Real-Time Personalized Atrial Fibrillation Prediction on Multi-Core Wearable Sensors

Elisabetta De Giovanni, Adriana Arza Valdés, Miguel Peón-Quirós, Amir Aminifar, *Member, IEEE*
David Atienza, *Fellow, IEEE*

***Abstract*—In the recent Internet-of-Things (IoT) era where biomedical applications require continuous monitoring of relevant data, edge computing keeps gaining more and more importance. These new architectures for edge computing include multi-core and parallel computing capabilities that can enable prevention diagnosis and treatment of diseases in ambulatory or home-based setups. In this paper, we explore the benefits of the parallelization capabilities and computing heterogeneity of new wearable sensors in the context of a personalized online atrial fibrillation (AF) prediction method for daily monitoring. First, we apply optimizations to a single-core design to reduce energy, based on patient-specific training models. Second, we explore multi-core and memory banks configuration changes to adapt the computation and storage requirements to the characteristics of each patient. We evaluate our methodology on the Physionet Prediction Challenge (2001) publicly available database, and assess the energy consumption of single-core (ARM Cortex-M3 based) and new ultra-low power multi-core architectures (open-source RISC-V based) for next-generation of wearable platforms. Overall, our exploration at the application level highlights that a parallelization approach for personalized AF in multi-core wearable sensors enables energy savings up to $24\,\%$ with respect to single-core sensors. Moreover, including the adaptation of the memory subsystem (size and number of memory banks), in combination with deep sleep energy saving modes, can overall provide total energy savings up to $34\,\%$, depending on the specific patient.**

***Index Terms*—Parallel computing, atrial fibrillation prediction, personalized, real-time, wearable sensors, multi-core**

I. Introduction and Motivation

In the last years, wearable technologies for health monitoring applications have gained a lot of attention in the commercial and research field [1]. Wearable sensors allow monitoring specific characteristics of a patient's pathology by measuring bio-signals with non-invasive sensors, e.g., electrocardiogram (ECG) and electroencephalogram (EEG). Therefore, they can enable accurate detection and prediction of major noncommunicable diseases (NCDs) and allow people to self-assess of their health status [2]–[7].

Despite recent advances in wearable technologies, major challenges exist to fully exploit such systems. In particular, energy efficiency and scalability (i.e., according to the specific pathology characteristics of each patient) are important factors to take into account in any wearable sensor design [8] for personalized remote long-term health monitoring [5]–[7], [9]–[11]. Modern ultra-low power (ULP) platforms [12]–[16] can offer many advantages in terms of parallelization capabilities that can be exploited in biomedical applications. Moreover, these platforms include direct memory access (DMA) modules that are more efficient than the main processors in data transfers. Additionally, they contain SRAMs structured in independent banks that can be power-gated depending on the application needs identified at design-time [12], [13] or at run-time [17].

In this work, we consider the prediction of atrial fibrillation (AF) as a case study for personalized online health monitoring on single- and multi-core platforms. In fact, the heterogeneous nature of AF requires a personalized diagnosis and treatment [18], using excerpts of ECG signals. During the initial stage of AF, an intermittent or paroxysmal form occurs. For paroxysmal atrial fibrillation (PAF), there exists a single oral dose of pharmacological cardioversion [19] to reduce the frequency of the episodes and prevent PAF recent-onset [20]. Therefore, there is a clear need for enabling continuous personalized monitoring of PAF patients, which can be feasible by exploiting novel ULP multi-core computing platforms.

In the context of PAF long-term monitoring, we propose a methodology to design a new online PAF prediction model targeting scalable computation on modern ULP wearable sensors, which considers the specific features of the individuals and their condition. The scalability is driven by the adaptive algorithm and architecture parameters, which affect the design in single- and multi-core platforms to reduce energy consumption for each individual patient. In order to draw a comparison with our method, we report two key cases of the recent literature [21], [22], describing offline methodologies, which include inter-patient variability and achieve higher accuracy compared to other methods [23]–[25]. Finally, we compare the accuracy results with our previous work on offline PAF prediction [26].

To the best of our knowledge, no study has been done on PAF prediction on wearable sensors with emerging multi-core computing platforms. Furthermore, the energy-scalability analysis of a personalized PAF prediction method, running on embedded devices with parallel computing capabilities and memory scaling, has not yet been tackled. Therefore, the main contributions of this work are the following:

- We design an online energy-efficient PAF prediction to

E. De Giovanni, A. Arza, M. Peón-Quirós, D. Atienza are with the Embedded Systems Laboratory (ESL), EPFL, Lausanne, Switzerland.

A. Aminifar is with the Embedded Systems Laboratory (ESL), EPFL, Lausanne, Switzerland, and also with the Department of Electrical and Information Technology, Lund University, Sweden.

This work has been partially supported by Swiss NSF ML-Edge Project (Grant No. 200020_182009), in part by the MyPreHealth (Grant no. 16073) project funded by Hasler Stiftung, and in part by the ONR-G (Award No. N62909-20-1-2063).

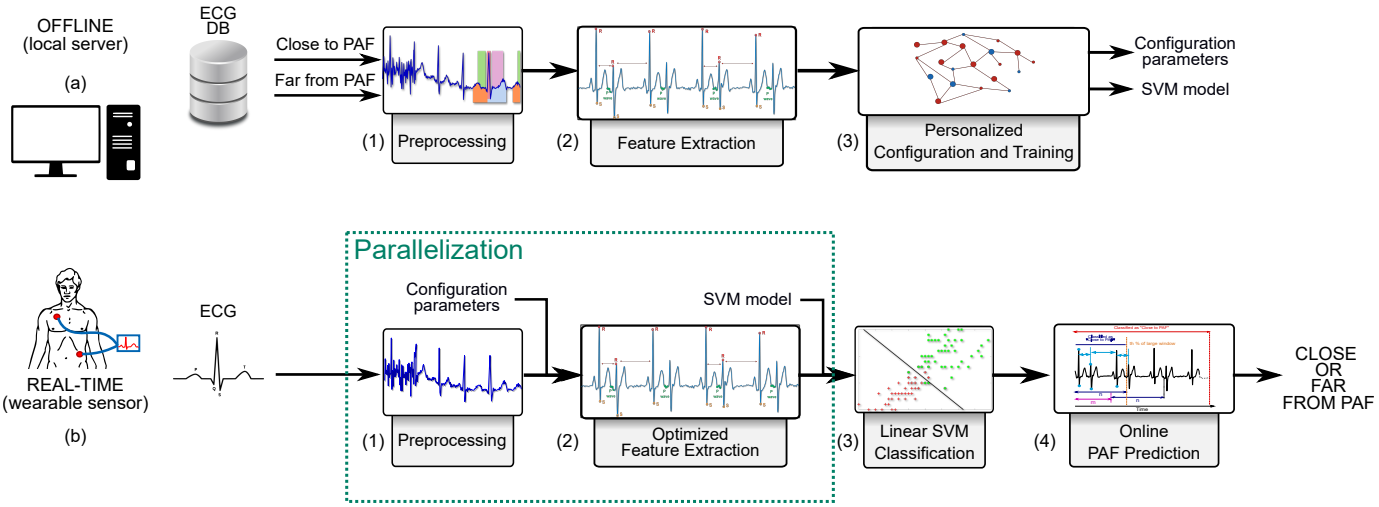

Fig. 1. Block diagram of the personalized PAF prediction. On top (a), the blocks of the configuration and training phase. At the bottom (b), the real-time prediction process executed on a wearable sensor.

be implemented on a single-core ULP wearable sensor INYU [9], which we personalize according to the characteristics of each patient. The optimized and personalized model allows to reduce the energy consumption and processing execution time, by considering the constraints of the wearable sensor. Additionally, we exploit the existing low-power sleep modes between sample acquisition to achieve a scalable battery lifetime of, at least, 37 days.

- We develop a personalized parallelization technique for new open-source multi-core RISC-V based computing architectures that can be included in wearable sensors, by using the GAP8 platform [14]. Our technique scales with the number of cores, i.e., distributing the computation among cores, according to a patient-specific model. Our proposed parallelization achieves energy savings of up to $24\,\%$ compared to the single-core design.
- We explore the memory design space to execute personalized AF algorithms in multi-core IoT and wearable platforms [14], by scaling the size of the memory banks ($8\,$KiB, $4\,$KiB, $2\,$KiB, and $1\,$KiB) and storing buffers of different lengths according to the number of cores. Also, we highlight the energy consumption reduction thanks to deep sleep modes that exploit the specific characteristics of the patient. Overall, the personalized multi-core design provides up to $34\,\%$ energy savings with respect to recent single-core wearable sensors.

