An Accelerator-Based Wireless Sensor Network Processor in 130 nm CMOS

Mark Hempstead, *Member, IEEE*, David Brooks, *Member, IEEE*, and Gu-Yeon Wei, *Member, IEEE*

***Abstract*—Networks of ultra-low-power nodes capable of sensing, computation, and wireless communication have applications in medicine, science, industrial automation, and security. Reducing power consumption requires the development of system-on-chip implementations that must provide both energy efficiency and adequate performance to meet the demands of the long deployment lifetimes and bursts of computation that characterize wireless sensor network (WSN) applications. Therefore, this work argues that designers should evaluate the design in terms of average power for an entire workload, including active and idle periods, not just the metric of energy-per-instruction.**

The proposed architecture fully embraces the accelerator-based computing paradigm, including acceleration for the network layer (routing) and application layer (data filtering). Moreover, the architecture can disable the accelerators via VDD-gating to minimize leakage current during the long idle times common to WSN applications. We have implemented the system architecture in 130 nm CMOS which has been tested to operate at 550 mV and 12.5 MHz. Our system uses 100× less power when idle than a traditional microcontroller, and 10–600× less energy when active. This work concludes with an analysis across a wide range of workloads that shows how the system provides energy efficient operation for both low, medium, and high intensity workloads.

***Index Terms*—Accelerator architectures, wireless sensor networks (WSNs).**

I. Introduction

WIRELESS sensor networks (WSNs) enable new applications by embedding computation and sensing in the physical world. However, realizing the full potential of WSNs requires energy efficient platforms that can provide computation across a wide range of workloads under a tight energy budget.

Some of the early deployments of WSNs reveal a range of resource requirements and applications, from high performance monitoring of volcanoes to intermittent measurements of weather [1], [2] and from battle field scenarios to industrial plants [3], [4]. These test deployments found that commercially available microprocessors consume too much power when idle to sustain long deployments over several years. Furthermore, these processors cannot supply the computationally intensive data filtering required to reduce radio traffic. System-on-chip (SoC) implementations of such nodes can provide both energy efficiency and adequate performance to meet the long deployment lifetimes and bursts of computation that characterize WSN applications. WSN workloads are more varied than typical CPU benchmarks they are a application dependent mixture of idle and active computation. Therefore, evaluating the efficacy of new architectures requires new methodologies and metrics.

There are several circuit approaches available to reduce total system power consumption, and each approach has its own architecture and performance implications, from subthreshold circuit design, to asynchronous circuits, to power gating. Researchers have designed several prototypes to operate with a supply voltage below or near the transistor threshold voltage [5]–[10]. By effectively trading off maximum clock rate to reduce energy, these prototypes consume just a few pJ per instruction while running at clock frequencies in the hundreds of kilohertz. Consequently, these designs are limited to only certain applications with low sample rates and limited computational requirements. Another design, the SNAP processor, consumes little clock power and can be easily configured to run at different supply voltages due to its asynchronous circuit implementation [11]. While implementing a system with asynchronous circuits does reduce clock power, it can be considered an orthogonal implementation technique to our approach. As transistor dimensions scale and leakage current density increases, techniques such as power gating are necessary to manage leakage power, especially for applications with long idle times. In this work, our modular architecture provides application support for power gating (VDD-gating) [12]. The aforementioned prototypes and commercially available systems are centered around a general purpose computing engine that limits the energy efficiency of the computation and the granularity of circuit blocks which can be power gated. If one assumes a modular architecture, created by replacing the general purpose engine with an event processor to handle interrupts and accelerator hardware to run regular tasks, the design would reduce the need for the general purpose processor and increase energy efficiency.

This work presents the design and implementation of a prototype developed by taking a holistic approach to the management of power. Through a study of WSN applications, described in Section II, and the PowerTOSSIM simulator [13] developed by our research group, we highlight some key characteristics of WSN applications that motivate our design, including the need for energy efficient computation that is often regular and characterized at compile time. The architecture was first introduced by Hempstead *et al.* in ISCA 2005, but this paper describes a detailed evaluation of the silicon implementation, new workload analysis methodology, and comparison to related work [14]. The prototype was implemented in 130 nm CMOS and characterized for a range of operating conditions described in Section IV.

Manuscript received December 31, 2010; accepted May 12, 2011. This paper was recommended by Guest Editor V. Narayanan.

M. Hempstead is with the Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA 19104 USA (e-mail: mhempstead@coe.drexel.edu).

D. Brooks and G.-Y. Wei are with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2011.2160751

2156-3357/$26.00 © 2011 IEEE

TABLE I
SENSOR SAMPLING RATES OF DIFFERENT PHENOMENA

Phenomena	**Sample Rate** *(in Hz)*
Very Low Frequency	
Atmospheric temperature	0.017 - 1
Barometric pressure	0.017 - 1
Low Frequency	
Heart rate	0.8 - 3.2
Volcanic infrasound	20 - 80
Natural seismic vibration	0.2 - 100
Mid Frequency (100Hz - 1000Hz)	
Earthquake vibrations	100 - 160Hz
ECG (heart electrical activity	100 - 250
High Frequency (>1kHz)	
Breathing sounds	100 - 5k
Industrial vibrations	40k
Audio (human hearing range)	15 - 44k
Audio (muzzle shock-wave)	1M
Video (digital television)	10M

In Section IV-C, we introduce the concept of *energy per task* and *energy per equivalent instruction* to compare the energy consumed by our implementation with the energy consumed by other systems designed for wireless sensor networks. In Section V-B, we compare our architecture approach to a general purpose microcontroller manufactured on the same chip, which provides a method to evaluate our architecture as most of the related systems include a general purpose computing engine. Our final analysis, presented in Section V-C, combines performance and power measurements of our system with application workloads. We show that our architecture provides reduced energy consumption for both low duty cycle applications and high performance computation, unlike many of the related systems that target only low duty cycle workloads.

II. WSNs BACKGROUND

WSNs combine sensing, wireless communication, and computing in one platform and then deploy the platform to observe and interact with the physical world. Nodes within a WSN are programed to collect observations and then perform an application specific filtering of those observations before transmitting the collected data. Often a node will relay samples collected by other nodes in the network which is typically referred to as multi-hop routing. Researchers face challenges delivering the necessary computation under a budget and deciding where to apply design effort.

