

Communication-Based Power Management

Kanishka Lahiri

University of California, San Diego

Sujit Dey

University of California, San Diego

Anand Raghunathan

C&C Research Labs, NEC USA

A new battery-driven power management methodology, CBPM uses the system-level communication architecture to regulate system execution. It aims to improve battery efficiency and life.

■ **BATTERY-POWERED SYSTEMS** account for a significant and rapidly expanding segment of the electronics and semiconductor industries. Unfortunately, projections of the complexity, functionality, and performance of such systems far exceed expected improvements in battery technologies, leading to a widening “battery gap.”^{1,2} Bridging this gap is a challenge that system designers must face for the foreseeable future.

The need to improve battery life has, in large part, driven the research and development of low-power design techniques for electronic circuits and systems.³ Low-power design successfully reduces the energy a system draws from the battery and hence improves battery life to some extent. However, truly maximizing battery life requires an understanding of both the energy source and the energy-consuming system.

Research has shown that the amount of energy supplied by a battery varies significantly, depending on how the system draws the energy (the “Principles of battery discharge” sidebar explains the basics). Consequently, designers should develop battery-efficient, rather than

merely energy-efficient, design techniques specifically tailored toward the characteristics and limitations imposed by battery-powered systems.

Recent contributions to the area of battery-efficient system design include battery models that capture important electrochemical effects during battery discharge.^{4,7} These models estimate battery life and capacity under various discharge scenarios, and are a key part of a battery-driven design methodology. Techniques for designing battery-efficient systems include battery-driven frequency selection,⁸ voltage setting,⁹ task scheduling,¹⁰ and extensions to conventional power management.¹¹ Researchers have proposed battery scheduling for multi-battery systems to optimize a battery subsystem to best meet system power requirements.^{12,13} Groups are developing standards that improve interoperation of batteries and electronic systems (<http://www.sbs-forum.org>). Lahiri et al. provide a survey of battery models and battery-driven design techniques.²

Communication-based power management (CBPM) is a new battery-driven system-level power management methodology. It aims at improving battery life beyond what is achievable using state-of-the-art low-power design techniques.

CBPM basics

CBPM uses the system-level communication architecture to regulate the execution of system components. Regulating execution provides the desirable system-level power profiles and improved battery efficiency, as the

Principles of battery discharge

Figure A shows a battery's basic components; this example shows a lithium-thionyl chloride cell. The battery consists of an anode (Li), cathode (C), and an electrolyte. During discharge, oxidation at the anode releases

- electrons, which flow through the load system; and
- positively charged ions (Li^+), which migrate through the electrolyte toward the cathode by diffusion.

At available reaction sites in the cathode, Li^+ ions combine with negatively charged ions (Cl^-) to create an insoluble compound, LiCl .

Cathode sites covered by deposits of LiCl become inactive—unavailable for further reaction. As discharge proceeds, more reaction sites become unavailable, eventually leading to a state of complete discharge.

Terminology

A battery's *standard capacity* is the energy that the battery can deliver when discharged under standard load conditions. For example, a battery could have a standard capacity of 500 mA-h when discharged at a *rated current* of 125 mA at 25° C. The battery's *actual capacity* is the amount of energy it delivers under a given load. Actual capacity and battery life are typical metrics for battery efficiency.

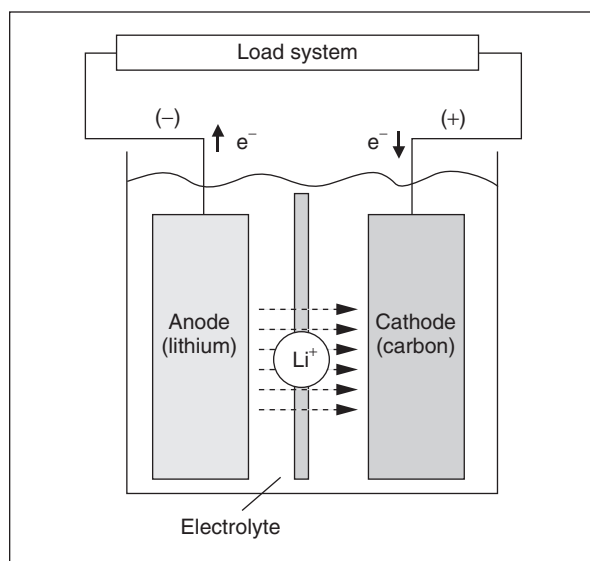


Figure A. Basic structure of a lithium-thionyl chloride battery.

Several electrochemical effects make battery capacity sensitive to the discharge current profile. Hence, the actual capacity can differ from the standard capacity, depending on the load conditions. In this article, we use battery capacity to refer to actual capacity.

Rate-related capacity effects

Battery life depends largely on the availability of active reaction sites throughout the cathode. Low discharge currents cause a uniform distribution of inactive reaction sites throughout the cathode's volume. However, intervals of high discharge current cover the outer surface of the cathode with inactive sites, making interior active sites unreachable. These rate-related capacity effects lead to an overall reduction in battery capacity at high discharge rates.¹

Recovery effects

When a device draws current from a battery, the chemical reaction consumes positively charged ions at the cathode-electrolyte interface, and replaces them with new ions, which diffuse through the electrolyte from the anode. When the drawn current is sufficiently large, the diffusion rate falls behind the ion consumption rate at the cathode. As a result, the concentration of positively charged ions decreases near the cathode and increases near the anode, degrading the battery's output voltage. However, if the battery is idle for a while, the concentration gradient decreases (because of diffusion), leading to an apparent charge recovery. As a result, the battery's capacity and life increase.²

Although this article demonstrates CBPM by explicitly targeting rate-related capacity effects, straightforward extensions can also target CBPM toward recovery effects. Also, the battery life improvements we report here use battery models that do account for both effects.

References

1. M. Doyle, T.F. Fuller, and J.S. Newman, "Modeling of Galvanostatic Charge and Discharge of Lithium/Polymer/Insertion Cell," *J. Electrochemical Soc.*, vol. 140, June 1993, pp. 1526-1533.
2. T.F. Fuller, M. Doyle, and J.S. Newman, "Relaxation Phenomena in Lithium-Ion Insertion Cells," *J. Electrochemical Soc.*, vol. 141, Apr. 1994, pp. 982-990.

CBPM rationale

Using the communication architecture to implement battery-driven power management has at least three significant advantages.

