Title
The Low Power Energy Aware Processing (LEAP) Embedded Networked Sensor System

Authors
Dustin McIntire
Kei Ho
Bernie Yip
et al.

Publication Date
2005

The Low Power Energy Aware Processing (LEAP) Embedded Networked Sensor System

Dustin McIntire, Kei Ho, Bernie Yip, Amarjeet Singh, Winston Wu, and William J. Kaiser
Electrical Engineering Department
University of California, Los Angeles
Los Angeles, California

***Abstract*— A broad range of embedded networked sensor (ENS) systems for critical environmental monitoring applications now require complex, high peak power dissipating sensor devices, as well as on-demand high performance computing and high bandwidth communication. Embedded computing demands for these new platforms include support for computationally intensive image and signal processing as well as optimization and statistical computing. To meet these new requirements while maintaining critical support for low energy operation, a new multiprocessor node hardware and software architecture, Low Power Energy Aware Processing (LEAP), has been developed. This architecture integrates fine-grained energy dissipation monitoring and sophisticated power control scheduling for all subsystems including sensor subsystems. The LEAP architecture enables complex energy-aware algorithm design by providing a simple interface to control numerous platform and sensor power modes and report detailed energy usage information. This paper also describes experimental results of a new distributed node testbed based on LEAP demonstrating that by exploiting high energy efficiency components and enabling proper on-demand scheduling, the LEAP architecture meets both sensing performance and energy dissipation objectives for a broad class of applications. This testbed including the network of distributed LEAP nodes and a system producing physical, mobile events provides a development environment for LEAP-hosted algorithms. New design principles, detailed implementation, and in-network programming and remote debugging capabilities of this platform are also described. While this is the first report of the LEAP system, it has been deployed for nearly one year with 50 users developing energy aware systems.**

Keywords-embedded wireless networked sensor, energy-aware multprocessor platform, sensor platform hardware and software architecture

I. INTRODUCTION

A broad range of embedded networked sensor (ENS) systems for important environmental monitoring [1,2] and other applications now require advanced capabilities to support high power sensor devices such as imaging devices. Many of these applications also require support for on-demand high performance computing and communication for complex information processing. This includes image processing, statistical computing, and optimization algorithms required for selection of proper sensor sampling [3]. Prior development of ENS platforms has resulted in low power systems well matched to the requirements for supporting low power sensor devices (for example, thermistor transducers for temperature sensing or photodiode sensors for light level sensing). The computing demands for such systems were matched to low data rate and low complexity sensors [4-8]. However, prior ENS platforms designed to support micropower sensor devices are not adapted to system level energy minimization for a new, expanded set of ENS requirements in environmental monitoring applications, ranging from ecosystem monitoring to public health monitoring, and security applications. These applications have large sensor and instrument device power dissipation (specifically with peak power levels far in excess of the ENS node computing and communication power levels). At the same time, computing and communication demands are also advanced in order to support the on-demand processing associated with these complex sensors. While performance needs have increased, it is still critical to minimize system energy dissipation. Solutions require both hardware and software architectural changes to enable this. By exploiting a new architecture and environmental phenomena characteristics, both advanced performance and low energy can be achieved.

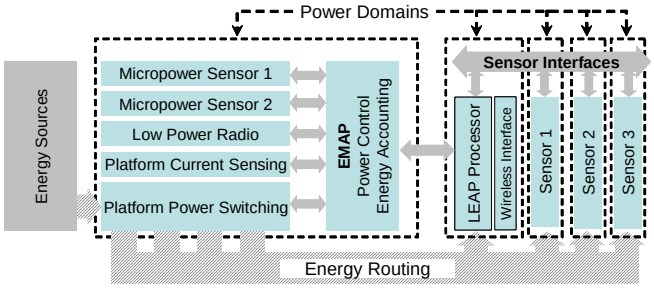

Figure 1. LEAP ENS architecture showing the Energy Management and Accounting Preprocessor (EMAP) and its defined power domains (shown in shaded rectangles). Energy routing and data interfaces are indicated.

To address these diverse applications, a new design approach is required. This must include the system's sampling schedules and computational demands *required* to meet information acquisition requirements established by the application. Subject to the sensor selection and sampling constraints, platform system operation must be optimized to minimize energy. This requires a design approach that focuses on minimizing energy required for *each* sensing, computing, and communication *task*. The approach leads to the new Low Power Energy Aware Processing (LEAP) multiprocessor architecture for ENS nodes. LEAP is based on hardware and software system partitioning specifically adapted to these new requirements.

LEAP, shown in Figure 1, includes an essential capability for independent energy monitoring and power control for each subsystem. The LEAP architecture has been developed to harness the use of properly scheduled, energy efficient multiprocessor components selected to achieve the lowest per task operating energy. It is partitioned such that high efficiency, high power components (used on demand) are assigned to a LEAP processor partition while continuously vigilant micropower components are assigned to a LEAP preprocessor partition. The Energy Management and Accounting Preprocessor (EMAP) provides fine-grained monitoring and control of energy dissipation in all ENS subsystems. Additionally it schedules operation and power delivery to sensor systems and the LEAP's host processor. Finally, while EMAP enables the entire LEAP system to operate at micropower vigilance, it also provides event detection and

triggering capability. This allows event-triggered transition to states where sensors and computing systems are available on-demand according to schedules that match application sampling requirements.

The LEAP architecture with the EMAP preprocessor further partitions ENS node subsystems into separately managed power domains supporting individual components (for example, individual sensor devices and processors). Scheduling operation within power domains enables the LEAP system to define a broad range of power modes that are then matched to environmental monitoring demands. This allows users to develop systems with application specific operating modes intended to meet the minimum energy required for information acquisition subject to specific sensor system and sampling requirements of that application.

The LEAP hardware architecture is combined with a software architecture providing developer access to system energy monitoring and management along with subsystem operation scheduling. Experimental results verify that this enables convenient developer access and promotes development of energy aware systems. It also provides an advance for in-network programmability and remote debugging of all the components.

