Algorithm-Driven Architectural Design Space Exploration of Domain-Specific Medical-Sensor Processors

Mohammed Shoaib, *Student Member, IEEE*, Niraj K. Jha, *Fellow, IEEE*, and Naveen Verma, *Member, IEEE*

Abstract— **Data-driven machine-learning techniques enable the modeling and interpretation of complex physiological signals. The energy consumption of these techniques, however, can be excessive, due to the complexity of the models required. In this paper, we study the tradeoffs and limitations imposed by the energy consumption of high-order detection models implemented in devices designed for intelligent biomedical sensing. Based on the flexibility and efficiency needs at various processing stages in data-driven biomedical algorithms, we explore options for hardware specialization through architectures based on custom instruction and coprocessor computations. We identify the limitations in the former, and propose a coprocessor-based platform that exploits parallelism in computation as well as voltage scaling to operate at a subthreshold minimum-energy point. We present results from post-layout simulation of cardiac arrhythmia detection with patient data from the MIT-BIH database. After wavelet-based feature extraction, which consumes 12.28 μJ, we demonstrate classification computations in the 12.00–120.05 μJ range using 10 000–100 000 support vectors. This represents 1170× lower energy than that of a low-power processor with custom instructions alone. After morphological feature extraction, which consumes 8.65 μJ of energy, the corresponding energy numbers are 10.24–24.51 μJ, which is 1548× smaller than one based on a custom-instruction design. Results correspond to V$_{dd}$ = 0.4 V and a data precision of 8 b.**

Index Terms— **Biomedical sensor processors, classification accelerators, embedded machine learning, low-energy design by voltage and precision scaling, structured hardware specialization, support-vector machines.**

I. Introduction

BIOMEDICAL devices are striving to provide high-value analysis of physiological signals. Through continuous analysis of such signals, outpatient monitoring networks, for instance, promise comprehensive healthcare delivery over large populations and potentially diverse disease states [1]. The central need, as these systems aim to provide actionable outputs [2], is the ability to detect specific physiological states of interest from signals that are available through minimally invasive and low-power sensors. This poses two essential challenges: 1) the signal correlations to clinically relevant states are often too complex to model based on physiology and 2) the precise correlations are hard to isolate in the presence of normal physiologic activity. In the case of arrhythmia detection based on electrocardiogram (ECG) sensing, for instance, abnormal ECG beat morphologies exhibit only subtle variations compared to normal waveform activity. Their discrimination from each other and from normal beats poses a major challenge for signal detection and analysis.

Both the targeted and background variations are often expressed through specific manifestations in physiological waveforms. A potential key to accurate signal detection could thus lie in the ability to develop models based on representative or exemplary data. In fact, data-driven modeling techniques are emerging as a powerful approach for overcoming the mentioned challenges [3]. This, in large part, has been prompted by the recent large-scale availability of data in the healthcare domain as well as the development of machine-learning techniques that are capable of exploiting large amounts of data to model specific correlations and then use the models within a decision function [4]. Furthermore, data-driven modeling often involves probabilistic methods for model construction and can thus also tolerate substantial variations due to background physiologic activity [5].

For the systems of interest, however, the computations involved must be achieved at very low power levels (e.g., 1–10 mW for wearable devices and 10–100 μW for implantable devices [6]). In the recent literature, low-power processing platforms for biomedical detection have emphasized an optimization of signal-processing computations, largely based on synthesis and analysis operations [7]–[9]. In contrast to these techniques, data-driven methods focus on modeling and classifying signals by exploiting potentially rich data that are available. With regard to data-driven methods, applications have been based not on low-power sensors, but rather on high-performance computing clusters with substantially relaxed energy constraints [10], [11].

In the biomedical domain, chronic patient monitoring devices are beginning to exploit data-driven techniques, but have thus far been limited in their ability to incorporate the complete computation [2], [12], [13]. It has been shown, for instance, that local feature computations can be performed on the signal, reducing the communication data so that computationally intensive data-driven classification can be performed on a separate device [12]. Another direction for medical sensors has been to employ application-specific architectures for low energy [14], [15]. Such systems, however,

Manuscript received March 1, 2012; revised June 3, 2012; accepted August 31, 2012. This work was supported in part by the Qualcomm Innovation Ph.D. Fellowship and the Gigascale Systems Research Center, one of six research centers funded under the Focus Center Research Program, the Semiconductor Research Corporation Entity. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Seda O. Memik.

The authors are with the Department of Electrical Engineering, Princeton University, Princeton, NJ 08544 USA (e-mail: mshoaib@princeton.edu; jha@princeton.edu; nverma@princeton.edu).

Digital Object Identifier 10.1109/TVLSI.2012.2220161

lack applicability to a broader range of detection problems. In [16], we presented a specialized computational platform for data-driven biomedical monitoring. In this paper, we investigate the principles behind a hardware-specialized architecture for data-driven biomedical algorithms by quantitatively evaluating various platform options, from general-purpose CPUs to custom instructions to hardware coprocessors. For the coprocessor approaches, we also evaluate the potential of microarchitecture and circuit-level opportunities, such as dynamic voltage and precision scaling. We thus propose an algorithm-driven methodology that takes advantage of the computational structure and the characteristics of data-driven patient monitoring algorithms. Our specific contributions are as follows.

1) We present an energy analysis of representative biomedical detection applications (cardiac arrhythmia detection is considered in detail). The analysis is based on patient data from the MIT-BIH database [17] and shows that classification, the complexity of which depends on the characteristics of the data, poses the primary energy limitation. We also show that the computational structure of classification limits the energy savings attainable through the use of custom instructions.
2) Based on the energy analysis and the computational requirements of various parts of the algorithm, we propose a generalized architecture for a biomedical computation platform. This attempts to employ programmability where computational flexibility is required, while leveraging hardware specialization for classification, where set computations are required at very high energy efficiency.
3) We propose a transistor-level design of a classification coprocessor that leverages a low-power technology (i.e., low-leakage FD-SOI). Specific requirements for computational flexibility are identified and incorporated through hardware scalability in a parallelized subthreshold implementation that operates at the minimum-energy supply voltage.

The rest of this paper is organized as follows. In Section II, we present an analysis of the data-driven biomedical algorithms and identify the computational steps involved. In Section III, we explore the architecture of a low-energy data-driven computational platform through custom instructions and a coprocessor. In Section IV, we describe specialized circuits at the transistor level, including a variable-precision multiply-accumulate (MAC) unit for the coprocessor-based implementation. In Section V, we present post-layout simulation results for the coprocessor. Finally, we conclude in Section VI.

II. Application-Domain Algorithmic Study

In this section, we describe the general computational structure of algorithms used for analyzing physiological signals. We focus on arrhythmia detection, which employs morphological and wavelet features and performs detection computations based on a support vector machine (SVM) classifier.

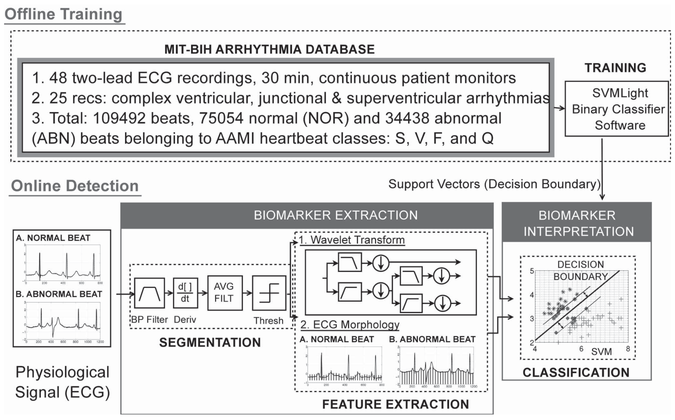
Offline Training
MIT-BIH ARRHYTHMIA DATABASE
1. 48 two-lead ECG recordings, 30 min, continuous patient monitors
2. 25 recs: complex ventricular, junctional & superventricular arrhythmias
3. Total: 109492 beats, 75054 normal (NOR) and 34438 abnormal (ABN) beats belonging to AAMI heartbeat classes: S, V, F, and Q
TRAINING
SVMLight Binary Classifier Software
Support Vectors (Decision Boundary)
Online Detection
BIOMARKER EXTRACTION
BIOMARKER INTERPRETATION
A. NORMAL BEAT
B. ABNORMAL BEAT
BP Filter
Deriv
AVG FILT
Thresh
SEGMENTATION
1. Wavelet Transform
2. ECG Morphology
A. NORMAL BEAT
B. ABNORMAL BEAT
FEATURE EXTRACTION
DECISION BOUNDARY
SVM
CLASSIFICATION
Physiological Signal (ECG)

Fig. 1. Structure of data-driven biomedical detection algorithms, including offline training and online detection (employing biomarker extraction and interpretation).

A. General Structure of Algorithms

Cardiac arrhythmias refer to abnormal heart beats that are indicative of a range of cardiovascular conditions. We focus on arrhythmia detection as an example in our algorithmic study.

Fig. 1 illustrates the structure of a typical machine-learning-based data-driven analysis algorithm. In the arrhythmia detection case study, ECG data are preprocessed for noise removal (band-pass filtering), QRS-complex detection [18], and beat segmentation [19]. Subsequently, the detection process involves the following two primary steps, namely, biomarker extraction, and biomarker interpretation (through a classifier) [20]. Biomarkers refer to specific signal parameters that are indicative of the physiological state of interest [21], [22]. For arrhythmia detection, a range of biomarkers have been used, including ECG morphology, beat intervals, and spectral features [23]–[26]. The diversity in the choice of biomarkers is due to the various clinical tradeoffs introduced by each, which can also be variable across patients [27], [28]. In this paper, we use two prominent biomarkers: waveform morphology [23] and spectral wavelets [29]. The associated processing steps, including segmentation (to isolate individual beats), are then implemented in software (enabling the energy analysis presented next). The outputs from these stages form the feature vectors that are used for classification.

