# A Micropower Programmable DSP Using Approximate Signal Processing Based on Distributed Arithmetic

Rajeevan Amirtharajah, *Member, IEEE,* and Anantha P. Chandrakasan, *Fellow, IEEE*

***Abstract*—A recent trend in low-power design has been the employment of reduced precision processing methods for decreasing arithmetic activity and average power dissipation. Such designs can trade off power and arithmetic precision as system requirements change. This work explores the potential of distributed arithmetic (DA) computation structures for low-power precision-on-demand computation. We present an ultralow-power DSP which uses variable precision arithmetic, low-voltage circuits, and conditional clocks to implement a biomedical detection and classification algorithm using only 560 nW. Low energy consumption enables self-powered operation using ambient mechanical vibrations, converted to electric energy by a MEMS transducer and accompanying power electronics. The MEMS energy scavenging system is estimated to deliver 4.3 to 5.6 $\mu$W of power to the DSP load.**

***Index Terms*—Digital signal processing (DSP), distributed arithmetic, energy scavenging, low power.**

## I. Introduction

A RECENT trend in low-power design has been the employment of reduced precision "approximate processing" methods for reducing arithmetic activity and average chip power dissipation. Such designs treat power and arithmetic precision as system parameters that can be traded off versus each other on an *ad hoc* basis. Ludwig *et al.* [1] have demonstrated an approximate filtering technique which dynamically reduces the filter order based on the input data characteristics. More specifically, the number of taps of a frequency-selective finite-impulse response (FIR) filter is dynamically varied based on the estimated stopband energy of the input signal. The resulting stopband energy of the output signal is always kept under a predefined threshold. This technique results in power savings of a factor of six for speech inputs, and can also be implemented using dedicated programmable processors [2]. Larsson and Nicol [3], [4] have demonstrated an adaptive scheme for dynamically reducing the input amplitude of a Booth-encoded multiplier to the lowest acceptable precision level in an adaptive digital equalizer. Their scheme simply involves an arithmetic shift (multiplication/division by a power of two) of the multiplier input depending on the value of the error at the equalizer output. They report power savings of 20%.

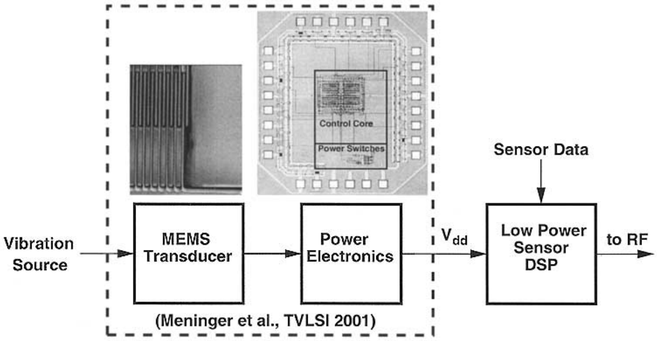

Fig. 1. System block diagram.

This work explores the potential of distributed arithmetic (DA) [5], [6] computation structures for low-power precision-on-demand computation. DA is a method of computing vector inner products without the use of a multiplier. It has a number of applications in fixed-function DSP VLSI implementations [7]–[9]. When used appropriately, it features stochastically monotonic successive approximation properties. In this paper, we present the theory behind DA, its approximate processing properties, its application to a physiological monitoring sensor, and a proof-of-concept VLSI implementation of a micropower DSP incorporating these concepts. The ultralow-power programmable DSP enables systems to be powered by scavenging energy from ambient mechanical vibration [10]. It is the key computational component of the three-chip system shown in Fig. 1, which also incorporates a MEMS transducer that converts vibration to a voltage and a power electronics IC that provides a stable power supply to the sensor DSP load. The MEMS device is a variable capacitor implemented by a floating proof mass etched to form a comb. The proof mass comb is interdigitated with a stationary comb to form the two plates of the variable capacitor. As the mass vibrates, the overlap area of the combs changes, changing the capacitance, and converting mechanical energy to electrical energy stored as voltage on the capacitor. The task of the power electronics is to charge and discharge the MEMS capacitor at appropriate times in the vibration cycle to maximize output power, while consuming as little power as possible. Depending

Manuscript received January 6, 2003; revised September 25, 2003.

This work was supported by the ARL Advanced Sensors Federated Laboratory Program under Contract DAAL01-96-2-0001.

R. Amirtharajan is with the Department of Electrical and Computer Engineering, University of California, Davis, CA 95616 USA (e-mail: ramirtha@ece.ucdavis.edu).

A. P. Chandrakasan is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: anantha@mtl.mit.edu).

Digital Object Identifier 10.1109/JSSC.2003.821774

on the feedback scheme used, we estimate between 4.3 and 5.6 $\mu$W will be available to the DSP load. The transducer and accompanying power electronics are described in detail in [11].

## II. Successive Approximation Using Distributed Arithmetic

DA [5], [6] is a bit-serial operation that computes the inner product of two vectors (one of which is a constant) in parallel. Its main advantage is the efficiency of mechanization and the fact that no multiply operations are necessary. DA has an inherent bit-serial nature, but this disadvantage can be completely hidden if the number of bits in each variable vector coefficient is equal or similar to the number of elements in each vector.

As an example of DA mechanization, let us consider the computation of the following inner (dot) product of $M$-dimensional vectors $\mathbf{a}$ and $\mathbf{x}$, where $\mathbf{a}$ is a constant vector:

$$y = \sum_{k=0}^{M-1} a_k x_k. \tag{1}$$

Let us further assume that each vector element $x_k$ is an $N$-bit two's complement binary number and can be represented as

$$x_k = -b_{k(N-1)} 2^{N-1} + \sum_{n=0}^{N-2} b_{kn} 2^n \tag{2}$$

where $b_{ki} \in \{0, 1\}$ is the $i$th bit of vector element $x_k$. Please note that $b_{k0}$ is the least significant bit (LSB) of $x_k$ and $b_{k(N-1)}$ is the sign bit.

Substituting (2) in (1)

$$y = -\sum_{k=0}^{M-1} a_k b_{k(N-1)} 2^{N-1} + \sum_{n=0}^{N-2} \left[ \sum_{k=0}^{M-1} a_k b_{kn} \right] 2^n. \tag{3}$$

Let us consider the term in brackets:

$$q_n = \sum_{k=0}^{M-1} a_k b_{kn}. \tag{4}$$

Because $b_{kn} \in \{0, 1\}$, $q_n$ has only $2^M$ possible values. Such values can be precomputed and stored in a ROM of size $2^M$. The bit serial input data ($\{b_{0i}, b_{1i}, b_{2i}, \ldots, b_{ki}\}$ for $i = 0, 1, \ldots, N-1$) is used to form the ROM address, and the ROM contents can be placed in an accumulator structure to form the outer sum of (3). Successive scalings with powers of two can be achieved with an arithmetic shifter in the accumulator feedback path. The first term of (3) ($\sum_{k=0}^{M-1} a_k b_{k(N-1)}$) is also stored in the ROM at address $\{b_{0(N-1)}, b_{1(N-1)}, b_{2(N-1)}, \ldots, b_{k(N-1)}\}$. Some extra control circuitry is necessary to ensure that the accumulator subtracts the partial sum to the total result at sign bit time. After $N$ cycles ($N$ is the bitwidth of the $x_k$ vector elements) the final result $y$ has converged to its final value within the accumulator.

