Dynamic Voltage Scaling Techniques for Distributed Microsensor Networks

Rex Min, Travis Furrer, and Anantha Chandrakasan
Department of EECS
Massachusetts Institute of Technology
{rmin, furrer, anantha}@mtl.mit.edu

Abstract

Distributed microsensor networks promise a versatile and robust platform for remote environment monitoring. Crucial to long system lifetimes for these microsensors are algorithms and protocols that provide the option of trading quality for energy savings. Dynamic voltage scaling on the sensor node's processor enables energy savings from these scalable algorithms. We demonstrate dynamic voltage scaling on the beginnings of a sensor node prototype, which currently consists of a commercial processor, a digitally adjustable DC-DC regulator, and a power-aware operating system.

1. Introduction

Distributed microsensor networks are emerging as a compelling new hardware platform for remote environment monitoring [1]. Researchers are considering a range of applications including remote climate monitoring, battlefield surveillance, and intra-machine monitoring [2]. A distributed microsensor network consists of many small, expendable, battery-powered wireless nodes. Once the nodes are deployed throughout an area of interest, they collect data from the environment and automatically establish *ad hoc* networks to transmit their data to a base station. The nodes collaborate to gather data and extend the operating lifetime of the entire system. Compared to larger macrosensor-based systems, microsensor networks offer a longevity, robustness, and ease of deployment that are ideal for environments where maintenance or battery replacement may be inconvenient or impossible [3].

The MIT μAMPS (μ-Adaptive Multi-Domain Power-Aware Sensors) project is developing the enabling technologies for distributed microsensor networks. We envision microsensor networks composed of hundreds to thousands of small, inexpensive, and homogeneous nodes. Once deployed, the nodes periodically organize themselves into clusters, based on the selection and location of *cluster head* nodes. Cluster heads receive data from the nodes in their cluster, aggregate the data locally through beamforming, and transmit the result to a base station. Figure 1 illustrates this organization and data flow. The use of a two-hop routing protocol and data fusion at the cluster head substantially reduce total transmission energy in the network, prolonging system lifetime [4].

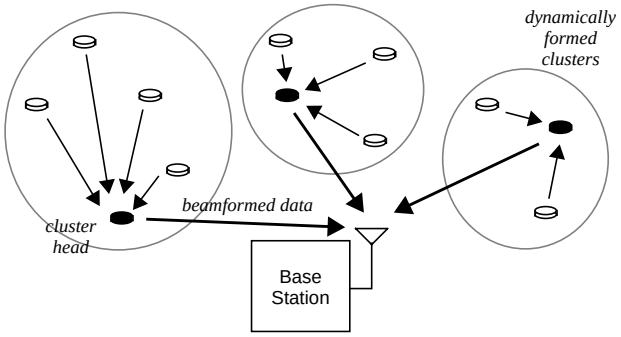

Figure 1. Operation of the μAMPS distributed microsensor network.

The μAMPS sensor node will be designed as a highly integrated solution. However, to demonstrate our enabling technologies immediately, early prototypes will be integrated with commercial, off-the-shelf components. Our first prototype is built around the StrongARM SA-1100 microprocessor. In the following sections, we overview energy scalability in our microsensor architecture and demonstrate how this scalability can be leveraged by dynamic voltage scaling on the SA-1100.

2. Energy Scalability in Microsensor Networks

Attaining months or years of useful life from a distributed microsensor network requires system design for *power awareness*. An essential component of power aware systems is the capability to make intelligent trade-offs between energy and quality.

In algorithms where additional computation incrementally refines a result, the energy of computation can be traded off for quality. There are many examples in microsensor networks. For instance, transmitted data can be encrypted with a key of varying length, allowing tradeoffs between computational energy and the security of the transmission [5]. A beamforming algorithm can fuse data from a varying number of sensors, with the mean-square error of the result decreasing as data from more sensors are combined. The number of taps in a FIR filter can be varied; longer impulse responses yield a more powerful (and power-hungry) filter.

So far, we have tacitly assumed that energy can indeed scale gracefully with these variations. We now consider how a variable voltage microprocessor can actually reduce the energy consumed when during low workload periods through dynamic voltage scaling.

