EmPro: an Environment/Energy Emulation and Profiling Platform for Wireless Sensor Networks

Chulsung Park and Pai H. Chou
Center for Embedded Computer Systems
University of California, Irvine
Irvine, CA 92697-2625 USA
Email: {chulsung, phchou}@uci.edu

***Abstract*— Quantitative evaluation of wireless sensor platforms is difficult. Unlike general purpose computers that can run SPEC benchmarks from a file, it is difficult to reproduce the environmental input needed to stimulate the sensor nodes. Even if possible, open-loop playback would be unable to correctly account for adaptivity built into the behavior of these nodes. As a result, researchers resort to simulations, which do not consider all relevant factors without significant speed penalty.**

To address this problem, we propose EmPro, an environment/energy emulation and profiling system for WSNs. It accurately outputs electrical signals to emulate not only digital and analog inputs to the sensors but also the power sources as well as RF attenuation according to pre-programmed sequences. This emulation approach enables researchers to run the networked sensors in real-time in a realistic manner with full controllability and reproducibility. EmPro in profiling mode can also capture the observable behavior of WSNs for detailed analysis. Experimental results on the Eco and MICA2 WSN platforms show that EmPro can drive these hardware systems in real-time with high accuracy. We expect EmPro will expedite testing and serve as a sorely needed standard benchmarking tool for WSN platforms.

I. Introduction

Many wireless sensor platforms have been proposed to date. Researchers now have access to an increasing variety of platforms that have been designed for different applications. At the same time, it is becoming increasingly difficult to compare their performance quantitatively. First, there is no equivalent of a benchmark suite for evaluating these WSN platforms. Even if they exist, one main obstacle is how to actually run such a benchmark suite. Unlike general purpose computers that can be benchmarked by executing programs that are stored on disk, sensor nodes need stimuli from the physical environment. It is difficult to reproduce the environment input, such as lighting, sound, wind, vibration, magnetic field, or any other signal that is observable by all the sensors across a whole network. Even if possible, an open-loop playback approach would not be able to correctly account for the adaptivity aspect often built into the behavior of these nodes.

As a result of the difficulty in benchmarking, researchers do not have a good way to compare different platforms in a fair, quantifiable way. Vendors of different platforms resort to citing power and performance figures from the datasheets of the various components, including the microcontroller (MCU), RF transceiver, battery, data flash, accelerometers, and other devices used to construct the sensing system. Unfortunately, these figures can be very misleading, because it is not always easy to predict the *system-level* performance by composing component-level performance figures. Other researchers rely on simulations. They may be useful for obtaining initial estimation of certain aspects of the WSN and are fully reproducible; however, they often do not consider all relevant factors and thus cannot substitute actual field deployment. More recently, system simulators have been proposed for simulating detailed hardware behavior of WSNs. They may model the MCU, analog-to-digital converters (ADC), and various other aspects of the hardware, possibly in real time. However, there are several drawbacks: a detailed executable model must be constructed and validated for each platform to be evaluated. Although large-scale, shared simulation testbeds are available, they are practical for mainly existing platforms; they are not applicable if the subject platform to be evaluated is available only as a black box (e.g., kept as a trade secret) without any easy way to create a detailed executable model.

To address all these problems, we describe EmPro, which is a platform for emulation of the environmental conditions and energy supplies, as well as a profiling tool for WSN. According to pre-programmed sequences, EmPro accurately outputs electrical signals to emulate all possible inputs to a sensor system, including the power sources, the output signals of sensing devices, radio activities, and digital inputs from external devices.

This emulation approach enables researchers to debug and verify their hardware and algorithms in their laboratories in a realistic manner, but without the lack of controllability and reproducibility of the real physical environment. In addition to actuation, EmPro can be used in profiling mode to capture the behavior of WSNs and provide intensive analysis in terms of power consumption, radio performance, response time, and so on. EmPro enables quantitative evaluation and therefore meaningful comparison among WSN platforms, thereby enabling the user to choose the right platform for the application. We believe quantitative evaluation is the only way that the entire field of WSN can be sure that it is truly making progress.

This paper describes the design details of the hardware and software architectures of EmPro. We also present case studies of applying EmPro to the Eco and MICA2 WSN platforms. We expect the use of EmPro to expedite the development processes of WSN, and EmPro can also be used as a standard benchmarking tool for WSN platforms.

II. Related Work

Related work can be roughly divided into *WSN simulators* and *signal emulators*. The former executes a functional model of the WSN on one or more computers, possibly with direct interactions with some sensor nodes; the latter generates signals to mimic those observable in a deployment environment.

WSN simulators are those that simulate aspects of a WSN on one or more computers. Although higher level simulators such as ns2 are commonly used, here we focus our discussion on lower-level, more detailed models. Most such simulators have been implemented for specific hardware or software platforms. TOSSIM is the TinyOS Simulator, which enables the simulation of programs written in nesC for TinyOS without requiring the sensor hardware [1]. It is useful for software development, but for general hardware benchmarking, as the battery, RF, and other conditions, it will require much more sophisticated modeling effort.

EmStar [2] improves on TOSSIM by enabling mixing of actual Mote hardware with simulated Motes. In fact, each simulation instance can access a real RF channel in the *ceiling array*. MoteLab [3] is a testbed that consists of Mote hardware connected to interface boards that can program and configure the Mote hardware remotely. However, it uses AC power rather than batteries or solar panels, and it does not control or aim to reproduce sensory input. WHYNET [4] is a testbed for more heterogeneous devices including not only WSNs but also WLANs and 3G cellular phones, although its main focus is on testing of wireless and networking aspects. Emulab [5] at University of Utah is a testbed consisting of several sub-testbeds: one is a fixed 802.11 wireless test bed; one is a cluster of several hundred PCs that simulate network traffic; the third, called Mobile Emulab [6], is a robotic testbed with Motes and Stargates on remotely controllable robots as well as static Motes with attached sensor boards. While it enables in-field testing with data, many other sensing conditions are difficult or impossible to reproduce exactly, and it has no provision for the energy emulation. It also does not provide the power and environment emulation capability needed for general benchmarking.

