AFRL-IF-RS-TR-2003-109
Final Technical Report
May 2003

POWER AWARE WIRELESS MICROSENSOR NETWORKS

Massachusetts Institute of Technology

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. J872/00

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2003-109 has been reviewed and is approved for publication.

APPROVED: Robert M. Husnay

ROBERT M. HUSNAY
Project Engineer

FOR THE DIRECTOR:

WARREN H. DEBANY JR., Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	MAY 2003	Final May 00 – May 02

4. TITLE AND SUBTITLE
POWER AWARE WIRELESS MICROSENSOR NETWORKS

5. FUNDING NUMBERS
C - F30602-00-2-0551
PE - 62301E
PR - J872
TA - 37
WU - C1

6. AUTHOR(S)
Anantha P. Chandrakasan

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge Massachusetts 02139-4307

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFGC
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2003-109

11. SUPPLEMENTARY NOTES
AFRL Project Engineer: Robert M. Husnay/IFGC/(315) 330-4821/ Robert.Husnay@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***
This effort developed power aware technologies and implemented an acoustic sensor network capable of performing audio direction finding. Strong ARM based acoustic sensor nodes were built and demonstrated at the Army Research Laboratory's Aberdeen MD test Facility in which they performed Line of Bearing measurements. Web based tools were developed to help StrongARM developers evaluate the power consumption of these systems.

Wireless protocols were also developed to optimize the power efficiency of the sensor network. An API framework was developed, in addition to a simulator that allows the end-user to measure and improve the power-awareness of data-gathering microsensor networks. This tool set allows a designer to develop and optimize protocols and algorithms in simulation, then rapidly transfer the design to the sensor hardware.

14. SUBJECT TERMS
Power Efficiency, Sensor Networks, Acoustic Tracking

15. NUMBER OF PAGES
38

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures and Tables

1 Introduction

In recent years, the idea of wireless microsensor networks has garnered a great deal of attention and interest. Distributed wireless microsensor networks consist of hundreds to several thousands of small sensor nodes scattered throughout an area of interest. Each node individually monitors the environment and collects data as directed by the user, and the network collaborates as a whole to deliver high quality observations to a central base station. The large number of nodes in a microsensor network enables high-resolution, multi-dimensional observations and fault-tolerance that are superior to more traditional sensing systems.

Four characteristics of the microsensor application domain are of particular interest to the node designer.

- Nodes operate with an extremely low duty cycle. Events of interest to the network may be spaced hours to days apart, meaning that nodes can be idle over 99% of the time.
- Data is sampled and transmitted at rates of bits to kilobits per second.
- Signal processing can occur within the network. For instance, nodes can aggregate multiple streams of data from adjacent nodes using beamforming algorithms.
- Performance demands on the node are variable and unpredictable before deployment. For instance, local variations in node density can modulate the distance between adjacent nodes, creating variations in the nodes' required radio transmission power. The ambient noise level of the environment will change, creating variations in the amount and type of signal processing required within the network.

Power-aware design is an important consideration for a variety of emerging applications. It is highly desirable for a system to have the ability to adjust dynamically to changing conditions, i.e., to be energy-efficient over a wide range of scenarios while still meeting the quality of service demands of the end-user. Our overall goal is to improve energy-efficiency by ensuring that a system is capable of a wide range of scalability. By providing a multitude of hardware knobs, energy-scalable software can adapt energy consumption to meet a given quality specification. In Phase I, we were able to demonstrate more than 20x system level energy scalability for real-world application scenarios. A distributed microsensor network is used as an application driver.

The MIT µAMPS (Adaptive, Multi-Domain, Power-Aware Sensors) project has developed power aware hardware, protocol, and algorithm building blocks for microsensor networks. A primary focus of the µAMPS project has been the design of a hardware substrate upon which complete microsensor applications can be developed and demonstrated.

2 Approach

Our approach is to provide energy-agile fabrics, where hardware and software knobs allow for system level energy-efficiency while still achieving the desired quality of ser-

vice and latency. We have developed a system-level power aware design methodology that allows for optimization at all levels of design ranging from communication protocols and OS-directed power management, to scalable algorithms, architectures and circuits. A system with a multitude of hardware and software knobs provides the end-user a high degree of freedom for optimal energy-management across a variety of operating scenarios.

We have developed a systematic approach for measuring and enhancing the power awareness of integrated systems. Power awareness is measured rigorously by quantifying the diversity of operating scenarios and by comparing a system to the maximally power aware solution. The metric has a broad applicability from low-level logic and memory modules to digital and analog architectures, and even networking protocols.

We have completed design of our second-generation wireless microsensor node using scenario-agile computation and communication fabrics. The node has been instrumental for demonstrating our approaches to advanced power-aware design concepts. The node has been designed using commercial, off-the-shelf (COTS) technology, and is primarily based around the StrongARM processor. Additional hardware allows for graceful degradation between performance and energy dissipation through dynamic voltage scheduling. Communication modules are also energy scalable through the use of reconfigurable fabrics for agile digital baseband functions and variable power amplifier hardware for optimal transmit power dissipation for ad hoc wireless networking. The node also allows for static power dissipation control, which for low duty cycle systems can dominate the overall power. Appropriate shut-down hooks have been added to ensure reduction of processor leakage and radio startup costs.

The node has been an instrumental tool for advancing our understanding of the physical nature of wireless sensors. We have utilized the node to validate energy and performance models, to characterize the operating and power consumption characteristics of software running on a processor, to test the limits of communication performance, and to parameterize large-scale simulations of nodes communicating in a network. The node has also proven effective in real-world scenarios (e.g., data collection at the Aberdeen Proving Grounds) and real-world applications such as Line of Bearing estimation.

The protocol architecture is central to minimizing the energy dissipation associated with data gathering from massively distributed sensors. Our cross-layer protocol architecture exploits application-specific information to achieve an order of magnitude improvement in energy efficiency over conventional approaches. Protocol designs consider both small, local regions and entire thousand-node networks. The protocol that governs local sensors is based on clustering. The clustering protocol exploits spatial correlation among adjacent sensors to reduce the amount of sensor data that is routed through the network. Intelligent partitioning of the computation across the network can yield further energy-efficiency. System partitioning and clustering have been demonstrated through an acoustic tracking application from ARL. For large-scale networking of thousands of nodes, formal derivations of bounds on the transmission distance for minimal energy dissipation yield insight

into new protocols for optimal multi-hop routing strategies. Protocols that avoid large routing tables and addressing, but are based on neighboring node information (e.g. distance, energy) are shown to be more energy-efficient and highly adaptable for sensor networks.

A framework has been developed that includes an application programming interface (API) and a simulator that allows the end-user to measure and improve the power awareness of data-gathering microsensor networks. A power aware API allows the end-user to easily design large-scale sensing applications and to specify desired performance requirements, without necessarily being exposed to the low-level energy knobs in the hardware. A complementary simulator will accurately model the energy consumption and operation of the sensor application. The complete toolchain allows a designer to develop and optimize protocols and algorithms in simulation, and transfer the final design to the sensor hardware for immediate functionality.

3 Node Implementation

3.1 Architecture

The μAMPS node is designed for flexibility. Since μAMPS is not dedicated to any one particular microsensor application, it was made easy to equip the node with virtually any kind of sensor. However, the node is optimized for acoustic sensing, since acoustic sensors are easy to test in the laboratory.

A block diagram of the μAMPS node is shown in Figure 1. The sensor block consists of a microphone, amplifier, anti-aliasing filter, and analog-to-digital converter. Power control signals from the processor allow each of the sensor components to be shut down when not needed. Although the sensor is only active when all of its components are powered, the ability to power down only selective portions of the sensor circuitry creates additional power scalability by creating standby modes where power consumption is intermediate between the full active and full shutdown states, but the delay required to transition back to the active state from idle is less than the delay to transition to the active state from the full shutdown state.

The processor block consists of a StrongARM microprocessor, along with low-power static RAM and a flash ROM. The StrongARM processor was chosen because of its high performance/power ratio, and its built-in variable frequency (59-206MHz) core clock generator. Varying the processor clock speed, in real time, is an important part of the μAMPS power-awareness strategy. In the μAMPS node, the processor voltage is varied along with the clock frequency. This is accomplished with a special dc/dc converter built into the processor board. Reducing the voltage applied to the processor core to the lowest level possible to support the current operating frequency increases power savings at low clock frequencies by reducing both switching and leakage currents.

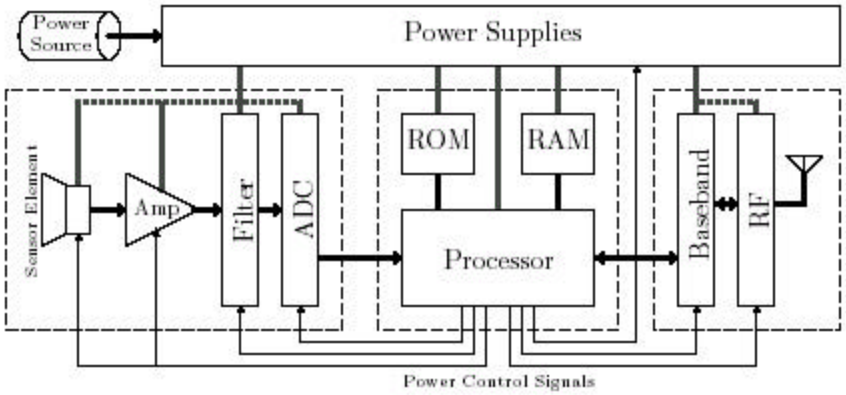

FIGURE 1. Block diagram of the mAMPS node.

The radio block is subdivided into a digital baseband component (implemented on an FPGA) and an RF component (implemented with discrete components and an integrated radio IC). The digital component is responsible for encoding, decoding, and error detection/correction, as well as controlling the timing of transmitter and receiver according to the TDMA scheme employed by the network protocol. The RF circuitry consists of a 2.4GHz radio, a VCO, low noise amplifiers, and a variable power amplifier, an antenna. As with the sensor circuitry, power to the various components of the radio is controlled by the processor, allowing components to be shutdown when idle.

Figure 2 shows the physical architecture of the mAMPS node. The node consists of a stack of three or four printed circuit boards. Each board is 55mm square. A system connector, present on each board, links the boards electrically, creating a common bus of control signals between the boards. The top-most board contains the radio, including the RF circuitry and the FPGA used for digital coding and decoding. The second board contains an Intel

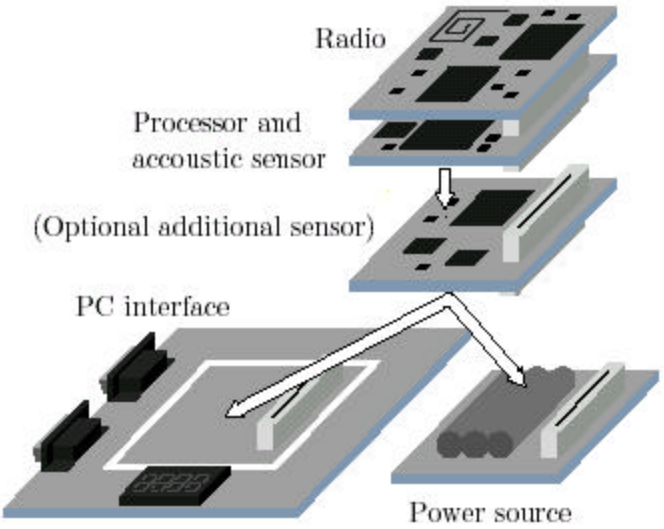

FIGURE 2. Physical architecture of the node.

