ENERGY-AWARE DESIGN OF HARDWARE AND SOFTWARE FOR ULTRA-LOW-POWER SYSTEMS

Dissertation

zur Erlangung des Grades eines

DOKTORS DER INGENIEURWISSENSCHAFTEN

der Technischen Universität Dortmund
an der Fakultät für Informatik

von

MARKUS BUSCHHOFF

Dortmund
2019

Tag der mündlichen Prüfung:	16.09.2019
Dekan:	Prof. Dr.-Ing. Gernot A. Fink
Gutachter:	Prof. Dr.-Ing. Olaf Spinczyk
	Prof. Dr. Peter Marwedel

ABSTRACT

Future visions of the *Internet of Things* and *Industry 4.0* demand for large scale deployments of mobile devices while removing the numerous disadvantages of using batteries: degradation, scale, weight, pollution, and costs. However, this requires computing platforms with extremely low energy consumptions, and thus employ ultra-low-power hardware, energy harvesting solutions, and highly efficient power-management hardware and software. The goal of these power management solutions is to either achieve power neutrality, a condition where energy harvest and energy consumption equalize while maximizing the service quality, or to enhance power efficiency for conserving energy reserves. To reach these goals, intelligent power-management decisions are needed that utilize precise energy data. This thesis discusses the measurement of energy in embedded systems, both online and by external equipment, and the utilization of the acquired data for modeling the power consumption states of each involved hardware component. Furthermore, a method is shown to use the resulting models by instrumenting preexisting device drivers. These drivers enable new functionalities, such as online energy accounting and energy application interfaces, and facilitate intelligent power management decisions. In order to reduce additional efforts for device driver reimplementation and the violation of the *separation of concerns paradigm*, the approach shown in this thesis synthesizes *instrumentation aspects* for an *aspect oriented programming* language, so that the original device-driver source code remains unaffected. Eventually, an automated process of energy measurement and data analysis is presented. This process is able to yield precise energy models with low manual effort. In combination with the instrumentation synthesis of aspect code, this method enables an accelerated creation process for energy models of ultra-low-power systems. For all proposed methods, empirical accuracy and overhead measurements are presented. To support the claims of the author, first practical energy aware and wireless-radio networked applications are showcased: An energy-neutral light sensor, a photovoltaic-powered seminar-room door plate, and a sensor network experiment testbed for research and education.

PUBLICATIONS

Some ideas and figures have appeared previously in the following publications:

[17] M. Buschhoff, C. Günter, and O. Spinczyk. "MIMOSA, a Highly Sensitive and Accurate Power Measurement Technique for Low-Power Systems." In: *Real-World Wireless Sensor Networks*. Ed. by K. Langendoen, W. Hu, F. Ferrari, M. Zimmerling, and L. Mottola. Vol. 281. Lecture Notes in Electrical Engineering. Springer International Publishing, 2014, pp. 139–151. ISBN: 978-3-319-03070-8. DOI: 10.1007/978-3-319-03071-5_16.

[18] M. Buschhoff, R. Falkenberg, and O. Spinczyk. "Energy-Aware Device Drivers for Embedded Operating Systems." In: *SIGBED Rev.* (2019). to appear.

[19] M. Buschhoff, D. Friesel, and O. Spinczyk. "Energy Models in the Loop." In: *Procedia Comput. Sci.* 130.C (May 2018), pp. 1063–1068. ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.04.154. URL: https://doi.org/10.1016/j.procs.2018.04.154.

[20] M. Buschhoff, C. Günter, and O. Spinczyk. "A unified approach for online and offline estimation of sensor platform energy consumption." In: *Proceedings of the 8th International Wireless Communications and Mobile Computing Conference (IWCMC '12)*. Limassol, Cyprus, Aug. 2012, pp. 1154–1158. DOI: 10.1109/IWCMC.2012.6314369.

[21] M. Buschhoff, J. Streicher, B. Dusza, C. Wietfeld, and O. Spinczyk. "MobiSIM: A Simulation Library for Resource Prediction of Smartphones and Wireless Sensor Networks." In: *Proceedings of the 46th Annual Simulation Symposium*. ANSS '13. San Diego, California: Society for Computer Simulation International, 2013.

[28] R. Falkenberg, M. Masoudinejad, M. Buschhoff, A. K. Ramachandran Venkatapathy, D. Friesel, M. ten Hompel, O. Spinczyk, and C. Wietfeld. "PhyNetLab: An IoT-Based Warehouse Testbed." In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*. Sept. 2017.

[30] D. Friesel, M. Buschhoff, and O. Spinczyk. "Annotations in Operating Systems with Custom AspectC++ Attributes." In: *Proceedings of the 9th Workshop on Programming Languages and Operating Systems (PLOS '17)*. PLOS'17. Shanghai, China: ACM, 2017, pp. 36–42. ISBN: 978-1-4503-5153-9. DOI: 10.1145/3144555.3144561. URL: http://doi.acm.org/10.1145/3144555.3144561.

[31] D. Friesel, M. Buschhoff, and O. Spinczyk. "Parameter-Aware Energy Models for Embedded-System Peripherals." In: *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*. June 2018, pp. 1–4. DOI: 10.1109/SIES.2018.8442096. URL: https://ieeexplore.ieee.org/document/8442096.

[52] M. Masoudinejad, A. K. Ramachandran Venkatapathy, D. Tondorf, D. Heinrich, R. Falkenberg, and M. Buschhoff. "Machine Learning Based Indoor Localisation Using Environmental Data in PhyNetLab Warehouse." In: *Proceedings of the European Conference on Smart Objects, Systems and Technologies 2018*. Smart Systech 2018. Dresden, Germany, 2018.

NON PEER-REVIEWED PUBLICATIONS

[40] S. Kerrison, M. Buschhoff, J. Nunez-Yanez, and K. Eder. "Measuring Energy." In: *ICT - Energy Concepts for Energy Efficiency and Sustainability*. Ed. by G. Fagas, L. Gammaitoni, J. P. Gallagher, and D. J. Paul. Rijeka: InTech, 2017. Chap. 03. DOI: 10.5772/65989. URL: http://dx.doi.org/10.5772/65989.

ACKNOWLEDGMENTS

Der größte Dank gebührt meiner Frau Petra, die seit 20 Jahren nicht nur jede meiner verrückten Ideen erträgt, sondern diese auch tatenreich unterstützt.