Fig. 2 shows a detailed example of a DA computation. The structure shown computes the dot product of a four-element vector $X$ and a constant vector $A$. All 16 possible linear combinations of the constant vector elements ($A_i$) are stored in a ROM. The variable vector X is repackaged to form the ROM address, most significant bit (MSB) first. We have assumed that the $X_i$ elements are 4-bit two's complement numbers. Every clock cycle, the RESULT register adds 2× its previous value (reset to zero) to the current ROM contents. Moreover, each cycle the four registers that hold the four elements of the $X$ vector are shifted to the right. The sign timing pulse $T_s$ is activated when the ROM is addressed by bit 3 of the vector elements (sign). In this case, the adder subtracts the current ROM contents from the accumulator state. After four cycles (bitwidth of the $X_i$ elements), the dot product has been produced within the RESULT register.

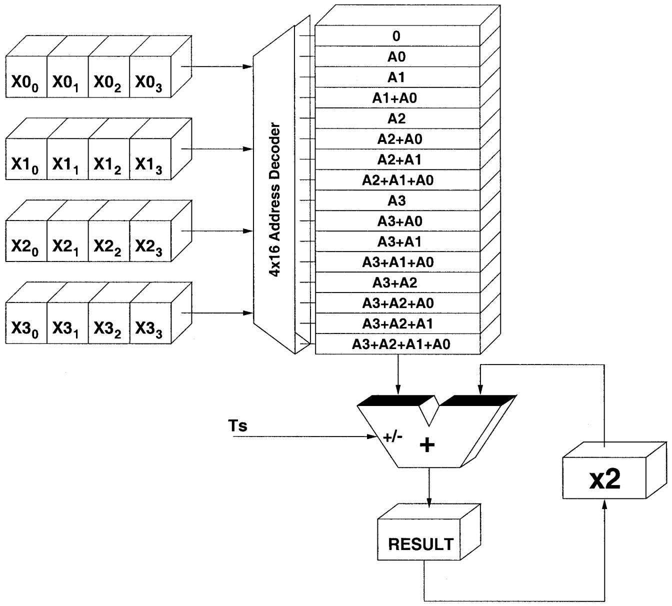

Fig. 2. Distributed arithmetic ROM and accumulator (RAC) structure.

When the DA operation is performed MSB first, it exhibits stochastically monotonic successive approximation properties. In other words, each successive intermediate value is closer to the final value in a probabilistic sense. An analytical derivation follows.

The $i$th intermediate result of an MSB-first DA computation ($i > 0$) is

$$y_i = -q_{(N-1)} + \sum_{n=N-1-i}^{N-2} q_n 2^n \tag{5}$$

where $q_n$ is as defined in (4). Note that when $i = N-1$, (5) yields (3).

Let us define an error term $e_i, i = 0, 1, \ldots, N-1$ as the difference between each intermediate value $y_i$ and the final value $y$:

$$e_i = y - y_i \tag{6}$$
$$= \sum_{n=0}^{N-2-i} q_n 2^n. \tag{7}$$

We model $q_n$ as experimental values of a discrete random variable $\mathbf{q}$. The underlying stochastic experiment is random accesses of the DA coefficient ROM in the presence of random

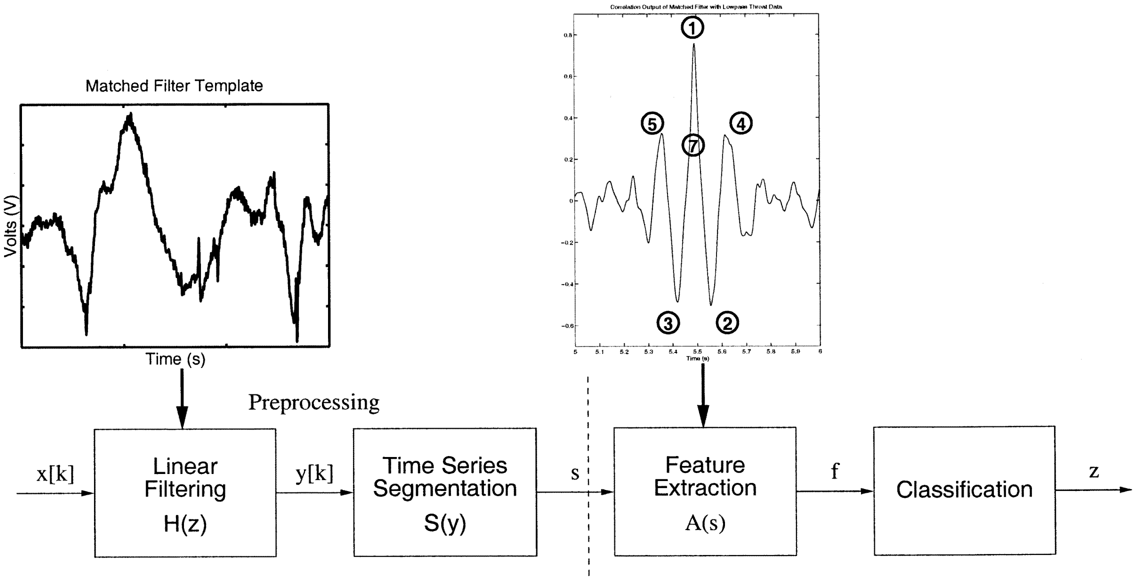

Fig. 3. Heartbeat detection and classification algorithm.

inputs. The experimental values of $\mathbf{q}$ are the DA ROM contents. The first- and second-order statistics of the error term $e_i$ are

$$E[e_i] = E[\mathbf{q}] \sum_{n=0}^{N-2-i} 2^n \tag{8}$$

$$= \frac{2^{N-1-i} - 1}{2} \sum_{k=0}^{M-1} a_k \tag{9}$$

$$\sigma_{e_i}^2 = \sigma_{\mathbf{q}}^2 \left(1 + 4 + \cdots + 2^{2(N-2-i)}\right)$$

$$= \frac{2^{2(N-1-i)} - 1}{3} \sum_{k=0}^{M-1} a_k^2 \sigma_{b_{kn}}^2 \tag{10}$$

$$= \frac{2^{2(N-1-i)} - 1}{12} \sum_{k=0}^{M-1} a_k^2 \tag{11}$$

where (10) and (11) have been computed under the assumption that the LSBs $b_{kn}$ ($i$ large) are independent identically distributed random variables uniformly distributed between 0 and 1 ($E[b_{kn}] = 1/2, \sigma_{b_{kn}}^2 = 1/4$). This is a valid assumption for input DSP data [12], [13]. The fact that (10) and (11) are monotonically decreasing functions of $i$ (RAC cycles) shows the successive approximation property in probabilistic terms of the DA mechanization. It is possible to reduce the approximation error by half by storing $E[e_i]$ in a register and adding it to the intermediate value $y_i$ in the final accumulation cycle.

## III. Detection and Classification Signal Processing for Physiological Monitoring

Recent work has demonstrated power consumption on the order of a few hundred microwatts for wearable [14] and implantable [15] biomedical devices. An application of power scalable processing using DA is a low-power DSP for physiological monitoring that uses a wearable microphone as a biomedical sensor for recording heartbeats, breathing sounds, and voice data. This data will eventually be used to determine the physical condition of the wearer. The first step is detection of the heartbeats, which can be used to determine heart rate as the basis for a physiological assessment.

Evaluation of the spectrogram of the acoustic data indicates that most of the energy from heartbeat sounds lies in the low-frequency range, below 200 Hz. We developed a classifier-based approach to heartbeat detection that takes advantage of this spectral characteristic to improve detection performance in the presence of speech and other high-frequency energy.