A. Overview

Deployed sensor networks measure a wide range of phenomena including atmospheric temperature, heart rate, volcanic eruptions, and even the sound of a sniper rifle [1]–[3], [15]. The performance target (cycles of computation per second) for a WSN node is set by the sampling rate for the measured phenomena and the amount of on-node data filtering required. Table I lists the range of sampling rates for different physical phenomena. Environmental measurements (such as temperature and pressure) have time constants on the order of minutes. Consequently, nodes deployed to measure low-frequency phenomena will be idle most of the time. In contrast, applications that measure higher-frequency phenomena (such as seismic vibrations and acoustic signals) will require a large amount of energy efficient computations to achieve the requisite performance under a strict power budget.

B. Challenges

Sensor nodes are sometimes deployed in hard to reach places, which makes changing batteries regularly both difficult and expensive. In this work, we classify node lifetime based on the availability of wired power sources or battery replacements. In some monitoring applications, nodes embedded deeply in the structure of a building would be difficult to manually maintain and, consequently, would require node lifetimes of several years on one battery. In medical domains (not including bio-implants) a patient or health care professional would be able to replace batteries daily. Table II lists a few example application domains with an estimate of their deployment lifetimes and computation requirements.

Scavenging energy from the environment, and using this energy to power the sensor network device would increase the effective lifetime of a wireless sensor node. There are many sources of energy such as solar, vibration, and electro-magnetic radiation [16]. While using vibration as an energy source is promising, the power output is limited to the order of a hundred μW (for mote-size devices). Consequently, an SoC for WSN applications with extremely long lifetimes should target a power budget less than 100 μW to take advantage of energy scavenging hardware.

C. Modeling

In this work, we target a class of habitat monitoring WSN applications that aim for long deployment lifetimes and that incorporate data filtering and multi-hop routing on their nodes. Specifically, this architecture was informed by a volcano monitoring system deployed by Werner-Allen *et al.* [1]. In that system, nodes sample both seismic and infrasound signals and use an exponentially weighted moving average (EWMA) filter to detect interesting events and transmit their data back to a team of vulcanologists.

In collaboration with systems researchers, we developed a hardware power model and simulator for off-the-shelf WSN nodes. PowerTOSSIM achieves excellent accuracy, with an average error of just 4.7% and maximum error of about 13%. In addition to total energy consumption data, PowerTOSSIM outputs real-time traces of simulated current draw. We verified that these traces correspond well with the measured oscilloscope trace for the same application. A more detailed evaluation is available in the PowerTOSSIM paper [13].

Through PowerTOSSIM, we can understand where power is consumed in a WSN device and thus where to focus our design effort. We chose the Surge application which includes both sensing and multi-hop routing. Surge includes two independent execution paths, the *Sense and Transmit* path and the *Receive, Route and Transmit* path. These execution paths are similar to the patterns we observed in many other WSN applications. During Sense and Transmit, Surge collects a sensor sample periodically (every second) and transmits the result to the next hop on the routing tree. When a message is received, the Receive, Route and Transmit path inspects the package and

TABLE II
EXAMPLE WSN APPLICATION DOMAINS

Application Domain	Desired Lifetimes	Computation Requirements (sample rates)	Example
Scientific Applications			
Habitat/Weather Monitoring	Months/decades	very low	Great Duck Island [2]
Volcanic Eruption Detection	Months/decades	mid	Volcano WSN [1]
Military and Security Applications			
Building/Border Intrusion Detection	Years/decades	low	
Structural and Earthquake monitoring	Years/decade	low/mid	
Active Battlefield Sensing	Months	mid/high	Sniper Detection + Localization [3]
Medical Applications			
Long-Term Health Monitoring (pulse)	Days	low	
Untethered Medical Instruments (ECG)	Days	med	EKG mote [15]
Business Applications			
Supply Chain Management	Months	low	
Expired/Damaged Goods Tracking	Months	low	
Factory/fab monitoring	Months/years	med/high	Industrial WSN [4]

checks the routing and recent message tables to decide whether to transmit the message.

We simulated a six node network running the surge application in PowerTOSSIM. Surge does not incorporate data filtering or intelligent power-cycling of the radio. Therefore, the radio is on and listening all the time so every sensor sample is transmitted. In this condition, the Radio consumes 59% of total system power and the CPU 35%. If the application incorporated more intelligent data filtering, fewer packets would be sent, but more CPU power would be required to run the filtering algorithm. Researchers have shown significant reductions in radio power through a low power listening MAC protocol [17] or through time division multiplexing. However, these techniques also require more computation, shifting the burden to the CPU. We concluded that focusing our design effort on building more energy efficient computation blocks would have the most potential to reduce the total power consumption of a WSN node. From an understanding of surge we learned the importance of examining all layers of the design space to reduce power. Applications that do not manage radio power by offloading work to the CPU have high total system power consumption even with an energy efficient processor.

III. ACCELERATOR-BASED WSN PROCESSOR IN 130 NM CMOS

Through the analysis and modeling of WSN applications, we showed the need for a new computing engine that can address a wide range of workloads under a strict power budget, thereby enabling energy scavenging. When the node is deployed to measure high-frequency phenomena, active power consumption will dominate total power consumption. Therefore, active power must be reduced by providing computation for common tasks that is more energy efficient than a general purpose microcontroller. Leakage current will dominate total power consumption when the node is running workloads with long idle times. Therefore, WSNs would benefit from a method of computation that supports the VDD-gating of computational elements when they are not needed. In this section we describe the architecture of our accelerator-based system and the prototype implementation in 130 nm CMOS.

A. Architecture

The system architecture combines the energy efficiency found in application specific integrated circuits with the flexibility and programmablility of a general purpose processor. As power consumption is the highest priority design constraint, the proposed event-driven system for WSNs uses three techniques to reduce power consumption.

1) *Lightweight Event Handling in Hardware:* Initial responsibility for handling incoming interrupts is given to a specialized Event Processor, removing the software overhead that would be required to provide event handling on a general-purpose processor.
2) *Hardware Acceleration for Typical WSN Tasks:* Modular hardware accelerators are included to complete regular application tasks such as data filtering and message routing.
3) *Application-Controlled Fine-Grained VDD-Gating:* Addressing leakage current with architecture support for VDD-gating enables accelerator blocks to be powered off when unused.