The design approach, high energy efficiency component selection methods, operation scheduling, and development of the LEAP platform are described in Sections II- V. Also, detailed experimental verification of LEAP operational capability is described in Section VI with a testbed system that supports complex sensors operating on-demand and displaying power demand varying by four orders of magnitude during event detection operations. This new experimental testbed combines distributed LEAP nodes along with physical environmental event generation systems presenting the distributed system with accurately scheduled events for performance evaluation. It has specifically demonstrated that LEAP on-demand scheduling of high energy efficiency components enables algorithms that self-adapt to event behavior and may adjust operational schedules to minimize energy dissipation for a specific detection objective. These experimental results and experience with many recent users of the platform also demonstrate the convenient development path for supporting LEAP applications. As will be described, while this is the first report of the LEAP system, it has been in use for nearly one year with 50 users successfully developing diverse energy aware systems on the distributed testbed. Prototype LEAP based systems have also been deployed for ecosystem monitoring. Open source release of LEAP hardware and software is available [9].

II. Limitations of Current platforms

Currently, in order to approach energy dissipation levels consistent with long term deployment, wireless sensors based on microcontroller architectures are often employed as described in [4-8]. However, as will be discussed, while still applicable for certain applications, these microcontroller-based systems (operating alone) do not satisfactorily support computationally demanding applications such as multiple object recognition and tracking via imaging. Nor are these platforms matched to high peak power dissipation sensors that must be scheduled for on-demand use.

Early ENS node platforms were designed to support low power dissipation sensors. These included, for example, geophone seismic sensors, or in the case of microclimate sensing, temperature, humidity and light sensors, and for security applications, microphones and magnetometers. It is important to note that these sensor elements share the common characteristics of not requiring substantial energy to support their operation. Specifically, this energy dissipation is less than that of the node platform itself. Indeed for some sensors, for example the photodiode, no energy source is required, and only proper preamplification and analog-to-digital data conversion energy dissipation is required for sensor support. These sensor systems also share the additional characteristic of producing a simple (scalar) output that may be sampled at low rate imposing limited computing demand that small microcontrollers can support.

However, emerging applications in environmental monitoring, science and public health, and security now require capable sensors, such as imaging to detect and identify events, and high performance chemical sensors to detect contamination in atmospheric and aquatic systems. These share the characteristics of high peak operating power, well in excess of that of the platform itself, but may not be operating during the entire application schedule. These new sensors also place demands on the ENS platform computation required to extract event information and to schedule adaptive sampling. Thus, it is now important to provide a platform that supports yet higher capability sensing and computation, while maintaining low average energy operation.

III. Design Requirements and Design Approach

The LEAP ENS platform design is developed to meet a set of design requirements derived directly from the current generation of environmental monitoring applications. These include requirements for computational resources, communication subsystems including wired and wireless interfaces, sensor interfaces, sensor energy dissipation measurement and control, local data storage, remote software debugging capability, and remote reprogrammability. Each of these design topics will be discussed further in the following sections.

A. Design Requirements

The LEAP design approach exploits the characteristics of environmental phenomena that permit sampling to occur at low rates or in an event-triggered fashion for a broad set of applications. For example, environmental imaging systems may only be required to operate infrequently (according to events detected using other sensing modalities or schedules). Many atmospheric and aquatic phenomena display slow rate of change and may be critically sampled at low rate or again according to events. This also implies that ENS sensor, processor, and other components may also be employed on-demand and only infrequently used. Thus, a new ENS platform intended to support the complete set of environmental monitoring applications must differ from previous ENS systems by introducing methods for proper scheduling of sensing, computing, and communication tasks.

B. Design Approach: Computing Platform

Many deployments demonstrate that ENS systems supporting environmental monitoring may operate in multiple modes while serving specific applications. For many applications a vigilant state is required to permit continuous phenomena monitoring. In this state minimum energy dissipation is critical since system may spend the majority of its time here. However, in these periods, signal processing and communication occurs at low duty cycle since the lack of events requires only infrequent coordination activities. Conversely, there are periods of high activity when background information indicates that an event of interest may be occurring. In this high vigilance state, computational requirements can be extensive and often with real-time constraints to support high performance sensor interfaces. Real-time demands may require large computing resources to meet all deadlines. In this phase energy efficiency of processor computation will be more important than short term average power since a task operation is bounded in time and the energy required to complete the task ultimately determines the contribution to platform energy usage. The desired effect is to minimize *total energy* used during these brief task activity periods rather than to minimize *peak power*. As will be seen, this results in architecture and component selection differing from that of prior work.

The LEAP design approach for computing platform selection begins with benchmark characterization of task energy efficiency for those operations required by a typical ENS platform. Three example benchmarks are described here. The first benchmark, the cyclic redundancy check (CRC), tests efficiency for executing ubiquitous error detection and correction algorithm tasks. The second benchmark tests the typical ENS requirements for digital filtering of sensor data streams using a finite impulse response (FIR) filter. The third benchmark tests energy efficiency for Fast Fourier Transform (FFT) data transformations on sampled data including images.

TABLE I. COMPARISON OF MICROPROCESSOR COMPONENTS

Benchmark	Array Size	Data Size	Platform	Execution Time (micro sec)	Energy (mJ)	**Relative Efficiency**
CRC-32	1024	8-bit	Stargate	24.8	0.013	**28.4**
			MICA2	5150	0.367	**1**
FIR	256	32-bit	Stargate	325	0.167	**70.7**
			MICA2	16,800	11.800	**1**
FFT	128	16-bit	Stargate	94.5	0.046	**20.4**
			MICA2	14,500	0.934	**1**

Candidate LEAP components were characterized by direct operational measurements on each processor using identical C code algorithms. Supply current was monitored by a digital sampling oscilloscope across a precision sense resistor. Benchmark execution times were indicated with minimal latency by toggling a processor I/O pin at the benchmark start and completion. Two platforms are compared in this experiment, an Intel Stargate platform [10] based on the Intel PXA 255 processor and the Crossbow MICA2 platform [11] based on Atmel ATmega128L microcontroller.