I want to thank Prof. Dr.-Ing. Olaf Spinczyk for the opportunity to work in his group, and for his good advice throughout all of my research. He proved himself as a rich source of profound knowledge, and always had an open door for productive scientific discussions. He strongly influenced all ideas presented here.

Next, I want to thank the Promotional Committee, Prof. Dr. Heinrich Müller, Prof. Dr.-Ing. Olaf Spinczyk, Prof. Dr. Peter Marwedel, and Prof. Dr. Jens Teubner, for their time and expertise spent on this thesis.

There are plenty more people who I feel the urge to thank: My friend and office mate Bogusław Jabłkowski for all of his support, Georg v.d. Brüggen, Anas Toma, and Prof. Jian-Jia Chen for the many discussions (and laughters) during lunch, and all of my research partners within the Collaborative Research Center SFB-876, among them Björn Dusza, Jan Emmerich, Robert Falkenberg, Christoph Ide, Alexander Lochmann, Mojtaba MassoudiNejad, Alexander Munteanu, Aswin Ramachandran, Moritz Roidl, and Jochen Streicher. Further thanks belongs to all colleagues at Computer Science XII, TU-Dortmund – they are too many to list them all.

There is a huge list of students and student assistants who supported my work, most mentionable Christian Günter, Jan Böcker, Daniel Friesel, all students from the Solar Doorplate project group, and many more.

The work presented in this thesis was partly supported by the German Research Council (DFG) within the Collaborative Research Center SFB 876, project A4.

Last but not least, I want to thank my friends and family for their steady support and motivation.

CONTENTS

1 INTRODUCTION

Nowadays, visions of the *Internet of Things* (IoT) portray a world that is pervaded by computational systems, so that even ordinary items have the power to autonomously manage their status and to communicate over a wireless network. However, batteries are currently the only viable solution for supplying power to mobile computational devices, thus limiting the range of possible system applications, especially in mass deployments, like envisioned in several Industry 4.0 scenarios. System software that utilizes precise energy models to become aware of the electrical energy[1] production and consumption of device components enables efficient power management solutions. This results in decreased recharge cycles of existing battery powered solutions. In combination with energy harvesting technologies, it enables new battery free applications. The next chapters of the following thesis will scientifically substantiate this claim.

Thesis statement

1.1 MOTIVATION AND RELEVANCE

> “Programmers will not be able to afford to be ignorant about the energy cost of the programs they write … You need tools that give you feedback and tell you how good your decisions are.”
>
> *S. Furber, principal designer ARM microprocessor [1]*

The purposes for the IoT vary widely, from Industry 4.0 applications to the consumer market. The vision behind the term *Industry 4.0* is to decouple production and logistic processes by using autonomous machines, conveyors and containers. In the consumer market, the focus rests on usability and convenience for the user, with a product lineup ranging from smart refrigerators that automatically replenish consumed food, to completely automated homes (*“smart homes”*) that optimize comfort and energy usage for residents. While IoT devices for cable-powered

1 If not mentioned otherwise, the term *energy* always refers to *electrical* energy in this thesis. Unlike energy in general, electrical energy can be produced and consumed.

environments are already available in the consumer market, the design of cordless systems is still challenging. Currently, there are only two technologies predominant on the market for IoT and Industry 4.0 systems:

Market-ready power supply technologies for Industry 4.0 and the IoT

Battery powered devices with considerable computational or communicational abilities, and *induction powered devices*, like smart cards and *radio-frequency identification* (RFID) tags that only work in a well defined environment and offer limited computational abilities.

Either of these power supply approaches has its downsides: Battery powered devices need frequent maintenance to recharge or replace the batteries. While this might seem inconvenient, yet still acceptable, for consumer devices like smart phones, it is unreasonable for some Industry 4.0 concepts, where several thousand to several millions of goods containers equipped with battery driven technology would require an untenable amount of maintenance. [51] Additionally, current lithium-based battery technologies suffer from a limited lifetime of two to three years. Furthermore, they increase weight and size, pose a fire hazard, and contribute to environmental pollution.

On the other hand, induction powered devices only work in close proximity to the source of an electromagnetic field. While this enables quick and cordless connections to a power source, it does not significantly increase mobility.

Energy harvesting

However, there is a plethora of matured technologies for harvesting energy from the environment, e.g., photovoltaics and kinetic harvesters. But, due to their low and transient power generation, they only find use in large-scale installations within highly exposed environments. Thus, besides the photovoltaic-powered pocket calculators of the 1980s, these technologies did not find their way into many designs of small-scale embedded systems until recently.

Example Applications

Still, energy harvesting technologies, combined with efficient power management software components, enable a broad variety of applications, for example:

GOODS CONTAINERS The aforementioned scenario of the mass deployment of embedded systems within goods containers (bins, pallets) enables monitoring and controlling of logistic processes.

MACHINE MONITORING Sensor systems that monitor external parameters of machines, like noise, vibration, radia-

tion, temperature, preferably powered by harvesting energy from these emissions.

INFORMATION DISPLAY Frequently updated display systems that are easy to install, or even mobile, like door plates for seminar rooms, or shelf labels in stores.

Furthermore, existing battery powered applications can be supported by methods and technologies of transiently powered devices. As an example, a smart watch can benefit from harvesting light, kinetic energy and temperature gradients [49], as well as from energy-aware system software. It is remarkable that in 2019, smart watches still require daily battery recharges, while the analogous automatic watch, harvesting energy from the arm movement of their owner, was invented in 1914.

To achieve energy efficient system designs, all hard- and software components need to be optimized. Manufactures for embedded system hardware now have a strong focus on energy efficiency. For example, TI has developed the *ultra-low power* (ULP) MSP430 *microcontroller unit* (MCU) family [75], with a plethora of concomitant components like radio transceivers and power supply solutions, and ARM offers the *Cortex-M* family as a low-power solution [5].

Premises of power-management are not fulfilled for ultra low power systems.

On the software side, power-management solutions need to become more efficient. For systems scaling up from smart phones, there is an implicit premise to power management: The power consumption of power-management algorithms and online power measurements is negligible, because the periphery and the actual software applications consume a great quantity of the available power. Thus, any reduction of periphery or application *central processing unit* (CPU) utilization quickly yields positive effects on overall energy consumption.

