Modular Design and Optimization of Biomedical Applications for Ultra-Low Power Heterogeneous Platforms

Elisabetta De Giovanni, Fabio Montagna, Benoît W. Denkinger, Simone Machetti, Miguel Peón-Quirós, Simone Benatti, Davide Rossi, *Member, IEEE* Luca Benini, *Fellow, IEEE* David Atienza, *Fellow, IEEE*

***Abstract*—In the last years, remote health monitoring is becoming an essential branch of health care with the rapid development of wearable sensors technology. To meet the demand of new more complex applications and ensuring adequate battery lifetime, wearable sensors have evolved into multi-core systems with advanced power-saving capabilities and additional heterogeneous components. In this paper, we present an approach that applies optimization and parallelization techniques uncovered by modern ultra-low power platforms in the SW layers with the goal of improving the mapping and reducing the energy consumption of biomedical applications. Additionally, we investigate the benefit of integrating domain-specific accelerators to further reduce the energy consumption of the most computationally expensive kernels. Using 30-second excerpts of signals from two public databases, we apply the proposed optimization techniques on well-known modules of biomedical benchmarks from the state-of-the-art and two complete applications. We observe speed-ups of $5.17\times$ and energy savings of $41.6\,\%$ for the multi-core implementation using a cluster of 8 cores with respect to single-core wearable sensor designs when processing a standard 12-lead electrocardiogram (ECG) signal analysis. Additionally, we conclude that the minimum workload required to take advantage of parallelization for a hearbeat classifier corresponds to the processing of 3-lead ECG signals, with a speed-up of $2.96\times$ and energy savings of $19.3\,\%$. Moreover, we observe additional energy savings of up to $7.75\,\%$ and $16.8\,\%$ by applying power management and memory scaling to the multi-core implementation of the 3-lead beat classifier and 12-lead ECG analysis, respectively. Finally, by integrating hardware (HW) acceleration we observe overall energy savings of up to $51.3\,\%$ for the 12-lead ECG analysis.**

I. Introduction

INCREASING healthcare costs [1] and hospital overcrowding call for new technological advances that improve remote health monitoring and enable self-diagnose, early intervention or prevention [2]. In addition, population aging and the consequent higher incidence of noncommunicable diseases (NCDs) create the need for long-term health monitoring. Within NCDs, cardiovascular diseases (CVDs) in particular—which are characterized by abnormal events that need to be detected in real-time—are the major cause of death globally [3]. To prevent, predict and detect NCDs, there is an increasing need of automatic applications that continuously and remotely monitor relevant biosignals, such as the electrocardiogram (ECG) [4].

In the context of remote health monitoring, wearable sensor nodes (WSNs) have proven capable of attaining accurate inference with minimal power consumption [5]. In this way, WSNs have evolved from single-core systems [6], [7] into ultra-low power (ULP) [8] and multi-core parallel computing platforms [9]–[13]. Most of the typical WSN-based biomedical applications in the state-of-the-art have been implemented on single-core processors [6], [7], [14], [15]. To exploit the new parallel capabilities of modern WSN platforms in the context of biomedical applications, per-lead (i.e. channel) multi-core computation is a natural option to achieve low-power operation, as in the case of multi-lead ECG analysis [10], [16]. However, more general WSN-based biomedical applications for monitoring of NCDs typically include several building blocks which often are not amenable to per-lead parallelization [7], [11], [14], [15], [17]–[22]. Modern platforms have also evolved into hybrid systems with a main core and an additional cluster of cores [9] that allow flexible design of efficient single-core and parallel modules, in applications where several modules cannot be parallelized easily.

In addition to parallelization, modern platforms offer clock- and power-gating mechanisms to reduce both dynamic and static (leakage) power when the system is not actively computing (e.g., when waiting for new samples to arrive in an input buffer considering the usual low sampling frequency of biomedical applications). Some platforms include specialized direct memory access (DMA) engines that execute data capturing tasks within tight power budgets while the rest of the system is clock-gated or executing other tasks [12], [13]. Additionally, other platforms contain SRAMs structured in independent banks that can be power-gated depending on the application needs [12], [13]. Moreover, application modules typically contain computationally expensive kernels that can be accelerated with domain-specific hardware such

This article was presented in the International Conference on Hardware/Software Codesign and System Synthesis 2020 and appears as part of the ESWEEK-TCAD special issue.

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, in part by the H2020 DeepHealth Project (GA No. 825111).

E. De Giovanni, B. Denkinger, S. Machetti, M. Peón-Quirós and D. Atienza are with the Embedded Systems Laboratory (ESL), EPFL, Lausanne, Switzerland.

F. Montagna is with the Department of Computer Science and Engineering, University of Bologna, Italy.

S. Benatti and D. Rossi are with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, Italy.

L. Benini is with the Integrated Systems Laboratory, ETH Zúrich, Switzerland, and also with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, Italy.

as coarse-grained reconfigurable arrays (CGRAs) [10]. Thus, hardware (HW) acceleration is an orthogonal benefit to the parallelization and it can benefit both single-core and multi-core application design.

In this work, we tackle the challenge of exposing the parallelization and power-saving capabilities of modern ultra-low power platforms to the designer of WSN-based biomedical applications. Our main contributions are:

- We show how to parallelize the modules of typical biomedical applications at different levels of abstraction (i.e., lead, sample analysis-window, heart beat or data-level) to maximize speed-up and consequently reduce energy consumption up to $41.6\,\%$.
- We explore how to reduce static power by exploiting power management and SRAM-bank memory scaling with additional energy savings of up to $16.8\,\%$ for a state-of-the-art application.
- We investigate the use of programmable domain-specific accelerators to perform intensive computations at lower power than with general-purpose processors obtaining energy savings up to $46.7\,\%$ in the multi-core implementation of the state-of-the-art application.
- Finally, we show the orthogonality of the previous optimizations achieving accumulated energy savings of up to $51.3\,\%$.

The rest of this paper is organized as follows. Section II explores the parallelization and power-saving features of modern ultra-low power platforms (Section II-A), an analysis on the optimal exploitation of these features and the typical modular organization of WSN-based biomedical applications (Section II-C). Then, Section III explains how to exploit those features during a typical WSN-based application implementation. Section IV presents the software / hardware experimental setup used in Section V to analyze the impact of our proposed methods. Finally, in Section VI we summarize the main conclusions of our work.

II. Background and Motivation

A. Modern ultra-low power WSN platforms

The main goal of multi-core ultra-low power WSN platforms is reducing energy consumption to maximize battery lifetime, while still running complex algorithms on the nodes. Multiprocessing has been proved effective in reducing energy consumption—through lower operating frequencies and supply voltages—while preserving performance in the biomedical [16] and multimedia [9] domains. However, SW tasks must be divided into parallel subtasks or organized as independent parallel ones, i.e. application modules. Often, a major obstacle to achieve adequate speed-ups is the overhead of synchronization. Fast HW event managers offer single-cycle synchronization and enable clock-gating the processors while waiting for events, hence saving significant amounts of energy even with fine-grained parallelization [13], [23]. A novel architecture that can overcome these obstacles and ensure flexible design of modular WSN-based biomedical applications, is the open-source RISC-V based PULP platform [9]. In this section, we describe the energy saving capabilities of parallel implementation on multi-core platforms based on PULP. Moreover, we describe the power and memory management possibilities in modern ULP platforms. Finally, we explore the architectural heterogeneity of adding CGRAs to accelerate computationally intensive kernels.

1) Parallelization in the PULP platform: In this work, we target the PULP platform [9], which is divided into a main streamlined processor, the fabric controller (FC), and an 8-core parallel compute cluster (CL). PULP includes a multi-banked $512\,\mathrm{KiB}$ L2 memory, a HW event synchronizer and a shared multi-banked $64\,\mathrm{KiB}$ L1 memory with single-cycle latency in the cluster side. Both FC and cluster are power-gated while the DMA fills the required L2 memory bank during sample acquisition. Each of the cores in the cluster can be independently clock-gated to reduce dynamic power. For example, the cluster cores become clock-gated after reaching a synchronization point. This flexibility allows to easily implement parallel and single-core modular applications.