Results of platform comparison summarized in Table I show that the selection of the high performance processor option results in dramatically reduced energy usage associated with an individual computing task. Thus, this heavily favors the LEAP design approach where such a processor is used on demand for execution of specific tasks and is otherwise operating in a low power or disabled (no applied power) state. Addressing the diversity of computing constraints, LEAP chooses a heterogeneous multiprocessor solution as has been suggested in [12-13]. By utilizing multiple processors, LEAP selects a solution adapted to varying sensing needs.

C. Design Approach: Communication Interfaces

Experience in ENS platform deployments for environmental monitoring demonstrates selection of wireless interfaces should benefit not only energy efficient internode communication, but also integration with existing deployed wireless infrastructure. Analogous to the diverse computational requirements described above, the wireless communications subsystem requires both a low-power, low-bandwidth, network paging system to remain active for extended intervals as well as a high bit rate data transport system for increased vigilance periods. In order to maintain communications compatibility with existing sensor platforms as well as to integrate with common wireless infrastructure, we chose a dual radio approach. Similar to the processing subsystem, the wireless communications subsystem must have the capability for both low power operation and for highly energy efficient bulk data transfer.

Energy efficiency analysis of widely used wireless devices was performed based on supplier measurements and our measurements of system energy dissipation for a range of broadband and narrow band devices. Results comparing the energy to transmit and receive data per bit using 802.11g and 802.15.4 standard devices demonstrate that with equal link margins, the 802.11g interface is approximately 9 times more energy efficient than its 802.15.4 alternative. However, the idle power in receive mode of the 802.11g solution is approximately 14 times greater than the 802.15.4 solution. Thus LEAP adopts a dual radio solution for both low power and high efficiency with the EMAP preprocessor including the Chipcon CC2420 802.15.4 radio compatible with numerous other existing low power platforms [8] along with standard 802.11 interfaces hosted by the high efficiency processor.

TABLE II. COMPARISON OF WIRELESS INTERFACE COMPONENTS

	802.11g	**802.15.4**
Chipset	Atheros 5006XS	CC2420
Output Power	16dbm	0dbm
Rx Sensitivity	-78dbm@36Mbps	-90dbm@250Kbps
Tx Power (Max Output)	1320mW	57.42mW
Rx Power	924mW	65.01mW
Total Power	2.24W	122.43mW
Effective Throughput	20Mbps	125Kbps
Efficiency (nJ/bit)	112	979

D. Design Approach: Sensing System Support and Interfaces

As environmental sensing becomes increasingly ubiquitous, ENS platforms must adapt to incorporate their wide ranging set of interfaces as well as per sensor energy monitoring and control. These have been provided on the PXA processor platform along with sensor interfaces ranging from analog (for simple low power devices) to RS-232, I2C, and SPI serial, and then finally to high speed interfaces including USB and Ethernet. Again for energy efficiency and high performance, high bandwidth sensors benefit from direct access to the host processor's memory subsystem. To enable these sensors, the LEAP system provides direct access to the PXA processor external memory bus and provides direct sensor to memory DMA to offload processing overhead. This is the most efficient means of data transfer as it eliminates any bus protocol controller or extraneous data copies.

E. Design Approach: Storage

Environmental monitoring deployments also demonstrate that platform local storage is a critical design requirement to support unattended and unserviced long term ENS node operation. Indeed, through addition of storage, high energy cost communication episodes may be scheduled to occur at times optimal for data transport. The LEAP design is directed to enable application developers to optimally select data allocation strategy to various memory types and to directly measure resulting energy and performance.

F. Design Approach: Energy Monitoring and Management

Development and deployment of ENS systems has demonstrated that the optimal choice of sensor system, processor, wireless interface, and memory technology is not only application dependent, but may also exhibit temporal dependence for a given application. For example, as in the experimental example to be discussed below, a target tracking system may display large resource demand for initial target acquisition, but, may otherwise operate with reduced resource demand when updating target bearing. The optimization problem is observed to be further compounded for multi-user, multi-application systems operating with different application objectives.

A primary challenge for fundamental ENS algorithm and application development is the allocation of shared resources including computing, storage, and communication, as well as critical energy and sensor systems. Advances in balancing users demands has been developed in operating systems design [14-16]. However, a critical hindrance to this development and application to ENS systems has been the lack of hardware support for system resource monitoring and management. Clearly, fine-grained device level monitoring and control must be included in the ENS design approach as has been considered for conventional embedded systems [17-18]. However, in the past, the lack of real time data has forced reliance only on off-line profile data to estimate an algorithm's performance.

The LEAP design approach provides this capability in the EMAP processor by partitioning devices into many power domains with the capability to monitor, enable or disable power to each domain, as well as to respond to trigger events or conditions that restore or remove power in each domain. The energy accounting information collected by the EMAP is periodically transferred to the host processor and a power management schedule provided by the host processor may be delivered to the EMAP for each power domain.

G. *Design Approach: Remote Access and Debugging*

Experience in ENS system deployment demonstrates that rapid development of algorithms and implementations in deployed systems becomes increasingly important. Additionally, multi-user systems require periodic retasking. The ENS platform design then must include transport and verification of software updates and application of new executable code images during unattended, remote, and uninterrupted platform operation. In addition, remote debugging of each component of the multiprocessor system is required. As will be described, LEAP includes remote reprogrammability of each storage element and debugging of the host processor and EMAP.

IV. LEAP HARDWARE ARCHITECTURE

The LEAP platform architecture is partitioned into a general purpose computing module with its associated memory systems and interfaces, and a preprocessor module dedicated to low power sensing, energy accounting, and power domain scheduling. These hardware modules will be discussed in the following section.