Yet, with ULP devices, this might not be true. It is easy to create situations where the improvident use of power management algorithms increases the overall power consumption in this system class. For example, when a CPU requires more power than a periphery component, the power-management algorithms for that said component can have a malicious impact. This is shown in more detail in Chapter 6.

Economic considerations are essential for mass-deployment of IoT devices

Apart from technical aspects, the costs for hardware components and system design must become part of research intentions, as they highly influence the acceptance of technical solutions, especially because mass deployment is an important property of many IoT applications. An important element for

reducing design expenses is the availability of ready-to-use system software, as it reduces the efforts of individual energy modeling, accounting and power management. For this reason, there is a focus on operating system integration of power management methods for ULP systems in this thesis.

As stated before, another important aspect is the production costs of hardware components. This requires simple solutions to complex problems like power measurement and adaptation to transient energy income. This special case will be addressed in Chapter 5, showing the economic impact of re-thinking conventional power-management solutions.

Requirement of interdisciplinary expertises

In conclusion, it is challenging to design an IoT system that uses radio technology, has reasonable computational power, predictable reliability, is power efficient and inexpensive, as this spreads among several disciplines of science. Understanding and optimizing energy-harvesting technologies and energy buffers (capacitors, batteries) requires an expertise in this special field of electronic engineering. Creating models for energy production and consumption that can be calculated on ULP processors requires special knowledge about power measurement next to an expertise in computer science. Optimizing power consumption, e.g. of radio communications, requires deep knowledge about radio technologies, network protocols and hardware/software interactions. Although there have been a plethora of approaches for each topic in one of the disciplines, a multi-disciplinary view and a compound understanding is still required to combine, extend and interweave them. This thesis is an endeavor to combine methods of electrical engineering with computer science and software engineering for optimizing energy consumption of ULP devices at the hardware/software interface.

1.2 RESEARCH QUESTIONS

Interdisciplinary approaches required

Existing approaches to energy measurement, energy modeling, and software implementation of energy models need to be integrated in a system platform concept to enable efficient power-management solutions. Yet, state-of-the-art research is barely interdisciplinary, i.e., approaches in energy measurement technologies do not take the special demands of measurements in the field of ULP energy modeling into account, namely the requirement for software-synchronized measurements with acceptable time resolution and accuracy. Research in the field of

energy models does not consider the acquisition of measurement values, thus making these approaches difficult to apply in automated measurement environments. Furthermore, energy models are typically constructed to be highly precise by respecting complex system interactions, but rarely suit the requirement for being applicable as low-overhead online models in an ULP (i.e., 8/16 bit) CPU environment. This leads to the following research questions:

QUESTION BLOCK 1: What are reasonable methods to measure and model energy consumptions of components of ULP embedded systems? What is the possible accuracy of such models? Can they be applied to operating system software to account energy consumption, and what is the energetic overhead of this?

QUESTION BLOCK 2: What are efficient and inexpensive online-measurement methods for transiently powered systems? How can they be used to adapt the system load to energy income? Does this have a positive impact on the system's up time?

QUESTION BLOCK 3: Can energy measurement and modeling be automated to become easily applicable for application developers? Can energy aware software be synthesized from these models?

1.3 SCIENTIFIC CONTRIBUTIONS

This thesis provides contributions to answer the research questions of the previous section.

CONTRIBUTION 1: An energy measurement approach for measuring the relevant parameters of ULP embedded systems. This approach includes the development of specialized measurement equipment.

CONTRIBUTION 2: A low-cost online measurement approach for photovoltaic powered ULP systems, allowing the adaptation of the system's workload to energy income.

CONTRIBUTION 3: A finite-state machine driven approach to modeling the energy consumption of typical ULP system components, and a method to utilize the model in the drivers of a system, enabling precise energy accounting at low overhead.

CONTRIBUTION 4: The combination of energy measurement, modeling and automated software generation to enable automated energy measurement of a hardware platform, automated cost-annotation of the state-machine models and the automated synthesis of energy-accounting hardware drivers.

1.4 OUTLINE OF THE THESIS

This thesis combines and researches approaches of different fields. After the definition of basic terms and a presentation of the state of the art, this thesis exhibits a construction flow for energy models and their implementation in ULP systems. Starting from energy measurement technologies, measured values are used to form models for CPU and peripheral devices. Then, these models are used to create implementations that allow efficient online accounting of energy consumptions. In a last step, the whole measurement, modeling and implementation process is integrated into an automated modeling and synthesis concept.

CHAPTER 2: This chapter introduces basic concepts of energy and the corresponding terms in the fields of physics and electronic engineering, and is intended to enable readers with a background in computer science to understand the used terms of the interdisciplinary approaches.

CHAPTER 3: This chapter presents the state of the art in the relevant fields of research.

CHAPTER 4: This chapter introduces energy measurement terminologies, technologies and important aspects of electrical engineering in this field. It showcases an own approach: *MIMOSA*, a device to measure the energy consumption of ULP embedded system components in their respective states.

CHAPTER 5: This chapter summarizes challenges in the field of in-situ energy measurements for photovoltaic powered ULP devices, especially under the aspect of economic consequences in mass production. A new concept to measurement is shown that enables a high degree of adaptivity to energy income while showing additional prototype costs below 10 eurocents. This is achieved by using parasitic effects of typical MCUs.

CHAPTER 6: Here, a finite-state-machine modeling concept is presented that enables energy accounting for hardware components. The concept is then implemented into the driver layer of an embedded operating system to show the feasible accuracy and overhead. A concept of *coherence* between driver implementation and energy model is shown, that allows to automate the process of *driver instrumentation*, i.e., the deployment of energy accounting code within the existing drivers of embedded system software.

CHAPTER 7: This chapter combines approaches of the previous chapters to an automated measurement and driver synthesis concept. Here, a structural (not cost-annotated) finite-state-machine model is automatically converted into system software for measuring the cost factors of the model. The results are analyzed and re-instantiated to refine or validate the model. The final result will then be used to generate energy-accounting drivers.

CHAPTER 8: This chapter shows the practical application of shown concepts in two projects. Both projects use a minimal, monolithic operating system named *Kratos* that I developed as testbed for this thesis. First, the *Solar Doorplate* project is portrayed, a door tag for office and seminar rooms that is completely driven by indoor photovoltaics. The Solar Doorplates are able to communicate their energetic status via radio communication, making it possible to schedule remote updates of their display. Second, the PhyNetLab project is shown, a network of ULP nodes constructed at the Fraunhofer IML institute in Dortmund. The PhyNetLab is used to evaluate concepts of ULP engineering in logistics.