- *Minimal hardware overhead.* The communication architecture physically and logically integrates all of the system components. Embedding power management functionality in this architecture eliminates the need for a separate power management unit and the connections to it.
- *System-level visibility.* To exercise control over the system power profile, power management policies need access to system-wide information, such as the current execution states of the system components. The communication architecture's connectivity makes it a natural choice for monitoring and detecting changes in the execution states of system components. It can also provide an efficient mechanism for delivering power management decisions to any system component.
- *Significant impact on power profile.* The communication architecture directly controls the timing of system-level communications and hence the execution of tasks in system components. In particular, the communication architecture can control the timing of a component's idle and active states, affecting its power profile. Because the system power profile is the sum of component power profiles, systems can use the communication architecture to regulate the system power profile.

“CBPM rationale” sidebar explains. CBPM differs significantly from conventional dynamic power management techniques, which only aim to prevent energy waste during periods when the system or its components are idle. It also differs from techniques such as dynamic voltage scaling that merely target an overall reduction in average power consumption.

CBPM exercises dynamic and proactive control over system components, delaying components that are executing less-performance-critical operations, even if they are not idle. Each component consumes significantly less power when blocked than when active. Hence, by dynamically blocking the execution of certain components, CBPM can regulate the system's power profile.

Example: IEEE 802.11 MAC processor

We illustrate CBPM's impact by considering its application to an IEEE 802.11 media access control (MAC) processor,¹⁴ comparing the power profiles and battery efficiency of

- the original performance-optimized MAC processor (which already incorporates dynamic power management to minimize idle-component power consumption), and
- a CBPM-enhanced MAC processor.

We discuss the MAC processor's details later. For this discussion, it is sufficient to consider only the system power profiles and execution traces. Figure 1a shows a snapshot of the current discharge profile for the original MAC processor as it processes MAC frames generated by streaming video over an IEEE 802.11b wireless LAN. Current discharge profiles come from dividing the power profiles, obtained using hardware-software cosimulation,¹⁵ by the supply voltage.

We used a stochastic model⁶ of a lithium-ion battery with a 250 milliampere-hour (mA-h) capacity and a 125-mA rated current to measure battery life and capacity for the profile of Figure 1a. The battery's capacity was 154.3 mA-h; its life was 3,209.2 s. Figure 1a shows that the current drawn is significantly higher than the rated current in some regions.

To understand how CBPM improves battery efficiency, consider a symbolic execution trace, such as the one shown in Figure 1c, which corresponds to the current profile in Figure 1a. The trace depicts nine macrostates, which indicate specific subtasks performed by each system component at any given time. For example, WEP, the component responsible for encrypting frame data, can be in one of several macrostates, depending on whether it is reading a frame body from memory (WEP_MEM), initializing the key state array (WEP_INIT), or computing an encrypted frame body (WEP_COMP).

The macrostates of individual components determine the system state at any time. From Figures 1a and 1c, it is clear that system states vary widely with respect to the current drawn, depending on the individual component macrostates. For example, Figure 1c indicates that, at $t = 55$ ms, two hardware components (ICV and FCS) are engaged in lengthy iterative computations (ICV_COMP and FCS_COMP), while a third (WEP) accesses memory over a shared bus (WEP_MEM). Figure 1a indicates that this system state leads to a drawn current of 280

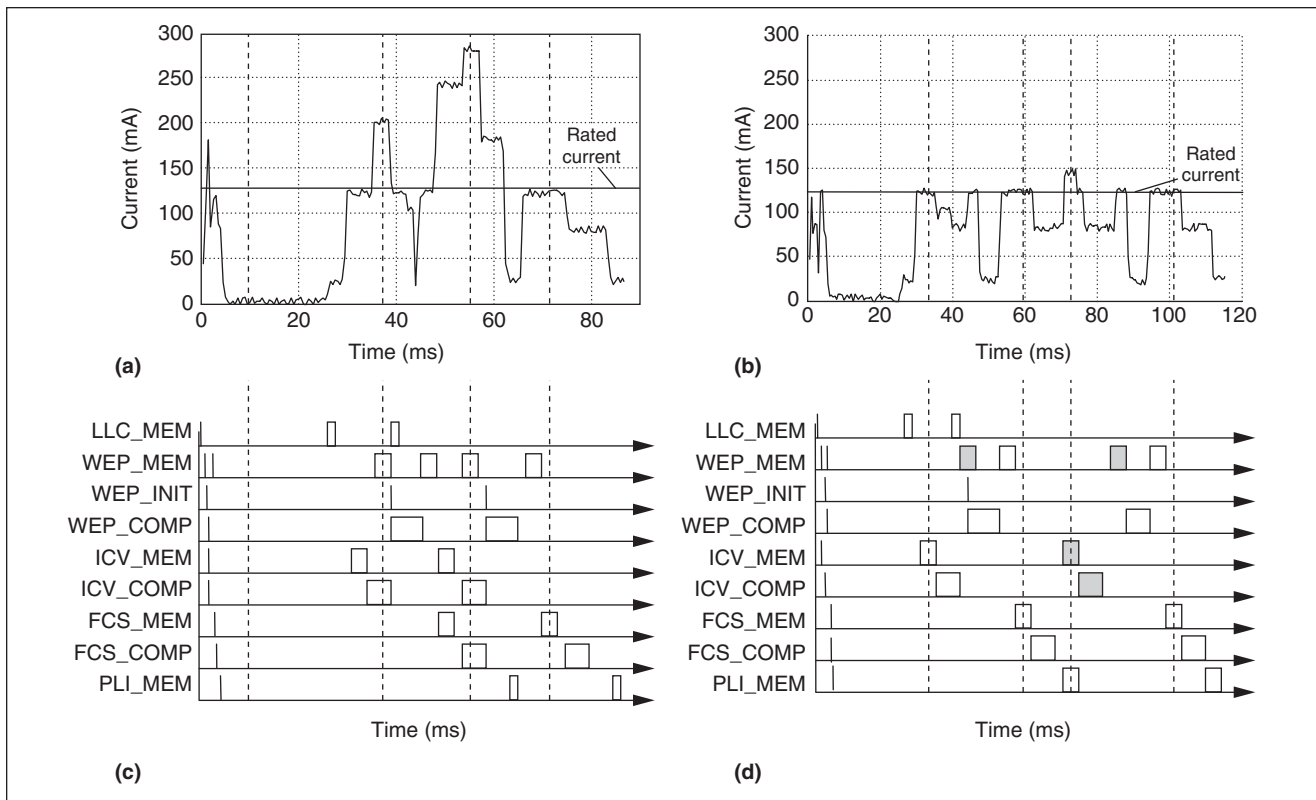


Figure 1. Discharge profiles for a MAC processor without CBPM (a) and with CBPM (b). Symbolic execution traces, without CBPM (c) and with CBPM (d), identify component macrostates.

mA, which exceeds the battery's rated current.

