

# Bespoke Co-processor for Energy-Efficient Health Monitoring on RISC-V-based Flexible Wearables

Theofanis Vergos\*, Polykarpos Vergos\*, Mehdi B. Tahoori\*, Georgios Zervakis<sup>§</sup>  
 \*Karlsruhe Institute of Technology, Germany, <sup>§</sup>National Technical University of Athens, Greece  
 \*{theofanis.vergos, polykarpos.vergos, mehdi.tahoori}@kit.edu, <sup>§</sup>zervakis@ece.ntua.gr

**Abstract**—Flexible electronics offer unique advantages for conformable, lightweight, and disposable healthcare wearables. However, their limited gate count, large feature sizes, and high static power consumption make on-body machine learning classification highly challenging. While existing bendable RISC-V systems provide compact solutions, they lack the energy efficiency required. We present a mechanically flexible RISC-V that integrates a bespoke multiply-accumulate co-processor with fixed coefficients to maximize energy efficiency and minimize latency. Our approach formulates a constrained programming problem to jointly determine co-processor constants and optimally map Multi-Layer Perceptron (MLP) inference operations, enabling compact, model-specific hardware by leveraging the low fabrication and non-recurring engineering costs of flexible technologies. Post-layout results demonstrate near-real-time performance across several healthcare datasets, with our circuits operating within the power budget of existing flexible batteries and occupying only 2.42mm<sup>2</sup>, offering a promising path toward accessible, sustainable, and conformable healthcare wearables. Our microprocessors achieve an average 2.35x speedup and 2.15x lower energy consumption compared to the state of the art.

**Index Terms**—Flexible Electronics, Machine Learning, RISC-V

## I. INTRODUCTION

The growing emphasis on personalized and preventive healthcare has intensified the need for continuous monitoring of physiological signals outside clinical settings [1]. Wearable devices play a central role in this shift, enabling tracking of vital signs during daily activities [2], [3]. Although health monitoring could greatly benefit from—and, in many cases, outright necessitates—flexible, conformable, and disposable devices [4], [5], most research has focused on Machine Learning (ML), a key enabler for such systems that extracts insights from complex biosignals [6]–[11], while flexible hardware remains overlooked [5]. Most commercial wearables rely on non-disposable, rigid, silicon systems, hindering comfort and skin conformability, while combined with their relatively high costs limit affordability and ubiquitous adoption [4], [5].

Flexible electronics have emerged as a promising solution, offering ultra-thin, lightweight, and stretchable circuits ideal for conformable health-monitoring systems [12]. Built on flexible substrates, flexible integrated circuits naturally conform to body contours, enhancing comfort during extended wear [13]. In this work, we target the FlexIC process of Pragmatic Semiconductor [14], which uses Indium Gallium Zinc Oxide (IGZO) thin-film transistors (TFTs). FlexIC enables rapid production cycles and extremely low-cost, even portable, fabrication [15], [16] of disposable hardware well-suited for inexpensive patches, valuable for both commercial applications and clinical settings. Moreover, its low-temperature fabri-

cation process promotes sustainable manufacturing, achieving orders-of-magnitude reductions in carbon footprint and water consumption compared to traditional silicon technologies [16].

However, flexible electronics face fundamental limitations, including low integration density, large feature sizes, and limited performance [17]. For example, current FlexICs integrate up to 20K NAND2-equivalent gates [17] and operate at frequencies of a few hundred kHz [14]. Additionally, the absence of p-type devices results in resistor-nMOS logic and elevated power consumption. These constraints make the integration of complex circuits—such as the ML classifiers typically required in healthcare wearables—extremely challenging.

Flexible microprocessors have been explored in [15], [18], [19], but they are either too large [18] or lack the computational power to support ML tasks [15], [19]. Ozer et al. [16] presented and fabricated Flex-RV, the first Bendable RISC-V microprocessor. Focusing on area efficiency, Flex-RV builds on the serial RISC-V variant, SERV [20], and integrates a multiply-accumulate (MAC) co-processor to improve system latency for neural network inference. The co-processor in the Flex-RV relies on conventional arithmetic units, overlooking the hardware efficiency of bespoke implementations. Similarly, the latency and energy efficiency of Flex-RV are limited by the need to transfer both activations and weights to its co-processor due to its conventional implementation.

Flexible electronics and bespoke design mutually enable each other. On one hand, the extremely low fabrication costs of flexible electronics, combined with their short production timelines and low non-recurring engineering costs [17], enable the design of bespoke (i.e., fully customized) circuits tailored to the target applications, e.g., ML circuits with hardwired coefficients. On the other hand, the unmatched hardware efficiency of bespoke designs [21]–[30] can help overcome inherent technology limitations, enabling realistic designs and unlocking a wide range of applications for FlexICs, such as ML-based healthcare monitoring.

In this work, we overcome these limitations by enabling bespoke ML acceleration within flexible RISC-V-based systems. We integrate the compact SERV core with a specialized co-processor that, with minimal area overhead, maximizes energy efficiency and minimizes inference latency for MAC-intensive classifiers such as Multi-Layer Perceptrons (MLPs). These design choices are particularly important for flexible healthcare wearables, which are battery-powered or rely on energy harvesters, thus demanding exceptional energy efficiency, low power, and compact circuitry to fit within small, lightweight form factors. Meanwhile, MLPs have been shown to achieve state-of-the-art accuracy in healthcare monitoring

This work was done while Dr. Zervakis was with University of Patras.

applications [31], making our solution well-suited for this domain. Exploiting the low fabrication cost, short lifecycle, and disposable nature of flexible circuits, we tailor the microprocessor design to each target MLP. Aligning the rapid FlexIC production timelines with equally fast design cycles, we propose an automated framework that uses constrained programming and (i) derives the fixed coefficients of the bespoke co-processor for each MLP and (ii) optimally maps MLP inference operations onto the hardware by decomposing each neuron’s products across the available bespoke multipliers.

