Aeon: Accurate Prediction of Power Consumption in Sensor Networks

Olaf Landsiedel Klaus Wehrle

Protocol Engineering and Distributed Systems Group
University of Tübingen, Germany
`firstname.lastname@uni-tuebingen.de`

***Abstract*— Due to limited resources, power consumption is a crucial characteristic of sensor node hardware and applications. Although many researchers contributed low power and energy aware applications, power consumption is still a limiting factor. As discovered by recent deployments, the sensor node lifetime is very often significantly shorter than expected. We present a model, which enables highly accurate quantitative estimation of the power consumption of a sensor node, thereby preventing such erroneous assumptions on node lifetime. A detailed and highly accurate prediction of the actual power consumption of a node allows to compare different approaches and to estimate the overall lifetime of a sensor network.**

I. Introduction

Recent research in low power hardware as well as highly embedded operation systems and wireless communication technologies enabled sensor networks. A sensor network consists of several thousand nodes, each with very limited processing, storage, sensing and communication abilities.

Their small size allows for ad-hoc deployment and enables monitoring the physical environment. Such deployment requires small and low cost devices. Due to these constraints sensor nodes are limited in terms of energy. For sensor network developers it is very important to estimate and evaluate the power consumption of applications. Although qualitative approximations can be derived from CPU duty cycles and transmitted packets, for detailed and accurate evaluation a precise quantitative approach is needed. We enable this by a detailed low-level model of the node.

For sensor networks, a deep evaluation in terms of energy consumption is crucial. Sensor applications are often required to run several years, and once deployed, it is nearly impossible to change their batteries. Erroneous models and estimations are non-reversible and highly expensive. As recently discovered [1], a sensor nodes lifetime may often be significantly shorter than expected. Thus, energy evaluation before deployment is extremely important. Although recent research provides many energy efficient or energy aware applications, only a very limited number [2, 3] has been deeply evaluated in terms of energy consumption by current measurements. Most implementations [4–7] count the number of packets sent and use this information as the one and only source to estimate the power consumption.

II. Related Work

Recently, Power Tossim (`PT`) [8] has been presented as an extension for the TinyOS [9] simulator Tossim [10] to provide energy models for the Mica2 [11] mote. Tossim is a discrete event simulator and simulates the execution of TinyOS applications. However, hardware abstraction (Fig. 1(a)) in Tossim leads to a high lack of detail and accuracy in PT.

For hardware developers it is very important to estimate the energy a device consumes. Since several years, energy models of processors have been developed [12, 13], focusing on the components of traditional CPUs, e.g. memory, cache, data-, and control path. Since the difference in energy per instruction is minimal, such models are not suited for software evaluation. The few tools [14, 15] available to estimate the energy consumption of software, focus on VLSI.

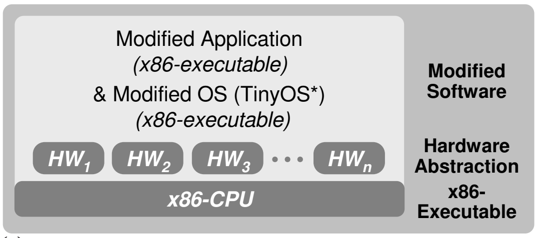
III. Highly Accurate Power Prediction

A sensor node consists of several devices, e.g. CPU, radio, sensors, and memory. A detailed model of all these devices is needed, to allow for a highly accurate prediction of power consumption. During program execution the application and external events change the state of a sensor node frequently: E.g. the CPU switches from a sleeping to an active mode as a result of interrupts, and devices like the radio are turned on and off based on the application's code. Each component state consumes a different amount of energy. Thus, the total current draw is determined by the sum of the component states. The exact knowledge of the state of the processor and its devices at every single point in time during program execution can only be achieved by an exact model and execution of real code, which only emulation of the sensor node can provide (Fig. 1).

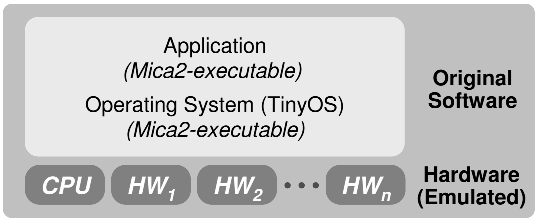
Since the Mica2 mote is a very popular sensor network node, we chose it to implement our energy models. However, the model can be easily ported to other sensor nodes.

A. Modeling Power Consumption

To model power consumption and calibrate our model, we developed applications for TinyOS 1.1, which kept the microcontroller and the attached devices in a certain state. Measuring the current draw with these benchmarks, allowed us to use standard mulitmeters and isolate the current of each component. We measured over twenty different states for three Mica2 nodes, see Table I for averaged results.



(a) Tossim uses hardware abstraction and compiles code to PC executables.



(b) Atemu emulates the hardware components of the Mica2 node and executes its code.

