Early Energy Estimation in the Design Process of Networked Embedded Systems

Patrick Heinrich and Christian Prehofer
Fraunhofer Institute for Communication Systems ESK, Hansastr. 32, 80686 Munich, Germany

Keywords: Embedded Systems, Energy-efficiency, Network-wide Optimization, Adaptive Systems, Automotive.

Abstract: This paper focuses on estimating energy consumption in networked, embedded systems during the early stages of design. In such systems with long development cycles, early decisions regarding hardware and software have a major impact on energy consumption, which places constraints on the design during later stages of the development process. The main objective of this paper is to model the energy consumption throughout the entire design process and to provide a structure for more detailed models and validation. This requires determining which design decisions and system parameters are available in each of the development phases. Using this approach as a basis, energy consumption models are created for each phase. By relying on different estimation techniques, energy consumption estimates are formulated based on the information that is available at every development phase. The main goal is to illustrate a design flow that produces increasingly accurate estimates in each phase.

1 INTRODUCTION

This paper focuses on estimating energy consumption in networked, embedded systems during the early stages of design. As embedded systems become more prevalent and powerful, they are consuming more energy. Our research concentrated on the area of networked, embedded systems commonly found in automobiles, aircraft and industrial systems. In today's luxury-class vehicles for instance, the electrical and electronic components draw up to 2.5 kW (Little, www.adlittle.com); (Monetti and Ulshöfer, 2011). Compared to what the engine requires, 2 kW seems small. However, the electrical components consume energy during every mode of operation, even when in standby mode. Electrical engines consume most of their energy during acceleration and even here the maximum power is seldom demanded. An increase of 100 W thus means that fuel consumption rises by 0.1 liter per 100 km, leading to an increase in CO_2 emissions of 2.5 g per km (Monetti and Ulshöfer, 2011). This illustrates the considerable potential for energy savings, an aspect that must be factored in during the development process.

Embedded systems designers, such as those active in the automotive industry, are frequently given energy consumption requirements for the finished product. This makes it necessary to estimate the energy consumption early in the design process. This also makes it possible to evaluate different designs with respect to energy consumption. Because the automotive industry has especially long development cycles, hardware and software choices must be made very early during the design process (Weber, 2009). Optimizing energy consumption early in the design process can therefore be a major challenge. Networked systems feature a wide range of technologies and topologies that significantly impact energy consumption down the road. For instance, the placement of functionality within an ECU impacts partial networking modes, which involves deactivating certain system components that are not being used. This can reduce energy by as much as 30 percent (Monetti and Ulshöfer, 2011).

Figure 1, which depicts the various phases of a typical system development process, reveals that the energy savings potential gradually decreases over the course of development. Energy savings estimates are also imprecise, as shown by the dotted line.

In the early phase, the developer designs the electronic control unit (ECU) system based only on the features outlined in the specification. Design choices such as software components or network technologies impact the energy consumption. Modifying these design choices at a later point in the

DOI: 10.5220/0004334502140220
In *Proceedings of the 3rd International Conference on Pervasive Embedded Computing and Communication Systems* (PECCS-2013), pages 214-220
ISBN: 978-989-8565-43-3

process is often difficult or too expensive to carry out (Ross and Perry, 1999). In many design phases, the designer has to evaluate different design variations to realize optimal energy consumption, but a detailed analysis is not possible in these phases.

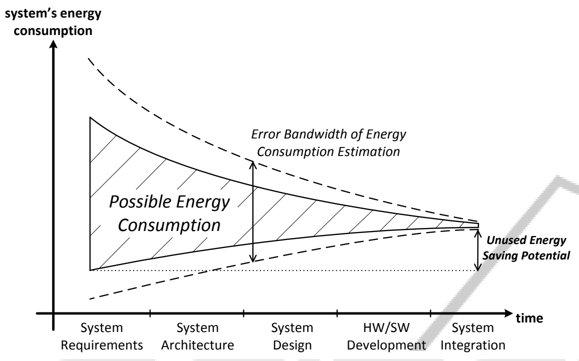
system's energy consumption
Error Bandwidth of Energy Consumption Estimation
Possible Energy Consumption
Unused Energy Saving Potential
time
System Requirements
System Architecture
System Design
HW/SW Development
System Integration

Figure 1: Energy Design Space during System Development.