The rest of the paper is structured as follows. Section II describes the proposed method, distinguishing between the offline learning phase and the real-time detection phase. Then, Section III explores different techniques to reduce energy consumption in modern ultra-low power platforms. Section IV presents the patient-specific optimizations targeting single-core and multi-core platforms. Next, Section V describes the experimental setup used in this work and Section VI presents the accuracy and energy consumption of the proposed method in the two selected platforms. Finally, Section VII summarizes the main conclusions.

II. Personalized PAF Prediction Method for Long-Term Monitoring

The personalized PAF prediction approach presented in this paper is designed to be used for long-term monitoring. First, we train a personalized model on a set of ECG signals previously acquired from a single patient. Then, the model is applied to a newly acquired ECG signal from the same patient, producing an output at least every $15\,$s to $45\,$s.

Fig. 1 shows our methodology for a single patient. In Fig. 1a, we show the offline method running on a local server, which performs the personalized analysis on previously acquired ECG data for the target patient. These data need to include one excerpt of ECG signal before the onset of a PAF event (i.e, "Close to PAF"). Additionally, we need one excerpt of normal sinus ECG signal at least $45\,$min far from any event (i.e., "Far from PAF"). We choose a training window of at least 350 R peaks for both excerpts, which corresponds to 3–9 min considering a heart rate varying from $40\,$BPM to $110\,$BPM, as done in [26]. Moreover, to avoid overfitting and ensure robustness and generalization of the training model we perform a learning curve analysis [27] on the number of training samples.

Referring to Fig. 1a, the ECG signals are, first, preprocessed by filtering and delineating their main waves, in Step 1. Then, the methodology extracts features from small windows of consecutive beats to capture short events that can happen before a PAF onset (Step 2). The features are the input to what we call "personalized configuration and training", in Step 3. The training is done using a linear classifier based on support vector machines (SVMs) and a 5-fold cross-validation (CV). The output of the offline process is a subset of optimal features specific to the patient and the corresponding classification model. After the personalized configuration parameters, the selected features and the classification model are loaded in the embedded device, which runs the automatic PAF prediction, shown in Fig. 1b. Once the device acquires a small window of consecutive beats of ECG signal, the algorithm starts the preprocessing (Step 1) and extraction of a reduced set of features

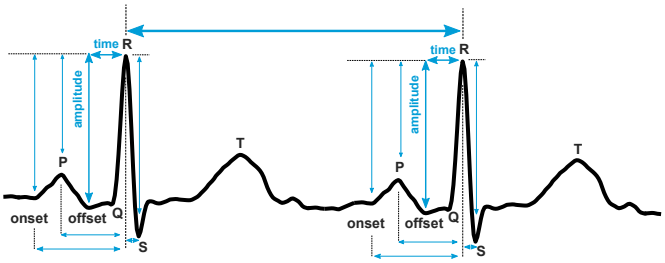

Fig. 2. Feature extraction on a small window of two consecutive beats.

personalized to the patient (Step 2). Finally, by using a linear SVM classification model (Step 3), the algorithm performs an online PAF prediction producing a binary output, as shown in Step 4. Additionally, the preprocessing and feature extraction steps are suitable for parallelization (cf. Section IV-B).

A. Preprocessing – Filtering and Delineation

In both phases of our methodology we start by preprocessing the signal. The common step to both phases is the filtering, which consists of removing the baseline wandering by applying a morphological filter. We use an implementation proposed by Sun et al. [28] and optimized for an embedded system by Braojos et al. [29]. The morphological filter removes the peaks and valleys of the signal with operations related to the shape or morphology of the signal features. Then, the signal baseline is finally subtracted from the original signal. This step can be parallelized for multiple leads [13], although in our case we only require one single-lead, hence we do not parallelize it.

For the delineation of both offline and online real-time processing, we use sequentially three methods. First, we use a real-time implementation of the wavelet transform (WT) delineation to detect the R peak [30], [31] for a design on standard platforms. This method uses the WT of a signal, which is proportional to its derivative with a smoothing impulse response at different scales. By applying the WT to the signal, the R-peaks are detected as the zero-crossings that are common across scales 2^1 through 2^4 (where most of the ECG signal energy lies [30]), and which are preceded by a positive peak and followed by a negative peak. Moreover, for the multi-core design we consider a more light-weight R peak detection, called REWARD [32], which is more suitable for parallelization. REWARD includes two main steps: signal enhancement and R peak detection. The signal enhancement method is called Relative Energy (Rel-En) and it uses the signal energy to amplify dominant peaks. The R peak detection step generates a set of adaptive hysteresis thresholds to isolate the highly dominant peaks and performs a subsequent check on their widths to eliminate false positives. Second, we apply a method described in our previous work [26] to detect the onset and offset of the P wave by comparing it with a triangular wave starting at the P peak and finishing at the isoelectric line. Third, we delineate the S wave as the minimum point after the R peak, by considering the standard physiological duration of the QRS complex.

B. Personalized Optimized Feature Extraction

The features considered in our prediction algorithm depend directly on the fiducial points delineated, as shown in Fig. 2. Specifically, we consider the distance in time from the main fiducial points analyzed (P wave onset, peak and offset, and S) to the R peak within the same beat, and beat-to-beat RR intervals if the fiducial point is the R peak itself. Additionally, we consider the signal amplitude of the five fiducial points related to the amplitude of the closest R peak. Moreover,

Algorithm 1 OFFLINE: Personalized Feature Selection

1: **Input**: windows of n consecutive beats of the filtered ECG; m beats sliding window
2: **Output**: selected $group$ of fiducial points
3: **for** $window = 1, 2, ..., \frac{ECGlength}{m}$ **do**
4: **for** $beat = 1, 2, ..., n$ **do**
5: Extract R peaks
6: Extract P, P onset, P offset, S
7: **end for**
8: **end for**
9: Consider 5 groups of fiducial points: (R); (P, R); (P, R, S); $(P, Pon, Poff, R)$; $(P, Pon, Poff, R, S)$
10: **for** $group = 1, 2, ..., 5$ **do**
11: Train and test SVM with 5-fold CV
12: Compute mean F-score on folds
13: **end for**
14: Choose $group$ with best mean F-score
15: **return** $group$

we select a personalized combination of features in a specific amount of consecutive beats that defines a small window of processing. In this way we analyze sudden events occurring within the window, as shown in our previous work [26]. Additionally, we include an overlapping window depending on the patient. By considering groups of features depending on the main ECG waves, we train the algorithm to choose the set that gives the best 5-fold cross-validation performance for each patient (F-score), as shown in Lines 9–14 of Algorithm 1. Then, in the online phase, we only delineate the fiducial points related to the selected group of features. The personalized features selection has the advantage of scaling the computation of the ECG delineation in the wearable sensor by performing a selective delineation for each patient.

C. Personalized Classification Parameters

The offline personalized configuration and training, shown in Fig. 1a, includes several steps, other than the feature selection. The personalization of the model for each patient captures the heterogeneity of the PAF pathology and optimizes its implementation on wearable sensors. We use a minimum time resolution window for the prediction of 25 beats. This value varies within 15 s and 45 s, considering a resting heart rate in the range of 40 BPM to 110 BPM, which is a fair prediction window length considering the physiology of the PAF event occurrence. The first set of configuration parameters includes the training model coefficients. We choose to use a supervised learning model based on SVMs to classify an excerpt of

Algorithm 2 OFFLINE: Personalized Configuration and Training

1: **Input**: filtered ECG; $n = 3:7$ consecutive beats; $m = 1:n$ sliding window; selected $group$; $predwlen = 25beats$
2: **Output**: selected (n, m); selected th; SVM model (mdl)
3: **function** SMALLWINDOWPARAMETERS
4: **for** $(n, m) = (3, 1), (4, 2), ..., (7, n)$ **do**
5: Compute fitting and F-score on learning curves
6: **end for**
7: Choose (n, m) with best fitting and F-score
8: **end function**
9: **function** THPARAMETER
10: $nw = \frac{predwlen}{m}$ ▷ small windows in $predwlen$
11: **for** $th = 0.1, 0.2, ..., 0.9$ **do**
12: Train and test SVM with 5-fold CV
13: Count small windows "Close to PAF" ($swpaf$)
14: **if** $\frac{swpaf}{nw} > th$ **then**
15: "Close to PAF"
16: **else**
17: "Far from PAF"
18: **end if**
19: Compute mean F-score on folds
20: **end for**
21: Choose th with best mean F-score
22: **end function**
23: Train SVM mdl with selected $group$, (n, m), th
24: **return** (n, m); th; mdl

