A Low-Energy Computation Platform for Data-Driven Biomedical Monitoring Algorithms

Mohammed Shoaib, Niraj Jha and Naveen Verma
Department of Electrical Engineering
Princeton University, NJ 08544
{mshoaib,jha,nverma}@princeton.edu

ABSTRACT

A key challenge in closed-loop chronic biomedical systems is the ability to detect complex physiological states from patient signals within a constrained power budget. Data-driven machine-learning techniques are major enablers for the modeling and interpretation of such states. Their computational energy, however, scales with the complexity of the required models. In this paper, we propose a low-energy, biomedical computation platform optimized through the use of an accelerator for data-driven classification. The accelerator retains selective flexibility through hardware reconfiguration and exploits voltage scaling and parallelism to operate at a sub-threshold minimum-energy point. Using cardiac arrhythmia detection algorithms with patient data from the MIT-BIH database, classification is achieved in 2.96 μJ (at V_{dd} = 0.4 V), over four orders of magnitude smaller than that on a low-power general-purpose processor. The energy of feature extraction is 148 μJ while retaining flexibility for a range of possible biomarkers.

Categories and Subject Descriptors

C.3 Real-time and embedded systems

General Terms

Electrocardiograph (ECG), support vector machine (SVM)

1. INTRODUCTION

Thanks to emerging sensors and stimulators as well as specialized networking technologies, biomedical devices are advancing to new frontiers. Deep-brain stimulators [2], for instance, offer unprecedented modalities for delivering therapy to patients affected by neurological conditions, ranging from Parkinson's disease to epilepsy; neural prosthesis (i.e., brain-machine interface) [3, 10] is beginning to show promise in restoring motor functions in disabled patients; out-patient monitoring networks raise the possibility of comprehensive yet cost-scalable healthcare delivery over large populations with increasingly diverse disease states [5].

The central need, as these systems advance towards *intelligent, closed-loop* operation [3], is the ability to detect specific physiological states of interest from signals that are available through chronic sensing. 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 often variable from patient-to-patient [14]. Data-driven modeling is emerging as a powerful approach to overcome these challenges [6]. This has been prompted by the recent large-scale availability of physiological data in the healthcare domain as well as the development of machine learning techniques for modeling specific correlations in the data and efficiently applying these models [11]. 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). Chronic patient monitoring devices have recently attempted to exploit data-driven techniques, but have thus far been limited in their ability to incorporate the complete computation [3, 13, 15, 21].

This paper proposes a general-purpose platform for biomedical detection of physiological states that takes advantage of the computational structure and characteristics of machine-learning-based data-driven patient monitoring algorithms. The specific contributions are as follows:

- 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 [12] and shows that classification, the complexity of which depends on the characteristics of the data, poses the primary energy limitation.
- Based on the energy analysis and the computational requirements for various parts of the algorithm, we propose a general-purpose architecture for a biomedical computation platform. This attempts to employ programmability where computational flexibility is required, while leveraging a hardware accelerator for classification, where set computations are required at a very high energy efficiency.
- We propose a transistor-level design of the classification accelerator that leverages an ultra-low-power technology (i.e., low-leakage FD-SOI). Specific requirements for computational flexibility are identified and incorporated through hardware reconfigurability in a parallelized subthreshold implementation that aims to operate at the minimum energy supply voltage.

2. APPLICATION ENERGY ANALYSIS

Cardiac arrhythmia refers to abnormal heart beats that are indicative of a range of cardiovascular conditions. In this section, arrhythmia detection serves as a representative example, demonstrating the key energy limitation in typical biomedical detection applications, namely, that *classification* poses the energy bottleneck due to the

*Acknowledgments: This work was supported by the Gigascale Systems Research Center, one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation entity.

DAC '11, Jun 05-10 2011, San Diego, CA, USA

complexity of the models required. Unfortunately, the model complexity depends on the characteristics of the application data and, thus, introduces an unavoidable trade-off with respect to accuracy.

Fig. 1 illustrates the structure of typical machine-learning-based data-driven detection algorithms. The two primary steps involved in these are: (1) biomarker extraction, and (2) biomarker interpretation (through a classifier). Biomarkers refer to specific signal parameters that have some correlation with the physiological state of interest. For arrhythmia detection, a range of biomarkers has been used (including ECG morphology, beat intervals, spectral features, etc. [4]). The range of biomarkers originates due to the diverse clinical trade-offs introduced by each, which can also be variable across patients [7]. In this study, two prominent biomarkers, including waveform morphology (as in [4]) and spectral wavelets (as in [20]), are used. Thus, the associated processing steps, including segmentation (to isolate individual beats), are implemented in software (enabling the energy analysis presented next), and their outputs form the feature vectors that are used for classification.