We next consider processing the same sequence of MAC frames using the CBPM-enhanced MAC processor. Figures 1b and 1d show the new current profile with its corresponding symbolic execution trace. A comparison of the symbolic execution traces in Figures 1c and 1d shows that CBPM has delayed the execution of certain macrostates (those indicated by shading) to stay within the rated current.

For example, Figure 1d shows that CBPM delays the occurrence of macrostate WEP_MEM to around $t = 42$ ms, to avoid overlap with macrostate ICV_COMP. From the current discharge profile in Figure 1a, it is clear that the CBPM policy greatly reduces the occurrence of system states that violate the rated current. We estimated battery life of the new system at 10,199 s (a 3.2-times improvement), and, more importantly, battery capacity was 225.4 mA-h (a 1.5-times improvement).

This example illustrates the significant

impact CBPM can have on system power profiles and overall battery efficiency.

CBPM design methodology

Figure 2 (next page) shows our methodology, which has four main inputs:

- a simulatable system specification mapped to a system architecture consisting of hardware and software components,
- the definition of a system-level communication architecture (see the "System-level communication architectures" sidebar, page 123, for an introduction),
- typical environment stimuli, and
- performance and battery life objectives.

CBPM depends on identifying macrostates, which execute under the communication architecture's control. Incorporating CBPM policies effectively changes the timing of certain macrostates to dynamically regulate the system power profile.

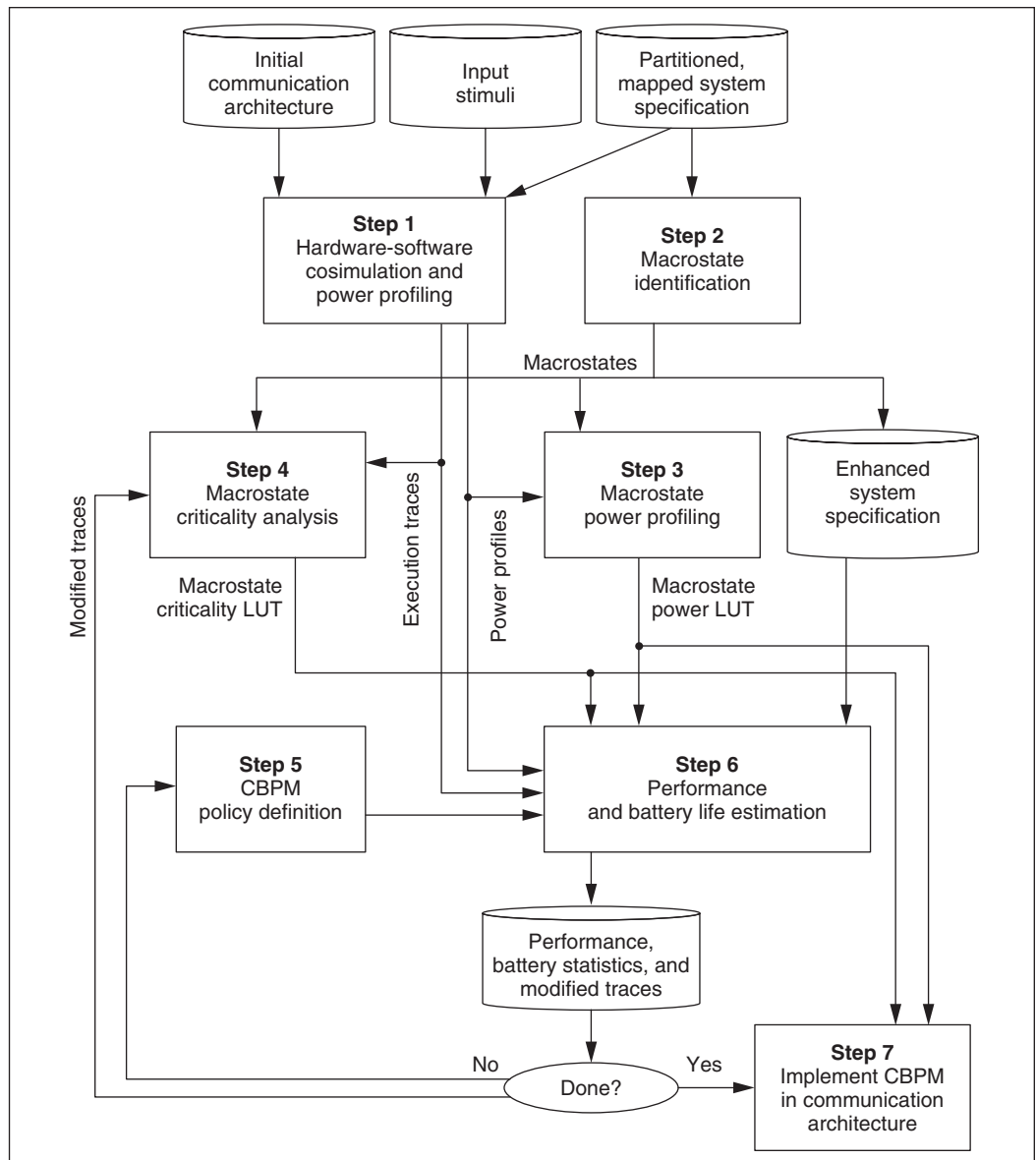


Figure 2. CBPM design methodology.

Step 1: Hardware-software cosimulation and power profiling. In this step, we perform hardware-software cosimulation using input stimuli to generate execution traces and power consumption profiles for each system component. We use the resulting execution traces and power profiles in steps 3, 4, and 6.

Step 2: Macrostate identification. A component's effect on system performance and its contribution to the system power profile can vary significantly, depending on which part of its specification it is currently executing. To

account for this dependence, we use the concept of a component macrostate. We identify component macrostates by analyzing the control flow graph (CFG) corresponding to the component's specification. A clustering algorithm is used to combine basic blocks in the CFG into macrostates. The algorithm is designed to constrain the number of identified macrostates (to minimize hardware and performance overheads) as well as ensure that the identified macrostates are small enough to enable sufficient control over the power profile.

To implement CBPM, we identify two types of

System-level communication architectures

Figure B shows the *system-level communication architecture*, the fabric that integrates system components and provides a mechanism for the exchange of data and control information between them. Rapid growth in system complexity is leading to an increase in the volume and diversity of on-chip communication. This increase, in turn, places stringent requirements on the communication architecture. As a result, the system-level communication architecture has begun to play an increasingly critical role in determining several system-wide metrics, including overall system performance and power consumption.

The design of system-level communication architectures must address several aspects, including architecture topology definition, protocol selection and configuration, and mapping.

Topology defines the communication architecture's physical structure. Topologies range from a single shared bus connecting all system components to an arbitrary network of shared and dedicated communication channels. When the topology has multiple channels, bridges interconnect the necessary channels.