Emulation of input to systems has been done for specific domains. Lynch [7] uses a signal generator to emulate input to sensors for structural health monitoring for civil engineering applications. In addition to environmental sources, energy sources such as the battery have also been emulated by the B# system [8]. It runs an electro-chemical, closed-loop simulation in real-time and can supply power to actual WSNs with full reproducibility while accurately modeling rate capacity and rate recovery effects within 1% error. B# has been shown to be an indispensable tool in the evaluation of battery-aware RF power optimization in WSN [9]. B#'s closed-loop emulation accurately tracks the behavior of actual batteries, whereas open-loop battery emulation results in high error. One extension of B# is S#, which replaces the battery simulation model with a model for solar panels [10]. These domain-specific emulators have proven very valuable in enabling the hardware to be tested with actual stimuli. What is needed is to generalize the concept to all of the possible signals to which a WSN is sensitive.

III. System Architecture of Wireless Sensor Nodes

Many different wireless sensor nodes have been developed in recent years, but most of them share the same fundamental system architecture. As depicted in Fig. 1 they usually consist of four subsystems: *microcontroller*, *sensing/actuating devices*, *radio interface*, and *power supply* [11], [12], [13], [14], [15], [16]. The subsystems interface with one another using digital/analog signals.

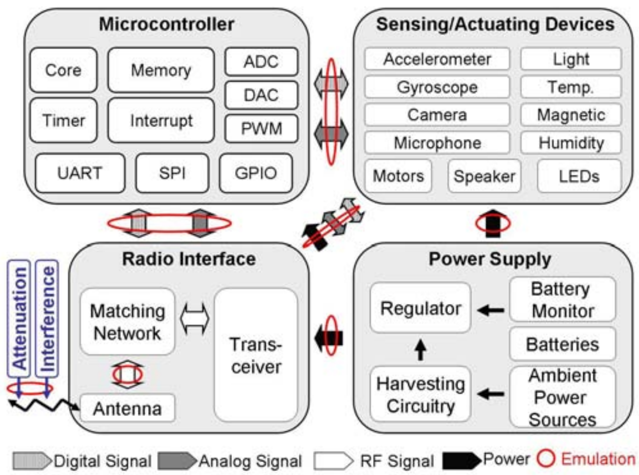

Fig. 1. System Architecture of a Wireless Sensor Node.

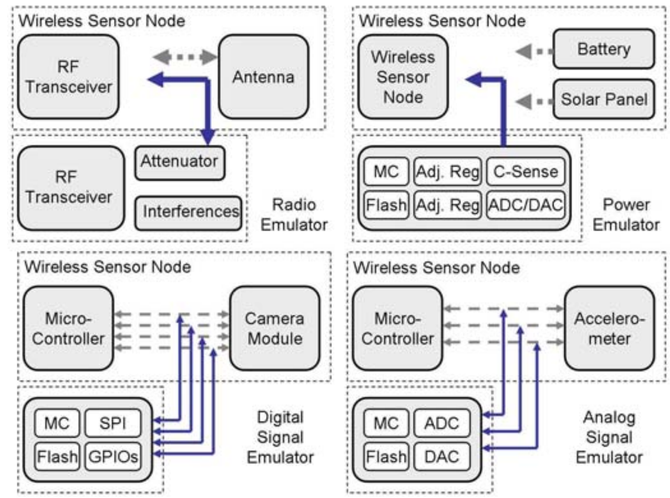

Fig. 2. Emulation Ideas for Wireless Sensor Node: Radio, Power, Digital Signal, and Analog Signal Emulation.

In this section, we first examine each subsystem of WSN system architectures. Then, we present our key ideas for emulation and monitoring.

A. *Microcontroller*

The microcontroller subsystem includes a processing core, timers, interrupt lines, different types of memory devices (internal or external RAM, ROM, and flash), analog peripherals (ADC and DAC), and digital peripherals (UART, SPI, PWM, and GPIO). Analog and digital peripherals are used to interface with other subsystems. For instance, the microcontroller samples and digitizes analog output signals from an acceleration sensor using its AD converter. It also drives a camera module through its GPIO pins or SPI port. In addition, by use of its PWM port, the microcontroller can control a motor or speaker.

B. *Sensing/Actuating Devices*

The sensing/actuation devices can be categorized into digital and analog devices. The former are those with a digital interface (parallel, SPI, or I2C) to a microcontroller, while the latter are those that use analog signals (voltage or current). Note that our classification is based on the interface to the microcontroller, rather than the type of sensing device itself. For instance, although accelerometers, light sensors, gyroscopes, magnetic sensors, and temperature sensors are commonly available with analog interfaces, some of them have digital ones. Analog interfaces can be further divided into single-ended vs. differential pairs: the former is simpler to interface but can have higher noise, while the latter is commonly used when data quality is important. Modern miniature cameras tend to be digital because the in-camera digitization and image or video compression reduces the required bandwidth at the interface, making it easier and lower power for microcontrollers to handle.