Following the feature extraction process, an SVM is used for data-driven modeling and classification. SVMs are popular machine-learning classifiers that can be efficiently trained offline. This process results in a set of vectors, called support vectors (SVs), which are used to model the data by representing a decision boundary. Although training can be done offline, classification, through the application of the SV model, must be performed in real time for chronic detection. The actual classification computation is shown below for radial basis function (RBF) and polynomial transformation kernels

$$\textsc{Data Class} = \operatorname{sgn}\left[\sum_{i=1}^{N_{\mathrm{SV}}} K\left(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv}}_{\mathbf{i}}\right) \alpha_i y_i - b\right] \qquad (1)$$

where

$$K\left(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv}}_{\mathbf{i}}\right) = \begin{cases} exp(-\gamma\,||\vec{\mathbf{x}} - \vec{\mathbf{sv}}_{\mathbf{i}}||^2) & \text{RBF kernel} \\ F(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv}}_{\mathbf{i}} + \beta)^d & \text{Poly. kernel.} \end{cases}$$

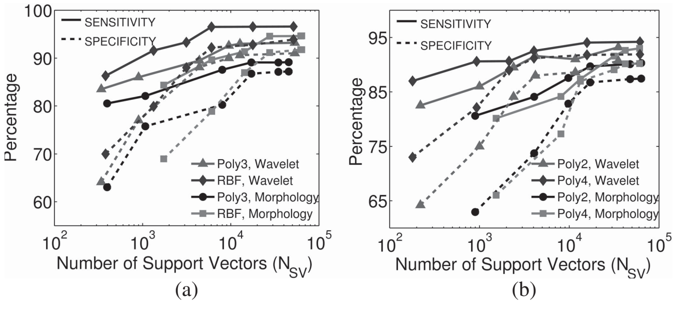

Fig. 2. Reducing model complexity by reducing NSV, degrading the accuracy of classification for (a) RBF and polynomial kernels of order 3 and (b) polynomial kernels of order 2 and 4.

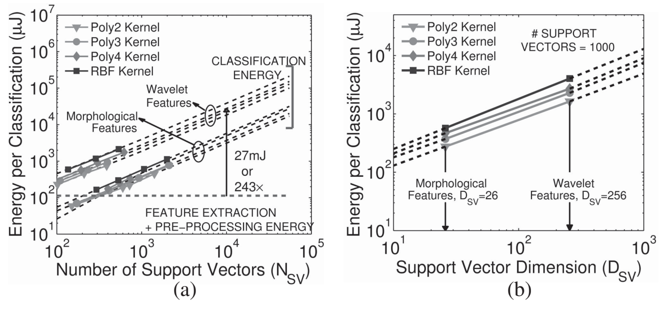

Fig. 3. Arrhythmia detector. (a) Classification energy scales with N_{SV} dominate those of feature extraction. (b) Classification energy scales with D_{SV}. N_{SV} and D_{SV} represent classification complexity.

TABLE I
XTENSA CUSTOM PROCESSOR CONFIGURATION

Parameter	Configuration
Instruction width	24 b
Pipeline length	5 stages
Pipeline type	Uniscalar
General-purpose registers	16
ALUs	1
Branch units	1
Core frequency	10 MHz
Instruction RAM	2 kB
Data RAM	4 kB
Datapath width	32 b
Disabled Options	
Multipliers (MUL32/MUL16), Viterbi unit, Single-cycle MAC, zero-overhead loop, Normalized shift, min/max unit, ICache/DCache associativity	

Here, $sgn[]$ is the signum function, $\vec{\mathbf{x}}$ is the feature vector to be classified, and $\vec{\mathbf{sv_i}}$ is the ith support vector (b, d, α_i, β, γ, and y_i are training parameters).

As the number of support vectors (N_{SV}) and the feature-vector dimensionality (D_{SV}) scale, the classification computations are dominated by the dot product between $\vec{\mathbf{x}}$ and $\vec{\mathbf{sv_i}}$. Further, in (2), K represents a kernel function, whose choice, along with N_{SV} and D_{SV}, can have a major impact on classifier complexity.

B. Need for Advanced Classification Models

In this section, we present the limitations of using simple classification models (which would address the complexity challenge described above). Experimental results show that, as the model complexity reduces, the classifier performance degrades, necessitating high-order models. The detection algorithms illustrated in Fig. 1 are implemented using SVM-Light [30], which is an open-source implementation of an SVM, for the classifier. To correctly analyze the computational complexities and tradeoffs imposed by the model, we use patient data from the MIT-BIH database [17].

If the feature vectors were linearly separable, a linear kernel function (k) could be used for classification [31]. The test vector could then be pulled out of the summation in (2), enabling precomputation of a single decision vector. As a result, even when N_{SV} scales, the classification energy can remain constant. However, biomedical applications have shown to perform poorly when linear decision functions are used with medical datasets [32]. Nonlinear functions, such as high-order polynomials, RBFs, or sigmoidal kernels, are thus needed for acceptable classifier accuracies [33]. Several biomedical detection systems in the literature employ nonlinear kernels for classification. For instance, using polynomial and RBF kernels, 15.8% and 47.4% improvements in detection accuracy are reported in [34] and [35], respectively.

The performance of the classifier depends on the SV model and the characteristics of the application data. Figs. 2(a) and (b) show how the sensitivity and specificity for arrhythmia detection degrade as N_{SV} is reduced.[1] In order to reduce N_{SV}, the training parameters b, d, α_i, β, γ, and y_i are adjusted along with the choice of the data subset used for training. Thus, the model complexity depends on the characteristics of the application data and introduces an unavoidable tradeoff with respect to accuracy performance. We next present an energy analysis of the end-to-end arrhythmia detection algorithm, which employs high-order detection models for accurate signal classification. This analysis will enable an architectural study toward a low-energy application-domain processor.

III. APPLICATION-DOMAIN ARCHITECTURAL STUDY

We take three approaches in our architectural study. First, we implement the entire arrhythmia detection algorithm on an embedded low-power base processor. We find that classification poses the energy bottleneck due to the complexity of the models required. We then explore opportunities for hardware specialization through custom instructions and finally through a coprocessor.

For the base processor, we use the Xtensa processor from Tensilica, which is a customizable and extensible platform [36]. A family of processors can be built around the base instruction set architecture (ISA) of the synthesizable Xtensa processor core [37], [38]. As a result, custom processor configurations can be obtained with optimized performance, power dissipation, code size, and die size. Design automation algorithms and tools for extensible and configurable processors are discussed in [39] and [40]. Design-space exploration of the

[1]Sensitivity = $(T_P/T_P + F_N)$ and specificity = $(T_N/T_N + F_P)$, where $T(F)_{N(P)}$ is the number of true (false) negatives (positives).

TABLE II
SOFTWARE ENERGY (PER TEST VECTOR) FOR PREPROCESSING AND FEATURE EXTRACTION ON THE BASE XTENSA PROCESSOR CORE

Computational Step	Energy/Test Vector
Preprocessing segmentation	84.02 μJ
Morphology feature extraction	2.61 μJ
Wavelet feature extraction	29.28 μJ

Tensilica processor parameters allows customizing the base processor to achieve minimum-energy consumption. The major parameters include the choice of multipliers (MUL32/MUL16), instruction/data cache sizes and associativities (range of 0–16 kB and 2–16), and datapath/instruction path widths (range of 8–32 b). We pick the design parameters that lead to minimum-energy consumption based on an initial parameter-space exploration. The configuration obtained for the base processor is shown in Table I. We present energy profiling results of the arrhythmia detector on the configured base processor.

A. Implementation on the Base Processor

In this section, we present energy analysis of a software implementation of the arrhythmia detector on the Tensilica base processor; initial design profiling leads to the configuration parameters shown in Table I. Note that even though classification involves dot product computations, including multipliers (MUL16 and MUL32) in the base processor leads to higher energy consumption. In Section III-B, we show that this energy increase is due to the overhead of fetching high-dimensional data for the multipliers.

The energy profiling results for the preprocessing and feature extraction steps are shown in Table II (results are shown at the operating frequency of 10 MHz and supply voltage of 1.2 V). A feature vector is derived every heart beat and consumes 84.02 μJ for segmentation, which includes the process of isolating individual heartbeats along with the filtering of noise and other interference sources. Subsequently, 2.61 and 29.28 μJ of energy is consumed for morphological and wavelet feature extraction, respectively.

Fig. 3(a) and (b) shows the energy of classification versus N_{SV} and D_{SV}, respectively. Both the number and dimensionality of the SVs are representative of the model complexity. It can be seen that, because of energy scaling, classification energy rapidly dominates that of feature extraction. $N_{SV} = 10\,000$, using wavelet features and a fourth-order polynomial kernel, for instance, leads to an energy consumption ratio of 941:3:1 for classification, preprocessing, and feature extraction, respectively. At $N_{SV} = 100\,000$, the ratio increases to 5172:3:1. As described in Section II-B, to avoid compromising accuracy, biomedical applications generally require complex models, causing the classification energy to be dominant. The energies reported in other biomedical applications exhibit a similar trend (e.g., seizure detection based on electroencephalograph classification [12]).

The output of the Xtensa profiling tool for classification computations is shown in Table III. The dot product computations required in (2) are shown in bold (sprod_ns and kernel).

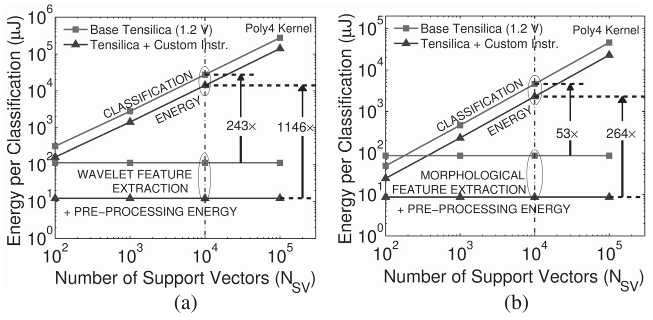

Fig. 4. Custom-instruction based implementation resulting in only modest energy improvement. Classification energy still dominates preprocessing + feature extraction (a) by 1146× for wavelet features and (b) by 264× for morphological features.

These functions correspond to the N_{SV} and D_{SV} analysis presented in Section II-B, and constitute over 70% of the computations. Thus, classification is the energy-intensive computational step and is targeted next for hardware specialization through custom instructions and a coprocessor.

B. Custom-Instruction-Based Platform

In this section, the use of custom instructions is explored as a hardware-specialization option. We primarily focus on the classifier because of its importance in determining the total energy. We find that the custom instructions are insufficient for achieving significant classifier energy savings. This is due to the overheads that remain for fetching high-dimensional data from memory. Thus, the energy reductions achievable through the use of custom instructions for classification are limited because of the large number of operands involved in the dot product computation.

The Tensilica Xpress compiler [36] is used to optimize the software implementation. This involves an automatic design space exploration of the potential custom instructions. SVM classification, wavelet transform with Daubechies wavelets of order four, and morphological feature extraction functions (including threshold selection and QRS isolation) are chosen for implementation as custom instructions. We perform Xpress synthesis using FLIX, Fusion, and SIMD instructions provided by Tensilica [36]. These design options provide optimization techniques, including automatic vectorization in the custom processor. Fusion instructions enable the lowest energy implementation. Table IV shows the top three custom instructions obtained for the feature-extraction and classification computations. The number of calls to each custom instruction is also shown as a percentage of the total instructions. We observe that there is no commonality in the custom instructions across feature-extraction and classification computations. Figs. 4(a) and (b) show the classification energy after a custom-instruction-based optimization (corresponding to wavelet and morphological features, respectively). The energy consumption is still substantially dominated by classification. At $N_{SV} = 10\,000$, for a fourth-order polynomial kernel, the ratios of energy consumption for classification, preprocessing, and feature extraction computations are 4432:3:1 and 720:3:1, for the wavelet and morphological features, respectively.