The basic algorithm is outlined below.

1) Preprocessing:
   - Low-pass filtering: The data is bandlimited to below 200 Hz to eliminate as much of the voice and breath energy as possible.
   - Matched filtering: The output of the low-pass filter is passed through a matched filter to determine the candidate heartbeat locations in the time domain.
   - Segmentation: The sensor output is divided into overlapping segments at least long enough to contain a full heartbeat in the time domain, but short enough not to contain more than one.
2) Feature Extraction: A subset of seven features is computed from the segmented matched filter output.
3) Classification: Each feature vector is classified into a heartbeat or nonheartbeat using a parametric Gaussian multivariate classifier [16].

The algorithm is summarized in the block diagram of Fig. 3. Assuming that the low-pass filtering occurs before sampling as an antialiasing step, the first computationally significant step is to perform the matched filtering. The matched filter impulse response, or filter template, is a cleaned up version of the acoustic signature of the heartbeat. When convolved with the input data, the filter output has a large correlation peak at the time location of a heartbeat in the input. The template used in the classification algorithm and an example of a correlation peak are also shown

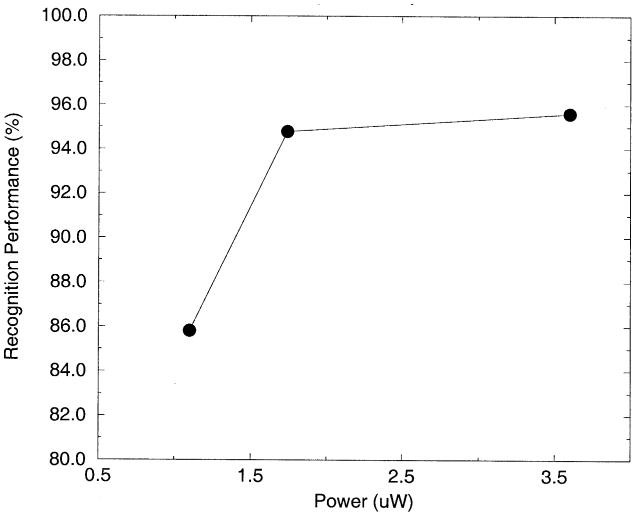

Fig. 4. Recognition performance tradeoffs with DA unit power (simulated).

in Fig. 3. The segmentation phase of the preprocessing localizes the regions of this time series which have the correlation peaks. Features are then extracted from these regions, as labeled in the figure, and classified. The preprocessing steps, in particular the matched filtering, require the most computation. This leads to a low-power DSP architecture that power optimizes these frequently performed computations.

Using DA, a complete dot product can be performed in as many cycles as correspond to the bit widths of the input samples. If the bitwidth $M$ is less than the filter length $N$, this implementation requires fewer clock cycles than a multiply accumulate. This is beneficial for long filters like the matched filter described above, where $N \gg M$. The reduced clock requirement results in low total power not just through frequency reduction, but also through increased voltage reduction since the delay constraint of the DA filter critical path is much less stringent than the multiply–accumulate architecture.

As discussed above, the bit-serial nature of the implementation also allows an alternative approach to approximate processing. By clocking the DA units at less than the full bitwidth, we are in effect reducing the input quantization level. This is roughly equivalent to injecting noise at the input of the filter. In a detection scheme like the heartbeat detection algorithm, this reduced signal-to-noise ratio (SNR) should result in lower performance, i.e., less reliable detection of heartbeat events. However, the reduced performance has also resulted in reduced power since the switched capacitance per filter output sample decreases linearly with the number of input bits clocked in. Fig. 4 shows the classifier performance reduction as the DA unit power is decreased.

## IV. Sensor DSP Architecture

Fig. 5 shows the architecture of the sensor DSP chip, which follows the algorithmic architecture described in Section III. The discrete-time matched filter is implemented using the DA unit. Its output is then passed to a nonlinear filtering unit to calculate quantities used in segmentation. The final segmentation, feature extraction, and classification is performed by the programmable microcontroller at the end to produce the class assignment $z$. The buffer provides a mechanism for synchronization between the front end filtering and the backend processing and is necessary for power reduction. The filtering front end must be running continuously to process the input samples, which arrive at a fixed rate. However, the back end classification only needs to be performed for every segment, not every input sample. The system operates as follows: first, the front end filters the input and writes results to the buffer. A small loop is continuously executed in the microcontroller, checking to see if a full segment has been written to the buffer. The filtering units could do this, but it involves adding circuits that already exist in the ALU of the microcontroller, which is idle while it is trying to detect a segment. To conserve area, we use the microcontroller rather than add complexity to the filter functional units. When a segment is detected, the microcontroller executes the feature extraction and classification code on the data in the buffer that was just written. Architectural simulation of the DSP chip using Verilog shows that 99.8% of the algorithm time is spent executing the matched filtering and other preprocessing functions. These computations therefore dominate both the time and power consumption of the algorithm.

To meet real-time performance demands, the front-end filtering is performed at 1.2 kHz while the back-end processing uses a 250-kHz clock. Many more instructions are executed in the back-end processing than the filtering. Synchronization is handled by multiplexing between a slow clock $\Phi 0$ (1.2 kHz) and a fast clock $\Phi 1$ (250 kHz) on chip. $\Phi 0$ is created by dividing down from $\Phi 1$ using a loadable counter. While using a reduced clock frequency in the preprocessing mode is equivalent to clock gating in terms of decreasing dynamic power consumption, it has the added benefit of relaxing the critical path constraint in the filtering units. When the microcontroller enters classification mode, it disables the front-end filters and requests a higher clock rate ($\Phi 1$). The long program is run at this higher rate so that it can complete before the next input sample appears. When finished, the microcontroller switches back to the slow clock $\Phi 0$ and enables the filtering units.

Fig. 6 shows the implementation of a DA unit. A configurable input shift register made using flip-flops and bypass multiplexers (muxes), which vary the input data bitwidth, addresses an SRAM look-up table of precomputed results. These are accumulated with the appropriate sign. Configuration bits gate the clocks to reduce power. If the entire matched filter were implemented using a single table, the size of the required memory would grow exponentially and would not be realizable. Instead, the filter is composed of 16 individual DA units whose outputs are accumulated into the final filter result. This enables another power performance tradeoff: by configuring how many DA units to use in the computation, we can vary the number of filter taps in addition to the input bitwidth.

Using multiple DA units with fixed look-up table size results in a linear scaling of power consumption with the number of filter taps [6]. Since the clock frequency is fixed by the bitwidth of the input samples, the supply voltage need not scale to accommodate a longer filter. More DA units will need to be implemented, however, resulting in a linear increase in switched capacitance. Fig. 7 shows the power scaling for DA units with

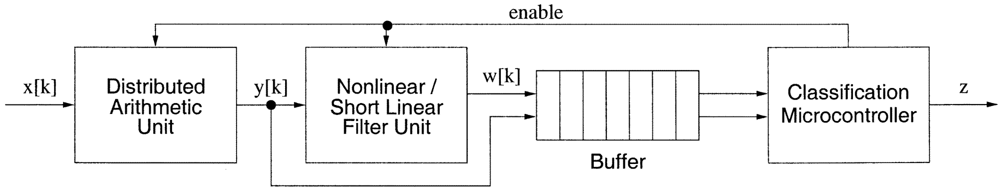
enable
x[k]
Distributed Arithmetic Unit
y[k]
Nonlinear / Short Linear Filter Unit
w[k]
Buffer
Classification Microcontroller
z

Fig. 5. Signal processing chip architecture.