2) Power and memory management: In addition to parallelization, WSN-based biomedical applications need power management to ensure continuous remote monitoring. A common technique to save energy is clock-gating, which reduces dynamic power. In the context of the PULP platform, architecture-level clock-gating is applied at different levels. The SoC is clock-gated when waiting for an event, such as a DMA transfer or the end of a computation on the cluster. Additionally, if no workload is assigned to some cores of the cluster, they are automatically clock-gated. This is relevant in the context of modular WSN-based biomedical applications, because an optimal assignment of resources to the modules reduces energy consumption. Conversely, power-gating interrupts the power supply to parts of the circuit that are unused for longer periods, hence suppressing leakage current. Power-gating has a larger physical overhead than clock gating—due to the power switches and controllers around the power gated area. Hence, it is applicable only for large blocks (e.g., a cluster of processors). Moreover, the recovery period for power-gating can be in the order of tens of thousands of cycles, particularly if clock generators are affected, making it suitable only for applications that undergo long idle periods. Typical WSN-based applications are characterized by low sampling frequency (e.g., ECG acquisition is in the standard range of $250\,\mathrm{Hz}$–$500\,\mathrm{Hz}$), hence, the main SoC can be power-gated while waiting for the next sample. Additionally, modern platforms divide SRAM memories in several banks that can be independently power-gated or set to retention mode according to the amount of memory required at a given moment.

3) HW acceleration: Finally, domain-specific accelerators, either programmable (e.g., CGRAs [10]) or task-specific (e.g., for FFT or sample-rate conversion [8]) are added to accelerate intensive application kernels. In this case, energy savings stem from the shorter execution times and the specialized implementations of the accelerators. Hardware accelerators can be introduced at the end of the optimization process to offload kernels assigned to particular cores. In this work, we implement a CGRA, designed to execute small loop-based kernels with high numbers of iterations. We describe in detail

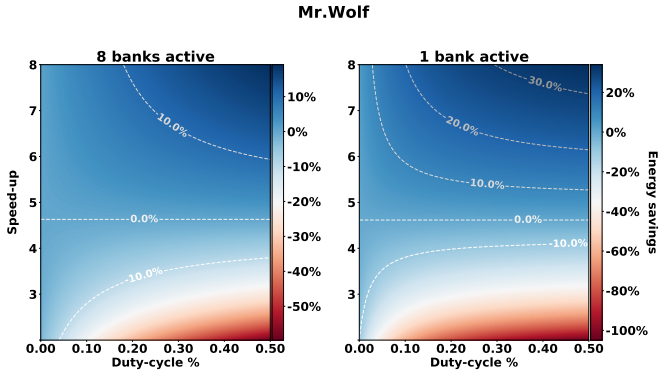

Fig. 1. Potential energy savings in Mr.Wolf, an implementation of the PULP platform, according to the application duty cycle and the attainable speed-up through an 8-core parallelization in the cluster. On the left, we show the analysis on Mr.Wolf with its 8 memory banks active. On the right, we show the analysis on Mr.Wolf with only 1 bank active. The dotted lines mark different levels of energy savings.

the architecture of the CGRA and the computational kernels accelerated in Section III-D.

B. Motivational analysis for optimizations in PULP

Considering the low duty cycle of WSN-based biomedical applications, we conduct an analysis of the impact of the application duty cycle and the attainable speed-up in an 8-core parallelization on the energy savings in the PULP platform. In this analysis, we assume that the 8 cores are all used during the active part of the duty cycle, while during idle periods they are power-gated. In contrast, in many biomedical applications or its modules, as the ones we present in Section V, it may happen that only some of the cores are active, while the remaining are clock-gated (i.e., unused). Moreover, we show how activating one bank (of 64 KiB) or the full memory (i.e., 8 banks for a total of 512 KiB) affects the energy savings. Finally, this analysis shows that the percentage of energy consumed during idle time is proportionally inverse to the duty cycle. Consequently, platforms that execute very low duty cycle applications need to optimize energy consumption during idle periods (e.g., turning off unused memory banks). In contrast, with higher duty cycles, the energy consumed during active time prevails, hence, it becomes more relevant to optimize computation (e.g., increasing the speed-up to reduce active time) in order to lower the total energy consumption.

Figure 1 shows the previous analysis on one evolution of the PULP platform, Mr.Wolf [13]. For each platform, the graph reports the energy savings compared to a single-core implementation of a generic application in the FC. Mr.Wolf includes a core for the FC (Zero-riscy [24]) that is simpler than the RI5CY cores of the cluster [25] and runs at a higher frequency (170 MHz for FC and 110 MHz for the cluster) but has a lower IPC. Moreover, Mr.Wolf is more efficient for higher duty cycles because it was designed to handle high computational load and the deep sleep mode is not optimized for long idle periods—different PULP implementations with a core optimized for deep sleep exist, tough. Therefore, our analysis is applied on the Mr.Wolf architecture with a more optimized deep sleep mode based on other PULP implementations. The graphs in Fig. 1 are generated using the energy models in (1) and (2) for the single-core (E_{SC}) and the multi-core (E_{MC}) configurations, respectively, where dc is the duty cycle of the application, FC_P_{dyn} and FC_P_{leak} are the dynamic and leakage power of the FC, respectively, DS_P is the power in deep sleep, CL_P_{dyn} and CL_P_{leak} are the dynamic and leakage power of the CL, and f_{cr} is the frequency correction ratio ($\frac{170\,\text{MHz}}{110\,\text{MHz}}$) for the FC and CL.

$$E_{SC} = dc \times (FC\_P_{dyn} + FC\_P_{leak}) + (1 - dc) \times DS\_P \quad (1)$$

$$E_{MC} = \frac{dc \times f_{cr}}{speedup} \times (FC\_P_{leak} + CL\_P_{leak} + CL\_P_{dyn}) + (1 - \frac{dc \times f_{cr}}{su}) \times DS\_P \quad (2)$$

Finally, the ratio (in percentage) of potential energy savings attainable by a multi-core configuration against the single-core one is computed using (3).

$$E_{\%} = (1 - \frac{E_{MC}}{E_{SC}}) \times 100 \quad (3)$$

On the left side of Fig. 1, we show the analysis for Mr.Wolf with the full memory active (i.e., 8 banks). It shows that the energy overhead of the multi-core cluster is recovered when a speed-up of $4.6\times$ is reached and becomes more energy efficient compared to the single-core implementation for higher speed-ups. Additionally, each of the Mr.Wolf 8 memory banks of 64 KiB can be powered-off depending on the application. Consequently, on the right side of Fig. 1, we show how the analysis changes if there is only one bank active. Whereas the threshold of speed-up does not change, for lower duty cycles it is possible to achieve higher energy savings.

We have also run the analysis on the full scale of duty cycle values to explore the benefits attainable under higher duty cycles. The architecture is able to achieve energy savings up to 42 % for 100 % duty cycle and maximum speed-up with the 8 cores and 8 banks always active. An interesting result is that, for high duty cycle applications, memory management has less impact than for low duty cycle ones. Nonetheless, in this work we focus on the energy savings attainable on low duty cycle which is a characteristic of typical biomedical applications.