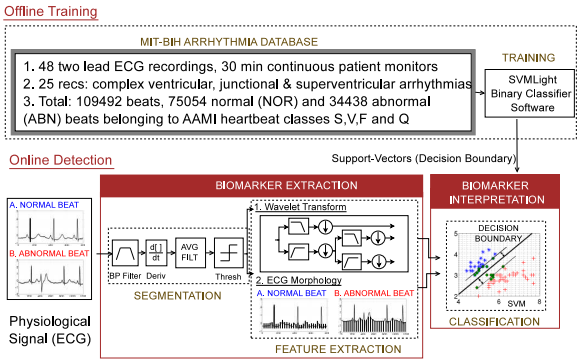

Figure 1: The structure of data-driven biomedical detection algorithms includes (1) offline training, (2) online detection (employing biomarker extraction & biomarker interpretation).

For data-driven classification, an SVM is used. SVMs are popular machine learning classifiers that can be efficiently trained offline to derive the support vectors, which are a set of vectors used to construct a decision boundary. Although training can be done offline, classification, through the application of the support-vector model, must be performed in real-time for chronic detection. To accurately represent the computational complexity imposed by the support-vector model, patient data from the MIT-BIH database [12] is used.

2.1 Energy Profiling

The detection algorithms illustrated in Fig. 1 have been implemented using SVM-Light [8] for the SVM, and the computational energy has been profiled via instruction set simulation of a Tensilica DC_108Mini Xtensa processor with a minimum-base configuration [19]. The energy results for feature extraction are shown in Table 1. A feature vector is derived every heart beat and consumes 94.10 μJ for segmentation, 3.22 μJ for morphological feature extraction, and 53.41 μJ for wavelet feature extraction.

Table 1: Energy per test vector for preprocessing and feature extraction on the Tensilica Xtensa core DC_108Mini.

Computational step	Energy/test vector
Pre-processing *segmentation*	94.10 μJ
Morphology *feature extraction*	3.22 μJ
Wavelet *feature extraction*	53.41 μJ

Figs. 2 and 3 show the energy of classification versus the number of support vectors (N_{SV}) and the number of dimensions (D_{SV}) in each support vector, respectively. Both the number and dimensionality of the support vectors are representative of the model complexity. It can be seen that with energy scaling, the classification energy rapidly dominates that of feature extraction. The actual classification computation is shown below (for radial-basis function (RBF) and polynomial transformation kernels):

$$\text{DATA CLASS} = sgn\left[\sum_{i=1}^{N_{SV}} K(\vec{\mathbf{x}} \cdot \vec{\mathbf{sv}}_\mathbf{i})\,\alpha_i y_i - b\right] \quad (1)$$

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

Here, $sgn[]$ is the *signum function*, $\vec{\mathbf{x}}$ is the feature vector to be classified, and $\vec{\mathbf{sv}}_\mathbf{i}$ is the i^{th} support vector (b, d, α_i, β, γ, and y_i are training parameters). The energy-dominating computation in Eq. (1) is the dot product of $\vec{\mathbf{x}}$ and $\vec{\mathbf{sv}}_\mathbf{i}$. The energy, thus, scales with both D_{SV} and, due to the summation, N_{SV}. K represents a kernel function. If K were a linear operation, test vector $\vec{\mathbf{x}}$ could be pulled out of the summation, annulling the energy scaling [9]. Unfortunately, however, linear classification kernels have been shown to be insufficient for biomedical datasets [17]. This raises the need for non-linear transformation kernels, which afford much higher flexibility in the classification decision boundary. As mentioned previously, the complexity of the support vector model required is dependent upon the application data. Although N_{SV} can be reduced,

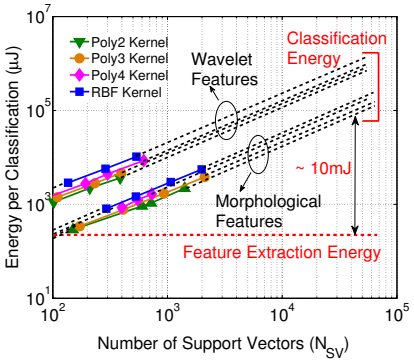

Figure 2: Classification energy scales with N_{SV} and thus dominates that of feature extraction.