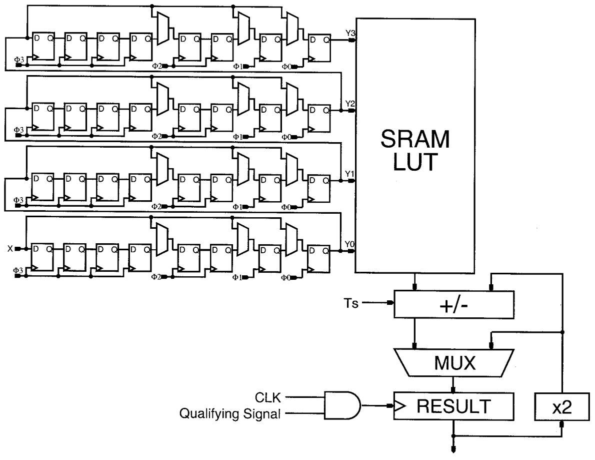

Fig. 6. DA unit with clock gating.

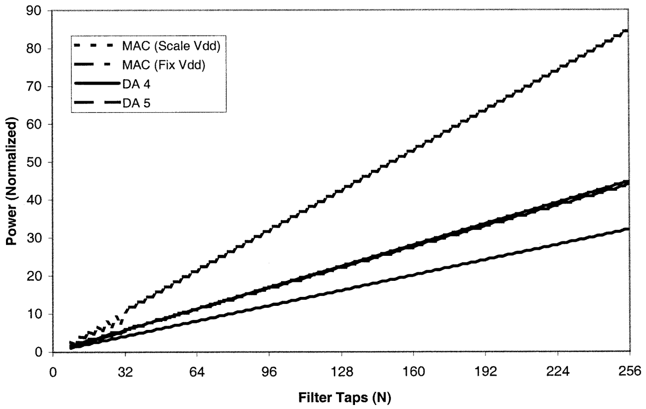

Fig. 7. DA power versus MAC power scaling with filter length.

four and five input look-up tables compared to several multiply–accumulate units (MACs) in parallel, implemented in the same area as the DA units and with the same throughput constraint on the filter output. Since the clock frequency for the MACs scales with the number of filter inputs, two curves are presented. One shows the power if the voltage is fixed while the clock frequency increases, and the other shows the power if the voltage must be scaled as well to meet the delay constraint. The four-input DA unit has consistently lower power than both MAC implementations for the same throughput and area constraints, while the five-input DA unit has roughly the same power as the fixed-voltage MAC implementation. This power savings is achieved because the DA technique implements more computation per area at a fixed clock frequency, while the MAC approach requires faster clocking for the same area to meet the same throughput constraint.

## V. VLSI Implementation

Much of the power reduction in this DSP chip is due to aggressive voltage scaling, selective clock gating, and minimization of switched capacitance. Operation of logic circuits on the edge of subthreshold conduction requires minimizing capacitance to further reduce power consumption. Since much of the die area is devoted to memories (DA unit tables, buffers, and instruction and data memories), reducing their power consumption is of primary importance. Low-voltage operation presents challenges for designing the on-chip SRAM sense amplifiers. Reducing switched capacitance results in a memory architecture that emphasizes small bank size.

### A. Low-Voltage Logic Design

Voltage reduction is the most direct way of decreasing power consumption and also yields the largest benefits because of the quadratic dependence of the dynamic power $fCV^2$. When the supply nears the sum of the absolute value of the device thresholds (in this process, $V_{Tn} = 730$ mV and $V_{Tp} = -810$ mV), the circuit delay increases exponentially. In this region of operation, small changes in voltage result in large changes in delay, but do not affect power consumption dramatically. In this regime it is more important to minimize the other main contributor to dynamic power, namely switched capacitance.

The low throughput requirements of typical sensor applications allow for the use of ripple-carry adders in the critical path of arithmetic units. All adders in the DSP chip use the ripple-carry architecture since the longest ones, 24 full adder cells wide, are still fast enough for the relevant applications even at low power supply voltages. Ripple-carry adders can have power dissipation due to glitching, but for this design the extra power is not significant.

The full adder design of Fig. 8 is a low-area low-switched-capacitance implementation. It only uses 16 transistors in its smallest implementation and is based on using pass-gate XOR gates [17]. The first XOR gate consists of two inverters and transistors *N0-1* and *P0-1*. This computes the propagate signal and its inverse. These signals then go into another XOR gate and a pass-gate multiplexer. Because the power supply voltage is low, it is essential that full pass gates are used to ensure adequate transmission of logic high levels.

Because the outputs of this full adder are not fully driven to the supplies, problems occur as adder cells are cascaded. The delay of the pass-gate chain grows quadratically with the number of cells and at low $V_{dd}$ the delay becomes intolerable even at low clock frequencies. The low throughput requirements imply that pipelining is undesirable for this design, as the required clock loads and switched capacitance overhead would

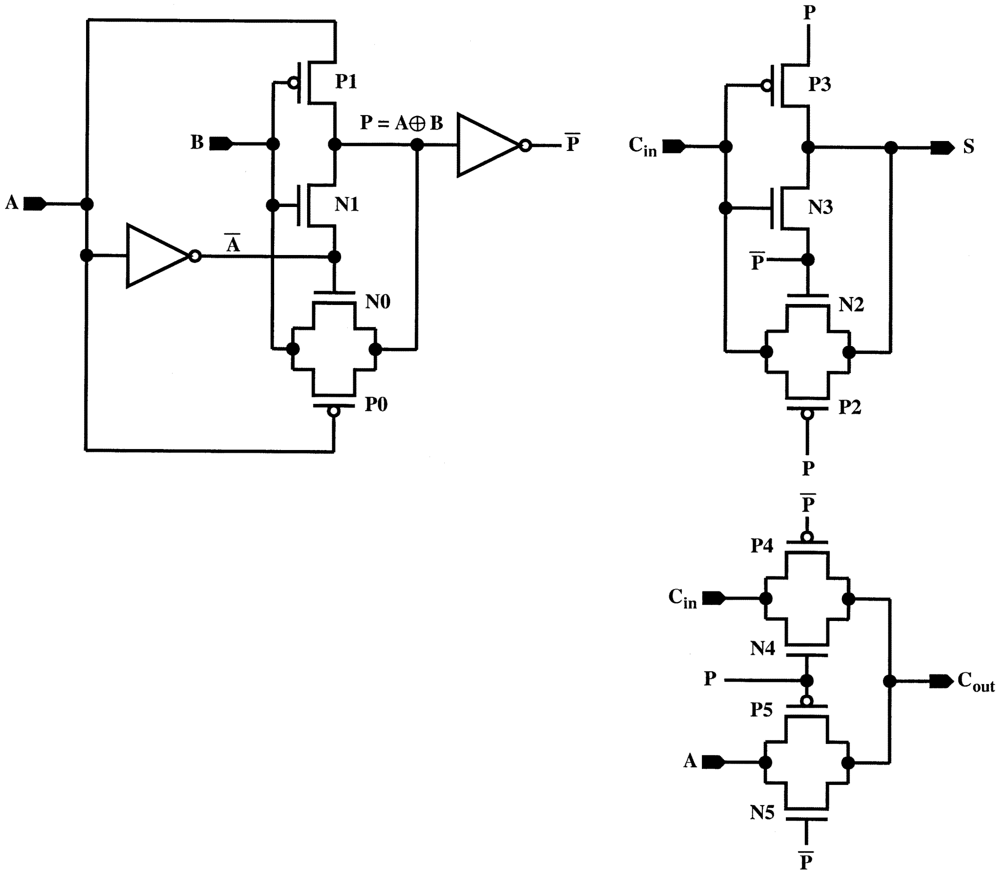

Fig. 8. Full adder cell schematic.

not pay off in terms of voltage reduction. Therefore the entire computation cycle of instruction fetch, decode, operand fetch, execution, and write back must be completed in one clock cycle. To meet the critical path constraints, the outputs along the carry chain are buffered. The number of additional transistors is small since inverting the output of an XOR gate simply requires inverting one of its inputs. In this chip, buffers are placed every four full adder cells.