signals labeled "Close to" or "Far from" PAF events. An SVM-based model is selected since at inference time it has very low complexity [33] and it achieves a high accuracy in our offline approach [26]. The second set of personalized configuration parameters includes the amount of consecutive beats (n) within a small window and the length of the sliding window in beats (m). These two parameters vary within the following ranges: $n = 3, 4, ...7$ and $m = 1, 2, ...n$ and affect the number of training samples. The best combination (n, m) was chosen by selecting the most robust model performance using leaning curve analysis [27] and F-score. In our methodology, summarized in Lines 3–8 of Algorithm 2, we analyze the F-score against the number of samples with an increment of 10 samples at each iteration applying a 5-fold CV for better generalization of the results. Next, within the predefined minimum prediction window, we cross-validate a threshold on the set of small windows the model classifies correctly, based on (n, m), described in detail in Lines 9–22 of Algorithm 2. The final step of the learning phase consists in choosing the amount of minimum prediction windows of ECG samples needed to process and predict the PAF onset, where abnormal events may occur. Considering the heterogeneity of the occurrence of small and sudden events [34], our algorithms learn on both normal sinus rhythm (i.e., signal far from any PAF event) and close to a PAF event to choose the prediction time length for each patient. The algorithm computes the maximum amount of consecutive prediction windows (each of 25 beats) in the normal sinus rhythm misclassified as "Close to PAF", using excerpts from a different set than the training and test ones. Therefore, the updated minimum window of prediction is a multiple of 25 beats, which corresponds approximately to the range from $15\,\mathrm{s}$ to $45\,\mathrm{s}$.

During the online PAF prediction monitoring, shown in Fig. 1b, once the personalized features within the small window (of n consecutive beats) are extracted, they are fed to the linear SVM model. Then, the algorithm classifies each small window as either "Close to PAF" or "Far from PAF". Next, our algorithm is optimized to stop the processing once the precomputed personalized threshold of small windows classified as "Close to PAF" is reached, reducing the energy consumption and scaling it for different patients. The final output of the classification is transmitted at least every $15\,\mathrm{s}$ to $45\,\mathrm{s}$ depending on the minimum window of prediction selected.

III. ULP PLATFORMS FOR WEARABLE SENSORS

Modern platforms designed for ULP operation combine several techniques to achieve high computing performance when required, while reducing their overall energy consumption. In the following paragraphs, we describe how we exploit these techniques to reduce energy consumption in the context of multi-core wearable sensors. Then, we analyze the main characteristics of the new open-source RISC-V based ULP multi-core architectures that can be used in next-generation wearable sensors (such as GAP8 [14], which is the multi-core platform that we use for the experimental validation of our ULP parallelization methodology). Additionally, we describe the architecture of a standard single-core wearable sensor.

A. Techniques to reduce energy consumption in multi-core wearable sensors

1) Multi-core Parallelism: ULP multi-core platforms improve the execution of inherently parallel computations, such as, multi-lead ECG signal processing [13], by distributing the work among several cores. This distribution of tasks allows the system to meet all the biosignal processing deadlines, while operating at a lower clock frequency and at a lower voltage level, which results in significant energy savings.

2) Efficient Hardware-Based Synchronization: Parallelization often requires synchronization between the cores that intervene in a computation. In fact, without a low-overhead synchronization mechanism, increasing the parallelization level (i.e., number of cores that work on a task) will not provide energy benefits. Hence, new ULP multi-core architectures include hardware-based single-cycle synchronization mechanisms. Furthermore, they also support clock-gating of cores while they are waiting for an event in the parallelized biosignal processing algorithms.

3) Multi-Banked Memories: The division of the platform memories into smaller banks that can be independently powered off, on, or placed into retention mode, enables fine-grained control on memory energy use. In the context of healthcare wearable sensors, this feature enables the processing of the acquired input biosignals in "windows" that drive which banks are written by the DMA (active), which ones

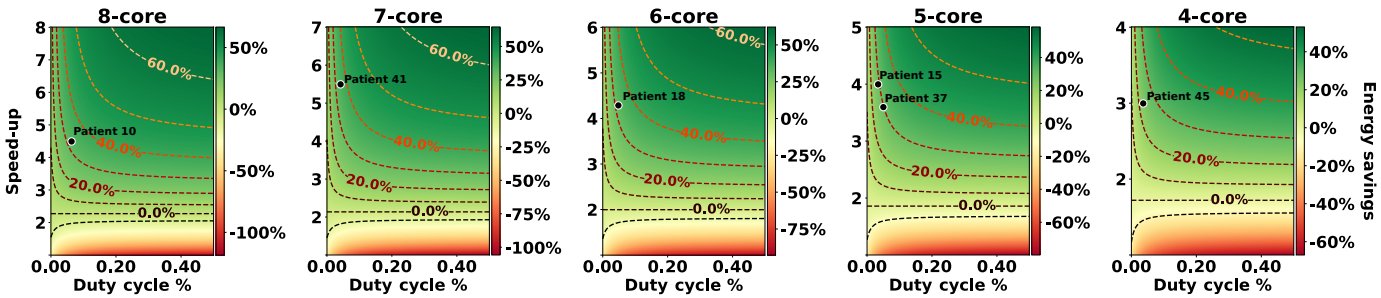

Fig. 3. Potential energy savings in the multi-core GAP8 platform according to the duty cycle of the application and the attainable speed-up compared to the single-core implementation. We report 5 configurations with a varying number of active cores in the cluster from 8 to 4. The dotted contour lines represent different levels of energy savings, which include the efficiency border between single and multi-core computation for GAP8 at 0 %. Additionally, we mark the working point for the six subjects studied in Section VI.

contain data to retain until the next processing interval, and which banks can remain off. Furthermore, the use of banks of different sizes enables an appropriate data placement for input data of accessed biosignals into smaller banks, which consume less energy per access (cf. Section VI-C).

4) Sleeping Modes: Low-power wearable sensors typically have the ability of power-gating many components independently to suppress leakage current. However, the time required to reactivate the element sets a lower limit for the duration of the sleeping periods. Modern ULP platforms use technology-level techniques to reduce the reactivation time, which enables sleeping even during short intervals. This characteristic can be exploited in biomedical applications to synchronize the use of the ULP multi-core architecture to the computing requirements of the different biosignals processing phases.

5) Heterogeneity: Having different groups of resources, or even resources of different types, enables matching the correct resource for each biosignal computation. In this way, ULP multi-core platforms, such as the RISC-V based PULP [12], propose a division of the platform into a simpler processor for tasks with low computation requirements and a second cluster of processors that can be activated to cope with more complex tasks. As we show in this paper, this type of heterogeneous multi-core architectures can be used very efficiently in the context of variable-complexity parallelized biosignal processing algorithms.

6) DMA Modules: Advanced DMA modules implement data transfers between different memory modules, i.e., implementing double-buffering. Additionally, DMA engines are simpler and consume less energy than general-purpose processors. Thus, they are crucial to achieve low power operation in biosignal processing applications that capture windows of data before processing, as cores can be deactivated until enough samples have been acquired for AF prediction for each patient.

B. ULP RISC-V Multi-Core Architectures: The GAP8 Sensor

GAP8 [14] is a commercial RISC-V implementation based on the PULP project [12] and built on a 55 nm technology. It consists of a main processor, termed "fabric controller" (FC), which performs short tasks and manages the complete platform, and a cluster of eight additional cores. The cluster cores can cooperate on a single task (e.g., with OpenMP) or work independently on different ones (e.g., in consumer-producer patterns). A hardware event unit implements single-cycle synchronization, while clock-gating the waiting cores.

The memory is divided into two blocks: The first one, an L2 (512 KiB), is connected to the FC, whereas the second one, an L1 (64 KiB), is connected to the cluster cores. Data transfers between both levels are performed by a dedicated DMA module. This organization allows the application to activate the cluster cores only during highly demanding phases and placing the currently processed data, which normally receives more accesses, in the banks of the smaller memory in the cluster. A second DMA module is in charge of the data acquisition from the ECG leads to the L2 memory. Thanks to it, the FC can remain clock-gated during the acquisition process.

In order to estimate the energy consumption during our experiments, we profile the active cycles of the different cores. Then, we use the reported power numbers for GAP8 [14]. For the FC, we use the lowest possible voltage supply (1.0 V), with an operating frequency of 150 MHz, to compute the execution time of our biosignal processing and AF prediction algorithms, and the lowest power leakage of 3.6 µW, since the deep sleep mode mostly dominates the consumption. Furthermore, the GAP8 platform contains a 30 µW fully retentive memory of 512 KiB that can be divided in 4 banks of 128 KiB. Then, since our personalized and parallelized AF prediction algorithm does not need more than 64 KiB of storage, we reduce the overall memory to 64 KiB, thus reducing the consumption to 10 µW, accounting for the leakage and the memory retention. Additionally, for the ULP multi-core wearable design we explore bank size scaling from 8 KiB, 4 KiB, 2 KiB to a minimum of 1 KiB.