C. *Radio Interface*

The radio interface consists of an RF transceiver, matching network, and antenna. The microcontroller controls the status of the RF transceiver (On/Off/Tx/Rx) and the transmission power level by a set of digital signals. The microcontroller and RF transceiver exchange data over SPI or a parallel bus. An analog signal is also used to report a Received Signal Strength Indication (RSSI) to the microcontroller, which samples it using an AD Converter. The transceiver outputs the RF signal and radiates it via an antenna into air, experiencing possible attenuation and interference.

D. *Power Supply*

The *power supply* subsystem not only supplies power to other subsystems but also exchanges digital/analog signals with the microcontroller. For example, it reports the remaining battery life time, instantaneous or accumulated current value, and instantaneous output voltage of a battery through digital/analog peripherals [17]. Also, energy harvesting circuitry for ambient power sources are controlled and monitored in the same way [18], [19].

E. *Key Idea for Emulation and Monitoring*

What we learned from the preceding study is that subsystems of a sensor node work together by exchanging signals (digital, analog, RF, and power). Our key

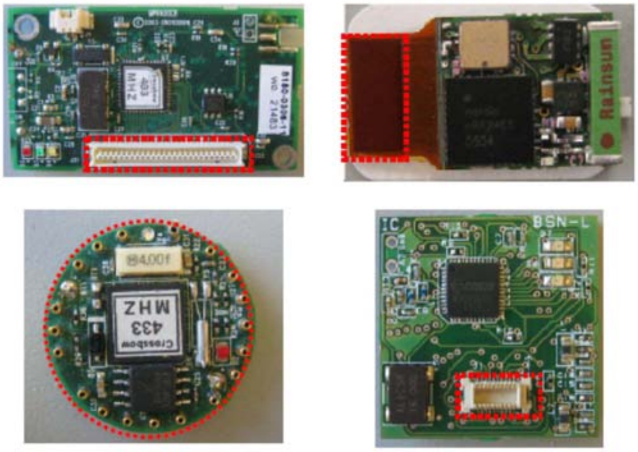

Fig. 3. Interface connector of wireless sensor nodes in clockwise order from top left: Mica2, Eco, BSN node, and Mica2Dot.

idea is to emulate and monitor these signals as shown in Fig. 2. For instance, if a radio emulator emulates an RF signal and feeds it to an RF transceiver of a WSN, then the WSN will react as if this signal comes from its neighboring nodes. Also, if an analog signal emulator feeds its signals to the microcontroller of a WSN, the WSN will not be able to distinguish them from an actual sensing device, even though the actual sensing device is not connected to the WSN. This idea also enables us to monitor WSN in a non-invasive manner. For instance, in order to monitor digital signals between the microcontroller and the camera module, we do not need to add any debugging code, which can slow down the system.

We can even emulate power sources such as a battery or solar panel and supply emulated power to the WSN. Power source emulation is important, because more systems are now designed to be *power aware* – that is, they adapt all aspects of their behavior according to the power availability status. A battery emulator not only takes into account all the non-ideal effects of real batteries, but more importantly, controls the state of charge precisely. As more and more sensors extend their battery life to months, years, or even decades, it will be impractical to take measurements over the entire battery lifetime; instead, a battery emulator will enable the charge state of the virtual battery to be set to exactly the desired level each time. This way, a benchmark can potentially be refined for several representative charge states of a battery: fully charged, half charged, and nearly discharged. A benchmark suite may also include actual sunlight profiles collected from representative regions around the world, such that the same sensor platform can be evaluated under diverse energy availability conditions.

In the process of emulating sensory signals, it may be necessary to bypass some devices. For example, one may choose to bypass an accelerometer and emulate vibration as an electrical signal by feeding it directly to the ADC. This way, the logical behavior on the MCU can be emulated, but the electrical behavior of the entire system is slightly altered, because the accelerometer does not consume the same power as before. To address this problem, we can compensate for this difference by using a load emulator in our other project [20]. It will draw additional current to make the power profile look nearly identical to the original, so that power sources can be emulated correctly.

One common feature of current wireless sensor nodes is that many of them have a connector that provides all digital and analog signals. Fig. 3 shows the connectors of different wireless sensor nodes. They are originally provided as an expansion port for the microcontroller/RF module to interface with a sensor board, or for firmware programming. We take advantage of this facility for the purposes of emulation and profiling.

IV. Design Concepts

EmPro is designed to provide an *integrated* and *distributed* emulation and profiling (monitoring) environment for wireless sensor networks. EmPro can solely emulate every component of a wireless sensor node such as a sensing/actuating device, RF transceiver, power source, and even RF attenuation and RF interference. A set of synchronized EmPros, which communicate with one another, can form a distributed emulation platform for an entire sensor network. In addition, our EmPro can be used to monitor every activity of each wireless sensor node. Furthermore, a combination of emulation and monitoring capabilities enables our EmPro to be used as a benchmarking tool for evaluating existing wireless sensor nodes. As shown in the labeled boxes in Fig. 4, EmPro emulates or monitors digital signals, analog signals, RF signals, and a power source. In this section, we present our methods for emulation and monitoring.