From this previous analysis, we can conclude that, for this implementation of PULP, the speed-up required by the parallel application has to be at least $4.6\times$. This shows the importance of suitable optimizations (e.g., parallelization techniques) in order to achieve energy efficiency on modern low power heterogeneous platforms, which is the main motivation for this paper. To achieve optimal speed-up, a modular approach to SW parallelization is necessary. For this reason, we present the typical modules of WSN-based biomedical applications in Section II-C. Then, to maximize the speed-up of the overall application, we consider different parallelization techniques and HW acceleration. Power management is also a significant factor in low duty cycle applications. Finally, memory bank management plays an important role in energy saving and,

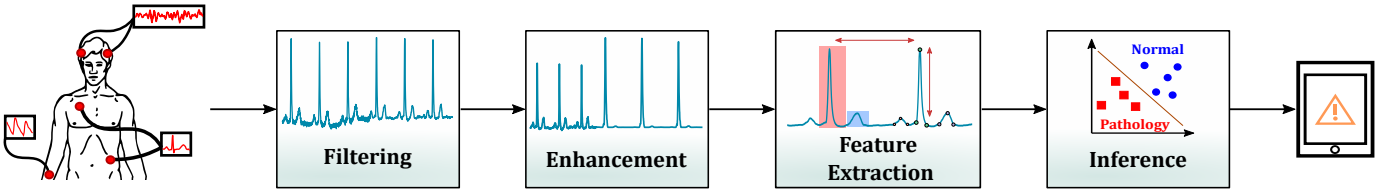

Fig. 2. Typical modules of a general WSN-based biomedical application.

specifically, for applications with low memory footprint. In Section III, we refer to a general conceptual architecture that takes advantage of all the benefits of the PULP platform discussed in this analysis.

C. Typical biomedical modules

Considering the characteristics of modern ULP platforms, we propose a modular design approach for biomedical applications that combines different types of SW parallelization to achieve optimal speed-up. Let us consider a typical WSN-based biomedical application for long-term health monitoring, described in Fig. 2. First, the single or multi-channel signal is filtered to remove high or low frequency noise, baseline wandering or muscle noise. The second module includes typically some additional preprocessing of the signal to enhance specific characteristics or combine different channels. The third module is the extraction of patterns or features, such as the signal main waveforms and time or frequency-domain parameters. The final step, inference, includes any kind of classification or regression technique that uses the information of the extracted features to predict an outcome, such as the occurrence of a pathology. In this work, we apply the energy-saving capabilities of modern platforms to an optimized single-core version of well-known instances of each of those modules. Then, we evaluate them as part of complete state-of-the-art applications.

1) Filtering: Digital filtering in biomedical applications is used to remove undesired noise at specific frequencies or isolate the frequencies of interest. In biosignal processing there exist different types of filtering [26]. In this work, we analyze the morphological filtering (MF), which extracts the signal baseline based on the shape of the original signal and then subtracts it. This method was originally used in image processing and then modified to be used on a single or a multi-lead ECG in embedded systems [27]. Additional techniques to filter the raw ECG input data that are suitable for embedded systems are described in [27].

2) Enhancement: Several techniques, such as the signal derivative or the root-mean-square (RMS) combination, are available to enhance a biosignal or combine different leads. We study a light-weight example of short-term event amplification: Relative Energy (Rel-En) [28]. In the context of an ECG signal, this technique extracts the energy of specific windows of analysis to amplify the R peaks, since the signal energy is larger when an R peak occurs. The Rel-En method is also used for K-complex detection in electroencephalography (EEG) and pulse extraction in imaging photoplethysmography (iPPG) [29]. Additionally, we consider the RMS lead combination as part of a full application in Section V.

3) Feature extraction: This module enables the biosignal abstraction through the extraction of the most relevant features, from its waveforms or points to time and frequency domain parameters [7], [11], [14], [17]–[19], [21], [22]. In ECG analysis, for example, a common technique, called "delineation," abstracts the signal main waves (i.e. QRS complex, P and T waves [30]) with three "fiducial points" representing the onset, offset and peak. These points are the input to the inference module or can be further processed extracting additional features (e.g., QRS complex duration, QT interval, etc.). In this work, we analyze the ECG delineation since it is a relevant and well-known method for long-term monitoring of NCDs. The process of delineation can be divided in two parts. First, the R peak or QRS complex are detected, often independently from the other ECG waves, since they describe the heart rhythm and are relevant for the detection of many arrhythmias [18]. As R peak detection technique, we choose to implement REWARD [28] for its claimed low computational load. REWARD uses amplitude thresholds to isolate the R peak. Moreover, it analyzes physiological peak-to-peak distance and peak width to filter false positives, such as dominant T-waves. The remaining fiducial points can be delineated in different ways. We choose a low-complexity method [17] that assumes that the signal's main waves are positive, which can be ensured by an RMS combination of leads or choosing lead II of the 12-lead ECG technique [31]. Under this assumption, the Q and S points are identified as minimum within a physiological interval near the R peak. The P and T peaks of the two other main waves are computed as maximum within physiological windows between two R peaks. Finally, the onset/offset of the P and T waves are computed considering the minimum euclidean distance between the original waves and their piece-wise linear approximation. The point with the minimum euclidean distance that intersects the isoelectric line is the onset/offset.

4) Inference: The last module is commonly a classification or regression problem applied to a set of features that performs an automatic diagnosis of a medical condition, such as the occurrence of abnormal beats. Several types of arrhythmia can change the heart electrical signal, thus causing abnormalities in the ECG main waves. Therefore, automatically detecting abnormal beats and their nature helps treating them and prevents further complications [6], [20]. Other biosignals (e.g., photoplethysmogram (PPG), respiration, impedance cardiogram (ICG), etc.) also contain relevant features to classify NCDs, such as sleep apnea [19], to monitor a subject state in

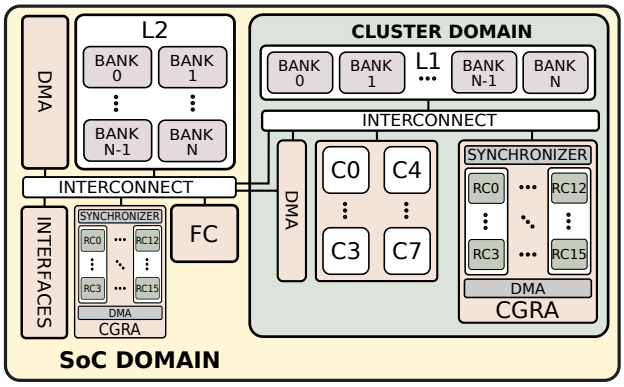

Fig. 3. Conceptual architecture following the PULP platform. From left to right: SoC domain containing the main processor (FC), the L2 memory, the DMA, and potentially a CGRA. The cluster domain contains the multi-core cluster (CL); the L1 memory; a CGRA.

stressful environments [22] or for gesture recognition [11]. In this work, we analyze a classification module for detection of abnormal beats from an ECG signal using random projections and a neuro-fuzzy classifier [32].

III. SW and HW Optimizations in Modular Biomedical Applications

In this section, we do a top-down exploration of parallelization techniques at different abstraction levels. This strategy helps to compose a modular biomedical application in such a way that it can exploit the energy-saving platform characteristics and maximize it taking into account the analysis done in Section II-B. Additionally, we apply memory and power management according to general characteristics of the application (e.g., duty cycle, memory needed for acquisition, etc.). Finally, we integrate a domain-specific accelerator that can execute intensive kernels faster (and consuming less energy) than the general purpose cores available. In Fig. 3 we draw a conceptual architecture, based on the analysis reported in Section II-B, which can be used to apply the SW and HW optimizations described in this section.

