Component-based computation-energy modeling for embedded systems

Seewald, A.; Schultz, U.P.; Roeder, J.; Rouxel, B.; Grelck, C.

DOI
10.1145/3359061.3362775

Publication date
2019

Document Version
Final published version

Published in
SPLASH companion '19

Component-Based Computation-Energy Modeling for Embedded Systems

Adam Seewald
Ulrik Pagh Schultz
ads,ups@mmmi.sdu.dk
University of Southern Denmark
Denmark

Julius Roeder
Benjamin Rouxel
Clemens Grelck
j.roeder,b.a.l.rouxel,c.grelck@uva.nl
University of Amsterdam
Netherlands

Abstract

Computational energy–efficiency is a critical aspect of many modern embedded devices as it impacts the level of autonomy for numerous scenarios. We present a component–based energy modeling approach to abstract per–component energy in a dataflow computational network executed according to a given scheduling policy. The approach is based on a modeling tool and ultimately relies on battery state to support a wider range of energy–optimization strategies for power–critical devices.

CCS Concepts • **Hardware → Power estimation and optimization**; • **Computing methodologies** → *Modeling and simulation*; • **Computer systems organization** → *Embedded systems*; Multicore architectures.

Keywords Energy Profiling, Energy Modeling, Embedded Platforms, Component-Based Development

ACM Reference Format:
Adam Seewald, Ulrik Pagh Schultz, Julius Roeder, Benjamin Rouxel, and Clemens Grelck. 2019. Component-Based Computation-Energy Modeling for Embedded Systems. In *Proceedings of the 2019 ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity (SPLASH Companion '19), October 20–25, 2019, Athens, Greece.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3359061.3362775

1 Introduction

Energy modeling for complex unpredictable embedded systems can be a challenging task. Modern embedded architectures range from microcontroller powered devices to heterogeneous platforms executing parallel programs on different cores and computational units. Application–side, parallel software featuring independent tasks runs as a result of different components being executed together. A component–based approach can significantly reduce development time and overall complexity. Although many modern systems benefit from it by, e.g., ensuring better scalability over time, its adoption in the embedded domain remains marginal [4]. Indeed, energy efficiency, time, security, and other non–functional properties requirements still pose a major obstacle to its adoption [1].

We propose a component–based energy modeling approach to assess the energy efficiency of power critical devices, built upon our previous work on coarse–grained modeling [7]. Coarse–grained modeling consists of profiling a discrete set of configurations by varying different component configuration parameters, measuring the overall power consumption, and approximating any missing values. The proposed approach handles computational aspects of heterogeneous hardware. For instance, it takes into account but is not limited to, different computational units such as CPU and GPU. For the software, it models an arbitrary number of components interacting together. The approach can be of particular interest in an energy–aware scheduling technique, where a scheduler varies non–functional properties based on the generated energy model. As an example, the developer specifies application–level energy and a worst–case execution time measure. The scheduler generates the best possible schedule and configuration that meets the requirements.

Our work is part of the TeamPlay project, that aims to develop formally–motivated techniques to address non–functional properties satisfiability [1], and is implemented by a profiling tool named `powprofiler` distributed under MIT license (https://bitbucket.org/adamseew/powprofiler). The current implementation builds an energy model designed for the optimization of computational energy, but it can be used in a future setup with a scheduler to potentially exploit optimized planning decisions such as dynamic and static optimal scheduling.

2 Energy Models

In the embedded systems domain, computational energy is traditionally modeled to lower power consumption. System–level configurations are among the most used [3], while some

SPLASH Companion '19, October 20–25, 2019, Athens, Greece

ACM ISBN 978-1-4503-6992-3/19/10.
https://doi.org/10.1145/3359061.3362775

approaches focus on optimal chip configuration [5]. Others include different computational units [2], rely on machine learning techniques [8], and some focus on a coarse–grained modeling [7].

Energy models are of particular interest for mobile robots, as they often present a constrained power budget. To this extent, a drone use case featuring pattern recognition on open water based on a coarse–grained component model was developed. Different components, like edge detection and base station communication, cooperate such that an image is stored for further analysis if an object is detected by the drone flying over the sea. The modeling strategy consists of defining different acceptable QoS parameters per component formally defined as non–functional properties.