In addition to adders, the arithmetic units of the chip also require multipliers and one squarer. Simulations show that using a custom squaring circuit consumes about half the power of using a multiplier with both inputs tied together. The multipliers in this chip are 12-bit array multipliers. Other architectures, like Wallace trees, are suboptimal at such low input bitwidth. Buffering is again required every four cells, particularly in the final 24-bit ripple-carry adder. The 12-bit squarer is implemented using a divide-and-conquer architecture [18] based on several smaller array multipliers whose products are combined using adders and some other standard cells. To prevent unnecessary glitching leading to extra power dissipation, the inputs to the array units are latched when the units are unused [19]. This latching is purely for power reduction and therefore its timing is not critical; if data flows through them it only results in increased power dissipation and not incorrect circuit operation. The latches are positive level sensitive static latches.

Fig. 9 is a schematic of a typical edge-triggered flip-flop used in the DSP design. The master stage is a passgate input followed by a $C^2$MOS latch. The slave stage is an inverter with a tristate buffer to implement feedback. This stage is static so that the clock can be gated low and the state preserved. When the clock goes high, the slave stage blocks the input while the master is transparent and the feedback is broken. While the clock is low, the master is transparent and the feedback is enabled as the slave blocks the data. Level sensitive latches are static based on the slave stage shown in the figure.

### B. SRAM Sensing Scheme

Conventional SRAM sensing schemes involve biasing a differential sense amplifier to magnify the voltage difference between the complementary bitlines [17], [20]. This approach has several drawbacks at low voltage. First, the transistors are biased in the subthreshold region and the resulting amplifier is too slow due to the lack of current drive. Second, biasing requires static current flow. This constant power dissipation is unacceptable for microwatt level designs with a significant amount of SRAM.

An alternative is to use a single-ended charge-transfer sensing scheme [21]. A diagram of the circuit is shown in Fig. 10. The SRAM cell is a simple 6-T cell with all minimum-sized devices. For operation near 1 V, the NMOS pass transistors cannot pass a high enough voltage to be read as a logic one. The PMOS devices are also substantially weaker than the NMOS due to their reduced carrier mobility and higher threshold voltage, so a low-to-high transition on the highly capacitive bitline would be exceedingly slow. To eliminate these problems, the bitline (labeled BL in the figure) is precharged high, but through an

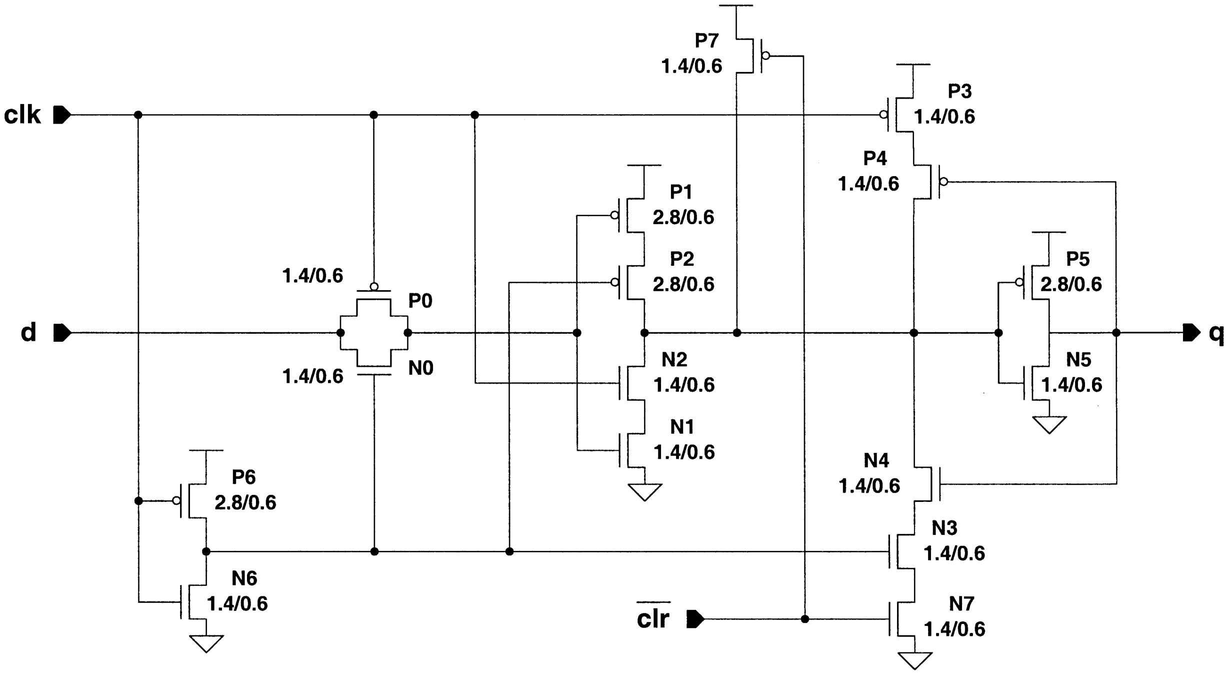

Fig. 9. Positive edge-triggered flip-flop with asynchronous clear schematic.

NMOS device to $V_{\rm dd} - V_{Tn}$ to save power. The SENSE node is also precharged, but directly to $V_{\rm dd}$ and the output OUT is predischarged to ground.

If the cell drives a one, the bitline remains at its precharged value as does the sense node and the output. If the cell drives a zero, then the bitline capacitance gets pulled slightly low. This voltage is close to the threshold of the pass-gate device *N2*. When *N2* turns on, the large bitline capacitance is connected to the relatively small sense node capacitance and charge is shared between them. This results in a dramatic voltage swing on the sense node. Once it crosses the output inverter threshold, the output goes to a valid logic high.

Because the bitline is precharged close to the threshold of the pass-gate device, the scheme is sensitive to noise events on the bitline: a small downward voltage glitch may inadvertently discharge the sense node. This scenario can be averted by ratioing *N2* such that its resistance is higher than the precharge NMOS device *N1*. Consequently, *N2* is a long-channel minimum-width device. This approach results in increased read time, but the low throughput requirement and the load-store architecture of the DSP make such a tradeoff acceptable.

The simulation results of the read operation are presented in Fig. 11. In the first part of the cycle, the bitline and the sense node are precharged to their respective voltages. The output node is driven low. For this example, the cell is storing a zero so when the word line is asserted, the bitline BL drops, the sense node is discharged, and the output is driven to a valid high level.

Because of the difficulty of passing a high voltage through the NMOS pass transistors at low $V_{\rm dd}$, the SRAM cell is written differentially. Some memories also support dual read ports and the sensing scheme described in this section is replicated on both bitlines. The second pass gate of the 6-T cell is not shown in Fig. 10 for clarity of presentation.