1) Energy savings versus resources assigned: One of the main goals of this paper is to show how a patient-specific assignment of resources in a multi-core platform, like GAP8, achieves significant energy savings. To this end, we did a preliminary analysis on the estimated energy consumption of the GAP8 platform by varying the resources assigned in terms of number of active cores, application duty cycle (i.e., percentage of application computational load in 1s) and parallel implementation speed-up. In this analysis, the energy consumption estimation accounts for the processing and idle time with 1 memory bank active out of the 4 banks available

in GAP8. Fig. 3 shows the energy savings compared to the single-core implementation in the 5 possible multi-core configurations of our design space with a varying number of active cores from 8 to 4. Each configuration reports the percentage of energy savings achieved at different attainable speed-up figures from 1 to 8, while the application duty cycle values can vary between 0 % to 0.5 %, according to the typical intervals that a single-lead ECG-based biomedical application has. Thus, the attainable speed-up for a specific number of cores is a measure of efficiency of the parallel application. However, as Fig. 3 shows, there is a threshold of speed-up that the parallel application needs to reach to start achieving energy savings compared to the single-core implementation. Nonetheless, interestingly enough, this threshold significantly varies with the number of active cores, namely, from 2.3 × for an 8-core implementation to 1.7 × for a 4-core implementation. Therefore, below the speed-up threshold the single-core implementation is more efficient than a particular multi-core implementation, while above the threshold, the multi-core option is more efficient.

In addition in Fig. 3, we report six cases of the analyzed dataset [34] to cover a wide range of computational requirements (i.e., including best, average and worst case scenarios) with different patients (cf. Section V-B and Section VI). In our design space, the duty cycle of the parallel application varies from 0 % to 0.07 %, while we assign between 4 and 8 cores depending on the window length (i.e., number of consecutive beats), thus achieving different speed-ups from 3 × to 5 ×. As a result, Fig. 3 indicates that the absolute energy savings increase by adding more active cores. Nonetheless, if the parallel application achieves its highest speed-up for a certain number of cores, for example 3 × in the 4-core configuration, it is then more energy-efficient to assign the lowest possible number of cores (i.e., it is better to use 4 cores instead of 5 or more for Patient 45). Moreover, within one configuration, for example the 5-core implementation in Fig. 3, in the context of limited duty cycle values, the case with higher duty cycle but lower speed-up (Patient 37) achieves higher energy savings than the case with lower duty cycle and higher speed-up (Patient 15). We further discuss the implications of this analysis on the six reported cases in Section VI-C.

C. Traditional Single-Core Homogeneous Wearable Sensors

As an instance of the previous generation of single-core homogeneous low power sensors we use an EFM32LG-STK3600 containing an EFM32 TM Leopard Gecko 32-bit microcontroller unit (MCU) [35], a 48 MHz ARM Cortex-M3 processor with a 3 V supply, 256 KiB flash memory and 32 KiB RAM. The platform and the corresponding Simplicity Studio software energy profiler are used for our energy consumption analysis. In particular, we estimate how much energy the MCU consumes over time considering its active mode, while running the PAF prediction process. Finally, we assess the use of power management techniques in real-life by using the SmartCardia ECG-based device [9], since it includes an MCU of the same family of the EFM32.

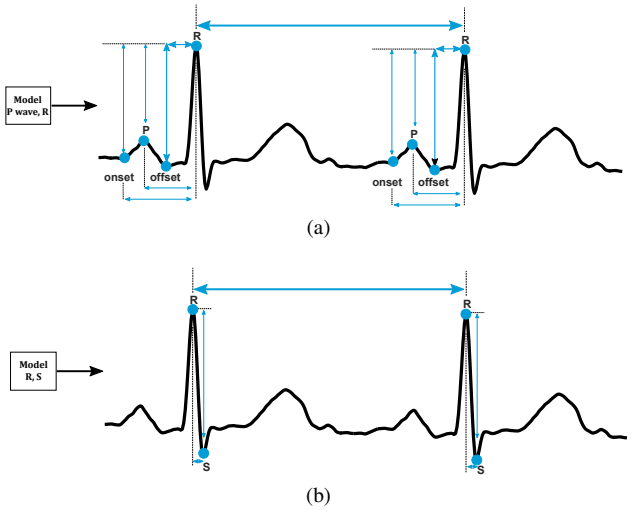

Fig. 4. Proposed selective online feature extraction within a beat for two patients with different training models.

IV. Patient-Specific Optimizations for Single and Multi-Core Ultra-Low Power Platforms

The parameters and model selected during the learning phase, described in Section II-C, are the configuration inputs to apply in the algorithm implemented on a wearable embedded device. With the final goal of saving energy for a personalized continuous monitoring, we apply a set of optimizations to implement an online method for a single-core platform. Then, we design a patient-specific parallelization technique targeting a multi-core platform and we apply memory and power management.

A. Patient-Specific Online Design for Single-Core platforms

Considering the model selected at training time for each patient, we present two main optimization techniques that decrease the computation within a window of analysis.

1) Selective Feature Extraction: As described in Section II-B, the algorithm only extracts for each patient the group of features within n consecutive beats with a sliding window of m beats, which were selected in our personalized patient configuration phase (see Section II-B). Aiming to save and scale computation, hence energy, the algorithm delineates within each consecutive beat only the fiducial points needed for the ECG waves selected at training time.

Fig. 4 describes the strategy used to optimize the delineation within a beat for two different patients and in input their corresponding selected group of features. In Fig. 4a, the trained group of features are within the P and R wave, while in Fig. 4b they are within the R and S wave. As presented in [26], the method to compute the P wave onset/offset compares the corresponding wave and a set of triangular waves with origin in the peak, which end in different points of the isoelectric line. This is computationally more expensive than finding a minimum such as S. Therefore, if for the patient in Fig. 4b the algorithm would delineate the three main ECG waves the computation load will be much higher. However if the personalized model for a patient includes full P wave feature set, as shown in Fig. 4a, then the algorithm will compute them.

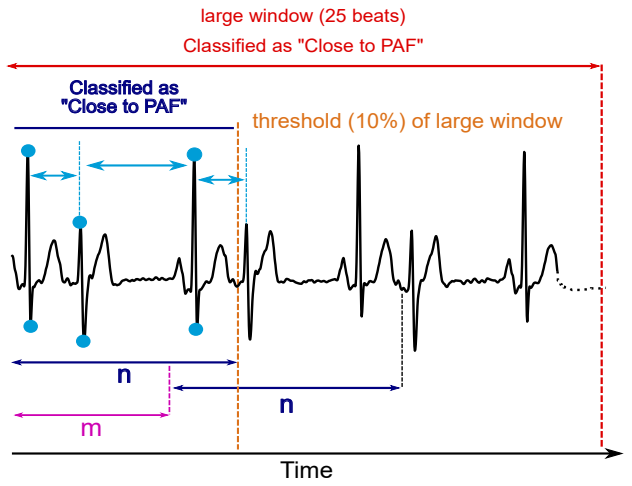

Fig. 5. Proposed optimized online prediction of a PAF event for Patient 1 of the chosen dataset (cf. Section V-B). The configuration parameters are $(n, m) = (3, 2)$, $th = 10\%$.

2) Optimization on the Classification Threshold: By applying the threshold on the small windows processed within a minimum window of prediction, defined in Section II-C, we increase the computational savings in the online PAF prediction process. As an example, Fig. 5 describes the real-time prediction on an excerpt of ECG signal before a PAF onset for Patient 1 of the chosen dataset (cf. Section V-B). As described in Section IV-A1, the algorithm extracts patient-specific features based on the fiducial points for each beat of the small window of consecutive beats. For this patient, the small window consists of three consecutive beats (n) with a sliding window of two (m) and the features extracted are time and amplitude of the R peaks and the S wave. Once the features are extracted, they are fed to the linear SVM model, which classifies the small window as "Close to PAF". Then, using the training parameter (th), the algorithm computes and checks the percentage of small windows classified as positive over a large window of 25 beats to define the whole large window as positive. In the case in Fig. 5, the algorithm reaches th with only one small window, i.e approximately $10\,\%$ of 25 beats. Till the end of the current large window, the algorithm stops extracting features and performing the classification. It starts over at the next large window. This implementation allows to save energy within a window of analysis, i.e., 25 beats, by stopping the feature extraction and classification. Finally, this process is different for each patient enabling scalability, thus the adaptability of the device lifetime (cf. Section VI-B).