TABLE III
TENSILICA CODE PROFILER OUTPUT FOR THE SVM CLASSIFIER

Morphological Features, $D_{SV} = 26$, $N_{SV} = 10\,000$, Frequency = 10 MHz				
Function Name	Percentage Time (%)	Self Secs.	Cumulative Secs.	Number of Cycles
sprod_ns	**59.12**	**0.09**	**0.09**	**925 748**
kernel	**11.67**	**0.02**	**0.11**	**182 738**
smult_s	7.95	0.01	0.12	124487
add_vector_ns	6.54	0.01	0.13	102408

Wavelet Features, $D_{SV} = 256$, $N_{SV} = 10\,000$, Frequency = 10 MHz				
Function Name	Percentage Time (%)	Self Secs.	Cumulative Secs.	Number of Cycles
sprod_ns	**63.30**	**0.57**	**0.57**	**5 688 984**
kernel	**12.89**	**0.12**	**0.69**	**1 158 467**
smult_s	8.34	0.08	0.77	753 113
add_vector_ns	6.60	0.06	0.83	593 164

Self Secs. is the number of seconds accounted for by a particular function alone.
Cumulative Secs. is a running sum of the number of seconds accounted for by a function and those listed above it.

TABLE IV
CUSTOM INSTRUCTIONS FROM THE XPRESS COMPILER AT $N_{SV} = 10\,000$

Custom Instruction	**% of Total Instr.**
Preproc. + Morphology	
fusion.nop.loopgt.extui	32.6
fusion.abs.add8×8.extui	11.4
fusion.nop.neg8.extui	9.0
Preproc. + Wavelet	
fusion.ssl.mul8×16_0.extui	21.9
fusion.l8rzl.extai	17.2
fusion.movt.z.extui	8.4
Classification	
fusion.movi8×16.extui	13.1
fusion.add.sdd8×16.simcw.extui	5.4
fusion.sll.sub16×16_0.extui	5.3

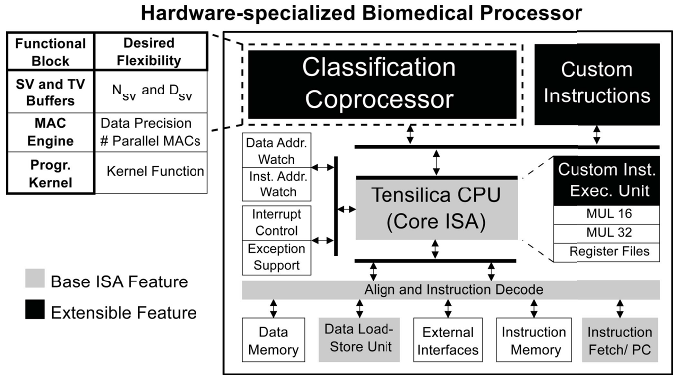

Fig. 5. General architecture of a biomedical processor.

Table V summarizes the resulting energy savings obtained through custom-instruction-based optimization. This optimization leads to roughly 10× energy improvement for the preprocessing and feature-extraction operations. However, optimization of classification computations leads to roughly only 2× reduction in energy. The limited energy reduction for classification is due to the large amount of data, which need to be fetched from memory, involved in the dot product computations (i.e., large N_{SV} and D_{SV}) [37], [39]. This limitation is not adequately addressed by custom-instruction-based optimization. To gain further intuition, consider [37], which is used to rank candidate templates for custom-instruction-based implementation

$$\text{Priority} = \frac{\text{OT}}{\max(\text{In} - \omega, 0) + \max(\text{Out} - \sigma, 0) + \rho}.$$

In the above equation, OT is the fraction of the total execution time of the original program spent in the template, In and Out are the number of inputs and outputs of the template, respectively, ω is the number of inputs that can be encoded in the instruction, σ is the number of outputs that can be encoded, and ρ is the number of cycles required by the template when implemented as a custom instruction. For custom instruction candidate templates for classification computations, OT has a large value of 0.70 (according to Table III). However, the high-dimensional input vectors (corresponding to D_{SV}) and the large number of cycles ρ required to fetch a high-order decision model (corresponding to N_{SV}) reduce the priority as a potential choice for custom-instruction-based implementation. Choosing custom instructions for the dot product computation based on an alternate priority function would still result in suboptimal energy savings. This is because the processor architecture would limit the data width in the classifier custom instructions, necessitating additional cycles for load-store operations. Thus, system memory overheads offset the benefits of custom-instruction-based speedup in classification.

The use of a vector processor core for handling high-dimensional input data can thus provide a potential solution for the classification computations. However, since the application data are not inherently vector in nature, such an architecture incurs unnecessary overheads for the general-purpose computations required, i.e., the feature-extraction computations required across clinical applications. Rather, the representation of data in a vector form is a specific transformation introduced by the classification framework. Thus, the complexity and associated energy overheads [41] incurred by a vector-processor-based implementation of the

TABLE V
SPEEDUP AND ENERGY SAVING USING CUSTOM INSTRUCTIONS WITH A BASE XTENSA PROCESSOR RUNNING AT $V_{dd} = 1.2$ V AND 10 MHz, WITH $N_{SV} = 10\,000$

Processor Configuration		**Preproc. + Morphology**	**Preproc. + Wavelet**	**Classification**	
				$D_{SV} = 26$	$D_{SV} = 256$
Base Xtensa	No. Cycles	27.56k	36.04k	1.57M	8.99M
	Energy (μJ)	86.62	113.29	4935.2	27574
Base Xtensa + Custom Instr.	No. Cycles	2.75k	3.91k	0.78M	4.58M
	Energy (μJ)	8.65	12.28	2455.3	14068
ENERGY IMPROVEMENT		**10.01x**	**9.23x**	**2.01x**	**1.96x**

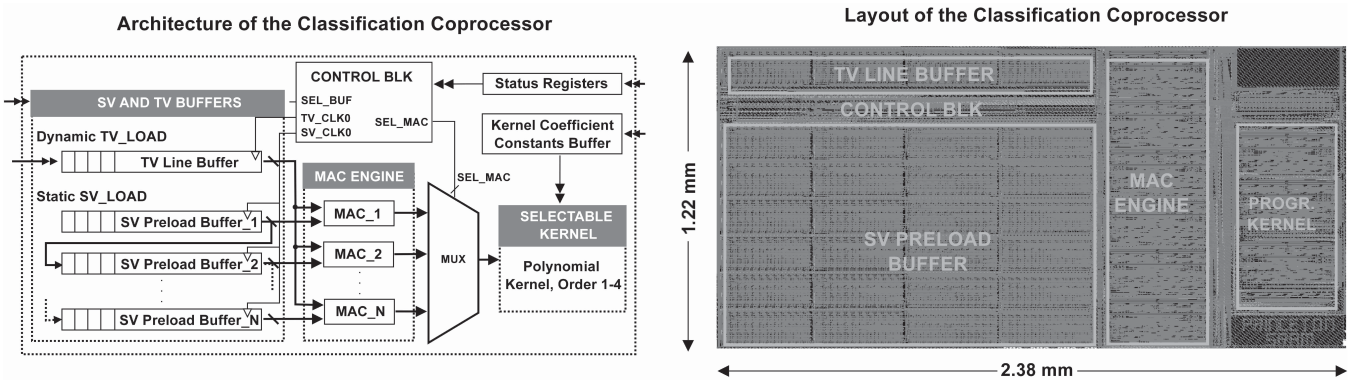

Fig. 6. Architecture and layout of the classification coprocessor designed in an FD-SOI process. The coprocessor architecture comprises an array of MAC units to compute the dot product of SVs and TVs. The output is then transformed by a kernel function in order to evaluate the classification result.

preprocessing and feature-extraction steps are best avoided for a platform-level design. To exploit both the canonical computations and data structuring required in the classification framework, we next turn our attention to coprocessor-based hardware specialization.

C. Coprocessor-Based Platform

In this section, we discuss a coprocessor-based specialization that allows the data structures used by the classifier to be efficiently handled, yielding substantial energy savings. Further, this degree of specialization raises the opportunity for microarchitectural optimizations based on parallelism, which can be readily exploited in the computation.

The architecture proposed in Fig. 5 aims to take advantage of the structure of biomedical algorithms where a high degree of flexibility is primarily required in the feature computations. The need for flexibility arises because of the range of feature computations involved in various applications. For instance, morphological and wavelet features are employed in arrhythmia detection [23], [29], proteomic classification [42], and heart-rate estimation [43]; spectral features are employed in seizure detection [12], brain–machine interfaces [44], and sleep disorder analysis [45].

Thus, a general-purpose processor is employed for feature computation, while an optimized coprocessor is employed for kernel-based SVM classification. The feature-extraction computations are optimized through custom instructions, providing significant energy savings, as shown in Table V. These computations involve floating-point operations in the Tensilica CPU, incurring somewhat higher energy than a fixed-point implementation. However, as explained in the previous section, the contribution of the feature-extraction energy to the overall processor energy is very small. Thus, we focus on optimizing the coprocessor. Here, in addition to hardware specialization, circuit and microarchitectural enhancements aim to achieve minimum-energy operation [46] through voltage scaling and parallelism, whereby the throughput constraints for real-time detection can be met. In addition to energy efficiency, the need for selective flexibility is also recognized so that the classification needs across a wide range of biomedical applications can be supported. For example, the rate of processing mass spectrometry data [42] could be different from EEG [12] or ECG signals [23], [29]. For these applications, N_{SV}, D_{SV}, data precision, as well as the kernel functions will also be different. The coprocessor thus introduces this flexibility through a precision-scalable multiplier. It also yields programmability in the classification model, computation precision, and the choice of kernel transformation function. These aspects are summarized in the block diagram of Fig. 5.

IV. LOW-ENERGY CLASSIFICATION COPROCESSOR

In this section, we describe the architecture of the classification coprocessor in further detail.

A. Coprocessor Microarchitecture

Fig. 6 shows the architecture and layout of the classification coprocessor. It has three major functional blocks: 1) SV and test vector (TV) buffers; 2) MAC engine; and 3) a programmable polynomial kernel core. Following offline training, SVs are loaded into the SV preload buffers. The TVs, produced through feature extraction by the general-purpose processor, are loaded into the TV line buffer. TVs and SVs are then fed dimension by dimension to the MAC array in order to perform the dot product operations in (2). Readout from these buffers is optimized using a multiplexer-based array decoder. Hardware parallelism is employed through an array of MAC units: MAC_1 to MAC_N, each of which is associated with an SV preload buffer. The coprocessor operates on integer data. Once multiplication over all the dimensions is complete, the dot products are multiplexed to the kernel transformation block, where a second-, third-, or fourth-order polynomial transformation is computed. Furthermore, a CORDIC module in the kernel block would potentially accommodate additional transformation functions, such as RBF, sigmoid, etc. The results are scaled and summed by a final accumulator whose output sign determines the classification result.

