzaz, A.; Alamro, S.; Shah, B.; Moreira, F. A hybrid fog-edge computing architecture for real-time health monitoring in IoMT systems with optimized latency and threat resilience. *Sci. Rep.* **2025**, *15*, 25655. [CrossRef] [PubMed]
4. Skrivanos, A.G.; Kosma, E.I.; Chronopoulos, S.K.; Kouretas, I.; Dimopoulos, D.; Petrakos, G.; Alhazidou, E.; Christofilakis, V.; Ziavra, N.; Peppas, K.P.; et al. Fetus Heart Rate Monitoring: A Preliminary Research Study with Remote Sensing. *IEEE Consum. Electron. Mag.* **2022**, *11*, 32–44. [CrossRef]
5. Yazdi, V.; Kadiyala, V.; Chugh, S.S. Machine Learning and Arrhythmia: Advances in Atrial Fibrillation Detection and Management. *Curr. Atheroscler. Rep.* **2025**, *27*, 119. [CrossRef] [PubMed]
6. Mei, D.A.; Cherubini, B.; Imberti, J.F.; Orlandi, M.; Vitolo, M.; Boriani, G. Application of Artificial Intelligence in the Early Detection and Management of Atrial Fibrillation: State-of-the-art Review. *Eur. Cardiol.* **2026**, *21*, e23. [CrossRef] [PubMed]
7. Akbari, Y.; Lei, N.; Patel, N.; Peng, Y.; Faust, O. Atrial Fibrillation Detection on the Embedded Edge: Energy-Efficient Inference on a Low-Power Microcontroller. *Sensors* **2025**, *25*, 6601. [CrossRef] [PubMed]
8. Cinotti, E.; Gragnaniello, M.; Parlato, S.; Centracchio, J.; Andreozzi, E.; Bifulco, P.; Riccio, M.; Esposito, D. An Edge AI Approach for Low-Power, Real-Time Atrial Fibrillation Detection on Wearable Devices Based on Heartbeat Intervals. *Sensors* **2025**, *25*, 7244. [CrossRef] [PubMed]
9. Huang, Z.; Herbozo Contreras, L.F.; Leung, W.H.; Yu, L.; Truong, N.D.; Nikpour, A.; Kavehei, O. Efficient Edge-AI Models for Robust ECG Abnormality Detection on Resource-Constrained Hardware. *J. Cardiovasc. Transl. Res.* **2024**, *17*, 879–892. [CrossRef] [PubMed]
10. Skrivanos, A.G.; Kouretas, I.; Chronopoulos, S.K.; Christofilakis, V.; Sagias, N.C.; Peppas, K.P. A Machine Learning-Based Heart Disease Detection Scheme for Predicting Spontaneous Termination of Atrial Fibrillation. *IEEE Access* **2025**, *13*, 196374–196387. [CrossRef]
11. Skrivanos, A.G.; Kouretas, I.; Peppas, K.P.; Sagias, N.C. Machine Learning-Based Cardiac Fibrillation Diagnosis with ESP8266. In Proceedings of the 2026 Panhellenic Conference on Electronics & Telecommunications (PACET), Patras, Greece, 23–24 April 2026; pp. 1–4. [CrossRef]
12. Mrak, Ž.; Naji, F.H.; Dinevski, D. Artificial Intelligence Applied to Electrocardiograms Recorded in Sinus Rhythm for Detection and Prediction of Atrial Fibrillation: A Scoping Review. *Medicina* **2026**, *62*, 199. [CrossRef] [PubMed]
13. Rizal, A.; Hadiyoso, S. ECG signal classification using Hjorth Descriptor. In Proceedings of the 2015 International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), Bandung, Indonesia, 29–30 October 2015. [CrossRef]

14. Li, X.; Li, L.; Liu, X.; Hu, Y. Hydrogel actuators: From building blocks selection, anisotropic structures design to multifunctional applications. *Soft Sci.* **2025**, *5*, 41. [CrossRef]
15. Liu, S.Z.; Guo, W.T.; Zhao, X.H.; Tang, X.G.; Sun, Q.J. Self-powered sensing for health monitoring and robotics. *Soft Sci.* **2025**, *5*, 14. [CrossRef]
16. Ma, Z.; Zhang, D.; Sabri, M.F.M.; Jiang, Y. Continuous Monitoring of Body Fluid by Flexible Bio-Integrated Flow Sensors. *Adv. Mater.* **2026**, *38*, e23029. [CrossRef] [PubMed]