B. Patient-Specific Parallelization for ULP Multi-Core Platforms

Considering the number of consecutive beats selected at training time for each patient (n), we apply a patient-specific parallelization by varying the number of cores depending on n. Our method consists in a three-step parallelization process by using $\#cores = n + 1$, where n changes depending on the patient. The filtering step was designed to be parallelized on leads [13] and since we use a single-lead ECG, this step

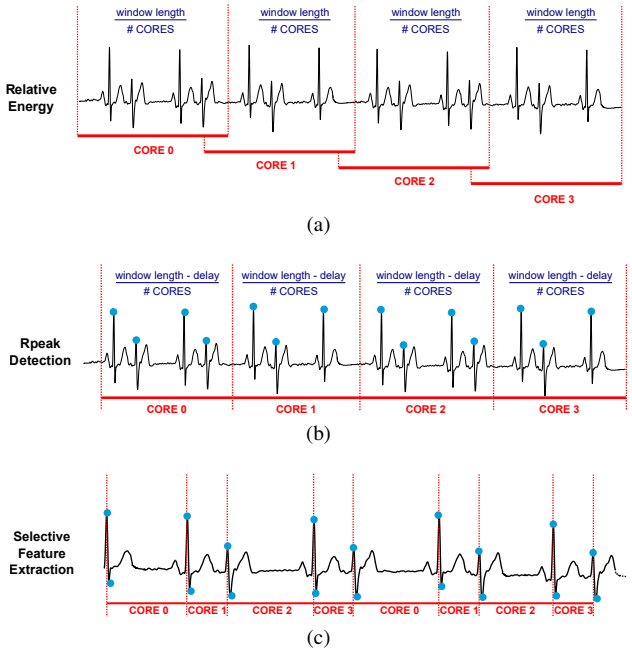

Fig. 6. Proposed personalized parallelization by choosing the number of cores based on n number of consecutive beats to analyze, so that $\#cores = n+1$.

is not parallelized. The three steps that can be parallelized are the signal enhancement preprocessing, Rel-En, and the R peak detection steps of REWARD [32] and the selective feature extraction (cf. Section IV-A1). Since the selective feature extraction includes the computation of the RR interval, the algorithm needs $n + 1$ consecutive beats to extract all the necessary features. Considering that the n parameter varies between 3 and 7, we can assign from 4 to 8 cores for the multi-core implementation. In order to parallelize on the feature extraction, we need to collect $n + 1$ consecutive beats, hence, the multi-core implementation needs to collect a specific window of analysis and then process the data. In this way, we exploit the characteristics of modern ULP architectures, by storing a buffer using the DMA until it reaches the desired length, processing at the lowest voltage but the maximum operating frequency and enabling a faster computation.

Fig. 6 shows the three steps of the algorithm that are parallelized. In this case $n = 3$, therefore $\#cores = 4$. Since the first two steps are part of the REWARD algorithm for R peak detection, we can define the window length of analysis based on how many peaks the R peak detection can detect. The R peak detection needs a minimum window length of $1.75\,\mathrm{s}$ to detect at least one peak. Since we need 4 peaks the window of analysis is $(1.75 * 4)\,\mathrm{s}$. Moreover, the RelEn step computes the energy of the signal for each sample by using the information of a specific window of $0.95\,\mathrm{s}$ [32]. Therefore, the final window length for this specific case is $(1.75 * 4 + 0.95)\,\mathrm{s}$. Fig. 6a shows the parallelization applied to the RelEn step. For this case we can parallelize on 4 cores by dividing the window of analysis in 4 and assigning different windows to different cores, with an overlap. Fig. 6b shows the parallelization applied to the R peak detection step. In this step the cores are assigned to 4 windows of $1.75\,\mathrm{s}$ subtracting the delay of $0.95\,\mathrm{s}$. Finally, the last step is the

selective feature extraction. This is performed within a peak-to-peak interval and each core is assigned to one of the beats within the small window of analysis, containing in this case four peak-to-peak intervals. Since within $1.75\,\mathrm{s}$ there might be more than one peak, the feature extraction will reassign the cores to the next small window of three consecutive beats. All in all, the proposed parallelization exploits the patient-specific model parameters to save energy for an optimal personalized continuous monitoring in an ULP multi-core platform.

1) Memory and Power Management: Considering the modern ULP multi-core wearable sensors, in particular the GAP8 architecture [14], we can apply memory management to the multi-core design to save energy during the signal buffering, which depends on the patient model. Considering the GAP8 L2 memory characteristics described in Section III-B, we explore the possibility of reducing the total L2 memory size, and therefore, the bank size to meet the conditions of our application. First, for the single-core design, we consider a total of $64\,\mathrm{KiB}$ as a baseline, increasing the number of banks to 8 and reducing the bank size to $8\,\mathrm{KiB}$, since we need up to $48\,\mathrm{KiB}$. Then, for the multi-core design we explore lower bank sizes of $4\,\mathrm{KiB}$, $2\,\mathrm{KiB}$, $1\,\mathrm{KiB}$ with a total of $32\,\mathrm{KiB}$, and increasing the number of memory banks. Considering smaller bank sizes, we are able to power off the unneeded banks, based on the window length of analysis, thus on the patient model. Moreover, since the buffer requires more banks we can alternate retention/active mode for buffer slices already stored. Hence, if for two patients we use windows of analysis from $4\,\mathrm{KiB}$ to $8\,\mathrm{KiB}$, by reducing the bank size resolution (for example to the minimum of $1\,\mathrm{KiB}$) we can have a more efficient patient-specific memory management and overall reduction in energy consumption. Finally, our design includes switching to deep sleep mode between the acquisition and storage of two samples, since the signal sampling frequency is low.

V. Experimental setup

In this section we describe the database used for training the models and the test bench for both single and multi-core design discussed in Section III.

A. Database for Offline Training and Online Testing

We apply our framework to the PAF Prediction Challenge (2001) Physionet database [34], containing 53 patients affected by PAF. The database includes two 30-minute ECG signals close to and far from a PAF event, acquired at a sampling frequency of $128\,\mathrm{Hz}$ for each patient, that we resample at $250\,\mathrm{Hz}$. For both signals we train the personalized model on the last 350 beats of the recording (approximately 3–$9\,\mathrm{min}$ considering a heart rate range from $40\,\mathrm{BPM}$ to $110\,\mathrm{BPM}$). For the signals close to a PAF event, we test on the remaining of the recording. For the signal far from any event, we use two thirds of the remaining signal to configure the minimum window of prediction (cf. Section II-C) and then test on the remaining third.

B. Test Bench and Platforms for Single-Core Design

The single-core design is a sample-by-sample method. We use two different R peak detection algorithms, a wavelet-based as done in the previous work [26] and REWARD [32]. We show the overall accuracy on the full database with both wavelet and REWARD algorithms. Then, we choose six cases that vary in terms of configuration parameters to evaluate one window of analysis when a PAF event occurs. We consider the window length related parameters n and m, the selected group of features and the classification threshold, described in Section II-B and Section II-C. Specifically, we select from a worst to a best case of computation in the context of the target multi-core architectures exploration, considering the sum of each configuration parameter computational cost. Finally, the window of analysis varies from $15\,\mathrm{s}$ to $45\,\mathrm{s}$, depending on the patient. We measure the energy consumption of the single-core wavelet-based design using the Simplicity Studio software energy profiler on the Cortex-M3 based EFM32LG-STK3600 (cf. Section III). Then, we use this energy measurement to show the battery lifetime estimation on the real-life SmartCardia INYU ECG-based wearable device [9]. Finally, we run the energy profiling on the single-core REWARD-based design for the six cases on the Cortex-M3 platform.

C. Test Bench for Multi-Core Design

We use the same six cases of the Physionet dataset within the same window of analysis used in the single-core design for the multi-core test bench. The multi-core method is designed to collect a window buffer depending on the number of cores (i.e., number of consecutive beats different for each patient). In order to observe the energy savings, we implement a window-based single-core design, similar to the multi-core one but running on the main core of GAP8 [14], instead of the cluster of cores. We use an open source SDK that simulates a RISC-V PULP platform [36] to profile the window-based single-core and multi-core designs. Then, we estimate the energy consumption using the power numbers for the GAP8 platform provided by [14], running at $1\,\mathrm{V}$ and $150\,\mathrm{MHz}$ in the SoC and at $90\,\mathrm{MHz}$ in the cluster. To account for the memory and power management we estimate the energy spent in storing the window buffer with different bank sizes and entering deep sleep mode between the samples.

VI. Experimental Results

In this section, we first report the results on prediction performance of our real-time personalized approach, after the optimizations described in Section IV, and we compare it with the state-of-the-art offline algorithms. Then, we report the energy consumption of our online single-core design. Finally, we show the energy savings between a personalized window-based on single- and multi-core designs.