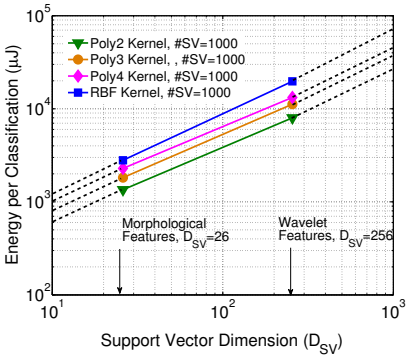

Figure 3: Classification energy scales with D_{SV} (both N_{SV} and D_{SV} are representative of classification complexity).

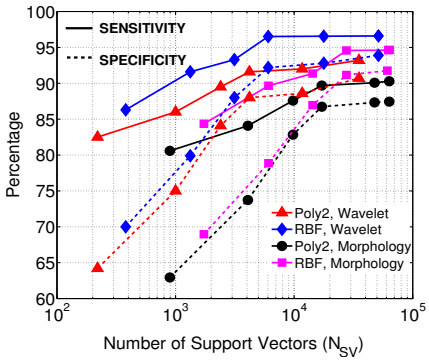

Figure 4: Easing model complexity (by reducing N_{SV}) degrades the accuracy of classification.

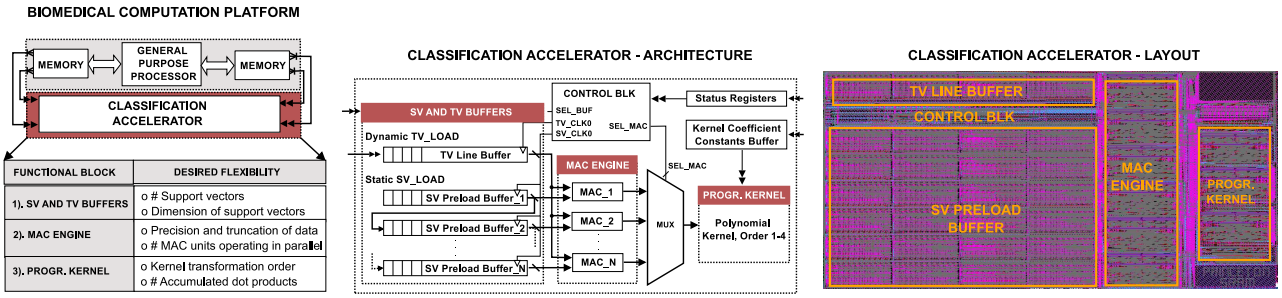

Figure 5: Architecture of the low-energy biomedical computation platform. Flexibility in the biomarkers is retained through a general-purpose processor and energy efficiency is achieved through a classification accelerator.

this comes at the cost of detection accuracy. Fig. 4 shows how the sensitivity and specificity for arrhythmia detection degrade as N_{SV} is reduced[1]. To avoid compromising accuracy, the datasets in biomedical applications often require complex models, causing the classification energy to be dominant (by over two orders of magnitude in the case of arrhythmia detection, as shown in Fig. 2). The energies reported in other biomedical applications are consistent with this result (e.g., seizure detection based on electroencephalograph classification [21]).

3. LOW-ENERGY BIOMEDICAL COMPUTATION ARCHITECTURE

The energy limitation posed by classification is an important characteristic of the applications, and it motivates a platform architecture for programmable low-energy biomedical devices. Although they dominate energy, the classification computations remain canonical across these applications. Feature extraction, on the other hand, has modest energy needs but requires a high degree of computational flexibility due to the range of clinical trade-offs associated with the choice of features (i.e., biomarkers) [4, 16]. The platform architecture proposed in Fig. 5 aims to take advantage of this computational structure in the algorithms by employing a general-purpose processor for feature extraction and an optimized accelerator for kernel-based SVM classification.