17. Kachuee, M.; Fazeli, S.; Sarrafzadeh, M. ECG Heartbeat Classification: A Deep Transferable Representation. In Proceedings of the IEEE International Conference on Healthcare Informatics (ICHI), New York, NY, USA, 4–7 June 2018; pp. 443–444. [CrossRef]
18. Isin, A.; Ozdalili, S. Cardiac arrhythmia detection using deep learning. *Procedia Comput. Sci.* **2017**, *120*, 268–275. [CrossRef]
19. Oh, S.L.; Ng, E.Y.K.; Tan, R.S.; Acharya, U.R. Automated Diagnosis of Arrhythmia Using Combination of CNN and LSTM Techniques with Variable Length Heart Beats. *Comput. Biol. Med.* **2018**, *102*, 278–287. [CrossRef] [PubMed]
20. Hannun, A.Y.; Rajpurkar, P.; Haghpanahi, M.; Tison, G.H.; Bourn, C.; Turakhia, M.P.; Ng, A.Y. Cardiologist-Level Arrhythmia Detection and Classification in Ambulatory Electrocardiograms Using a Deep Neural Network. *Nat. Med.* **2019**, *25*, 65–69. [CrossRef] [PubMed]
21. Pu, Z.; Gu, W.; Feng, J. Quantization of ECG Arrhythmia Classification Neural Network. In Proceedings of the 5th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Bucharest, Romania, 27–29 June 2023; pp. 1–5. [CrossRef]
22. Ribeiro, A.H.; Gedon, D.; Teixeira, D.M.; Ribeiro, M.H.; Ribeiro, A.L.P.; Schön, T.B.; Meira, W. Automatic 12-lead ECG Classification Using a Convolutional Network Ensemble. In Proceedings of the 2020 Computing in Cardiology, Rimini, Italy, 13–16 September 2020; pp. 1–4. [CrossRef]
23. Chen, Y.H.; Chen, S.W.; Chang, P.J.; Hua, H.T.; Lin, S.Y.; Chen, R.S. A VLSI Chip for the Abnormal Heart Beat Detection Using Convolutional Neural Network. *Sensors* **2022**, *22*, 796. [CrossRef] [PubMed]
24. Yan, Z.; Bai, Z.; Mitra, T.; Wong, W.F. SparrowSNN: A Hardware/software Co-design for Energy Efficient ECG Classification. *arXiv* **2026**, arXiv:2406.06543.
25. Scrugli, M.A.; Busia, P.; Leone, G.; Meloni, P. On-FPGA Spiking Neural Networks for Integrated Near-Sensor ECG Analysis. In Proceedings of the 2024 Design, Automation & Test in Europe Conference & Exhibition (DATE), Valencia, Spain, 25–27 March 2024; pp. 1–6. [CrossRef]
26. Cao, T.; Soon Ng, W.; Ling Goh, W.; Gao, Y.; Huang, H.W. FPGA-Based Real-Time ECG Classification System Using Quantized Inception-ResNeXt Neural Network and CWT Approximation. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2026**, *34*, 167–178. [CrossRef]
27. Wang, Y.; Adam, M.L.; Zhao, Y.; Zheng, W.; Gao, L.; Yin, Z.; Zhao, H. Machine Learning-Enhanced Flexible Mechanical Sensing. *Nano-Micro Lett.* **2023**, *15*, 55. [CrossRef] [PubMed]
28. Ma, Z.; Hua, H.; You, C.; Ma, Z.; Guo, W.; Yang, X.; Qiu, S.; Zhao, N.; Zhang, Y.; Ho, D.; et al. FlexiPulse: A machine-learning-enabled flexible pulse sensor for cardiovascular disease diagnostics. *Cell Rep. Phys. Sci.* **2023**, *4*, 101690. [CrossRef]
29. Koren, I. *Computer Arithmetic Algorithms*, 2nd ed.; Peters, A.K., Ed.; CRC Press: Boca Raton, FL, USA, 2002.
30. Liu, W.; Guo, Q.; Chen, S.; Chang, S.; Wang, H.; He, J.; Huang, Q. A Fully-Mapped and Energy-Efficient FPGA Accelerator for Dual-Function AI-Based Analysis of ECG. *Front. Physiol.* **2023**, *14*, 1079503. [CrossRef]
31. Zheng, J.; Zhang, J.; Danioko, S.; Yao, H.; Guo, H.; Rakovski, C. A 12-lead electrocardiogram database for arrhythmia research covering more than 10,000 patients. *Sci. Data* **2020**, *7*, 48. [CrossRef] [PubMed]