StrongARM processor, and associated RAM and flash ROM. Also on the processor board are an acoustic sensor (microphone, amplifier, and analog-to-digital converter) and a collection of dc/dc power converters that service the entire node. The optional third board down in the stack is an additional sensor module, to replace the acoustic sensor on the processor board. The *m*AMPS node can be easily adapted to different applications by designing an appropriate sensor board. The bottom board in the stack contains the power source. For a typical node, this consists of a battery pack, containing four AAA cells.

To convert a typical sensor node into a basestation node, the battery board is replaced with a PC interface board. This board provides standard connectors (RS-232 and USB) for interfacing to a larger computer. The basestation board also contains voltage regulators that allow the basestation node to be powered from ac line power using a 6-12V plug-in wall transformer. Special connectors on the interface board allow easy connection of a logic analyzer to facilitate debugging of the node.

Figure 3 shows a photograph of the complete node.

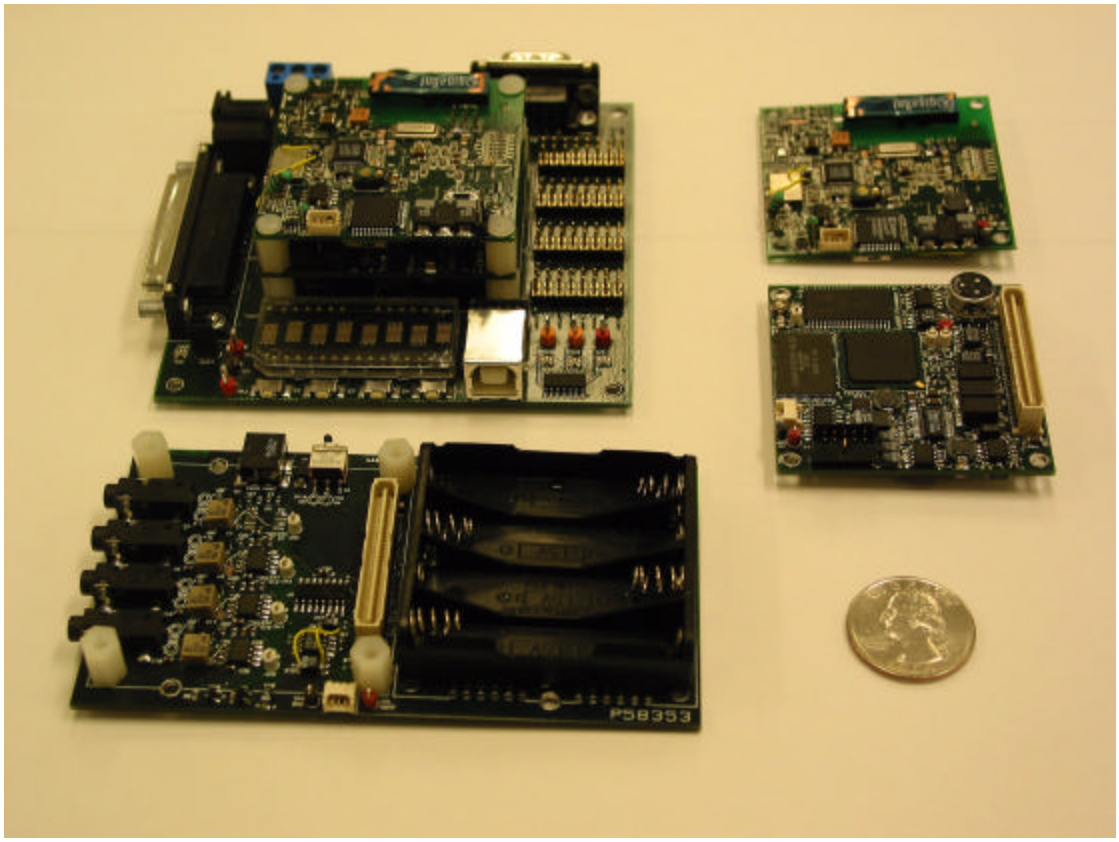

FIGURE 3. Variations on the mAMPS node. (Upper right) Radio board. (Lower right) Processor board. (Bottom) Optional, enhanced sensor and battery adapter board for four-channel acoustic sensing. (Upper left) mAMPS base station, consisting of processor and radio boards stacked atop a PC interface board with serial and USB connectivity.

3.2 Operating Modes

During the initial design of the μAMPS node, four standard modes of operation were identified. These are as follows.

- **Sleep:** In sleep mode, almost none of the node's systems are active, resulting in absolute minimal power consumption. The node awakens after a preset time, or in response to an event in its environment.
- **Sense:** In this mode, the node records data from its sensors. The data may be buffered locally, or may be simultaneously transmitted to surrounding nodes.
- **Relay:** *m*AMPS nodes form a multi-hop ad-hoc network. In this mode, a node listens for incoming radio packets and retransmits them when necessary.
- **Aggregation:** The node collects data from two or more sources (possibly including its own sensor) and performs an analysis on the composite data. The results of this analysis are passed on to other nodes. An example analysis would be line-of-bearing estimation based on acoustic data from multiple microphone locations.

Minimizing sleep mode power consumption is critical due to the low duty cycle of the node. Collecting data requires both the sensor and the processor to be active. In the current, COTS-component based node implementation, the processor is used to initiate periodic A/D conversions. In a future system-on-a-chip version of the node, a DMA engine will relieve the processor of this task, reducing the power required for data collection. The processor will still be necessary, however, to perform post-processing or compression of the raw data. In relay mode, the radio and processor are both active, as the processor shuffles data packets to and from the radio. Future implementations will include a hardware protocol processor that will again eliminate the need to use the general purpose microprocessor in this mode. Finally, aggregation is most power consuming of the node's operating modes, making extensive use of the radio, processor, and possibly even the sensor.

3.3 Power Management Techniques

Each component of the μAMPS node has been optimized to minimize standby power consumption and provide maximum power scalability. Figure 4 illustrates the power scaling controls implemented. Most node components can be independently shut down. The only exceptions are the microprocessor's flash ROM and static RAM, both of which inherently draw very little power in their idle state. Independent shut-down controls have been implemented even for highly dependent components, because this creates intermediate standby states from which a quick return to full active status is possible. For example, the anti-aliasing filter is not useful when the analog-to-digital converter is in shutdown, but by leaving the filter on while the ADC is shutdown, it is possible to power up the ADC and quickly resume recording data samples without waiting for the switched-capacitor filter to stabilize. Of course, if the ADC is to be shut down for a long period of time, then it is advantageous to also shut down the filter.

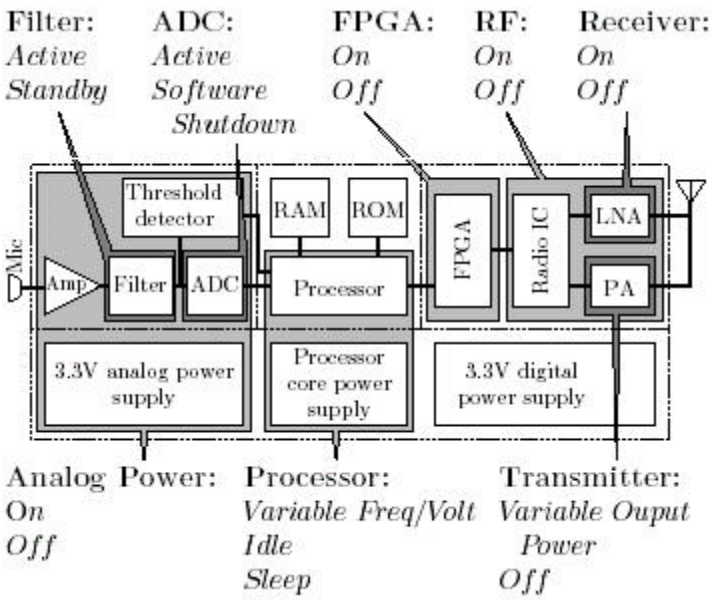

FIGURE 4. Power-scaling controls on the mAMPS node. Most node components can be shut down. Many components incorporate more complex power controls which allow gradual power scaling.

3.4 The Processor Board

A block diagram of the circuitry contained on the processor board is shown in Figure 5. The processor board contains a 32-bit microprocessor and its associated ROM and RAM, a collection of dc/dc converters, a small acoustic sensor, and assorted I/O interface circuitry, including RS-232 drivers and a USB port.

3.4.1 Microprocessor and Memory

The *m*AMPS node is based on an Intel StrongARM SA-1110 processor. The StrongARM processor is a good match for the design goals of the *m*AMPS project: it provides a significantly above average balance of computational performance and power consumption (235MIPS and <400mW at 206MHz), a programmable PLL for core clock generation (59-206MHz), and many important peripherals (three UARTs, SPI interface, USB peripheral controller) on-chip. Its fully-static CMOS design permitted the development of a dynamically variable core power supply, which further reduces the power consumption of the CPU core by up to 60%.

The SA-1110 provides 28 general purpose I/O (GPIO) pins, which can be individually be configured as inputs or outputs, and can even generate interrupt signals. The *m*AMPS node primarily uses these pins for power management signals. GPIO pin 0 serves as a general purpose interrupt request line (IRQ) for the node. The IRQ input present in the main system connector, so any interrupts can be signaled by any of the node's board.

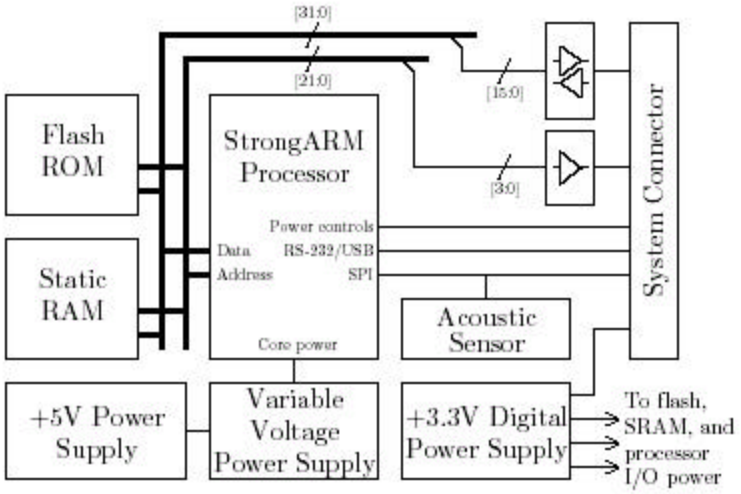

FIGURE 5. Block diagram of the processor board

3.4.2 Power Supplies

The *m*AMPS node was designed to operate from a wide range of battery voltages, allowing flexibility in the choice of batteries used to power the node and ensuring that the node can continue to operate at decreasing battery voltages as the battery is discharged. The intended rechargeable battery configuration for the node is two lithium ion batteries in series. The processor board carries four different power supplies which convert the variable voltage from the battery into multiple regulated power busses used by various node components. The three main supplies provide +3.3V for all digital logic on the node, +0.9-2.0V for the StrongARM core, and +3.3V (with low noise) for the analog circuitry in the sensor portion of the board. A fourth supply provides the +5V needed by the StrongARM core supply controller chip.