A. Digital Signals Emulation/Monitoring

To emulate and monitor digital signals of a wireless sensor node such as GPIOs, UART, PWM and SPI, we use a microcontroller equipped with all four digital interfaces and a flash memory. In emulation mode, it reads out time-stamped data from the flash memory and outputs digital signals according to the time stamps. In

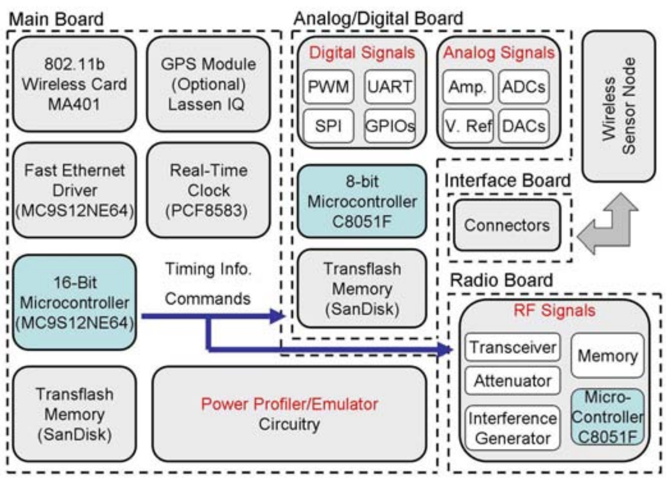

Fig. 4. Block Diagram of EmPro

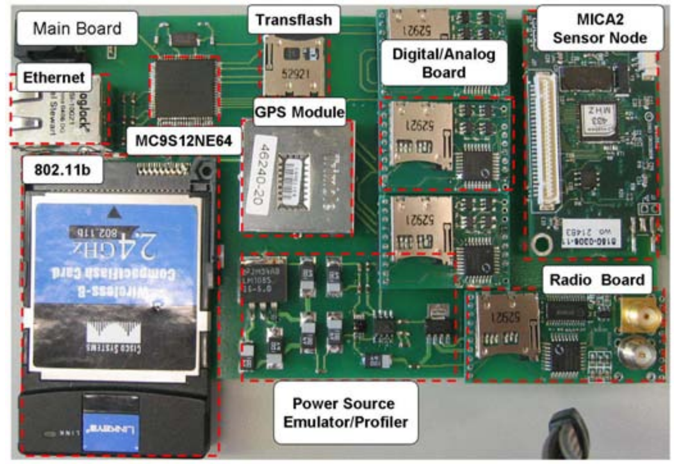

Fig. 5. Photo of EmPro: Digital/Analog, Radio, Interface Boards are plugged into Main Board

profiling mode, it monitors digital input signals from the wireless sensor node and stores them into the flash memory with proper time-stamps. This method can be used to emulate or monitor sensing/actuating devices that have a digital interface (camera module or digital temperature sensor), battery monitor IC, energy harvesting circuitry, and radio transceiver. For instance, as shown in Fig. 2, a radio transceiver such as CC1000 [21], CC2420 [22], and nRF2401 [23], all controlled through SPI and a few GPIOs, can be emulated and monitored by wiretapping between a microcontroller and transceiver. Also, an SPI or UART communication can be emulated and monitored in the same way.

B. Analog Signals Emulation/Monitoring

To emulate and monitor a wireless sensor node's analog signals such as analog I/Os, RSSI, and battery output voltage, we also use a microcontroller that has both ADCs and DACs, a flash memory, amplifiers for signal-conditioning, and voltage reference ICs. We use its ADC for monitoring and its DAC for emulation. This method can be used to emulate and monitor analog-output sensing/actuating devices such as an accelerometer, gyroscope, light sensor and so on, as shown in Fig. 2.

C. RF Signals Emulation/Monitoring

In the current implementation, we emulate environmental conditions that have the greatest impact on WSNs: attenuation and interference. For example, given that microwave (2.4GHz) gets absorbed by water, we would like to emulate how rainy weather or humidity level affects this frequency band. Interference can come from either other sensor nodes within the network, or from RF devices outside the network (e.g., laptop computers with WiFi, nearby radio stations, etc). To emulate these conditions, EmPro includes a programmable attenuator and interference generator to emulate air radiation. Off-the-shelf programmable attenuators are readily available. We use the Mini-FDPM system [24] as a digitally programmable, broadband RF signal generator. It will be described in more detail in Section V-C.

D. Power Source Emulation/Monitoring

EmPro can emulate multiple power sources. The hardware follows the principles of the B# battery emulator [8] for closed-loop emulation. It uses a current sensor to measure the load, runs one step of the simulator on the host computer to compute the voltage response, and then sets the output voltage on the board accordingly. Different types of batteries can be emulated by simply replacing the battery simulation program on the host computer. To emulate solar panels, depending on the way the solar cells are composed, we can plug in the booster board as in our S# solar panel emulator to increase the voltage range to up to 24V DC [10]. On the host side, a table-driven solar-panel simulator looks up the impedance value based on the current sunlight intensity and the load.

V. EmPro Design

The most critical issues in designing EmPro are the *accuracy* of emulation/monitoring and *real-time* operation. Considering that all wireless sensor node's subsystems operate interactively and simultaneously, it is difficult even for high-performance microprocessors. We resolved this problem by adopting a distributed emulation/monitoring architecture shown in Fig. 4. An EmPro system consists of one main board and three different types of daughter boards: Digital/Analog Board, Radio Board, and Interface Board. Each board except for the Interface Board has its own microcontroller (highlighted by blue color in Fig. 4) and memory, and they perform emulation/monitoring tasks in parallel as a small-scale distributed system. The main board coordinates all tasks running on each daughter board by sending commands and timing information over an SPI port.

In emulation mode, each Digital/Analog Board emulates digital or analog output signals of its corresponding sensing/actuating device as programmed in the flash memory. Also, the radio board emulates RF signals taking into account any possible attenuation and interference as if the RF signals were sent from neighboring sensor nodes. In addition, a power emulator supplies power to the wireless sensor node mimicking current-voltage characteristics of a power source.

In monitoring mode, the Digital/Analog Board and Radio Board monitor their corresponding components of the wireless sensor node, and store collected data into the flash memory. Also, the power profiler collects current and voltage profiles of the power source. These collected data can be played-back in emulation mode.