CHAPTER 9: This chapter discusses the results of the various research directions shown before as a compound approach to ULP hardware and software engineering.

1.5 CONTRIBUTIONS OBTAINED IN COOPERATION

According to §10(2) of the examination regulations of the department of computer science, TU Dortmund, 2011, a doctoral dissertation must include a separate list of the author's contribution to the scientific results that were obtained in cooperation with other persons. As such, the following list provides a

chapter-wise overview of my contributions to results that were obtained in cooperation.

For all of my work, *Olaf Spinczyk* was supervisor and discussion partner, and provided ideas for several details of the shown concepts, implementations and evaluations.

CHAPTER 4: The presented MIMOSA measurement platform was a collaborative work by Christian Günter and me. I contributed the fundamental measurement concepts and the respective parts of the circuit schematics, Christian Günter created the first prototype and solved many detail problems of the circuitry.

CHAPTER 5: This formerly unpublished work was completely developed and evaluated by me. My schematics and results were presented to Sebastian Drywa as introductory material for his master thesis under my supervision in 2018. By mistake, Sebastian Drywa omitted to denote the provided material as an external contribution in his thesis. This circumstance is documented in the official assessment to his master thesis.

CHAPTER 6: I developed the concept of finite-state-machine based energy accounting within the drivers of an embedded operating system. The implementation and evaluation of the concept was a result of the two master theses (under my supervision) of Robert Falkenberg and Daniel Friesel.

CHAPTER 7: I created the concept of using a measurement loop with automated evaluation and driver generation. In his master thesis, Daniel Friesel implemented and evaluated this concept and added further analysis methods for function parameter dependencies. Although the general idea and demand to use machine learning for this purpose was by me, most of the conceptual work of the parameter analysis was contributed by Daniel Friesel.

CHAPTER 8: Kratos is based on an open-source operating-system implementation called OO-StuBS. Most of the OO-StuBS code was implemented by Olaf and Ute Spinczyk. I contributed the Kratos specific conceptual work and parts of the implementation. Many students contributed ideas and code under my supervision during

their thesis works, project groups or assistant employments. The Solar Doorplate project was initiated as a project for students by Olaf Spinczyk, Alexander Lochmann and me, and twelve students (Jan Böcker, Timo Cramer, Daniel Friesel, Sebastian Lukas Hauer, Michael Müller, Todor Nikolov, Benedikt Ruthenberg, Daniel Smit, Merlin Stampa, Martin Strzelczyk, David Tondorf, Franz-Bernhard Tuneke) worked on it for one year under our supervision. The PhyNetLab hardware platform was completely created at the Chair of "Förder und Lagerwesen (FLW)",TU-Dortmund, and the Fraunhofer IML institute. I supported the development of hardware and software with energy measurements, software components and test results.

2 ENERGY IN ULTRA LOW POWER SYSTEMS

This chapter will briefly discuss energy relevant components of mobile computational systems. A more comprehensive introduction into the basic concepts of electrical energy and energy measurement will be found in Chapter 4.

2.1 BASIC TERMS

An electrical system converts energy of other forms, e.g., chemical bound energy in batteries, or mechanical energy using generators, into *potential energy* that is able to transfer electrical charges. In the following, this step will be denoted as *energy production*.

Charge transfers can convert electrical energy into another form, e.g., heat, light or other emissions. This step will be denoted as *energy consumption* or *energy storage*, depending on its purpose.

Thus, an electrical system can produce, transfer, store, and consume electrical energy.

Power is a measure for the rate at which energy is transferred. In consequence, energy is power integrated over time. Electrical power is the product of the electric potential, measured in volts, and the resulting charge current, measured in amperes. Formal definitions of these terms can be found in Chapter 4.

2.2 SOURCES OF ELECTRICAL ENERGY

Mobile computing systems require special power supply components. This section will briefly discuss both storage and energy harvesting technologies for embedded systems.

2.2.1 *Batteries*

Batteries are currently the most important power source for mobile systems, as they offer high energy densities while usually being safe to handle. They use chemically bound energy to generate a mostly constant voltage. Batteries are segregated into

primary cells, secondary cells

two categories: *Primary cells*, which are single-use devices, and *secondary cells*, which are rechargeable and thus can revert the chemical reaction of the discharge process when a sufficient source of electrical energy is connected.

Primary cells are alkaline or lithium-based devices and offer standardized form factors. They show low leakage currents and thus can conserve their charge over long times. This allows them to be stocked for several months and makes them an ideal solution for infrequently used items. Their recharging potential is limited to about ten cycles, but only if treated properly and by using special constant-current recharging stations. They contribute considerably to environmental pollution.

long storage, standard form factors, limited re-use

Secondary cells, currently mostly based on lithium-ion (Li-Ion) technology, are the most common type of batteries and are used in systems with a high and continuous energy demand, like smart phones and mobile computers. Depending on the battery construction and materials, charge behavior and storage temperature, Li-Ion batteries allow for several thousand recharge cycles[1], but undergo a continuous aging process, limiting their lifetime to 2-5 years[2].

rechargeable, limited lifetime

Another battery technology are *rechargeable alkali manganese* (RAM) cells. These are available in standard form factors and based on alkaline cells, allowing up to 100 recharge cycles [42] while still showing low leakage currents. Their purpose is to replace primary alkaline cells to lower the environmental impact.

RAM cells as a compromise

Besides offering several advantages, like constant voltage, high energy density, and ease-of-use, batteries expose several disadvantages:

- Li-Ion batteries have a limited lifetime of 2-5 years.
- Missing standardization and hard-wiring limits replacement options for many secondary cells.
- Due to aging processes (partially influenced by environmental factors), the charge state is hard to determine and the time of full discharge is hard to predict.
- For large scale sensor network deployments of several hundreds or thousands of nodes, e.g., in *Industry 4.0* sce-

1 e.g., 4711 recharge cycles at 20% depth of discharge, C/2 charge rate, and 25 °C [79]

2 e.g., 80 % discharge resistance increase after 400 days storage at 25 °C and 50 % charge [37]

narios, replacement and material costs can become critical [3].

- Environmental conditions (temperature and humidity) are highly constrained.
- Batteries greatly contribute to system size and weight, often being the heaviest component of a system.
- Lithium-based batteries expose danger of fire after physical damage.
- Battery leaks release toxic and corrosive materials.
- Battery production and disposal contribute to environmental pollution.