Because the StrongARM microprocessor is the second largest power consumer on the node, the design of its power supply is especially critical. The microprocessor core supply is generated by a Maxim buck-mode dc/dc controller (Figure 6). The MAX1717 includes a built-in, 5-bit digital-to-analog (DAC) converter which can be used to program the out-

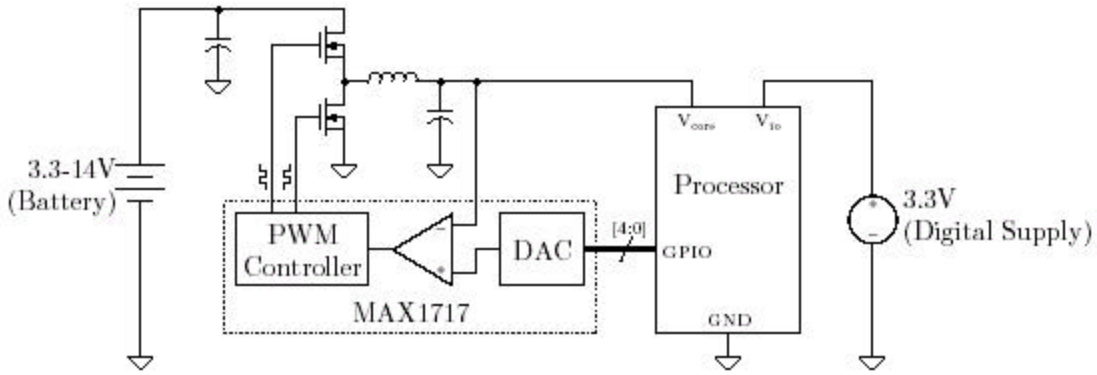

FIGURE 6. Microprocessor core supply

put voltage from 0.9V to 2.0V. The DAC inputs are wired to GPIO pins on the StrongARM processor, giving the processor dynamic control over its own supply voltage. Figure 7 show the energy/cycle as a function of frequency and supply.

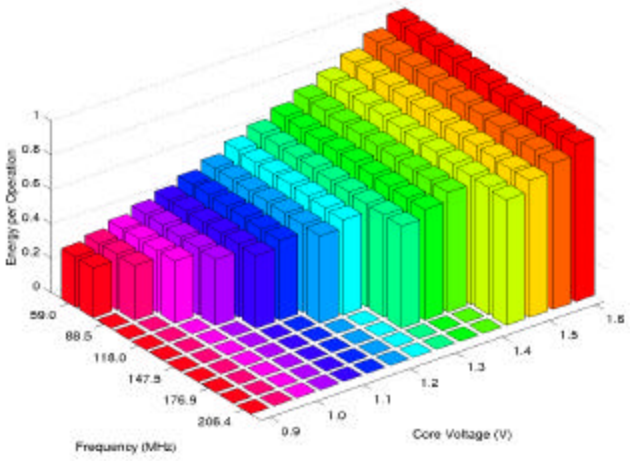

FIGURE 7. Dynamic voltage scaling on the SA-1100.

3.4.3 Acoustic Sensor

The default acoustic sensor for the µAMPS node occupies a corner of the processor board and consists of an electret microphone, variable-gain amplifier, an analog-to-digital converter and anti-aliasing prefilter, and a threshold detector. The sensor was designed for a 2kHz sampling rate, for analyzing sounds from 20Hz to 1kHz.

The microphone, amplifier, filter, and analog converter require very little power-no more than a few milliwatts, but in order to trigger conversions and examine the output of the analog-to-digital converter, the processor must be active, which consumes almost 100mW even at low clock frequencies. Given the low duty cycle of the node, it is unacceptable to operate the processor while the node is waiting for the microphone to hear something. The addition of a fully-analog threshold detector allows the processor to sleep whenever no external stimulus is present, but to be awakened by an interrupt if the ambient noise level reaches a programmable level.

This threshold detector consists of a peak detector and a voltage comparator. The peak detector is implemented from a diode, capacitor, resistor, and op-amp, as shown in Figure 8. The output of the peak detector is fed into a voltage comparator. The inverting input of the comparator is driven by a digital potentiometer which is wired as a voltage divider, effectively implementing a 5-bit DAC. The digital potentiometer makes the comparator trip point programmable. the voltage comparator is connected to a StrongARM. The output of the voltage comparator is connected to a StrongARM I/O pin, which can be configured to generate an interrupt, waking the processor from sleep when a sufficiently loud sound is detected by the microphone.

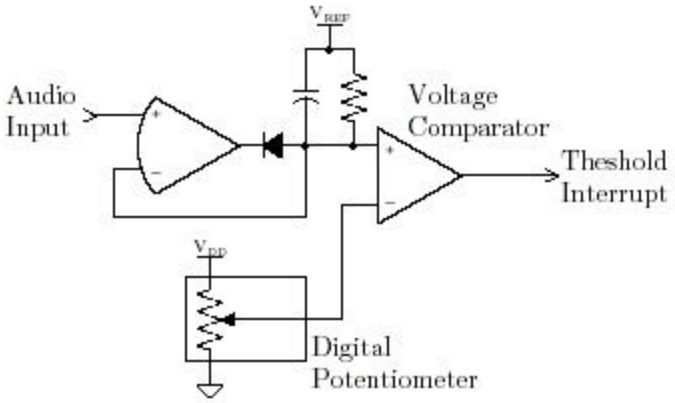

FIGURE 8. Sensor circuitry.

3.5 Radio Board

3.5.1 Overview

As with most commercial transceivers, the µAMPS radio board operates in the Instrumentation, Scientific, and Medical (ISM) band from 2.4GHz to 2.48GHz. The FCC requires power emissions in this band be below 20dBm. This limitation in power output does not affect the specifications for wireless sensor nodes, because wireless sensor nodes are not required to transmit very far distances. A block diagram of the radio is shown in Figure 9

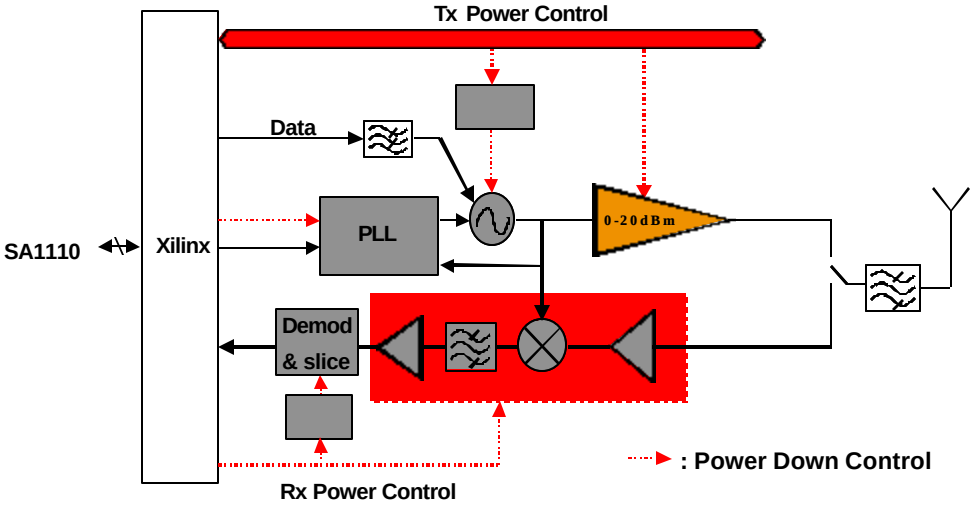

FIGURE 9. Block diagram of power aware radio.

3.5.2 Transmit Path

A stream of binary data to be transmitted is shaped into a Gaussian form by a 3rd order low-pass Gaussian filter, with cutoff frequency set at 500kHz, so that the bit rate is set to

1Mbps. The Gaussian data is fed to a second input pin of a VCO that is embedded in a PLL. This PLL is locked at half of the transmit frequency. A frequency doubler doubles the frequency when the signal is close to being amplified by the power amplifier. This architecture avoids the problem commonly known as “load pulling” or “VCO pulling” from the power amplifier. Since the Gaussian data is directly fed into the PLL, the RF frequency is said to be directly modulated by the data. In this case, the loop of the PLL is closed at all times, to allow the option of large packet lengths to be transmitted, if necessary. The power amplifier can be controlled to operate at one of six distinct levels of power gain, varying the output power between 0dBm and 20dBm. The antenna is omnidirectional, and sits flush against the board.

3.5.3 Receive Path

Electromagnetic waves received by the antenna are fed into the matching network of the LNA. The signal is amplified by 10-20dB, and then passed into a band-pass filter for image rejection, since the receiver architecture is heterodyne in design. The same PLL that was used in the Tx path is also used in the Rx path. While the receiver is active, the PLL is set to generate a frequency that is 110.6MHz lower than the RF frequency that is desired to demodulate. Both of these frequencies (from the LNA and from the PLL) are sent to a mixer, where sum and difference frequencies are generated. The IF frequency is set at 110.6MHz, which is the frequency that is generated by the difference term of the mixer output. This signal is filtered by a sharp saw-filter to clean up side band interferers, and sent to an intermediate frequency amplifier. This amplifier typically has gains of 50-70dB. The IF frequency is finally sent to a discriminator, where ones and zeros are discriminated via a tuned circuit and mixer, and sent to a bit-slicing circuit to recover the bits.

3.6 µAMPS radio Power Aware Features

The importance of power-awareness in the radio design is primary if extending the lifetime of an entire network is of interest. A combination of low-power circuits and low-power system-design are the basis of power-aware circuit design. The µAMPS radio can be generalized into three states that consume energy: Idle, Tx, and Rx. The Off state consumes virtually no power (a few microwatts).

The µAMPS radio uses the idle state as a reliable state-transition mode and a low-power radio configuration bit-setting mode. In the Rx state, the µAMPS radio is set to receive. When the radio receives data, it analyzes the receive-signal-strength-indicator to determine if additional bit-recovery circuitry is needed to be switched on to recover the bits. In the Tx state, because the radio board is built with a single scalable power amplifier, the radio has six levels of output power to radiate: 0dBm, 3dBm, 5dBm, 10dBm, 15dBm, and 20dBm. The transmit power states are set dependent on how far of a hop a particular node would like to make, and allow flexibility in the design of the MAC layer for additional power-saving features to incorporate multi-distance data-hopping. From 0dBm to 20dBm, the radio is theoretically able to transmit in a 10 meter to 100 meter radius. The chart in

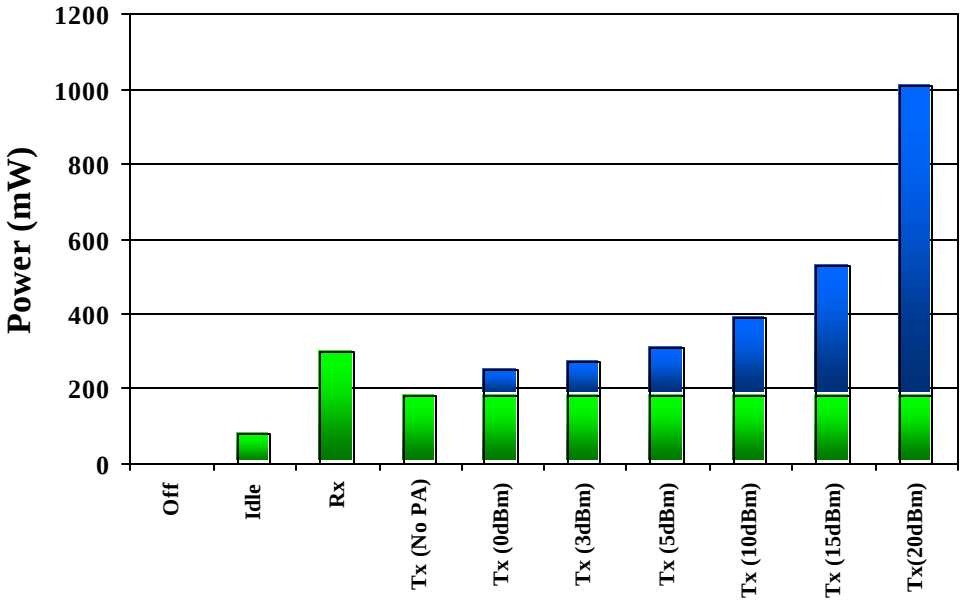

FIGURE 10. Power-aware states of the mAMPS radio

Figure 10 shows the corresponding power consumption of each of the power-aware states of the radio.