Components connected to the communication architecture include

- master components—such as CPUs, digital signal processors, and direct-memory-access controllers—that can initiate communication transactions and
- slave components—such as memories and peripherals—that merely respond to transactions initiated by a master.

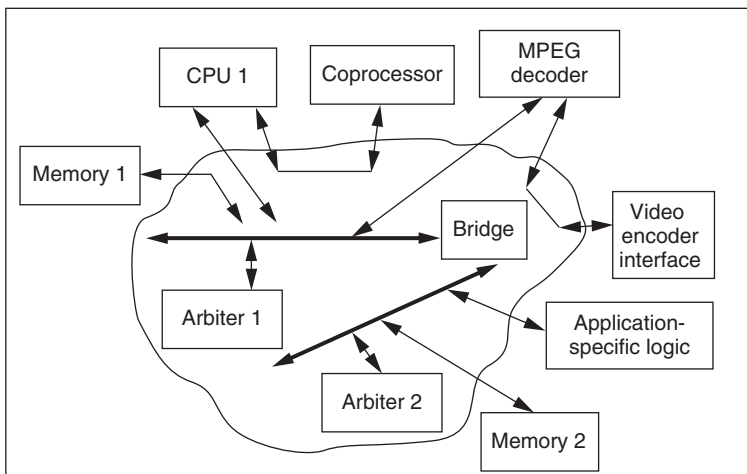


Figure B. System-level communication architecture with interconnected shared and dedicated communication channels.

Because multiple components often share channels, communication protocols manage access to each channel. These protocols implement arbitration algorithms such as round-robin access, priority-based selection, and time division multiple access. In addition, these protocols implement other functions such as burst transfers and pipelined transactions.

Communication mapping associates a system's communications to physical paths in the architecture. Although this step is trivial for systems with a single shared bus, it becomes much more complex when the communication architecture has several channels, resulting in many potential mapping alternatives.

communication requests. Type I requests represent conventional bus access requests for transferring data across the system bus, and occur in short intervals, potentially every bus cycle. Circuitry implementing the underlying bus protocol handles these requests based on any standard arbitration algorithm (such as static priority, time division multiple access, and so on). A type II request occurs whenever a component reaches a macrostate boundary; it represents a request to transition to a new macrostate.

CBPM works by enhancing each component's specification to generate a type II communication request at each macrostate

transition. Type II requests convey the component's next macrostate to the CBPM controller. Before executing its next macrostate, a component with a pending type II request must wait until the CBPM controller generates a grant.

Step 3: Macrostate power profiling. Next, we must estimate the average power consumption of each macrostate identified in step 2. This requires the following actions:

3a. Construct symbolic execution traces from each component's execution trace, indicating the timing of each macrostate.

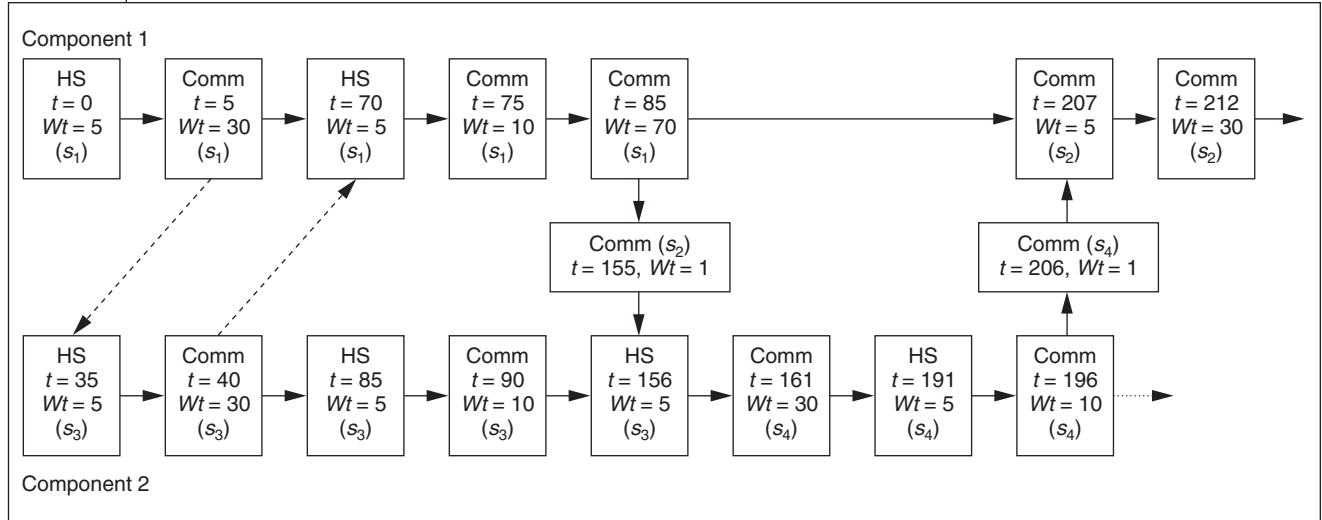


Figure 3. Sample CAG showing vertices labeled with timing and macrostate information.

- 3b. Calculate $pow(s_{ij})$, the average power consumed by each instance j of macrostate s_i , from the power profiles and symbolic traces.
- 3c. Find $pow(s_i)$, the average power consumed by macrostate s_i , by averaging $pow(s_{ij})$ over all j .

The output of step 3 is a macrostate power table, consisting of tuples $[s_i, pow(s_i)]$, representing macrostates and their respective average power consumptions.

Step 4: Macrostate criticality analysis. We estimate sensitivity of overall system performance to each macrostate by using a trace-based performance analysis framework.¹⁶ The analysis proceeds in two phases. A preprocessing phase analyzes execution traces (from step 1) to construct a communication analysis graph (CAG), an abstract yet accurate representation of the system's execution over time. In the second phase, a performance analysis tool rapidly transforms the CAG to account for effects of a specified communication architecture—such as changes required for topologies, protocols, and CBPM policies. The output includes a transformed CAG like the one in Figure 3. We determine macrostate criticalities as follows:

- 4a. Label each CAG vertex with the macrostate associated with it, as shown in the sample CAG of Figure 3.

- 4b. For each instance s_{ij} of macrostate s_i , delay the start time of the first vertex in s_{ij} by δ . For example, in Figure 3, we delay the start time of the first vertex of macrostate s_4 to $t = 161 + \delta$.
- 4c. Analyze the CAG¹⁶ to estimate this delay's effect, $crit(s_{ij})$, on system performance. The results of this analysis are the values of $crit(s_{ij})$ for all i and j .
- 4d. Obtain $crit(s_i)$, the average criticality of macrostate s_i , by averaging $crit(s_{ij})$ over all j .