2.2.2 *Super Capacitors*

Super capacitors are a subsequent development of electrical capacitors, which store electrical energy in a self-preserving electrostatic field. Super capacitors increase charge density by using electro-chemical reactions in combination with advanced double-layer capacitor technologies. They achieve significantly higher capacitances than conventional electrolyte capacitors, but still only yield about 5-15% of the energy density of batteries [67]. Their voltage range is usually limited to below 4 V.

super capacitors complement batteries

Super capacitors complement the advantages and disadvantages of batteries: They show reduced aging effects (20% capacitance reduction in 5-10 years [34]) and thus reduce replacement requirements, but their high leakage currents limit their energy storage abilities to several hours.

This makes super capacitors a viable choice for long time deployments with frequent recharge possibilities, especially in energy-harvesting supplied systems, where abundant energy needs to be buffered for a limited amount of time.

2.2.3 *Energy Harvesters*

Energy harvesters remove the requirement to store energy by producing electrical power directly from energy available in the environment. The most prominent technologies will be discussed in the following subsections.

2.2.3.1 *Electromagnetic Coils and Induction*

generators, microphones

Mechanical energy, including that of sound waves, can be harvested by electromagnetic coils. This principle is used in classical generators of any scale, from power plants to bicycle dynamos down to dynamic microphones. Generators induce current into a coil by transferring kinetic energy into the movement or rotation of a magnetic field. Alternatively, electro-magnetic fields of the environment, e.g., radio waves, can induce electrical current into a coil in the same way. As a prominent example, RFID tags are powered by energy induced from the peer station.

2.2.3.2 *Piezo Harversters*

Piezo materials are special ceramic and crystal materials that convert pressure or tension into electricity. Next to common applications in ignition systems, e.g., igniters in gas heaters or cigarette lighters, they can be used to power electronic systems by vibration, concussion and mechanical stress.

2.2.3.3 *Thermal Harvesters*

Temperature forms a potential energy, so the gradient space between two spatial points with different temperatures results in a thermal flow. This can be used to generate electrical current by using the Peltier-Seebeck effect: The junction of two conductive materials creates a temperature-dependent potential voltage. Thus, two connected junctions in different thermal conditions result in an electrical current between the junctions and a heat transfer at the same time.

2.2.3.4 *Photovoltaic Cells*

Photovoltaic cells (PVCs) convert light into electricity. They consist of a semiconductor material with doped p-n junctions. When a photon hits the semiconductor material, it can excite an electron and move it to the inductor band, thus creating a free charge carrier. Because of natural charge diffusion at the p-n junction and the resulting space charge zone, these free charge carriers can only move to one electrode, creating a static charge.

PVCs offer a nearly constant voltage (the material dependent voltage of the space charge zone, e.g., 0.68 V for silicon), un-

til the external charge transfer (load current) between the electrodes removes the space charge zone. That means, to maximize the power output of a PVC, the load current must be set to the point just before the space charge zone collapses. This point is considered the *maximum-power point* (MPP). PVCs are often produced in on-chip serial circuit setups to increase the output voltage to a desired value. The maximum output current is determined by the size of the p-n junction and the illumination. It can be further increased by parallel setups, which also increases the surface area exposed to light.

2.3 CONSUMERS OF ELECTRICAL ENERGY

A computational system consumes energy for various purposes. In the following, the most important power transfer mechanisms will be discussed.

2.3.1 *Computational Components*

This section will discuss computational components as assemblies of logic gates in the commonly used *metal-oxide semiconductor* (MOS) transistor technology. Energy is consumed by the transfer of electric charges between the transistors of the logical gates, which causes heat dissipation. The amount of dissipated power can be characterized from different perspectives: The gate level perspective observes power dissipation by characteristics of the used transistor technology. The instruction level perspective characterizes power consumption by instructions and processor operations. The mode level perspective characterizes power consumption by the system state, e.g., sleep modes and different computational tasks.

2.3.1.1 *Gate Level Observation*

The energy consumption of a computational unit, i.e. a CPU or *graphics-processing unit* (GPU), is a result of physical parameters of the used transistor technology. The power dissipation P_G for a logic gate is summed up in Equation 2.1.

$$P_G = P_d + P_l + P_{th} + P_s \tag{2.1}$$

P_d is the dynamic dissipation. It occurs when switching the transistors of the gate, and is dependent on C, the capacity of the transistors (a technology constant), the switching frequency

f, and the supply voltage V, as shown in Equation 2.2. Consequently, reducing the supply voltage V has the highest impact on reducing P_d. However, the switching duration of a transistor increases by reducing V, so that the operating frequency has to be reduced as well. Still, the transistor can be operated more efficiently at lower voltages. The energetic optimum of a MOS transistor in a digital circuit can be achieved by reducing f as far as acceptable for the purpose, and reducing the supply voltage V to the lowest feasible value for that frequency. Continuously adapting the supply voltage and operating frequency is called *dynamic voltage and frequency scaling* (DVFS) and is an important concept for power saving.

voltage/frequency scaling

$$P_d = CV^2 f \tag{2.2}$$

P_l in Equation 2.1 denotes the power dissipated by gate- oxide leakage currents. Leakage currents occur by charge carriers tunneling through the transistor gates and are technology dependent.

P_{th}, the *sub-threshold leakage* dissipation is a special case of leakage that is a result of using a transistor gate in a narrow voltage band below the threshold voltage. This region is especially important for analog circuits, but is to be avoided in digital applications. Due to the heavy size reduction of present technologies, the threshold voltage has become so low that the subthreshold band exceeds the lower supply voltage. That means, a digital LOW signal is already in the sub-threshold band for theses processors, forcing the transistor into inversion mode, with notable leakage currents.

P_s, the switching current power dissipation, occurs in logic gates with complementary transistors. When multiple transistors switch simultaneously, a transient current path between the power supply rails occurs, causing high power peaks.

The distribution of power dissipation between the various components of Equation 2.1 varies widely, with P_l usually being below 1%, and continuously loweing since the introduction of high-k gate dielectrics [2, 55]). The majority of dissipated power is shared between the others, depending on structural size, supply voltage and operating frequency. By reducing the structure size of transistors, P_{th} rises exponentially, and is one of the most important limiting factors for size and supply voltage reduction.