Post-layout evaluation across healthcare datasets demonstrates that our circuits achieve  $< 1s$  inference latency, i.e., near-real-time operation [31], with an energy consumption of only 0.48mJ per inference, while meeting conformability and stretchability requirements. Compared to the state-of-the-art Flex-RV [16], we achieve 2.35x lower latency and 2.15x lower energy consumption. **Our novel contributions are as follows:**

- 1) To the best of our knowledge, we propose and design the first bespoke accelerated RISC-V microprocessor for flexible ML-based applications.
- 2) We propose an automated methodology to jointly select the constant parameters of the bespoke co-processor and determine the optimal MLP to co-processor mappings, thereby minimizing inference latency and energy.

## II. BACKGROUND

Pragmatic’s Gen3 FlexIC [14] platform enables the fabrication of flexible integrated circuits on 200mm/300mm polyimide wafers. FlexIC employs n-type metal-oxide IGZO TFTs, achieving a minimum channel length of 0.6 $\mu$ m. The polyimide base features a thickness of less than 30 $\mu$ m, allowing for ultra-thin circuits. FlexICs have demonstrated mechanical robustness and can bend without any damage to circuitry [17].

IGZO TFTs combine mechanical flexibility with compatibility for low-temperature, cost-efficient fabrication [16]. By integrating all components directly on the substrate, the process avoids rigid wafers, high-temperature steps, and additional packaging, producing lightweight and conformable devices that are robust under bending and stretching. The streamlined manufacturing significantly reduces prototype development time, allowing lab-scale devices to be fabricated in days rather than months, and typically delivers finished circuits in about six weeks. The intrinsic flexibility further eliminates the need for encapsulation layers, simplifying production and improving reliability under mechanical stress.

## III. PROPOSED BESPOKE ACCELERATED RISC-V

### A. System Overview

Our RISC-V based system for flexible ML-powered wearables is illustrated in Fig. 1. Specifically, we use the SERV core tightly coupled with our bespoke co-processor. SERV employs a bit-serial arithmetic logic unit (ALU), shift registers to minimize hardware complexity, and a simple finite-state machine (FSM) for control. By processing one bit at a time, SERV achieves exceptionally low area and power consumption, making it well-suited for flexible electronics applications.

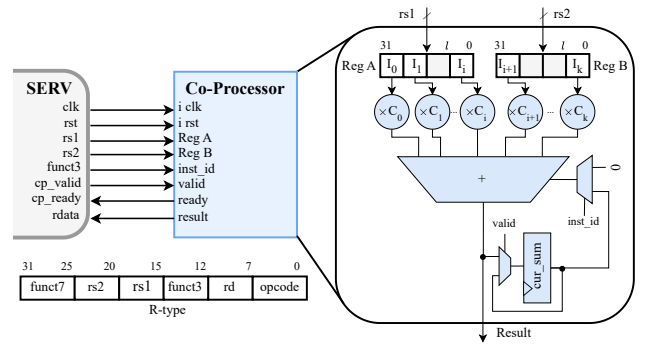


Fig. 1: Our flexible RISC-V-based systems comprising a SERV core and a bespoke co-processor.

Moreover, SERV has already been fabricated with FlexIC and tested under varying tension and compression corners, demonstrating both feasibility and robustness [16]. Compared to other fabricated bendable microprocessors [15], SERV is powerful enough to support more complex requirements, e.g., as in flexible healthcare monitoring wearables.

The proposed co-processor reuses the operands and result signals of R-type instructions and the decoded *funct3* field to distinguish co-processor operations (see Fig. 1). Custom instructions, issued via inline assembly, are added to the SERV ISA and seamlessly supported by minor modifications to SERV’s decode unit, which detects them using the opcode and the 25<sup>th</sup> instruction bit. Upon detection, the decoder triggers the write-back path and notifies the FSM and related modules. Data transfer and synchronization follow a ready/valid handshake: SERV issues transactions only when the co-processor is ready, and the pipeline stalls until a valid response is received.

With the co-processor seamlessly integrated, determining its most efficient architecture is crucial. To maximize hardware efficiency, we propose a bespoke MAC-based co-processor. Specifically, instead of costly conventional multipliers, we use bespoke multipliers, where one operand is a runtime input and the other is a predefined constant. However, this design choice requires efficiently folding the entire MLP inference over the limited set of available constants, a challenge that is addressed, together with optimal constant selection, in Section IV.

Two main benefits are obtained from our proposed design. First, using bespoke multipliers, hardware savings (area and power) are obtained from their simpler implementation. For example, Fig. 2 shows the area of flexible bespoke signed multipliers for 4-bit inputs and compares it with the area of a conventional 4x4-bit signed multiplier. Note that we consider fixed-point arithmetic in ML inference, allowing all operations to be performed entirely with integer operations. As expected, the area of the bespoke multipliers varies with the constant value. Nevertheless, on average, bespoke multipliers require 3x less area. Secondly, since the coefficients are hardwired into the co-processor, the entire SERV-to-ML co-processor bandwidth (i.e., registers *rs1* and *rs2* in each call) can be dedicated solely to transferring inputs (e.g., activations in MLPs). Leveraging this and trading some of the area savings from the bespoke multipliers (see Fig. 2), we can instantiate

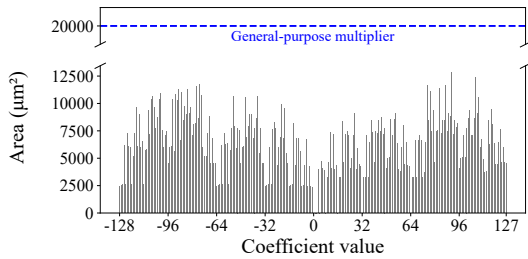


Fig. 2: Area of signed bespoke multipliers in FlexIC technology [14] for varying constants. For reference, conventional 4x4-bit and 8x4-bit multipliers occupy 0.020 mm<sup>2</sup> and 0.038 mm<sup>2</sup>, respectively. Post-synthesis results are reported.

more multipliers, increasing the number of MAC operations per 64-bit transfer from SERV to the co-processor, which can reduce inference latency and lower energy consumption.