Fig. 1: Comparing Simulation and Emulation

Device	**Current**	**Device**	**Current**
CPU		**Radio** (933 MHz)	
Active	7.6 mA	Core	60 μA
Idle	3.3 mA	Bias	1.38 mA
ADC Noise	1.0 mA	Rx	9.6 mA
Power down	116 μA	Tx (-18 dBm)	8.8 mA
Power Save	124 μA	Tx (-13 dBm)	9.8 mA
Standby	237 μA	Tx (-10 dBm)	10.4 mA
Ext Standby	243 μA	Tx (-6 dBm)	11.3 mA
		Tx (-2 dBm)	15.6 mA
LED (each)	2.2 mA	Tx (0 dBm)	17.0 mA
		Tx (+3 dBm)	20.2 mA
		Tx (+4 dBm)	22.5 mA
		Tx (+5 dBm)	26.9 mA

TABLE I: Typical currents for the Mica2 mote

We based our implementation on Atemu [16], since it is currently the most complete emulator (Fig. 1(b)). It implements the CC1000 radio, the LEDs and all timers and interfaces of the microcontroller. To model energy consumption, we extended Atemu to record every state change in each component. By accumulating the cycles spent in each state, the consumed energy is estimated. With these so called traces the node is analyzed. They show how many cycles the microcontroller or a device spent in a certain state at a certain time.

To validate our model we measured standard sensor node applications with a precise oscilloscope over a period of time. Our evaluation for standard TinyOS applications shows high accuracy. For the TinyOS `Blink` application the average error is about 0.4% (standard deviation: 0.24). Measurements for applications using the radio show a slightly higher error, here clock skew and other inaccuracies of electronic node components have a higher impact

Test Application	**Predicted Energy Consumption** (in mJ)					
	CPU		Radio		LEDs	Total
	active	idle	rx	tx		
`Blink`	0.34	601.6	0	0	197.7	799.6
`BlinkTask`	0.36	601.6	0	0	197.7	799.6
`CntToLeds`	0.74	601.4	0	0	593.0	1195
`CntToL&R`	74.0	569.0	1600	220.1	589.8	3273
`CntToRfm`	73.9	569.1	1600	220.1	0	2684
`Surge`	72.5	569.7	1704	37.4	0	2421
`MantisBlnk`	1362	0	0	0	198.2	1560

TABLE II: Component breakdown for TinyOS and Mantis applications. Applications were executed for 60 emulated seconds.

B. Example: TinyOS

Our model allows to evaluate the power consumption of each sensor node component. We ran such an evaluation for standard applications of TinyOS 1.1 and the Blink application of Mantis 0.3 [17]. Table II shows a component breakdown. Furthermore, a detailed current graph is provided for the `CntToLedsAndRfm` application (Fig. III). As standard Mantis applications currently do not make use of the CPU idle mode, comparing power consumption of Mantis and TinyOS does not provide valuable information.

C. Model Evaluation

The accurate quantitative model predicts power consumption for sensor network applications precisely. Our flexible approach allows comparison of power consumption of operating systems, e.g. Mantis vs. TinyOS. Additionally, different OS versions can be evaluated to see the progress in terms of energy savings, e.g. to test a new scheduler or power saving scheme.

Although PT and our approach base on nearly the same measurements[1], the resulting power consumption is quite different. The predicted cycles for idle and active CPU mode differ significantly (Table III). Tossim is a simulator using hardware abstractions for modeling. It does not model all interrupts and so all state changes of the CPU, resulting in a lack of accuracy in PT. For example, the radio in receive mode causes the SPI interface to fire interrupts approximately every 460 μs (Fig. 2), which wake up the CPU from idle mode every time, and are not modeled in PT. A discrepancy of more than 4700% (Table III) between PT and the presented emulation approach is the result. Such a miscalculation of power consumption in a deployed sensor network will result in extreme costs and may even render a sensor network useless far before the expected lifetime is exceeded.

We measured and built the energy model using a standard Ampere meter and small tracing programs, providing independence from sensor platform and operating system. A port to related sensor platforms like the Mica2Dot and MicaZ [11] is easy to achieve. Similar models can be applied to other

[1]Except the radio: Our nodes use the 933 MHz band and so consume more energy than nodes using the 433 MHz band.

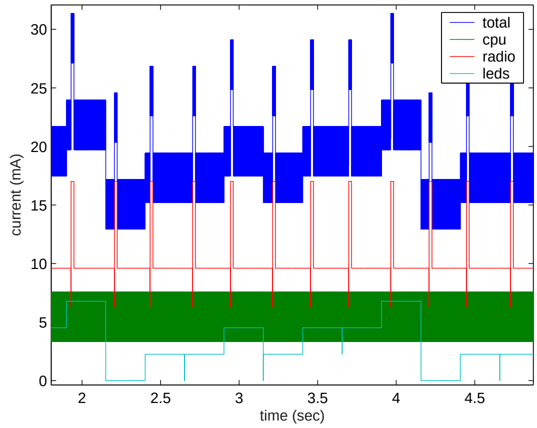


Fig. 2: Component breakdown for `CntToLedsAndRfm`.