3.6.1 Power Analysis on Node to Node Radio Link

To elicit the impact of the µAMPS node in all of its specifications, features, design motivations, and power-aware hooks with regard to wireless networks, a node-to-node link is taken and analyzed with respect to the Energy/bit system-level performance metric. For a wireless node-to-node link, Energy/bit can be calculated according to the following equation.

$$E_{bit} = \frac{2P_{idle}t_{pll} + (P_{tx} + P_{rx})\left(\frac{B_{bits} + B_{sync}}{BPS}\right)}{B_{bits}} \quad (1)$$

Using the values in Table 1, the plots in Figure 11 were made.

Figure 11 shows that there comes a point where the energy/bit metric can no longer be reduced, because the "active" portion of the equation dominates over the fixed-cost component. The "active" portion of the equation lies in the number of bits transmitted. The fixed-cost portion lies in the fixed-cost associated with each time the radio begins to send a packet. This includes the lock time of the PLL and the time it takes to send an 80 bit header/sync in the preamble of the transmit data. It is also important to note that the "corner" bits/transmit-cycle is key in designing packet-lengths. For optimal operation, it is desirable to operate at the lowest Energy/bit section on the graph to squeeze the maximum

Variable	Measured Value	Description
P_{idle}	80mW	Idle mode power
P_{rx}	300mW	Receiver power
P_{tx}	250mW to 1100mW	Transmitter power
t_{pll}	400µs	PLL lock time
B_{bits}	2^1 to 2^{20} bits	Bits transmitted
Bsync	80 bits	Bits in packet synchronization header
BPS	1Mbps	Bitrate

TABLE 1. mAMPS radio parameters

lifetime and data-gathering a network can deliver. The equation for energy per bit can be rewritten much like a transfer function is written,

$$E_{bit} = \left(2P_{idle}t_{pll} + \frac{(P_{tx}+P_{rx})B_{sync}}{BPS}\right)\frac{B_{bits}\left(\frac{P_{tx}+P_{rx}}{2BPSP_{idle}t_{lock}+(P_{tx}+P_{rx})B_{sync}}\right)+1}{B_{bits}} \quad (2)$$

with the frequency variables of a transfer function replaced by B_{bits}. In this way, it is possible to extract the "1/bit constant", or "zero" of the system. It is at this "zero" location that the optimal choice for bits/transmission-cycle resides. It is true that the energy/bit decreases asymptotically to some value, but it is important that the decay beyond the zero location is asymptotic, and thereby nominal. It would cost too much energy to transmit a million bits, or in some applications, to wait long enough to gather enough information to construct and transmit a packet of that length. Packet length should be dependent on application as well, but in this case, it is in the interest of intelligently optimizing the energy/bit ratio, only.

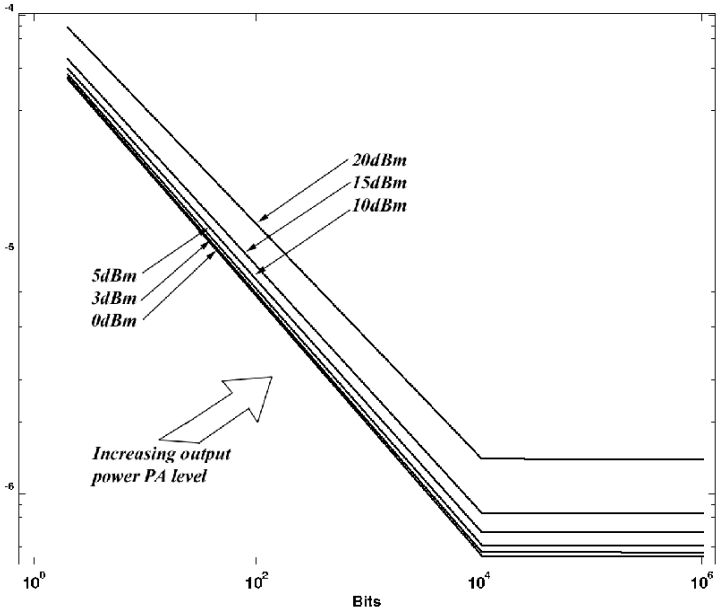

FIGURE 11. Energy/bit.

From equation (2), it is possible to see that if the 1/bit constant in front of the B_{bits} of the numerator is reduced, the bits/transmit-cycle can become smaller for the optimization of the energy/bit ratio. The lower this zero occurs, the more adaptable, reconfigurable, and power aware the radio becomes. It is apparent that changing P_{tx} or P_{rx} has little impact on the zero location, since they both appear in the numerator and denominator of the 1/bit constant. Adjusting B_{sync} and making it as small as possible is a viable option, but this variable has its limitations in required minimum size. The best option is to tackle the P_{idle} and t_{pll} variables (*BPS* is usually fixed). However, if it is desired to shift the graph in Figure 11 vertically downwards (which contributes to overall low-power operation), then a reduction P_{tx} and P_{rx} are greatly effective

3.7 Node Performance

3.7.1 Measured Energy Dissipation

Tabulating the power consumption of individual node components, as in Table 2, illustrates the degree of power scaling achieved by the μAMPS hardware. It is also important to consider the expected total power consumption of the node. This is a difficult prediction to make for a power-aware system, because the average total power consumption varies not just with the type of sensor application, but also with the characteristics of the actual signals recorded. The problem can be simplified by considering some important characteristic operating modes.

System	Mode	Power (mW)
Processor (core)	Active (59MHz)	60
	Active (206MHz)	552
	Idle (59MHz)	9
	Idle (206MHz)	71
	Sleep	0
Processor (I/O)	Active (caches on)	12
	Active (caches off)	19
	Idle	12
	Sleep	2
RAM	Active (caches off)	31
	Idle (caches on)	0
ROM	Active (caches off)	49
	Idle (caches on)	0
On-board Sensor	Active	28
	Idle	15
	Shutdown	0
Radio	Transmit (0dBm)	250
	Transmit (20dBm)	1020
	Receive	300

TABLE 2. Power consumption of individual node subsystems in all supported modes of operation.

System	Mode	Power (mW)
	Standby	24
Residual		28

TABLE 2. Power consumption of individual node subsystems in all supported modes of operation.

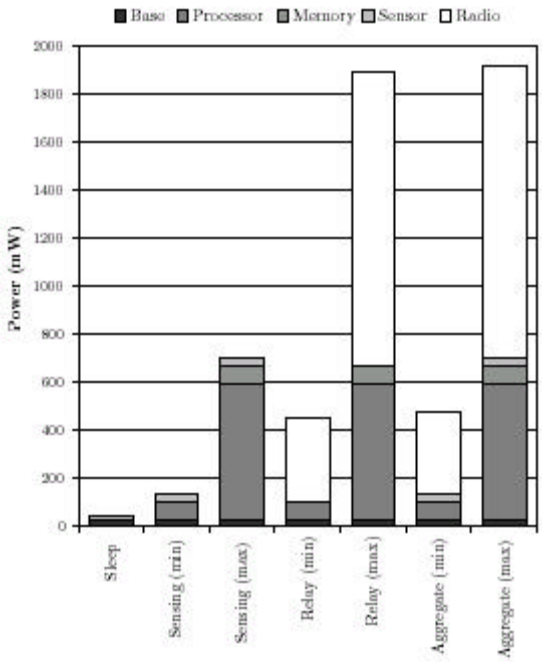

FIGURE 12. Node power consumption in the four standard operating modes. Minimum and maximum cases are shown for the sense, relay and aggregate modes, in order to illustrate the effects of processor frequency scaling and variable transmit power.

Figure 12 illustrates the total node power consumption, and its breakdown into consumption by individual components, for the four standard node operating modes. Because processor voltage scaling and variable transmitter power amplifier induce wide power consumption variations in each of the three active modes, both minimum and a maximum power variations of each of these modes are illustrated.

Power consumption ranges from 28mW in deep sleep, to almost 2W while transmitting at maximum power. A battery pack of four high-capacity AAA alkaline cells offers about 30 kJ of energy. Even allowing for a 25% efficiency loss from the battery and regulation inefficiencies, this translates into over 9 days of deep sleep, 2 days of continuous sensing, 14 hours of continuous relay, and 13 hours of continuous aggregation (assuming minimal values for each active mode). A node with a 1% duty cycle each of sensing and aggregation, at the lowest active power values, would operate for one week. With the active performance and power for sensing and aggregation set to their maximum values, lifetime drops just under 4 days. The provision of energy scalability knobs gives a great deal of control to the user, who can determine what energy and performance trade-offs are acceptable for a particular application.

3.7.2 Field Testing

In April 2002, a field trial was held at the Aberdeen Proving Ground in Maryland. Using the four-channel acoustic sensor board, three of the four microphone channels were sampled at 1ksps, and the data was again sent over a serial link to a PC. On the PC, a Java program displayed the data and recorded it to a file. Acoustic samples of military vehicles collected during this event was later analyzed using a beamforming algorithm to determine a line-of-bearing from the sensor node to the vehicle.

In June 2002, this three-microphone application was modified so that the sensor node itself performed the beamforming analysis. The sensing node transmitted the calculated line-of-bearing information to a basestation node, which communicated via RS-232 with a PC-based Java program that dynamically displayed the line-of-bearing on the PC's display. The beamforming algorithm used in this application was jointly developed by the Army Research Laboratory (from where the orginal algorithms came from), MIT, and ISI.

Figure 13 illustrates the effectiveness of the µAMPS node's power-management controls while running the beamforming application just described. While collecting data, the processor alternates rapidly between active and idle modes, because no computation is needed in between data samples. When the computationally-intensive beamforming algorithm is run, the processor remains in active mode continuously. Power consumption rises sharply when the radio transmitter is enabled, and then falls rapidly as the entire node is put to sleep.