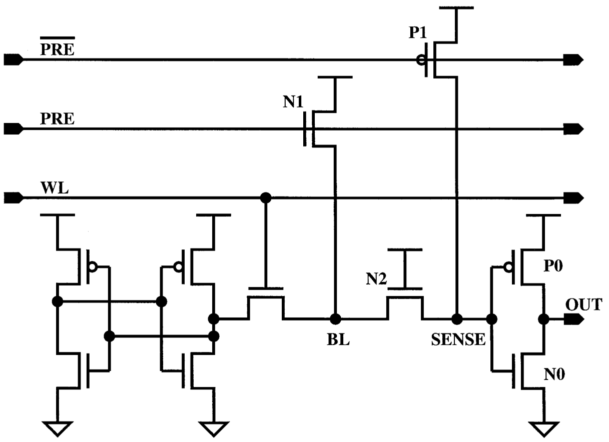

Fig. 10. SRAM sensing circuit.

### C. SRAM Bank Partitioning

A substantial portion of the DSP chip area is devoted to memories, but the memory area is not a limiting factor in the total chip area. Therefore, to avoid having the memory power consumption dominate the chip power, these memories were partitioned into banks optimized for power at the expense of area. High-level power estimation was done using switched capacitance estimates and parameters from the CMOS process [22].

Fig. 12 shows a schematic representation of a single SRAM bank for purposes of power estimation. The problem is how to partition a total number of required memory words among banks to realize the lowest power for the memory. It is important to account for the possible predecoding of addresses as well as the various drivers, precharges, and other circuit operations required for memory reads and writes. Fig. 12 shows the main contributions to power within a bank: predecoded address lines, normal

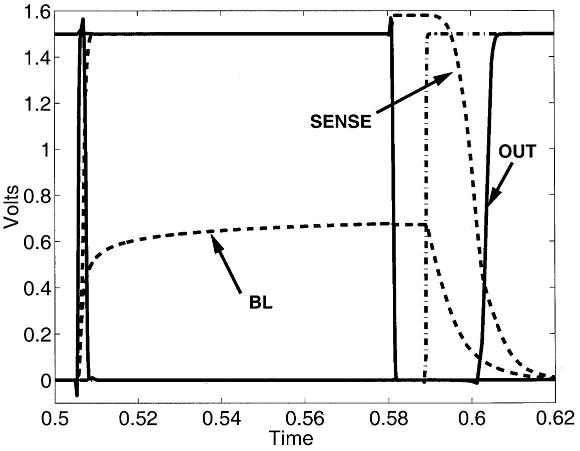

Fig. 11. Simulated SRAM read waveforms.

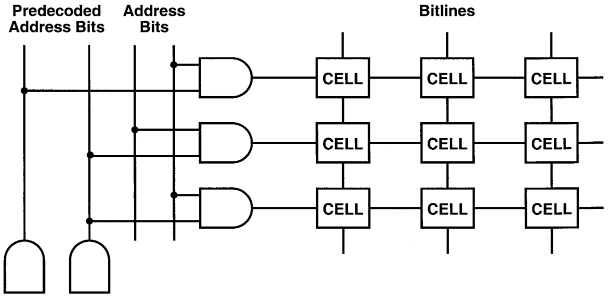

Fig. 12. SRAM bank schematic for high-level power estimation.

address lines, wordlines to access the cells, and bitlines. Since true and complement versions of the normal address lines must be distributed in the address decoder, the probability that an address *line* is charged per address *bit* is $1/2$ assuming the address bits are uniformly distributed. Although this assumption does not strictly hold for all of the bits in highly correlated address traces, this is a worst case assumption since the probability of an energy dissipating transition is lower than one half. The predecoded address lines have different transition probabilities because the AND gates used in the predecoding combine the input bit transition probabilities [23].

Several SRAM subarrays are multiplexed onto a single read bus using tristate buffers. As more subarrays are added, the bitlines become shorter and have less capacitance, but the bus wires get longer and add capacitance. Thus, there is an optimum in the number of subarrays that the total amount of memory is partitioned into. The total power for the memory is

$$P_{\mathrm{TOT}} = P_{\mathrm{mux}} + P_{\mathrm{BL}} + P_{\mathrm{WL}} + P_{\mathrm{AL}} \tag{12}$$

where $P_{\mathrm{TOT}}$ is the total memory bank power, $P_{\mathrm{mux}}$ is the subarray readout bus multiplexer, $P_{\mathrm{BL}}$ is the bitline power, $P_{\mathrm{WL}}$ is the wordline power, and $P_{\mathrm{AL}}$ is the address line power, including the predecoding overhead. Since power consumption in this implementation is dominated by dynamic power dissipation, each of the terms in (12) is proportional to $\alpha C V_{\mathrm{dd}}^2$, where $\alpha$ is the activity factor, $C$ is the capacitance associated with the switching

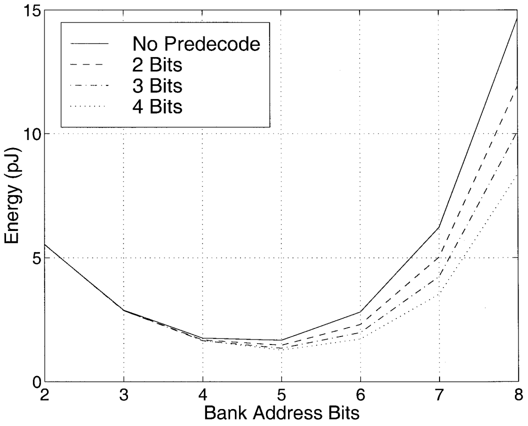

Fig. 13. Optimal size of SRAM banks for various levels of predecoding.

node, and $V_{\mathrm{dd}}$ is the power supply voltage [23]. Substituting expressions for the various terms yields a more detailed equation for the SRAM power:

$$\begin{aligned} P_{\mathrm{TOT}} = {} & \frac{M2^{L-N}}{2}\left(\frac{M}{4}+6\right) C_u V_{\mathrm{dd}}^2 \\ & + \frac{M}{2}\left(\frac{2^N}{2}+2^N\right) C_u V_{\mathrm{dd}}(V_{\mathrm{dd}}-V_{\mathrm{tn}}) \\ & + \frac{N}{2}\left(2^N+\frac{2^N(2+N)}{2}\right) C_u V_{\mathrm{dd}}^2 \end{aligned} \tag{13}$$

where $L$ is the total number of memory address bits, $M$ is the width of memory word, $C_u$ is a unit capacitance representing gate capacitance, drain capacitance, or wire capacitance all normalized to the same standard and scaled by the number of memory cells, $N$ is the number of address bits for each subarray, $V_{\mathrm{dd}}$ is the power supply voltage, and $V_{\mathrm{tn}}$ is the NMOS transistor threshold voltage. The equation assumes that the bitlines are precharged to $V_{\mathrm{dd}} - V_{\mathrm{tn}}$, the activity factor is $1/2$, and each capacitance in terms of $C_u$ is computed using 0.6-$\mu$m logic design rules. Equation (13) assumes no predecoding of the address lines for simplicity, but predecoding is straightforward to incorporate.

Fig. 13 shows a plot of the estimated SRAM power for a fixed memory size and varying levels of bank partitioning and address predecoding. There is a relatively shallow minimum for a subarray size of 32 words. This is small for a high-density memory, but since the memory requirements for the DSP chip are purposefully small, this partitioning can be used without a large area impact. Although many assumptions and simplifications were made in this high level analysis, the shallowness of the minimum indicates that there is not a large penalty for choosing a suboptimal partitioning.