Step 4's output is a macrostate criticality table that consists of tuples $[s_i, crit(s_i)]$, representing macrostates and their respective criticalities.

Step 5: CBPM policy definition. In defining a CBPM policy, the objectives are to minimize violations of battery rated current and blocking of performance-critical macrostates. The basic CBPM policy is as follows. Let C be the set of currently executing macrostates. Let S be the set of macrostates associated with pending type II requests. At each CBPM arbitration, select a set of macrostates, $S' \subseteq S$, such that

$$\sum_{x \in S' \cup C} pow(x) \leq P_{\max}$$

$$crit(x) \geq crit(y), \forall (x, y) \mid x \in S', y \in S - S'$$

Clearly power constraint P_{\max} in the first equation determines the extent to which a

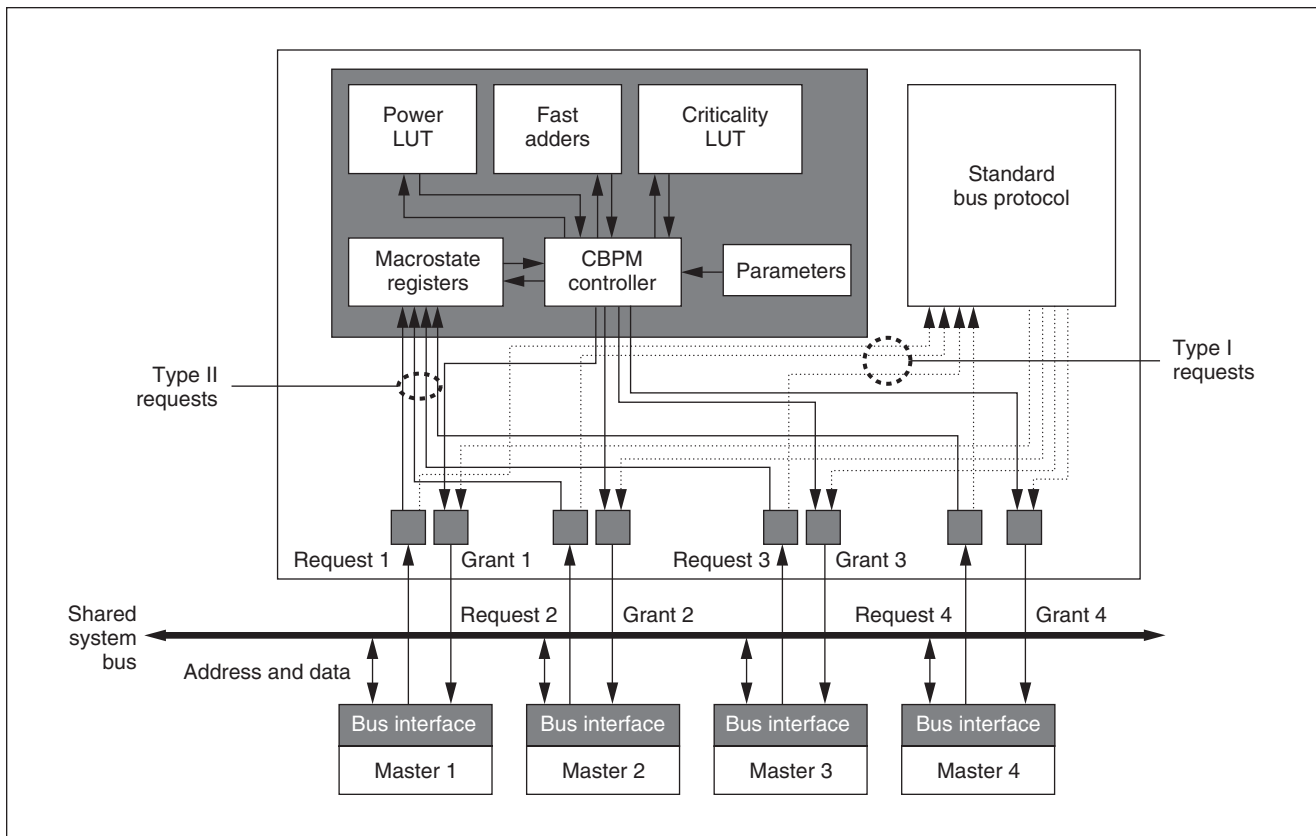


Figure 4. CBPM-enhanced bus controller. Shaded components indicate enhancements for CBPM.

CBPM policy trades off performance for battery life improvements. A large P_{max} leads to the inclusion of more macrostates in S' , resulting in fewer blocked components, enhanced performance, and decreased battery efficiency. A small P_{max} constrains the size of S , inhibiting the number of simultaneously active components and potentially trading off performance for battery efficiency.

The second equation ensures that if the policy cannot grant all the pending macrostates for a given value of P_{max} , it grants execution rights to the most performance-critical subset from the set of pending macrostates.

Step 6: Performance and battery life estimation. Our methodology uses a detailed analysis to examine CBPM's impact on system performance and battery life. For performance analysis, we use the technique described in step 4, which generates a CAG incorporating the CBPM policies' effects. We estimate the system's battery efficiency as follows:

- 6a. From the enhanced CAG, generate a symbolic execution trace of the system, indicating the timing of various macrostates under the selected CBPM policy.
- 6b. Generate the corresponding system power profile from the symbolic execution traces and the macrostate power table.
- 6c. Use the power profiles as input to a battery model, which simulates battery discharge to provide battery capacity and life estimates.

If the system meets desired performance and battery goals, we execute step 7. Otherwise, we modify the CBPM policy by adjusting power constraint P_{max} and repeating steps 4, 5, and 6. In each iteration, a CAG that includes effects of the CBPM policy defined in the previous iteration drives macrostate criticality analysis (step 4).