To optimize the efficiency of the classification accelerator, several approaches are pursued. Most importantly, minimum-energy operation [23] is achieved through voltage scaling and parallelism, whereby the throughput constraint for real-time detection is met. In addition, an ultra-low-leakage technology (150nm FD-SOI) is employed. In addition to energy efficiency, however, the need for *selective flexibility* is also recognized so that the classification needs across a wide range of biomedical applications can be supported. The accelerator attempts to incorporate these needs through hardware reconfigurability. This yields flexibility in the classification model, computation precision, and kernel transformation function (as summarized in Fig. 5).

3.1 Accelerator Microarchitecture

The accelerator has three major functional blocks (shown in Fig. 5): (1) the support vector (SV) and test vector (TV) buffers, (2) the multiply-accumulate (MAC) engine, and (3) the programmable polynomial kernel core. Following off-line training, SVs are loaded into the SV preload buffers. Since this is a one-time process, preload buffers are connected as a shift register to simplify load control signals. 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 Eq. (1). Readout from these buffers is optimized using a multiplexer array decoder (as described in Sec. 4.1). 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. Once multiplication over all the dimensions is complete, the dot products are multiplexed to the kernel transformation block, where a 2nd, 3rd, or 4th order polynomial transformation is computed. The results are scaled and summed by a final accumulator whose output sign determines the classification result. Sec. 4.1 provides details of how reconfigurability is implemented for feature dimensionality, number of support vectors, computation precision, and kernel transformation.

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

3.2 Energy Minimization

Since the dot-product derivation (in the MAC array) dominates the computation, we optimize its energy. The total energy is determined primarily by the sum of active-switching (E_{act}) and subthreshold leakage (E_{lk}) sources:

$$E_{total} = E_{act} + E_{lk} = C_{eff} V_{dd}^2 + I_{leak} V_{dd} T_{MAC} \quad (2)$$

where C_{eff} is the effective switched capacitance of a MAC unit, V_{dd} is its supply voltage, I_{leak} is its leakage current, and T_{MAC} is its circuit delay.

Choice of technology. Due to the modest performance requirements in typical biomedical applications (i.e., due 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/second is required. The technology we use is, thus, a 1.5V 150nm ultra low-leakage FD-SOI CMOS process. FD-SOI results in steep subthreshold slopes [22], and the devices have high threshold voltages (i.e., $V_{t,N} = 0.65V$, $|V_{t,P}| = 0.53V$).

Voltage scaling and parallelism. In Eq. (2), the reduction in E_{act} due to V_{dd} scaling is opposed by the increase in leakage energy (due to longer resulting leakage-current integration time T_{MAC}). The energy-optimal point, thus, typically occurs in the subthreshold region, since here the circuit delay begins to degrade rapidly [23]. Although this implies that energy optimization leads to low circuit performance, computational throughput constraints can be efficiently met if the circuit units can be operated in parallel with minimal overhead [18]. 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

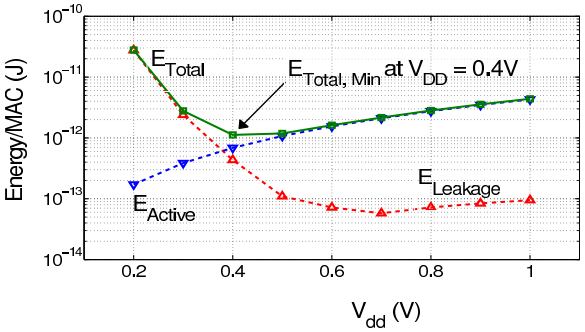

Figure 6: The active-switching and leakage energy profiles for a MAC unit with the min. total energy occurring at $V_{dd} = 0.4\ V$.

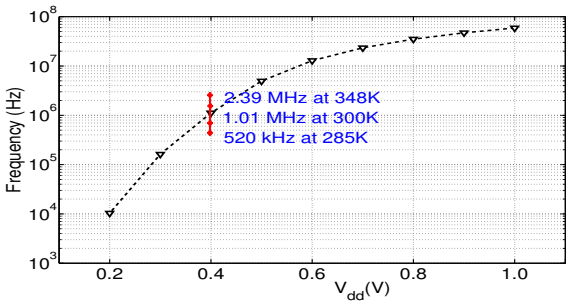

Figure 7: The operating frequency at $V_{dd} = 0.4\ V$ is 520.0 kHz at 285K (low temperature is slowest in subthreshold).

($R_{T.MAC}$) required in the classifier computation (of Eq. (1)) is given by

$$R_{T.MAC} = \lceil N_{SV} \times D_{SV} \times R_{CLASS} \rceil \quad (3)$$

where R_{CLASS} is the classification rate. The required parallelism is then $R_{T.MAC} \times T_{MAC}$.