Fig. 1 presents a block diagram of the prototype system. The event processor (EP) is a small programmable state machine that runs interrupt service routines (ISRs) to control the flow of data between the on-chip memory and multiple accelerators, such as the message processor, programmable data filter, and timer, which are memory mapped and connected via the system bus [14]. The EP also acts as a power manager, turning accelerators on and off as needed by the running application. While the system also includes an 8-bit general-purpose microcontroller to handle infrequent and irregular tasks, it can usually be disabled. During long idle times, only the EP—and perhaps select blocks such as the timer—must be powered. The tester I/O block facilitates testing to verify functionality.

A key benefit of the modular design of the architecture is its ability to employ fine-grained power management of individual components (both masters and accelerators). Selectively turning off components and using VDD-gating enables the system to minimize leakage power. For example, the general-purpose microcontroller core could be relatively complex and power-hungry when active, but can be VDD-gated most of the time when idle. The event processor handles all interrupts,

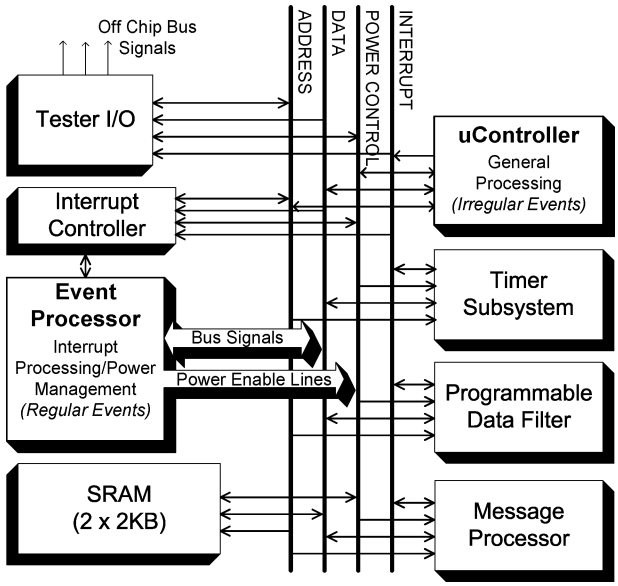


Fig. 1. System block diagram.

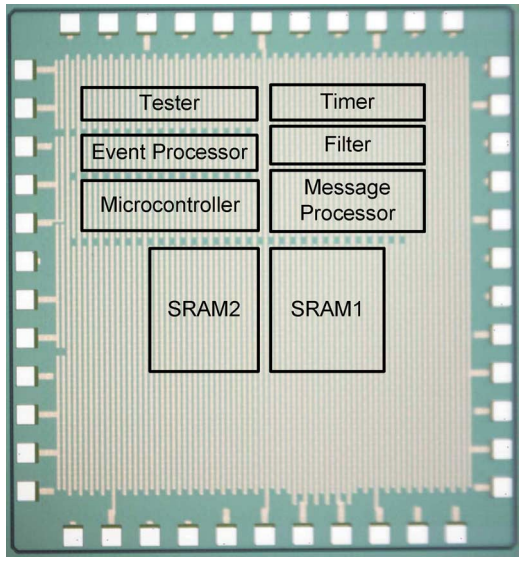


Fig. 2. Die photograph of the prototype. System includes an event processor and several accelerators for regular operation. The system has been realized in 130 nm CMOS on a 2 mm × 2 mm die.

distributes tasks to accelerator devices, and wakes up the microcontroller only as necessary; which is rarely.

When executing applications, the event processor coordinates the actions of the hardware accelerators and other components connected on the system bus. Our prototype includes several accelerators for data filtering, message routing and timing. However, we envision future systems based on this architecture including a richer set of hardware accelerators. This paper presents the silicon implementation and evaluation of the system; a more detailed description of the architecture was presented at ISCA'05 [14].

B. Implementation

Because the architecture is new and the power consumption targets are aggressive, physical measurements are necessary to verify that the architecture meets our goals. We designed a prototype in a semi-custom design flow to evaluate the energy efficiency of the accelerator-based design, the energy savings of event processor control, and the efficacy of VDD-gating to manage leakage current. The major components of the system—the event processor and accelerator blocks—were designed from the ground up and described in Verilog. We used synthesis from RTL with a foundry-supplied standard cell library to generate layout. The accelerator blocks communicate on the shared bus through a custom lightweight signaling protocol with either the event processor or microcontroller serving as the bus master. We were able to easily incorporate an opensource TV80 (z80-based) microcontroller because of our use of Verilog synthesis. This implementation includes nine different power domains on chip because isolating the individual contribution of each major system component was an important part of our test plan.

The chip was manufactured in a 130 nm bulk CMOS process with eight layers of metal. A die photograph is shown in Fig. 2. The system contains 444 982 transistors including 4 KB of foundry-supplied SRAM. Using these SRAMs saved significant time over a full custom design but limited the ability of

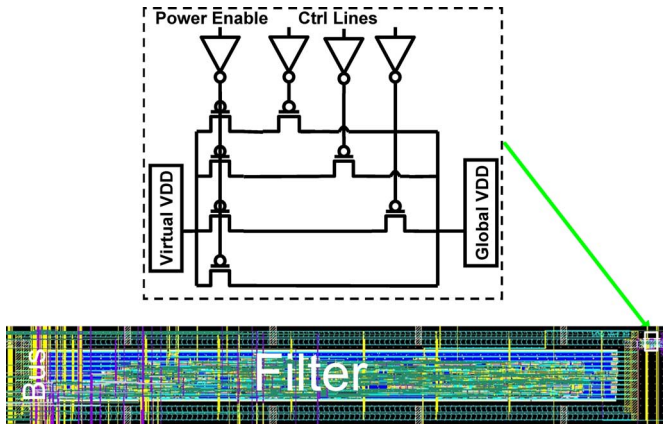
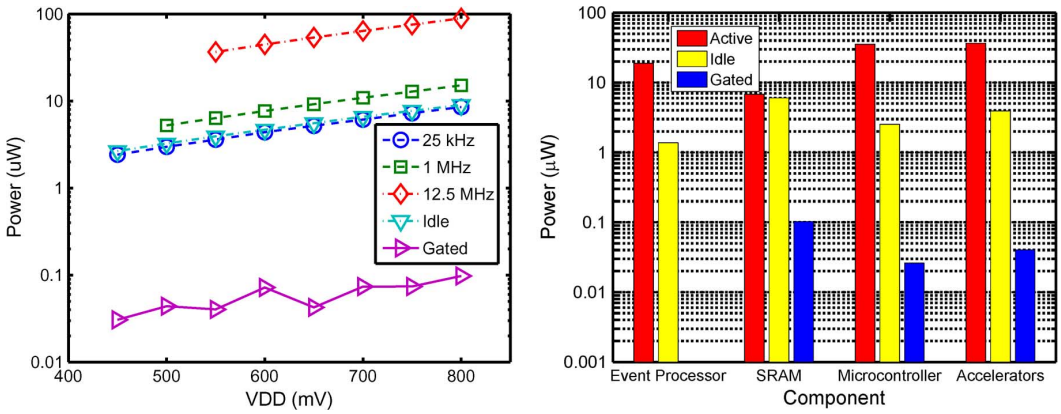


