The Platforms Enabling Wireless Sensor Networks

By Jason Hill, Mike Horton, Ralph Kling, *and* Lakshman Krishnamurthy

All emphasize low-cost components operating on shoestring power budgets for years at a time in potentially hostile environments without hope of human intervention.

Wireless sensor networks combine processing, sensing, and communications into tiny embedded devices. Peer-to-peer communication protocols then combine the individual devices into an interconnected mesh network where data is seamlessly routed among all the nodes. These networks require no external infrastructure and can scale to hundreds or even thousands of nodes.

Here, we explore the standard sensor network platforms where devices range from millimeter-size custom silicon to PDA-size integrated units. Critical to the operation of any sensor network device is the ability to satisfy harsh always-on power requirements. Unlike cell phones and wireless laptops, periodic recharging is not possible for most wireless sensor networks. In many cases, devices are placed in the field for years at a time without maintenance or human intervention of any kind.

In sensor networking, special-purpose sensor nodes are purposely designed to sacrifice flexibility in order to be as small and inexpensive as possible. Generic sensor nodes provide a rich expansion interface for making flexible connections with an array of simple sensors. High-bandwidth sensor nodes contain the built-in processing and communication capabilities needed to deal with complex sensor streams, including video and voice processing. Gateway nodes provide a critical link between the sensor network and traditional networking infrastructures, including Ethernet, the 802.11 communication standard, and wide-area networks.

Traditional network abstractions are generally not suitable for wireless sensor networks. Unlike traditional operating systems, operating systems for wireless sensor networks must tightly integrate wireless connectivity. For example, in TinyOS [5], a specialized component model exploits advanced compiler technology to simultaneously provide efficiency and reliability [1, 5]. These same concepts are now being incorporated into more traditional operating systems in gateway-class and high-bandwidth nodes [2].

Here, we outline the four main platform classes that have emerged in recent years in wireless sensor networks; devices from multiple platform classes often work together in real-world application deployments. We also review the architectural similarities of sensor network devices, exploring the core differences among classes, and consider the recent progression of

sensor-network hardware, extrapolating future capabilities in future devices.

Platform Classes

Initial deployment experience has shown that sensor network systems require a hierarchy of nodes starting at low-level sensors and continuing up through high-level data aggregation, analysis, and storage nodes (see Figure 1). This tiered architecture is common in virtually all sensor networks and is best illustrated by example. Consider a sensor network for an advanced security system in which a majority of the sensors cover window breakage, contact closure, and motion detection. The quantity and range of locations for these simple sensors require that they be battery powered. They are complemented by a handful of more advanced sensors, including cameras and acoustic and chemical detectors, placed in key locations. Both simple and complex sensor data is routed over a mesh network into a building-monitoring-and-control facility that provides a continuous monitoring capability.

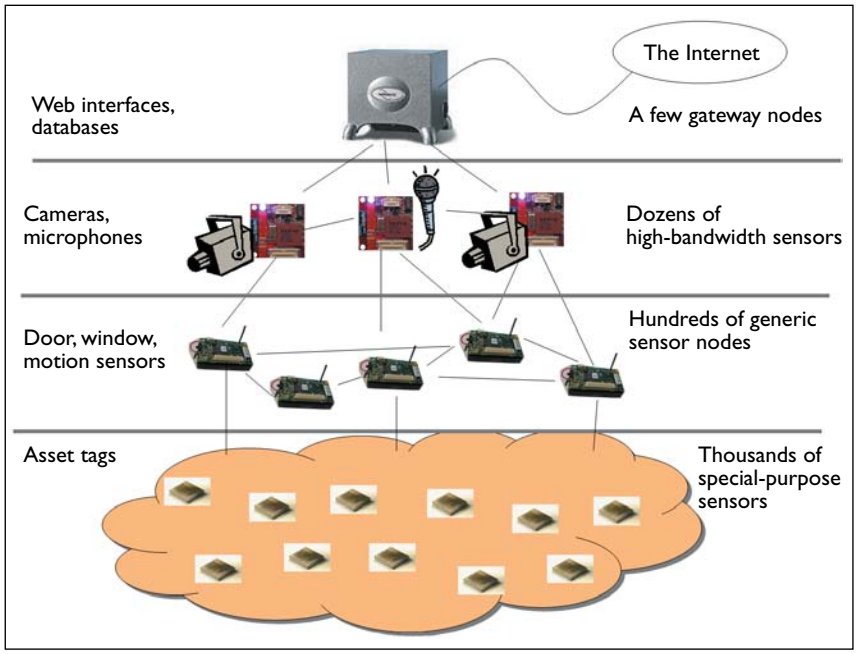

Figure 1. Hierarchical deployment of a wireless sensor network. Each platform class handles different types of sensing.