B. Voltage Scaling and Paralellism

In this section, we describe the energy optimization pursued for the coprocessor through voltage scaling. Since the dot product derivation (in the MAC array) dominates the computation, we focus on optimizing its energy.

The total energy is determined primarily by the sum of active switching (E_{act}) and subthreshold leakage (E_{lk}). The reduction in E_{act} due to V_{dd} scaling is opposed by the increase in leakage energy (due to the longer resulting leakage current integration time T_{MAC}). The energy-optimal point thus typically occurs in the subthreshold region, since here the circuit speed begins to degrade rapidly [46]. Although this implies that energy optimization leads to low circuit performance, computational throughput constraints can be efficiently met if the required computations can be performed in parallel without imposing substantial overheads due to parallelization [47]. We can thus exploit the parallelism possible in the classifier dot product computation (i.e., MAC array) to achieve minimum-energy operation for real-time biomedical detection. To do this, we first determine the minimum-energy V_{dd} of a MAC unit. We then determine its performance at this V_{dd} (i.e., seconds per MAC operation, T_{MAC}). The total rate of MAC operations ($R_{T.MAC}$) required in the classifier computation [of (2)] is given by

$$R_{T.\mathrm{MAC}} = \lceil N_{SV} \times D_{SV} \times R_{\mathrm{CLASS}} \rceil \qquad (2)$$

where R_{CLASS} is the classification rate. The required parallelism is then $R_{T.MAC} \times T_{MAC}$. For the application considered, the $R_{T.MAC}$ required ranges from 2.7 M to 7.7 M MACs per second [16].

Fig. 7 shows E_{act} and E_{lk} of a MAC unit (based on a transistor-level simulation) implemented in the target 150-nm FD-SOI CMOS process (described further in Section IV-D). The total energy E_{total} is minimized at a V_{dd} of 0.4 V, which is in the subthreshold region for the technology. Fig. 8 shows the performance achieved by a MAC unit as V_{dd} is scaled. Under worst case process and temperature conditions (i.e., low temperature in subthreshold), the maximum frequency at the minimum-energy V_{dd} is 520 kHz (i.e., $T_{MAC} = 1.92\ \mu$s). The level of parallelism required is thus 6–15 MAC units. Fig. 7, however, shows that the energy minimum is shallow, particularly if V_{dd} is increased slightly. For instance, to increase the MAC performance by a factor of three (in order to cover the target $R_{T.MAC}$ range), V_{dd} must be increased by less than 50 mV (based on Fig. 8), causing a negligible increase in total energy (based on Fig. 7). We thus optimize for the lower performance (by employing six MAC units) and use voltage scaling, with minimal impact on the optimization, to elevate the performance when required.

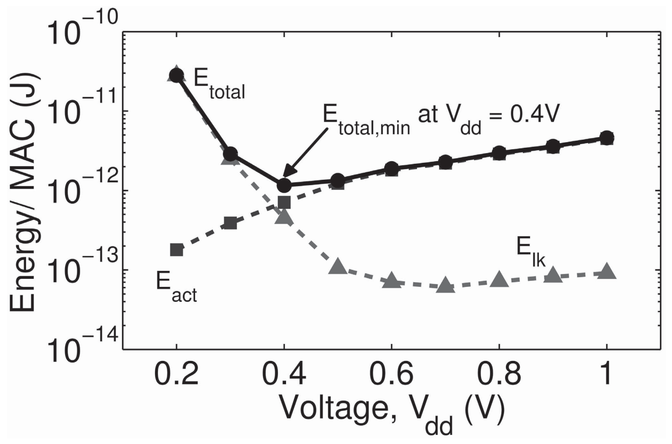

Fig. 7. E_{act} and E_{lk} profiles for a MAC unit with the minimum total energy occurring at $V_{dd} = 0.4V$.

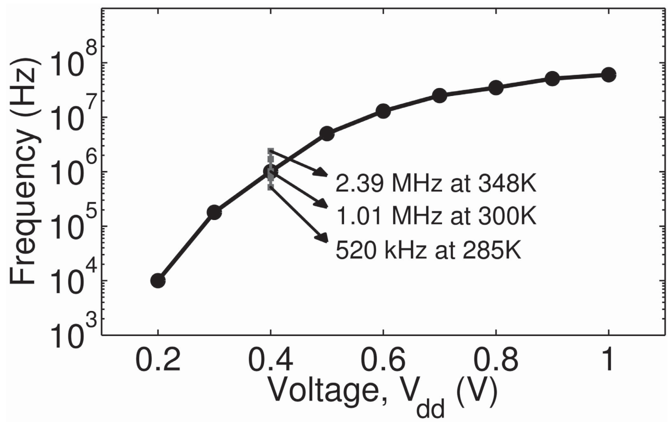

Fig. 8. Operating frequency at $V_{dd} = 0.4V$ is 520 kHz at 285 K (low temperature is slowest in subthreshold).

C. Circuit-Level Optimization

In this section, we describe how the scalability desired in the classification coprocessor is achieved.

1) SV and TV Buffers: The energy of the buffers is optimized for read operations since the SVs are loaded infrequently (i.e., only when a new classification model is required). The coprocessor buffers support a $D_{SV} \times N_{SV}$ of 64. If additional storage is required to represent the classification model, the control block permits expansion by allowing up to 16 384 write sequences from the processor cache or from off-chip memory to the local buffers. As an example, 4095

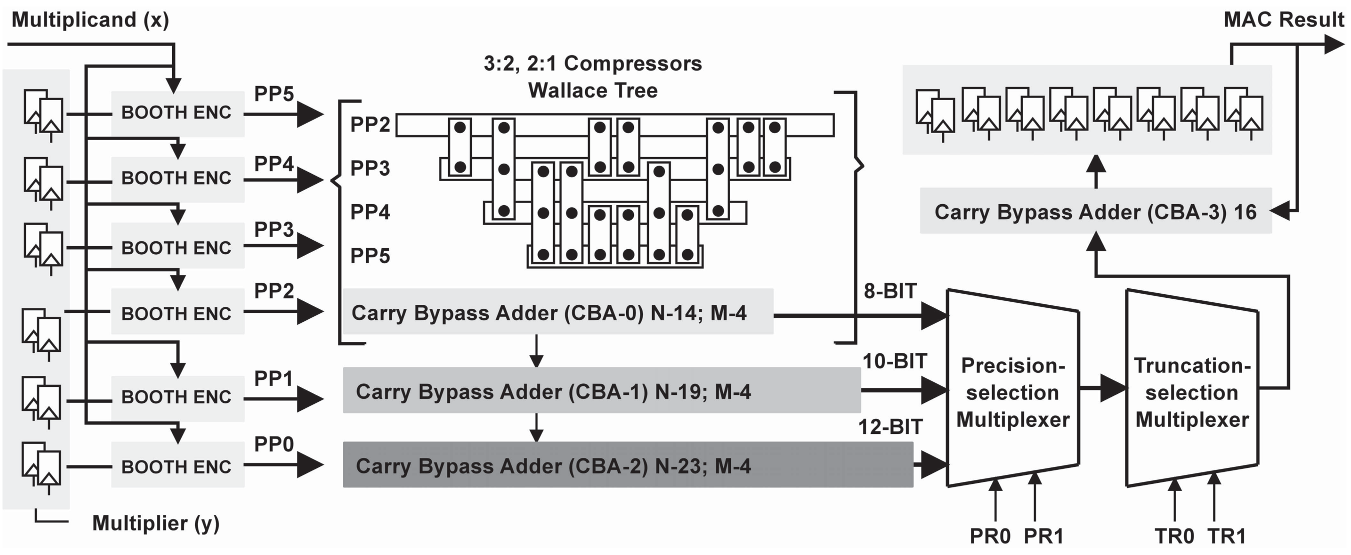
Multiplicand (x)
MAC Result
3:2, 2:1 Compressors Wallace Tree
BOOTH ENC
PP5
PP4
PP3
PP2
PP1
PP0
Carry Bypass Adder (CBA-0) N-14; M-4
Carry Bypass Adder (CBA-1) N-19; M-4
Carry Bypass Adder (CBA-2) N-23; M-4
Carry Bypass Adder (CBA-3) 16
8-BIT
10-BIT
12-BIT
Precision-selection Multiplexer
Truncation-selection Multiplexer
PR0 PR1
TR0 TR1
Multiplier (y)

Fig. 9. Variable-precision MAC unit. Partial product additions can be terminated at CBA-0/1/2 to scale the precision for 8/10/12-bit inputs.

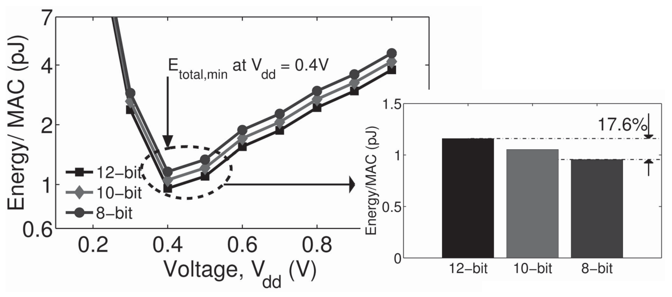

Fig. 10. Scaling precision of input data enabling a second level of energy optimization. The figure shows energy savings of 17.6% while scaling the data-representation precision from 12 to 8 b.

SVs and 256 feature dimensions can be supported, along with any other combination that results in the same product.

2) Variable-Precision MAC: Due to the wide range of SVs and feature dimensions across applications, the precision requirements of the classifier computation are variable. Several approaches for scalable-precision multipliers have been reported (e.g., [46]). The approach used here exploits the efficiency of the Booth encoding algorithm [48].

Fig. 9 shows the architecture of the variable-precision MAC unit. In the MAC unit, the BOOTH ENC blocks compute the partial products based on the select bits of the multiplier (y). The shifted partial products are output as PP_i, $i \in [0, 5]$. This allows a maximum precision of 12 b for the input operands (corresponding to six partial products). In Fig. 9, PP_0 and PP_1 are the partial products used when precisions of 12 and 10 b, respectively, are required; otherwise the precision is 8 b. The carry-bypass adders (CBAs) consist of $M = 4$-bit full adder chains, and N represents the total input bit width of each adder. The common partial products required for the 8/10/12 precision bits are added using 3:2 and 2:1 compressors in a Wallace tree. The outputs of CBA-0/1/2 are read out via a precision-selection multiplexer (for 8/10/12-bit precision, respectively). The unused CBAs can be power-gated. Fig. 10 shows the energy reductions due to precision scaling. Although the minimum-energy V_{dd} remains the same, scaling the precision from 12 to 8 b reduces the energy per multiplication by 17.6%. Following precision selection, the output of the multiplier has either a 24-, 20-, or 16-bit output. The truncation-selection multiplexer selects a level of truncation (to 12, 10, or 8 b, programmable via the status register). The output of the truncation-selection multiplexer is accumulated into an output register using a 16-bit final CBA.