A. Modular SW optimizations

Considering the characteristics of the algorithms described in Section II-C, we present several techniques to extract parallelism. We also propose a top-down order for exploring them, as follows. The first choice of parallelization is by lead (or channel). In fact, if leads are processed independently throughout the application, it is the simplest and most efficient implementation. However, many biomedical applications and their modules only work on single-lead or multi-lead combination. Then, a window parallelization should be considered where the cores work on subsegments of the signal. In some cases, the characteristics of the signal and the application make it necessary to consider a more specific type of parallelization, such as a beat parallelization for cardiovascular-based signals. This method can be extended to any kind of periodic or pattern signals where the features within a period or pattern need to be captured. When the previous methods cannot be applied, a general data-level parallelization should be considered. Finally, if none of the previous methods can be applied, or if the obtained speed-up is not satisfactory, a pipelining strategy can be considered, where a subset of the cores is assigned to each of the pipeline stages. The cores at one stage process segments of input data and produce segments of output, which are processed by the cores in the next stage in a parallel consumer/producer pattern. However, for accuracy and standardization purposes, biomedical applications often include checks or feature combinations that need to be executed once the complete output of a module has been generated [14], [28], [33]. Given this limitation, and the fact that the effort to implement pipelining is larger, we consider only the first four types of per-module parallelization in the proposed top-down order. Table I summarizes the different types of parallelization techniques applied to each module described in Section II-C.

TABLE I
Summary of parallelizations applied to each module

Module	Algorithm	Opt.	Notes
Filtering	Morph. Filt. (8L-MF)	Lead	Data-dependent
Enhance.	Relative energy (Rel-En)	Window	Homogen., overlap
Enhance.	Lead combination (RMS)	Data	Homogen., $1/8$ samples
Feat. Extr.	R-peak (REWARD)	Window	$8 \times 1.75\,\mathrm{s}$ windows
Feat. Extr.	Fiducial points	Beat	Data-dependent
Inference	Beat classification	Beat	Data-dependent

1) Lead parallelization: WSN-based biomedical applications commonly acquire multi-lead signals (e.g., 3–12 ECG leads) to extract more information for highly accurate monitoring. Multi-lead parallelization, where each core processes the data corresponding to one lead in parallel, should be applied first as it typically offers almost linear speed-ups. The most common application is the filtering module, which often works on multiple leads or channels [10], [11] or even on multiple signals [14]. Another example from the literature where this parallelization is applied is a multi-lead delineation using multi-scale morphological derivatives (MMD) [10].

As shown in Fig. 3, in the PULP architecture the DMA can access both the L2 and L1 memories. It can be used to transfer the samples of each lead from L2 into separate areas of L1, thus allowing the cluster of cores to implement the per-lead filtering without interference. The MF algorithm of our example is data-dependent; hence, the workload of each core depends on the amount of noise of each lead (e.g., due to problems in the electrode positioning). We consider an 8-lead ECG (8L-MF).

2) Window parallelization: For subsequent modules in the processing chain, or in the case of applications that obtain data from a single lead, the data to be processed can be divided in multiple windows [11], [17]. In this way, each window is processed in parallel by a different core. Furthermore, if the samples are directly collected by the DMA module, this method enables power-gating of the platform cores over larger periods. Energy savings stem from operating at lower frequency and voltage than a single core and by a more aggressive application of power-gating than is possible when operating on a sample-by-sample basis.

In our example, we apply this technique to the signal enhancement (Rel-En) and the feature extraction (R peak detection) modules. In the case of Rel-En, we divide the window in smaller windows, with each core starting from the first sample of each sub-window as explained in [34]. Since the Rel-En algorithm computes the signal energy at the sample n using information starting from $(t(n) - \frac{0.95}{2})\,\mathrm{s}$, a small window overlapping is necessary. Therefore, the computational workload is in this case homogeneous among the cores, but the speed-up is reduced by the introduced overhead.

On their side, R peak detection techniques usually consider fixed windows of analysis to extract the peaks based on physiological characteristics. In our case, the REWARD algorithm [28] uses a fixed window of $1.75\,\mathrm{s}$. Therefore, considering 8 cores, our method collects a buffer of $(8 \times 1.75)\,\mathrm{s}$ so that each core will compute one fixed window.

3) Beat parallelization: The ECG and other cardiovascular-based signals (e.g., PPG) are characterized by beats. Applications often perform the same operation for each beat in which there is essential information. Therefore, beat parallelization is the next step to explore in the top-down proposed order. This technique can be applied to any upper-level feature, time series or excerpt of relevant information from the signal. There are several examples of classifiers and feature extraction techniques in the literature where this type of parallelization can be applied [17], [20], [32], [35]. However, for simplicity we only analyze two of them, corresponding to two of the modules described in Section II-C, namely the beat classification [32] and the delineation of fiducial points [17]. In the former case, the beat is centered to the R peak; in the latter, it comprises the signal between two R peaks. Again, to match the characteristics of our platform, we collect 8 beats, one per core. During beat classification, the workload is data-dependent and varies also with the window length (which may be fixed). In the case of the fiducial points delineation, each core's workload is linked to the natural variability of the RR intervals (i.e., heart rate). In Section V, we show the effect of the different workloads on speed-up and energy consumption.

4) General data-level parallelization: General-purpose parallelization techniques can be applied on the inner kernels of each module. Good candidates at this stage common to multiple applications are sorting algorithms, RMS combination [11], [32], training algorithms running on node [11] or several filtering techniques, such as those presented in [7]. In this work, we study the RMS combination algorithm, which is also used in the complete application that we analyze in Section V. RMS is a signal enhancement technique that computes the root-mean-square of a buffer of data. In WSN-based biomedical applications, this is used to combine a multi-lead signal into a single-lead one. Following the work presented in [32], our implementation first computes the sum of squares of the samples of the different leads and then applies a square-root to the result. Since RMS works on sample i_{th} from each lead independently of the other samples, each core receives a similarly-sized subset of the samples from all the leads.

B. Power management and memory bank scaling

When combining the modules into a full application, we apply SoC and SRAM power-management. The FC in the PULP platform is power-gated whenever the data is acquired, while it needs to be clock-gated when the DMA stores the data in L2. Considering the low duty cycles of typical WSN-based biomedical applications, such as the one reported in Section V, the time spent during acquisition and storing is significantly high compared to the processing. The power management strategy of power- and clock-gating during idle time allows to significantly reduce the energy consumption. Moreover, during the acquisition phase, banks not containing new data (nor application code) can be powered off. Banks that contain captured samples waiting to be processed can be placed in retention mode. Finally, only the bank currently receiving samples needs to be active. However, since the memory needed for the analyzed biomedical applications is significantly lower than $512\,\mathrm{KiB}$, we explore the possibility of reducing the overall memory to $128\,\mathrm{KiB}$ and assuming 8 banks scaling each bank size to $16\,\mathrm{KiB}$. This strategy allows a smaller resolution in bank size and a better management of the activated banks depending on the specific application, hence, reduced energy consumption. For example, let us consider an application that needs to process a signal window of $30\,\mathrm{s}$ integer $16\,\mathrm{bit}$ acquired at a sampling frequency of $250\,\mathrm{Hz}$. Since the buffer to store is $30\,\mathrm{s} * 250\,\mathrm{Hz} * 2 = 15\,\mathrm{KiB}$, only one bank needs to be active, on top of the banks needed for the code. As shown in Fig. 3, the scaling strategy can be pushed to the limits of feasibility and significantly lower energy consumption, especially for applications with low memory footprint. Memory scaling and management is a relevant design factor orthogonal to parallelization for typical low duty cycle biomedical applications.

C. Application-level optimizations

In addition to general purpose power and memory management, specific algorithmic-level optimizations for WSN-based biomedical applications need to be applied. For example, one of the applications we evaluate is the beat classifier discussed in Section II-C, which requires several of the modules described previously. The single-core implementation of this algorithm adapts its computational complexity based on the outcome of the classification. First, it analyzes a single-lead ECG and performs only R peak detection to save energy. Then, if the algorithm detects an abnormal beat, it performs an RMS combination of a 3-lead ECG and a full delineation. However, this approach can be counter-productive in multi-core platforms because the direct execution of the 3-lead ECG analysis on three cores consumes roughly half time than the "1+2" analysis approach. In particular, with the database used in our experiments (MITDB, see Section IV), approximately $27\,\%$ of the patients experience abnormal beats more than $50\,\%$ of the time, thus requiring the full 3-lead processing. This can be exploited at run-time by determining the frequency of execution of the full analysis: if a certain threshold is exceeded, the system switches to the parallel version. Another application that we evaluate is the delineation of a complete set

of 12-lead ECG. The resources assigned in this case include the full 8-core cluster. However, after processing 8 leads with an approximately equal distribution of computation, 4 cores are automatically clock-gated while the other 4 process the remaining leads.