### B. Bespoke Co-processor

The proposed co-processor comprises several by-constant multipliers, an adder tree to accumulate the products and add them with the current partial sum, along with two multiplexers for control. The first, controlled by `inst_id` (the `funct3` field in the RISC-V R-type instruction), selects between 0—to start a new weighted-sum computation—and the current sum for ongoing computations. The second signal, controlled by `valid`, ensures that only correct sums, i.e., after `rs1` and `rs2` have been fully received, are written to the `cur_sum` register. For implementation simplicity and to avoid the overhead of multiplexers and control circuitry, each bespoke multiplier input is fixed and directly connected to predefined bit positions of the two input registers. Thus, assuming  $l$ -bit inputs, if the product  $x \times C_0$  is required,  $x$  must be placed at position  $I_0$  of register `reg A` (see Fig. 1), i.e., in the  $l$  most significant bits (MSBs) of the instruction’s `rs1` register.

Although flexible applications are highly resource-constrained, they typically feature low precision requirements [32]. Inputs in the range of 4 or 5 bits have been shown sufficient for most flexible ML applications [17], [21], [26]. As deduced from Fig. 2, the area of our co-processor may vary depending on the constant values selected for the bespoke multipliers. Fig. 3 evaluates configurations with 4-bit inputs and varying number of bespoke multipliers. Specifically, we perform a Monte Carlo analysis, randomly sampling each bespoke multiplier’s constant. As shown, area and power increase with the number of bespoke multipliers and vary with the selected constants. Though, the dominant factor is the total number of multipliers, as evidenced by the minimal overlap among the box plots, primarily due to the growing contribution of the adder tree with more summands.

Fig. 3 also shows the area and power of a conventional co-processor that employs eight 4x4-bit conventional multipliers (red line). Since each conventional multiplier requires 8 input bits in total (4 bits for each operand), only eight such multipliers can be instantiated given that registers `reg A` (`rs1`) and `reg B` (`rs2`) together provide 64 bits. The area and power of this conventional co-processor are 0.23mm<sup>2</sup> and 0.21mW,

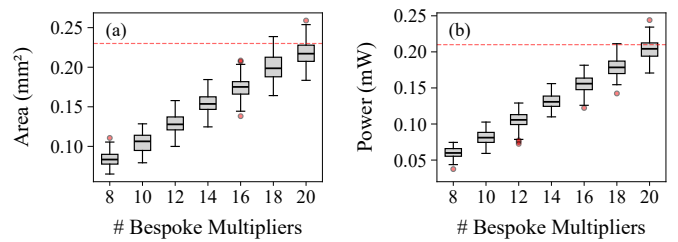


Fig. 3: Monte Carlo analysis of (a) area and (b) power of our bespoke co-processor as a function of the number of bespoke multipliers. Results are on 200 post-synthesis samples with 4-bit inputs and random constants in  $[-8, 7]$ . The red line refers to a co-processor with eight 4x4 conventional multipliers.

respectively. As shown in Fig. 3, for 4-bit inputs, up to sixteen bespoke multipliers can be instantiated without incurring any area or power overhead compared to the conventional co-processor with eight 4x4-bit multipliers. Moreover, since each bespoke multiplier requires only 4 input bits, sixteen is also the maximum number of bespoke multipliers that can be simultaneously fed by `reg A` and `reg B`. Identical results are obtained for 5-bit inputs, where twelve bespoke multipliers can be instantiated, compared to a conventional co-processor that integrates seven 5x4-bit multipliers.

Nevertheless, as aforementioned, one drawback of using bespoke multipliers is that some products may need to be decomposed into multiple partial products if the required constant is not directly available. For example, assume that our co-processor has sixteen constant multipliers, with  $C_0 = 3$ ,  $C_1 = 15$ , and  $C_i \neq 18, \forall i > 1$ . If the product  $x \times 18$  is required, it can be computed by storing  $x$  in both the  $I_0$  and  $I_1$  positions (i.e., the 8 MSBs of the instruction’s `rs1` register). Although this approach produces the correct result, two bespoke multipliers are used to compute a single product.

The key takeaway from Fig. 3 is that, while the selected constant values do influence the area and power of our bespoke co-processor, this variation is relatively constrained for a fixed number of multipliers and can be largely ignored during constant selection. Instead, to design the most energy-efficient system, we should focus on selecting constants that minimize the required decompositions, thereby reducing system latency and improving overall energy efficiency.

## IV. LATENCY-AWARE CO-PROCESSOR OPTIMIZATION

Employing a bespoke co-processor can improve metrics such as area and power. However, the overall system performance and energy efficiency will be primarily determined by the choice of constants for the bespoke co-processor and by how the required operations (e.g., products) are executed over the available bespoke multipliers. This observation may also explain why, despite the wide adoption of bespoke design in small, application-specific flexible circuits [21]–[30], our work is the first to design flexible bespoke accelerated RISC-V microprocessors, which are essential for implementing more complex systems in an area-efficient manner [19], [33], [34].