3) MAC Delay Estimation: Based on the proposed MAC architecture (Fig. 9), the critical path delay through a MAC unit can be estimated as follows:

$$\begin{aligned} T_{\text{delay}} &= T_{CQ} + T_{\text{BOOTH}} + T_{\text{WAL}} + T_{\text{CBA,19-4}} + T_{\text{CBA,23-4}} \\ &\quad + 2T_{\text{AND}} + 3T_{\text{MUX}} + T_{\text{CBA,16-4}} + T_{SU} \end{aligned}$$

where $T_{\text{CBA,N-M}}$ is the delay of a CBA, which has a segment length of M and an operator bit length of N; T_{BOOTH} is the delay through the Booth encoder unit; T_{WAL} is the delay through the Wallace-tree compressor chain; T_{CQ} and T_{SU} are the clock-to-output and clock setup delays, respectively; and T_{AND} and T_{MUX} are the AND gate and multiplexer delays, respectively. Further, these delays can be simplified using a sum of delays through basic subblocks as follows:

$$\begin{aligned} T_{\text{CBA,N-M}} &= T_{SU} + MT_C\left(\frac{N}{M} - 1\right)T_{\text{MUX}} \\ &\quad + (M-1)T_C + T_S \\ T_{\text{BOOTH}} &= T_{\text{MUX}} + 4T_{\text{NOR}} + T_{\text{CBA,12-4}} \\ T_{\text{WAL}} &= 3T_S + T_{\text{CBA,14-4}} \end{aligned}$$

where T_C and T_S are the delays for the carry and sum paths in a full adder, respectively, and T_{NOR} is the NOR gate delay.

The delay through the critical path of the MAC unit can thus be estimated systematically. Estimating the MAC delay based on the performance of the mentioned subblocks thus facilitates rapid estimation of the maximum operating frequency, level of parallelism, and, hence, the associated system parameters,

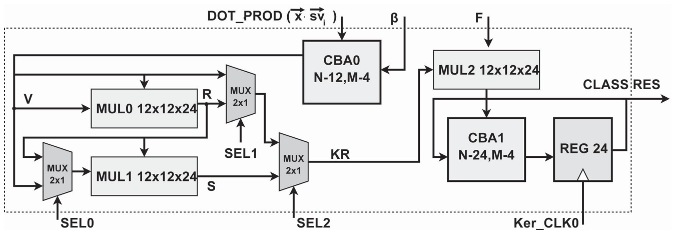

Fig. 11. Programmable kernel enabling choice among kernels of degree 1–4.

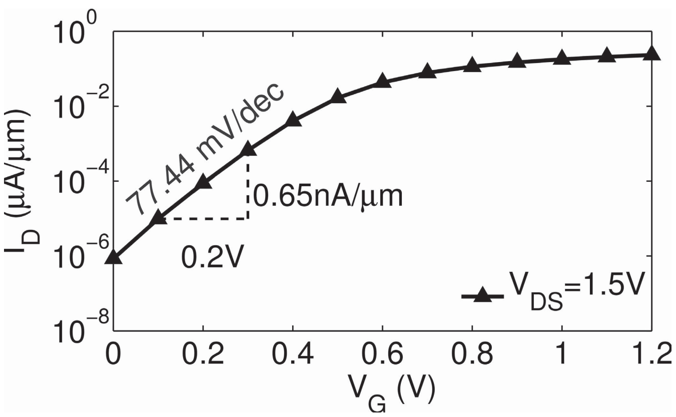

Fig. 12. FD-SOI device having a steep subthreshold slope to minimize leakage and maintain high transistor on-to-off ratios in subthreshold CMOS gates.

thereby overcoming the need for extensive transistor-level simulations in the early phases of system design.

4) Flexible Polynomial Kernel: A polynomial kernel can be selected to transform the dot product output from the MAC engine. The flexible kernel module comprises two $12 \times 12 \times 24$ multipliers to support polynomial transformations of order 2–4. Only one such multiplier is needed for a second-order polynomial function. Going from a second-order to a third-order polynomial kernel, however, incurs the cost of using an additional $12 \times 12 \times 24$ multiplier. Further, the difference between a fourth-order and a third-order polynomial function is only an additional array of multiplexers. These aspects are summarized in Fig. 11.

D. Choice of Technology

Owing to the modest performance requirements of typical biomedical applications (i.e., to the relatively low bandwidth of physiological signals), employing a technology that is aggressively optimized for low leakage is beneficial. As an example, for arrhythmia detection, a performance on the order of 5 million MACs per second is required. The technology we use in this paper is thus a 1.5 V 150-nm ultralow-leakage FD-SOI CMOS process [49]. FD-SOI allows the technology to exhibit reduced process variations due to a reduction in random dopant fluctuations and gives the transistors steep subthreshold slopes; the I_d–V_{gs} characteristic for the devices is shown in Fig. 12. Additionally, the devices are designed to have high threshold voltages to reduce the leakage current (i.e., $V_{t,N} = 0.65\ V$, $|V_{t,P}| = 0.53\ V$) [49].

V. Results and Analysis

In this section, we present post-layout simulation results of the coprocessor.

Table VI summarizes the impact of the architectural optimizations considered (which include feature extraction on the Tensilica processor and classification using custom instructions or a coprocessor). A 2× improvement in the energy consumption per SV dimension is obtained while going from an implementation on a base Tensilica processor (which consumes 11.85 *n*J) to a design with custom instructions (which consumes 5.89 *n*J). The limited energy reductions, as mentioned in Section III-B, is due to the high-dimensional input data required for classification. More aggressive specialization, through the use of a coprocessor, leads to a 519× energy reduction over the base case. Subsequent voltage scaling, taking advantage of the parallelism possible in the coprocessor implementation, leads to energy reductions of 2119× and 2231× for V_{dd} corresponding to 0.6 and 0.4 V, respectively.

A. Coprocessor Energy Measurements

In this section, we present an analysis of the energy consumption of the classification coprocessor versus N_{SV} and D_{SV}. Further, we also quantify the energy reductions achievable through voltage and precision scaling.

1) Energy Versus N_{SV} and D_{SV}: We perform energy measurements on the post-layout extracted netlist at various values of N_{SV} for the wavelet and morphological features. Fig. 13 shows the scaling in the energy consumption versus the number of SVs for the classification computation. The energy for classification scales roughly linearly with N_{SV}. A similar behavior is observed with D_{SV} (as shown in Fig. 14 where the energy numbers are provided at $V_{dd} = 1.2$ V). Fig. 13 shows results for a fourth-order polynomial kernel and spectral wavelet features ($D_{SV} = 256$) for arrhythmia detection. At 100 000 SVs, the optimization of a base Tensilica processor with custom instructions leads to an energy reduction by 1.96× (Section III-B). For the FD-SOI Coprocessor-based design, energy reductions by 228× are achieved at a supply voltage of 1.2 V. Voltage and precision scaling applied to the coprocessor leads to further energy reductions. For instance, as shown in Fig. 13, classification computations after wavelet feature extraction ($D_{SV} = 256$) consume 12.00–120.05 μJ using 10 000–100 000 SVs at 8 b of data precision and a supply voltage of 0.4 V. This represents 1170× lower energy than that of a Tensilica processor with custom instructions alone. Classification computations after morphological feature extraction, at the same precision and voltage levels, consume 10.24–24.51 μJ of energy, which is 1548× smaller than one based on a custom-instruction-based design. Detailed analysis results for the voltage- and precision-scaling experiments are presented next.

2) Energy Versus V_{dd}: E_{act} and E_{lk} measurements from the coprocessor are shown in Fig. 15, demonstrating the benefit of voltage scaling. The results are shown at 12 b of data precision. A second-order polynomial kernel is used for the simulations. On average, for a data precision of 12 b, E_{act} accounts for 98.1%, 95.1%, and 71.2% of the total energy at

TABLE VI
ENERGY PER SV DIMENSION

Tensilica (at 1.2 V)	**Tensilica + Custom Instr.**	**Tensilica + Custom Instr. + Coprocessor**		
		$V_{dd}^{\text{COPROC}} = 1.2\ V$	$V_{dd}^{\text{COPROC}} = 0.6\ V$	$V_{dd}^{\text{COPROC}} = 0.4\ V$
11.85 nJ – **1×**	5.89 nJ – ↓**2×**	22.84 pJ – ↓**519×**	5.59 pJ – ↓**2119×**	5.31 pJ – ↓**2231×**

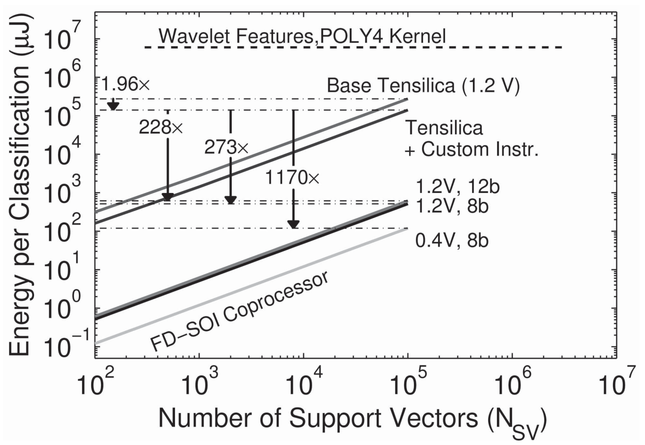

Fig. 13. Classification coprocessor enabling energy reductions of 228× at 1.2 V. Energy reductions are increased to 1170× at V$_{dd}$ = 0.4 V and 8 b of precision.

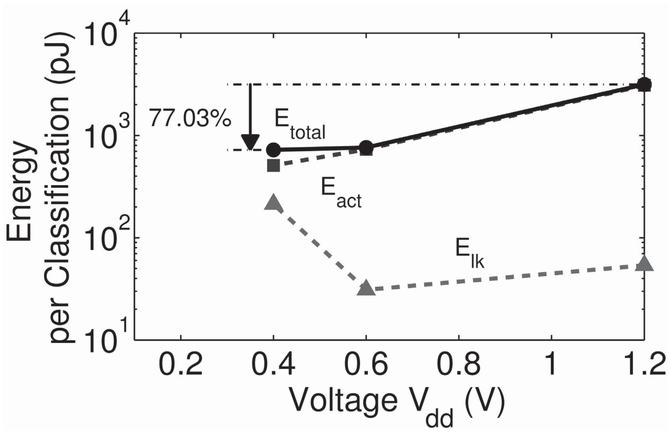

Fig. 15. Coprocessor energy versus V_{dd} per TV (at $N_{SV} = 10$, $D_{SV} = 8$, and 12-bit data precision). V_{dd} scaling enables energy reduction by up to 77%.