Fig. 3. Custom VDD-gating circuit. This schematic shows four different parallel legs that are used to control VDD-gating strength. The layout of the filter component shows where the VDD-gating circuit is attached. In this example, the VDD-gating circuit requires an additional area of 3.2%.

the prototype to operate in subthreshold. The chip area is pad limited due to the large number of pins purposely added for testing and the fine-grain power measurements of nine different power domains. Decoupling-capacitors were included on all of the top level power domains as well as on the virtual power domains separated by VDD-gating transistors. We used a high performance 130 nm process with a low energy per switch but a large amount of leakage current per unit area, which our results indicate is reduced by VDD-gating and the architecture. A lower leakage process with high-VT transistors would most likely consume less leakage current but more active power consumption when computing.

A custom VDD-gating block was designed; the schematic and example layout placement is shown in Fig. 3. The schematic shows four different parallel legs which are used to control VDD-gating strength and measure its effect on power consumption and performance. The layout of the VDD gating block is modular so that blocks can be tiled in parallel and attached



(a) (b)

Fig. 4. Measured power consumption of the prototype under different supply voltages and clock frequencies. Plot of the power consumption for the Accelerator power domain while sweeping voltage from 450 mV to 800 mV and frequency from 25 kHz to 12.5 MHz. Idle power is measured with the external clock off (0 MHz at 550 mV). The VDD-gating transistor is off (not-conducting) during the measurement of gated power. (a) Accelerators; (b) full system—550 mV and 12.5 MHz.

to the power ring of the accelerator. We chose the number of VDD-gating blocks so the width of the gating transistors was equal to the active width of the transistors switching in the circuit, to minimize the impact on performance [12].

IV. Measurements

We built the prototype described above to evaluate the power and performance of our architectural approach. Our implementation and test setup provide distinct power domains that enable detailed power measurements across different voltage and frequency pairs. In this section, we describe the test methodology and power measurements for each major block both when the block is actively computing and when it is sitting idle and consuming leakage current. As the concept of energy-per-instruction does not apply to accelerators without explicit instructions, measuring the energy efficiency of an accelerator-based design requires new metrics. In this section we introduce the concept of *energy per task*, a metric used to compare this architecture to other systems in the literature.

A. Test Methodology and Functional Verification

We evaluated the 130 nm test chip to verify functionality and measure power consumption. Our custom test board was designed to evaluate the functionality of the system across voltage and frequency as dynamic voltage and frequency scaling is a commonly used power management technique. By collecting data for a wide range of voltage and frequency pairs, we were able to find the best operating point for our system under a given workload; this analysis will be presented later in Section V-C.

We wrote a set of test applications made up of interupt service routines (ISRs) expressed in the event processor instruction set. Twenty-two test benchmarks were written that fall under the following categories.

1) *Full Application—Functional Verification:* This benchmark implemented a sense and transmit path of the volcano monitoring application, described in Section II-C, and uses all of the accelerator blocks. The results of each test are transmitted off-chip and compared with the expected output.
2) *Idle and Gated Benchmarks:* The benchmark uses a set of configuration registers to configure the VDD-gating transistors for each accelerator block individually so idle current and gated current can be measured for each block.
3) *Active Power Virus:* We wrote a set of benchmarks for each block containing a tight infinite loop, that keeps the block running continuously to measure the upper bound of active power consumption.

Our first experimental measurements have verified reliable operation across a range of lower clock frequencies—25 kHz to 12.5 MHz—that are suited to the low power needs of typical WSN applications. The system functions correctly at 12.5 MHz with a supply voltage of just 550 mV. Our operating frequency was limited by the ICs on the test board, though post layout simulations indicated the chip can run up to 100 MHz with a nominal supply voltage of 1.2 V. We swept the supply voltage down to the point at which the chip stopped functioning correctly; 450 mV at a maximum of 25 kHz. Further experiments revealed that the foundry-supplied SRAM is the component that fails first, as it does not reliably operate with a supply voltage less than 450 mV. Several researchers have proposed SRAM cells that could solve this problem through the use of additional access transistors [18]. Our functional tests showed a wide range of possible operating points for our test system.

B. Block Level Power Measurements

We designed our test chip with nine different power domains so that we could measure the power consumption of the different functional blocks. We ran power virus benchmarks on our system, sweeping supply voltage and clock frequency, to measure active power consumption. We isolated leakage current by measuring the chip with the power supply connected, the event processor in the ready state, and the clock signal held at zero (0 MHz). Once the system was in steady-state we measured the leakage current with a DMM, using a wide integration window for increased accuracy. We repeated the same experiment with the VDD-gating transistors off to measure the effect of VDD-gating on power consumption.

Fig. 4(a) presents measurements of just the accelerator block power domain where VDD is swept from 450 mV to 800 mV and clock frequency is measured up to 12.5 MHz. At low frequencies most of the power consumed is due to leakage current, as is illustrated by the overlap of the 25 kHz active power and idle power curves. Idle leakage power is high for all of the power domains—between 1 μW and 10 μW, depending on the power domain and supply voltage. However, because the system architecture supports VDD-gating, leakage power is reduced by two orders of magnitude (98$\times$ at 550 mV) when VDD-gating is used on the accelerators.