The energy required for a computation on a static CMOS device is [6]:

$$E = C_{tot} V_{DD}^{\ 2} + I_{leak} V_{DD} \left(\frac{N}{f}\right)$$

with the parameters defined as follows:

C_{tot},	the total switched capacitance for the computation
N,	the number of clock cycles taken by the computation
V_{DD},	the supply voltage
f,	the clock frequency
I_{leak},	the leakage current

Variations in the quality of an algorithm appear to a processor as variations in processor utilization, which affect N and C_{tot}. When a processor is idle due to a light workload, clock cycles—and hence, energy—are wasted. Gating the clock during idle cycles reduces the switched capacitance of idle cycles. Reducing f during periods of low workload eliminates most idle cycles altogether. Neither approach, however, affects $C_{tot}V_{DD}^{\ 2}$ for the actual computation or substantially reduces the energy lost to leakage. Reducing V_{DD} in conjunction with f achieves energy savings for the actual computation. Scaling the frequency and supply voltage together results in a nearly quadratic savings in energy and reduces leakage current.

Dynamic voltage scaling (DVS) is the active adjustment of V_{DD} and f in response to fluctuations in a processor's utilization. A voltage scheduler, running in tandem with an operating system's task scheduler, can adjust voltage and frequency in response to *a priori* knowledge or predictions of the system's workload. DVS has been successfully applied to custom chip sets [5][7]. In the next two sections, we demonstrate an implementation of DVS on an unmodified, off-the-shelf processor.

3. Dynamic Voltage Scaling Implementation

We have demonstrated dynamic voltage scaling capabilities on the SA-1100, the processor chosen for the µAMPS wireless microsensor prototype. A DC-DC converter circuit with a digitally adjustable voltage delivers power to the SA-1100 core and is controlled by a multithreaded, power-aware operating system.

The SA-1100 operates at a nominal core supply voltage of 1.5 Volts and is capable of on-the-fly clock frequency changes in the range of 59 MHz to 206 MHz. Each frequency change incurs a latency of up to 150µs while the SA-1100's on-board PLL locks to the new frequency. The SA-1100 is a completely static component [8] and thus facilitates energy savings with DVS as discussed above.

A PCB containing a custom DC-DC converter circuit provides a dynamically adjustable voltage to the SA-1100's core. Figures 2 and 3 illustrate the operation of this circuit. A buck regulator composed of discrete components is driven by a commercial step-down switching regulator controller. This controller is programmed with a 5-bit digital value to regulate one of 32 voltages between 0.9 and 2.0 Volts. Our operating system running on the SA-1100 commands the core voltage as a 5-bit digital value that is passed to the regulator controller. External programmable logic between the SA-1100 and the regulator controller prevents the regulator from delivering a voltage beyond the SA-1100 core's rated maximum.

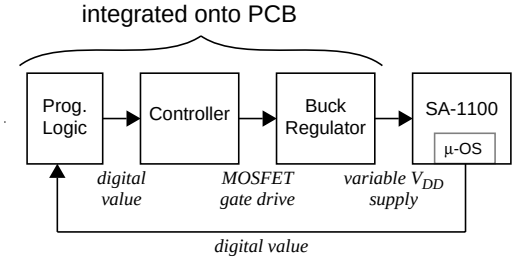

Figure 2. Overview of the adjustable DC-DC converter.

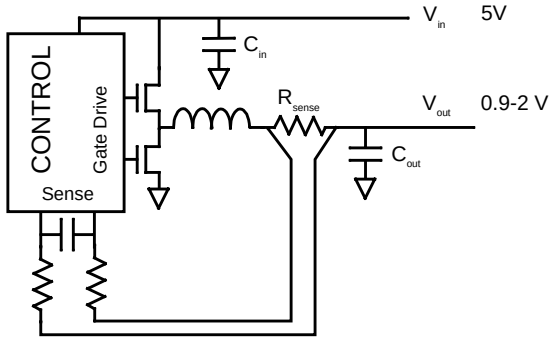

Figure 3. Simplified schematic for the buck regulator.

Our power-aware operating system is based on the Cygnus eCOS kernel [9]. This "µOS" supports preemptive multitasking, allowing threads such as data fusion, data packetization, network protocol handling, and voltage scheduling to operate simultaneously within a microsensor node. The µOS resides within a bootable instruction ROM.

In our initial implementation, the µOS monitors load on the processor and adjusts the clock frequency and supply voltage together to meet throughput requirements imposed by the tasks. Though we expect the majority of throughput requirements and real-time deadlines on a microsensor network to be known *a priori*, we plan to incorporate more sophisticated load prediction algorithms [10][11] for more optimal voltage scheduling.