### A. Optimization Problem Definition

Preserving security and privacy, all optimizations are performed post-training using only the model weights. Within this setting, given a set of weighted sums (e.g., the neurons in an MLP), our goal is to identify a set of constants that minimize the total latency of folding all required weighted sums on a bespoke co-processor configured with these constants, while also accounting for the cost of products that are not directly represented within each sum. Since, as discussed in Section I, we target ML classification using MLPs, we hereafter adopt MLP-specific terminology, though our approach can be seamlessly extended to any ML classifier that relies on MAC computations. The objective of energy-efficient co-processor design can be formulated as an optimization problem that selects the optimal set of constants and corresponding product decompositions to minimize the total number of cycles per inference (i.e., latency, since the co-processor operates at the same clock frequency as SERV regardless of the selected constants) while respecting hardware constraints. We reiterate that, for each target MLP, a unique bespoke co-processor is designed. Formally, the optimization problem is defined as:

$$CSPD : (N, W, C) \mapsto (C', D, L) : \min_{C', D} L, \quad (1)$$

subject to *multipliers constraints*

where  $N$  is the computational graph of the input MLP, consisting of all neurons.  $W = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_r\}$  is the set of weight vectors, with each  $\mathbf{w}_n$  representing the weights associated with neuron  $n \in N$ .  $C = \{c_1, c_2, \dots, c_m\}$  denotes the set of candidate constants for the bespoke multipliers. The solution  $C' = \{(c, k_c) \mid c \in C\}$  specifies the selected constants  $c$  along with their replication factors  $k_c$ , i.e., the number of bespoke multipliers instantiated for each constant. Finally,  $D$  denotes the decompositions and  $L$  the total inference latency.

The problem is structured into two tightly coupled stages: i) from the candidate set  $C$ , a set  $C'$  is selected to instantiate the bespoke multipliers in hardware, and ii) given this configuration  $C'$ , the operations in  $N$  are scheduled onto the available multipliers while respecting dependencies and minimizing overall latency. The bespoke multipliers can implement weights in  $W$  either directly or through decomposition, meaning that the joint choice of constants and decomposition decisions ultimately determines the system latency  $L$ . Thus, solving  $CSPD$  requires finding both an optimal set of constants, along with a valid scheduling strategy, to minimize  $L$ .

### B. Constraint Programming-Based Solution

To solve (1), we employ the  $CP-SAT$  solver from Google's OR-Tools [35], an open-source constraint programming engine consistently ranking among the top performers in the *MiniZinc Challenge* [36], [37]. The solver combines Boolean satisfiability (SAT) solving with constraint-programming-style propagation and learning, enabling efficient pruning of infeasible solutions through conflict analysis and constraint propagation.

We model this problem as a constraint instance by defining: (i) decision variables that capture the design space, (ii) constraints that enforce hardware and dependency requirements,

---

### Algorithm 1 Constraint Programming Model

---

1: **Inputs:**

- $N$ : Set of neurons
- $W$ : Set of weight vectors for each neuron  $n \in N$ 
  - $w_{n,i}$ : the  $i$ -th weight of neuron  $n$
- $C$ : Set of candidate constants (e.g.,  $[-8, 7]$ )
- $M_{\text{total}}$ : Maximum number of bespoke multipliers
- $T = \{1, \dots, T_{\text{max}}\}$ : Set of available scheduling cycles.  $T_{\text{max}}$  can be used as the latency constraint or set to a large value to always enable a solution.

2: **Decision Variables:**

- $k_c$ : Number of multipliers of type  $c$
- $d_{n,t,i,c}$ : Number of multipliers of type  $c$  used in cycle  $t$  for  $w_{n,i}$
- $l_n$ : Latency of neuron  $n$

3: **Latency Definition:**  $l_n = \max\{t \in T \mid \exists i, c : d_{n,t,i,c} > 0\}$

4: **Objective:**  $\min \sum_{n \in N} l_n$

5: **Constraints:**

- **Total multiplier budget:**  $\sum_{c \in C} k_c \leq M_{\text{total}}$
  - **Cycle capacity:**  $\sum_i d_{n,t,i,c} \leq k_c, \quad \forall n \in N, \forall t \in T, \forall c \in C$
  - **Decomposition:**  $\sum_{t \in T} \sum_{c \in C} d_{n,t,i,c} \cdot c = w_{n,i}, \quad \forall n \in N, \forall i$
- 

and (iii) an objective function that drives the minimization of total inference latency. The formal formulation for neuron weight scheduling is presented in Algorithm 1. Here, the total multiplier budget  $M_{\text{total}}$  enforces hardware limits on the number of bespoke multipliers, while the cycle capacity constraint prevents resource overuse in any given cycle. The product (weight) decomposition constraint ensures computational correctness. Decomposition may map a weight directly to an available multiplier with a matching constant or partition it when no exact match exists or when the matching multiplier is already in use, with partitioning possibly unfolding over multiple cycles. Finally, the objective function minimizes the sum of neuron latencies, steering the solver toward minimal total inference time. Feasibility is ensured as, for sufficiently large execution time (large  $T_{\text{max}}$ ), one  $+1$  and one  $-1$  bespoke multiplier suffice to realize any weight via decomposition.

$CP-SAT$  with Algorithm 1 explicitly solves the decomposition problem and the variables  $d_{n,t,i,c}$  (i.e., extracted decompositions for each weight of every neuron) are used to automatically generate the co-processor calls in software. However, the set  $C'$  is also implicitly determined. All constants  $c \in C$  with  $k_c \neq 0$  form the final set  $C'$  and define the constant values hardwired into the bespoke multipliers.

Given the identified set  $C'$ , the Verilog description of our co-processor is automatically generated from our parameterized templates and seamlessly integrated with the SERV core. As concluded in Section III, the specific constant values have less impact on overall system efficiency. Therefore, the candidate set  $C$  is populated according to the weight quantization scheme. For instance, for 4-bit quantized MLP weights, we set  $C = [-8, 7]$ . In a more restrictive case,  $C$  can be set to hardware-friendly constants only (see Fig. 2). Finally, since  $CP-SAT$  solves only single-objective optimization problems, we may explore different trade-offs by running the solver for multiple values of  $M_{\text{total}}$ . With 4-bit inputs we run the solver for  $M_{\text{total}} = 16$ . Exploring smaller  $M_{\text{total}}$  values may optimize energy consumption alongside area and power (see Fig. 3).