A. *Slauson Processor Module*

The Slauson processor module (SPM) shown in Figure 2, is based on the Sensoria Slauson platform [19]. The SPM contains a PXA255 400Mhz processor and is populated with an SDRAM bank and an Intel K3 Strataflash flash bank of up to 128MB and 64MB, respectively. The core processor and memory subsystem may be suspended with either 1, 2, or all 4 of the mobile SDRAM's memory banks preserved during the suspend state for reduced leakage current while in self refresh.

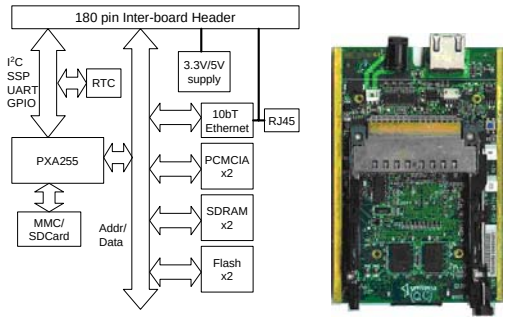

Figure 2. Processor Platform Block Diagram (left) and Image (right)

In addition to the processor and memory components, the SPM has dual PCMCIA interfaces configurable for either 3.3V or 5V devices. Each of the PCMCIA slots may be independently isolated and powered down. Local non-volatile data storage may be expanded with the addition of a memory device into the SPM's SD Card socket. Communications are provided by an on-board SMC9196 10baseT Ethernet controller (a 10baseT chipset was selected instead of the more common 10/100 chipset due to the increased idle power needed by the 10/100 chipset operating at higher clock rate). This Ethernet controller may be suspended by software or hardware controls. Table III describes the SPM operating states and current requirements.

The SPM includes an extensive set of interfaces via a 180 pin inter-board header including serial buses such as two RS232 ports, two SPI ports, I2C, and AC97 audio sampling ports. Additionally, the full parallel memory bus is available with a total of 192MB of memory space. Full memory-to-memory DMA transfers are possible via either the PXA's internal DMA or an external DMA controller. Power is distributed via the inter-board connector as will be described. Time synchronization options include an on board real time clock as well as accessible inputs for external GPS synchronization. Various SPM power states are shown in Table III.

B. *EMAP Module*

EMAP preprocessor, shown in Figure 3, developed by these authors for ENS applications is mated to the SPM via the inter-board connector. The EMAP utilizes a Texas Instruments MSP430F1611 microcontroller. This version of the MSP430 processor was chosen for its large on chip RAM space (10KB) and its set of hardware peripherals.

TABLE III. LEAP SPM CURRENT REQUIREMENTS FOR SEVERAL OPERATING MODES

SPM Power State	Supply Current at 5V
Suspend	5mA
Operating System Idle Task	41mA
Operating System Idle with SD Card Based Filesystem	50mA
Operating System Idle with Ethernet without traffic	48mA
iperf execution with Ethernet at maximum throughput	140mA
Execution at 100 percent computing load	165mA

The EMAP allows the LEAP system to be subdivided into 5 power domains. Each domain is independently powered and isolated. Power is supplied to each domain through a low-resistance current sensing resistor. Detection of current is by differential high common mode rejection ratio current sense amplifiers followed by antialiasing filters. Five of the MSP430 internal 12-bit ADC inputs sample the currents in each domain and current values are integrated to obtain charge values. For each power domain, the EMAP exploits the power off state to perform sense amplifier offset correction for the respective domain.

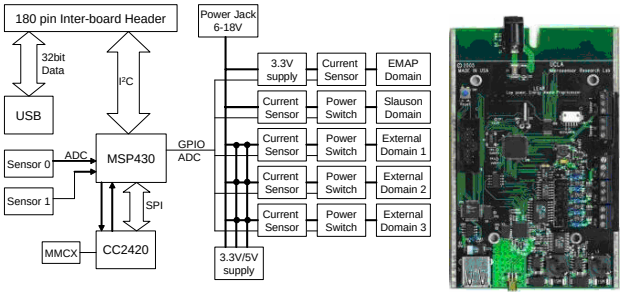

Figure 3. EMAP Preprocessor Block Diagram (left) and image (right)

1) *EMAP Module - Energy Measurement, Management, and Sensor Interfaces*

The EMAP's power domains may be allocated according to platform requirements. For the results described here these are allocated to 1) EMAP module, 2) Slauson processor module, 3) up to three external sensor systems (including in the experimental results here, the imager). EMAP power supply rails, either 3.3 or 5V, are jumper selectable. The EMAP may power down unused voltage rails to eliminate the quiescent current draw of the voltage regulator. Up to 2A current is available for high power sensors.

The SPM may request the most recent charge accumulation values from the EMAP. The EMAP will respond with each domain's voltage rail selection, sense amplifier offset value, and sense amplifier gain

constants in addition to the accumulated sum values. The host processor then may accurately compute integrated charge and energy for each of the power domains.

In addition to detailed energy monitoring, the EMAP provides a power management scheduling capability. Each of the power domains is electrically isolated from one another when powered off. This is critical since current leakage paths (for example via current conveyed by input protection diodes) to ground may appear when nonisolated systems are operated in a suspend condition. Current inrush limiters also protect the LEAP system from individual domain current inrush and resulting supply voltage droop when enabling domains.

Two low voltage analog sensor inputs are provided with 12bit ADC inputs operating at sampling rates up to 10 kHz, and with a 3.0V precision voltage reference. These ADC inputs may be connected to a variety of low bandwidth sensors for simple event detection.

2) EMAP Module – Processor Communication

The MSP's dual USART controllers may be configured to support I2C, SPI, or UART serial protocols. I2C is chosen for inter-board communications since it provides a multi-master capability with implicit bus arbitration. This permits convenient expansion of the LEAP platform to include multiple EMAP modules and multiple high-performance processors. The LEAP system is implemented with the SPM and EMAP processors operating as I2C peers. Either device may initiate transactions with the other or with any other device. The second MSP USART controller is configured as an SPI master for access to the CC2420 radio. An MMCX external antenna connector is included.