For the application considered, wavelet-based arrhythmia detection results in the severest throughput constraint since the feature dimensionality (D_{SV}) is 256 (compared to 26 for morphology features). The N_{SV} required is between 3,500 and 10,000 for reasonable accuracy (based on Fig. 4), and beats must be classified at the rate of up to 3 beats per second (i.e., 180 beats per minute). Thus, the $R_{T.MAC}$ required ranges from 2.7M-7.7M MACs/second. Fig. 6 shows the active-switching and leakage energies of a MAC unit (based on a transistor-level simulation). The total energy is minimized at a V_{dd} of 0.4V, which is in the subthreshold region for the technology employed. Fig. 7 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) and the minimum-energy V_{dd}, the maximum frequency is 520 kHz (i.e., $T_{MAC} = 1.92\ \mu s$). The level of parallelism required is thus 6 to 15 MAC units. Fig. 6, 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. 7), causing a negligible increase in total energy (based on Fig. 6). We thus optimize for the lower performance (by employing 6 MAC units), and use voltage scaling with minimal impact on the optimization to elevate the performance when required.

4. CIRCUIT DESIGN AND OPTIMIZATION

In this section, we describe how the selective flexibility desired in the classification accelerator is achieved using reconfigurable circuit blocks. The complete classifier has been laid out, and simulation results are from post-layout extraction.

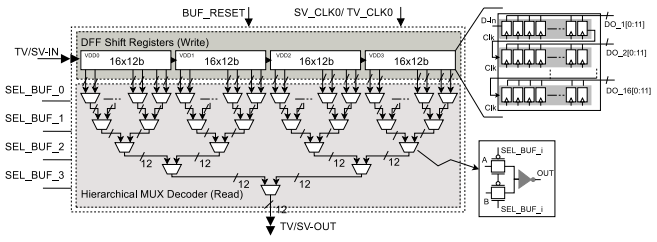

Figure 8: The buffers are implemented by clocked register-file cells forming a shift register for writes and employing hierarchical multiplexers for read.

4.1 SV and TV Buffers

The energy of the buffers is optimized for read operations since the SVs are loaded infrequently (i.e., only when changes to the classification model are required). Fig. 8 shows the buffer architecture in detail. Each buffer consists of an array of edge-triggered register-file cells connected into a shift register. Writes are, thus, performed using clocks SV_CLK0 and TV_CLK0; these are generated and appropriately gated by the control block (in Fig. 5), which enforces the write phase. For readout, hierarchical multiplexers are used to avoid long bit lines, since these result in higher active-switching energy and worse robustness to bit line leakage in the deep subthreshold regime [23]. The multiplexers are daisy-chained and arrayed for a compact layout.

The multiplexers are controlled by a counter in the control block that saturates at a value of $D_{SV} \times N_{SV}$, allowing programmability in both D_{SV} and N_{SV} (via the status register). The voltage of the 16×12b bank in Fig. 8 can be independently gated to scale the leakage energy for the total buffer size needed. The accelerator 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 8,192 write sequences from the processor cache or from off-chip memory to the local buffers. Thus, a maximum of $4,095$ support vectors and 256 feature dimensions are supported, along with any other combination that results in the same product.

4.2 Variable-precision MAC

Due to the wide range of support vectors 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., [23]). The approach used here aims to exploit the efficiency of the Booth encoding algorithm [1]. Suppose $x \times y$ must be computed, where the operand bit-widths are $[n-1:0]$ and $[2m-1:0]$, respectively. Using radix-4 Booth encoding multiplication, m partial products are required rather than $2m$, as in a conventional shift-and-add multiplier. Consider an integer encoding:

$$\delta_i(y) = y[2i-1] + y[2i] - 2y[2i+1] \quad (4)$$

implying that $-2 \le \delta_i \le 2$ for $y[j] = 0,\ \forall j < 0$ and $\forall y, i \in N$. Also, by definition:

$$x \times y = x \sum_{i=0}^{m-1} 2^{2i}\delta_i(y) = \sum_{i=0}^{m-1} 2^{2i} x\delta_i(y). \quad (5)$$

Thus, there are m partial products involved in Eq. (5). Moreover each of these products is a shifted version of x scaled by one of