The sensors placed on windows and doors for intrusion detection are examples of generic sensing devices. Their function is simple and specific and requires long-term battery operation. Moreover, their on-board processing and communication rates are minimal. In contrast, acoustic, video, and chemical sensors are examples of high-bandwidth nodes requiring more computational resources and communication. They may, in some cases, require battery power but often need to be plugged into the public power system for long-term operation.

In addition to traditional security applications, wireless sensor networks are being designed to track mobile assets, as well as personnel, through attached tiny, low-cost security tags (mini-motes). These special-purpose sensor nodes are synonymous with Smart Dust [8], or cubic-millimeter-scale devices supported by extremely limited energy resources. They could trigger an alarm when an asset leaves a facility without authorization. Moreover, they must be highly integrated and very inexpensive.

In security systems, the mesh network of sensors is likely to have one or more end points containing a database or other aggregation software designed to process and store individual sensor readings. These head, or gateway, nodes provide an interface into many existing types of networks.

Table 1 outlines typical operating characteristics of the four classes of nodes—specialized sensing platform, generic sensing platform, high-bandwidth sensing, and gateway—implemented with state-of-the-art technology. The Spec node the author Hill designed at the University of California, Berkeley, is representative of the special-purpose sensor class. It is a single-chip node designed specifically for ultra-low-cost production and low-power operation. Requiring just 2.5 mm × 2.5 mm of silicon, it includes data RAM, processing, and communication capabilities. In order to reduce size and complexity, the Spec node was built so it would interface only with simple sensors and communicate only over short distances. The prototype versions (produced February 2003) contained only a transmitter (without a receiver); future versions will contain a full transceiver. The Spec node is an ideal asset tag; combined with a tiny battery, it is capable of periodically reporting its presence for years to come.

The Berkeley Motes are a notable example of a general-sensing-class device, used today by more than 100 research organizations. The Mica2 is the most recently developed commercially available version, constructed from off-the-shelf components to provide the greatest possible flexibility (see Figure 2). It includes a large interface connector allowing its attachment to an array of sensors. By providing a large number of I/O pins and expansion options, the

Mica2 is a perfect sensor node option for any application where size and cost are not absolutely critical. It is, for example, easily connected to motion detectors and door-and-window sensors as the foundation of a building security system. Moreover, the Mica2 is capable of receiving messages from Spec nodes attached to high-value assets, including personal computers and laptops, at risk of being stolen. The memory and processing power available on the Mica2 node easily handles the computation required to keep track of several dozen Spec-based asset tags.

Node Type	Sample "Name" and Size	Typical Application Sensors	Radio Bandwidth (Kbps)	MIPS Flash RAM	Typical Active Energy (mW)	Typical Sleep Energy (uW)	Typical Duty Cycle (%)
Specialized sensing platform	Spec mm^3	Specialized low-bandwidth sensor or advanced RF tag	<50Kbps	<5 <0.1Mb <4Kb	1.8V*10–15mA	1.8V *1uA	0.1–0.5%
Generic sensing platform	Mote 1-10cm^3	General-purpose sensing and communications relay	<100Kbps	<10 <0.5Mb <10Kb	3V*10–15mA	3V *10uA	1–2%
High-bandwidth sensing	Imote 1-10cm^3	High-bandwidth sensing (video, acoustic, and vibration)	~500Kbps	<50 <10Mb <128Kb	3V*60mA	3V *100uA	5–10%
Gateway	Stargate >10cm^3	High-bandwidth sensing and communications aggregation Gateway node	>500Kbs–10 Mbps	<100 <32Mb <512Kb	3V*200mA	3V *10mA	>50%

Table 1. Typical operating characteristics of the four classes of sensor-network nodes.