A. Accuracy of the PAF Event Prediction

In Table I we compare the accuracy of the two inter-patient variability approaches presented by Martínez et al. [21] and Ebrahimzadeh et al. [22], the offline personalized algorithm

TABLE I
PERFORMANCE SCORE FOR PATIENT-SPECIFIC (OFFLINE AND REAL-TIME) AND INTER-PATIENT VARIABILITY APPROACHES

	Inter-patient variability		Personalized		
Evaluation parameters	[21] Martínez et al. (%)	[22] Ebrahimzadeh et al. (%)	[26] Offline (%)	Real-time with wavelet-based [31] (%)	Real-time with REWARD [32] (%)
Accuracy	93.0	98.2	97.1	93.4	91.5
F-measure	–	–	97.1	93.6	91.6
Sensitivity	–	100.0	96.2	96.2	92.5
Specificity	–	95.5	98.1	90.6	90.6

TABLE II
AVERAGE CURRENT CONSUMED BY DIFFERENT MODULES OF THE INYU IN THE WORST CASE

		Current (mA)	Duty cycle (%)	Average current (mA)
ECG acquisition	ADS1291 active [37]	0.427	100	0.427
Accelerometer (idle)	MPU600 [38]	0.005	100	0.005
Data processing	PAF prediction process	10.5	3.2	0.336
Power saving	Low-Power sleep mode [39]	0.018	96.8	0.017
BLE	NRF8001 [40]	0.014	100	0.014
Total				**0.799**

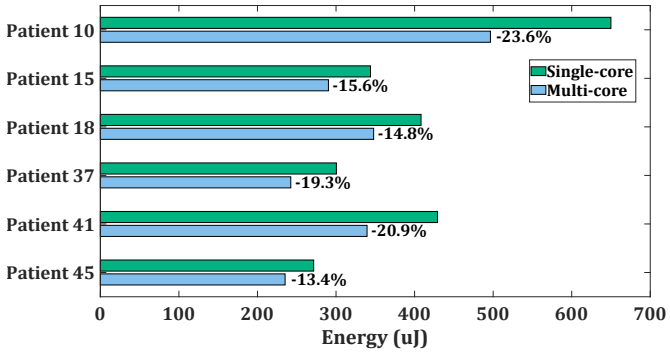

Fig. 7. Energy consumption in µJ during processing for the selected six cases for single-core and multi-core.

presented in our previous work [26], our real-time optimized personalized single-core approach considering the wavelet-based R peak detection [31], and the optimized version with the REWARD algorithm [32] presented in Section IV-B. Moreover, we report sensitivity and specificity of the personalized approaches.

The online single-core implementation using the wavelet-based delineation reduces the accuracy by $4\,\%$ compared to the reference offline algorithm [26], while keeping the same sensitivity. However, the accuracy is comparable with the state-of-the-art offline algorithms. We also used a more light-weight and suitable for parallelization R peak detection [32] with a $2\,\%$ reduction in accuracy due to the misdetection of peaks.

B. Energy Consumption in Standard Single-Core platforms

In the EFM32LG-STK3600 Cortex-M3 based sensor, the energy consumed within one window of analysis varies between $16\,\mathrm{mJ}$ and $9\,\mathrm{mJ}$ for the worst case and best case scenario. The personalized training of the optimized PAF prediction model (cf. Section IV-A) allows energy savings between patients with $84\,\%$ to $56\,\%$ difference compared to the worst case of the six selected cases. Then, using the energy results of EFM32, we estimate the battery lifetime of the different components and execution modes of the real-life ULP ECG monitoring device, INYU [9], which uses the same MCU family to run our PAF prediction algorithm. We show the worst case in terms of computational burden, calculated as execution time of the processing part over the actual prediction time. Since we used the maximum operating frequency of $48\,\mathrm{MHz}$ on the EFM32 and the STM32 on the INYU device can operate up to $32\,\mathrm{MHz}$, we compute a $1.5\times$ execution time increase with respect to EFM32.

Table II shows the worst case of total average current consumed, among the six cases chosen. The duty cycle represents the percentage of computational load of the five main modules of the INYU over a 30-minute window. The ECG signal acquisition is always on for the 30-minute window, as well as the BLE module to send the output of the classification every $15\,\mathrm{s}$ to $45\,\mathrm{s}$. In addition, by computing the total average current of $0.799\,\mathrm{mA}$ consumed along $30\,\mathrm{min}$ and a battery of $710\,\mathrm{mA\,h}$, the battery lifetime of the worst case is 889 hours (approximately 37 days) in the six cases analyzed. In the best case, the battery lifetime is 41 days.

The results demonstrate how a personalized online PAF prediction algorithm in single-core can save energy depending on the patient features. Next, we show how our personalization can be applied to adapt an ULP multi-core architecture to the characteristics of each patient.

C. Energy Consumption in an ULP Multi-Core Architecture

1) Energy Savings in Personalized Multi-Core design: The ULP multi-core architecture described in IV-B assigns different number of cores depending on the patient model. In this section, we discuss the energy savings derived by applying the patient-specific parallelization.

Fig. 7 shows the energy consumption of the six cases of the test bench. We report the energy consumed in the single-core window based design compared to the multi-core. For the six cases, the multi-core design performs better than the single-core design with energy savings from approximately $13\,\%$ up to $24\,\%$. The difference between the patients is directly depending on the corresponding training model and patient-specific parallelization. Since the number of cores depends on the small window length of n consecutive beats, the personalized selection of the window length affects directly the energy consumption. To describe this effect, we can consider the different configurations of the model described in Fig. 3 (cf. Section III-B1). In this figure, we report the cases

TABLE III
SUMMARY OF THE ENERGY SAVINGS DURING PROCESSING FOR THE SIX ANALYZED CASES COMPARED TO THE APPLICATION FEATURES

	Application features				Results energy processing		
Patient	# Active cores	Speed-up	Duty cycle parallel (%)	Parallel code (%)	Single-core energy (uJ)	Multi-core energy (uJ)	Energy savings (%)
10	8	4.53 ×	0.069	43.05	649.9	496.6	23.6
15	5	3.96 ×	0.038	28.95	343.9	290.3	15.6
18	6	4.20 ×	0.036	27.04	408.2	347.9	14.8
37	5	3.63 ×	0.044	38.15	300.5	242.4	19.3
41	7	5.46 ×	0.041	33.73	429.1	339.5	20.9
45	4	3.02 ×	0.039	30.43	271.5	235.2	13.4

TABLE IV
ENERGY SAVINGS WITH MEMORY MANAGEMENT CONSIDERING DIFFERENT BUFFER LENGTHS AND BANK SIZES

		Bank size (KiB)			
		8	4	2	1
Buffer size (KiB) (patient-specific)	7.5	18.2%	26.1%	27.8%	**29.8%**
	4.8	10.3%	19.6%	24.0%	24.9%
	5.7	10.3%	19.2%	21.2%	23.4%
	4.8	10.3%	19.6%	21.5%	23.7%
	6.6	10.3%	18.93%	20.87%	23.03%
	4	0%	11.1%	15.9%	**16.9%**

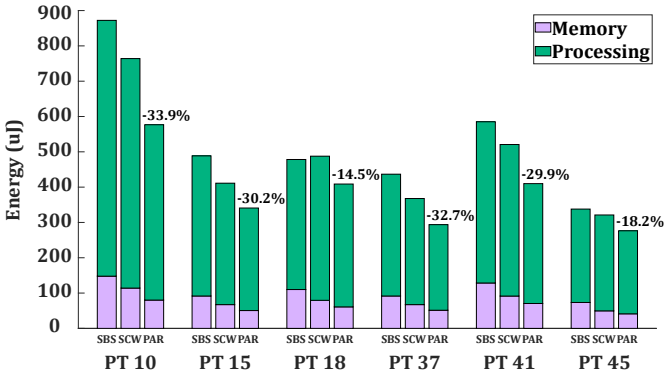

Fig. 8. Energy consumption in µJ during processing for the selected six cases ("PT" as "Patient") for single-core sample-by-sample (SBS), single-core window based (SCW) and parallel (PAR).

analyzed by considering only the potential energy savings of the parallel implementation, while Fig. 7 shows the energy savings of the full implementation. Moreover, in Fig. 3 we consider also the energy consumed in idle time with 1 bank active. However, the final results in energy savings reflect the analysis previously mentioned. Furthermore, Table III reports a summary of the application features for the six analyzed cases and the corresponding energy results during processing, according to the analysis done in Section III-B1.