## VI. DSP Test Results

The chip described in this section was fabricated in a 0.6-$\mu$m standard CMOS process with three levels of metallization. The implementation of the matched filter using DA results in the ability to scale the power consumption for this unit with the required performance. Fig. 14 shows the simulated power reduction in the DA unit as fewer bits of the input are shifted into the filter. As expected, the power varies roughly linearly with

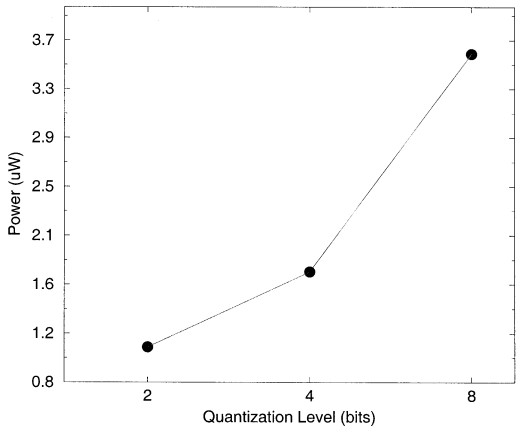

Fig. 14. Power reduction versus input quantization (simulated).

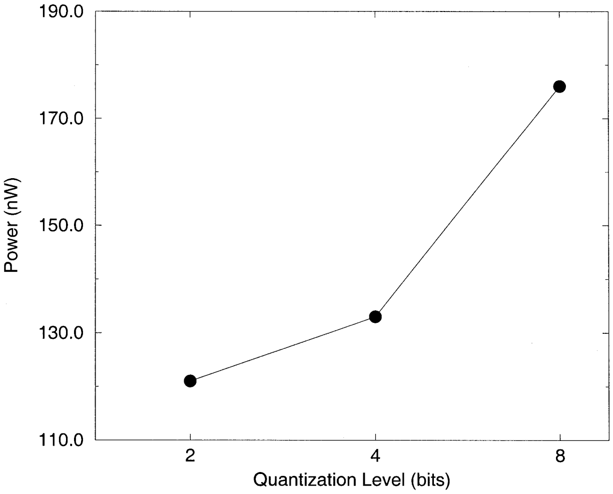

Fig. 15. Measured sensor DSP chip power, DA unit and microcontroller active.

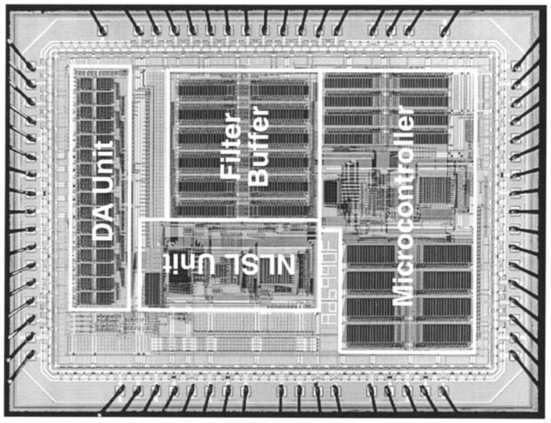

Fig. 16. Sensor DSP chip photograph.

TABLE I
SENSOR DSP CHIP SPECIFICATIONS

| | |
|---|---|
| Area | 4.4 mm x 5.8 mm |
| Frontend Clock Frequency | 1.2 kHz |
| Backend Clock Frequency | 250 kHz |
| $V_{dd}$ | 1.5 V |
| Transistor Count | 190 K |
| Process | 0.6 $\mu$m CMOS |
| $V_{tN}, V_{tP}$ | 0.70 V, -0.90 V |
| Predicted MEMS Power Out | 4.29$\mu$W |
| SensorDSP Chip Power | 560 nW |
| **SensorDSP Chip Energy** | **26.6 pJ per sample** |
| **StrongARM SA-1100 Energy** | **11 $\mu$J per sample** |

the bitwidth, however the lower end of the curve saturates since there is a constant power cost associated with the control logic and clocking of the DA unit. The simulated data reflects the relative scalability of the filter power very accurately. However, the switch-based simulation tool uses lookup tables for device currents which are not accurate in absolute terms at the edge of subthreshold operation. This regime is where the chip operates due to the aggressive voltage scaling. The simulation error was evidenced by comparison to hand analysis and confirmed by the measured power results described below.

The chip was tested and verified using 8-bit sensor data at a clock rate of 1.2 kHz. At this frequency, the voltage could be scaled down to 1.1 V before logical failure. This voltage results in a power consumption of 300 nW. However, to satisfy the back-end processing requirements in real time, the voltage would have to be increased to 1.5 V, even for the low-frequency filtering operations since in this implementation we did not incorporate a variable-voltage power supply or separate the power nets for the different functional units. This results in 560 nW of average power consumption, again far below what is achievable for a vibration-based energy scavenging system [10].

Fig. 15 shows the measured power consumption of the DSP chip as it is configured for decreasing input data bitwidth. Only the DA unit and microcontroller are enabled for these measurements. The graph shows that the power decreases monotonically as the bitwidth decreases, however, the range of power scalability in relative terms (not absolute microwatts) is substantially less than in Fig. 14 because we have reduced the power of the filter so much that the microcontroller now dominates the total power. These functional units have about 100 nW of overhead power at 1.1 V and a 1.2 kHz clock frequency. The NLSL filter unit and the filter buffer add about another 110 nW of power consumption for the chip. Future work should focus on reducing the power of the these units and the microcontroller. The tradeoff between recognition performance and power consumption shown in Fig. 5 was verified.

Table I summarizes the chip and process parameters. A chip photo is shown in Fig. 16. While 0.6-$\mu$m technology is not cutting edge, for applications such as this one which do not demand the highest performance and where switching occurs infrequently, its low leakage characteristics help reduce average power consumption. This particular system implementation has several microwatts of projected power budget, and so it is feasible to integrate new features into the DSP using a more advanced process technology. Static current consumption, including subthreshold leakage, is projected to become comparable to switching current in the sub-100-nm process nodes, at which point new circuit styles which incorporate multiple threshold voltages [24] or body biasing [24] will be necessary for low-power circuits to enable self-powered operation.

## VII. Conclusion

We have described how DA may be used to implement power scalable approximate signal processing by allowing a DSP to vary the input data bitwidth according to performance requirements. This type of data truncation causes increased noise at the input of the implemented filter, degrading output SNR and resulting in reduced recognition performance in detection and classification algorithms. However, the reduced switched capacitance due to the truncation also results in reduced power consumption. Thus, it is possible to trade power consumption for required recognition performance. This technique may be used in conjunction with other approximate signal processing methods such as varying the number of filter taps.

A DSP chip implemented in 0.6-$\mu$m CMOS demonstrates this tradeoff. Its architecture is targeted to detection and classification algorithms where emphasis is placed on reducing the power consumption of preprocessing steps, which dominate the computation in these applications. Aggressive voltage scaling and reduction of switched capacitance present challenges for memory design, which may be addressed by using single-ended charge-sharing sense schemes and optimal predecoding and partitioning of memory subarrays. The resulting power consumption for heartbeat detection algorithms is 560 nW, substantially less than the power available from mechanical vibration converted by a MEMS transducer. The low energy consumption of the DSP per input data sample enables operation of the system from scavenged energy.

## References

[1] J. T. Ludwig, S. H. Nawab, and A. P. Chandrakasan, "Low-power digital filtering using approximate processing," *IEEE J. Solid-State Circuits*, vol. 31, pp. 395–399, Mar. 1996.