Step 7: CBPM implementation. In this step, we implement the CBPM policy in the communication architecture illustrated in Figure 4. The

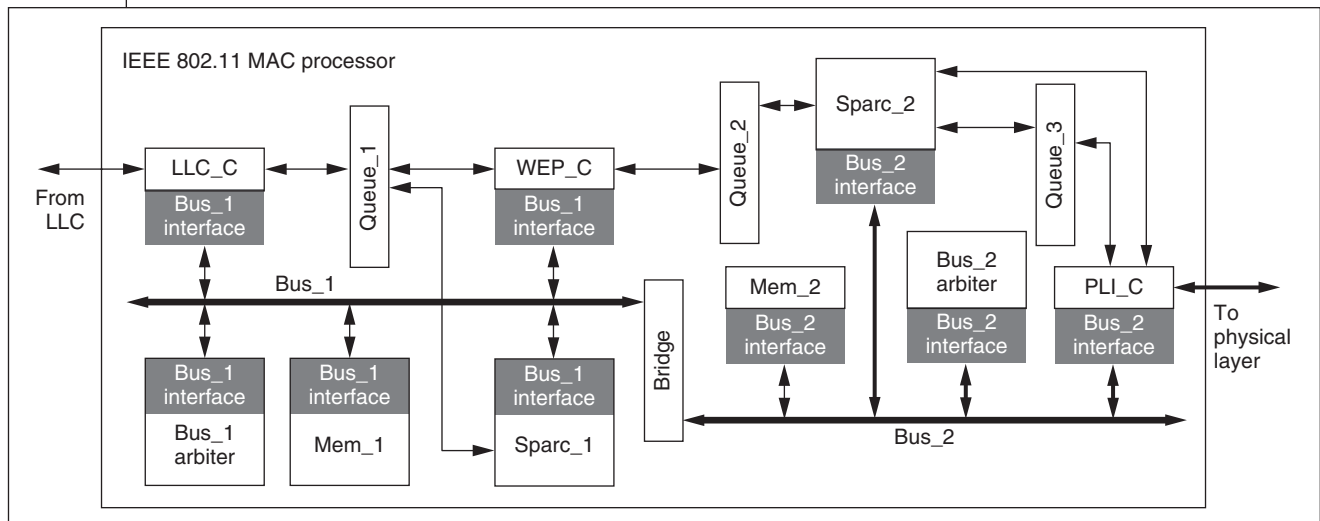


Figure 5. IEEE 802.11 MAC processor architecture.

figure shows a set of four master components connected to a shared system bus. To resolve contention, a bus controller handles bus access requests.

Additional circuitry in the bus controller implements CBPM policies:

- *Macrostate registers.* These registers store a current and pending macrostate identifier for each component.
- *Lookup tables.* One LUT stores the macrostate power table; the other stores the macrostate criticality table.
- *CBPM controller.* Based on the policy defined in step 5, this controller implements the algorithm for handling type II (macrostate transition) requests. At each arbitration, it calculates the total power consumed by the currently executing macrostates. The controller then considers the set of pending macrostates in descending order of criticality by consulting the criticality LUT. Starting with the most critical, the controller grants execution rights to pending macrostates until it runs out of pending macrostates or until granting another macrostate violates the power constraint.
- *Parameter registers.* Additional registers also store settable parameters to provide for run-time configuration. For example, the system can vary the power constraint P_{\max} to dynamically trade off performance for battery life.

Experiments

Our experiments used an IEEE 802.11 MAC processor system.¹⁷ As the “Energy efficiency in wireless communication networks and devices” sidebar by Rao indicates, battery efficiency is a critical challenge in wireless communications.

System

Figure 5 shows the MAC processor architecture, which includes two embedded SparcLite CPUs, three hardware coprocessors, two memories, and three queue buffers. Component **llc_c** is a hardware implementation of the logical-link control (LLC) interface task, which receives frames from the LLC layer, writes them to **mem_1**, and queues frame addresses in **queue_1**. Two components read these addresses:

- **wep_c**, a hardware coprocessor implementing the Wired Equivalent Privacy (WEP) task, which encrypts frame data; and
- **sparc_1**, which computes an integrity checksum vector (ICV) for each frame.

ICV and WEP tasks proceed in parallel. Component **wep_c** writes encrypted frames to **mem_2** and frame addresses to **queue_2**. Embedded CPU **sparc_2** implements three tasks: HDR, which generates the MAC header; FCS, which computes checksums for the frame; and MAC_CTRL, which implements the CSMA/CA algorithm.¹⁴ Component **pli_c**, a

Energy efficiency in wireless communication networks and devices

Ramesh Rao, University of California, San Diego

The reliability and energy capacity of radio communication terminals are crucial to the success of a wireless network product, because greater energy capacity enables longer system lifetimes. Because of the disparity between the advances in battery technology and the evolution of the market for portable communication equipment, software and hardware solutions are necessary to reduce energy consumption and improve system performance.

Several publications have addressed the issue of energy efficiency in wireless communication networks. The proposed approaches fall into two categories: communication protocol solutions and system design techniques. The former includes energy-aware schemes for implementation at different layers of the protocol stack in a wireless system.¹ The latter approach includes power management policies,² energy-scaling algorithms, design of low-power processors, and algorithm partitioning among multiple communication nodes. The work of Anantha Chandrakasan and the μ Amps project (<http://www-mtl.mit.edu/research/icsystems/uamps>) covers these topics.

In general, the network scenarios that have received the greatest attention from the scientific community are ad hoc and sensor networks. These networks typically have more stringent energy limitations than other networks. Moreover, in ad hoc networks, leaving terminals without battery power can lead to topology and partition changes. In sensor networks, a sensor without battery power can stop monitoring a portion of its assigned area, significantly degrading the quality of service that the system provides.

The article presented here falls into the latter category and reminds me of findings about battery behavior that a colleague and I reported earlier.³ This work pointed out

that the amount of energy obtainable from a battery depends on the discharge current's intensity, the power level drained from the cell, and whether the discharge is constant or pulsed. We also presented a stochastic model that mathematically represented the battery behavior in terms of parameters related to the battery's physical characteristics. Park, Savvides, and Srivastava experimentally confirmed these results.⁴ Interdisciplinary work is especially critical in advancing the scientific frontier in energy-efficient communication.

References

1. C. Jones et al., "A Survey of Energy Efficient Network Protocols for Wireless Networks," *ACM Wireless Networks*, vol. 7, no. 4, Sept. 2001, pp. 343-358.
2. T. Simunic et al., "Dynamic Power Management for Portable Systems," *Proc. 6th Ann. Int'l Conf. Mobile Computing and Networking (MobiCom 00)*, ACM Press, New York, 2000, pp. 11-19.
3. C.F. Chiasserini and R.R. Rao, "Importance of a Battery Pulsed Discharge in Portable Radio Devices," *Proc. 5th Ann. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MobiCom 99)*, ACM Press, New York, 1999, pp. 88-95.
4. S. Park, A. Savvides, and M.B. Srivastava, "Battery Capacity Measurement and Analysis Using Lithium Coin Cell Battery," *Int'l Symp. Low-Power Electronics and Design (ISLPED 01)*, ACM Press, New York, 2001, pp. 382-387.

Ramesh Rao is the San Diego Division Director of the California Institute for Telecommunications and Information Technology at the University of California, San Diego. He is also a professor of electrical and computer engineering. Contact him at rrao@ucsd.edu.

hardware implementation of the physical-layer interface (PLI), retrieves frame addresses from **queue_3** and frames from **mem_2**, then passes them to the physical-layer hardware.