4. Experimental Results

Figure 4 below demonstrates dynamic voltage adjustments commanded by the µOS. The oscilloscope trace measures the voltage supplied to the SA-1100 core over a thirty second interval. Our µOS coordinates these changes to the core voltage with simultaneous changes to the clock frequency. The system continues to run when the processor is moved from 59 MHz, 0.9 V to 206 MHz, 1.5 V, the largest changes supported by our hardware.

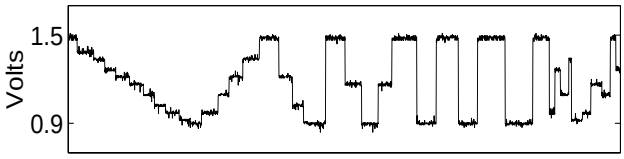

Figure 4. Scope trace of dynamic voltage adjustments. The µOS also adjusts the clock frequency for each voltage change.

We now demonstrate DVS on our system by presenting two example scenarios, each with and without voltage scaling. The first scenario trades energy for computational latency on a fully loaded processor, and the second trades energy for output quality on a constant-throughput FIR filter. All energy measurements below are based on measured current into the regulator circuit and therefore include losses in the variable-voltage regulator.

Figure 5 characterizes the energy consumption of the SA-1100 processing core versus the latency of computation (inverse of clock frequency) and the supply voltage. The SA-1100 is running the µOS with multiple threads and no idle time. The absence of a bar indicates that the SA-1100 did not function at a particular frequency-voltage combination, meaning that our results can also be interpreted as a three-dimensional shmoo plot.

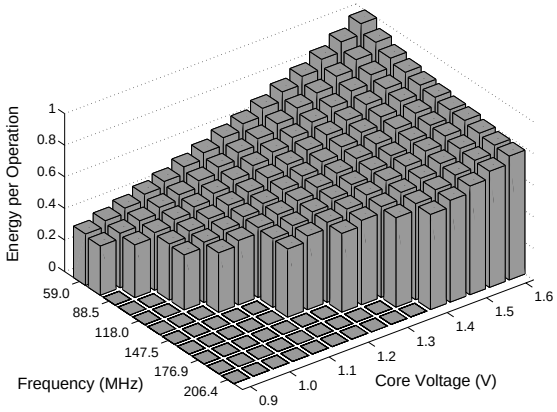

Figure 5. Measured energy per operation vs. frequency and supply voltage.

As stated in section 2, reducing the clock frequency without altering the voltage does not decrease the energy per operation. Supply voltage scaling, on the other hand, can reduce the energy cost of an operation dramatically. An ideal system with DVS would operate at the lowest voltage possible for each supported frequency. In terms of the above figure, the system would operate along the "cliff" along the diagonal of the graph, with an additional safety margin in voltage.

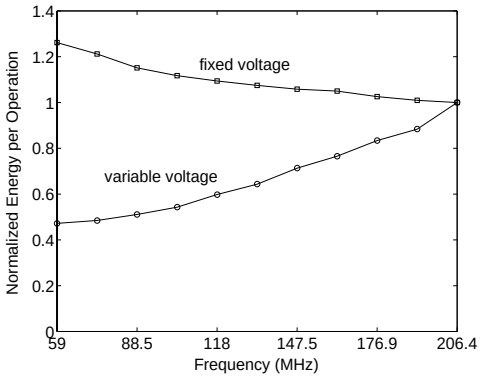

Figure 6. Energy savings realized by voltage scaling.

Figure 6 compares the energy consumption of the system with and without voltage scaling. The upper curve, representing a fixed voltage, is the cross-section along $V_{dd} = 1.5$ from the previous graph. The energy per operation is actually *greater* for lower frequencies since the energy lost to leakage is distributed over fewer computations. The lower curve, representing a voltage that scales appropriately with frequency, takes a diagonal cross-section of frequency-voltage pairs. Our initial system demonstration of voltage scaling exhibits up to 60% in energy savings over fixed-voltage approaches. A regulator control-

ler optimized for low loads would provide immediate improvements; all of the commercial options at the time were designed for a higher voltage and much higher average currents than those typically used by the SA-1100.

To evaluate the energy consumption of energy-scalable algorithms on our system, we run a FIR filter under our µOS, adjusting the number of filter taps to alter the quality of the filter. Figure 7 shows the effect of impulse response length on the stopband rejection of a bandpass filter.