3) EMAP Module - Low Power Operation

For purposes of energy and performance control, the MSP's CPU frequency may be controlled from 100KHz to 8MHz or fixed under software control by an external crystal. Further, the EMAP hardware and software has been designed to provide various power modes. The MSP enters the LPM3 power state when running the operating system's idle thread. Further, a suspend (LPM4) state may be entered through software. The MSP processor and all EMAP peripherals are disabled. The system wakes only due to a sensor signal transition. These states are shown in Table IV.

TABLE IV. LEAP EMAP CURRENT REQUIREMENTS FOR SEVERAL OPERATING MODES

EMAP Power State	Supply Current at 5V
Suspend (LPM4)	385µA
Current sensing task and idle task (LPM3)	419µA
Previous with all power domains enabled	862µA
Previous with 1.8V RF power enabled	950µA
Previous with all power domains and supplies enabled	7.6mA

4) EMAP Module - Remote Debugging

To facilitate remote software upgrade and source level debugging, the MSP processor's JTAG interface has been provided to the SPM's PXA processor through the inter-board connector. This allows the SPM to assume control of the MSP processor's execution, to program internal flash, and to perform any action on the MSP's I/O pins. The host processor acts as the proxy agent for a remote user's debugging system. Debugging commands that are issued by a remote user's debugger are routed to the host processor and converted to JTAG command sequences. The software requirements for remote debugging will be discussed in a later section.

V. LEAP SOFTWARE ARCHITECTURE

A. SPM PXA Processor

In order to support the increasing demand to host complex applications, such as the EmStar runtime environment [20] and R statistical computing package [3], the LEAP software framework is comprised of multiple tiers to ensure dependable operation. It is designed to allow recovery from many common faults. The tiers are described below in the order at which they appear at boot time.

The first LEAP software tier is the system bootloader, Redboot (a configuration of the eCos real time operating system) [22]. This provides methods for flash memory manipulation, support for remote file retrieval, and loading and execution of other operating systems. The LEAP bootloader itself may be updated with RTOS library elements or completely replaced over remote links. Boot commands are stored in flash-based configuration script and boot the Linux operating system, the next tier. The Slauson PXA processor supports the Linux 2.6 version kernel, the second tier, compiled with module support for device drivers, network protocols, and power management.

The third tier to appear at boot time is a compressed, read-only cramfs filesystem [23] containing the Busybox [24] utility set. At boot time, this tier validates the integrity of the read-write filesystem composed of a JFFS2 image.

Upon validation of the JFFS2 image, the cramfs image transfers control to the fourth software tier starting the init program located on the JFFS2 filesystem. The JFFS2 image contains the standard Linux directory structure and boot scripts and a larger set of filesystem utilities and libraries linked against the glibc library.

This tier also includes the SPM to EMAP communication system. Applications operating over Linux on the SPM access EMAP utility functions that enable interaction with the EMAP via I2C communication. A utility, msp-client, provides a convenient interface with both an interactive model for development and testing, as well as a command sequence model. EMAP control provided by msp-client includes access to sensor data, energy and charge data, and power control for all domains. Sensor data access provides the ability to measure instantaneous, average, peak, minimum, and other signal attributes.

The msp-client command set also includes a powerful control interface to set, query, and modify power management schedules on a per power domain basis. These schedules may become arbitrarily complex by setting future start times, repeat period, power domain, and power management action. Each power management command issued by msp-client to the EMAP is assigned a unique key permitting additional msp-client access to observe and manipulate schedules. It is important to note that not only may sensor resources be scheduled for power, but the SPM may be powered down (or placed in suspend) to conserve energy and then to be re-enabled at a future time according to a scheduled action.

Finally, msp-client also includes many functions that permit control of system response to trigger events that may be set and manipulated. These are based on sensor input signals and may trigger an EMAP action to enable the SPM or another power domain.

B. EMAP MSP Microcontroller

Many operating systems are available for the SPM's PXA processor and the EMAP's MSP microcontroller. The EMAP's experimental results reported here are obtained using a traditional priority driven, preemptive RTOS known as uC/OS [24]. This operating system was chosen to meet development requirements including supporting source level debugging, such as through the GNU debugger, gdb.

The EMAP software architecture includes a set of uC/OS objects designed for compatibility with the small MSP memory footprint. The workload for our design was partitioned into the following software tasks: host processor communication using I2C messages, power

management scheduling, sensor interface monitoring and threshold triggering, and CC2420 radio communications.

Figure 4 shows the EMAP software components. First, the interface to the MSP microcontroller is supported by I2C and SPI drivers noted above. The I2C device driver layer performs both blocking and nonblocking operations to the I2C hardware. Atomic hardware access is maintained by use of a guard semaphore. Tasks requesting blocking reads or writes may set optional timeout values to prevent deadlock occurrences. The I2C device driver utilizes the MSP's DMA hardware to transfer data blocks to and from the I2C hardware controller unit. Upon completion of the requested transfer, the DMA's ISR is run which signals the I2C device driver of the completed event by posting to the driver's transaction complete semaphore. This will unblock the device driver and allow the waiting task's read or write request to complete.

Using the MSP's DMA substantially reduces the interrupt overhead incurred versus doing single byte transfers through programmed I/O operations. Similarly, blocking tasks with a semaphore when waiting for DMA hardware to complete allows other tasks to run during the blocked period or for the CPU to shutdown to reduce power.

The I2C messaging task is responsible for receiving and composing messages to and from the host processor. It communicates with the msp-client application mentioned in the previous section. This task utilizes the I2C device driver layer to provide master and slave read and writes through MSP's I2C hardware controller.