While the Mica2 node is easily interfaced by hardware engineers to an array of sensors, it cannot handle the high bandwidth of data coming from complex sensors. When attempting to process video or high-bandwidth audio, the Mica2 node falls short. The iMote, developed by Intel Research (first produced in May 2003), is designed to be a high-bandwidth sensor platform, including significantly more on-chip RAM and processing power, as well as a Bluetooth-based radio capable of communication rates in excess of 500Kbps.

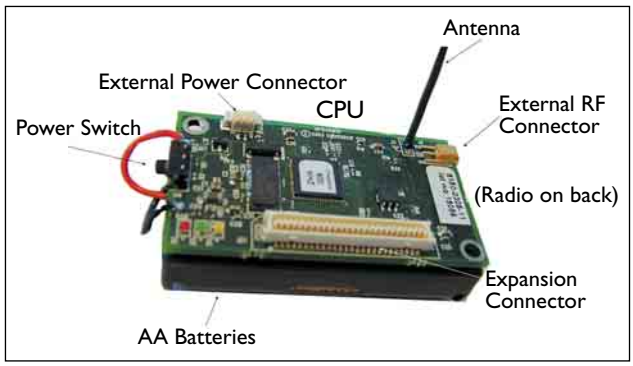

Figure 2. The Mica2 node is the most popular sensor-network research platform. The large expansion connector allows it to be connected to a wide range of sensors.

The Stargate platform developed by Intel and sold by Crossbow Technology is representative of gateway-class devices and includes a 400MHz X-scale architecture-based processor (also developed by Intel) with several megabytes of RAM and up to gigabytes of persistent storage. It is capable of interfacing directly to Mica2- and iMote-based devices and bridging the data from low-power mesh networks to traditional networks, including 802.11, Ethernet, and wide-area varieties. Moreover, the processing and memory provisions on the Stargate node allow it to act as a Web front-end to sensor networks where users access its data via a Web browser.

The operating system running on a particular platform must be matched to the platform's underlying hardware capabilities. For special-purpose and generic-sensor-class devices, a special operating system called TinyOS (developed at the University of California, Berkeley) is designed to run on platforms with limited CPU power and memory space. Unlike many embedded operating systems, it provides tight integration between wireless connectivity and networking functions. However, as platform capabilities improve with, for example, the Stargate platform, more advanced operating system support is required to meet the demands of more complex applications. Multiprocessing, preemptive task switching, and even virtual memory support become desirable when managing multiple system functions. The Stargate node runs an embedded version of the Linux operating system. In addition to providing a range of system capabilities, Linux provides a suite of device drivers for enabling gateway nodes to bridge to legacy networks. Drivers for Ethernet cards and 802.11 wireless networking cards are essential for allowing the gateway nodes to bridge to a range of wide-area networking systems.

Architectural Differences

The overall architectures of the four sensor-network platform classes are remarkably similar despite significant differences in device capabilities. The architectural similarity follows from the requirement that they seamlessly integrate wireless networking. Network support must be transparent and self-configuring to allow sensor networks to scale in size and complexity. In contrast, their core differences come

from their developers' desire to optimize the power consumption of each platform for a certain class of application. Some of the fundamental decisions that must be made by application engineers include the amount of on-board memory, whether to include flash memory, the amount of CPU processing power, and the type and bandwidth of the wireless link. Since most implementations employ off-the-shelf components, some of these decisions are dictated by the availability of suitable parts. In the end, cost and power consumption are major factors influencing the final design of any given sensor node.

A major difference between sensor-network nodes and more traditional computing platforms, including PCs, PDAs, and even embedded devices, is the extreme emphasis in sensor networks on power management. A large number of applications require battery-powered operation for extended periods of time. In order to manage power efficiently, each subsystem of the platform is powered individually. For example, a radio should have to be turned on only during active communication, and it should be possible to shut down the CPU between processing requests. Similarly, it should be possible to power down the sensor and I/O subsystems individually when not in use. The operating system, TinyOS in many cases, controls the activity and power state of the various subsystems. The exact timing of the power-down cycles is determined by a number of factors, including application requirements and the particular hardware being used. In TinyOS, power management pervades every aspect of the system, and all components are designed to save power when not active.