Fig. 4(b) plots the per-block power consumption of the system, running custom microbenchmarks written to exercise each block in three operating modes—active (12.5 MHz at 550 mV), idle (0 MHz at 550 mV), and powered off (VDD-gated). For clarity, we graph one particular operating point—550 mV and 12.5 MHz—though power was measured at other voltage and frequency pairs. The Event Processor was designed to replace the microcontroller for most system management tasks with the aim to reduce power. The data illustrate the benefits of this approach, as the microcontroller consumes 1.9$\times$ active and 1.8$\times$ the leakage power consumed by the EP. VDD-gating reduces the power consumption of individual blocks by 50–100$\times$, which helps to minimize power consumption during long periods of inactivity. The event processor block cannot be VDD-gated since it must always be available to handle interrupts. The accelerator blocks consume more power when fully active than the microcontrollers but, as shown in Section V-B, the more computationally efficient accelerators lead to energy savings.

For the SRAM, active power consumption does not significantly increase with a higher clock frequency. Because the transistors within an SRAM have a lower active switching factor (α) than combinational logic, almost all power is due to idle leakage. Due to higher transistor density, the idle power consumed by the SRAM is between 3$\times$ and 5$\times$ larger than the idle power of any of the other blocks. As at least one block must be powered on to retain system state, future system architectures for WSNs must address SRAM leakage power. In the following subsections we use the data presented in this section to evaluate our specific design choices and accelerator-based architectures in general.

C. Energy per Task and Energy per Instruction

Ultimately, battery life is determined by the energy consumption of the system and thus comparing this work with other prototypes requires a measure of energy efficiency. Traditionally, computer architects have used the metric of *energy per instruction* to compare the energy efficiency of different CPUs. In our system, an *instruction* does not represent a fixed amount of computation, because computation is handled primarily in the accelerators. For example, a TRANSFER instruction could trigger a read or write operation in the SRAM, a full filter operation, or a CAM look-up—these operations have vastly different transistor switching rates and consume markedly different amounts of energy. Because the commonly used metric of energy-per-instruction cannot be easily applied to accelerator-based systems, we introduce the concept of *energy per task*. We defined a task as a collection of dependent computations that are executed periodically. We present measurements of a task similar in nature to the volcano monitoring application. This task, the *Sense and Transmit Task*, takes 131 cycles to execute.

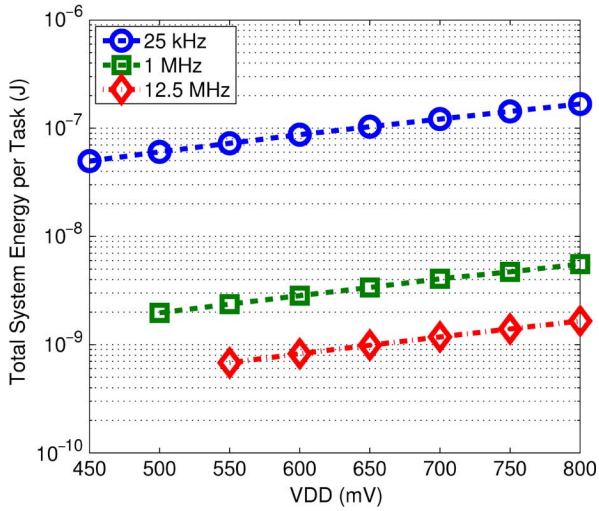


Fig. 5. Energy per task of Sense and Transmit Task. Application includes all accelerator blocks and power contributions from the SRAM and EP.

Fig. 5 presents the energy per task of the Sense and Transmit Task across voltage and frequency. This version of Sense and Transmit includes an EWMA filter operation before the data is sent to the message processor. Consequently, this task includes all accelerator blocks, the event processor, and SRAM. For a fixed supply voltage, the value of energy per task is lower at higher clock frequencies, because leakage energy increases and active energy remains the same. The amount of leakage energy consumed by the processor is an integration of leakage current over the duration of the task. At a given supply voltage, active energy for a given task is independent of frequency because the number of transistor switches remains fixed.

Energy per task is lowest, 678.9 pJ/task, at 550 mV and 12.5 MHz. The Sense and Transmit task takes 131 cycles to execute on our system. On the Mica2 mote, an equivalent routines requires 1532 Mica2 instructions. Using this information we compute the energy per equivalent instruction as 0.44 pJ. This is significantly lower than systems for WSNs in the literature.

V. Evaluation

Potential users of computing platforms for WSNs are concerned with overall performance and average power consumption and not just the maximum power and leakage power of the system. Similarly, system designers want to be able to compare architectures independent of a particular implementation (process technology, memory size, clock frequency). Many computing platforms for WSNs are based around a general purpose processor. Therefore, by running the same workloads on general propose and accelerator-based architectures implemented in the same technology we can isolate the performance and power differences due to the architecture. The average power of a system will depend on fraction of the time spent idle vs. the time actively computing. Using the measured data and sweeping workload we are able to give users a more comprehensive picture of total power consumption. The evaluation we present in this section compares our particular implementation

TABLE III
COMPARISON TO OTHER SYSTEMS DESIGNED FOR WSNS

System	Arch Style	Datapath width (bits)	Circuit Techniques	Memory (KB)	Process (nm)	VDD (V)	Clk Freq (MHz)	Energy (pJ/ins)
Atmel ATmega128L	General Purpose (GP)	8	NA	4	350	3.0	7.3	3200
TI MSP430	GP	16	NA	10	Unknown	3.0	8.0	750
S. Hanson [20]	GP	8	Subthreshold	0.3125	130	0.35	0.354	3.5
B. Zhai [10]	GP	8	Subthreshold	0.25	130	0.36	0.833	2.6
Phoenix [9]	GP	8	Near-Threshold	0.41	180	0.5	0.106	2.8
J. Kwong [5]	GP	16	Subthreshold	128	65	0.5	0.434	27.3
Charm [21]	GP + protocol	NA	Two power domains	68	130	1/0.3	8	96
SNAP [22]	GP	16	Asynchronous	8	180	0.6	23	24
SmartDust [23]	GP	8	Two Clocks	3.125	250	1.0	0.5	12
THIS WORK	Accelerator	8	VDD-Gating	4	130	0.55	12.5	0.44
THIS WORK	GP	8	VDD-Gating	4	130	0.55	12.5	3.4

to related work in the literature then we assess the accelerator-based approach against general purpose architectures and finally sweep different workload intensities.