2.1 Component-Based Modeling

To address the possible uncertainty resulting from running components in a wider, dependency–constrained network, our approach relies on a component–based modeling technique. A simple static scheduling algorithm [6] that runs components sequentially while calling `powprofiler` is generated such that the scheduling algorithm executes components individually at profile–time. In particular, when a given component executes, a profiling thread is invoked through the callback `powprof_start`, along with the configuration that the profiling corresponds to. The scheduler similarly invokes the callback `powprof_stop` to stop the profiling thread. In our system, dependencies between components are handled by the scheduler during profiling, such that components only execute once all of their inputs are available. In fact, in a complex application, the execution of a given component may often influence the energy consumed by others. All the profiled data are used to generate an energy model per–component later saved in a non–functional properties definition file for the subsequent use by the TeamPlay toolchain. Base energy, that consists of profiling the system without components for a certain amount of time, can be subtracted from the per–component energy model to obtain a component cost value.

Per–component energy can be used by the scheduler to select different implementations of components to meet an energy budget. The approach can considerably reduce profiling time and will potentially consist of adding costs one on top of each other to model components in parallel within certain system–specific energy boundaries. The resulting value, even if potentially higher as it might include the energy cost of the system utilities along with the components, can be still a valid approximation.

2.2 From Abstraction to Real Behavior

In the current setup, when evaluating an energy model per–component in a wider dataflow computational network, the model can approximate energy consumption of the network with the cost values. Both coarse–grained (presented in our previous work) and component–based approaches can be used as a basis for a wider power modeling decision. The current approach is faster in execution and simpler from the developer side, as it requires less effort while helping to determine the best possible configuration within the energy and QoS constraints. Its validity can be assessed by an equivalent coarse–grained energy model more closely related to real behavior since it was validated against a fine–grained model and showed a close relation.

2.3 Battery State of Charge

A robust correlation in `powprofiler` to real scenarios is achieved by modeling the battery state of charge (SoC), as an abstraction for handling energy efficiency of power–critical devices. SoC–featured model can be used for an energy–efficient scheduling policy as an optimal policy can considerably impact battery life. For instance, a stable power drain of components scheduled in a battery aware fashion can drain less energy, as distributing the same computation in a smaller number of power–intensive operations. A utility that correlates the SoC to the optimal configuration with an optimization algorithm is considered future work.

Acknowledgments

This work is supported and partly funded by the European Union's Horizon2020 research and innovation program under grant agreement No. 779882 (TeamPlay).

References

[1] 2019. Public deliverables of the TeamPlay Horizon2020 project. https://teamplay-h2020.eu/index.php?page=deliverables. Accessed: 2019-08-25.

[2] Peter E Bailey, David K Lowenthal, Vignesh Ravi, Barry Rountree, Martin Schulz, and Bronis R De Supinski. 2014. Adaptive configuration selection for power-constrained heterogeneous systems. In *2014 43rd International Conference on Parallel Processing*. IEEE, 371–380.

[3] Princey Chowdhury and Chaitali Chakrabarti. 2005. Static task-scheduling algorithms for battery-powered DVS systems. *IEEE transactions on very large scale integration (VLSI) systems* 13, 2 (2005), 226–237.

[4] Ivica Crnkovic. 2005. Component-based software engineering for embedded systems. In *Proceedings of the 27th international conference on Software engineering*. ACM, 712–713.

[5] Ami Marowka. 2017. Energy-aware modeling of scaled heterogeneous systems. *International Journal of Parallel Programming* 45, 5 (2017), 1026–1045.

[6] Benjamin Rouxel, Roeder Julius, Altmeyer Sebastian, and Grelck Clemens. 2019. A time, energy and security coordination approach. In *10th International Workshop on Analysis Tools and Methodologies for Embedded and Real-Time Systems (WATERS 2019)*.

[7] Adam Seewald, Ulrik P Schultz, Emad Ebeid, and Henrik S Midtiby. 2019. Coarse-grained computation-oriented energy modeling for heterogeneous parallel embedded systems. (2019). Manuscript submitted for publication.

[8] Gene Wu, Joseph L Greathouse, Alexander Lyashevsky, Nuwan Jayasena, and Derek Chiou. 2015. GPGPU performance and power estimation using machine learning. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 564–576.