D. HW acceleration for intensive computational kernels

MorphoSys [36] is one of the earliest examples of CGRAs originally proposed to accelerate multimedia applications with strong computational demands. Later works showed how a CGRA can be used in the domain of biomedical applications to reduce power by both accelerating common operations and reducing the energy cost of executing those operations [10]. We extend the open-source PULP platform [37] with a CGRA following the design presented in [10] for biomedical applications, which is composed of 16 reconfigurable cells (RCs) forming a 4×4 torus interconnect. The CGRA can be integrated in the SoC-domain (i.e., connected to the FC), or in the cluster-domain (i.e., connected to the cores of the cluster), accessing the L2 or L1 memories directly, respectively, as shown in Fig. 3. In this work, we use a CGRA divided into four independent columns of RCs; each kernel may use 1, 2 or 4 columns. Unused columns remain clock-gated. The configuration memory is implemented as a $2\,\mathrm{KiB}$ standard cell memory (SCM). The cores make acceleration requests by writing a kernel ID to the CGRA peripheral registers (one per core). The CGRA synchronizer takes care of mapping the request to the number of columns necessary to execute the specified kernel. When a core requests an acceleration, it becomes clock-gated until the request is completed. The CGRA RCs have a 16-bit datapath, which is adequate for most WSN-based biomedical applications whose input data is normally limited by ADC resolution. However, several modules, such as the signal enhancement, require 32-bit accumulation; thus, it cannot be accelerated with the current platform design.

1) Kernel selection: The kernel selection procedure for the CGRA follows the steps described in [38, Chap. 3]. LLVM is used to analyze the application from the C code and generate an execution profile report. This enables the identification of computationally intensive loops that are good candidates for CGRA acceleration.[1] Finally, kernels that do not meet the design constraints of our CGRA are discarded. In that sense, the main limiting factor is the small instruction memory of the CGRA (16 32-bit instructions per RC), which restricts the selection to short kernels. Table II lists the kernels executed on the CGRA.

2) Kernel mapping: To map kernels on the CGRA, we inspect the C code disassembly to identify operations that can be parallelized. Then, these operations are translated to the CGRA instruction set and distributed over the CGRA's RCs and columns. This last step is done manually to fully exploit the torus interconnect of the CGRA—each RC is connected to its neighbours—generating the data flow execution that is one of the advantages of this CGRA design.

[1]If LLVM is not available for the target platform, cycle accurate simulators, such as those available in the PULP SDK, can be used in combination with processor hardware counters to profile the application main blocks.

TABLE II
COMPUTATIONAL KERNELS EXECUTED ON THE CGRA

Algorithm	Kernel	Notes
Morph. Filt.	dblmin / dblmax	Linear 1st and 2nd min./max. search in a vector
Fiducial points	maxpeak	Linear peak (absolute max.) search in a vector
Beat classification	min_max	Circular min. and max. search in a vector

IV. Experimental Setup

A. Test benches for biomedical modules

We design a test bench for each module that includes appropriate input signals. For the filtering, signal enhancement and signal delineation modules, we consider excerpts of signals from the Physionet QT database (QTDB) [39]. This database was used to analyze the three single-core benchmarks presented in Section II by [28]. We choose four signals from the QTDB, as four examples that represent worst, best and two average cases in terms of a combination of noise and shape of the three ECG waves. For the inference module, we consider the MIT-BIH Arrhythmia Database (MITDB) [40], as reported in [32]. We choose four signals as worst, best and two average cases in terms of percentage of abnormal beats over the total number of annotated beats. Its output is a label classifying the beat depending on the pathology: "N" for normal beats, "V" for premature ventricular contraction, etc. For all the modules, the choice of four cases should describe most of the design space in terms of complexity and energy consumption due to data-dependent variability.

B. Test benches for biomedical application

To better evaluate the impact of the proposed optimizations, we evaluate two applications, with data capturing periods, using our biomedical modules. First, we consider a 3-lead heartbeat classifying application [32]. This application applies filtering, relative energy, and R peak detection on one lead (lead I). If the heartbeats are classified as normal, it stops there. However, if any abnormality is detected, then it applies the same methods to the other two leads (leads II & III) to supply additional information. Second, we implement an application processing the complete set of 12-lead ECG signals. Such application is required for medical compliance and used in intensive care units of hospitals, or in athletic or military training supervision. It combines the modules MF, RMS (to combine all the signals into a single one), R peak and fiducial points detection. Both applications capture ECG samples during $15\,\mathrm{s}$; then, the system becomes active to process.

C. Multi-core WSN platform: PULP+CGRA

To measure the execution time of both independent modules and full applications we used the open PULP platform [37]. PULP provides the RTL description of the multi-core platform and an SDK to run RTL simulations, using Modelsim, in order to obtain cycle accurate timings. Additionally, to further explore the advantages of heterogeneous platforms, we added

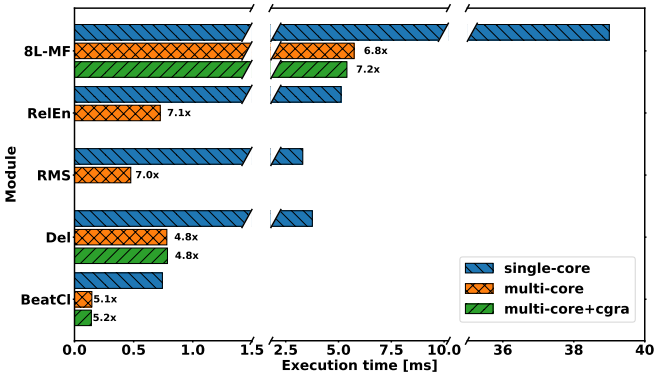

Fig. 4. Computation time of each module.

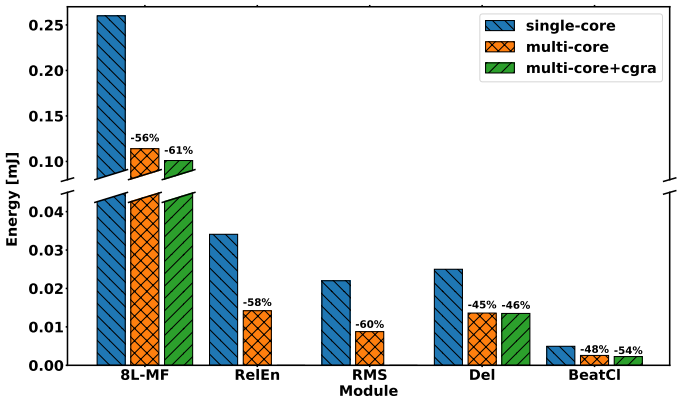

Fig. 5. Per-module energy consumption and savings (geometric mean) compared to the single-core design.

