Poster Abstract: A Networked Embedded System Platform for the Post-Mote Era

Pat Pannuto†, Michael P Andersen‡, Tom Bauer§, Bradford Campbell†, Amit Levy§, David Culler‡, Philip Levis§, and Prabal Dutta†

†Computer Science & Engineering
University of Michigan
Ann Arbor, MI 48109
{ppannuto,bradjc,prabal}@umich.edu

‡Computer Science & Engineering
University of California, Berkeley
Berkeley, CA 94720
{m.andersen,culler}@berkeley.edu

§Computer Science & Engineering
Stanford University
Stanford, CA 94305
{tbauer01,levya,pal}@stanford.edu

Abstract

For the last fifteen years, research explored the hardware, software, sensing, communication abstractions, languages, and protocols that could make networks of small, embedded devices—motes—sample and report data for long periods of time unattended. Today, the application and technological landscapes have shifted, introducing new requirements and new capabilities. Hardware has evolved past 8 and 16 bit microcontrollers: there are now 32 bit processors with lower energy budgets and greater computing capability. New wireless link layers have emerged, creating protocols that support rapid and efficient setup and teardown but introduce novel limitations that systems must consider. The time has come to look beyond optimizing networks of motes. We look towards new technologies such as Bluetooth Low Energy, Cortex M processors, and capable energy harvesting, with new application spaces such as personal area networks, and new capabilities and requirements in security and privacy to inform contemporary hardware and software platforms. It is time for a new, open experimental platform in this post-mote era.

1 Introduction

Over a decade ago, a flurry of hardware platforms and supporting software empowered the research community to investigate and explore wireless sensor networks and their applications. Many projects today still use these "mote"-class devices to research systems software, low-power networking, and application design.

Technology has progressed a great deal in the past decade. The Cortex M series of 32-bit processors finally have sleep currents competitive with mote microcontrollers (MCUs).[1] 802.15.4 has grown far beyond the closed world of ZigBee, with new physical layers for new applications. The recent incorporation of Bluetooth Low Energy (BLE) into mobile phones allows ubiquitous sensing networks to directly interact with human-centric devices. Sensors are orders of magnitude more energy efficient and precise.

Simultaneously, applications have become much richer. Applications in early sensor network research focused on fixed rate, long-term sensing, guiding a research agenda of ultra-low power operation and robust multi-hop networking. Today personal area networks (PANs) tether to phones and interact with proximity networks such as iBeacon. Building-area networks share knowledge, like occupancy, among security, HVAC, and lighting control. In addition, the rise of "maker culture" and their platforms and communities (e.g. Arduino [1]) has led to a level of diversity and accessibility that early research platforms simply could not provide.

We have an explosion of new applications and developers, each with new and challenging requirements. We have reached a turning point in hardware, enabling new devices and operational models. It is time for a new platform: a new OS and a family of hardware devices to explore and research embedded networked systems in the post-mote era.

2 Technology Today

Advancements in three major areas—networking, MCUs, and energy harvesting—open new opportunites for networked embedded systems and generate new implications on how an OS should manage activity, security, and privacy.

Integrating TinyOS and Bluetooth Low Energy. BLE provides highly efficient beaconing, bidirectional communication, fast device discovery, authentication, and optional piconetwork formation and management [3]. BLE explicitly incoporates ideas of periodic beacons and duty cycles, such that a device with a wakeup latency of a second can have an average current draw of < 40 µA (e.g. Nordic Semiconductor NRF8001). Tight timing requirements mean that most chips have tightly integrated software stacks that present limited abstractions such as getting and setting attributes. Furthermore, these software stacks assume a threaded, sequential execution model that in our experience is hard to adapt efficiently to an event-driven paradigm. Finally, the current BLE model of embedded device coupled with a mobile app is built in the image of an application layer gateway, rather than an end-to-end addressable and routable network: general protocols are an open research question.

[1] e.g. the NXP LPC11U6X family draws 1 µA in sleep with active RTC.

SenSys'14, November 3–5, 2014, Memphis, TN, USA.
ACM 978-1-4503-3143-2/14/11. http://dx.doi.org/10.1145/2668332.2668364

Cortex M0–M4. "Low-power processors" now includes powerful cores such as the 0.5–120 MHz Atmel SAM4S with a 32 bit Cortex M4 (2 MB flash, and 160 kB SRAM) that draws 1 μA–30 mA. Centimeter-scale chips now support primitives such as memory segmentation and dynamic frequency scaling. This presents an opportunity to enhance the robustness and capability of embedded OSes, if techniques such as memory protection and process isolation can be adapted to the embedded domain of tight hardware/software coupling.