The main objective of this paper is to model the energy consumption throughout the entire development process. To do this, we have to determine which design decisions and system parameters are available in the various phases. Using this information, we can then develop energy consumption models for each phase. While using several different estimations, the main goal is to show a design flow that produces increasingly accurate estimates in each phase.

A wealth of research has been conducted into designing energy efficient systems comprised of individual components. What has been lacking is corresponding research considering the entire networked embedded systems. With these systems, it is particularly important to examine early on to what extent a specific design impacts energy consumption.

Our work focuses on systems comprised of multiple ECUs connected via a communication network. While this type of system can be found in various environments, the impetus for our research stems from the automotive industry. Vehicle systems feature many technologies with energy-saving designs, such as the partial networking modes available in AUTOSAR (www.autosar.org). The next step is to determine how to use these technologies as efficiently as possible. The aim of this paper is to illustrate how system developers can easily and accurately evaluate the impact of their design choices on energy consumption during the early stages of the design process. The results can be used beyond automotive embedded systems.

2 IMPACT OF ENERGY CONSUMPTION ON THE SYSTEM DEVELOPMENT PROCESS

This section outlines the parameters for estimating energy consumption when developing networked embedded systems. These parameters are systematically analyzed with respect to accuracy and availability during the development process.

Developers have to make design choices in each phase. These decisions are intertwined and define the system in more detail as the development process moves forward. For example, software design begins by defining the required features, which are then evolved into more detailed, intertwined individual tasks. This is followed by the definition of the internal structure of the tasks. After implementation, the detailed timing parameters are available.

The typical system development processes are structured into distinct phases as described in the waterfall model (Boehm, 1981) or V-Model XT (V-Modell XT: Part 1, www.ftp.tu-clausthal.de). The relevant phases which influence the energy consumption and the available information to estimate the energy consumption are discussed below.

Table 1 summarizes the information available at every design phase. The objective here is to illustrate how accuracy increases from phase to phase or how the available information is replaced by more accurate data in the following phase. The parameters presented here are used to estimate the energy consumption in the next section. The values of the parameters are not only influenced by the design choices (the task allocation), but how the choices are implemented, such as the chosen hardware or operating system. While much research has been carried out in the area of energy-saving technologies at the component level, less attention has been paid to understanding the potential for energy saving at the network-wide level.

2.1 System Requirements

This phase defines the functional and non-functional requirements of the entire system, such as the number of features or maximum energy consumption. Afterwards, developers have a clear picture of the required features, which allows them to estimate the number of ECUs and how the features are allocated to the individual ECUs. (In 2005, automobiles were equipped on average with around 65 ECUs and 165 features (Little, www.adlittle.com)). Energy consumption can be estimated by classifying each feature (navigation system, engine controls, electric windows for instance) based on the required processing power, necessary ECU size and what function the feature is intended to provide. The required size and purpose of the ECUs are also factored in to estimate the energy consumption. Large semiconductor manufacturers such as Freescale Semiconductors already group their microcontrollers according to the function or application. Freescale has six classes of automotive applications such as (Freescale's portfolio, www.freescale.com) advanced driver assistance systems, chassis, safety and infotainment. Each class of microcontroller features specific energy consumption parameters, which can be used to define ECU energy classes.

2.2 System Architecture

In embedded systems, the features are implemented in a distributed fashion to create specialized ECUs that are installed in close proximity to the function they serve. One feature consists of several tasks, which are often allocated across different ECUs. This phase defines the system architecture, including hardware, number of ECUs, topology and software. In other words, it defines how the tasks are allocated across the ECUs and how they interact. Understanding how the tasks interact also permits the developer to estimate the required network bandwidth. While the internal structure of the tasks is unknown, the developer can estimate the processing power by relying on various processing power classifications, e.g. derived from existing tasks.

Industry benchmark tests are already available to evaluate automotive microcontroller performance. The Embedded Microprocessor Benchmark Consortium (AutoBench 1.1, http://www.eembc.org) defines 15 benchmark tests that include angle to time conversion, bit manipulation, fast Fourier transform and road speed calculation. The classifications can be formed by grouping tasks that require similar processing power, thus allowing a more detailed estimate.