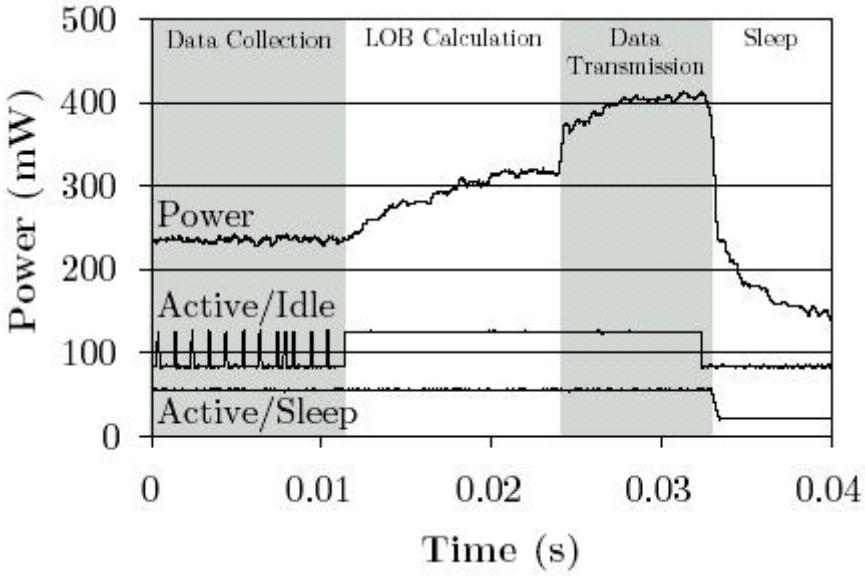

FIGURE 13. Power consumption during the execution of a beamforming application. The upper trace indicates node power consumption. The middle trace is high when the processor is active, and low when the processor is in idle mode. The lower trace goes low when the processor enters sleep mode. The node's power consumption continued to fall beyond the left edge of the graph, stabilizing at approximately 28mW. Because this graph was generated using the four-channel acoustic sensor board, rather than the processor board's built-in acoustic sensor, the numbers in this graph are slightly different than those indicated by Figure 12 and Table 2.

4 Applications for mAMPS

One prime example of a microsensor application is the use of acoustic sensors for environment monitoring. Acoustic sensors are highly versatile and can be used in a variety of applications, such as speech recognition, traffic monitoring and medical diagnosis. The sensor application which will be investigated in this article is source tracking and localization. Multiple sensors can be used to pinpoint the location of an acoustic source (e.g. moving vehicle, speaker), by using a Line of Bearing estimation technique. Source localization can be useful for traffic monitoring, speech applications and military exercises.

Figure 14 shows examples of various microsensor networks. There are many new challenges to be faced in implementing signal processing algorithms for microsensor networks. The first challenge is to model the node's energy and performance in order to design applications with large node densities. A second challenge is the high node densities, which results from a large number of microsensors within the network. The amount of sensing data will be tremendous and it will increasingly difficult to store and process the data. An efficient network protocol layer and signal processing application are needed to extract the important information from the sensor data. A last challenge is that all nodes are energy-constrained. In order to prolong the lifetimes of the wireless sensors, all aspects of the sensor system should be energy-efficient, so that algorithm and protocol design should focus on minimizing both computational and communication energy.

4.1 Sensor Application Case Study

One application for the μAMPS sensor node is vehicle tracking and localization. In this section, we will introduce algorithms which would be running locally at the acoustic sen-

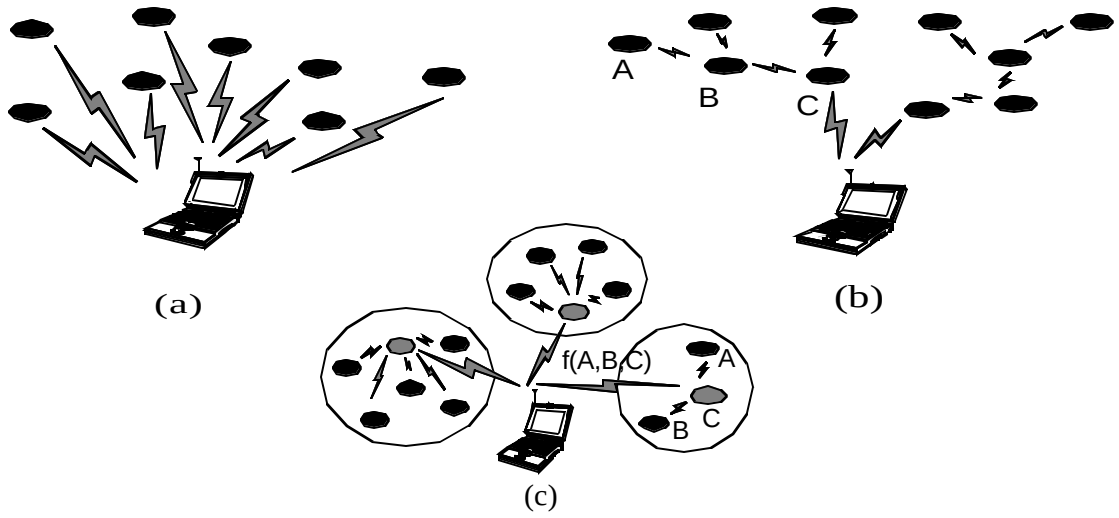

FIGURE 14. Here are three examples of microsensor networking protocols. In the first protocol, the microsensors are using direct communication (a) with the end-user. In the second protocol, the sensors transmit their data using multi-hop routing communication with the basestation (b). The third protocol is a clustering algorithm. Sensors are grouped into clusters and data is transmitted from sensors to clusterheads. Clusterheads perform data aggregation and transmit the result to the basestation.

sor cluster, and show how system partitioning can yield a more energy-efficient sensor system. Suppose a vehicle is moving over a region where a network of acoustic sensing nodes has been deployed. In order to determine the location of the vehicle, we first need to find the Line of Bearing (LOB) or direction from which sound is being detected. Figure 15 shows the scenario for vehicle tracking using LOB estimation. Multiple clusters of sensors determine the source's LOB to be the direction with maximum sound energy from their perspective. The intersection point of multiple LOB's will determine the source's location.

To perform LOB estimation, often beamforming algorithms are used. A beamformer is a spatial filter that operates on multiple sensor data in order to enhance the amplitude of a desired coherent waveform and to diminish the effects of background noise on the desired signal. By pointing the beam in the direction of the source, the signal in the desired direction is amplified while ambient noise from all other directions are diminished. Delay-and-sum beamforming is a conventional beamforming algorithm, which applies delays on the multiple sensor data before summing over all sensors. Beamforming of sensor data is beneficial in two ways. First, by scanning over multiple directions, the direction of arrival of the signal with the most signal energy relative to the orientation of the sensor cluster, can be found. This means that the direction of arrival of sound is correlated to the Line of Bearing (LOB) of the source signal. Secondly, the beamformer output in the direction of arrival will have better SNR since the effect of the ambient noise sources have been reduced and the desired signal is coherently added. Other beamforming algorithms include the Maximum Power Beamforming algorithm and the Least Mean Square algorithm.

Delay-and-sum beamforming can be performed in the frequency-domain, where delays in the time-domain correspond to phase shifts in the Fourier or frequency-domain. First the sensor data is transformed in to the frequency domain. In the digital domain, the Discrete Fourier Transform or the Fast Fourier Transform (FFT) can be used. Then phase shifts are applied before summing over all sensors. The main advantage of a frequency-domain implementation is that there is no need for oversampling the sensor output, as compared to the time-domain approach. For this application a 1024-pt. FFT is used.

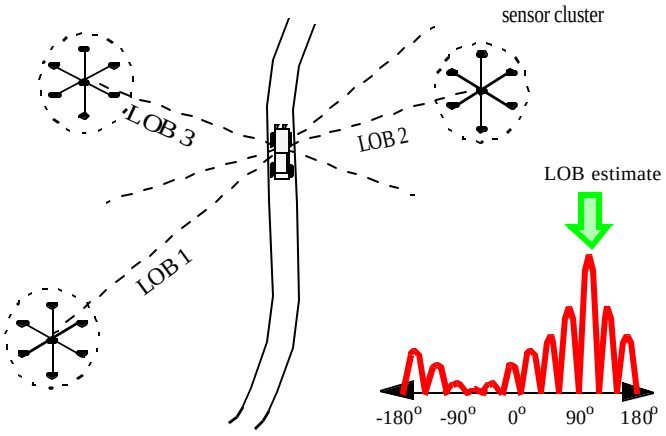

FIGURE 15. LOB estimation to do vehicle tracking

The output of the delay-and-sum beam former is then fed into a LOB estimator. A simple LOB estimator takes the beamformer output and calculates the signal energy for each direction. The maximum weighted average of signal energy over all directions is the LOB estimate for the signal.

4.2 Energy-Modeling of the mAMPS node

Accurate estimates of the energy specifications of the hardware will be instrumental for the design of applications for sensor networks with large node densities. Modeling the node will also be helpful so that the sensors are able to dynamically estimate the energy requirement of an application, make decisions about their processing ability based on user-input and sustainable battery life, and configure themselves to meet the required goals. For example, based on the energy model for the application and the system lifetime requirements, the sensor node should be able to decide whether a particular application can be run. If not, the node might reduce its voltage using an embedded DC/DC converter and run the application at reduced throughput or run at the same throughput but with reduced accuracy. Both of these configurations would reduce energy dissipation and increase the node's lifetime. These energy-accuracy-throughput trade-offs necessitate robust energy models for software based on parameters such as operating frequency, voltage and target processor.

4.2.1 Processor Energy Model

The computation needed is performed by the StrongARM microprocessor in software. We have developed a simple energy model for software using frequency and supply voltage as parameters that incorporates explicit characterization of both switching and leakage energy. Most current models only consider switching energy, but in microsensor nodes which have low duty cycles, leakage energy dissipation can become large.

$$E_{tot}(V_{dd}, f) = NC_L V_{dd}^2 + V_{dd}\left(I_0 e^{\frac{V_{dd}}{nV_T}}\right)\left(\frac{N}{f}\right) \quad (3)$$

where C_L is the average capacitance switched per cycle and N is the number of cycles the program takes to execute. Both these parameters can be obtained from the energy consumption data for a particular supply voltage, V_{dd}, and frequency, f, combination. The model can then be used to predict energy consumption for different supply-throughput configurations in energy-constrained environments, such as wireless microsensor networks.

Experiments on the StrongARM SA-1100 have verified this model. For the SA-1100, the processor-dependent parameters I_0 and n are computed to be 1.196mA and 21.26 respectively. Then at V_{dd}=1.5V and f=206MHz, for several typical sensor DSP routines, C_L is calculated from equation (3), and E_{tot} is measured from the StrongARM. This C_L is used

with our processor energy model to estimate E_{tot} for all possible V_{dd}, f combinations. Table 3 shows the maximum error produced by the model was less than 5% for a set of benchmark programs.

		Model Parameters		
DSP Routines	**Measured Energy (mJ)**	***N* (x 10^6)**	**C_L**	**Error (%)**
fft	53.89	43.67	0.65	1.24
dct	0.10	0.08	0.66	4.22
idct	0.13	0.10	0.66	2.59
fir	1.23	0.97	0.70	3.28
tdlms	21.29	17.10	0.71	1.91

TABLE 3. Software energy model performance.

A more advanced level of processor energy modeling is to profile the energy for different instructions. It is natural that for different instructions the processor will dissipate different amounts of energy. Figure 16 shows the average current drawn from the StrongARM SA-1100 while executing different instructions at V_{dd}=1.5V. This figure shows that there are variations in current drawn for different classes of instructions (e.g. memory access, ALU), but the differences are not appreciable. Thus, the common overhead associated with all instructions (e.g. instruction fetch, caches) dominate the energy dissipated per operation. We expect that the variation between instructions will be more prominent in processors that use clock gating. Clock gating is a widely used low power technique where the clock is only enabled for those circuits that are active. Disabling non-active circuits eliminates unnecessary switching, which leads to energy savings.