New chips are evolving beyond the traditional definiton of an MCU. More diverse and capable peripherals, such as encryption co-processors or even pseudo-TPMs (e.g. ARM TrustZone), are incorporated on-die as semi-integrated peripherals. While they often hang off a shared bus, these peripherals can be separately powered, programmed, and can even communicate with one another while the CPU core is asleep. A platform must provide expressive methods to leverage the growing set of capable heterogenous resources beyond the CPU, both on- and off-die.

Impact of Energy Harvesting. A decade of research in energy-harvesting transducers, efficient power electronics, and compact energy storage technologies, coupled with radio power reductions, instant-on/instant-write phase-change memory, and the efficacy of modern MCUs make compact perpetual sensors a reality, even indoors [4]. However, how an operating system supports writing applications for intermittently-powered, energy-harvesting systems is an open question. It requires revisiting many assumptions about system startup, maintaining state across activation cycles, discovering and communicating with neighbors, predicting future energy availability, and scheduling operations under energy uncertainty [2].

3 Applications Today

A decade ago, stationary fixed rate (or occasionally event-driven) data reporting applications dominated sensor networks and their research. New applications that incorporate mobility, hybrid networking, and personal privacy are emerging. A new platform should embrace and enable these new opportunities.

Human-centric. The mobile device is a gateway to an embedded network. Instead of sending data to a fixed collection point, the network can send directly to a mobile device, on demand. This enables interaction with surrounding devices and infrastructure without needing to know URLs or logging into a cloud system. Proximity networks provide a degree of basic, physical security that enables opportunistic interactions.

Because these interactions are human-centric, they require low latency and are highly bursty, two properties unimportant in the dominant network model of the past decade. Furthermore, these low-latency interactions are more complex than event notifications or alarms. They can involve significant queries and exchanges of data. Finally, because they involve mobile devices, existing approaches of long-term link estimation are of limited use. These new application requirements require new communication abstractions, rethinking the tradeoffs between latency, storage, energy, snooping, and how an OS will support them.

Perpetual Networks. Energy harvesting and low-power peripherals will finally enable a long-term goal of sensor networks: perpetual networks. Imagine iBeacons and smoke detectors that need no battery replacements. If the world will be filled with thousands of smart objects per person, energy must recede to be a non-issue for almost all of them.

We cannot predict the performance of a solar cell. This means an OS is stuck between two big unknowns: the future energy available as well as the potential energy needs from bursty, human-centric interactions. For an embedded device to be truly perpetual, there must be platform support to scale behavior and performance based on these two factors, optimizing needs, wants, and energy use.

Privacy and Proximity in Networks. Interactions between PANs (centered around a user's mobile phone) and proximity applications such as iBeacons happen in public, with never-seen-before peers. This presents a security and privacy problem. On the one hand, connections between the PAN and proximity device must be confidential and authentic – e.g., for payments. On the other hand, casual interactions with proximity devices must not enable ubiquitous tracking of users. Unfortunately, confidential and authentic communication and anonymity are difficult to achieve simultaneously. Operating systems can play a role in addressing these issues, for example by coordinating security features in the BLE stack with application specific knowledge.

4 A New Platform, Hardware and Software

To begin, we design two complementary hardware platforms. First, Storm, a full-featured platform with a Cortex M4 processor with many advanced features (memory protection, DSP instructions) to explore novel embedded OS concepts, 802.15.4 and BLE radios to support connectivity with both established low-power wireless technology and emerging PANs, and a highly instrumented and flexible power network to enable fine-grained measurement and control. Second, Squall, a price and area-optimized platform for massive dissemination and large-scale testing with BLE connectivity that leverages an integrated SoC Cortex M0 for computation.

The increased complexity of peripherals, power management, and clock domains and the needs of new applications motivate a new operating system: Tock. The Tock kernel builds on the past decade of research to provide stable, simple implementations of core abstractions such as IPv6, sensing, and local storage. The core idea and abstraction in Tock is time – global time for timestamps, local time that stretches and shrinks in response to energy availability, and clock source management for efficiency and performance. Tock will enable mechanisms for emerging embedded applications, including opportunistic and privacy-preserving data muling and sharing.

In summary, we aim to create a foundation for a new generation of groundbreaking research by capitalizing on a decade of technological improvements and on knowledge gleaned from fifteen years of networked embedded systems research.

5 References

[1] Arduino. http://www.arduino.cc/. Accessed 28-July-2014.

[2] M. Buettner, B. Greenstein, and D. Wetherall. Dewdrop: An energy-aware runtime for computational RFID. NSDI '11.

[3] C. Gomez et al. Overview and evaluation of Bluetooth Low Energy: An emerging low-power wireless technology. *Sensors*, 12(9), 2012.

[4] P. Martin et al. DoubleDip: Leveraging thermoelectric harvesting for low power monitoring of sporadic water use. SenSys '12.