[2] C. J. Pan, "A stereo audio chip using approximate processing for decimation and interpolation filters," *IEEE J. Solid-State Circuits*, vol. 35, pp. 45–55, Jan. 2000.

[3] C. Nicol, P. Larsson, K. Azadet, and J. O'Neil, "A low-power 128-tap digital adaptive equalizer for broadband modems," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1777–1789, Nov. 1997.

[4] P. Larsson and C. Nicol, "Self-adjusting bit-precision for low-power digital filters," in *Symp. VLSI Circuits Dig. Tech. Papers*, June 1997, pp. 123–124.

[5] A. Peled and B. Liu, "A new hardware realization of digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-22, pp. 456–462, Dec. 1974.

[6] S. A. White, "Applications of distributed arithmetic to digital signal processing: A tutorial review," *IEEE ASSP Mag.*, vol. 6, pp. 4–19, July 1989.

[7] T. Xanthopoulos and A. Chandrakasan, "A low-power DCT core using adaptive bitwidth and arithmetic activity exploiting signal correlations and quantization," *IEEE J. Solid-State Circuits*, vol. 35, pp. 740–750, May 2000.

[8] S. Uramoto, Y. Inoue, A. Takabatake, J. Takeda, Y. Yamashita, H. Terane, and M. Yoshimoto, "A 100 MHz 2-D discrete cosine transform core processor," *IEEE J. Solid-State Circuits*, vol. 36, pp. 492–499, Apr. 1992.

[9] M. T. Sun, T. C. Chen, and A. M. Gottlieb, "VLSI implementation of a 16 × 16 discrete cosine transform," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 610–617, Apr. 1989.

[10] R. Amirtharajah, S. Meninger, J.-O. Mur-Miranda, A. Chandrakasan, and J. Lang, "A micropower programmable DSP powered using a MEMS-based vibration-to-electric energy converter," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2000, pp. 362–363, 469.

[11] S. Meninger, J. O. Mur-Miranda, R. Amirtharajah, A. Chandrakasan, and J. Lang, "Vibration-to-electric energy conversion," *IEEE Trans. VLSI Syst.*, vol. 9, pp. 64–76, Feb. 2001.

[12] P. E. Landman and J. M. Rabaey, "Architectural power analysis: The dual bit type method," *IEEE Trans. VLSI Syst.*, vol. 3, pp. 173–187, June 1995.

[13] C.-Y. Tsui, K.-K. Chan, Q. Wu, C.-S. Ding, and M. Pedram, "A power estimation framework for designing low-power portable video applications," in *Proc. 34th Design Automation Conf. (DAC'97)*, June 1997, pp. 415–420.

[14] D. G. Gata *et al.*, "A 1.1-V 270-$\mu$A mixed-signal hearing aid chip," *IEEE J. Solid-State Circuits*, vol. 37, pp. 1670–1678, Dec. 2002.

[15] K. Stangel, S. Kolnsberg, D. Hammerschmidt, B. J. Hosticka, H. K. Trieu, and W. Mokwa, "A programmable intraocular CMOS pressure sensor system implant," *IEEE J. Solid-State Circuits*, vol. 36, pp. 1094–1100, July 2001.

[16] M. Nadler and E. Smith, *Pattern Recognition Engineering*. New York: Wiley, 1993.

[17] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, 2nd ed. Reading, MA: Addison-Wesley, 1993.

[18] J.-T. Yoo, K. F. Smith, and G. Gopalakrishnan, "A fast parallel squarer based on divide-and-conquer," *IEEE J. Solid-State Circuits*, vol. 32, pp. 909–912, June 1997.

[19] J. Goodman and A. Chandrakasan, "An energy/security scalable encryption processor using an embedded variable voltage DC/DC converter," *IEEE J. Solid-State Circuits*, vol. 33, pp. 1799–1809, Nov. 1998.

[20] J. Rabaey, *Digital Integrated Circuits: A Design Perspective*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[21] M. Tsukude, S. Kuge, T. Fujino, and K. Arimoto, "A 1.2- to 3.3-V wide voltage-range/low-power dram with a charge-transfer presensing scheme," *IEEE J. Solid-State Circuits*, vol. 32, pp. 1721–1727, Nov. 1997.

[22] R. Evans and P. Franzon, "Energy consumption modeling and optimization for SRAMs," *IEEE J. Solid-State Circuits*, vol. 30, pp. 571–579, May 1995.

[23] A. P. Chandrakasan and R. W. Brodersen, *Low Power Digital CMOS Design*. Norwell, MA: Kluwer, 1995.

[24] J. T. Kao and A. Chandrakasan, "Dual-threshold voltage techniques for low-power digital circuits," *IEEE J. Solid-State Circuits*, vol. 35, pp. 1009–1018, July 2000.

**Rajeevan Amirtharajah** (S'97–M'99) received the S.B. and M.Eng. degrees in 1994 and the Ph.D. degree in 1999, all in electrical engineering from the Massachusetts Institute of Technology, Cambridge. His doctoral research concerned the development of micropower digital signal processing systems that scavenge energy from mechanical vibrations in their environment and use that energy to process information provided by embedded and wearable sensors.

From 1999 to 2002, as a Senior Member of the Technical Staff with High Speed Solutions Corporation (an Intel Company), Hudson, MA, he helped create innovative high performance multidrop bus technologies using electromagnetic coupling and pulse-based modulated signaling. He worked as an ASIC and mixed-signal circuit design Consultant with SMaL Camera Technologies, Cambridge, in 2003. In July 2003, he joined the Electrical and Computer Engineering Department, University of California, Davis, where he is currently an Assistant Professor. His research interests include low-power VLSI design for sensor applications, powering systems from ambient energy sources, and high-performance circuit and interconnect design.

Dr. Amirtharajah is a member of the American Association for the Advancement of Science (AAAS) and Sigma Xi.

**Anantha P. Chandrakasan** (S'87–M'95–SM'01–F'03) received the B.S, M.S., and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 1989, 1990, and 1994, respectively.

Since September 1994, he has been with the Massachusetts Institute of Technology, Cambridge, where he is currently a Professor of electrical engineering and computer science. He held the Analog Devices Career Development Chair from 1994 to 1997. His research interests include low-power digital integrated circuit design, distributed wireless microsensors, ultrawideband radios, and emerging technologies. He is a coauthor of *Low Power Digital CMOS Design* (Norwell, MA: Kluwer, 1995) and *Digital Integrated Circuits* (Englewood Cliffs, NJ: Prentice-Hall, 2002, 2nd ed.). He is also a coeditor of *Low Power CMOS Design* (Piscataway, NJ: IEEE Press, 1997) and *Design of High-Performance Microprocessor Circuits* (Piscataway, NJ: IEEE Press, 2000).

Dr. Chandrakasan has received several Best Paper Awards, including the 1993 IEEE Communications Society's Best Tutorial Paper Award, the IEEE Electron Devices Society's 1997 Paul Rappaport Award for the Best Paper in an EDS publication during 1997, and the 1999 Design Automation Conference Design Contest Award. He served as a Technical Program Co-Chair for the 1997 ISLPED, VLSI Design'98, and the 1998 IEEE Workshop on Signal Processing Systems. He was the Signal Processing Subcommittee Chair for ISSCC 1999–2001, the Program Vice-Chair for ISSCC 2002, and the Technical Program Chair for ISSCC 2003. He was an Associate Editor for the IEEE JOURNAL OF SOLID-STATE CIRCUITS from 1998 to 2001. He serves on the SSCS AdCom. He is the Technology Directions Chair for ISSCC 2004.