2.3 System Design

This phase specifies the elements of the previous phase in more detail. With respect to the software, this means the internal structure of the tasks, including the algorithms and operating systems used. This makes it possible to estimate the execution time of the tasks, which is carried out using function point analysis (Albrecht, 1979), cyclomatic complexity (McCabe, 1976) or equivalent metrics. The actual execution times are available after the software has been implemented on specific hardware. After choosing the processor and other hardware, the developer can then estimate the energy consumption of the ECU.

2.4 Hardware/Software Development

This phase involves hardware and software implementation, which provides more detailed information concerning task processing time, worst case execution time (WCET) and hardware power consumption. The beginning of this phase can be used to simulate ECU execution and active/sleep times or to measure the energy consumption using evaluation boards, assuming they are nearly identical to the installed hardware. A fairly accurate estimate of the energy consumption is then possible at this point.

Table 1: Breaking down the information into more detail during the development process.

	Hardware	Software	
System Require-quire-ments	➢ ≈ Number of ECUs ➢ ECU Energy Class	➢ Number of Features ➢ Feature Computation Class	
System Archi-tecture	➢ Number of ECUs *➢ ECU Energy Class* ➢ Topology ➢ ≈ Network Band-width	➢ Software Architecture (tasks as black box): - Dependencies - Cycle Times - Allocation	➢ ≈ Numb. of Tasks ➢ Task Computa-tion Effort Class
System Design	*➢ Number of ECUs* *➢ Topology* ➢ Network Bandwidth ➢ ≈ ECU- and Network-specific Energy Consump-tion	*➢ Software Architecture (tasks as black box):* *- Dependencies* *- Cycle Times* *- Allocation*	➢ Numb. of Tasks ➢ Task (internal) Computation Effort, e.g. using function point analysis
HW/SW Devel-opment	*➢ Number of ECUs* *➢ Topology* *➢ Network Bandwidth* ➢ ECU- and Network-specific Energy Consumption ➢ ECU Active and Sleep Times	*➢ Software Architecture (tasks as black box):* *- Dependencies* *- Cycle Times* *- Allocation* - Active and Sleep Times	*➢ Numb. of Tasks* ➢ Task WCET

3 ESTIMATING ENERGY DURING THE DEVELOPMENT PROCESS

This section outlines equations for estimating energy consumption in each of the development phases. The equations rely on the information that is available in each phase, building upon each other as the development process progresses. Since information might be missing or estimated in the various phases, each equation features a specific degree of accuracy that improves over time (see figure 1).

3.1 System Requirements

Equation 1 depicts an energy estimates based on the number of features and ECUs. The number of features n_{feat} of one feature class is multiplied by the individual feature computation factor c_{feat}. Multiplying by the required energy consumption per ECU, which is represented by the ECU energy classes $E_{HWclass}$, then provides the estimated energy consumption per ECU.

$$E_1 = \sum_{i=1}^{\substack{Number\\of\ ECUs}} \left[\underbrace{\left(\sum_{j=1}^{\substack{Number\ of\\Feature\\Classes}} n_{feat,i,j} \cdot c_{feat,j} \right)}_{Software} \cdot E_{HWclass,i} \right] \quad (1)$$

E_1 represents the energy estimate of the entire system within the system requirements phase. The accuracy of this equation is based on the number and accuracy of the feature computation and hardware energy classes.

3.2 System Architecture

Equation 2 depicts an energy consumption estimate using the software and hardware architecture information. This means the number of tasks of one class n_{task} is multiplied by the task computation class factor c_{task}. $E_{HWclass}$ represents the ECU energy classes as mentioned above, but is more accurate since the exact number of ECUs is known. At the system architecture level, the energy consumption of communication E_{com} can be estimated since the hardware architecture and the allocation of tasks are known factors. (Note: While equation 1 and 2 appear to be very similar, it is important to note that one feature consists of multiple tasks. The sum of the tasks is therefore much more detailed. This is also caused by more task classes than feature classes.)

$$E_2 = \sum_{i=1}^{\substack{Number\\of\ ECUs}} \left[\underbrace{\left(\sum_{j=1}^{\substack{Number\\of\ Task\\Classes}} n_{task,i,j} \cdot c_{task,j} \right)}_{Software} \cdot E_{HWclass,i} \right] + E_{com} \quad (2)$$

E_2 represents the energy estimate of the entire system within the system architecture phase. The accuracy of this equation is based primarily on the number and degree of accuracy of the task computation and hardware energy classes.