To facilitate proper power management, sensor-network platforms give applications direct, fine-grain control over the underlying hardware. Traditional layered abstractions for both network [2] and sensor stacks lead to inefficiencies in power usage. Recent research suggests a common approach to solving this challenge across the range of platforms [2, 5] by way of three additional architectural components:

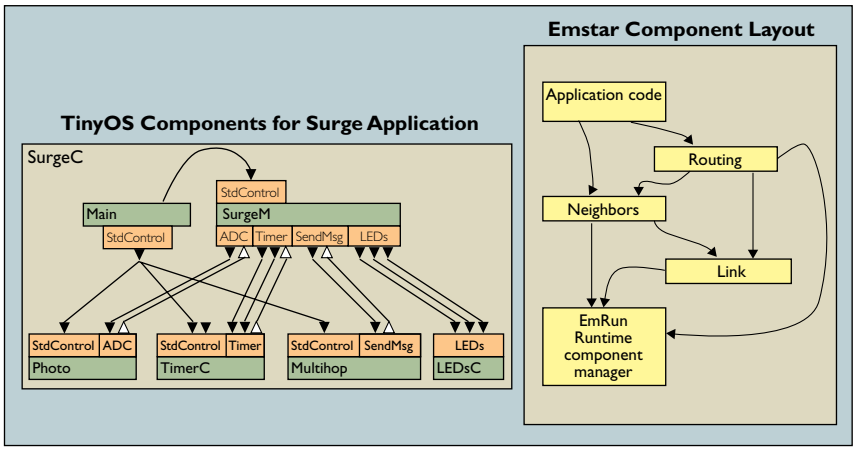

Figure 3. Emstar applications consist of active processes and Linux device drivers using message passing to communicate; the component model allows nonlayered interactions among components. The system includes a runtime component manager called EmRun that uses a configuration script to wire up components and provide logging services and runtime component performance monitoring. Similarly, the TinyOS component model gives applications direct access to low-level components (such as LEDs and timers) and complex operations (such as multihop routing).

- A general-purpose component framework that eliminates layering;
- Hardware functions that are exposed to applications and middleware; and
- Virtualization, or a unifying layer of abstraction, and interpreted scripts, or a simplified programming process, for writing sensor network applications.

On mote-class devices (such as Spec and Mica2), TinyOS provides low-level hardware control through a built-in component model that eliminates layering. The software abstractions in TinyOS are designed to allow application-level components direct access to hardware as needed. While this capability is also found in other embedded operating systems, it is generally not present in more traditional operating systems, including Linux.

Using Linux on gateway-class nodes (such as Stargate) requires additional support for precise hardware control, so it was added by developers. On Stargate, processor registers and general-purpose I/O lines are made available to applications via special-purpose drivers. In turn, sensor-network development environments (such as Emstar [2]) employ these drivers to give applications control over the timing and state of the hardware peripherals they need (see Figure 3).

The TinyOS and embedded-Linux development efforts have each embraced virtualization of processing and communication resources to simplify the sensor-network development process. One trade-off in providing precise hardware control to application-

level software is that hardware-specific modules sometimes render sensor-network software nonportable. TinyOS and Emstar both feature hardware abstractions that attempt to maintain portability without sacrificing precise control. Each provides the option of using high-level interpreters [2, 6, 7] to simplify application development. However, researchers must still determine how to ensure a high degree of efficiency, along with resource virtualization.

Platform Road Map

The recent research and development of first-generation wireless sensor network platforms is now feeding back on itself to help systems engineers define a new generation of hardware better able to meet network demands. Table 2 outlines the sensor network hardware platforms available today.

Hardware progression. Analyzing the progression of sensor-network hardware must account for the influence of Moore's Law on the design and development of the networks. For all platform classes except special-purpose sensor nodes, Moore's Law promises an increase in performance for a given power budget. As shown in Table 2, the Mica2 node has roughly eight times the memory and communication bandwidth as its predecessor, the Rene node, developed in 1999, despite involving the same power and cost. The gateway and high-bandwidth devices have achieved similar performance jumps without significantly changing their power or cost requirements. In contrast, the special-purpose sensor nodes (such as Spec) use advances derived from Moore's Law to reduce their power consumption and cost requirements while maintaining the same performance level.