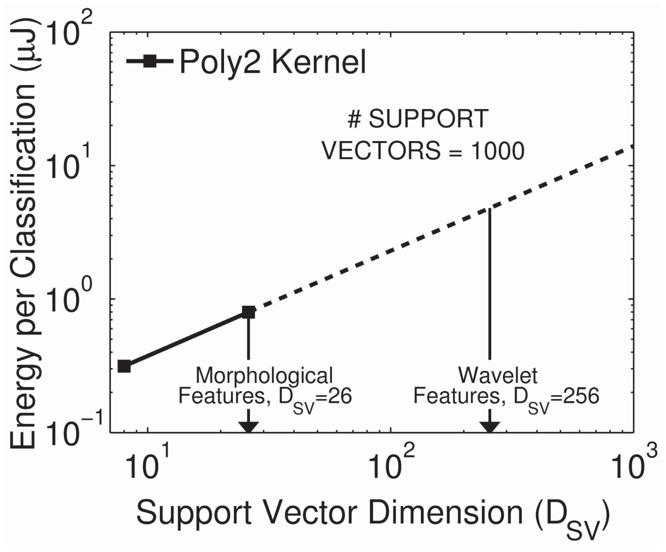

Fig. 14. Coprocessor energy versus D_{SV} per TV (at 12-bit data precision).

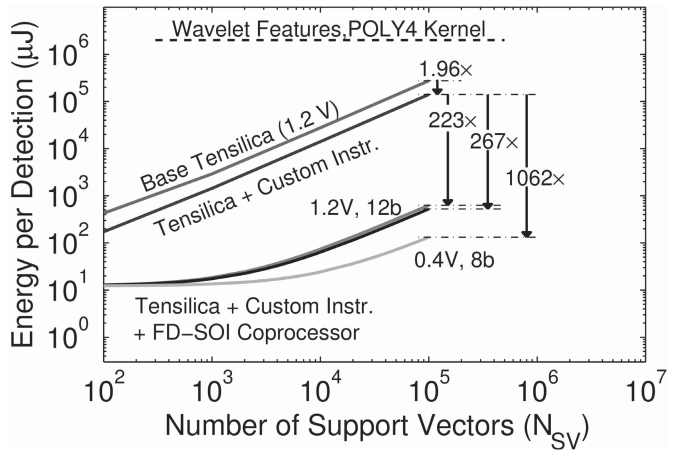

Fig. 16. Total application energy (segmentation + feature extraction + classification) reduced by up to 1062× through the use of a coprocessor for classification.

supply voltages of 1.2, 0.6, and 0.4 V, respectively. Voltage scaling thus enables energy reduction by up to 77%.

3) Energy Versus Precision: Table VII shows the E_{act} and E_{lk} measurements from post-layout simulation of the coprocessor using 8 b of data-representation precision. A second-order polynomial kernel is used for the results shown. On average, for a data precision of 12 b, E_{act} accounts for 98.1%, 95.2%, and 71.3% of the total energy at supply voltages of 1.2, 0.6, and 0.4 V, respectively. Consequently, performing computations at a data-representation precision of 8 b enables an overall 9.25%, 9.09%, and 9.09% reduction in total energy as compared to an implementation that relies on 12 b of data precision [most savings come from a 17.6% reduction from the MAC unit alone (see Fig. 10)].

B. Energy Versus Kernel Selection

Table VIII shows the energy consumption for various classifier kernels at a data precision of 12 and 8 b. Since the kernel transform is not the dominant computation, the energy scaling is modest.

On average, going from a second- to a third- to a fourth-order polynomial costs 30.92% and 7.56% extra energy for 12 b and 33.57% and 4.12% for 8 b of data-representation precision, respectively. The incremental change in energy between a third- and a fourth-order polynomial kernel is due to the optimization enabled by the programmable kernel architecture [16].

TABLE VII
PRECISION SCALING ENABLING UP TO 9.25% REDUCTION IN THE COPROCESSOR ENERGY

V_{dd} (V)	D_{SV}	N_{SV}	Precision	E_{act} (pJ)	E_{lk} (pJ)	E_{total} (pJ)		f_{op}, T = 287K
1.2	8	10	12 bit	3089.1	53.7	3142.8	**9.25%**↓	10 MHz
			8 bit	2799.8	52.4	2852.2		
0.6	8	10	12 bit	730.5	31.0	761.5	**9.09%**↓	2 MHz
			8 bit	667.2	25.1	692.3		
0.4	8	10	12 bit	508.7	213.2	721.9	**9.09%**↓	550 kHz
			8 bit	457.1	199.2	656.3		

TABLE VIII
COPROCESSOR ENERGY SCALING WITH RESPECT TO THE KERNEL FUNCTION

Specification	E_{act} (pJ)	E_{lk} (pJ)	E_{total} (pJ)		Kernel Order
V_{dd} = 1.2V, N_{SV} = 10, D_{SV} = 8, 12-bit precision	3078.7	64.1	3142.8	**30.92%**↓	Poly2
	4036.2	78.2	4114.4	**7.56%**↓	Poly3
	4344.4	81.1	4425.5		Poly4
V_{dd} = 1.2V, N_{SV} = 10, D_{SV} = 8, 8-bit precision	2799.8	52.4	2852.2	**33.57%**↓	Poly2
	3738.7	70.9	3809.6	**4.12%**↓	Poly3
	3914.6	74.7	3989.3		Poly4

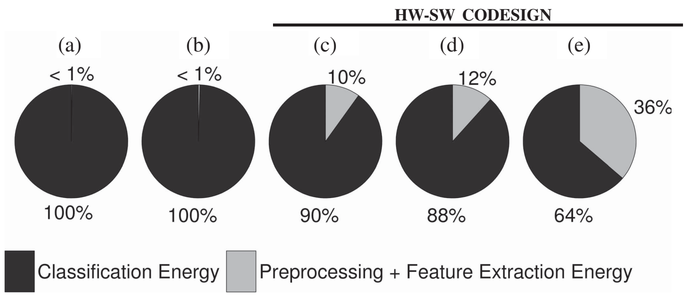

Fig. 17. Energy proportions for preprocessing, feature extraction, and classification computations, illustrating the benefits of voltage and precision scaling applied to the classification coprocessor for low-energy operation. (a) Base Tensilica processor operating at V_{dd} = 1.2 V. (b) Custom-instruction-based implementation. (c) Hardware–software codesign with custom instructions for preprocessing + feature extraction and the classification computations implemented on a coprocessor at V_{dd} = 1.2 V and 12 b of data precision. (d) Coprocessor at V_{dd} = 1.2 V and 8 b. (e) Coprocessor at V_{dd} = 0.4 V and 8 b.

C. Subblock Energy Measurements

Table IX shows the computational energy contributions for the coprocessor subblocks during online classification at a data-representation precision of 8 and 12 b. In the table, the energy consumed in the TV and SV buffers, the MAC array engine, kernel, and the control block are shown in the BUF, MAC, KER, and CNTRL columns, respectively. The MAC engine consists of six MAC units and the KER block consists of three MUL 12 × 12 × 24 multipliers, which are subblocks within a MAC unit. As shown, the MAC + KER energy dominates (∼ 84%), validating the need to focus on its energy through the optimization discussed in Section IV-B. Although the buffers dominate the transistor count, their low energy contribution shown in the table is due to the low leakage enabled by the choice of technology. The buffers have a very weak influence on the minimum energy point of the coprocessor due to the low activity factor and low leakage energy.

D. Processor-Level Energy Measurements

In this section, we present energy measurements for the entire processor. The architecture, which employs custom instructions for feature extraction and a coprocessor for classification, achieves over two orders of magnitude energy reductions as compared to an implementation that employs only custom instructions for the entire computation.

1) Energy Versus N_{SV}: In this section, we show energy scaling results versus N_{SV} for the full signal detection process (which, along with classification, includes the energy numbers for a custom-instruction-based implementation of preprocessing and feature extraction computations).

Fig. 16 shows simulation results for a fourth-order polynomial kernel and spectral wavelet features (D_{SV} = 256). Significant energy reductions are observed for the total detection process. For instance, using N_{SV} = 10 000, the total detection energy at 1.2 V and 12 b of data precision is 73.98 μJ. It is reduced to 63.69 μJ using 8 b of precision at 1.2 V. Voltage scaling applied to this optimized coprocessor configuration results in a total detection energy of 24.29 μJ at 0.4 V (this is about 580× lower than an implementation using a base Tensilica processor, which consumes 14.08 mJ at V_{dd} = 1.2 V).

2) Computational-Energy Contributions: Fig. 17 shows the proportions of energy consumption for the preprocessing, feature extraction, and classification computations following the various optimizations. The design space ranges from a full software-based Tensilica implementation to a coprocessor-based architecture. The proportions are shown for N_{SV} = 20 000. It is observed that, even after a custom-instruction-based optimization, the classification computations dominate feature extraction and preprocessing (less than 1% of the total energy). After an optimization through the use of the FD-SOI coprocessor, however, the classification energy is reduced substantially. The energy proportions are 1.8:1 for classification

TABLE IX
ENERGY (PER TV) AND AREA OF THE SUB-BLOCKS

Meas. Condition			**Total**	**BUF**	**MAC**, pJ	**KER**, pJ	**CNTRL**
V_{dd}	D_{SV}	N_{SV}	pJ	pJ	(% Total)	(% Total)	pJ
8-bit Data Precision							
1.2 V			**2852.2**	38.8	1511.8 (**53.0**)	900.9 (**31.6**)	400.7
0.6 V	8	10	**692.3**	18.9	360.7 (**52.1**)	220.2 (**31.8**)	92.5
0.4 V			**656.3**	18.5	343.8 (**52.4**)	203.9 (**31.1**)	90.1
12-bit Data Precision							
1.2 V			**3142.8**	40.2	1640.5 (**52.2**)	990.0 (**31.5**)	472.1
0.6 V	8	10	**761.5**	19.3	402.8 (**52.9**)	242.2 (**31.8**)	97.2
0.4 V			**721.9**	20.1	383.3 (**53.1**)	223.8 (**31.0**)	94.7
Area (in mm^2)			**2.90**	**1.62**	**0.61**	**0.65**	**0.02**

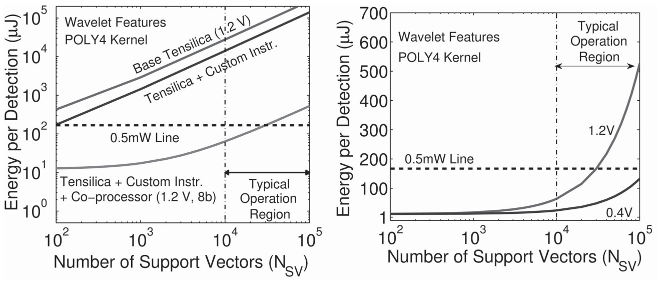

Fig. 18. Benefits of voltage scaling applied to the coprocessor (with the Tensilica processor at 1.2 V) using a fourth-order polynomial kernel and a beat classification rate (R_{CLASS}) of 3 beats/s with wavelet features ($D_{SV} = 256$).