4.2.2 Joule Track

Estimation of software energy consumption is becoming crucial in embedded applications. Instruction level power estimation tools have been proposed to computed the energy consumption of a given software. The basic idea is to run each instruction or short sequences of instruction in a loop and measure the current/power consumption. The primary drawback, however, is that these tools rely on exhaustive characterization of the energy consumption of the entire ISA (Instruction Set Architecture) and inter-instruction effects. The estimation model is computationally intensive, requiring a complete trace analysis of the program's instructions and is therefore slow. Our experiments on the StrongARM, SA-1100 microprocessor show that the variation in the current consumption is quite small. A lot of overheads are common across instructions and as a result the overall current consumption of a program is a weak function of the actual instruction stream and to a first order depends only on the operating frequency and voltage. Second order variations do exist but were measured to be less than 7% for a set of benchmark programs. Therefore, a complete instruction level trace analysis is unnecessary and a simple cycle

accurate simulation can be used. We propose a simple fast technique to estimate software energy and estimate second order variations. Initial experiments indicate an accuracy within 3% of actual measurements.

The experimental setup consisted of the Brutus SA-1100 Design Verification Platform which is essentially the StrongARM SA-1100 microprocessor connected to a PC using a serial link. It can operate from 59 MHz to 206 MHz, with a corresponding core supply voltage of 0.8 V to 1.5 V. The power supply to the StrongARM core was provided externally through a variable voltage sourcemeter with the I/O pads running at a fixed supply voltage. Current measurements were performed using the sourcemeter built into the variable power supply. The instruction and data caches were enabled before the programs were executed. To measure the current that is drawn by a subroutine, the subroutine was placed inside a loop with multiple iterations till a stable value of current was measured.

Figure 16 shows the current consumption of all the instructions of the ARM instruction set on SA-1100. Each of the 33 current values are themselves the averages of the various addressing modes and inputs in which the instruction can be executed accounting for a total of about 280 data points. The important point to observe is that the current consumptions are pretty uniform. On an average, arithmetic and logical instructions consume 0.178A, multiplies 0.196A, loads 0.196A, stores 0.229A while the other instructions consume about 0.170A. The total variation in current consumption is 0.072A which is 38% of the overall average current consumption.

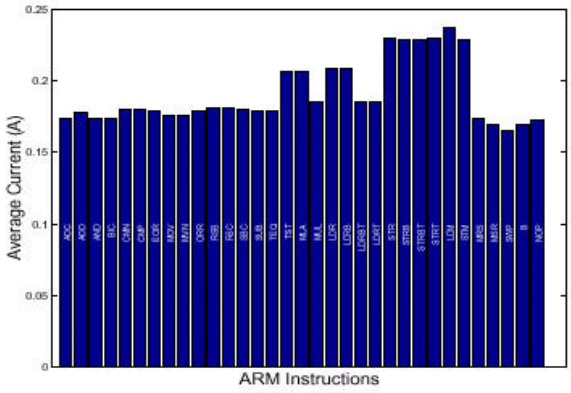

FIGURE 16. Current profiling of different instructions executed on the StrongARM SA-1100.

Based on the previous discussion, it is reasonable to conclude that the common overheads (such as caches, decode logic etc.) in contemporary microprocessors are large and overshadow any instruction specific variations. Therefore, estimating software energy consumption with an elaborate instruction trace and inter-instruction analysis is overkill.

The estimation techniques described in the previous sections were implemented in a web-based tool called JouleTrack. The tool is available at **http://dry-martini.mit.edu/JouleTrack**

The broad approach in the tool is summarized in Figure 17. The user uploads his C source code. The webserver uses Common Gateway Interface (CGI) scripts to create a temporary work area for the user. His programs are compiled and linked with any standard C libraries. The user also specifies any command line arguments that the program might need along with a target operating frequency. Compiler optimization options are also available. The user can choose the current prediction model. Compile/ link time errors are reported back by the CGI to the user. If no errors exist the program is executed on an ARM simulator which produces the program outputs (which the user can view), assembly listing (which can also be viewed) as well as run-time statistics like execution time, cycle counts etc. These statistics are fed into an estimation engine which computes the energy profile and charts the various energy components using the methodology described in the previous sections.

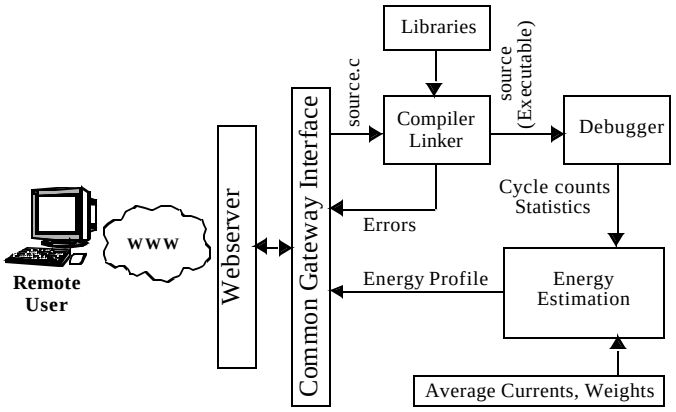

FIGURE 17. JouleTrack block diagram

4.2.3 Radio Energy Model

In order to collaborate with neighboring sensors and with the end-user, the data from the data and control module is passed to the radio or communication module. The primary component of the radio is a commercial single-chip transceiver optimized for ISM 2.45 GHz wireless systems. The radio module is capable of transmitting up to 1 Mbps at a range of up to 10 meters.

An energy model for the communication module has also been developed to model the energy dissipated by a sensor node when transmitting and receiving data. The radio module energy dissipation can be characterized into two types. The energy dissipated to run the transmit or receive electronics which is given by E_{elec} (J/bit) and the energy dissipated by the transmit power amplifier to achieve an acceptable E_b/N_o at the receiver is given by ε_{amp} (J/bit/m^2). We assume an r^2 energy loss for transmission between sensors since the distances between sensors are relatively short. To transmit a *k*-bit packet a distance, *d*, the energy dissipated is

$$E_{tx}(k, d) = E_{\text{elec}} \cdot k + \varepsilon_{\text{amp}} \cdot k \cdot d^2 \quad (4)$$

and to receive the *k*-bit packet, the radio expends

$$E_{rx}(k) = E_{\text{elec}} \cdot k \tag{5}$$

With good node energy models we can begin analyzing and designing more elaborate sensor applications such as the wireless sensor network scheme for acoustic sensors.

4.3 Energy-Efficient Protocols

Communication protocols can have significant impact on the overall energy dissipation of these networks. Based on our findings that the conventional protocols of direct transmission, minimum-transmission-energy, multi-hop routing, and static clustering may not be optimal for sensor networks, we proposed LEACH (Low-Energy Adaptive Clustering Hierarchy), a clustering-based protocol that utilizes randomized rotation of local cluster base stations (cluster-heads) to evenly distribute the energy load among the sensors in the network.

Microsensor networks tend to have the following attributes:

- The base station is fixed and located far from the sensors.
- All nodes in the network are homogeneous and energy-constrained.

Thus, communication between the sensor nodes and the base station is expensive, and there are no “high-energy” nodes through which communication can proceed. Sensor networks contain too much data for an end-user to process. Therefore, automated methods of combining or aggregating the data into a small set of meaningful information is required. In addition to helping avoid information overload, data aggregation, also known as data fusion, can combine several unreliable data measurements to produce a more accurate signal by enhancing the common signal and reducing the uncorrelated noise.

There have been several network routing protocols proposed for wireless networks that can be examined in the context of wireless sensor networks. We examine two such protocols, namely direct communication with the base station and minimum-energy multi-hop routing using our sensor network and radio models. In addition, we discuss a conventional clustering approach to routing and the drawbacks of using such an approach when the nodes are all energy-constrained.

Using a direct communication protocol, each sensor sends its data directly to the base station. If the base station is far away from the nodes, direct communication will require a large amount of transmit power from each node. This will quickly drain the battery of the nodes and reduce the system lifetime. However, the only receptions in this protocol occur at the base station, so if either the base station is close to the nodes, or the energy required to receive data is large, this may be an acceptable (and possibly optimal) method of communication.

The second conventional approach we consider is a "minimum-energy" routing protocol. There are several power-aware routing protocols discussed in the literature. In these protocols, nodes route data destined ultimately for the base station through intermediate nodes. Thus nodes act as routers for other nodes' data in addition to sensing the environment. These protocols differ in the way the routes are chosen. Some of these protocols, only consider the energy of the transmitter and neglect the energy dissipation of the receivers in determining the routes. In this case, the intermediate nodes are chosen such that the transmit amplifier energy (e.g., $E_{Tx\text{-}amp}(k,d) = \varepsilon_{amp} * k * d^2$ is minimized; thus node A would transmit to node C through node B if and only if:

$$E_{Tx\text{-}amp}(k,d = d_{AB}) + E_{Tx\text{-}amp}(k,d=d_{BC}) < E_{Tx\text{-}amp}(k,d=d_{AC}) \quad (6)$$

or

$$d_{AB}^2 + d_{BC}^2 < d_{AC}^2 \quad (7)$$

However, for this minimum-transmission-energy (MTE) routing protocol, rather than just one (high-energy) transmit of the data, each data message must go through n (low-energy) transmits and n receives. Depending on the relative costs of the transmit amplifier and the radio electronics, the total energy expended in the system might actually be greater using MTE routing than direct transmission to the base station.

By analyzing the advantages and disadvantages of conventional routing protocols using our model of sensor networks, we have developed LEACH (Low-Energy Adaptive Clustering Hierarchy), a clustering-based protocol that minimizes energy dissipation in sensor networks. The key features of LEACH are:

- Localized coordination and control for cluster set-up and operation.
- Randomized rotation of the cluster "base stations" or "cluster-heads" and the corresponding clusters.
- Local compression to reduce global communication.

The use of clusters for transmitting data to the base station leverages the advantages of small transmit distances for most nodes, requiring only a few nodes to transmit far distances to the base station. However, LEACH outperforms classical clustering algorithms by using adaptive clusters and rotating cluster-heads, allowing the energy requirements of the system to be distributed among all the sensors. Thus a set *C* of nodes might elect themselves cluster-heads at time t_1, but at time t_{1+d} a new set *C*, of nodes elect themselves as cluster-heads, as shown in Figure 18. The decision to become a cluster-head depends on the amount of energy left at the node.

In addition, LEACH is able to perform local computation in each cluster to reduce the amount of data that must be transmitted to the base station. This achieves a large reduction in the energy dissipation, as computation is much cheaper than communication.

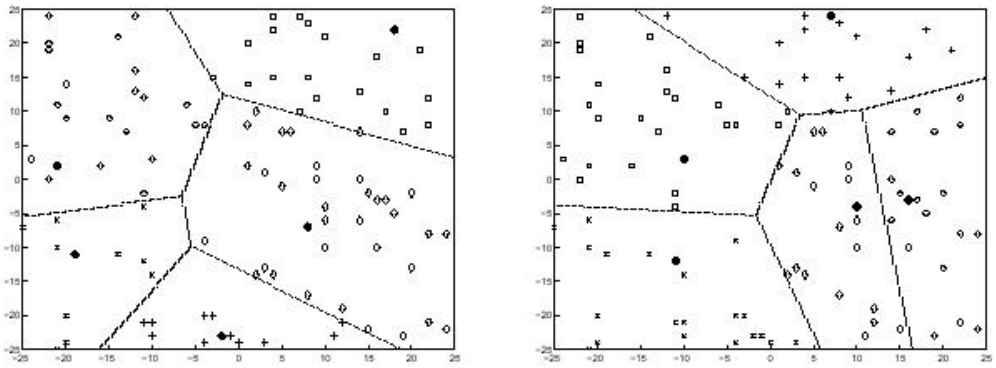

FIGURE 18. Dynamic Cluster Formation.