Additionally, a set of EmPros can form an emulation/monitoring network by use of their wired/wireless communication interfaces. This network enables us to build a hybrid experimental environment, where some sensor nodes are being emulated while the others being simulated. Time synchronization among EmPros in the network can be achieved by using a GPS module or running synchronization algorithms.

A. Main Board

Fig. 5 shows the Main Board of EmPro. It includes a 16-bit microcontroller (Freescale's MC9S12NE64 [25]), Fast Ethernet Connector, 802.11b Wi-Fi card (Netgear's MA401), Transflash Memory (Sandisk's 128MB [26]), Real-Time Clock [27], power-profiling/battery emulation circuitry, and interface connectors for daughter boards and a wireless sensor node. Also, it has a connector for Trimble's GPS module [28], which can be used for time synchronization and positioning. The microcontroller, which runs at 25MHz, is responsible for performing three tasks. The first task is to emulate or profile power sources. It controls the Power Profiler/Emulator circuitry, communicating with a host computer or playing-back power profiles stored in the flash memory. It also coordinates operations of daughter boards. It sends out timing information and commands for the daughter boards to emulate and monitor their corresponding devices accurately and timely. In addition, it communicates with a host computer or its neighbor emulator to form an emulation/monitoring network through either a Fast Ethernet port or 802.11b Wi-Fi PC card. Using this network, EmPro can provide a distributed emulation/monitoring environment and users can send emulation scenarios and monitor the status of each sensor node in real-time.

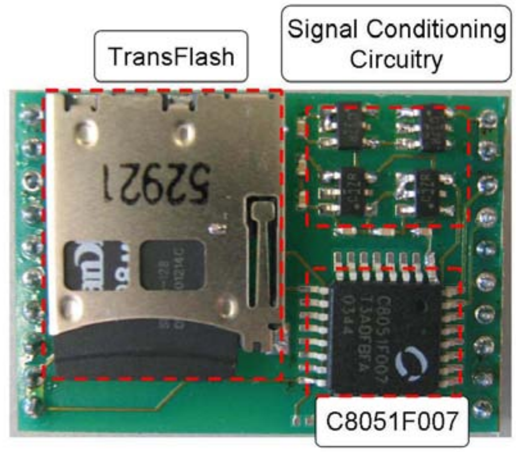

Fig. 6. Photo of EmPro Digital/Analog Board.

B. Digital/Analog Board

As shown in Fig. 6, the Digital/Analog Board consists of one 8-bit microcontroller (Silicon Lab's C8051F007 [29]), a Transflash memory, and signal conditioning circuitry. The C8051F007 has a 4-channel, 12-bit ADC, 2-channel 10-bit DAC, SPI, UART, and 8 GPIO ports. In emulation mode, this microcontroller reads out time-stamped emulation data stored in the flash and plays them back according to timing information and commands from the Main Board. In monitoring mode, it monitors a sensing/actuating device, collects monitoring data and stores them into the flash memory with proper time-stamps. This board can also be used to monitor digital signals between a sensor node's microcontroller and radio chip.

C. Radio Board

The Radio Board consists of a microcontroller, transceiver, flash memory, attenuator, interference generator input, and RF signal output. In emulation mode, it transmits RF signal to an RF transceiver of the wireless sensor node. The RF signal first goes to an attenuator (HMC472LP4), which can attenuate the signal from

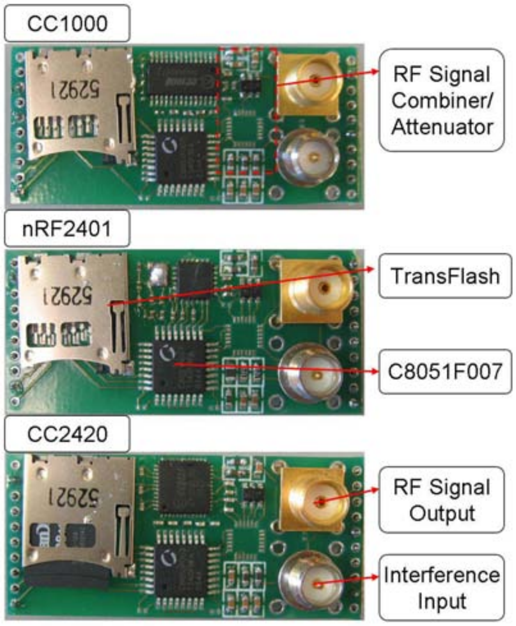

Fig. 7. Photo of EmPro Three Radio Boards: CC1000, nRF2401, and CC2420

60.5dB to 91.5dB (HMC472LP4 + 60dB passive attenuator). This attenuator is to emulate possible attenuation of the RF signal by air or weather condition (for example, rain or humidity). The attenuation level is controlled by the microcontroller according to an emulation scenario stored in the flash memory. Next, the RF power combiner (SCN-2-35) adds interference generated by an interference generator. This emulates interference by any other RF devices or neighbor sensor nodes. We use the Mini-FDPM [24] as the interference generator (Fig. 8). Mini-FDPM implements a broadband frequency synthesizer from 10MHz to 3GHz at power levels from -20dBm $\sim$ 23dBm, all digitally programmable over Fast Ethernet. In addition, a heterodyne detector on the Mini-FDPM enables EmPro to tune to any frequency it can synthesize for both amplitude and phase comparisons. This setup enables EmPro to operate in all frequency bands used by virtually all of today's WSN platforms.

Because different wireless sensor nodes use different radio transceivers, as shown in Fig. 7 we designed three radio boards that use the CC1000 [21], CC2420 [22], and nRF2401 [23] transceivers, respectively. When we need a different radio interface such as BlueTooth, we can just design one more Radio daughter board without any significant modification to EmPro.