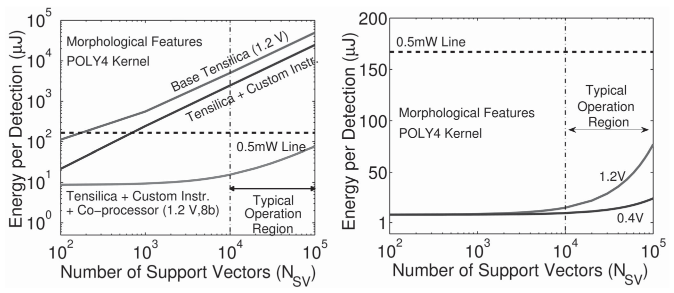

Fig. 19. Voltage scaling enabling the full detector computations at less than 500 μW. Results are shown for morphological features ($D_{SV} = 26$) with a quartic kernel at $R_{\text{CLASS}} = 3$ beats/s.

and preprocessing + feature extraction computations. However, at $N_{SV} = 100\,000$, the corresponding proportions are 9:1 (the total energy consumption for the associated computations being 120.32 μJ at a supply voltage of 0.4 V and a data-representation precision of 8 b).

3) Energy Versus V_{dd}: We next illustrate the benefits of voltage scaling on the coprocessor. This leads to operation of the arrhythmia detector at a power level < 500 μW.

Figs. 18 and 19 show the energy optimizations achieved for the detection process using a polynomial kernel of order four for the wavelet and morphological features, respectively. It is observed that voltage and precision scaling applied to the coprocessor enable computations in data-driven biomedical monitoring algorithms within an energy consumption range of 24.29–132.33 μJ for $N_{SV} = 10\,000$–$100\,000$, respectively (for wavelet features). Similar experiments with morphological features demonstrate the full detection process within 10.24–24.51 μJ for $N_{SV} = 10\,000$–$100\,000$, respectively (see Fig. 19).

VI. CONCLUSION

Machine-learning-based algorithms for biomedical detection are emerging as a highly promising means to detect specific states in physiologically complex signals. The structure in these algorithms can be exploited toward the design of a generalizable low-energy computation platform. In the detection algorithms, kernel-based classification is found to pose the primary energy bottleneck and is thus targeted for optimization through the use of hardware specialization. It is observed that, although feature-extraction computations can be implemented efficiently as custom instructions on a low-power processor, the energy reductions achievable through the use of custom instructions for classification are limited due to the large number of operands involved in the dot product computation. We thus explored opportunities for optimization through the use of a hardware coprocessor. This specialization provides an approach for hardware–software codesign expanding the scope of the biomedical processor architecture to a broader range of applications. In the classification coprocessor, the fixed kernel computations required were exploited, but selective flexibility required across a range of applications was also incorporated through specific hardware configurability. The optimized coprocessor reduced the computational energy of the biomedical platform by over three orders of magnitude compared to that of a low-power processor with custom instructions alone. Implementation of data-driven algorithms on a base Tensilica processor consumed about 100 mW for the entire computation. Thus, a wearable device (which runs on a typical 3 V, 560-mAh capacity coin-cell battery) employing a Tensilica-like processor would have an average recharge cycle of 16.8 h. If the computational power for the entire processor can be reduced by 2–3 orders of magnitude, continuous patient monitoring can become more viable (with battery lifetimes extended to 2–24 months). This introduces great promise for healthcare networks, as an increasingly wide range of real-time patient signal correlations are being discovered with new clinical states of interest.

References

[1] E. Dishman, "Inventing wellness systems for aging in place," *IEEE Comput.*, vol. 37, no. 5, pp. 34–41, May 2004.

[2] A. Csavoy, G. Molnar, and T. Denison, "Creating support circuits for the nervous system: Considerations for brain-machine interfacing," in *Proc. Int. Symp. VLSI Circuits Conf.*, Jun. 2009, pp. 4–7.

[3] D. Hau and E. Coiera, "Learning qualitative models from physiological signals," in *Proc. AAAI Symp. Artif. Intell. Med. Conf.*, 1994, pp. 67–71.

[4] G. Meyfroidt, F. Guiza, J. Ramon, and M. Bruynooghe, "Machine learning techniques to examine large patient databases," *Best Pract. Res. Clin. Anaesthesiol.*, vol. 23, no. 1, pp. 127–143, Mar. 2009.

[5] A. Shoeb, S. Schachter, D. Schomer, B. Bourgeois, S. T. Treves, and J. Guttag, "Detecting seizure onset in the ambulatory setting: Demonstrating feasibility," in *Proc. IEEE Int. Conf. Eng. Med. Biol.*, Sep. 2005, pp. 3546–3550.

[6] A. Chandrakasan, N. Verma, and D. Daly, "Ultralow-power electronics for biomedical applications," *Annu. Rev. Biomed. Eng.*, vol. 4, pp. 247–274, Aug. 2008.

[7] J. Kwong and A. P. Chandrakasan, "An energy-efficient biomedical signal processing platform," *IEEE J. Solid-State Circuits*, vol. 46, no. 7, pp. 1742–1753, Jul. 2011.

[8] H. Kim, R. F. Yazicioglu, T. Torfs, P. Merken, C. V. Hoof, and H.-J. Yoo, "An integrated circuit for wireless ambulatory arrhythmia monitoring systems," in *Proc. IEEE Int. Conf. Eng. Med. Biol.*, Sep. 2009, pp. 5409–5412.

[9] S. R. Sridhara, M. DiRenzo, S. Lingam, S.-J. Lee, R. Blazquez, J. Maxey, S. Ghanem, Y.-H. Lee, R. Abdallah, P. Singh, and M. Goel, "Microwatt embedded processor platform for medical system-on-chip applications," in *Proc. Symp. VLSI Circuits Conf.*, Feb. 2010, pp. 15–16.

[10] S. Cadambi, I. Durdanovic, V. Jakkula, M. Sankaradass, E. Cosatto, S. Chakradhar, and H. P. Graf, "A massively parallel FPGA-based coprocessor for support vector machines," in *Proc. Int. Symp. Field Program. Custom Comput. Mach. Conf.*, Apr. 2009, pp. 115–122.

[11] T.-W. Chen, C.-S. Tang, S.-F. Tsai, C.-H. Tsai, S.-Y. Chien, and L.-G. Chen, "Tera-scale performance machine learning SoC (MLSoC) with dual stream processor architecture for multimedia content analysis," *IEEE J. Solid-State Circuits*, vol. 45, no. 11, pp. 2321–2329, Nov. 2010.

[12] N. Verma, A. Shoeb, J. Guttag, and A. Chandrakasan, "A micro-power EEG acquisition SoC with integrated seizure detection processor for continuous patient monitoring," in *Proc. Symp. VLSI Circuits Conf.*, Jun. 2009, pp. 62–63.

[13] A. Shoeb, D. Carlson, E. Panken, and T. Denison, "A micropower support vector machine based seizure detection architecture for embedded medical devices," in *Proc. IEEE Int. Conf. Eng. Med. Biol.*, Sep. 2009, pp. 4202–4205.

[14] B. Gyselinckx, R. Vullers, C. V. Hoof, J. Ryckaert, R. F. Yaziciogl, P. Fiorini, and V. Leonov, "Human++: Emerging technology for body area networks," in *Proc. IEEE/IFIP Int. Conf. VLSI*, Oct. 2006, pp. 175–180.

[15] Z. D. Nie, L. Wang, W. G. Chen, T. Zhang, and Y. T. Zhang, "A low power biomedical signal processor ASIC based on hardware software codesign," in *Proc. IEEE Int. Conf. Eng. Med. Biol.*, Sep. 2009, pp. 2559–2562.

[16] M. Shoaib, N. K. Jha, and N. Verma, "A low-energy computation platform for data-driven biomedical monitoring algorithms," in *Proc. Design Autom. Conf.*, Jun. 2011, pp. 591–596.

[17] Physionet. (2012). *MIT-BIH Physionet Database*, Cape Town, South Africa [Online]. Available: http://www.physionet.org/physiobank/database

[18] J. Pan and W. J. Tompkins, "A real time QRS detection algorithm," *IEEE Trans. Biomed. Eng.*, vol. 32, no. 3, pp. 232–236, Mar. 1985.

[19] P. Laguna, N. V. Thakor, P. Caminal, R. Jane, H.-R. Yoon, A. B. D. Luna, V. Marti, and J. Guindo, "New algorithm for QT interval analysis in 24-hour Holter ECG: Performance and applications," *Med. Biol. Eng. Comput.*, vol. 28, no. 1, pp. 67–73, Jan. 1990.

[20] P. Sajda, "Machine learning for detection and diagnosis of disease," *Annu. Rev. Biomed. Eng.*, vol. 8, pp. 537–565, Aug. 2006.

[21] U. Manne, R.-G. Srivastava, and S. Srivastava, "Keynote review: Recent advances in biomarkers for cancer diagnosis and treatment," *Drug Dis. Today*, vol. 10, no. 14, pp. 965–976, Jul. 2005.

[22] T. Sunderland, R. E. Gur, and S. E. Arnold, "The use of biomarkers in the elderly: Current and future challenges," *Biol. Psych.*, vol. 58, no. 4, pp. 272–276, Aug. 2005.

[23] P. D. Chazal, M. O'Dwyer, and R. B. Reilly, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 7, pp. 1196–1206, Jul. 2004.

[24] T. H. Yeap, F. Johnson, and M. Rachniowski, "ECG beat classification by a neural network," in *Proc. IEEE Int. Conf. Eng. Med. Biol. Soc.*, Nov. 1990, pp. 1457–1458.

[25] O. Stanislaw and T. H. Linh, "ECG beat recognition using fuzzy hybrid neural network," *IEEE Trans. Biomed. Eng.*, vol. 48, no. 11, pp. 1265–1271, Nov. 2001.

[26] L. Senhadji, G. Carrault, J. J. Bellanger, and G. Passariello, "Comparing wavelet transforms for recognizing cardiac patterns," *IEEE Eng. Med. Biol. Mag.*, vol. 14, no. 2, pp. 167–173, Mar.–Apr. 1995.