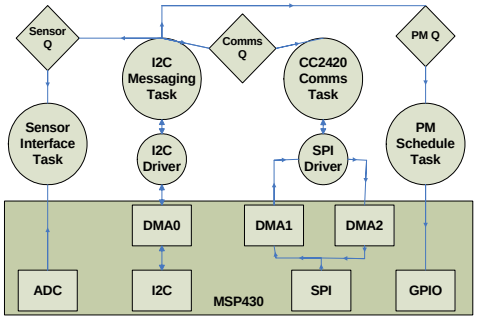

Figure 4. EMAP software architecture showing uC/OS objects operating over the MSP hardware devices in the shaded rectangle.

All commands received from the msp-client application are parsed within the I2C messaging task and any reply messages are composed, and returned. The I2C messaging task will handle messages that do not require specialized actions without waking other threads including read back of system settings or of charge accumulation values.

The power management task is responsible for processing and scheduling all power management commands issued by the msp-client application. When a new msp-client message arrives it is processed by the I2C messaging task. If that command is a power management command, it is added to the power management task's list of actions and the task is signaled to run. The power management task then chronologically sorts all new and existing power management actions in the scheduled actions list. By chronologically sorting the list, it can be parsed quickly for expired actions. The power management task then suspends itself until the time of the first scheduled action or until a new power management command arrives. If no new command message arrives, the action's delay timer expires. The power management task then examines the scheduled action list looking for runnable actions. An action is runnable when it is on the scheduled actions list and it is in the past. The power management task removes each runnable action from the actions list. The action, which can be to enable or disable power, or to trigger a wakeup, is performed on the specified power domain. If the action is not periodic, the action object is recycled to the free actions list. If the action is marked as periodic, then the period interval is added to the actions scheduled run time and the action is reinserted into the actions list.

The sampling and triggering task is responsible for setting the periodic sample rates for the on chip ADC channels connected to the two external sensor inputs and to set the wakeup trigger threshold values. Sensor sampling and threshold settings commands issued by msp-client are first parsed by the I2C messaging task. Relevant messages to the sampling and triggering task will wake this thread to process the message contents. The sampling and triggering task checks the validity of sensor sampling commands such as assuring that the selected sensor sample rate is compatible with the charge accumulation sampling rate. This task is also responsible for resetting triggered wakeup actions and setting sensor threshold values and detection edges. Should this task be run by the ADC interrupt handler due to a threshold excursion event, it will perform all wakeup actions on power domains that are registered for wakeup event handling.

The last EMAP task is for communication through the CC2420 low-power radio. The CC2420 communications thread utilizes the SPI driver for reads and writes to the CC2420's data port. Like the I2C driver the SPI driver uses DMA for data transfer. Unlike I2C, the SPI bus operates in full duplex. To support this, two DMA channels are necessary. One DMA channel reads incoming data from the SPI receive buffer while the second directs outbound data into the transmit buffer. The CC2420 communications task leverages a CC2420 utility library that abstracts the CC2420 into basic access functions such as power mode settings and to transmit or receive packets.

VI. Experimental Results

Experience with development of applications in energy aware environmental monitoring using LEAP demonstrate that a broad range of algorithms may be classified into reactive, proactive, and hybrid methods. Reactive algorithms respond to external events captured by sensor data (and then trigger operation of high peak power and high performance LEAP subsystems) while proactive methods attempt to estimate an event arrival in advance such that the system is able to perform some action without the latency associated with response to trigger events. Hybrid algorithms combine the reactive and proactive approaches in any combination. Further, it is important to note that multiple LEAP nodes contribute to detection and tracking of phenomena, vastly expanding the capability for achieving both low operating energy and high detection and tracking performance.

Algorithm selection depends largely on the phenomenon of interest as well as sensor platform capabilities. Reactive algorithms may be well suited to applications where the sensors and sensor platforms are highly agile in time and energy usage or where the sensed phenomena are poorly understood. Alternatively, proactive approaches may suit less agile sensors and sensor platforms or where the sensed phenomena are well understood and may be predictable with sufficient certainty over short time periods. These defining characteristics may even evolve over time as a system learns environment characteristics. The LEAP system may support each algorithmic class with resource management implemented to permit the lowest operating power consistent with sensing requirements.

A. LEAP Testbed

An experimental system is required to enable detailed characterization of each algorithm with respect to energy and distributed sensing performance. This has been accomplished by deploying many LEAP based nodes in a distributed network, each supporting multiple sensor inputs for environmental event detection. In addition, a new testbed has been developed that provides accurately reproducible physical events that may be detected both by the

micropower, constantly vigilant sensors as well as by the on-demand use of high performance imaging devices supported by each LEAP based node. The testbed includes six distributed nodes each supporting 1) an SMC2532 802.11b wireless interface, 2) a SNC-RC30N high performance embedded networked cameras capable of zoom, pan, and tilt operating in a sensor power domain, and 3) a photodiode sampled by the EMAP ADC. The photodiode measures only light intensity and does not enable localization or color identification.

B. *Event Generator*

An essential testbed component is a physical event generator producing a moving target signal that may be detected using imaging sensors as well as using a limited capability but micropower sensor contained in the EMAP power domain. This allows us to exercise the sensor device and sensor power domain. The LEAP testbed, shown in Figure 5, relies on a physical event generator consisting of two horizontal linear arrays of 32 individually controlled lamps distributed over an 8 m length. Both red and green lamps are attached to the rigid assembly at fixed intervals and power for each lamp is sequenced by an independent relay control, itself supported by an event generator server platform. The event generator system is remotely accessible with capability to repeatedly perform diverse experiments thereby extracting both instantaneous discrete and statistical characteristics of system performance.