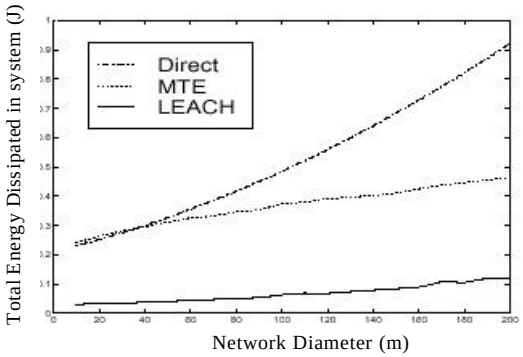

FIGURE 19. Energy Comparison between different protocols.

We simulated the direct transmission, minimum-transmission-energy, and LEACH protocols for a random 100-node network using the radio parameters E_{elec}= 50 nJ/bit and ε_{amp}= 100 pJ/bit/m^2 and a computation cost of 5 nJ/bit/message to fuse 2000-bit messages while varying the percentage of total nodes that are cluster-heads. Figure 19 shows how the three algorithms compare as the diameter of the network is increased. This plot shows that LEACH achieves between 7x and 8x reduction in energy compared with direct communication and between 4x and 8x reduction in energy compared with MTE routing.

4.4 Energy-Scalable Algorithms

4.4.1 Energy-Quality Scalability for Algorithms

Energy-scalability can be achieved by monitoring energy resources, latency and performance requirements to dynamically reconfigure system functionality. Energy-Quality (E-Q) trade-offs have been explored in the context of encryption processors. Energy-scalability at the algorithm and protocol levels is highly desirable because a large range of both

energy and quality can be achieved by varying algorithm parameters. A large class of algorithms, as they stand, do not render themselves to such E-Q scaling.

Let us assume that there exists a quality distribution $p_Q(x)$, the probability that the end-user desires quality x. Then the average energy consumption per output sample can then be expressed as

$$\bar{E} = \int p_Q(x)E(x)\,dx \tag{8}$$

where $E(x)$ is the energy dissipated by the system to give quality x. A typical energy-quality (E-Q) distribution is shown in Figure 20. It is clear that Algorithm II is desirable over Algorithm I because it gives higher quality at lower energies and especially when $p_Q(x)$ is large.

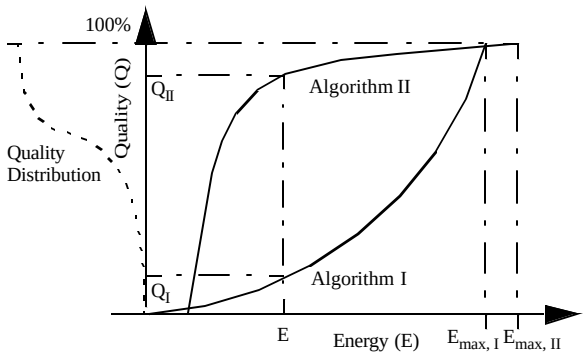

FIGURE 20. Examples of Energy-Quality curves.

When the quality distribution is unknown, the E-Q behavior of the algorithm can be engineered such that it has two desirable traits. First, the quality on average should be monotonically increasing as energy increases and second, the E-Q curve should be concave down:

$$Q(E_1) \geq Q(E_2) \text{ if } E_1 \geq E_2 \tag{9}$$

$$\frac{d^2Q(E)}{dE^2} \leq 0 \quad \text{for} \quad 0 \leq E \leq E_{max} \tag{10}$$

where $Q(E)$ is an accurate model of the algorithm's average quality as a function of computational energy and is the inverse of $E(Q)$. These constraints lead to intelligent energy-scalable systems. An E-Q curve that is concave downward is highly desirable since close to maximal quality is achieved at lower energies. Conversely, a system that has a concave upwards E-Q curve can only guarantee high quality by expending a large amount of energy.

Algorithmic transformations can be used to improve the E-Q characteristics of a system. For example, we will show that the E-Q curves for both FIR filtering and LMS beamforming for data aggregation can be transformed for better energy-scalability systems.

4.4.2 Energy-Agile Filtering

Finite Impulse Response (FIR) filtering is one of the most commonly used DSP operations. FIR filtering involves the inner product of two vectors one of which is fixed and known as the impulse response, $h[n]$, of the filter. An N-tap FIR filter is defined by equation (11).

$$y[n] = \sum_{k=0}^{N-1} x[n-k]h[k] \tag{11}$$

However, when we analyze the FIR filtering operation from a pure inner product perspective, it simply involves N multiply and accumulate (MAC) cycles. For desired *E-Q* behavior, the MAC cycles that contribute most significantly to the output $y[n]$ should be done first. Each of the partial sums, $x[k]h[n-k]$, depends on the data sample and therefore its not apparent which ones should be accumulated first. Intuitively, the partial sums that are maximum in magnitude (and can therefore affect the final result significantly) should be accumulated first. Most FIR filter coefficients have a few coefficients that are large in magnitude and progressively reduce in amplitude. Therefore, a simple but effective *most-significant-first transform* involves sorting the impulse response in decreasing order of magnitude and reordering the MACs such that the partial sum corresponding to the largest coefficient is accumulated first as shown in Figure 21(a). Undoubtedly, the data sample multiplied to the coefficient might be so small as to mitigate the effect of the partial sum. Nevertheless, on an average case, the coefficient reordering by magnitude yields a better *E-Q* performance than the original scheme.

Figure 21(b) illustrates the scalability results for a low pass filtering of speech data sampled at 10kHz using a 128-tap FIR filter whose impulse response (magnitude) is also outlined. The average energy consumption per output sample (measured on the StrongARM SA-1100 operating at 1.44V power supply and 206MHz frequency) in the original scheme is 5.12µJ. Since the initial coefficients are not the ones with most significant magnitudes the *E-Q* behavior is poor. Sorting the coefficients and using a level of indirection (in software that amounts to having an index array of the same size as the coefficient array), the *E-Q* behavior can be substantially improved. It can be seen that fluctuations in data can lead to deviations from the ideal behavior suggested by equation (10), nonetheless overall concavity is still apparent. The energy overhead associated with using a level of indirection on the SA-1100 was only 0.21µJ which is about 4% of the total energy consumption.

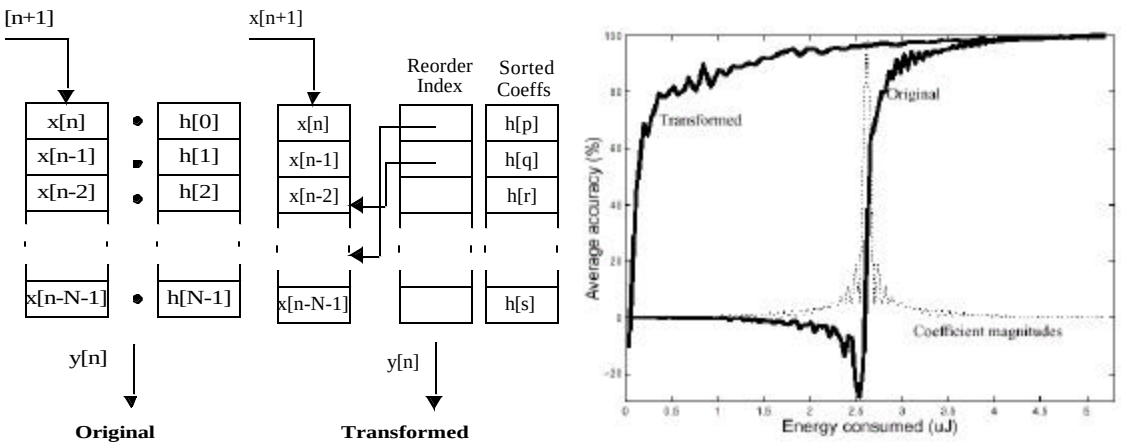

FIGURE 21. (a) FIR filtering with coefficient reordering (b) E-Q graph for original and transformed FIR filtering

4.4.3 Energy-Efficient system partitioning between sensors and cluster-head

One way to reduce energy dissipation is to recognize that within a sensor cluster multiple sensors translate to multiple microprocessors and to distribute the computation among the sensors. This can be shown by implementing the frequency-domain LOB estimation technique for one sensor cluster. Figure 22 is a block diagram breaking down the computation involved in the Line of Bearing algorithm. The first part is to transform collected acoustic sensor data from each sensor into the frequency domain using a 1024-pt. Fast Fourier Transform (FFT). Then, we beamform the FFT data into twelve uniform directions. The direction of the signal with the most energy is the LOB of the source.

The LOB estimation algorithm can be implemented in two different ways. In the *direct* technique, each sensor has a set of acoustic data, $s_i(n)$. This data is transmitted to the clusterhead where all of the LOB estimation algorithm is done. This technique is demonstrated

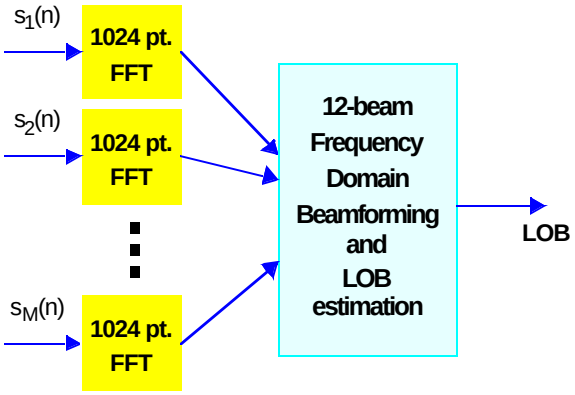

FIGURE 22. Block diagram of the Line of Bearing (LOB) estimation algorithm.

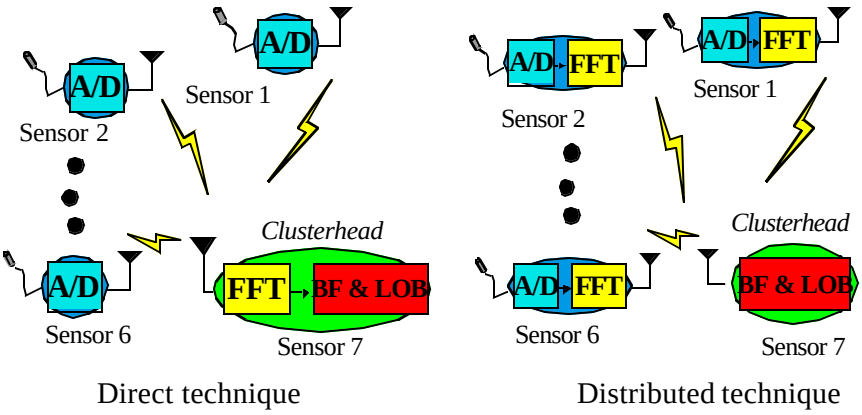

FIGURE 23. (a) Direct technique: All of the computation is done at the clusterhead. (b) Distributed technique: Distribute the FFT computation among all sensors.

in Figure 23(a). Alternatively, we can first perform the FFT's at each sensor and then send the FFT results to the clusterhead. This is the *distributed* method and is shown in Figure 23(b). If we assume the processor models discussed previously, then performing the FFT's with the distributed technique has no computational energy savings over the direct technique, because the same total amount of computation is being done. However, by having a dynamic voltage scaling (DVS) enabled sensor node, the node can take advantage of the parallelized computational load by allowing voltage and frequency to be scaled while still meeting latency constraints.