[27] A. S. Jaffe, L. Babuin, and F. S. Apple, "Biomarkers in acute cardiac disease: The present and the future," *J. Amer. Coll. Cardiol.*, vol. 48, pp. 1–11, Jun. 2006.

[28] P. D. Chazal and R. B. Reilley, "A comparison of the ECG classification performance of different feature sets," in *Proc. IEEE Conf. Comput. Cardiol.*, Aug. 2000, pp. 327–330.

[29] E. D. Ubeyli, "ECG beats classification using multiclass support vector machines with error correcting output codes," *Digit. Signal Process.*, vol. 17, no. 3, pp. 675–684, May 2007.

[30] T. Joachims. (2010). *SVM-Light, Support Vector Machine* [Online]. Available: http://svmlight/joachims.org

[31] O. Chapelle and V. Vapnik, "Model selection for support vector machines," in *Advances in Neural Information Processing Systems*. Cambridge, MA: MIT Press, 2000, pp. 230–236.

[32] R. Somorjai, M. Alexander, R. Baumgartner, S. Booth, C. Bowman, A. Demko, B. Dolenko, M. Mandelzweig, A. Nikulin, and N. Pizzi, "A data-driven, flexible machine learning strategy for the classification of biomedical data," *Artif. Intell. Meth. Tools Syst. Biol., Comput. Biol.*, vol. 5, pp. 67–85, Jan. 2004.

[33] K. H. Lee, S.-Y. Kung, and N. Verma, "Improving kernel-energy trade-offs for machine learning in implantable and wearable biomedical applications," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Conf.*, May 2011, pp. 1597–1600.

[34] C. H. H. Tang, P. M. Middleton, A. V. Savkin, G. S. H. Chan, S. Bishop, and N. H. Lovell, "Non-invasive classification of severe sepsis and systemic inflammatory response syndrome using a nonlinear support vector machine: A preliminary study," *Physiol. Meas.*, vol. 31, no. 6, pp. 775–793, May 2010.

[35] A. H. Shoeb, "Application of machine learning to epileptic seizure onset detection and treatment," Ph.D. thesis, Dept. Electr. Med. Eng., Massachusetts Inst. Technol., Boston, Sep. 2009.

[36] Tensilica Inc. (2012). *The Xtensa Processor*, Santa Clara, CA [Online]. Available: http://www.tensilica.com

[37] F. Sun, S. Ravi, A. Raghunathan, and N. K. Jha, "Custom-instruction synthesis for extensible-processor platforms," *IEEE Trans. Comput.-Aided Design*, vol. 23, no. 2, pp. 216–228, Feb. 2004.

[38] F. Sun, S. Ravi, A. Raghunathan, and N. K. Jha, "Application-specific heterogeneous multiprocessor synthesis using extensible processors," *IEEE Trans. Comput.-Aided Design*, vol. 25, no. 9, pp. 1589–1602, Sep. 2006.

[39] F. Sun, S. Ravi, A. Raghunathan, and N. K. Jha, "A scalable synthesis methodology for application-specific processors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 11, pp. 1175–1188, Nov. 2006.

[40] F. Sun, S. Ravi, A. Raghunathan, and N. K. Jha, "A synthesis methodology for hybrid custom instruction and coprocessor generation for extensible processors," *IEEE Trans. Comput.-Aided Design*, vol. 26, no. 11, pp. 2035–2045, Nov. 2007.

[41] L. Seiler, D. Carmean, E. Sprangle, T. Forsyth, P. Dubey, S. Junkins, A. Lake, R. Cavin, R. Espasa, E. Grochowski, T. Juan, M. Abrash, J. Sugerman, and P. Hanrahan, "Larrabee: A many-core $\times 86$ architecture for visual computing," *IEEE Micro*, vol. 29, no. 1, pp. 10–21, Jan. 2009.

[42] F.-M. Schleif, M. Lindemann, M. Diaz, P. Maab, J. Decker, T. Elssner, M. Kuhn, and H. Thiele, "Support vector classification of proteomic profile spectra based on feature extraction with the bi-orthogonal discrete wavelet transform," *Comput. Vis. Sci.*, vol. 4, no. 12, pp. 189–199, Dec. 2007.

[43] N. Krupa, M. A. Mohammed, E. Zahedi, S. Ahmed, and F. M. Hassan, "Antepartum fetal heart rate feature extraction and classification using emperical mode decomposition and support vector machine," *Biomed. Eng. Online*, vol. 10, pp. 1–15, Jan. 2011.

[44] A.-T. Avestruz, W. Santa, D. Carlson, R. Jensen, S. Stanslaski, A. Helfenstine, and T. Denison, "A 5 μW/channel spectral analysis IC for chronic bidirectional brain-machine interfaces," *IEEE J. Solid-State Circuits*, vol. 43, no. 12, pp. 3006–3024, Dec. 2008.

[45] W. B. Mendelson, D. A. Sack, S. P. James, J. V. Martin, R. Wagner, D. Garnett, J. Milton, and T. A. Wehr, "Frequency analysis of the sleep EEG in depression," *Psych. Res.*, vol. 21, no. 2, pp. 89–94, 2007.

[46] A. Wang and A. P. Chandrakasan, "A 180-mV subthreshold FFT processor using a minimum energy design methodology," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 310–319, Jan. 2005.

[47] V. Sze and A. P. Chandrakasan, "A 0.4 V UWB baseband processor," in *Proc. IEEE Int. Symp. Low Power Electron. Design Conf.*, Aug. 2007, pp. 262–267.

[48] I. S. Abu-Khater, A. Bellaouar, and M. I. Elmasry, "Circuit techniques for CMOS low-power high-performance multipliers," *IEEE J. Solid-State Circuits*, vol. 31, no. 10, pp. 1535–1546, Oct. 1996.

[49] S. A. Vitale, P. W. Wyatt, N. Checka, J. Kedzierski, and C. L. Keast, "FD-SOI process technology for subthreshold-operation ultralow-power electronics," *Proc. IEEE*, vol. 98, no. 2, pp. 333–342, Feb. 2010.

Mohammed Shoaib (S'08) received the B.Tech. and M.Tech. degrees in electrical engineering with a specialization in microelectronics and VLSI design from the Indian Institute of Technology (IIT) Madras, Chennai, India, in 2007 and 2008, respectively, and the M.A. degree in electrical engineering from Princeton University, Princeton, NJ, in 2010, where he is currently pursuing the Ph.D. degree in electrical engineering.

His current research interests include data-driven platforms for intelligent biomedical systems, with an emphasis on low-energy circuits and signal processing. He has authored or co-authored more than 20 technical papers, and he holds 4 U.S. patents.

Dr. Shoaib was a recipient of the IIT Madras MCM Scholarship from 2004 to 2007, the Ph.D. Fellowship and an Excellence in Electrical Engineering Award from Princeton University in 2008 and 2011, respectively, the Roberto-Padovani Scholarship in 2011, the Harold W. Dodds (Princeton University Honorific) Fellowship in 2012, and the Princeton University Gordon Wu Prize for Excellence in 2012. He was a co-recipient of the Qualcomm Innovation Ph.D. Fellowship in 2011, a nominee for the Best Paper Award at Healthcomm in 2011. He is a fellow of the McGraw Center for Teaching and Learning and has had industrial experience through internships with research groups at IBM Zurich Research Laboratory, Ricoh California Research Center, and Qualcomm Corporate Research and Development, CA.

Niraj K. Jha (S'85–M'85–SM'93–F'98) received the B.Tech. degree in electronics and electrical communication engineering from the Indian Institute of Technology Kharagpur, Kharagpur, India, the M.S. degree in electrical engineering from the State University of New York at Stony Brook, Stony Brook, and the Ph.D. degree in electrical engineering from the University of Illinois, Urbana, in 1981, 1982, and 1985, respectively.

He is currently a Professor of electrical engineering with Princeton University, Princeton, NJ. He has authored or co-authored more than 360 technical papers, and authored 12 book chapters. He has co-authored or co-edited five books entitled *Testing and Reliable Design of CMOS Circuits* (Kluwer, 1990), *High-Level Power Analysis and Optimization* (Kluwer, 1998), *Testing of Digital Systems* (Cambridge University Press, 2003), *Switching and Finite Automata Theory*, 3rd edn. (Cambridge University Press, 2009), and *Nanoelectronic Circuit Design* (Springer, 2010). He holds 13 U.S. patents. His research interests include FinFETs, low-power hardware and software design, computer-aided design of integrated circuits and systems, secure computing, quantum computing, and energy-efficient buildings.

Dr. Jha is a recipient of the AT&T Foundation Award, the NEC Preceptorship Award for Research Excellence, the NCR Award for Teaching Excellence, and the Princeton University Graduate Mentoring Award. He is a fellow of ACM. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON VLSI SYSTEMS and an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I AND II, the IEEE TRANSACTIONS ON COMPUTER-AIDED-DESIGN, the IEEE TRANSACTIONS ON VLSI SYSTEMS, and the *Journal of Electronic Testing: Theory and Applications*. He is currently an Associate Editor of the IEEE TRANSACTIONS ON COMPUTERS, the *Journal of Low Power Electronics*, and the *Journal of Nanotechnology*. He was the Program Chairman of the Workshop on Fault-Tolerant Parallel and Distributed Systems in 1992, the International Conference on Embedded and Ubiquitous Computing in 2004, and the International Conference on VLSI Design in 2010. He was the Director of the Center for Embedded System-on-a-Chip Design funded by the New Jersey Commission on Science and Technology. He has co-authored 14 papers, which have won various awards, including the Best Paper Award at ICCD'93, FTCS'97, ICVLSID'98, DAC'99, PDCS'02, ICVLSID'03, CODES'06, ICCD'09, and CLOUD'10. A paper of his was selected for "The Best of ICCAD: A collection of the best IEEE International Conference on Computer-Aided Design papers of the past 20 years," two papers by the IEEE *Micro Magazine* as one of the top picks from the Computer Architecture conferences in 2005 and 2007, and two others as being among the most influential papers of the last 10 years at the IEEE Design Automation and Test Conference, in Europe. He has co-authored another six papers that are nominated for Best Paper or Most Influential Paper Awards. He has given several keynote speeches in the area of nanoelectronic design and test.

Naveen Verma (M'05) received the B.A.Sc. degree in electrical and computer engineering from the University of British Columbia, Vancouver, Canada, in 2003, and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 2005 and 2009, respectively.

He has been an Assistant Professor of electrical engineering with Princeton University, Princeton, NJ, since 2009. His research focuses on ultralow-power integrated circuits, including low-voltage digital logic and SRAMs, low-noise analog instrumentation and data conversion, and energy-efficient processing algorithms especially for biomedical applications.

Prof. Verma was a co-recipient of the ISSCC Jack Kilby Award for Outstanding Student Paper in 2008 and the DAC/ISSCC Student Design Contest Award in 2006.