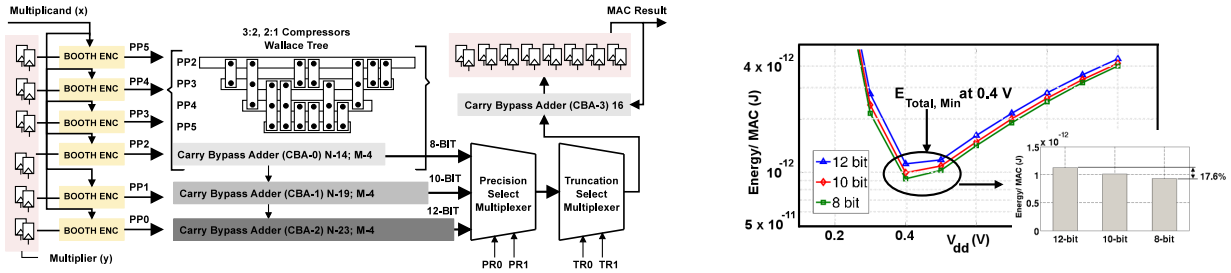

Figure 9: The variable-precision MAC unit. The partial product additions can be terminated at CBA-0/1/2 to scale the precision for 8/10/12 bit inputs. The figure also shows energy savings of 17.6% while scaling the precision from 12 to 8 bits.

the five possible values of δ_i, which can be implemented efficiently using a look-up table.

For our precision-scalable implementation, consider the specialized case of the Booth structure where $y[0] = y[1] = 0$. The integer encoding $\delta_0 = 0$ and the first partial product is, thus, zero. Furthermore, if $x[n-1] = x[n-2] = 0$, Eq. (5) represents the product $\hat{x} \times \hat{y}$ where $\hat{x}$ and $\hat{y}$ are lower-precision numbers given by $\hat{x} = x[n-3:0]$ and $\hat{y} = y[2m-1:2]$. The summation in Eq. (5) now has $m-1$ partial products (running from $i = 1$ to $i = m-1$). Hence, $\hat{x} \times \hat{y}$ represents the multiplication of two numbers whose precision has been reduced by two bits. In general, setting $y[i] = x[n-j] = 0$ for $i \in [0, k-1]$, $j \in [1, k]$ produces $(m - k/2)$ partial products, which can be viewed as reducing the precision of x and y by k bits. The multiplication, thus, requires fewer partial product additions, which, when derived by an intermediate result of the adder hardware, allows adaptive energy scaling with precision.

Fig. 9 shows the architecture of the variable-precision MAC unit. The *BOOTH ENC* blocks compute the $x\delta_i$ functions of Eq. (5) based on the select bits of multiplier y, as described above. The shifted partial products are output as PP_i, $i \in [0, 5]$. This allows a maximum precision of 12 bits for the input operands (corresponding to 6 partial products). In the figure, PP_0 and PP_1 are the partial products used when precisions of 12 and 10 bits, respectively, are required; otherwise the precision is 8 bits. 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-select multiplexer (for 8/10/12-bit precision, respectively). The unused bypass adders are power-gated. The inset in Fig. 9 shows how the energy scales with precision. Although the minimum-energy V_{dd} remains the same, scaling the precision from 12 to 8 bits 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 bits, programmable via the status register). The choice of truncation can be determined based on the number of dimensions in the support vectors; higher truncation allows accumulation of more multiplication results, enabling dot-product computations with higher feature dimensionality. The output of the truncation-selection multiplexer is accumulated into an output register using a 16-bit final CBA.

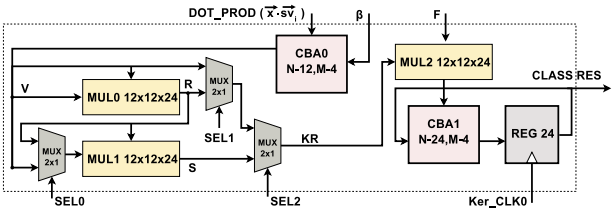

Figure 10: The programmable kernel allows selection between polynomial kernels of order one through four.