A. Comparison to Related Work

Several research groups have recognized the need for ultra-low power systems designed specifically for wireless sensor networks. The systems differ significantly because of the architecture decisions and circuit techniques used to implement the system. For example, several systems are based around a traditional general purpose core but the circuits are designed to operate in subthreshold or near-threshold—trading-off performance for reduced power consumption. Our work operates above threshold voltage but uses hardware acceleration to increase energy efficiency. First, we categorized systems based on the circuit techniques employed to improve energy consumption.

- *Subthreshold Operation:* By using a power supply less than the threshold voltage, systems such as the Subliminal and Phoenix processors from the University of Michigan and a subthreshold MSP430 from MIT are able to trade off performance for reduced active power consumption [5]–[10].
- *Asynchronous Circuits:* Processors such as SNAP from Cornell University eliminate clock power by relying on asynchronous circuits [11], [22].
- *Power Supply Gating:* To address increasing leakage current, the Charm processor, from the University of California at Berkeley, and our work employ transistors that switch the power supplies of unused blocks [14], [20].

Along with different circuit techniques, designers of WSN processors differ in their approach to architecture support for applications.

- *General Purpose Computation:* Off-the-shelf and custom designed systems employ load-store or accumulator-based processors as the core processing engine of the system.
- *Application Acceleration:* Our work and the Charm processor from University of California at Berkeley provide hardware acceleration for common tasks to reduce active energy consumption and increase system performance.

We tabulated key parameters for each of the discussed systems, including circuit techniques, architecture style, datapath width, throughput and energy per instruction. Table III presents the results of the tabulation. The processors at the top (Atmel ATMega128 L, TI-MSP430) are off-the-shelf microcontrollers included in commercially available WSN nodes such as the Mica2. The remaining processors are prototype systems designed specifically for WSN applications.

From Table III, we observe a relationship between the use of subthreshold operation and the performance and energy consumption of the system. All of the systems that operate in subthreshold are limited to clock frequencies less than 1 MHz. However, the low supply voltage results in a low energy per instruction between 2 and 4 pJ. Our system uses transistor switches more efficiently through hardware acceleration. Consequently, our system has the lowest measurement of energy per equivalent instruction when the accelerators are used (0.44 pJ). For irregular tasks that employ the general purpose microcontroller, our system has a higher energy per instruction than the systems operating in subthreshold (3.4 pJ). As our per-block power measurements show, SRAMs can consume a dominant fraction of total energy consumption. Consequently, systems that contain larger memories (greater than a few KB) consume larger amounts of energy compared to similar systems at the same voltage, frequency and architecture.

Unfortunately, standard benchmark suites do not exist for the WSN space, though a few research groups have proposed some possibilities [23], [24]. Without running the same application on each system, it is not possible to judge the programmablility, energy efficiency, and performance of the different systems fairly. The efficacy of the energy per instruction metric to compare different systems has been questioned before but, in this case, using it could result in misleading conclusions. The notion of an *instruction* is lost on both the Charm processor and our system because most of the processing is handled by custom hardware accelerators. Even among the general-purpose architectures the differences in instruction set architectures (ISAs), process technologies, memory sizes, and clock frequencies, make selecting the most energy efficient architecture is difficult. Also, WSN applications often experience long periods of inactivity.

B. Comparison to General Purpose and Workload Analysis

The metric of energy per instruction does not isolate the benefits of an accelerator-based architecture from the process technology, circuit implementation, and amount of SRAM.

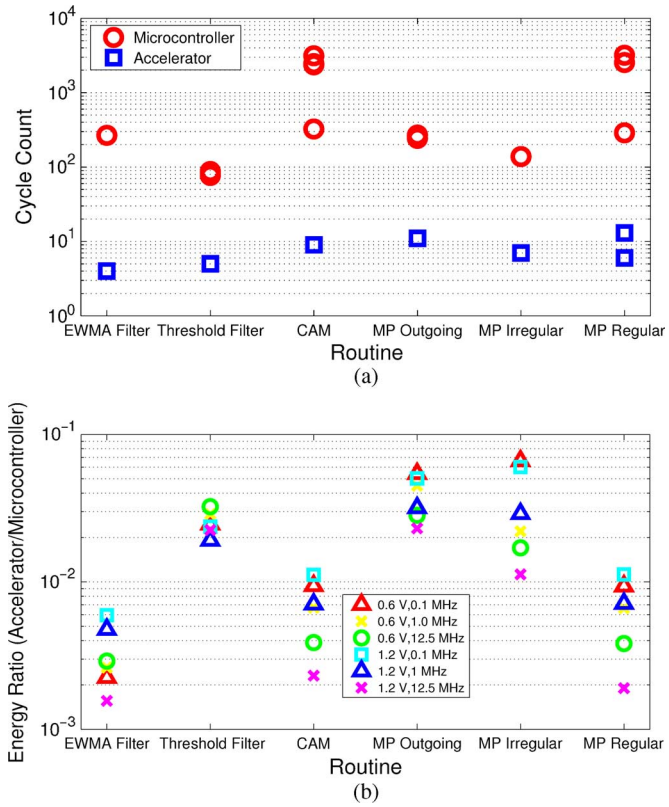


Fig. 6. Performance and power benefits of specialization. Test routines were executed both on the hardware accelerators and the microcontroller. Cycle count and energy savings are presented. (a) Cycle count comparison; (b) energy savings.

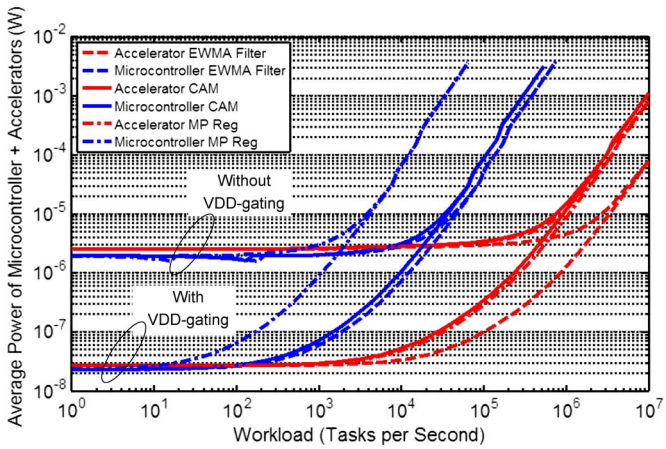


Fig. 7. Power versus workload.