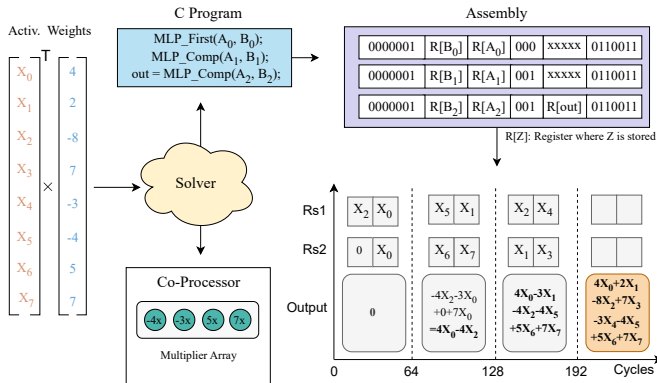


Fig. 4: Cycle-level example of neuron output accumulation.

Fig. 4 presents an illustrative decomposition example. For simplicity, inputs are assumed to be 16-bit (i.e., two inputs per register), and the co-processor features four bespoke multipliers. The solver produces two outputs: (i) the configuration of the by-constant multipliers, which is directly reflected in the generated co-processor, and (ii) a schedule list of decompositions, indicating which products must be computed at each co-processor call (see Fig. 4) by assigning the respective activations to the corresponding register positions.

## V. RESULTS AND ANALYSIS

### A. Experimental Setup

We evaluate our framework on diverse popular healthcare benchmarks (Table I) spanning stress monitoring, cardiology, dermatology, activity recognition, and emotion-aware driving. Specifically, we use WESAD [38], DriveDB [39], StressInNurses [40], and SPD [41] for stress detection; Arrhythmia [42] and ECG5000 [43] for cardiac analysis; Dermatology [44] for skin disease classification; HAR [45] for activity recognition; and AffectiveRoad [46] for driving-related emotion recognition. Each dataset is split into training and testing sets with an 70/30 ratio, normalized to  $[-1, 1]$ , and used to train MLPs in TensorFlow Lite. Considering the strict constraints of flexible electronics, we limit MLPs to a maximum of two hidden layers with up to ten neurons per layer and apply quantization and pruning during training to produce compact models that meet the requirements of flexible applications. For each dataset, we perform a neural architecture search to identify the smallest network that achieves top accuracy, using 4-bit weights and either 4- or 5-bit activations for quantization. Table I reports the extracted MLP topologies and their achieved accuracies. As shown, the optimized FlexIC-oriented MLPs incur at most 0.75% accuracy loss compared to the unpruned floating-point model.

We perform full-chip design (from RTL to GDSII) of our ML-accelerated microprocessors. Synopsys Design Compiler S-2021.06 is used for synthesis and mapping to the FlexIC Gen3 3V standard-cell library [14]. Synopsys VCS T-2022.06 is used for timing simulations, while PrimeTime T-2022.03 for timing and power analysis. Physical implementation is carried out using Cadence Innovus 21.35, where we design the chip

TABLE I: Accuracy Evaluation

Dataset	Topology	#MAC	Acc. FP32 <sup>1</sup> (%)	Acc. QP <sup>2</sup> (%)
AffectiveRoad	63-9-3	594	97.68	97.17
Arrhythmia	279-9-11	2610	70.00	70.00
Dermatology	34-9-6	360	95.06	94.44
DriveDB	61-9-3	576	93.06	92.51
ECG5000	140-3-5	435	92.84	92.10
HAR	561-7-6	3969	93.72	93.18
SPD <sup>3</sup>	75-9-3	702	90.75	90.05
StressInNurses	72-9-3	675	74.11	73.36
WESAD	96-9-3	891	82.56	82.46

<sup>1</sup>Accuracy Floating-point model. <sup>2</sup>Accuracy optimized model.

<sup>3</sup>SPD required 5-bit activations, while all other MLPs use 4-bit ones.

floorplan and power delivery network, followed by placement, clock tree synthesis, routing, and post-routing optimizations. Finally, timing and power sign-off tools are used to accurately report the chip’s delay and power after parasitic extraction. All syntheses and evaluations are conducted at 150kHz, as at this frequency SERV provided a good area and power trade-off.

In our analysis, we design and evaluate one flexible chip per MLP. The bespoke co-processor is generated using our automated optimization flow (Section IV), while the SERV core is based on the FlexIC-modified version of [16], which implements the RV32E ISA, reduces the register file to 16 registers, and employs a custom serial peripheral interface (SPI) to access external memory, overcoming technology limitations but incurring a 46/47-cycle latency per read/write.

For comparisons, we perform a full-chip design of FlexRV [16], which integrates a generic, conventional ML co-processor featuring two 8x4 and two 4x4 multipliers, along with dedicated post-processing hardware for bias addition, and ReLU activation. Additionally, we design a Semi-bespoke system that uses the same SERV core but couples it with a co-processor customized only to the precision required by each trained model. Specifically, for SPD, the Semi-bespoke co-processor integrates seven 5x4 conventional multipliers, while for the other MLPs it uses eight 4x4 conventional multipliers.

Finally, all systems are also prototyped on a Digilent Arty S7-50T FPGA board. Bitstreams are generated using Vivado 2020.1 with default synthesis and place-and-route constraints. Each FPGA prototype is then validated by running inference on the test dataset to verify correct operation.