P_{th} limits downscaling of modern CPUs

While power observations at the transistor or gate level yield important results for choosing technologies, supply voltage and

operating frequencies, it is difficult to determine the accurate momentary power dissipation of a whole CPU at this level, because the status of all gates has to be considered. Simulations enable accurate results for discrete logic circuits, but cannot efficiently handle the complexity of modern processors.

2.3.1.2 *Instruction Level Observation*

Observing the power dissipated per CPU instruction opens the door for optimizing the energy consumption of algorithms. This can be achieved by ordering instructions and by avoiding power-intensive instructions where possible, e.g. by switching from floating point arithmetics to fixed point arithmetics.

The accurate determination of per-instruction power consumption is challenging, as it requires to model the effects of execution order, bus utilization and memory hierarchies [71]. Additionally, instruction arguments influence the energy consumption of an instruction [56].

However, the control flow within a system is usually not deterministic, as preemptive scheduling, hardware interrupts, and varying external data impede the prediction of instruction orders and the utilization of caches. This limits the use of instruction level energy models to either general predicates or to highly deterministic systems, e.g. highly critical hard real-time systems.

indeterministic control flow limits accuracy

2.3.1.3 *Mode Level Observation*

Processing units often have a set of operational modes, e.g. multiple sleep modes for energy management. In the special case of MCUs, there may be an extended set of modes, as they often allow to control the power supply of on-chip peripheral components as well.

To estimate processor power consumptions, a mode-based energy model can be used. Most often, especially for online power management purposes, this model simply uses an average power consumption value for each mode.

The accuracy of this method depends on the variance of the power consumption within a single mode. If the variance is high, e.g. because computational operations consume considerably more power than memory operations in a given architecture, the system becomes hard to predict. In this case it can be useful to extend the model of physical modes of the processor by virtual modes for different types of algorithms. E.g.,

physical modes describe the power state of a controller

virtual modes describe different algorithmic CPU loads

a floating-point computational mode can yield higher accuracies for processes with this respective computational load. This method implies knowledge about computational characteristics of different processes, tasks or functions.

Mode tracking can occur in hardware and software. Texas Instrument's EnergyTrace technology (as used in the MSP430 MCU series) is an example for hardware mode tracking: Here, the CPU sleep modes can be continuously tracked during debugging sessions. By software tracking, modes of CPUs and periphery can be tracked on the operating system level to drive power management decisions. This will be further discussed in Chapter 6.

2.3.2 *Peripheral Components*

The periphery of a CPU consists of a plethora of devices that suit various purposes and that consume energy by distinct mechanisms. A brief overview – without any claim to completeness – will be presented here.

2.3.2.1 *Displays*

Displays can be divided into two categories:

no energy for image display, high energy for updates

ghosting effect

STATIC DISPLAYS ("ePaper", "eInk") do not require energy to sustain an image, but require energy to change it. This type of displays uses mechanical mechanisms, usually controlled by electro-magnetic fields, to rotate or move pigmented particles. They perform well for the long-term display of information. However, the mechanical work to move the pigmented objects requires considerable energy. Current technologies show high update latencies, so that animated images cannot be displayed. Also, they suffer from magnetic imprinting effects ("ghosting"). To prevent this effect, multiple passes of image updates (including inverted images) are required to remove resident magnetic charges from previous images. This multiplies the energy demand for display updates and creates a visible and time consuming "flashing" sequence. To conserve energy and increase consumer acceptance, some applications, like e-Book readers, limit the anti-ghosting measures to every n-th display update, thus trading readability for energy savings and convenience.

DYNAMIC DISPLAYS require continuous power supply to sustain an image. These displays are combinations of light emitting, light reflecting and/or light occluding devices. Most common technologies are *liquid-crystal display* (LCD) which use liquid crystals and pole filters to occlude background light by polarization, and *light-emitting diode* (LED) displays that use a matrix of LEDs to directly create a luminescent image. Dynamic displays consume energy for background illumination (if required) and to power the LEDs or liquid crystals. Self-illuminating displays have the highest energy consumption, partly dependent on the amount of lit pixels. Reflective LCDs show the lowest energy consumption among dynamic displays.

continuous power supply required

2.3.2.2 *Sensors*

By their energy consumptions, sensors can be categorized into two classes: Active and passive sensors. Active sensors create emissions and sense the reflections. Typical examples are ultrasonic sensors used for distance measurements. Passive sensors deliver information about the environment without using emissions. Active sensors usually show a notably higher energy consumption.

active, passive

Depending on their physical parameters and purpose, sensors often have multiple modes or configurations for power management. For prolonged inactivity periods, active sensors may (automatically) switch off their signal emission, but this can be inefficient for continuous use if warm-up or calibration phases are required after power on. To preserve CPU resources, some sensors can be configured to sense and signal only special events. As an example, some accelerometers offer a free-fall detection mode and various distance sensors allow for configurable proximity alarms.

2.3.2.3 *Radio Transmitters*

The major part of the consumed energy of radio transmitters is dissipated by the antenna to form an oscillating electromagnetic field.

Digital radio transmitters use *modulation* schemes to convert data bits to radio signals. A signal carrying information is called *symbol*. A single symbol can carry multiple bits, so that data rate and symbol rate can deviate.

To conserve energy, the goal is to maximize the data rate while maintaining the necessary signal strength at the receiving stations.

However, symbol rate and signal strength are interdependent: The symbols to send occupy a set of frequencies within the radio spectrum. These frequencies are a mix of the modulation frequencies, carrier frequencies if needed, and the frequencies of the symbol rate (identical symbol sequences of different length create different frequencies). Also, neighbor frequencies and harmonic frequencies occur by the naturally limited performance of the used filters. A broad frequency range requires more energy to obtain the same signal quality, because the output energy of the antenna is distributed over the used spectrum. Also, higher frequencies require more energy than lower. The quotient of data rate and used spectrum denotes the *spectral efficiency*.

spectral efficiency

This means, a single frequency and low data rate signal (e.g. Morse code) can be received within a large area, while at the same output energy a multi-channel audio signal (e.g. FM radio) only covers a fraction of this area.

Consequently, to conserve energy on the sender, the output power can be reduced by choosing the minimum viable data rate and the most spectrum efficient modulation scheme available. If possible, transmitters should be switched off between prolonged periods of silence.