Let us consider two examples within the 5-core configuration, namely, Patient 15 and Patient 37, which have a duty cycle of $0.038\,\%$ and $0.044\,\%$, respectively, for the parallel implementation. Even though the case of Patient 15 reaches a higher speed-up of $3.96\times$ ($3.63\times$ for Patient 37), the case of Patient 37 achieves higher energy savings. Moreover, if we consider the case of Patient 45, which achieves a speed-up of $3.02\times$ with only 4 cores active, the parallel implementation (that only represents $30\,\%$ of the total computation) achieves better energy savings compared to a higher number of active cores for the same speed-up. Finally, if we consider the case of Patient 10, implemented in a 8-core configuration, it has a duty cycle of $0.069\,\%$ (approximately $43\,\%$ of the total computation) and it reaches a speed-up of $4.53\times$. This case achieves the highest overall energy savings of $23.6\,\%$, which is explained by the higher value of duty cycle compared to the other cases. In fact, the speed-up is comparable to the case of Patient 18, implemented with a 6-core configuration, but its duty cycle is double. Moreover, if we compare the case of Patient 10, Patient 37 and Patient 41 (duty cycle of $0.041\,\%$ and highest speed-up of $5.46\times$ with a 7-core configuration), it is possible to observe the effects of the trade-off between the three variables analyzed in Section III-B1. Indeed, as shown in Table III, the three cases achieve high and comparable energy savings due to a varying number of active cores, speed-up and duty cycle. This effect is visible also in the cases of lower energy savings in Fig. 3 and the corresponding values in Table III.

The results show how the patient-specific multi-core design allows to scale the computational load by assigning the number of cores depending on the personalized application model, and, therefore, scale and save energy.

2) Energy Savings with Memory Banks Management: As described in Section IV-B1, we apply memory management scaling the bank sizes of the GAP8 L2 memory to $8\,\mathrm{KiB}$, $4\,\mathrm{KiB}$, $2\,\mathrm{KiB}$ and $1\,\mathrm{KiB}$. In Table IV, we show the results of the energy savings for the memory and sleep mode in a window of analysis compared to the single-core window based design. The single-core bank size is $8\,\mathrm{KiB}$ with the necessary number of banks powered on depending on the patient. The results show two orthogonal levels of energy savings. The first one is the buffer length which depends on the number of cores, hence, varying from $4\,\mathrm{KiB}$ to $7.5\,\mathrm{KiB}$ in this application. On this level, by keeping the bank size fixed for all the buffer size cases we get energy savings up to $18\,\%$ compared to the single-core design. The second level is fixing the buffer size and varying the bank size. By scaling from $8\,\mathrm{KiB}$ to $4\,\mathrm{KiB}$, the memory management reaches energy savings up to approximately $10\,\%$. Further scaling to $8\,\mathrm{KiB}$ or $1\,\mathrm{KiB}$ results in a significant improvement, up to $17\,\%$ (i.e., for the smallest buffer length of $4\,\mathrm{KiB}$). Overall, the platform can get up to $30\,\%$ of savings during buffer storage and sleeping between samples compared to the single-core design by scaling the bank size to the minimum of $1\,\mathrm{KiB}$.

3) Comparison of Total Energy Consumption in Single and Multi-Core Design: We combine the results of the energy consumed during the processing and in storage and deep sleep mode to show the overall savings of the multi-core design

compared to the single-core design. Additionally, we show the energy consumed in the single-core original sample-by-sample design. Fig. 8 shows the energy consumed in µJ for the sum of processing (green) and memory and sleep (pink). The energy consumed during the processing is $6\times$ of the energy consumed in the memory and sleep, in all the cases. Therefore, the impact of the memory management is reduced to 5–6%. The multi-core design reduces the energy consumption more in the cases with more computational load (i.e., Patient 10) up to $33.9\,\%$. The minimum value of energy savings reached in the six cases analyzed is $14.5\,\%$ compared to the single-core designs. In this case the single-core sample-by-sample designs can be more or less efficient than the window based because of different conditions. First, since the MF filtering is always run on the SoC this step accounts for $57\,\%$ to $73\,\%$ of the total computation, which varies depending on the patient model. Second the model itself varies the computational load considering the different configuration parameters for each patient (i.e.,the features extracted, the small window of consecutive beats and sliding window, and the classification threshold). Finally, the multi-core design uses the L1 memory to store the buffers and variables used by the parallelization steps. Accessing the L1 memory from the cluster is more efficient than accessing the L2 from the SoC. The results show how a multi-core design where number of cores and memory bank sizes are scaled according to personalized models for biomedical applications is advised in modern ULP platforms. From the results, even if the parallelization step counts for only $30\,\%$ of the total computation, the energy savings of the two combined optimizations show a sufficient improvement compared to the single-core design.

VII. Conclusion

In this paper we have proposed an online, energy-efficient and personalized PAF prediction method targeting emerging ULP wearable sensors. Our method enables energy savings for a continuous PAF event monitoring in single-core resource-constrained wearable sensors, and scales its energy consumption depending on patient's characteristics. By considering our algorithm running on the INYU wearable ECG-monitoring sensor, we calculated a battery lifetime of at least 37 days. Next, we showed the portability, energy saving and scalability of our algorithm on the new generation of ULP multi-core architectures, based on the ULP GAP8 IoT architecture. We proposed a parallelization technique that assigns the number of cores depending on each patient's characteristics. Moreover, we explored the use of memory bank sizing (from $8\,\mathrm{KiB}$ to a minimum of $1\,\mathrm{KiB}$) and buffer length sizing for different number of cores, specific to each patient. Our final ULP multi-core design combining processing and memory scalability achieves up to $34\,\%$ of energy savings with respect to the original single-core sensor design. A dynamic reconfiguration of the personalized parameters and resource assignment after the occurrence of new PAF events is a highly suitable extension of this work in future research.

Acknowledgment

The authors acknowledge the help of Mr. Fabio Montagna to efficiently use the PULP platform in this work, and Dr. Eric Flamand for providing the additional power numbers for the GAP8 platform.

References

[1] M. Haghi *et al.*, "Wearable devices in medical Internet of Things: scientific research and commercially available devices." *Healthcare informatics research*, vol. 23, no. 1, pp. 4–15, Jan. 2017.

[2] C. C. Cheung *et al.*, "The emerging role of wearable technologies in detection of arrhythmia," *Can. J. Cardiol.*, vol. 34, no. 8, pp. 1083 – 1087, May 2018.

[3] D. Sopic *et al.*, "Touch-based system for beat-to-beat impedance cardiogram acquisition and hemodynamic parameters estimation," in *Proc. of DATE*, vol. 1, no. 1. IEEE/ACM Press, Mar. 2017, pp. 6. 150–155.

[4] D. Sopic *et al.*, "Hierarchical cardiac-rhythm classification based on electrocardiogram morphology," in *Proc. of CinC*, Sep. 2017, pp. 1–4.

[5] D. Sopic *et al.*, "Real-time classification technique for early detection and prevention of myocardial infarction on wearable devices," in *Proc. of BioCAS*, vol. 1, no. 1. IEEE, Oct. 2017, pp. 1–4.

[6] T. N. Gia *et al.*, "Energy efficient fog-assisted IoT system for monitoring diabetic patients with cardiovascular disease," *Future Generation Computer Systems*, vol. 93, pp. 198 – 211, Apr. 2019.

[7] F. Forooghifar *et al.*, "A self-aware epilepsy monitoring system for real-time epileptic seizure detection," *Mobile Networks and Apps*, pp. 1–14, Aug. 2019.

[8] A. Sinha *et al.*, "Algorithmic transforms for efficient energy scalable computation," in *Proc. of ISLPED*. ACM Press, Jul. 2000, pp. 31–36.

[9] G. Surrel *et al.*, "Online Obstructive Sleep Apnea Detection on Medical Wearable Sensors," *IEEE Trans. Biomed. Circuits Syst.*, pp. 1–12, May 2018.

[10] V. Kartsch *et al.*, "Ultra Low-Power Drowsiness Detection System with BioWolf," in *Int. IEEE/EMBS Conf. on Neural Engineering (NER)*. IEEE, Mar. 2019, pp. 1187–1190.

[11] M. Magno *et al.*, "Self-sustainable smart ring for long-term monitoring of blood oxygenation," *IEEE Access*, vol. 7, pp. 115 400–115 408, Jul. 2019.

[12] F. Conti *et al.*, "PULP: A ultra-low power parallel accelerator for energy-efficient and flexible embedded vision," *J. Signal Process. Sys.*, vol. 84, no. 3, pp. 339–354, Sep. 2016.