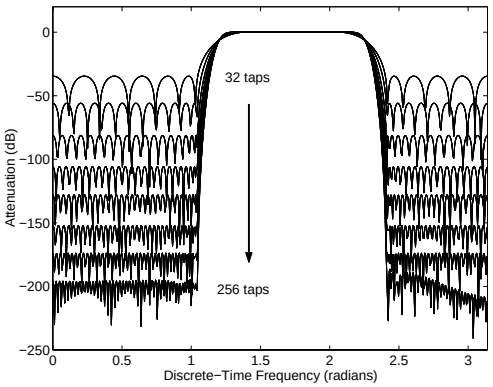

Figure 7. Effects of impulse response length on the quality of a FIR filter response.

The FIR filter is run at a constant throughput, and the impulse response length is varied. The µOS dynamically adjusts the core voltage and frequency to meet the throughput requirement with the lowest possible energy. Figure 8 demonstrates how DVS enables this energy *vs.* quality tradeoff, providing a comparable energy savings to the prior example in figure 6. The distinct series of steps in both plots are due to the eleven discrete frequencies supported by the SA-1100.

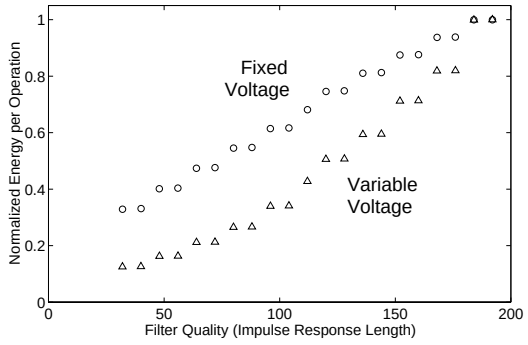

Figure 8. Energy vs. quality comparison for FIR filtering.

5. Conclusion

We have demonstrated the energy savings offered by dynamic voltage scaling on an unmodified commercial microprocessor using an adjustable DC-DC regulator and a power-aware µOS. Energy-scalable algorithms running under our initial implementation consume up to 60% less energy with DVS than with a fixed supply voltage. Refinements to the regulator circuit and the addition of voltage scheduling algorithms to the µOS will further increase energy savings. Our DVS-enabled system, when incorporated into a complete prototype sensor node for µAMPS, will enable the energy-quality tradeoffs inherent to power-aware algorithms for distributed microsensor networks.

Acknowledgments

The authors wish to thank Manish Bhardwaj for enlightening discussions on energy scalability and assistance with the FIR filter experiments, and Cygnus Solutions for providing the eCos microkernel. This work was supported in part by DARPA.

References

[1] K. Bult et al., "Low Power Systems for Wireless Microsystems," *Proc. ISLPED 1996*, pp. 17-21.

[2] A. Wang, W. Heinzelman, and A. Chandrakasan, "Energy-Scalable Protocols for Battery-Operated Microsensor Networks," *Proc. SiPS '99*, Oct. 1999, pp. 483-492.

[3] A. Chandrakasan et al., "*Design Considerations for Distributed Microsensor Systems*," *Proc. CICC 1999*, pp. 279-286.

[4] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," *Proc. HICSS 2000, Jan. 2000.*

[5] J. Goodman, A. Dancy, and A. Chandrakasan, "An Energy/ Security Scalable Encryption Processor using an Embedded Variable Voltage DC/DC Converter," *Journal of Solid-State Circuits*, Vol. 33, No. 11, Nov. 1998, pp. 1799-1809.

[6] A. Sinha, "Energy-Scalable Software," S.M. Thesis, Department of EECS, Massachusetts Institute of Technology, February 2000.

[7] T. Burd, T. Pering, A. Stratakos, and R. Brodersen, "A Dynamic Voltage Scaled Microprocessor System," *Proc. ISSCC 2000*, pp. 294-295.

[8] DIGITAL Semiconductor, *DIGITAL Semiconductor SA-1100 Microprocessor Technical Reference Manual*, 1998.

[9] "eCos Documentation," *http://sourceware.cygnus.com/ecos/docs.html*

[10] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for Reduced CPU Energy," in *Low Power CMOS Design*, A. Chandrakasan and R. Brodersen, eds., 1998, pp. 177-187.

[11] T. Pering,. et al., "The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms," *Proc. ISLPED 1998*, pp.76-81.