### B. Evaluation

Table II reports the hardware evaluation of the designed chips. Compared to the vanilla, non-accelerated SERV, our co-processors introduce only a 8% area and 10% power overhead, on average. Still, all our chips maintain a compact footprint, averaging 2.42mm<sup>2</sup>, making them suitable for most flexible wearables, while their average power consumption of 1.5mW allows them to be powered by existing flexible batteries or even flexible energy harvesters [27], [30]. On average, our chips contain 4200 NAND2-equivalent gates, ensuring realistic implementations and their co-processor critical path delay is at most 3.9μs, easily satisfying the 150kHz target frequency. For reference, SERV’s critical path delay is 5.3μs.

TABLE II: Post-layout evaluation. Target frequency 150kHz.

Design	Total			Co-processor					
	Area (mm <sup>2</sup> )	Power (mW)	#Gates	Area (mm <sup>2</sup> )	Power (mW)	Delay (μs)	#Gates		
Ours	AffectiveRoad	2.433	1.511	4235	0.181	0.144	3.9	585	
	Arrhythmia	2.410	1.493	4140	0.158	0.126	3.6	490	
	Dermatology	2.417	1.504	4183	0.165	0.137	3.6	533	
	DriveDB	2.442	1.527	4291	0.190	0.160	3.8	641	
	ECG5000	2.432	1.519	4244	0.180	0.152	3.9	594	
	HAR	2.425	1.506	4208	0.173	0.139	3.6	558	
	SPD <sup>†</sup>	2.413	1.498	4172	0.161	0.131	3.9	522	
	StressInNurses	2.445	1.523	4281	0.191	0.154	3.8	631	
	WESAD	2.443	1.521	4281	0.191	0.154	3.8	631	
	SoA	SERV <sup>§</sup>	2.252	1.367	3650	–	–	–	–
		Flex-RV [16]	2.542	1.627	4531	0.290	0.260	4.8	881
SB* w/ 4x4 x8		2.492	1.587	4635	0.240	0.220	3.7	985	
SB* w/ 5x4 x7		2.476	1.619	4628	0.224	0.252	4.0	978	

<sup>†</sup>SPD uses 12 5-bit bespoke multipliers while all others use 16 4-bit ones.  
<sup>§</sup>SERV delay is 5.3μs. \*Semi-bespoke.

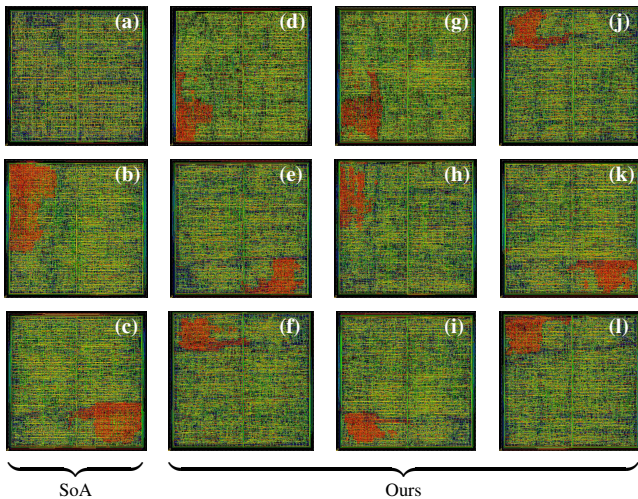


Fig. 5: Layout of (a) SERV (non-accelerated), (b) Flex-RV [16], (c) Semi-bespoke w/ eight 4x4 multipliers, and ours: (d) AffectiveRoad, (e) Arrhythmia, (f) Dermatology, (g) DriveDB, (h) ECG5000, (i) HAR, (j) SPD, (k) StressInNurses, and (l) WESAD. Red regions highlight the co-processor.

Compared to Flex-RV [16] and Semi-bespoke, our chips are slightly smaller and more power-efficient. This is explained by the fact that compared to Flex-RV [16], our co-processor is on average 39% smaller and consumes 45% less power. Similarly, against Semi-bespoke, our co-processor is on average 26% smaller with 36% lower power consumption. Finally, Fig. 5 illustrates the layouts for SERV, Flex-RV [16], Semi-bespoke with eight 4x4 conventional multipliers, and our chips.

Next, we evaluate ML inference on each flexible chip. We run inference on the test datasets of the optimized models in Table I on our chips, as well as SERV, Flex-RV [16], and Semi-bespoke. Table III reports the inference latency and the energy consumed per inference. Latency is measured as the average cycles per sample multiplied by the clock period (i.e., 6.6μs), where cycle measurement includes memory accesses. As shown in Table III, all our flexible chips achieve

TABLE III: Inference latency and energy efficiency evaluation

Dataset	Inference Latency (s)				Energy/Inference (mJ)			
	Ours	[16]	SERV	SB*	Ours	[16]	SERV	SB*
AffectiveRoad	0.639	0.917	647	0.810	0.51	0.74	786	0.66
Arrhythmia	0.999	2.625	2402	1.558	0.66	1.72	2889	1.16
Dermatology	0.504	0.788	404	0.623	0.45	0.64	489	0.60
DriveDB	0.581	1.042	624	0.813	0.47	0.82	757	0.80
ECG5000	0.212	0.499	437	0.434	0.18	0.33	526	0.36
HAR	0.638	3.586	4492	2.082	0.45	2.30	5446	1.37
SPD	0.756	1.675	654	0.976	0.63	1.15	790	0.82
StressInNurses	0.598	0.965	620	0.726	0.50	0.78	747	0.66
WESAD	0.703	1.174	817	0.924	0.50	0.89	985	0.73

\*Semi-bespoke

near-realtime operation, with latencies under 1s. In contrast, Flex-RV and Semi-bespoke violate this 1s constraint in 55% and 22% of the cases, respectively. More specifically, our chips deliver a 1972x speedup over the non-accelerated SERV, 2.35x over Flex-RV [16], and 1.6x over Semi-bespoke.