2.3.2.4 *Radio receivers*

Radio receivers use energy to filter and amplify radio signals received by their antenna. The power consumption of a listening receiver is independent of the actual reception of a signal.

Thus, the permanent power consumption of a listening receiver can pose a high energetic load on a low power device. To encounter excessive energy profusion for waiting on absent transmissions, multiple measures can be implemented. If time synchronization is possible, sender and receiver can use fixed time slots for transmission. As an extension, *time-division multiplex* (TDM) technologies can be used in multi-node networks, so that bandwidth resources are shared while at the same time reducing the listening periods of the receivers. However, dynamic TDM networks require a complex time management and increase the overhead of network protocols.

low-power listening

Another commonly used method is *low-power listening*. Here, the sender prepends radio transmissions by an easy to recognize *preamble*, e.g. an alternating sequence of symbols that cannot occur in the actual data stream, for a preconfigured duration. This increases the power usage of the sender, but limits the listening duration of the receiver to the minimum time required for detecting the preamble. The receiver only continues listening if the preamble was detected. This preamble check must be repeated at least once per preamble duration. Thus, a longer preamble duration increases the energy consumption at the sender, but reduces the energy consumption at the receiver.

2.4 LOW POWER AND ULTRA-LOW POWER

The terms *low power* and *ultra low power* are widely used among the semiconductor industry, but there seems to be no consistent definition. Often, they are used for marketing purposes to denote systems that somehow use less power than "standard" devices.

Nonetheless, there seems to be a common understanding of these terms when used in a scientific context in the field of embedded systems: Low-power CPUs reduce power on the semiconductor level by using transistor technologies that work under reduced supply voltage, usually allowing for DVFS (e.g. in [29, 58]).

Ultra-low power in semiconductor research...

The term *ultra-low power* is often used in conjunction with sub-threshold voltage supplies and enhanced hardware power management, like supply gating, to reduce the sub-threshold leakage current [78].

...and in IoT research

In the field of IoT research and engineering, ULP is often correlated to net power consumption. Here, devices with an average current consumption below 1 mA are considered ultra-low power. However, the perception of this term is not limited to computational units, but also extends to periphery (with different current thresholds), so that "ultra-low power radio transceivers" and similar terms can be found. Accordingly, several manufacturers, like Texas Instruments and Analog Devices, have made up whole ULP product families of controllers and a broad variety of compatible periphery, including displays, transceivers, and power management hardware.

2.4.1 *Transiently-Powered Systems*

The term transiently-powered system (TPS) denotes a class of systems that is constructed under the assumption of a suddenly failing power supply. The segregation of transiently-powered and non-transiently-powered systems is not strictly binary, as all power supplies have a probability to fail. However, a transiently-powered system must continuously handle power events to even become functional. Thus, power failure and reboot is considered the normal state and has to be dealt with in hardware and software.

In practice, systems solely powered by energy harvesters are the most common examples of transiently-powered systems. If the transient power supply is stabilized by a battery that is able to bridge all expectable low-energy periods (and thus, special software treatment of sudden power failures could be omitted), the system is usually not considered as transiently-powered within the communities around wireless-sensor networks and ultra-low-power systems.

2.4.2 *Energy-Neutral Systems*

Energy-neutral systems are energy-harvesting systems that adapt their energy consumption to the energy income, e.g., by using DVFS and other means of power scaling. By that, energy-neutral systems have low energy-buffer requirements while maximizing the availability of their service.

3

STATE OF THE ART

This chapter gives an overview of the current state of the art in fields of research of this thesis.

3.1 TRANSIENTLY POWERED SYSTEMS

To implement transiently-powered systems, there are two lines of approach:

saving the system state

First, there are solutions that can save and restore the system state over a power fail [7, 44, 61–63]. Saving the application state adds only a slight overhead in hardware and software, and many MCUs have means to detect decreases in voltage supply ("brown-out") before they finally fail, which can be used to achieve state saving. Low overhead and hardware support makes this approach attractive for avoiding data loss in applications that store important state data in volatile memory. However, the process of accessing non-volatile storage devices may use an extensive amount of energy in a situation short of energy supply.

adapting to harvest

power-neutral systems

Second, there are systems that adapt their energy consumption to the energy harvest [6, 14, 33, 43]. The latter are often referred to as *power-neutral* systems [6]. These systems try to avoid power fails by restricting their energy consumption when necessary, e.g., by reducing data transmissions. This involves some overheads in technology, as the available energy needs to be tracked continuously. Chapter 5 will show new approaches to reduce the additional hardware (and expenses) of online energy income measurements to the minimums, while Chapter 6 will show that energy consumption tracking can be handled in software if sufficient energy models are available.

Both approaches to transiently powered systems in their pure form have their distinct advantages and disadvantages, but can be combined.

DVFS cannot adapt to periphery

To tackle the problems of power-neutral systems, proposed system designs make use of energy adaptation by switching sleep modes [10], or by dynamic frequency scaling (DFS) [6] or dynamic voltage and frequency scaling (DVFS) [14]. These adaptive approaches only consider CPU energy consumption.

However, IoT devices, WSN nodes and Industry 4.0 applications are likely to utilize high-energy periphery, e.g., sensors and radio transceivers. The proposed system designs cannot adapt to this type of energetic load. Additionally, I/O communication under DVFS can even become erroneous, as I/O devices may not be able to handle reduced voltages on data buses (or require additional hardware), and prolonged I/O latencies (by reduced CPU frequency) may have a malicious impact on energy consumption.

Thus, dealing with energy-hungry I/O devices in power-neutral systems is challenging, especially in situations where the power consumption of the periphery might be in the range (or even above) of the maximum power harvest. In this case, small size energy-buffers, like capacitors, are still needed to gather enough energy for performing a single peripheral activity. Yet, the rate of activities has now to be adapted to allow for charging the capacitors.

online measurement required

This adaptation process requires a form of energy measurement at either the harvester or the energy buffers. This has to come at minimal cost and technical effort, since the reduction of costs, size and weight are some of the most important benefits over solutions with high-capacity energy storage.

Efficient adaptive circuits for battery-free operations were researched by Brunelli et al. [14]. Here, a maximum power-point tracker was used to charge a supercap instead of a rechargeable battery. A charging efficiency of almost 80% was reached, which they show to be near the optimum for super caps. The approach proposed in Chapter 5 achieves a charging efficiency of about 90% for an electrolyte capacitor. The overall system efficiency reaches almost 70% (see Section 5.4). This is a loss of only about 10% between the optimum supercap charging behavior and the power actually consumed for successful activities by the periphery. This value was reached without any additional *maximum-power-point tracker* (MPPT) hardware.