Node	CPU	Power	Memory	I/O and Sensors	Radio	Remarks
Special-purpose Sensor Nodes						
Spec 2003	4–8Mhz Custom 8-bit	3mW peak 3uW idle	3K RAM	I/O Pads on chip, ADC	50–100Kbps	Full custom silicon, traded RF range and accuracy for low-power operation.
Generic Sensor Nodes						
Rene 1999	ATMEL 8535	.036mW sleep 60mW active	512B RAM 8K Flash	Large expansion connector	10Kbps	Primary TinyOS development platform.
Mica-2 2001	ATMEGA 128	.036mW sleep 60mW active	4K RAM 128K Flash	Large expansion connector	76Kbps	Primary TinyOS development platform.
Telos 2004	Motorola HCS08	.001mW sleep 32mW active	4K RAM	USB and Ethernet	250Kbps	Supports IEEE 802.15.4 standard. Allows higher-layer Zigbee stardard. 1.8V operation
Mica-Z 2004	ATMEGA 128		4K RAM 128K Flash	Large expansion connector	250Kbps	Supports IEEE 802.15.4 standard. Allows higher-layer Zigbee stardard.
High-bandwidth Sensor Nodes						
BT Node 2001	ATMEL Mega 128L 7.328Mhz	50MW idle 285MW active	128KB Flash 4KB EEPROM 4KB SRAM	8-channel 10-bit A/D, 2 UARTS Expandable connectors	Bluetooth	Easy connectivity with cell phones. Supports TinyOS. Multihop using multiple radios/nodes.
Imote 1.0 2003	ARM 7TDMI 12-48MHz	1mW idle 120mW active	64KB SRAM 512KB Flash	UART, USB, GPIO, I^2C, SPI	Bluetooth 1.1	Multihop using scatternets, easy connections to PDAs, phones, TinyOS 1.0, 1.1.
Gateway Nodes						
Stargate 2003	Intel PXA255		64KNSRM	2 PCMICA/CF, com ports, Ethernet, USB	Serial connection to sensor network	Flexible I/O and small form factor power management.
Inrysnc Cerfcube 2003	Intel PXA255		32KB Flash 64KB SRAM	Single CF card, general-purpose I/O		Small form factor, robust industrial support, Linux and Windows CE support.
PC104 nodes	X86 processor		32KB Flash 64KB SRAM	PCI Bus		Embedded Linux or Windows support.

Table 2. Current sensor network platforms organized by device class.

Part of the performance increase in the generic-sensor-node class is due to new CMOS radios specifically designed for low data rates and low power consumption. In addition to improving raw radio performance metrics, the communication interfaces provided by low-power radio now include specialized hardware support to help reduce the peak load placed on the CPU. Low-power controllers can burst data out over the RF channel at rates several times faster than with the previous generation of radios. Moreover, early hardware designs used the microcontroller to duty cycle the radio and check for channel activity [3]. Next-generation radios to be released this year will have built-in state machines that perform this operation automatically.

Software and interface standards. Engineers and researchers in the field of low-power wireless technology are pursuing a protocol-standardization effort aimed at allowing future devices to interoperate with one another. The 802.15.4 standard provides a specification of the RF channel and signaling protocol to be used. Built atop 802.15.4 is the Zigbee protocol, a specification of the application-level communication protocol between devices. To put Zigbee and 802.15.4 in perspective relative to the platforms we've discussed here, 802.15.4 determines which radio hardware to use, and Zigbee determines the content

of messages transmitted by each networked node. Following the availability of the first 802.15.4 radios in early 2004, researchers have sought to develop TinyOS drivers. When these drivers are completed and released, existing sensor-network applications will be able to take advantage of the new capabilities of the 802.15.4 chips.