3.3 System Design

Equation 3 shows the energy estimate using the early-stage information regarding hardware power consumption and task internal software structure. The equation uses metrics to estimate the evaluation times of the tasks as mentioned in section 2.3. Factor n_{metric} represents the number of metric-specific elements within a task, e.g. the number of function points. The number of metric-specific elements multiplied by the metric-element-specific computation time results in an estimation of task execution time. The estimated computation time multiplied by the ECU-specific power consumption P_{HW} per ECU results in the energy consumption per ECU. The energy required for communication is then added to the aggregate ECU energy consumption. This energy model results in more accurate energy estimates than using the above equations.

$$E_3 = \sum_{i=1}^{\substack{Number\\of\ ECUs}} \left[\underbrace{\left(\sum_{j=1}^{\substack{Number\\of\ Tasks}} n_{metric,i,j} \cdot t_{metric,j} \right)}_{\substack{Software\ Execution\\Time\ Estimation}} \cdot P_{HW,i} \right] + E_{com} \quad (3)$$

E_3 represents the energy estimate of the entire system within the system design phase. The accuracy of this equation is based primarily on the accuracy of the task execution time estimation metric and the hardware power consumption estimate.

3.4 Hardware/Software Development

Equations 4-7 depict a model for estimating the energy consumption of a system configuration as outlined in (Heinrich and Prehofer, 2012). The energy consumption for a specific configuration is modeled by adding together the energy consumed during task executions (E_{task}), sleep periods (E_{sleeps}) including the associated mode changes n_s and communications (E_{com}). Energy consumption is calculated using a common multiplier t_{mult} of all task periods, which also determines the number of sleep states n_s.

Apart from the ECU power consumption P_{HW} at a specific CPU frequency f_{CPU}, the equation also factors in the energy required to communicate via a network E_c per bit (including network overhead), as well as the energy consumed by components during sleep states.

The tasks and the communication between the tasks have specific parameters. The assumption is that the task execution time at a specific frequency can be converted to the task execution time t_e at another frequency f_{te}. The cycle time is specified by t_c and the necessary communication per cycle between tasks by c_t. The sum of the single task execution times yields the task active time t_{tasks} per ECU.

$$E_4 = \sum_{i=0}^{\substack{No.of\\ECUs}} (E_{tasks,i} + E_{sleeps,i}) + E_{com} \qquad (4)$$

With

$$E_{tasks} = t_{tasks} \cdot P_{HW} = \left(\sum_{i=0}^{\substack{Tasks\ at\\Hardware}} \frac{t_{e,i} \cdot f_{te,i}}{t_{c,i}} \right) \frac{t_{mult}}{f_{CPU}} \cdot P_{HW} \qquad (5)$$

and

$$E_{sleeps} = ((t_{mult} - t_{tasks} - n_s \cdot t_{cs}) \cdot P_s) + n_s \cdot E_{cs} \qquad (6)$$

and

$$E_{com} = \sum_{l=0}^{\substack{Number\ of\ network\\communication}} (c_{t,l} \cdot E_{c,l}) \qquad (7)$$

The accuracy of these equations depends on the accuracy of the individual parameters. Some parameters are still difficult to specify after system integration because of cross-influences or because the parameters are difficult to measure. A good example is the energy required to execute one task on a complex ECU with many parallel tasks scheduled by an operating system.

4 RELATED RESEARCH

The method described in (Donnelly et al., 2006) includes energy estimates during the development process. The concept relies on a holistic approach to developing energy-efficient products by using checklists to ensure that specific energy aspects are factored in. Our approach differs in that it focuses specifically on the impact of energy consumption in networked embedded systems. The paper referenced in (Seo et al., 2009) discusses the early-stage estimation of energy consumption for communication tasks based on the architecture of the system. In this case however, the assumption is that the energy demands are the same for each device, which is not the case in our system. The tool referenced in (Tensilica Inc, http://www.tensilica.com) enables early-stage energy estimates for system-on-a-chip designs to optimize processor and memory. This approach requires software source code however. As part of our paper "Network-Wide Energy Optimization for Adaptive Embedded Systems" (Heinrich and Prehofer, 2012), we previously presented models for estimating energy consumption at the network level. The publication "Energy Consumption Estimation in Embedded Systems" (Konstantakos et al., 2008) furthermore examines ECU-wide energy estimation. While both of the latter two approaches are carried out during the system design phase, they do not allow for estimating energy consumption during the earlier development phases.