Thus, we compare the cycle count and energy of full applications running on accelerators to applications running on the on-die general-purpose microcontroller. These applications combine data filtering, outgoing message preparation, and flood-based message routing, which are prototypical WSN routines. We analyze routines for data filtering (EWMA and threshold); network routing using a CAM structure; recording an outgoing message; detecting an incoming irregular message; and automatically relaying a regular message. The on-die Z80 microcontroller closely resembles 8-bit architectures employed in other WSN SoCs. For fairness, all routines were written in assembly and hand-tuned for accelerator-based and microcontroller-based operation, respectively. Fig. 6(a) presents the cycle count of each routine for both scenarios. Accelerators process data in parallel and include simplified decode logic, enabling the speedups. Multiple points for a particular routine reflect different inputs that yield different performances. Accelerator implementations see cycle speedups from 15 to 635×, which directly translate in to energy savings.

In addition to cycle count improvements, specialized hardware offers energy savings. Fig. 6(b) plots a ratio of energy consumption when the same routines are run on the accelerators against the same routines run on the microcontroller across a range of voltages and frequencies. The test chip includes multiple independent power domains, implemented to allow the isolated measurement of power consumed by each major block. Each test routine was looped in order to measure average power using a board-level digital multimeter. Hardware accelerators consume 1/10th to 1/600th the energy consumed by software-based routines running on the microcontroller. Routines that see larger speedups also enjoy lower relative energy consumption. It is interesting to note that higher-frequency operation consistently leads to lower relative energy consumption with specialized hardware.

Our analysis illustrates the performance and energy benefits of hardware acceleration for the computation of specific benchmarks. However, real application workloads are made up of many tasks executing over a period of time with idle periods between bursts of computation.

C. Workload Analysis and DVFS

By incorporating the concept of workload in our analysis, we bring together all features of the architecture (speedup, energy efficiency, VDD-gating) and calculate total energy consumption. As detailed in Section II, the amount of computation required to execute WSN application varies by orders of magnitude depending on the phenomena being sensed, the amount of computation required, and the complexity of the operation. System clock frequency and supply voltage also depend on the computation requirements of the workload. Idle leakage power consumption between tasks was not captured in the calculation of speedup but can be a large contributor to total power consumption.

Building on individual characterizations in Section IV, we compare block-level power consumption for different workload requirements and include idle power in our analysis. In order to clarify the comparison, these results exclude additional system power overheads (e.g., EP and SRAM) common to both types of systems. WSN workload intensity varies significantly depending on the observed phenomena—from 1 task/minute for weather observations to $> 10^5$ tasks/s for high-frequency data collection. While most workload requirements are low, sometimes short bursts of high-performance, time-sensitive activity are followed by long idle times (e.g., bursty seismic activity proceeding a volcanic eruption).

Fig. 7 plots the average power consumption of routines run on either the accelerators or the microcontroller while varying workload intensity. For each datapoint, the lowest power voltage/frequency operating point was chosen, For light workloads (< 10 tasks/s), the system can operate at the lowest

voltage and frequency (450 mV, 25 kHz) and power consumption is dominated by leakage current. For medium-intensity workloads (10^4 tasks/s), using accelerators provides 1000× power savings due to a 635× speedup in cycle counts and a 50% lower supply voltage. As workload increases, active power dominates until the clock frequency required by the microcontroller reaches the performance limit of the system at the maximum supply voltage of 1.2 V. Routines run on the accelerator can operate up to 10^7 tasks per second with a voltage less than 1.1 V. The plot also demonstrates that VDD-gating lowers the power consumption for both scenarios under light loads, but the accelerators' higher inherent performance enables components to be VDD-gated for longer periods of time, which results in additional power savings.

VI. Conclusion

To explore some of the microarchitecture challenges of accelerator-based designs, we developed a system architecture and prototype processor for WSN applications. Our system architecture includes a set of hardware accelerators for typical WSN tasks and an event processor to facilitate communication and power management among the accelerator devices. We include application support for VDD-gating because leakage current can dominate the total power consumption of some WSN applications with long idle times. We constructed a prototype in 130 nm CMOS and measured the power consumption of each major functional block.

We found that evaluating the efficacy of our architecture and comparing it to related systems required the creation of several new metrics and analysis methodologies. Compared to similar microprocessors proposed for WSNs, our system has the lowest energy per equivalent instruction (0.44 pJ). Because the performance requirements of WSN nodes vary widely, we conducted an analysis of our system while sweeping workload and scaling voltage and frequency. The results of this analysis show that our system architecture sees a reduction of 100× for low intensity workloads due to VDD-gating. Due to a combination of hardware acceleration and voltage scaling, our system sees 10–1000× power reduction over general purpose-based designs for medium-intensity workloads.

As technology continues to scale and energy efficiency remains important to consumers, mobile and desktop processors will need to incorporate an increasing amount of specialization to maintain growth in microprocessor performance. In the future, we aim to adapt our holistic approach and lessons learned from building this prototype to the mobile and server computing markets.

References

[1] G. Werner-Allen, K. Lorincz, J. Johnson, J. Less, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *Proc. USENIX Symp. Oper. Syst. Design Implem. (OSDI)*, Nov. 2006, pp. 381–396.

[2] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *1st Eur. Workshop Wireless Sensor Netw. (EWSN)*, Berlin, Germany, Jan. 2004.

[3] G. Simon *et al.*, "Sensor network-based countersniper system," in *Proc. 2nd ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Baltimore, MD, Nov. 2004, pp. 1–12.

[4] L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, and M. Yarvis, "Design and deployment of industrial sensor networks: Experiences from a semiconductor plant and the north sea," in *Proc. 3rd ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, San Diego, CA, Nov. 2005, pp. 64–75.

[5] J. Kwong, Y. Ramadass, N. Verma, M. Koesler, K. Huber, H. Moormann, and A. Chandrakasan, "A 65 nm sub- V_t, microcontroller with integrated SRAM and switched-capacitor DC-DC converter," presented at the IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, Feb. 2008.

[6] L. Nazhandali, B. Zhai, J. Olson, A. Reeves, M. Minuth, R. Helfand, S. Pant, T. Austin, and D. Blaauw, "Energy optimization of subthreshold-voltage sensor network processors," presented at the 32nd Annual International Symposium on Computer Architecture (ISCA), Madison, WI, Jun. 2005.