Even as the standardization process advances, it is not clear whether a comprehensive set of standard protocols will ever be available to meet all application requirements. Unlike traditional Internet applications—nearly all of which use TCP/IP—sensor-network applications demand protocols that are optimized for their unique communication patterns. Additionally, the for-members-only nature of Zigbee standards and other proprietary solutions impose additional hurdles on any widespread sensor-network standard-setting process and adoption. In this environment, TinyOS's ability to allow application developers to assemble custom protocols from individual networking building blocks will continue to be the preferred sensor-network development strategy. Developers will likely start with generic TinyOS protocol implementations, then customize as needed to satisfy application-specific requirements.

Conclusion

The age of ubiquitous sensing and actuation is being fueled by Moore's Law and the development of advanced wireless sensor-networking platforms. Hardware can be used to deploy data-collection networks capable of operating for years without maintenance in remote, often hostile, environments. As capabilities improve, these systems will thus be able to automatically act on sensor data to manage our environment for us.

Most current sensor-networking deployments include square-inch-size generic sensor devices that represent an interconnected mesh tied to the Internet through one or more gateway-class devices. More advanced networks include high-bandwidth sensor nodes capable of dealing with complex data streams, including voice and video. Alternatively, they may include tiny special-purpose sensor nodes that are just millimeters on a side and weigh only a few grams each. While the capabilities, cost, and size of each class of device will change with technological advances, these four fundamental classes of device will likely remain for the foreseeable future.

Integral to the performance of sensor-network nodes is the software supporting them. Sensor-network applications require precision control over the underlying hardware in order to meet the strict power limitations they must satisfy. Current development efforts focus on two software platforms for use in wireless sensor networks. TinyOS provides the precise, efficient, low-level control demanded by both general- and special-purpose networking nodes. In contrast, special kernel modifications have been added to Linux to enable it to support gateway and high-bandwidth-class device operation. Combined with the hardware platforms, TinyOS and embedded Linux are together being shaped into a powerful toolbox for building wireless sensor-network applications. C

References

1. Gay, D., Levis, P., von Behren, R., Welsh, W., Brewer, E., and Culler, D. The nesC language: A holistic approach to network embedded systems. In *Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation* (San Diego, CA, June 9–11, 2003).
2. Girod, L., Elson, J., Cerpa, A., Stathopoulos, T., Ramanathan, N., and Estrin, D. *Em*: A Software Environment for Developing and Deploying Wireless Sensor Networks.* Tech. Rep. 0034, Center for Embedded Networked Sensing, UCLA Computer Science Department, Los Angeles, Dec. 16, 2003.
3. Hill, J. and Culler, D. *System Architecture for Wireless Sensor Networks.* Ph.D. thesis, University of California, Berkeley, May 2004; see www.jlhlabs.com/jhill_cs/jhill_thesis.pdf.
4. Hill, J. and Culler, D. Mica: A wireless platform for deeply embedded networks. *IEEE Micro. 22,* 6 (Nov./Dec. 2002), 12–24.
5. Hill, J., Szewczyk, R., Woo, A., Hollar, S.,Culler, D., and Pister, K. System architecture directions for network sensors. In *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS2000)* (Cambridge, MA, Nov. 12–15, 2000).
6. Levis, P. and Culler, D. Maté: A virtual machine for tiny networked sensors. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS2002)* (San Jose, CA, Oct. 5–9, 2002).
7. Madden, S., Szewczyk, R., Franklin, M., and Culler, D. Supporting aggregate queries over ad-hoc wireless sensor networks. In *Proceedings of the Workshop on Mobile Computing and Systems Applications* (New York June 20–21, 2002).
8. Pister, K., Kahn, J., and Boser, B. *Smart Dust: Wireless Networks of Millimeter-scale Sensor Nodes.* Electronics Research Laboratory Research Summary, 1999.

Jason Hill (jhill@jlhlabs.com) is president and CEO of JLH Labs in Capistrano Beach, CA.

Mike Horton (mhorton@xbow.com) is president and CEO of Crossbow Technology, Inc. in San Jose, CA.

Ralph Kling (rkling@mipos2.intel.com) is a principal researcher and leader of the Intel Mote project at Intel Research and the Systems Technology Laboratory at Intel Corp. in Santa Clara, CA.

Lakshman Krishnamurthy (lakshman.krishnamurthy@intel.com) is a senior staff engineer in the Communication Technology Lab at Intel Corp. in Hillsboro, OR.