System-level communication architecture

Dedicated channels implement low-volume communications, such as synchronization data passed between **sparc_1** and **queue_1**. Two 32-bit shared buses carry larger data transfers. A bridge connects these buses; the buses use a static-priority-based protocol similar to the Peripheral Interconnect Bus (<http://www.omimo.be>).

Components **llc_c**, **wep_c**, **sparc_1**, and **bridge** contend for access to **bus_1**. Components **bridge**, **sparc_2**, and **pli_c** contend for access to **bus_2**. In general, the CBPM methodology also applies to systems with other communication architectures.

Experimental methodology

We designed the MAC processor in the Polis hardware-software codesign environment (<http://www-cad.eecs.berkeley.edu/~polis>), using a combination of Esterel and C to specify the system tasks. For performance

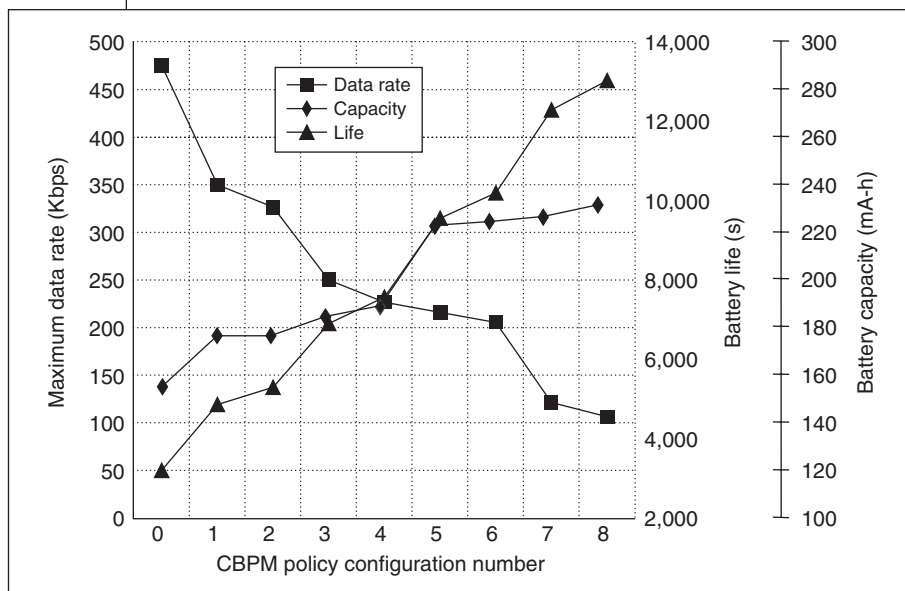


Figure 6. Battery efficiency versus performance for MAC processor configurations based on various CBPM policies.

analysis, we used a tool we described in an earlier publication.¹⁶ A modified version of the power estimation framework described by Lajolo et al.¹⁵ provided system power profiles. We drove simulations using actual IEEE 802.11 traffic, obtained by running *Etherreal* 0.8.18 (<http://www.ethereal.com>), a packet-capture utility, on a laptop connected to a wireless LAN.

Battery efficiency and performance tradeoff

To evaluate CBPM's effectiveness, we experimented with various MAC processor configurations, each having a different power constraint. The input trace consisted of 4,818 frames corresponding to 1.5 minutes of streaming video over an 802.11b-based wireless network, while running Microsoft Windows Media Player (<http://www.microsoft.com/windows/windowsmedia/>) on a laptop.

Figure 6 shows battery capacity, battery life, and data rate for different CBPM configurations. From this figure, we observe the following:

- The CBPM-enhanced MAC processor shows a substantial increase in battery efficiency. Battery life improves from 3,209.2 s to 10,199.9 s (a 3.2-times increase). More importantly, capacity improves from 154.3 mA-h to 225.4 mA-h (a 1.5-times increase).

- Improvements in battery capacity come at a price in terms of system performance. For example, configuration 6 provides a 3-times improvement in battery life, while degrading the data rate by 2.3 times. However, as the next experiment shows, the battery efficiency versus performance tradeoff of CBPM is significantly superior to that achieved by conventional power management techniques.

Comparison with other power management techniques

For this experiment, we considered a MAC processor that supports clock frequency scaling, a commonly used technique for extending battery life in mobile appliances. We first operated the MAC processor over a wide performance range, scaling the system clock frequency over a discrete range of values. For each frequency, we exercised the MAC processor with a long trace of frame arrivals and evaluated battery life and capacity. Next, we operated the CBPM-enhanced MAC processor over the same performance range and measured battery capacity for each point. Figure 7 shows the battery capacity versus performance tradeoff curves obtained by both methods. Examining them, we make the following observations:

- CBPM is superior to frequency scaling as a tool for trading off battery efficiency for performance. For example, to improve battery capacity from 128 mA-h to 182 mA-h (a 42% improvement), frequency scaling incurs a performance degradation of 22%, whereas CBPM's performance overhead is only 7%.
- These results indicate that the frequency scaling approach has a major problem in improving battery efficiency: In some cases, as the frequency decreases (decreasing performance), battery capacity actually decreases. This occurs when the decreasing clock frequency results in the overlap of various power-hungry macro-states that were disjoint at higher frequencies. This overlap leads to vio-

lations of the rated current. We don't observe such anomalies in the CBPM tradeoff curve.

A common dynamic power management technique is idle shutdown, in which the system forces idle components into a low-power state. Recall that our original architecture without CBPM already incorporates dynamic power management. Hence, the reported battery efficiency improvement of the CBPM-enhanced MAC processor is over and above that achievable using dynamic power management.

Hardware complexity

We synthesized and mapped the CBPM-enhanced bus controller to UMC's 0.18-micron standard-cell library (<http://www.umc.com/english/process/>) using Synopsys' Design Compiler (http://www.synopsys.com/products/logic/design_compiler.html). The controller's area was 15,104 μm^2 , a 56% increase over the area of the original bus controller. To investigate the impact of such an increase at the system level, we estimated the total chip area by using fast synthesis for the hardware coprocessors and data sheets for the memories and embedded CPUs. The system-level overhead from CBPM enhancements was less than 0.5%.

The minimum clock period obtained for the CBPM-enhanced bus controller was 4 ns. This implies that the controller can perform single-cycle CBPM arbitrations for bus speeds of up to 250 MHz. It can support significantly higher bus speeds by incorporating multicycle arbitration for less-frequent CBPM arbitrations.

IN THIS WORK, we have established that CBPM is an effective methodology to maximize battery efficiency, and it enables battery life versus performance tradeoffs that are superior to those achievable using conventional power management techniques.