TABLE III
EXECUTION TIME OF THE DELINEATION MODULE FOR DIFFERENT SUBJECTS FROM THE PHYSIONET QTDB [39] AND THE SUBSEQUENT VARYING SPEED-UPS

SUBJECT	SINGLE-CORE (ms)	MULTI-CORE (ms)	SPEED-UP
1	2.77	0.67	4.13 ×
2	3.66	0.78	4.69 ×
3	4.72	0.93	5.08 ×
4	3.88	0.75	5.17 ×

TABLE IV
ENERGY SAVINGS IN THE DELINEATION MODULE ON FOUR SUBJECTS FROM THE PHYSIONET QTDB [39] FOR THE SINGLE-CORE (S) AND MULTI-CORE (M) PLATFORMS

SUBJECT	SINGLE-CORE (µJ)	MULTI-CORE (µJ)	SAVINGS (%)
1	18.4	11.3	38.6
2	24.3	13.6	44.0
3	31.4	16.4	47.8
4	25.8	13.2	48.8

the CGRA to the cluster domain integrating it in the existing cycle-accurate simulation flow. We use the power numbers reported for a chip based on the PULP architecture implemented in TSMC 40 nm LP CMOS technology, Mr.Wolf [13]. This SoC features a streamlined 12 k-gates RISC-V main processor (Zero-riscy [24]) and an 8-core compute cluster with DSP extensions (RI5CY). This platform includes 8 physical memory banks for the 512 KiB L2 memory. We pick the lowest energy point of the platform, at 0.8 V. The platform requires 3.6 µW when power-gated[2] and 12.6 µW with full L2 retention—since typical biomedical applications require small amounts of memory, we reduce the size of the L2 to one fourth (i.e., 128 KiB), maintaining the same bank number, and correspondingly reduce its power requirements. When the SoC is active, it requires 0.98 mW with its main processor clock-gated, and 6.66 mW with it operating at 170 MHz. Once the cluster is activated, it requires 0.61 mW with all cores clock-gated and 18.87 mW with the 8 cores running at 110 MHz. We obtained the power estimations for the CGRA through pre-layout netlist simulation with the TSMC 40 nm LP CMOS technology. The CGRA requires 104 µW when idle, with an average power of 669 µW when active. The CGRA and the cluster are power-gated together.

First, we performed the RTL simulation and estimated the energy consumption on the test benches for biomedical modules to show the impact of the modular SW optimizations, as shown in Section V-A. Then, we ran the RTL simulation and estimated the energy consumption on the full applications to report in Section V-B the impact of parallelization, memory scaling and HW acceleration.

[2]As reported for GAP-8 [12], which is an industrial version of PULP with SoA deep sleep optimizations not yet included in its academic counterpart.

V. EXPERIMENTAL RESULTS

A. Per-module speed-ups and energy savings on PULP

Figure 4 shows the execution time of each module with the single- and multi-core implementations and the geometric mean of the obtained speed-ups. The maximum speed-up (7.1 ×) is reached in the Rel-En module, despite its small overhead due to the window overlapping scheme. For the remaining modules, the speed-up varies between 4.8 × and 7.0 ×, which is above the threshold of speed-up for the PULP platforms discussed in Section II-B. The RMS module, which applies a data-level parallelization, reaches a speed-up of 7.0 ×, since the eight cores work independently on similar workloads. The MF module is executed on the same trace repeated for the 8 leads in order to have the same workload and show a data-independent multi-core processing. This module achieves a similar speed-up of 6.8 ×, which is justified by two factors: the 8 cores in the CL run at a lower frequency than the FC (i.e., ≈ 0.65 ×), but they have a higher IPC. The minimum speed-up (4.8 ×) is obtained for the delineation module (Del) because the workload cannot be divided evenly among the cores: first, the R peak detection algorithm has several data-dependent conditional branches that change the execution path for different cores; second, the beat parallelization used during the delineation depends on how many peaks are detected; finally, the beat length (i.e., the RR interval) is variable and hence the size of the input varies for each core. This effect can be observed in the time spent in the delineation module (Table III) for four different subjects from QTDB.

TABLE V
AVERAGE RESULTS OF ENERGY CONSUMPTION (INCLUDING DATA CAPTURE) AND EXECUTION TIME ON PULP (WITH MEMORY SCALING) FOR THE COMPLETE APPLICATIONS ON FOUR SUBJECTS

# of leads	Single core		Multi-core			
	Energy (mJ)	Time (s)	Energy (mJ)	Savings (%)	Time (s)	Speed-up (×)
1 lead	**0.326**	0.025	0.302	7.3	0.019	1.28
1+2 leads	**0.611**	0.068	0.588	3.7	0.041	1.66
3 leads	0.611	0.068	**0.493**	19.3	0.023	2.95
12 leads	1.78	0.238	**1.00**	**43.5**	0.046	5.14

The previous speed-ups translate neatly into energy savings. Figure 5 reports the geometric mean of the energy consumption for each module over the four chosen subjects of [39] and [40]. The maximum energy savings of the multi-core design correspond to the RMS (60 %) and Rel-En (58 %) modules, which are also the modules with the highest speed-up, whereas it can save at least a 45 % in the remaining modules.

B. Application-level energy savings on PULP

We evaluate the impact of the previous optimizations on two different modular applications, including the energy spent during data capturing periods. First, we consider a 3-lead heartbeat classifying application [32] in three different configurations depending on the optimizations discussed in Section III-C. Then, we consider the 12-lead ECG delineation application. Table V shows the energy and time results for these applications. The values reported include memory scaling to banks of 16 KiB on both single- and multi-core implementations.

The multi-core configuration of the platform is the most efficient option in the four cases analyzed. Even for the 1-lead application, where MF is the most expensive module (i.e., 81.6 % of the active time) and it is executed on the FC, by parallelizing the other modules on the cluster we obtain modest energy savings (7.3 %). The total speed-up is low (1.28 ×) due to the small percentage of parallel code. However, the average speed-up of all the other parallel modules (approximately 5.6 ×) and the memory scaling are enough to achieve fair savings. However, when the application detects abnormal beats the following strategies (1+2 leads and 3 leads) can be applied. In the first case, which follows the optimizations of [32], processing the additional two leads *after* the first one limits the energy savings since the obtained speed-up is not enough to offset the energy of the cluster cores during the extended period. However, if the beat classifier detects abnormal events often enough, the application can use the second strategy and process the three leads in parallel. In that case, the parallel version would achieve a reduction in computation time of 66 % and 19.3 % in energy. In this way, the 3 leads are analyzed simultaneously on 3 active cores of the cluster while the others are clock-gated, enabling better energy savings.

Considering the low computational load of this application, the energy savings of the multi-core optimization are modest but still significant. However, applications requiring medical compliance such as in intensive care units of hospitals, or in athletic or military training supervision, must process the complete set of 12-lead ECG signals, which generates higher computational load. The last row of Table V shows that the parallel version achieves in this case a speed-up of 5.14 × and energy savings of 43.5 %.

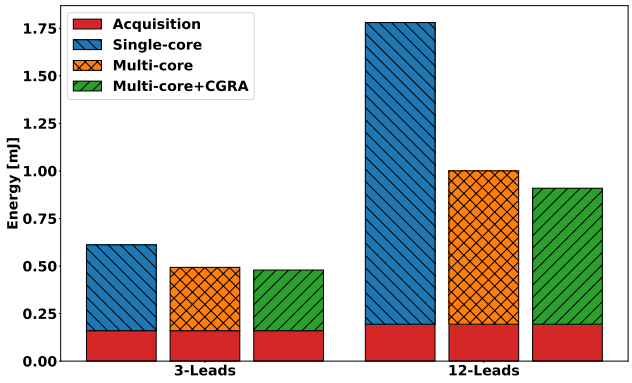

Fig. 6. Decomposition of energy consumption for the 3-leads and 12-leads ECG applications for PULP, including memory scaling.