Test Appli-cation	Power Consumption (in mJ)				Relative Error (in %)	
	Atemu		PT			
	active	idle	active	idle	active	idle
`Blink`	0.34	601.6	0.25	742.5	36.0	-19.0
`BlinkTask`	0.36	601.6	0.27	742.5	33.3	-19.0
`CntToLeds`	0.74	601.4	0.57	743.7	29.8	-19.1
`CntToL&R`	74.05	569.0	1.61	741.9	4499	-23.3
`CntToRfm`	73.95	569.1	1.54	741.9	4701	-23.3
`Surge`	72.59	569.7	1.50	727.9	4739	-21.7

TABLE III: Comparison of predicted CPU power consumption (in mJ). Applications were executed for 60 emulated seconds.

platforms like Nymph, PicoRadio and Cricket [17–19], when an appropriate emulator is available.

IV. Conclusion

We present highly accurate quantitative prediction of power consumption in sensor networks. This work forms the basis for our ongoing research, as it allows us to evaluate the energy awareness of routing, MAC layer protocols, and power saving schemes. Next to the evaluation of individual nodes, our model enables the analysis of complete sensor networks. Using a virtual battery, e.g. shutting down a node, when a energy limit is exceeded, the behavior of a sensor network can be evaluated. As nodes run out of power, new routing pathes have to be set up and eventually the network will cluster. Since clustering renders a sensor network useless, it is crucial to evaluate multihop and routing protocols towards this property. Based on this the level of redundancy required to prevent such a clustering can be estimated. Additionally, stack analysis enables energy evaluation of single methods in applications, providing energy profiling. As erroneous estimation of power consumption and lifetime results in high costs and may even render a deployed sensor network useless, accurate energy evaluation of applications and systems is a must.

References

[1] R. Szewczyk *et al.*, “Lessons from a Sensor Network Expedition,” in *Proc. of EWSN*, 2004.

[2] J. H. Wei Ye and D. Estrin, “An Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *Proc. of INFOCOM*, June 2002.

[3] L. D. Barbara Hohlt and E. Brewer, “Flexible Power Scheduling for Sensor Networks,” in *Proc. of IPSN*, April 2004.

[4] S. R. Madden *et al.*, “TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks,” in *Proc. of OSDI*, Dec. 2002.

[5] M. Duarte and Y.-H. Hu, “Distance Based Decision Fusion in a Distributed Wireless Sensor Network,” *Telecomm. Systems*, vol. 26, no. 2-5, pp. 339–350, 2004.

[6] A. Woo and D. E. Culler, “A transmission control scheme for media access in sensor networks,” in *Proc. of MobiCom*, July 2001, pp. 221–235.

[7] R. C. Shah and J. Rabaey, “Energy Aware Routing for Low Energy Ad Hoc Sensor Networks,” in *Proc. of WCNC*, March 2002.

[8] V. Shnayder and other, “Simulating the Power Consumption of Large-Scale Sensor Network Applications,” in *Proc. of SenSys 2004*, to be published.

[9] J. Hill, “System Architecture for Wireless Sensor Networks,” Ph.D. dissertation, UC Berkeley, Spring 2003.

[10] P. Levis *et al.*, “TOSSIM: accurate and scalable simulation of entire tinyOS applications,” in *Proc. of SenSys*, 2003, pp. 126–137.

[11] Crossbow Technology Inc., *Mica2, Mica2Dot and MicaZ Mote Datasheets*.

[12] P. Landman and J. Rabaey, “Activity-Sensitive Architectural Power Analysis,” *IEEE Trans. on CAD*, 1996.

[13] M. Kamble and K. Ghose, “Energy-Efficiency of VLSI Caches: A Comparative Study,” in *10th Int. Conf. on VLSI Design*, 1997, pp. 261–267.

[14] V. Tiwari *et al.*, “Instruction level power analysis and optimization of software,” *Journal of VLSI Signal Processing.*, pp. 1–18, 1996.

[15] Y. Li and J. Henkel, “A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems,” in *Proc. of the Great Lakes symposium on VLSI*, 1998.

[16] M. Karir, *Atemu - Sensor Network Emulator / Simulator / Debugger*, Center for Satellite and Communication Networks, Univ. of Maryland, 2003. [Online]. Available: http://www.isr.umd.edu/CSHCN/research/atemu/

[17] S. Bhatti *et al.*, “MANTIS OS: An Embedded Multithreaded Operating System for Wireless Micro Sensor Platforms,” *MONET, Special Issue on Wireless Sensor Networks*, 2004.

[18] J. Rabaey and other, “PicoRadios for Wireless Sensor Networks: The Next Challenge in Ultra Low-Power Design,” in *Proc. of the Int. Solid-State Circuits Conf.*, Feb. 2002.

[19] H. Balakrishnan *et al.*, *Lessons from developing and deploying the Cricket indoor location system*, MIT, Nov. 2003.