Increasing parallelism is a common technique used in circuit design at the architectural level to reduce energy dissipation when there is a fixed latency. One example is to use two functional units in the database, versus one unit. By adding an one more functional unit, this allows each unit to run at half the original rate, for the same throughput. Since the clock frequency requirement has decreased, the voltage supply can be dropped to $V_{dd}/2$, and the energy is reduced by almost 4 times over the non-parallel case. However, this does come at an increase of area and overhead control hardware.

In the DVS enabled sensor node, there is a large advantage to have the computation distributed among the sensor nodes, since the voltage supply can be reduced. Table4 shows the computation energy for a 7 sensor cluster. In the direct technique, with a computation latency constraint of 20 msec, all of the computation is performed at the clusterhead at the fastest clock speed, f=206 MHz at 1.44V. The energy dissipated by the processor is measured to be 6.2 mJ and the latency is 19.2 msec. In the distributed technique, the FFT is parallelized to the sensor nodes. In this scheme, the sensor nodes sense data and perform the 1024-pt. FFT's on the data before transmitting the FFT data to the clusterhead. At the clusterhead, the beamforming and LOB estimation is done. Since the FFT's are parallelized, the clock speed and voltage of both the FFT's and the beamforming can be lowered. For example, if the FFT's at the sensor nodes are run at 0.85V voltage supply and 74 MHz clock speed while the beamforming algorithm is run at 1.17V and 162 MHz clock speed then with a latency of 18.4 msec, only 3.4 mJ energy is dissipated by the processor. This is

a 45.2% improvement in energy dissipation. This example shows that energy-efficient system partitioning by parallelism in system design can yield large energy savings.

		Direct	Distributed
Nodes	V_{dd}	-	0.85 V
	f	-	74 MHz
Clusterhead	V_{dd}	1.44V	1.17 V
	f	206 MHz	162 MHz
	Latency	19.2 msec	18.4 msec
	Energy	6.2 mJ	3.4 mJ

TABLE 4. Energy results for direct and distributed technique for a 7 sensor cluster.

An additional bonus in distributing the FFT is the reduction of communication energy between sensors. Due to the nature of the sound source, the signal of interest has a bandwidth between 20Hz and 250 Hz. This means that after doing the 1024-pt. FFT, only 230 complex Fourier coefficients need to be transmitted. This means that the communication energy for the distributed technique can be reduced by 50% over the direct technique, where all 1024 samples are transmitted.

5 Conclusions

Our work in power aware wireless microsensors has demonstrated power-aware design at all levels of the system hierarchy, from CMOS circuits to signal processing algorithms. Moreover, power awareness is extended to the entire network of nodes, with the interactions among multiple nodes determine the total energy consumption of the network.

A microsensor node that can gather and transmit data for years must operate at energy efficiencies unheard of in today's wireless systems. Sensor nodes must control energy consumption in the active and idle modes, respectively, by scaling power consumption with changing performance demands, and shutting down during the long periods of idle time between interesting events. The user must precisely define the network's performance requirements using metrics ranging from latency to accuracy to reliability, so that the network performs just enough computation (and no more) to meet the user's specific demands. The energy consumption of our μAMPS node is aggressively managed for both active and idle operation. Almost every subsystem provides a means of complete shutdown, creates maximal power savings over long idle periods. The largest power consumers (the processor and radio power amplifier) provide scalable performance when in active modes, allowing dynamic quality/energy tradeoffs. For flexibility as a testbed, the node is constructed with stacked boards to facilitate the addition of sensor modules, and offers fully-programmable signal processing and link layer functionality. Our node is a complete, flexible, power aware solution for the creation and evaluation of microsensing applications.

The network must consider itself as a single entity, where collaborative communication protocols remove redundancies in computation and communication, and maintain an even spatial distribution of energy. Clustering techniques allow the network to aggregate redundant data and reduce the energy of long-range transmissions, and LEACH in particular improves the lifetime of the network even further through collaborative rotation of the clusterhead role. Signal processing algorithms performed within a cluster can be partitioned to increase parallellism, allowing deeper voltage scaling under fixed latency constraints. Protocols and collaborative algorithms for microsensor networks must be designed specifically for the application at hand.

Through the development of power aware hardware, protocols, algorithms, and tools, we have revealed the potential for energy savings of multiple orders of magnitude, leading to dramatic increases in the operational lifetime of microsensor nodes. Careful attention to the details of energy consumption at every point in the design process and the aggressive inclusion of energy-scalable fabrics will be the key enablers for dense, robust microsensor networks that deliver maximal system lifetime in the most challenging and operationally diverse environments.

PUBLICATIONS

Papers in Refereed Journals

1. Sinha, A., A. Chandrakasan, "Dynamic Power Management in Wireless Sensor Networks," *IEEE Design and Test of Computers*, pp. 62-76, March-April 2001.
2. Baradwaj, M., R. Min, A. P. Chandrakasan, "Quantifying and Enhancing Power-Awareness for VLSI Systems," accepted for publication in the *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 757-772, December 2001.
3. Wang, A., Heinzelman, W., A. P. Chandrakasan, "Energy-Scalable Protocols for Battery-Operated Microsensor Networks," *Journal of VLSI Signal Processing*, 2001.
4. Goodman, J., A. P. Chandrakasan, "An energy-efficient reconfigurable public-key cryptography processor," *IEEE Journal of Solid-State Circuits*, November 2001.
5. Sinha, A., A. Wang, A.P. Chandrakasan, "Energy Scalable System Design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 135-145, April 2002.
6. Heinzelman, W., A. P. Chandrakasan, H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor" to appear in the *IEEE Transactions on Wireless Networking*, 2003.
7. Min, R., M. Bhardwaj, S. Cho, N. Ickes, E. Shih, A. Sinha, A. Wang, A. P. Chandrakasan, "Energy-Centric Enabling Technologies for Wireless Sensor Networks", to appear in the *IEEE Communications Magazine*, 2002.

8. Shih, Eugene, “Physical Layer Design Considerations For Energy-Efficient Radios in Wireless Microsensor Networks,” accepted for publication in the *Journal of VLSI Signal Processing*.

9. Bhardwaj, B., A. P. Chandrakasan, “Upper Bounds on the Lifetime of Sensor Networks,” accepted for publication in the *IEEE Transactions on Wireless Networking*.

10. Kao, J. M. Miyazaki, A. P. Chandrakasan, “A 175mV Multiply-Accumulate DSP Core Using an Adaptive Supply Voltage and Body Bias (ASB) Architecture,” accepted for publication in the *IEEE Journal of Solid-state Circuits*.

11. Wang, A., A. P. Chandrakasan, "Energy-Efficient DSP for Wireless Sensor Networks," IEEE Signal Processing Magazine, pp. 68-78, July 2002.

Proceedings of Refereed Conferences

12. Heinzelman, W., A. Sinha, A. Wang, A. Chandrakasan, “Energy-scalable Algorithms and Protocols for Wireless Microsensor Networks,” ICASSP 2000.

13. Sinha, A., A. Wang, and A. Chandrakasan, “Algorithmic Transforms for Efficient Energy Scalable Computation,” The 2000 IEEE International Symposium on Low-Power Electronic Design (ISLPED ’00), August 2000.

14. Goodman, J., A.P. Chandrakasan, “An Energy Efficient Reconfigurable Public-Key Cryptograhpy Processor Architecture,” CHES 2000, pp. 175-190, August 2000.

15. Min, R., B. Bhardwaj, S. Cho, A. Sinha, E. Shih, A. Wang, A.P. Chandrakasan, “An Architecture for a Power-Aware Distributed Microsensor Node,” IEEE Workshop on Signal Processing Systems (SiPS ’00), pp. 581-590, October 2000.

16. Bhardwaj, M., R. Min and A. P. Chandrakasan, “Power-Aware Systems,” 34th Asilomar Conference on Signals, Systems, and Computers, November 2000.

17. Sinha, A., A. P. Chandrakasan, “Operating System and Algorithmic Techniques for Energy Scalable Wireless Sensor Networks”, 2nd International Conference on Mobile Data Management (MDM ’01), Hong Kong, January 2001.

18. Sinha, A., A. P. Chandrakasan, “Dynamic Voltage Scheduling Using Adaptive Filtering of Workload Traces,” VLSI Design 2001, Bangalore, India, January 2001.

19. Min, R., M. Bhardwaj, S. Cho, E. Shih, A. Sinha, A. Wang, A. P. Chandrakasan, “Low-Power Wireless Sensor Networks,” 14th Interconational Conference on VLSI Design 2001, Bangalore, India, January 2001. (INVITED)

20. Goodman, J., A.P. Chandrakasan, “An Energy Efficient IEEE 1363-based Reconfigurable Public-Key Cryptography Processor,” IEEE International Solid-State Circuits Conference, pp. 330-331, Feb. 2001, San Francisco, California.

21. Wang, A., A.P. Chandrakasan, “Energy Efficient System Partitioning For Distributed Wireless Sensor Networks”, Proceedings of ICASSP, May 2001.

22. Bharadwaj, M., T. Garnett, A. P. Chandrakasan, “Upper Bounds on Lifetime of Sensor Networks,” Proc. of ICC, June 2001.

23. Shih, E., B. H. Calhoun. S. Cho, A.P. Chandrasakan, "Energy-Efficient Link-Layer for Wireless Microsensor Networks", Proceedings of IEEE Workshop on VLSI '01, Orlando, Florida, April 2001.

24. Sinha, A., A. P. Chandrakasan, "JouleTrack - A Web Based Tool For Software Energy Profiling," ACM/IEEE Design Automation Conference, pp. 220-225, Las Vegas, Nevada, June 2001.

25. Shih, E., S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical Layer Driven Algorithm and Protocol Design for Energy-Efficient Wireless Sensor Networks", Proceedings of MOBICOM 2001, Rome, Italy, pp. 272-287, July 2001.

26. Sinha, A., A. P. Chandrakasan, "Energy Efficient Real-Time Scheduling," IEEE/ ACM International Conference on Computer-Aided Design, 2001.

27. Min, R., A. P. Chandrakasan, "Energy-Efficient Communication for Ad-hoc Wireless Sensor Networks", 35thAsilomar Conference on Signals, Systems, and Computers, November 2001.

28. Miyazaki, M. J. Kao, A. Chandrakasan, "A 175mV Multiply-Accumulate Unit Using an Adaptive Supply Voltage and Body Bias (ASB) Architecture," IEEE ISSCC, pp. 58-59, San Francisco, California, February 2001.

29. Bhardwaj, M., A. P. Chandrakasan, "Bounding the Lifetime of Sensor Networks Via Optimal Role Assignments," IEEE Infocom 2002.

30. Min, R., A.P. Chandrakasan, "A Framework for Energy-Scalable Communication in High-Density Wireless Networks", IEEE International Symposium on Electronics and Design, August 2002.

31. Chandrakasan, A.P., R. Min, M. Bhardwaj, S. Cho, A. Wang, "Power Aware Wireless Microsensor Systems," Keynote Address, IEEE ESSCIRC, September 2002.

Book Chapters

32. Sinha, A., A. P. Chandrakasan, "Software Energy Profiling," included in *Power Aware Computing*, editors Robert Graybill and Rami Melhem, Kluwer Academic Publishers, 2002.

33. Min, R., S. Cho, M. Bhardwaj, E. Shih, A. Wang, A. P. Chandrakasan, "Power Aware Wireless Microsensor Networks," in *Low Power Design Methodologies*, editors M. Pedram and J. Rabaey, to be published by Kluwer Academic Publishers, 2002.