Our chips consume, on average, as low as 0.48mJ per inference, featuring 53.5% lower energy compared to the state-of-the-art Flex-RV [16] and 39.3% versus the Semi-bespoke. Energy gains are lower than latency gains, since, despite our co-processors being lower power and our chips achieving significantly lower latency, a substantial portion of execution time is spent on SERV due to its serial execution and memory accesses. Still, energy gains are substantial, making our circuit a highly energy-efficient and a promising solution.

Finally, it is noteworthy that the execution time of our optimization framework (Section IV) is negligible, averaging 20 minutes and ranging from 5 to 30 minutes.

## VI. CONCLUSION

In this work, we propose and implement compact and energy-efficient ML-accelerated flexible RISC-V microprocessors, targeting conformable healthcare wearables. Our bespoke MAC-based co-processor for the SERV core, maximizes performance and energy efficiency. Our proposed constrained programming-based optimization automatically selects co-processor constants and decomposes MLP operations on the available hardware, enabling a tailored microprocessor for each model and aligning rapid fabrication with equally fast design cycles. Post-layout results across healthcare datasets demonstrate near-real-time inference within 2.42mm<sup>2</sup>, under the power budget of flexible batteries, and requiring on average 4200 gates, enabling realistic implementations despite the inherent limitations of flexible electronics.

## VII. ACKNOWLEDGMENT

This work is partially supported by the European Research Council (ERC) (Grant No. 101052764) and co-funded by the H.F.R.I call “Basic Research Financing (Horizontal support of all Sciences)” under the National Recovery and Resilience Plan “Greece 2.0” (H.F.R.I. Project Number: 17048).