In the future, we will investigate CBPM in the context of emerging on-chip communication net-

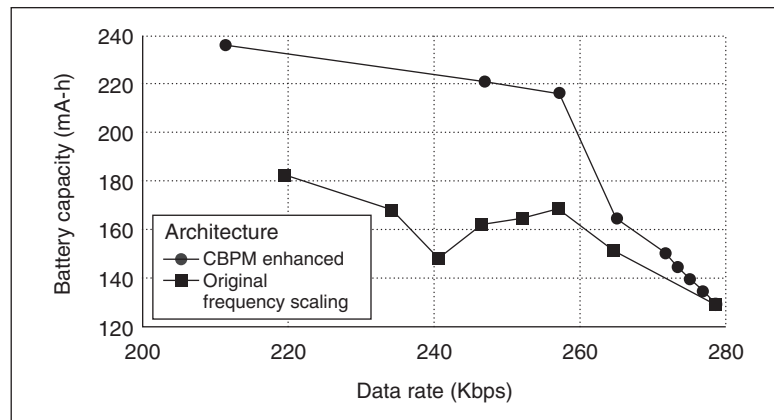


Figure 7. Comparison of tradeoff characteristics for CBPM and frequency scaling.

works. We will also investigate how software (operating system, protocol stacks, and applications) can take advantage of the dynamic configurability that CBPM offers. ■

References

1. I. Buchmann, *Batteries in a Portable World*, Cadex Electronics, Richmond, Canada, 1997.
2. K. Lahiri et al., "Battery-Driven System Design: A New Frontier in Low-Power Design," *Proc. 7th Asia & South Pacific Design Automation Conf. / 15th Int'l Conf. VLSI Design 2002 (ASPAC/VLSID 02)*, IEEE CS Press, Los Alamitos, Calif., 2002, pp. 261-267.
3. A.R. Chandrakasan and R.W. Brodersen, *Low-Power Digital CMOS Design*, Kluwer Academic, Norwell, Mass., 1995.
4. M. Doyle, T.F. Fuller, and J.S. Newman, "Modeling of Galvanostatic Charge and Discharge of Lithium/Polymer/Insertion Cell," *J. Electrochemical Soc.*, vol. 140, June 1993, pp. 1526-1533.
5. D. Rakhmatov and S.B.K. Vrudhula, "An Analytical High-Level Battery Model for Use in Energy Management of Portable Electronic Systems," *Proc. Int'l Conf. Computer-Aided Design (ICCAD 01)*, IEEE CS Press, Los Alamitos, Calif., 2001, pp. 488-493.
6. D. Panigrahi et al., "Battery Life Estimation for Mobile Embedded Systems," *Proc. 14th Int'l Conf. VLSI Design (VLSID 01)*, IEEE CS Press, Los Alamitos, Calif., 2001, pp. 55-63.
7. L. Benini et al., "A Discrete-Time Battery Model for High-Level Power Estimation," *Proc. Design Automation & Test in Europe Conf. (DATE 00)*, IEEE CS Press, Los Alamitos, Calif., 2000, pp. 35-39.

8. T.L. Martin, *Balancing Batteries, Power and Performance: System Issues in CPU Speed-Setting for Mobile Computing*, doctoral dissertation, Dept. Electrical and Computer Eng., Carnegie Mellon Univ., Pittsburgh, 1999.
9. M. Pedram and Q. Wu, "Design Considerations for Battery-Powered Electronics," *Proc. Design Automation Conf. (DAC 99)*, ACM Press, New York, 1999, pp. 861-866.
10. J. Luo and N.K. Jha, "Battery-Aware Static Scheduling for Distributed Real-Time Embedded Systems," *Proc. Design Automation Conf. (DAC 01)*, ACM Press, New York, pp. 444-449.
11. L. Benini et al., "Battery-Driven Dynamic Power Management," *IEEE Design & Test of Computers*, vol. 18, no. 2, Mar.-Apr. 2001, pp. 53-60.
12. Q. Wu, Q. Qiu, and M. Pedram, "An Interleaved Dual-Battery Power Supply for Battery-Powered Electronics," *Proc. 2000 Asia and South Pacific Design Automation Conf. (ASP-DAC 00)*, ACM Press, New York, 2000, pp. 387-390.
13. L. Benini et al., "Extending Lifetime of Portable Systems by Battery Scheduling," *Proc. Design Automation & Test in Europe Conf. (DATE 01)*, IEEE CS Press, Los Alamitos, Calif., 2001, pp. 197-201.
14. IEEE Std. 802.11-1999, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Press, Piscataway, N.J., 1999.
15. M. Lajolo et al., "Efficient Power Co-Estimation Techniques for System-on-Chip Design," *Proc. Design Automation & Test in Europe Conf. (DATE 00)*, IEEE CS Press, Los Alamitos, Calif., 2000, pp. 27-34.
16. K. Lahiri, A. Raghunathan, and S. Dey, "System Level Performance Analysis for Designing On-Chip Communication Architectures," *IEEE Trans. Computer-Aided Design*, vol. 20, no. 6, June 2001, pp. 768-783.
17. K. Lahiri, A. Raghunathan, and S. Dey, "Battery Efficient Architecture for an 802.11 MAC Processor," *Proc. IEEE Int'l Conf. Comm.*, IEEE Press, Piscataway, N.J., 2002.



Kanishka Lahiri is a doctoral candidate in the Electrical and Computer Engineering Department at the University of California, San Diego. His research interests

include design methodologies and architectures for embedded systems, focusing on system-level performance analysis, power estimation, and architectures for high-performance on-chip communication. Lahiri has a BTech in computer science and engineering from the Indian Institute of Technology, Kharagpur, India; and an MS in electrical and computer engineering from the University of California, San Diego. He is a student member of the IEEE.



Anand Raghunathan is a senior research staff member at the NEC USA Computers & Communications Research Laboratories (CCRL) in Princeton, N.J. His research

interests include system-on-a-chip architectures, design methodologies, and design tools, with emphasis on high-performance, low-power, and testable designs. Raghunathan has a BTech in electrical and electronics engineering from the Indian Institute of Technology, Madras, India; and an MA and PhD in electrical engineering from Princeton University. He is a senior member of the IEEE, golden core member of the IEEE Computer Society, and vice chair of the Tutorials and Education group for the IEEE Computer Society's Test Technology Technical Council.



Sujit Dey is a professor in the Electrical and Computer Engineering Department at the University of California, San Diego. His research interests include configurable

platforms, comprising adaptive wireless protocols and algorithms, and deep-submicron adaptive systems-on-chips for next-generation wireless appliances and network infrastructure devices. Dey has a PhD in computer science from Duke University. He is a member of the IEEE.

■ Direct questions and comments about this article to Kanishka Lahiri, 3927A Miramar St., La Jolla, CA 92037; klahiri@ece.ucsd.edu.