[13] L. Duch *et al.*, "HEAL-WEAR: An Ultra-Low Power Heterogeneous System for Bio-Signal Analysis," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 9, pp. 2448–2461, Sep 2017.

[14] E. Flamand *et al.*, "GAP-8: A RISC-V SoC for AI at the edge of the IoT," in *IEEE Int. Conf. on Application-specific Systems, Architectures and Processors (ASAP)*, Jul. 2018, pp. 1–4.

[15] S. Benatti *et al.*, "Online learning and classification of EMG-based gestures on a parallel ultra-low power platform using hyperdimensional computing," *IEEE TBioCaS*, vol. 13, no. 3, pp. 516–528, Jun. 2019.

[16] V. Kartsch *et al.*, "Biowolf: A sub-10-mw 8-channel advanced braincomputer interface platform with a nine-core processor and ble connectivity," *IEEE Trans. on Trans. Biomed. Circuits Syst.*, vol. 13, no. 5, pp. 893–906, Oct. 2019.

[17] D. Atienza *et al.*, "Systematic dynamic memory management design methodology for reduced memory footprint," *ACM Trans. on Design Automation for Embedded Systems*, vol. 11, no. 2, pp. 465–489, Apr. 2006.

[18] P. Kirchhof *et al.*, "2016 ESC Guidelines for the management of atrial fibrillation developed in collaboration with EACTS," *Eur. Heart J.*, vol. 37, no. 38, pp. 2893–2962, Oct. 2016.

[19] I. A. Khan, "Pharmacological cardioversion of recent onset atrial fibrillation," *Eur. Heart J.*, vol. 25, no. 15, pp. 1274–1276, Aug. 2004.

[20] P. Alboni *et al.*, "Outpatient treatment of recent-onset atrial fibrillation with the pill-in-the-pocket approach," *N. Engl. J. Med.*, vol. 351, pp. 2384–2391, Dec. 2004.

[21] A. Martínez *et al.*, "Alteration of the P-wave non-linear dynamics near the onset of paroxysmal atrial fibrillation," *Med. Eng. Phys.*, vol. 37, no. 7, pp. 692–697, Jul. 2015.

[22] E. Ebrahimzadeh *et al.*, "Prediction of paroxysmal Atrial Fibrillation: A machine learning based approach using combined feature vector and mixture of expert classification on HRV signal," *Comput. Methods Programs Biomed*, vol. 165, pp. 53–67, Oct. 2018.

[23] W. Zong *et al.*, "A methodology for predicting paroxysmal atrial fibrillation based on ECG arrhythmia feature analysis," in *Proc. of CinC*, vol. 28. IEEE, 2001, pp. 125–128.

[24] M. Mohebbi *et al.*, "Prediction of paroxysmal atrial fibrillation based on non-linear analysis and spectrum and bispectrum features of the heart rate variability signal," *Comput. Methods Programs Biomed.*, vol. 105, no. 1, pp. 40–49, Jan. 2012.

[25] K. Boon *et al.*, "Paroxysmal atrial fibrillation prediction based on HRV analysis and non-dominated sorting genetic algorithm III," *Comput. Methods Programs Biomed.*, vol. 153, pp. 171–184, Jan. 2018.

[26] E. De Giovanni *et al.*, "A Patient-Specific Methodology for Prediction of Paroxysmal Atrial Fibrillation Onset," in *Proc. of CinC*, vol. 44, Sep. 2017, pp. 285–191.

[27] C. Perlich, "Learning curves in machine learning," in *Encyclopedia of Machine Learning*, C. Sammut *et al.*, Eds. Springer US, 2010, pp. 577–580.

[28] Y. Sun *et al.*, "ECG signal conditioning by morphological filtering," *Comput. Biol. Med.*, vol. 32, pp. 465–79, Nov. 2002.

[29] R. Braojos *et al.*, "Embedded real-time ECG delineation methods: A comparative evaluation," in *Proc. of BIBE*. IEEE, Nov. 2012, pp. 99–104.

[30] J. P. Martínez *et al.*, "A wavelet-based ECG delineator: evaluation on standard databases." *EEE Trans. Biomed. Eng.*, vol. 51, no. 4, pp. 570–81, apr 2004.

[31] F. Rincón *et al.*, "Development and evaluation of multilead wavelet-based ECG delineation algorithms for embedded wireless sensor nodes," *IEEE Trans. Inf. Technol. Biomed.*, vol. 15, no. 6, pp. 854–863, Nov. 2011.

[32] L. Orlandic *et al.*, "REWARD: Design, optimization, and evaluation of a real-time relative-energy wearable r-peak detection algorithm," in *Presented at 41st Int. Conf. EMB, Berlin*. IEEE, 23-27 Jul. 2019.

[33] B. Scholkopf *et al.*, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[34] G. Moody *et al.*, "Predicting the onset of paroxysmal atrial fibrillation: the Computers in Cardiology Challenge 2001," in *Proc. of CinC*, vol. 28. IEEE, 2001, pp. 113–116.

[35] Silicon Labs, "EFM32 Leopard Gecko Reference Manual," Apr. 2017.

[36] (2019) GitHub - pulp-platform/pulp-sdk.

[37] Texas Instruments, "Low-Power,2-Channel,24-Bit AnalogFront-End for Biopotential Measurements," Sep. 2012.

[38] InvenSense, "MPU-6000 and MPU-6050 product specification revision 3.4," Aug. 2013.

[39] STMicroelectronics, "STM32L151RD - Ultra-low-power ARM Cortex-M3 MCU with 384 Kbytes Flash, 32 MHz CPU, USB, 3xOp-amp - STMicroelectronics," Oct. 2017.

[40] Nordic Semiconductor, "nRF8001 single-chip Bluetooth low energy solution," Mar. 2015.

Elisabetta De Giovanni received the M.Sc. degree in Biomedical Engineering, with specialization in Technology for Healthcare, at University of Pavia, Italy, in April 2016. She was one of 25 italian students selected for a project challenge with IBM foundation in Italy to design accessible software. She is currently a PhD candidate in the Embedded Systems Laboratory (ESL) at École polytechnique fédérale de Lausanne (EPFL). Her work involves system level co-design of smart ultra-low power multi-parametric wearable sensors.

Adriana Arza Valdés is a Post-Doctoral researcher in the Embedded Systems Laboratory at the the Swiss Federal Institute of Technology Lausanne, Switzerland. She received her PhD degree in Microelectronic and Electronic Systems in 2017 and M.Sc. degree in Micro and Nanoelectronics Engineering in 2012, both from Autonomous University of Barcelona. Her research interests include energy-efficient algorithms and machine learning for edge health and wellness applications. Her expertise also covers system-level design of multiparametric smart wearables and analisys and interperation of mulimodal physiological data.

Miguel Peón-Quirós received a Ph.D. on Computer Architecture from UCM, Spain, in 2015. He collaborated as a Marie Curie scholar with IMEC (Leuven, Belgium) and as postdoctoral researcher with IMDEA Networks (Madrid, Spain). He has participated in several H2020 and industrial projects. He is currently a postdoctoral researcher at EPFL. His research includes optimizations for embedded devices and AI for IoT.

Amir Aminifar is an Assistant Professor in the Department of Electrical and Information Technology at Lund University, Sweden. He received his Ph.D. degree from the Swedish National Computer Science Graduate School (CUGS), Linköping University, Sweden, in 2016. During 2016-2020, he held a Scientist position in the Institute of Electrical Engineering at the Swiss Federal Institute of Technology (EPFL), Switzerland. His research interests include Internet of Things (IoT), mobile-health technologies, and medical informatics.

David Atienza (M'05-SM'13-F'16) is an associate professor of electrical and computer engineering, and head of the Embedded Systems Laboratory (ESL) at the Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland. He received his PhD in computer science and engineering from UCM, Spain, and IMEC, Belgium, in 2005. His research interests include system-level design methodologies for high-performance multi-processor system-on-chip (MPSoC) and low power Internet-of-Things (IoT) systems, including new 2-D/3-D thermal-aware design for MPSoCs and many-core servers, and edge AI architectures for wireless body sensor nodes and smart consumer devices. He is a co-author of more than 300 papers in peer-reviewed international journals and conferences, one book, and 12 patents. Dr. Atienza received the DAC Under-40 Innovators Award in 2018, IEEE TCCPS Mid-Career Award in 2018, an ERC Consolidator Grant in 2016, the IEEE CEDA Early Career Award in 2013, the ACM SIGDA Outstanding New Faculty Award in 2012, and a Faculty Award from Sun Labs at Oracle in 2011. He served as DATE 2015 Program Chair and DATE 2017 General Chair. He is an IEEE Fellow, an ACM Distinguished Member, and served as IEEE CEDA President (period 2018-2019).