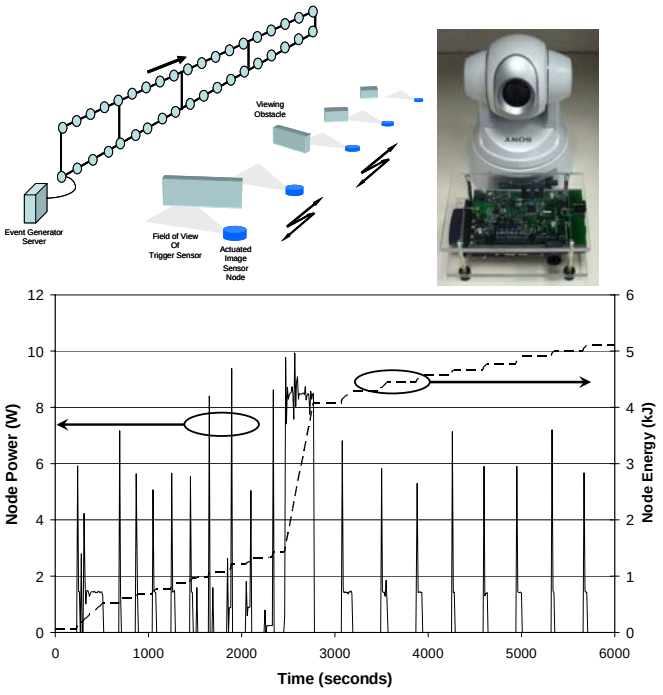

Figure 5. The Event Generator and distributed LEAP nodes are shown at upper left with a typical LEAP node shown at upper right. Power and energy dissipation (dashed line) for one typical node in the network is shown in the bottom panel (all nodes display similar behavior for this algorithm).

The event generator server platform manages a series of lamp sequence test vectors yielding dynamic events. For example, sequencing of lamp state, such that only one lamp is illuminated at any time, causes an apparent motion of the illumination providing a target that must be detected and tracked. Test vectors for a typical experiment produce event patterns that are classified into contexts mirroring many forms of environment phenomena. For example, events were classified into slow, medium, and fast motion corresponding to velocities and events appeared at slow, medium, and fast issues rates. An environmental context in this instance may consist of many events with a specific choice of velocity and issue rates and may itself remain fixed for a period, prior to a change in context and a resulting new velocity, new issue rate, or both. A distribution of random events and context classifications may be introduced as well. An example testbed configuration is shown in Figure 5.

C. *Algorithm Design and Implementation*

In addition to enabling fundamental investigations of energy aware algorithms in a precise, reproducible fashion, the LEAP systems and testbed have supported both an undergraduate and graduate course. Student course projects have ranged from energy aware detection and tracking of moving objects to energy aware fault detection and recovery systems that all adapt to environmental context to reduce energy. All algorithms are distributed and involve software systems operating only on the LEAP nodes. Course project management has been enabled also by a unique testbed system that manages LEAP node software distributions automatically on each node, for each user, according to a usage schedule that is accessible to all users. Demonstration of robust operation results from having supported both research and over 50 student users.

A current topic of investigation is the development of novel algorithms that are now enabled to manage energy, schedule resource usage, and seek to optimize sensing performance. The experimental results from testbed characterization of an example algorithm are shown in Figure 5. This algorithm was developed to solve the problem of event detection and identification with the requirement that a distributed set of nodes must detect and identify an object (the moving lamp signal) and determine its color (red or green) and detect its precise location using the imager, and finally compute velocity. This all must be accomplished while minimizing energy usage by limiting the time of operation of the SPM and camera image sensor. Camera power usage is large at seven watts peak, thus strongly encouraging the algorithm designer to apply LEAP EMAP capabilities to minimize its operation time. This encourages the development of hybrid algorithms that operate both in a reactive mode for discovery of instantaneous environmental context and a proactive mode for operation at minimum resource usage. Algorithm designs are constrained to those that uniformly distribute energy usage demands to all nodes. Finally, algorithm designers seek to minimize the probability of false positive or false negative detection error.

The algorithm for which results are shown in Figure 5, reactively seeks to determine the rate at which events occur and the velocity associated with events, then proactively schedules the operation of distributed nodes to minimize their energy usage. Supporting applications, hosted on the SPM were developed using the EMAP msp-client. Energy in each power domain was logged..

Figure 5 displays data from the period immediately after a test initiates at $t = 0$. Within 500 seconds the system has classified the environment behavior and has settled into a self-determined operation cycle where at approximately each 200 seconds this LEAP node is triggered from a sleep state for event characterization – no misdetections occur during this period. A second node also must operate to ensure localization in the event of imaging obstacles that may obscure the target. It is important to note that energy is used only episodically during servicing of the event. The large energy power excursions seen in the figure are due to imager operation. Then note that at $t = 2400$ seconds a change appears in the environment and a new context appears with a reduced event issue rate. Initially unaware of this change, the LEAP system detects this new context and expends energy in sensing and communication until the distributed LEAP nodes discover the new event context and again settle into a properly proactive optimized cycle of operation for $t > 3000$ s. This algorithm is a demonstration of capability and represents one member of a broad class of new investigations that may now be pursued.

VII. Conclusions and Future Work

The new LEAP ENS platform including a heterogeneous multiprocessor architecture has been developed based on a design approach addressing the challenge of supporting complex and powerful sensor systems, embedded computing platforms, and high performance communication interfaces. To achieve desired performance goals while simultaneously meeting energy dissipation requirements, this design approach focuses on exploiting high energy efficiency components that are scheduled for operation on-demand operation. The LEAP system relies on the EMAP module for maintaining low power, constantly vigilant operation while providing event detection and fine-grained energy accounting.

The LEAP system is now in active use for development of a wide range of ENS applications in many environment monitoring applications. The feasibility of operation at low duty cycle with multiple power domain scheduling has been demonstrated and experimentally verified, as discussed here. Many users have developed complex and robust algorithm implementations based on the SPM tool set including the EMAP msp-client interface.