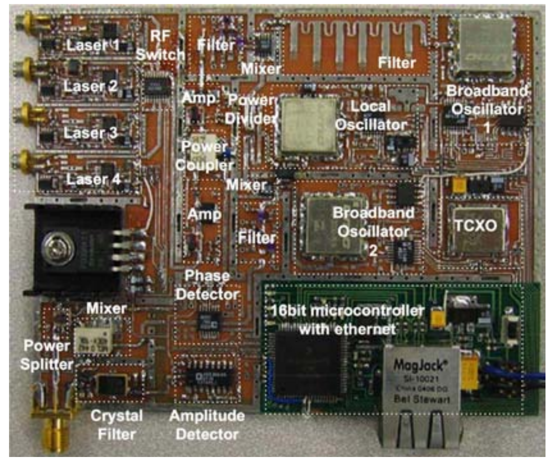

Fig. 8. Photo of Mini-FDPM: an interference generator

D. Interface Board

We use an interface board between a wireless sensor node and our EmPro instead of directly connecting them. EmPro currently has four different Interface boards for MICA2, MICA2DOT, Eco, and BSN node [30]. By having this interface board, we can use our EmPro for virtually all exiting wireless sensor nodes as long as they have pin-outs for their digital/analog signals. Each wireless sensor node needs its own interface board to be connected to EmPro.

VI. Evaluation

This section presents evaluation results of EmPro. The ability to emulate energy sources such as Lithium-ion batteries and solar panels has been reported previously. Here we first show the emulation and profiling accuracy of the Digital/Analog Board. Next, we demonstrate how the attenuation of the Radio Board can be used to emulate RF signal attenuation.

A. Accuracy of Digital/Analog Board

We measure the emulation and monitoring accuracy of the Digital/Analog Board, as shown in Fig. 9. Our signal source is the voltage output of the ADXL202 accelerometer [31] on the MICA2 wireless sensor platform. More specifically, we monitor its X-axis output signal for 20 seconds. The emulation and monitoring accuracy of EmPro can be measured by comparing three waveforms:

(1) original waveform measured by a DAS (National Instrument's PCI-6221 Data Acquisition System),
(2) the same waveform but digitized by EmPro, and
(3) EmPro's playback waveform, as measured by the DAS.

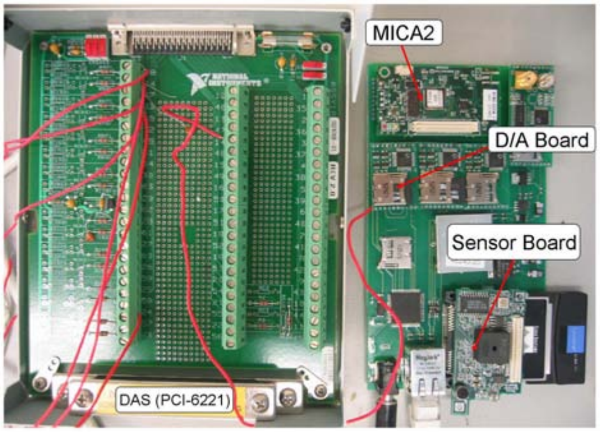

Fig. 9. Experimental Setup for Emulating Sensor Output Signals

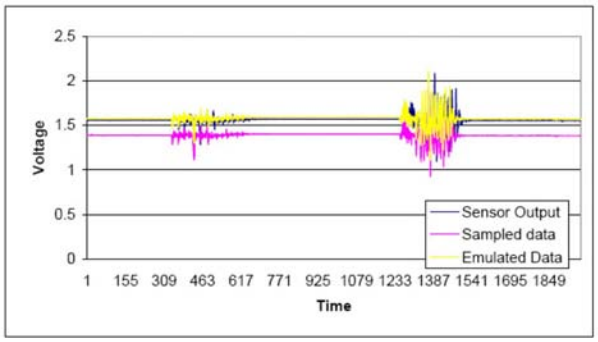

Fig. 10. Emulation and Monitoring Accuracy of the Digital/Analog Board

For this test, the sampling frequency is set to 100Hz for both EmPro and PCI-6221 DAS. The maximum sampling rate is limited primarily by the type of microcontroller and memory chosen, and is not an inherent limitation of the concepts. By the Nyquist Theorem, the accuracy is compared for the signal at 25Hz, because waveform (3) is the result of sampling by the DAS based on EmPro's samples.

Fig. 10 shows the three waveforms. In our experiment, both sampled and emulated data were accurate within 1.75% and 2.5%, respectively, comparing to the original sensor output.

B. Attenuator

To evaluate how effectively the Radio Board emulates RF signal attenuation, we conduct our experiment using the Eco wireless sensor node. As shown in Fig. 11, we directly connect the Radio Board with Eco via an RF cable. In this setup, the RF signal is transmitted through the cable, not air. Therefore, it is not affected by any environmental interference and we can solely observe

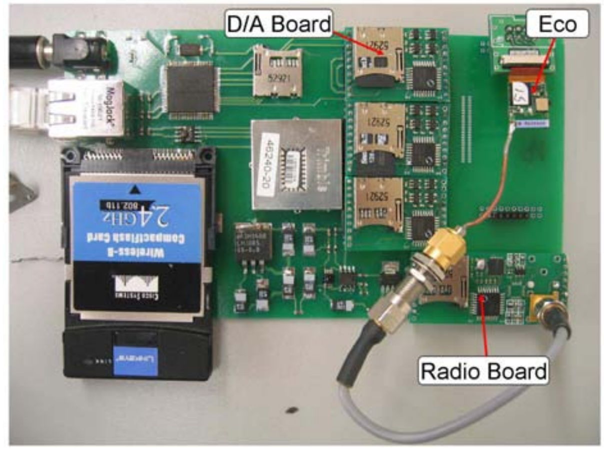

Fig. 11. Experimental Setup for Emulating Attenuation.