We investigate the use of HW acceleration for the cases of 3-lead and 12-lead ECG signals, which can be observed in Fig. 6. The savings achieved by accelerating some intensive computational kernels in the 3-lead beat classifier application are 67 % in time and 2.9 % in energy compared to the multi-core implementation. The reason for the modest energy saving is the low computational load of the 3-lead application. Moreover, our minimalist CGRA design covers only a small amount of the total number of executed instructions, which limits its impact. Compared to the single-core implementation, it represents 21.6 % of energy savings. For the 12-lead application, the impact is more significant due to the higher computational load, with 9.6 % of additional energy savings compared to the multi-core implementation. However, as the figure shows, for low duty cycle applications, such as the 3-leads beat classifier, the energy consumed by the memories during sampling, although not dominant, is significant. In the case of the 12-lead application, the energy consumed during computation is much higher than the energy consumed by the memories during the sampling period (Fig. 6), hence the higher savings achieved. In fact, we scale the size of each memory bank from the original 64 KiB of [13] to 16 KiB and we apply memory management to keep only the bank needed by the application in active or retentive state. In applications with low computation load, one possible solution would be to design the SRAMs with a larger number of banks and scale to the feasible resolution to enable a more aggressive power management during data sampling periods.

Finally, in Table VI we show a summary of the energy savings compared to the single-core configuration applying the optimizations described in Section III. The three main optimizations, including parallelization, memory scaling and HW acceleration, can be applied orthogonally and significantly reduce the energy consumption compared to the traditional single-core implementation. For example, we can apply mem-

TABLE VI
SUMMARY OF ENERGY SAVINGS APPLYING THE SW PARALLELIZATION TECHNIQUES, THE HW ACCELERATION AND THE MEMORY SCALING FOR THE ANALYZED APPLICATIONS ON THE PULP PLATFORM

	Single-core Energy (mJ)	Multi-core (%)	Single-core + CGRA (%)	Single-core + Memory scaling (%)	Multi-core + CGRA (%)	Multi-core + Memory scaling (%)	Single-core + CGRA + Memory scaling (%)	Multi-core + CGRA + Memory scaling (%)
1 lead	0.43	6.51	3.61	23.45	6.59	29.95	27.05	**30.03**
1+2 leads	0.71	3.18	4.68	14.05	6.37	17.23	18.73	**20.42**
3 leads	0.71	16.56	4.68	14.05	18.58	30.60	18.73	**32.62**
12 leads	1.86	41.57	3.61	4.53	46.73	46.10	9.03	**51.26**

ory scaling directly to the single-core implementation and have energy savings of up to $23.45\,\%$ (this result corresponds to the value of the first column of Table V within a small rounding error). Additionally, we can apply HW acceleration not only on the multi-core implementation but on the single-core design, achieving energy savings from $9.03\,\%$ up to $27.05\,\%$. Therefore, the designer of WSN-based biomedical applications should take into account modularity and parallel implementation, memory scaling and HW acceleration.

VI. CONCLUSIONS

Modern ultra-low power WSN platforms offer characteristics such as multiprocessing, clock- and power-gating that enable power and memory management and HW acceleration. In this work, we have proposed a top-down approach to expose these characteristics to the SW layers via parallelization techniques to improve the mapping of modular biomedical applications. Additionally, we have shown how heterogeneous platforms can benefit from domain-specific accelerators, such as CGRAs, and memory scaling to further reduce energy consumption.

We have demonstrated our proposal on a set of independent modules typical of WSN-based biomedical applications and on two composed multi-lead ECG-based applications. Our results show energy savings of up to $60\,\%$ for the RMS module and up to $41.6\,\%$ for a complete multi-core application processing 12-lead ECG signals for a general PULP platform. Furthermore, we demonstrated that memory scaling is an orthogonal optimization that can be exploited to achieve additional energy savings up to $23.45\,\%$. Finally, our experiments have also established that our domain-specific accelerator can increase the energy savings to $46.7\,\%$ for the 12-lead delineation and $18.6\,\%$ for the complete heartbeat classifier. Thus, the overall combined energy savings reach up to $51.3\,\%$.

REFERENCES

[1] K. Xu *et al.*, "Public spending on health: A closer look at global trends," *WHO*, 2018.

[2] M. Al-khafajiy *et al.*, "Remote health monitoring of elderly through wearable sensors," *Multimedia Tools and Applications*, Jan. 2019.

[3] World Health Organization, "Cardiovascular diseases (CVDs)," *WHO*, 2018.

[4] A. Kumar, R. Komaragiri, and M. Kumar, "From pacemaker to wearable: Techniques for ecg detection systems," *Journal of Medical Systems*, vol. 42, p. 34, 02 2018.

[5] K. Guk *et al.*, "Evolution of wearable devices with real-time disease monitoring for personalized healthcare," *Nanomaterials*, vol. 9, no. 6, May 2019.

[6] F. Rincon, P. R. Grassi, N. Khaled, D. Atienza, and D. Sciuto, "Automated real-time atrial fibrillation detection on a wearable wireless sensor platform," in *Engineering in Medicine and Biology Society*. IEEE, Aug. 2012, pp. 2472–2475.

[7] G. Surrel, A. Aminifar, F. Rincon, S. Murali, and D. Atienza, "Online obstructive sleep apnea detection on medical wearable sensors," *IEEE Trans. on Biomedical Circuits and Systems*, pp. 1–12, Aug. 2018.

[8] M. Konijnenburg *et al.*, "A 769μW battery-powered single-chip SoC with BLE for multi-modal vital sign health patches," in *Int. Solid-State Circuits Conference (ISSCC)*. IEEE, Feb. 2019, pp. 360–362.

[9] F. Conti, D. Rossi, A. Pullini, I. Loi, and L. Benini, "PULP: A ultra-low power parallel accelerator for energy-efficient and flexible embedded vision," *Journal of Signal Processing Systems*, vol. 84, no. 3, Sep. 2016.

[10] L. Duch *et al.*, "HEAL-WEAR: An ultra-low power heterogeneous system for bio-signal analysis," *IEEE TCAS-I*, vol. 64, no. 9, pp. 2448–2461, Sep. 2017.

[11] S. Benatti *et al.*, "A sub-10mW real-Time implementation for EMG hand gesture recognition based on a multi-core biomedical SoC," in *Proc. of 7th International Workshop on Advances in Sensors and Interfaces, IWASI*. IEEE, Jul. 2017, pp. 139–144.

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

[13] A. Pullini, D. Rossi, I. Loi, G. Tagliavini, and L. Benini, "Mr.Wolf: An energy-precision scalable parallel ultra low power SoC for IoT edge processing," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 7, pp. 1970–1981, Jul. 2019.

[14] E. De Giovanni, S. Murali, F. Rincon, and D. Atienza, "Ultra-Low Power Estimation of Heart Rate under Physical Activity Using a Wearable Photoplethysmographic System," in *Proceedings - 19th Euromicro Conference on Digital System Design, DSD*, Oct. 2016.

[15] 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.

[16] A. Y. Dogan, J. Constantin, M. Ruggiero, A. Burg, and D. Atienza, "Multi-core architecture design for ultra-low-power wearable health monitoring systems," in *IEEE DATE*, Mar. 2012.

[17] E. De Giovanni *et al.*, "A patient-specific methodology for prediction of paroxysmal atrial fibrillation onset," in *Proc. of CinC*. IEEE, Sep. 2017.

[18] D. Sopic, E. De Giovanni, A. Aminifar, and D. Atienza, "Hierarchical cardiac-rhythm classification based on electrocardiogram morphology," in *Proc. of CinC*. IEEE, Sep. 2017, pp. 1–4.

[19] M. Jayawardhana and P. De Chazal, "Enhanced detection of sleep apnoea using heart-rate, respiration effort and oxygen saturation derived from a photoplethysmography sensor," in *Proc. of the Annual International Conference of the Engineering in Medicine and Biology Society, EMBS*. IEEE, sep 2017, pp. 121–124.

[20] G. Sannino and G. De Pietro, "A deep learning approach for ECG-based heartbeat classification for arrhythmia detection," *Future Generation Computer Systems*, vol. 86, pp. 446–455, Sep. 2018.