4.3 Programmable Polynomial Kernel

Nonlinear kernels have been shown to be important in order to accurately model biomedical datasets. Polynomial transformations are widely-used SVM kernels and are incorporated in our accelerator. The polynomial transformation kernel is defined as

$$K(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i}) = F(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i} + \beta)^d, \quad (6)$$

where d represents the order of the polynomial transformation (β and F are offset and normalization scaling factors, respectively). Fig. 10 shows the reconfigurable transformation kernel used for d equaling one to four. *DOT_PROD* is the input to the transformation kernel which is the dot-product output $(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i})$ of MAC_i, $i \in [0, 5]$ (as shown in Fig. 5). The SEL_MAC signal from the control block (as shown in Fig. 5) sequentially chooses from the MAC outputs. $CBA0$ is a 12-bit adder that computes $(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i} + \beta)$. Multiplier (MUL0 12×12×24) computes $(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i} + \beta)^2$. Multiplier (MUL1 12×12×24) computes either $(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i} + \beta)^3$ or $(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i} + \beta)^4$ based on the value of $SEL0$ (which is programmable via the status register). Select bits, $SEL1$ and $SEL2$, are used to choose between the multiplier outputs (V, R or S) to select linear, quadratic, cubic, or quartic transformations. This result is then scaled by normalizing factor F using multiplier (MUL2 12×12×24), giving the result $K(\vec{\mathbf{x}} \cdot \mathbf{s}\vec{\mathbf{v}}_\mathbf{i})$ of Eq. (6). This is then accumulated over all the support vectors to yield the classifier output.

5. SIMULATION RESULTS

The classification accelerator was designed and laid out in the 150-nm FD-SOI CMOS process. Table 2 shows the energy measurements from post-layout simulation. The polynomial kernel of order 2 was used. The measurements are performed at different supply voltages. Extrapolating for 10,000 support vectors, at a V_{dd} of 400 mV, the total energy for arrhythmia classification per test vector using the wavelet transform features ($D_{SV} = 256$) is 2.96 μJ and for the morphology features ($D_{SV} = 26$) is 0.34 μJ. This is over four orders of magnitude lower than that obtained on the Tensilica processor (Sec. 2.1). The Tensilica processor consumes 102.2 mJ and 9.01 mJ for classification of the wavelet and morphology features, respectively. As seen from Table 2, the classification energy scales roughly linearly with N_{SV} and D_{SV} (this can be seen clearly in the numbers provided for $V_{dd} = 0.6\ V$, $0.4\ V$). The bottom of the table also shows the energy for various classifier kernels. Since the kernel transform is not the dominant computation, the energy scaling is modest.

Table 3 shows the computational energy contributions for the accelerator sub-blocks during online classification. 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,

Table 2: Accelerator energy measurements per test vector in a 150 nm FD-SOI process with $V_{t,N} = 0.65V$ and $|V_{t,P}| = 0.53V$.

V_{dd} (V)	D_{SV}	N_{SV}	E_{act} (pJ)	E_{lk} (pJ)	E_{tot} (pJ)	f_{op}, T=287K
1.0	4	5	1223.2	23.3	1246.5	10.0 MHz
	8	**10**	**2289.9**	**41.9**	**2331.8**	
	16	15	3362.5	93.2	3455.7	
0.6	**8**	**10**	**667.2**	**25.1**	**692.3**	2.0 MHz
		25	1604.4	96.0	1700.4	
		50	3170.4	175.0	3345.4	
0.4	4		276.2	100.0	376.6	550 kHz
	8	**10**	**457.1**	**199.2**	**656.3**	
	16		818.2	336.3	1154.5	
V_{dd} (V)	D_{SV}	N_{SV}	E_{act} (pJ)	E_{lk} (pJ)	E_{tot} (pJ)	Kernel Order
1.0	8	10	2294.9	41.9	2336.8	Poly2
			2377.1	42.3	2419.4	Poly3
			2459.0	42.7	2500.8	Poly4

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 sub-blocks within a MAC unit. As shown, the MAC+KER energy dominates (~ 84%), confirming the benefit of its energy optimization (Sec. 3.2). Although the buffers dominate the transistor count, their low energy contribution shown in the table is due to the low leakage afforded by the choice of the technology.

Table 3: Energy and area (per test vector) of the sub-blocks.