Some of the research activity in this area focuses on the hardware/software development or system runtime phases. One approach to simplifying the implementation of energy saving techniques during hardware/software development is outlined in (Apel et al., 2010). This relies on a modular approach in which energy-saving functions can be selected and implemented separately. The technique described in (Shankaran et al., 2009) uses adaptive planning and resource management to adapt the system to varying workloads, but without focusing on energy efficiency.

Most energy saving technologies optimize the energy consumption of one component (i.e. CPU) after the development phase (at runtime). Examples include dynamic hardware resource management and dynamic voltage/frequency scaling. Other methods focus on resource optimization, such as energy-efficient task scheduling (Kang et al., 2002); (Hu and R. Marculescu, 2004) or trade-offs such as reducing quality of service to lower energy consumption (Rusu et al., 2003); (Baker, 2011). A lot of research concentrates on the reduction of system-wide energy consumption, which in this paper equates to component or ECU-wide. Another method outlined in (Benini et al., 2000) uses dynamic power management (DPM), which deactivates idle system components to save energy. The approach described in (Jejurikar and Gupta, 2004)uses dynamic voltage scaling and DPM to reduce energy consumption across all components. It also factors in the energy consumption of peripherals, such as memory, in standby and processor activation/deactivation periods. The software framework described in (Zeng et al., 2008)uses various energy-saving techniques and looks for corresponding trade-offs. While the objective is to reduce component-wide energy consumption for different applications during runtime, this approach neglects the energy required for reconfiguration.

None of these approaches use the possibilities of optimizing energy consumption at the network level. The approach discussed in (Heinrich and Prehofer, 2012) looks at the potential for network-wide energy optimization by analyzing an embedded system via a case-study. Network-wide optimization within wireless sensor networks has also been evaluated, but with a focus on energy-efficient message routing.

5 CONCLUSIONS

This paper outlines a phase-by-phase process for estimating energy consumption in networked embedded systems for typical development processes. The goal is to integrate energy estimation in the design process and to give developers the possibilities to evaluate how design variants impact energy consumption. To make this possible, energy estimation must be enabled in the early phases of design since every decision further limits the ability to make design changes during later phases.

In this paper we created formalized energy consumption estimates using various existing techniques and the information that is available at each stage of the process. The models we developed for networked embedded system designs, such as for automotive systems, include the effort required for each task, sleep states and network communication.

Our research focused on the development of a series of models that factor in the information acquired during each development phase and refine the model of the prior phase. To date, other research work has not considered a systematic analysis and examination of energy consumption during each development phase. Future research will include validation of the models using measurements during system design.

REFERENCES

Arthur D. Little, *Market and Technology Study Automotive Power Electronics 2015*. Available: http://www.adlittle.com/downloads/tx_adlreports/ADL_Study_Power_Electronics_2015.pdf.

A. Monetti, T. Otter, and N. Ulshöfer, "Spritverbrauch senken, Reichweite erhöhen: System-Basis-Chip für den Teilnetzbetrieb am CAN-Bus," *Elektronik Automotive*, no. 11, pp. 24–27, 2011.

J. Weber, *Automotive Development Processes: Processes for Successful Customer Oriented Vehicle Development*. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2009.

J. E. Ross and S. Perry, *Total quality management: Text, cases and readings,* 3rd ed. Boca Raton, Fla: St. Lucie Press, 1999.

AUTOSAR, *Automotive Open System Architecture - Homepage.* Available: www.autosar.org.

B. W. Boehm, *Software engineering economics*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

V-Modell XT: Part 1: Fundamentals of the V-Modell. Available: http://ftp.tu-clausthal.de/pub/institute/informatik/v-modell-xt/Releases/1.3/V-Modell XT HTML English (2012, Aug. 06).