## REFERENCES

- [1] M. M. Khan and M. Alkhatami, "Anomaly detection in iot-based healthcare: machine learning for enhanced security," *Scientific Reports*, vol. 14, no. 1, p. 5872, 2024.
- [2] K. Mahato, T. Saha, S. Ding, S. S. Sandhu, A.-Y. Chang, and J. Wang, "Hybrid multimodal wearable sensors for comprehensive health monitoring," *Nature Electronics*, vol. 7, no. 9, 2024.
- [3] S. Jeong, H. E. Barkam, S. Yun, Y. Kim, S. Angizi, and M. Imani, "Exploiting boosting in hyperdimensional computing for enhanced reliability in healthcare," in *Design, Automation & Test in Europe Conference (DATE)*, 2025, pp. 1–7.
- [4] V. Mishra *et al.*, "Continuous detection of physiological stress with commodity hardware," *ACM Trans. Comput. Healthcare*, 2020.
- [5] M. Shatta, K. Balaskas, P. C. Lozano Duarte, G. Panagopoulos, M. B. Tahoori, and G. Zervakis, "Invited paper: Feature-to-classifier co-design for mixed-signal smart flexible wearables for healthcare at the extreme edge," in *Int. Conf. Computer Aided Design (ICCAD)*, 2025.
- [6] A. Kumar, K. Sharma, and A. Sharma, "Hierarchical deep neural network for mental stress state detection using IoT based biomarkers," *Pattern Recognit. Lett.*, vol. 145, pp. 81–87, 2021.
- [7] S. A. H. Aqajari, E. K. Naeini, M. A. Mehrabadi, S. Labbaf, A.-M. Rahmani, and N. D. Dutt, "GSR Analysis for Stress: Development and Validation of an Open Source Tool for Noisy Naturalistic GSR Data," *ArXiv*, 2020.
- [8] S. Jiang, F. Firouzi, K. Chakrabarty, and E. B. Elbogen, "A resilient and hierarchical iot-based solution for stress monitoring in everyday settings," *IEEE Internet Things J.*, vol. 9, 2022.
- [9] C. Turetta, F. Demrozi, and G. Pravadelli, "B-har: An open-source baseline framework for in-depth study of human activity recognition datasets and workflows," *IEEE Access*, vol. 12, pp. 166 911–166 922, 2024.
- [10] A. Singh, T. Margaria, and F. Demrozi, "Cnn-based human activity recognition on edge computing devices," in *2023 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, 2023, pp. 1–4.
- [11] L. Borzi *et al.*, "Freezing of gait detection: The effect of sensor type, position, activities, datasets, and machine learning model," *Journal of Parkinson's Disease*, vol. 15, no. 1, pp. 163–181, 2025.
- [12] N. Lovecchio *et al.*, "Flexible electrochemical sensor for detecting chlorine ions in sweat," in *2025 10th International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2025, pp. 1–6.
- [13] W. Gao *et al.*, "Fully integrated wearable sensor arrays for multiplexed in situ perspiration analysis," *Nature*, vol. 529, pp. 509–514, 01 2016.
- [14] Pragmatic, "Flexic Platform Gen3," <https://www.pragmaticsemi.com/foundry/flexic-platform-gen-3>, 2025.
- [15] N. Bleier, C. Lee, F. Rodriguez, A. Sou, S. White, and R. Kumar, "Flexicores: low footprint, high yield, field reprogrammable flexible microprocessors," in *International Symposium on Computer Architecture (ISCA)*, 2022, p. 831–846.
- [16] E. Ozer *et al.*, "Bendable non-silicon risc-v microprocessor," *Nature*, 2024.
- [17] M. B. Tahoori, E. Ozer, G. Zervakis, K. Balaskas, and P. Pal, "Computing with printed and flexible electronics," *IEEE European Test Symposium (ETS)*, 2025.
- [18] J. Biggs *et al.*, "A natively flexible 32-bit arm microprocessor," *Nature*, vol. 595, no. 7868, pp. 532–536, 2021.
- [19] A. Raisiardi, K. Jordanou, J. Kufel, K. Gudimetla, K. Myny, and E. Ozer, "Flexing risc-v instruction subset processors (risps) to extreme edge," 2025. [Online]. Available: <https://arxiv.org/abs/2505.04567>
- [20] O. Kindgren, "Bit by bit: How to fit 8 risc-v cores in a \$38 fpga board," in *RISC-V Workshop*, Zürich, Switzerland, 2019, presentation at the RISC-V Workshop, Zürich.
- [21] E. Özer *et al.*, "A hardwired machine learning processing engine fabricated with submicron metal-oxide thin-film transistors on a flexible substrate," *Nature Electronics*, vol. 3, pp. 1–7, 07 2020.
- [22] E. Ozer, J. Kufel, J. Biggs, J. Myers, C. Reynolds, and G. Brown, "Binary neural network as a flexible integrated circuit for odour classification," in *IEEE International Conference on Flexible and Printable Sensors and Systems (FLEPS)*, 2020, pp. 1–4.
- [23] V. Pillai *et al.*, "arrWNN: Arrhythmia-detecting weightless neural network flexic," in *IEEE International Flexible Electronics Technology Conference (IFETC)*, 2024, pp. 1–4.
- [24] M. H. Mubarak *et al.*, "Printed machine learning classifiers," in *Annu. Int. Symp. Microarchitecture (MICRO)*, 2020, pp. 73–87.
- [25] G. Armeniakos, G. Zervakis, D. Soudris, M. B. Tahoori, and J. Henkel, "Cross-layer approximation for printed machine learning circuits," in *Design Automation and Test in Europe (DATE)*, 2022, pp. 190–195.
- [26] G. Armeniakos, G. Zervakis, D. Soudris, M. B. Tahoori, and J. Henkel, "Co-design of approximate multilayer perceptron for ultra-resource constrained printed circuits," *IEEE Trans. Comput.*, 2023.
- [27] A. Kokkinis, G. Zervakis, K. Siozios, M. B. Tahoori, and J. Henkel, "Enabling printed multilayer perceptrons realization via area-aware neural minimization," *IEEE Transactions on Computers*, 2024.
- [28] V. Mrazek *et al.*, "Evolutionary approximation of ternary neurons for on-sensor printed neural networks," in *International Conference on Computer-Aided Design (ICCAD)*, 2025.
- [29] F. Afentaki, G. Saglam, A. Kokkinis, K. Siozios, G. Zervakis, and M. B. Tahoori, "Bespoke approximation of multiplication-accumulation and activation targeting printed multilayer perceptrons," in *International Conference on Computer-Aided Design (ICCAD)*, 2023.
- [30] F. Afentaki, M. Hefenbrock, G. Zervakis, and M. B. Tahoori, "Embedding hardware approximations in discrete genetic-based training for printed mlps," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2024.
- [31] F. Afentaki *et al.*, "Exploration of low-power flexible stress monitoring classifiers for conformal wearables," in *International Symposium on Low Power Electronics and Design (ISLPED)*, 2025.
- [32] J. Henkel *et al.*, "Approximate computing and the efficient machine learning expedition," in *Int. Conf. on Computer-Aided Design (ICCAD)*, 2022.
- [33] E. Ozer *et al.*, "Bendable non-silicon risc-v microprocessor," *Nature*, vol. 634, no. 8033, pp. 341–346, 2024.
- [34] P. Vergos, T. Vergos, F. Afentaki, K. Balaskas, and G. Zervakis, "Support vector machines classification on bendable risc-v," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, vol. 1, 2025, pp. 1–6.
- [35] L. Perron and F. Didier, "Cp-sat," Google. [Online]. Available: [https://developers.google.com/optimization/cp/cp\\_solver/](https://developers.google.com/optimization/cp/cp_solver/)
- [36] P. J. Stuckey, T. Feydy, A. Schutt, G. Tack, and J. Fischer, "The minizinc challenge 2008–2013," *AI Magazine*, vol. 35, no. 2, pp. 55–60, Jun. 2014.
- [37] M. Team, "The minizinc challenge," MiniZinc, 2025. [Online]. Available: <https://www.minizinc.org/challenge/>
- [38] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. Van Laerhoven, "Introducing wesad, a multimodal dataset for wearable stress and affect detection," in *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, ser. ICMI '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 400–408. [Online]. Available: <https://doi.org/10.1145/3242969.3242985>
- [39] J. Healey and R. Picard, "Detecting stress during real-world driving tasks using physiological sensors," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 156–166, 2005.
- [40] S. Hosseini *et al.*, "A multimodal sensor dataset for continuous stress detection of nurses in a hospital," *Scientific Data*, vol. 9, no. 1, p. 255, 2022.
- [41] T. Iqbal *et al.*, "Stress monitoring using wearable sensors: a pilot study and stress-predict dataset," *Sensors*, vol. 22, no. 21, p. 8135, 2022.
- [42] H. A. Guvenir, B. Acar, H. Muderrisoglu, and J. R. Quinlan, "Arrhythmia," UCI Machine Learning Repository, 1998, DOI: <https://doi.org/10.24432/C5BS32>.
- [43] Y. Chen *et al.*, "The ucr time series classification archive," <https://www.timeseriesclassification.com/description.php?Dataset=ECG5000>, 2015, accessed: 2025.
- [44] N. Ilter and H. Guvenir, "Dermatology," UCI Machine Learning Repository, 1998, DOI: <https://doi.org/10.24432/C5FK5P>.
- [45] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. L. Reyes-Ortiz *et al.*, "A public domain dataset for human activity recognition using smartphones," in *Esam*, vol. 3, no. 1, 2013, pp. 3–4.
- [46] N. E. Haouij, J.-M. Poggi, S. Sevestre-Ghalila, R. Ghazi, and M. Jaïdane, "Affectiveroad system and database to assess driver's attention," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, 2018, pp. 800–803.