Meas. Condition			Total	BUF	MAC, pJ	KER, pJ	CNTRL
V_{dd}	D	N	pJ	pJ	(% Total)	(% Total)	pJ
1 V	8	10	2336.8	30.4	1224 (52.8)	736.6 (31.5)	345.9
0.6 V			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
Area (in mm^2)			**2.90**	**1.66**	**0.63**	**0.67**	**0.02**

6. CONCLUSIONS

Machine-learning-based algorithms for biomedical detection are emerging as a highly promising means to extract clinically relevant correlations from physiologically-complex signals. The structure in these algorithms can be exploited towards the design of a low-energy platform. Kernel-based classification is found to pose the primary energy bottleneck and is, thus, targeted for optimization through the use of a hardware accelerator. The fixed kernel computations required are exploited, but selective flexibility required across a range of applications is also incorporated through specific hardware reconfigurability. The optimized accelerator reduces the computational energy by over four orders of magnitude compared to a software implementation in a generic low-power processor.

7. REFERENCES

[1] I. S. Abu-Khater, A. Bellaouar, and M. I. Elmasry. Circuit techniques for CMOS low-power high-performance multipliers. *IEEE J. Solid-State Circuits*, 31(10):1535–1546, Oct. 1996.

[2] A. L. Benabid. Deep brain stimulation for Parkinson's disease. *Current Op. in Neurobiology*, 13:696–706, Dec. 03.

[3] 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*, pages 4–7, Jun. 2009.

[4] P. de Chazal, M. O'Dwyer, and R. B. Reilly. Automatic classification of heartbeats using ECG morphology and heartbeat interval features. *IEEE Trans. Biomedical Engineering*, 51(7):1196–1206, Jul. 2004.

[5] E. Dishman. Inventing wellness systems for aging in place. *IEEE Computer*, 37(5):34–41, 2004.

[6] D. Hau and E. Coiera. Learning qualitative models from physiological signals. In *Proc. AAAI Symp. Artificial Intelligence in Medicine*, pages 67–71, 1994.

[7] A. S. Jaffe, L. Babuin, and F. S. Apple. Biomarkers in acute cardiac disease: The present and the future. *J. American College of Cardiology*, 48:1–11, 2006.

[8] T. Jaochims. SVM-Light, support vector machine. http://svmlight/jaochims.org.

[9] F. M. Khan, M. G. Arnold, and W. M. Pottenger. Hardware-based support vector machine classification in logarithmic number systems. In *Proc. IEEE Int. Symp. Circuits and Systems*, pages 23–26, May 2005.

[10] M. A. Lebedev and M. A. L. Nicolelis. Brain-machine interfaces: Past, present and future. *Elsevier Trends in Neurosciences*, 29(9):536–546, 2006.

[11] G. Meyfroidt, F. Guiza, J. Ramon, and M. Bruynooghe. Machine learning techniques to examine large patient databases. *Best Practice & Research Clinical Anaesthesiology*, 23(1):127–143, Mar. 2009.

[12] Physionet. MIT-BIH Physionet database. http://www.physionet.org/physiobank/database.

[13] S. Cadambi et al. A massively parallel FPGA-based coprocessor for support vector machines. In *Proc. Int. Symp. Field Programmable Custom Computing Machines*, pages 115–122, Apr. 2009.

[14] A. Shoeb, B. Bourgeois, S. T. Treves, S. C. Schachter, and J. Guttag. Impact of patient-specificity on seizure onset detection performance. In *Proc. Int. Conf. IEEE EMBS*, pages 4110–4114, Aug. 2007.

[15] 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. EMBS*, pages 4202–4205, 2005.

[16] A. Shoeb and J. Guttag. Application of machine learning to seizure detection. In *Proc. Conf. Machine Learning*, Jun. 2010.

[17] A. H. Shoeb. *Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment*. Electrical and Medical Engineering, Massachusetts Institute of Technology, Boson, Massachusetts, Sep. 2009.

[18] V. Sze and A. P. Chandrakasan. A 0.4-V UWB baseband processor. In *Proc. IEEE Int. Symp. Low Power Electronics and Design*, pages 262–267, Aug. 2007.

[19] Tensilica Inc. The Xtensa processor. http://www.tensilica.com.

[20] E. D. Ubeyli. ECG beats classification using multiclass support vector machines with error correcting output codes. *DSP*, 17(3):675–684, May 2007.

[21] 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*, pages 62–63, Jun. 2009.

[22] 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*, 98(2):333–342, Feb. 2010.

[23] A. Wang and A. P. Chandrakasan. A 180-mV subthreshold FFT processor using a minimum energy design methodology. *J. Solid-State Circuits*, 40(1):310–319, Jan. 2005.