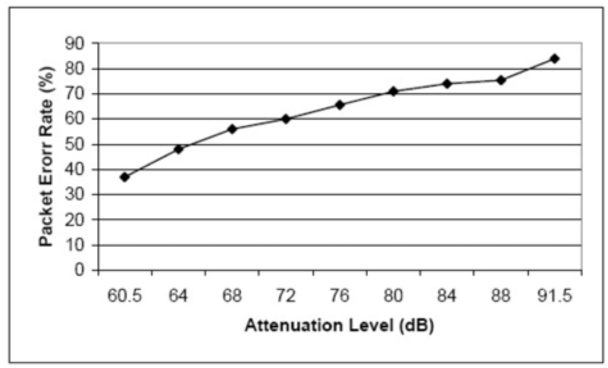

Fig. 12. Packet Error Rate vs. Attenuation Level.

the effect of the attenuator. In this experiment, we set the Eco sensor node as a transmitter and the Radio Board as a receiver. We use the Digital/Analog module to monitor Eco's behavior over a range of transmission power levels, packet sizes, and the number of packets transmitted. We set up the Radio Board to monitor the number of received packets and whether a packet is corrupted or not. Finally, all the data collected are reported to the Main Board so that it can compute the packet error rate. We actually measure the packet error rate, varying the attenuation level from 60.5dB to 91.5dB. Fig. 12 shows that the packet error rate increases proportionally to the attenuation level. While the result looks exactly as predicted, the more important point here is the controllability. That is, the ability to emulate attenuation synchronized to other sensory events is very powerful and enables us to conduct many different wireless communication experiments in the context of sensing events and energy in an entire sensor network.

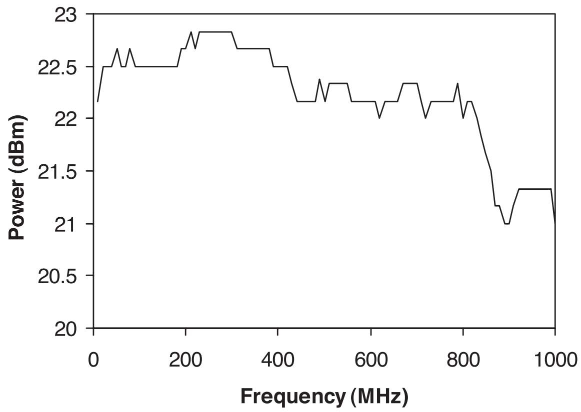

Fig. 13. RF Output Power Level over Frequencies for Interference Generation.

C. RF Interference Power

For interference generation, we measured the RF output level of our broadband frequency synthesizer from 10MHz to 1GHz in 10MHz steps. It is actually digitally controllable in 5KHz increments. Over the entire band, the gain is as high as almost 23dBm and is consistently above 21dBm. Given that Mica2 has a maximum Tx power of 10dBm, and most other low-power nodes 0dBm, our board is more than sufficient as an interference source, especially when used in conjunction with the attenuator.

VII. CONCLUSIONS

Today, it is nearly impossible to compare performance of alternative wireless sensor designs due to not only the lack of benchmarks, but also the inability to execute benchmarks in a reproducible way. Vendors today resort to citing figures from component datasheets, even though they are not very meaningful at the system level. At the same time, simulation is reproducible but is slow and can be inaccurate, and it is not applicable if the platform under evaluation is to remain a blackbox. To overcome these problems, we have presented EmPro, a powerful emulator and profiling system for quantitative evaluation of wireless sensor platforms. EmPro acts as the environment by providing all inputs to stimulate a sensor node, including digital, analog, power, and RF, but with full controllability. Our methodology combines the full reproducibility of simulation approaches with the accuracy and full speed of measurement-based approaches. More importantly, for the first time, end users will be able to run benchmark suites on many new WSN platforms for meaningful, quantifiable comparisons. This will be an essential step in making informed decisions when choosing a wireless sensor platform that best suits one's applications. More importantly, we believe that such quantitative evaluation will be the only way for the entire field of WSN to make progress in a real way.

ACKNOWLEDGMENTS

The authors would like to thank Keun Sik No and Qiang Xie for their help with Mini-FDPM. The authors also thank Jinfeng Liu, Dexin Li, and Sungjun Kim for their contributions to B# and S#.

This work was supported in part by the National Science Foundation grant CCR-0205712, NSF CAREER Award CNS-0448668, and NSF CMS-0509018. The Mini-FDPM work was sponsored in part by the National Institute of Health through an NTROI (Network for Translational Research on Optical Imaging) seed grant.

REFERENCES

[1] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in *ACM Sensys*, 2003.

[2] L. Girod, T. Stathopoulos, N. Ramanathan, J. Elson, D. Estrin, E. Osterweil, and T. Schoellhammer, "A system for simulation, emulation, and deployment of heterogeneous sensor networks," in *ACM Sensys*, Nov. 2004.

[3] G. Werner-Allen, P. Swiekowski, and M. Welsh, "MoteLab: A wireless sensor network testbed," in *IPSN*, Apr. 2005.

[4] Z. Ji, M. Marina, M. Varshney, Z. Xu, Y. Yang, J. Zhou, and R. Bagrodia, "WHYNET: A hybrid testbed for large-scale, heterogeneous and adaptive wireless networks," UCLA Computer Science Department Technical Report CSD-TR060002, Tech. Rep., Jan. 2006.