[21] P. Pławiak and U. R. Acharya, "Novel deep genetic ensemble of classifiers for arrhythmia detection using ECG signals," *Neural Computing and Applications*, pp. 1–25, Jan. 2019.

[22] F. I. T. Dell'Agnola, N. Momeni, A. Arza Valdes, and D. Atienza, "Cognitive workload monitoring in virtual reality based rescue missions with drones," in *12th International Conference on Virtual, Augmented and Mixed Reality in Copenhagen, Denmark*, 2020.

[23] R. Braojos *et al.*, "A synchronization-based hybrid-memory multi-core

architecture for energy-efficient biomedical signal processing," *IEEE Trans. on Computers*, vol. 66, no. 4, pp. 575–585, Apr. 2017.
[24] P. D. Schiavone, "zero-riscy: User Manual - PULP platform," Jan. 2018.
[25] P. Davide Schiavone *et al.*, "Slow and steady wins the race? A comparison of ultra-low-power RISC-V cores for Internet-of-Things applications," in *PATMOS*, Sep. 2017.
[26] J. L. Semmlow and B. Griffel, *Biosignal and Medical Image Processing*, 3rd ed. CRC Press, 2014.
[27] R. Braojos, G. Ansaloni, D. Atienza, and F. Rincon, "Embedded real-time ECG delineation methods: A comparative evaluation," in *Int. Conf. on Bioinformatics & Bioengineering (BIBE)*. IEEE, Nov. 2012.
[28] L. Orlandic *et al.*, "REWARD: Design, optimization, and evaluation of a real-time relative-energy wearable R-peak detection algorithm," in *Engineering in Medicine and Biology Conference (EMBC)*. IEEE, Jul. 2019.
[29] S. Yazdani, "Novel Low Complexity Biomedical Signal Processing Techniques for Online Applications," Ph.D. dissertation, École Polytechnique Fédérale de Lausanne, 2018.
[30] J. P. Martínez, R. Almeida, S. Olmos, A. P. Rocha, and P. Laguna, "A wavelet-based ECG delineator: evaluation on standard databases." *IEEE Trans. Biomed. Eng.*, vol. 51, no. 4, pp. 570–81, Apr. 2004.
[31] P. Kligfield *et al.*, "Recommendations for the standardization and interpretation of the electrocardiogram," *Circulation*, vol. 115, no. 10, pp. 1306–1324, 2007.
[32] R. Braojos, G. Ansaloni, and D. Atienza, "A methodology for embedded classification of heartbeats using random projections," in *IEEE DATE*, May 2013.
[33] M. Malik *et al.*, "Heart rate variability: Standards of measurement, physiological interpretation, and clinical use," *European Heart Journal*, vol. 17, no. 3, pp. 354–381, Mar. 1996.
[34] M. La Scala, G. Sblendorio, and R. Sbrizzai, "Parallel-in-time implementation of transient stability simulations on a transputer network," *IEEE Trans. on Power Systems*, vol. 9, no. 2, pp. 1117–1125, May 1994.
[35] V. Mondéjar-Guerra, J. Novo, J. Rouco, M. G. Penedo, and M. Ortega, "Heartbeat classification fusing temporal and morphological information of ECGs via ensemble of classifiers," *Biomedical Signal Processing and Control*, vol. 47, pp. 41–48, Jan. 2019.
[36] H. Singh *et al.*, "MorphoSys: An integrated reconfigurable system for data-parallel computation-intensive applications," *IEEE Transactions on Computers*, vol. 49, no. 5, pp. 465–481, May 2000.
[37] (2019) GitHub - pulp-platform/pulp-sdk.
[38] S. S. Basu, "Hardware/software co-design and reliability analysis of ultra-low power biomedical devices," Ph.D. dissertation, EPFL, Lausanne, 2019.
[39] P. Laguna, R. Mark, A. Goldberg, and G. Moody, "A database for evaluation of algorithms for measurement of QT and other waveform intervals in the ECG," in *Computers in Cardiology*. IEEE, 1997, pp. 673–676.
[40] G. Moody and R. Mark, "The impact of the MIT-BIH Arrhythmia Database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

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.

Fabio Montagna received the Ph.D. degree in Electrical, Electronic and Information Engineering from the University of Bologna, Bologna, Italy, in 2020. He is currently working as Research Fellow at DISI, University of Bologna, Bologna, Italy. His main research topic is energy-efficient parallel architectures for ultra-low power biosignal processing. His research interests include embedded wearable and implantable systems, parallel computing, signal processing, and machine learning.

Benoît W. Denkinger is currently pursuing the PhD degree with the Embedded Systems Laboratory (ESL), École polytechnique fédérale de Lausanne (EPFL), Lausanne, Switzerland. His research interests include low-power architectures for biomedical applications and artificial intelligence (AI)-enabled Internet-of-Things (IoT) devices. He has an MSc in robotics and autonomous systems from the Institute of Electrical Engineering, EPFL.

Simone Machetti received his M.Sc. degree in Computer Engineering, with major in Embedded Systems, from Politecnico di Torino, Italy. He worked as a Firmware Engineer at SPEA, a world leading company in the design and manufacture of Automatic Test Equipment (ATE). He is currently a PhD student at the Embedded Systems Laboratory (ESL) of EPFL, Switzerland. His research interests include hardware and software co-design for ultra-low-power embedded devices and Artificial Intelligence algorithms for the Internet of Things (IoT).

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.

Simone Benatti received the Ph.D. degree in electrical engineering and computer science from the University of Bologna, Bologna, Italy, in 2016. He currently holds a postdoctoral position with the University of Bologna. His research interests include energy efficient embedded wearable systems, signal processing, sensor fusion and actuation systems. This includes hardware/software co-design to efficiently address performance, as well as advanced algorithms. In this field, he has authored/coauthored more than 25 papers in international peer-reviewed conferences and journals. He has collaborated with several international research institutes and companies. Previously, he worked eight years as an Electronic Designer and R&D Engineer of electromedical devices.

Davide Rossi received the PhD from the University of Bologna, Italy, in 2012. He has been a post doc researcher in the Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi" at the University of Bologna since 2015, where he currently holds an assistant professor position. His research interests focus on energy efficient digital architectures in the domain of heterogeneous and reconfigurable multi and many-core systems on a chip. This includes architectures, design implementation strategies, and runtime support to address performance, energy efficiency, and reliability issues of both high end embedded platforms and ultra-low-power computing platforms targeting the IoT domain. In these fields he has published more than 100 papers in international peer-reviewed conferences and journals. He is recipient of Donald O. Pederson Best Paper Award 2018, - 2020 IEEE Transactions on Circuits and Systems Darlington Best Paper Award, 2020 IEEE Transactions on Very Large Scale Integration Systems Prize Paper Award.

Luca Benini holds the chair of digital Circuits and systems at ETHZ and is Full Professor at the Università di Bologna. He received a PhD from Stanford University in 1997. Dr. Benini's research interests are in energy-efficient parallel computing systems, smart sensing micro-systems and machine learning hardware. He has published more than 1000 peer-reviewed papers and five books. He is a Fellow of the ACM and a member of the Academia Europaea. He is the recipient of the 2016 IEEE CAS Mac Van Valkenburg award and of the 2019 IEEE TCAD Donald O. Pederson Best Paper Award.

Prof. David Atienza (M'05-SM'13-F'16) is associate professor of electrical and computer engineering, and heads the Embedded Systems Laboratory (ESL) at EPFL. He received his Ph.D. degree in computer 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 IoT systems, including new thermal-aware design for MPSoCs and many-core servers, and ultra-low power edge AI architectures for IoT. He has co-authored more than 300 papers, several book chapters, and eleven patents. He 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, and the ACM SIGDA Outstanding New Faculty Award in 2012. He is an IEEE Fellow, an ACM Distinguished Member, and has served as IEEE CEDA President (period 2018–2019).