Future work enabled by the LEAP architecture includes the development of many additional platforms now purposefully designed for energy monitoring tasks. Other systems will provide new operating system instrumentation providing high resolution, per-task and per user monitoring of energy usage. Also, since boot times for sensor systems and the Linux OS can be many seconds, the energy expended in dynamic power management transition may dominate the overall energy expenditure and thus reduce the effectiveness of dynamic power management in the limit of rapidly arriving events. Thus, future LEAP development includes new systems including methods that automatically preserve minimal operational state when entering low power modes to reduce the live memory footprint during dormancy periods. This further includes systems that detect and preserve active data before entering a suspended state while discarding unused or recoverable information. Data would further be preserved to the most energy efficient storage medium based upon its estimated volatility and lifetime such as previously proposed [25-26]. During resume operations, data will be restored to its previous state according to a lazy algorithm such that immediate execution of a foreground operation will be possible without incurring the delay of complete memory restoration and further reduce transition energies.

The LEAP system reported here is being deployed in critical environmental monitoring systems for both static and actuated sensor networks as well as in research testbeds. These energy aware capabilities now may be added to existing sensor networking run time systems such as EmStar [19] and included into the new Tenet microserver and mote architectures [27].

Acknowledgment

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. ANI-00331481. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] D. Estrin, G.J. Pottie, M. Srivastava, “Instrumenting the world with wireless sensor networks,” ICASSP 2001, 2001.

[2] C. M. P. Ozanne, D. Anhuf, S. L. Boulter, M. Keller, R. L. Kitching, C. Korner, F. C. Meinzer, A. W. Mitchell, T. Nakashizuka, P. L. Silva Dias, N. E. Stork, S. J. Wright, M. Yoshimura, “Biodiversity meets the tmosphere: A global view of forest canopies,” Science, vol. 301, pp. 183-186, July 2003.

[3] Maxim Batalin, Gaurav S. Sukhatme, Yan Yu, Mohammad H. Rahimi, Mark Hansen, Gregory Pottie, William Kaiser, and Deborah Estrin, "Call and Response: Experiments in Sampling the Environment," ACM SenSys, Baltimore, Maryland, Nov 2004, pp. 25-38,

[4] K. Bult, A. Burstein, D. Chang, M. Dong, M. Fielding, E. Kruglick, J. Ho, F. Lin, T. H. Lin, W. J. Kaiser, and others, 1996 International Symposium on Low Power Electronics and Design, Digest of Technical Papers, “Low Power Systems for Wireless Microsensors”, Proceedings of 1996 IEEE International Symposium on Low Power Electronics and Design. pp 17-21. 1996.

[5] J. Agre, L. Clare, G. Pottie, and N. Romanov, “Development platform for self-organizing wireless sensor networks,” Proceedings of Aerosense, International Society of Optical Engineering, 1999, pp. 257-268.

[6] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister. "System Architecture Directions for Network Sensors”. Proceedings of ASPLOS, 2000, pp 93-104, 2000.

[7] J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling Ultra-Low Power Wireless Research. Fourth International Symposium on Information Processing in Sensor Networks, IPSN 2005, pp 364-369 April 2005.

[8] L. Nachman., R. Kling, R. Adler, J. Huang, V. Hummel. The Intel Mote Platform: A Bluetooth-Based Sensor Network for Industrial Monitoring. Fourth International Symposium on Information Processing in Sensor Networks, IPSN 2005, pp 437-442, April 2005.

[9] LEAP Systems: http://www.cens.ucla.edu/portal/nims/

[10] http://platformx.sourceforge.net

[11] http://www.xbow.com/Products/products.htm

[12] B. Schott, M. Bajura, J. Czarnaski, J. Flidr, T. Tho, and L. Wang. A modular power-aware microsensor with 1000x dynamic power range. Fourth International Symposium on Information Processing in Sensor Networks, IPSN 2005, pp 469-474, April 2005.

[13] C. Worth, M. Bajura, J. Flidr, and B. Schott, “On-demand Linux for Power-aware Embedded Sensors,” Proceedings of the Ottawa Linux Symposium, Ottawa, Ontario, Canada, July 26-29, 2004.

[14] J. Flinn, M. Satyanarayanan, “Managing Battery Lifetime with Energy-Aware Adaptation”, ACM Transactions on Computer Systems, Vol 22 No. 2, pp 137-179, May 2004.

[15] M. Waitz “Accounting and Control of Power Consumption in Energy-Aware Operating Systems”, Diploma Thesis, 2002, University of Erlangen-Nurnberg

[16] G. Banga, P. Druschel, and J.C. Mogul, ‘Resource Containers: A New Facility for Resource Management in Server Systems”, Proceedings of the 3rd Symposium on Operating Systems Design and Implemenatation, 1999

[17] H. Zeng, X. Fan, C. Ellis, A. Lebeck, and A. Vahdat. ECOSystem: Managing energy as a first class operating system resource. In Tenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS X), 2002.

[18] M. Anand, E. Nightingale, J. Flinn. Ghosts in the Machine: Interfaces for Better Power Management. In Proceedings of the 2nd Annual International Conference on Mobile Systems, Applications, and Services (MOBISYS '04), Boston, MA, June 2004.

[19] Sensoria Slauson Processor Module http://www.sensoria.com

[20] L. Girod and J. Elson and A. Cerpa and T. Stathopoulos and N. Ramanathan and D. Estrin, “EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks,” USENIX, 2004.

[21] Redboot http://ecos.sourceware.org/redboot/

[22] cramfs http://sourceforge.net/projects/cramfs/

[23] Busybox http://www.busybox.net/

[24] J. Labrosse, MicroC/OS-II, The Real Time Kernel, CMP Books; 2nd Edition Edition, June 2002

[25] H. G. Lee and N. Chang, "Energy-Aware Memory Allocation in Heterogeneous Non-Volatile Memory Systems," in Proceedings of International Symposium on Low Power Electronics and Designs(ISLPED 2003), pp. 420-423 , Seoul, Korea, August 2003.

[26] H. Huang, P. Pillai, and K. G. Shin, "Design and Implementation of Power-Aware Virtual Memory," in Proceedings of USENIX Technical Conference, June 2003.