[5] "Emulab – network emulation testbed," http://www.emulab.net/, 2006.

[6] D. Johnson, T. Stack, R. Fish, D. M. Flickinger, L. Stoller, R. Ricci, and J. Lepreau, "Mobile Emulab: A robotic wireless and sensor network testbed," in *IEEE INFOCOM*, Apr. 2006. [Online]. Available: http://www.cs.utah.edu/flux/papers/robots-infocom06.pdf

[7] J. P. Lynch, K. J. Loh, T.-C. Hou, Y. Wang, J. Yi, C.-B. Yun, K. Lu, and C.-H. Loh, "Validation case studies of wireless monitoring systems in civil structures," in *Proceedings of the 2nd International Conference on Structural Health Monitoring of Intelligent Infrastructure (SHMII-2)*, Nov. 2005. [Online]. Available: http://www-personal.umich.edu/~jerlynch/papers/SHMII2Lynch.pdf

[8] C. Park, J. Liu, and P. H. Chou, "B#: a battery emulator and power profiling instrument," *IEEE Design and Test of Computers*, vol. March-April, pp. 150–159, 2005.

[9] C. Park, K. Lahiri, and A. Raghunathan, "Battery discharge characteristics of wireless sensor nodes: An experimental analysis," in *IEEE Conf. on Sensor and Ad-hoc Communications and Networks (SECON)*, Sep. 2005.

[10] D. Li and P. H. Chou, "Maximizing efficiency of solar-powered systems by load matching," in *Proc. ISLPED*, Aug. 2004.

[11] "MPR400/410/420 MICA2 Mote," http://www.xbow.com/Products/productsdetails.aspx?sid=72.

[12] Crossbow, "MICA2DOT," http://www.xbow.com/Products/productsdetails.aspx?sid=73.

[13] C. Park and P. H. Chou, "Eco: Ultra-wearable and expandable wireless sensor platform," in *Proc. Third International Workshop on Body Sensor Networks (BSN 2006)*, April 3-5 2006.

[14] H. Chung, C. Park, Q. Xie, P. H. Chou, and M. Shinozuka, "DuraNode: wireless network sensor for structural safety monitoring," in *SPIE Nondestructive Detection and Measurement for Homeland Security III Conference*, Mar. 2005.

[15] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *Proc. IPSN*, May 2005.

[16] E. M. Tapia, N. Marmasse, S. S. Intille, and K. Larson, "MITes: Wireless portable sensors for studying behavior," in *Ubicomp 2004: Ubiquitous Computing*, 2004.

[17] MAXIM, "DS2764 high-precision Li+ battery monitor with 2-wire interface," http://pdfserv.maxim-ic.com/en/ds/DS2764.pdf, 2003.

[18] K. Lin, J. Hsu, S. Zahedi, D. C. Lee, J. Friedman, A. Kansal, V. Raghunathan, and M. B. Srivastava, "Heliomote: Enabling long-lived sensor networks through solar energy harvesting," in *ACM Sensys*, Nov. 2005.

[19] X. Jiang, J. Polastre, and D. Culler, "Perpetual environmentally powered sensor networks," in *Proc. IPSN*, May 2005.

[20] F. Simjee and P. H. Chou, "Accurate battery lifetime estimation using high-frequency power profile emulation," in *ISLPED*, Aug. 2005, pp. 307–310.

[21] Chipcon, "CC1000 single chip very low power RF transceiver," http://www.chipcon.com/files/CC1000_Data_Sheet_2_2.pdf, 2004.

[22] ——, "CC240: 2.4 GHz IEEE 802.15.4/ Zigbee-ready RF Transceiver," http://www.chipcon.com/files/CC2420_Data_Sheet_1_3.pdf.

[23] Nordic Semiconductor, "nRF2401: Single chip 2.4GHz Transceiver," http://www.nordicsemi.no/files/Product/data_sheet/nRF2401rev1_1.pdf.

[24] K.-S. No and P. H. Chou, "Mini-FDPM and heterodyne Mini-FDPM: Handheld non-invasive breast cancer detectors based on frequency domain photon migration," *IEEE Transactions on Circuits and Systems*, vol. 52, no. 12, pp. 2672–2685, Dec. 2005.

[25] Freescale, "MC9S12NE64 16-bit microcontroller," http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC9S12NE64&nodeId=0162468636K100.

[26] "SanDisk TransFlash memory module 128MB," http://www.sandisk.com/Products/Item(1042)-SDQCJP-128-SanDisk_TransFlash_Memory_Module_128MB.aspx, 2006.

[27] "PCF8583: Clock/calendar with 240 x 8-bit RAM," http://www.semiconductors.philips.com/pip/PCF8583U.html, 2006.

[28] "Trimble GPS," http://www.trimble.com/, 2006.

[29] "C8051F00/1/2/5/6/7 mixed signal 32k ISP flash MCU family," http://www.silabs.com/public/documents/tpub_doc/dsheet/Microcontrollers/Precision_Mixed-Signal/en/C8051F0xx.pdf, 2003.

[30] B. P. Lo, S. Thiemjarus, R. King, and G.-Z. Yang, "Body sensor network – a wireless sensor platform for pervasive healthcare monitoring," in *Adjunct Proceedings of the Third International Conference, PERVASIVE 2005*, May 2005, pp. 77–80. [Online]. Available: http://www.pervasive.ifi.lmu.de/adjunct-proceedings/demo/p077-080.pdf

[31] Analog Devices, "Low-cost $\pm$2g dual-axis accelerometer with duty cycle output, ADXL202E," http://www.analog.com/UploadedFiles/Data_Sheets/53728567227477ADXL202E_a.pdf.