Freescale's portfolio of automotive microcontrollers. Available: www.freescale.com/webapp/sps/site/homepage.jsp?code=IFATOATMTV

The Embedded Microprocessor Benchmark Consortium, *AutoBench 1.1: Software Benchmark Data Book.* Available: http://www.eembc.org/techlit/datasheets/autobench_db.pdf.

A. J. Albrecht, "Measuring Application Development Productivity," Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium, Monterey, California, October 14–17, IBM Corporation (1979), pp. 83–92.

T. McCabe, "A Complexity Measure," *IIEEE Trans. Software Eng,* vol. 2, no. 4, pp. 308–320, 1976.

P. Heinrich and C. Prehofer, "Network-Wide Energy Optimization for Adaptive Embedded Systems," in *Proceedings of the 4th Workshop on Adaptive and Reconfigurable Embedded Systems (APRES 2012)*, 2012, pp. 24–27.

K. Donnelly, Z. Beckett-Furnell, S. Traeger, T. Okrasinski, and S. Holman, "Eco-design implemented through a product-based environmental management system," *Journal of Cleaner Production*, vol. 14, no. 15-16, pp. 1357–1367, 2006.

C. Seo, G. Edwards, D. Popescu, S. Malek, and N. Medvidovic, "A framework for estimating the energy consumption induced by a distributed system's architectural style," in *Proceedings of the 8th international workshop on Specification and verification of component-based systems - SAVCBS '09*: ACM Press, 2009, p. 27.

Tensilica, Inc, *Optimizing for Energy Using the Xenergy Energy Estimator Tool: Application Note.* Available: http://www.tensilica.com/uploads/pdf/XenergyEstimation.pdf.

V. Konstantakos, A. Chatzigeorgiou, S. Nikolaidis, and T. Laopoulos, "Energy Consumption Estimation in Embedded Systems," *IEEE Trans. Instrum. Meas,* vol. 57, no. 4, pp. 797–804, 2008.

S. Apel, D. Batory, K. Czarnecki, F. Heidenreich, C. Kästner, O. Nierstrasz, N. Siegmund, and M. Rosenmüller, "Automating energy optimization with features," in *Proceedings of the 2nd International Workshop on Feature-Oriented Software Development - FOSD '10*: ACM Press, 2010, pp. 2–9.

N. Shankaran, J. S. Kinnebrew, X. D. Koutsoukas, C. Lu, D. C. Schmidt, and G. Biswas, "An Integrated Planning and Adaptive Resource Management Architecture for Distributed Real-Time Embedded Systems," *IEEE Trans. Comput,* vol. 58, no. 11, pp. 1485–1499, 2009.

Dong-In Kang, S. Crago, and Jinwoo Suh, "A fast resource synthesis technique for energy-efficient real-time systems," in *Proceedings of the 23rd IEEE Real-Time Systems Symposium RTSS 2002*, IEEE, Ed, 2002.

Jingcao Hu and R. Marculescu, "Energy-aware communication and task scheduling for network-on-chip architectures under real-time constraints," in

Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'04), IEEE, Ed, 2004, pp. 234–239.

C. A. Rusu, R. Melhem, and D. Mosse, "Maximizing the system value while satisfying time and energy constraints," *IBM J. Res. & Dev,* vol. 47, no. 5, pp. 689–702, 2003.

M. Baker, *Topics in power and performance optimization of embedded systems*, Dissertation, Arizona State University, 2011, Available: http://hdl.handle.net/2286/jq8f23vj0yh

L. Benini, A. Bogliolo, and G. de Micheli, "A survey of design techniques for system-level dynamic power management," in *Transactions on Very Large Scale Integration (VLSI) Systems*, IEEE, Ed, 2000.

R. Jejurikar and R. Gupta, "Dynamic voltage scaling for systemwide energy minimization in real-time embedded systems," in *Proceedings of the International Symposium on Low Power Electronics and Design (ISLPED'04)*, 2004, p. 78.

G. Zeng, H. Tomiyama, H. Takada, and T. Ishihara, "A Generalized Framework for System-Wide Energy Savings in Hard Real-Time Embedded Systems," in *Proceedings of the 5th International Conference on Embedded and Ubiquitous Computing EUC 2008*, 2008, pp. 206–213