[7] L. Nazhandali, M. Minuth, B. Zhai, J. Olson, S. Hanson, T. Austin, and D. Blaauw, "A second-generation sensor network processor with application-driven memory optimizations and out-of-order execution," presented at the ACM/IEEE International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES), San Francisco, CA, Sep. 2005.

[8] M. Seok, S. Hanson, Y.-S. Lin, Z. Foo, D. Kim, Y. Lee, N. Liu, D. Sylvester, and D. Blaauw, "Performance and variability optimization strategies in a sub-200 mV, 3.5 pJ/inst, 11 nW subthreshold processor," presented at the IEEE Symposium on VLSI Circuits (VLSI-Symp), Kyoto, Japan, Jun. 2008.

[9] S. Hanson, B. Zhai, M. Seok, B. Cline, K. Zhou, M. Singhal, M. Minuth, J. Olson, L. Nazhandali, T. Austin, D. Sylvester, and D. Blaauw, "The phoenix processor: A 30 pw platform for sensor applications," presented at the IEEE Symposium on VLSI Circuits (VLSI-Symp), Honolulu, HI, Jun. 2007.

[10] B. Zhai, L. Nazhandali, J. Olson, A. Reeves, M. Minuth, R. Helfand, S. Pant, D. Blaauw, and T. Austin, "A 2.60 pJ/inst subthreshold sensor processor for optimal energy efficiency," presented at the IEEE Symposium on VLSI Circuits (VLSI-Symp), Honolulu, HI, Jun. 2006.

[11] V. Ekanayake, C. Kelly, and R. Manohar, "An ultra low-power processor for sensor networks," in *Proc. 11th Int. Conf. Architectural Support Program. Lang. Oper. Syst.*, Boston, MA, Oct. 2004, pp. 27–36.

[12] M. Powell, S.-H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar, "Gated-Vdd: A circuit technique to reduce leakage in deep-submicron cache memories," presented at the International Symposium on Low Power Electronics and Design (ISLPED), Rapallo/Portacino Coast, Italy, Jun. 2000.

[13] V. Shnayder, M. Hempstead, B.-R. Chen, G. Werner Allen, and M. Welsh, "Simulating the power consumption of largescale sensor network applications," in *Proc. 2nd ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Baltimore, MD, Nov. 2004, pp. 188–200.

[14] M. Hempstead, N. Tripathi, P. Mauro, G.-Y. Wei, and D. Brooks, "An ultra low power system architecture for sensor network applications," presented at the 32nd Annual International Symposium on Computer Architecture (ISCA), Madison, WI, Jun. 2005.

[15] T. R. F. Fulford-Jones, G.-Y. Wei, and M. Welsh, "A portable, low-power, wireless two-lead EKG system," in *Proc. 26th IEEE EMBS Annu. Int. Conf.*, San Francisco, CA, Sep. 2004, pp. 2141–2144.

[16] S. Roundy, P. K. Wright, and J. Rabaey, "A study of low level vibrations as a power source for wireless sensor nodes," *Comput. Commun.*, vol. 26, no. 1, pp. 1131–1144, Jul. 2003.

[17] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. 2nd ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Baltimore, MD, Nov. 2004, pp. 95–107.

[18] B. H. Calhoun and A. Chandrakasan, "A 256 kb sub-threshold SRAM in 65 nm CMOS," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, Feb. 2006, pp. 2592–2601.

[19] S. Hanson *et al.*, "Exploring variability and performance in a sub-200-mv processor," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 881–891, Apr. 2008.

[20] M. Sheets, F. Burghardt, T. Karalar, J. Ammer, Y. H. Chee, and J. Rabaey, "A power-managed protocol processor for wireless sensor networks," in *Proc. VLSI*, Jun. 2006, pp. 212–213.

[21] V. Ekanayake, C. Kelly, IV, and R. Manohar, "An ultra low-power processor for sensor networks," in *Proc. ASPLOS*, Oct. 2004, pp. 316–317.

[22] V. Ekanayake, C. Kelly, and R. Manohar, "BitSNAP: Dynamic significance compression for a low-energy sensor network asynchronous Processor," in *Proc. 11th Int. Symp. Asyncron. Circuits Syst.*, Mar. 2005, pp. 144–154.

[23] M. Hempstead, M. Welsh, and D. Brooks, "TinyBench: The case for a standardized benchmark suite for tinyOS based wireless sensor network Devices," in *Proc. 1st IEEE Workshop Embedded Netw. Sensors (EmNets)*, Tampa, FL, Nov. 2004, pp. 585–586.

[24] L. Nazhandali, M. Minuth, and T. Austin, "Sensebench: Toward an accurate evaluation of sensor network processors," presented at the IEEE International Workload Characterization Symposium, Austin, TX, Oct. 2005.

Mark Hempstead (M'08) received the B.S. in computer engineering from Tufts University, Medford, MA, and the S.M. and Ph.D degrees in engineering from Harvard University, Cambridge, MA, in 2003, 2005, and 2009, respectively.

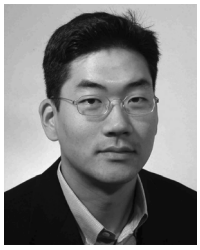
He is the Junior Colehower Chair Assistant Professor in the Department of Electrical and Computer Engineering, Drexel University, Philadelphia, PA. His research interests in low power computing span the areas of VLSI design, computer architecture, mobile computing, and wireless sensor networks.

Prof. Hempstead is a member of ACM and USENIX.

David Brooks (M'02) received the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, in 2001.

He is a Gordon McKay Professor of Computer Science in the School of Engineering and Applied Sciences at Harvard University. His research interests include power-efficient computer system design, variation-tolerant computer architectures, and embedded system design.

Dr. Brooks is a member of ACM.



Gu-Yeon Wei (M'00) received the B.S., M.S., and Ph.D. degrees in Electrical Engineering from Stanford University, Stanford, CA, in 1994, 1997, and 2001, respectively.

He is Associate Dean of Academic Programs and Gordon McKay Professor of Electrical Engineering and Computer Science in the School of Engineering and Applied Sciences at Harvard University. His research interests span several areas—mixed-signal circuit design, computer architecture, and compilers—to address variability and power consumption issues in computing systems that use nanoscale CMOS technologies.