To scale peripheral loads in power-neutral systems, Gomez et al. show a hardware approach that collects small charges from a harvester to create power-bursts for activities [33]. Their approach shows an efficiency of about 70%. However, their hardware effort is much higher that the solution proposed in Chapter 5.

An augmented approach to online current measurement without additional components is *iCount* [26]. Dutta et al. count the distinct pulses created by pulse frequency modu-

lated switching regulators and show that this method exhibits a maximum error of ±20% for estimating the load current of the regulator. However, while this method is inexpensive, it does not allow to efficiently estimate buffered energy (which requires voltage measurements) or energy income (which requires continuous and precise measurements) and thus is unsuitable adaptation in energy-neutral systems.

In an endeavor to create a converter free sensor node, Lee and Chang created *SmartPatch* [43], a power-neutral UV-level meter that utilizes an external power-cycling power-management unit. Basically, this unit consists of two comparators that switch a flip-flop according to an upper and a lower reference voltage. The flip-flop controls the power output of a PVC using a transistor. With the approach proposed in this thesis, these additional components become unnecessary for low-power MCUs. Additionally, the approach of Lee and Chang can not handle tasks that consume more power than currently harvested, as it has no energy storage. A comparable measure for the efficiency of their design was not given.

A comparison of the above approaches will be presented in Chapter 5, Table 5.4, as far as comparisons are applicable.

Next to measurement based approaches, pure software prediction and accounting methods, like the finite horizon scheme and astronomic prediction proposed by Buchli et al. [15], have shown good results for systems in a predictable environment. Bergonzini et al. [11] give a more comprehensive overview of prediction and adaptation algorithms suitable for energy harvesting nodes. But, prediction methods are of minor use for energy neutral solutions, which continuously have to adapt in short intervals to the most recent energy income. Thus, only energy buffering systems with longer decision intervals benefit from these approaches.

Comprehensive literature for further reading on energy harvesting system fundamentals can be found in [3, 10, 27, 60, 72].

3.2 ENERGY MODELS FOR ONLINE DEPLOYMENT

In this thesis, automata models are used to describe the energy consumption of embedded periphery and processors. The concept is based on preceding research, so that variants of *finite-state machines* (FSMs) were a common base for resource models during the recent decades [12, 57, 80, 82] (and many others).

The reason for the popularity of automata models lies in their simplicity and flexibility. However, the precision and size of distinct automata instances can vary widely, and finding a complete and precise state machine representation for complex hardware can be challenging.

REMES

A comprehensive foundation for automata based resource modeling was contributed by Seceleanu et al. [66], who introduced *REMES*, a hierarchical FSM model for resource consumptions of embedded systems. In their approach, priced timed automata (PTA) models are used, which are able to incorporate cost constants and timed transitions. The approach proposed in Chapter 6 is the minimal subset of *priced timed automata* (PTA) model functionality that is required for the representation of ULP system energy consumptions.

CPU modeling

Next to automaton models, there is a variety of modeling approaches for different purposes: Steinke et al. describe a highly detailed model for the simulation of the energy consumption of processors that even considers changes in electrical charge on all bus connections [71]. While this model yields accurate results for CPU and memory energy consumption, it can not be calculated online, and it does not include peripheral energy consumption. However, this model forms a solid foundation for detailed CPU energy simulations.

mapping energy to code locations

The abstraction from actual hardware is driven further by Tan et al. [73], with the goal to estimate the energy consumption of distinct system services. The authors create an empiric model by measuring the energy consumption of distinct operating system services (by using an energy simulator) in a fixed system configuration. Again, peripheral components can not be considered: As the energy consumption of peripheral devices occurs in parallel to code execution, approaches that do not use a functional abstraction of peripheral hardware cannot map measurement data to distinct code paths: Between the start of an asynchronous I/O operation and its end (marked by an *interrupt request* (IRQ)), any subsequent code path, including process switching, may occur. Thus, measuring/simulating I/O device energy during that time, and mapping to the (randomly) passed code locations, must lead to random results.

operating system level approaches

Approaches implemented on the operating system level are shown in *TinyOS* [77]. *TinyOS* is a popular operating system for sensor networks. It uses drivers that support power management by keeping track of the on-/off state of peripheral devices. Kellner and Bellosa describe an approach to add en-

ergy accounting to TinyOS [38], and also employ state machine models for this purpose. Their approach is focused on the special infrastructure of TinyOS and a networked multi-node environment. There was no evaluation available to determine overheads, constraints, or the achieved accuracy. In comparison, the approach shown in Chapter 6 traces distinct states of a device within a device driver. The principle of model coherence (Section 6.2) makes it applicable for all systems that allow to syntactically isolate code locations and to insert source code (e.g. by using an aspect-oriented language extension like AspectC or AspectC++), overheads and achievable accuracy were empirically evaluated.

online/offline models

Combined online/offline approaches that implement models into the runtime system are typically achieved by reducing complex offline models for online usage. As a prominent example, Kellner et al. propose an optimization of nodes running TinyDB ([47, 48]) by load balancing [39]. This is done by simulating a model synchronously to the actual functionality of the system, thus reducing a complex model for a distinct application. However, as this modeling scheme is strictly application, infrastructure and operating-system specific.

task mapping

A different approach is shown by Martinez et al., who model IoT devices at the system level when a cyclic execution paradigm is fulfilled [50]: Here, the devices execute sub-tasks, like sensing and radio transmissions, in a serial, repeating, and static order. By that, the energy consumption of every sub-task can be measured, simulated or calculated, and the results are summed up to the overall energy consumption of a cycle. This approach supports the dimensioning of energy buffers and harvesters. However, this energy model is highly application specific and cannot be reused. By that, its support for dynamic power-management at runtime is limited. In contrast, the approach shown in this thesis is device specific and reusable. For optimization purposes, the concrete implementation of a model can be synthesized to become application specific, e.g. by assuming a distinct hardware configuration.

3.3 AUTOMATED MEASUREMENT AND MODELING

There are two complementary domains of automated measurement and modeling: Online approaches perform in-situ measurements with integrated measurement hardware and use the resources of the device to analyze the measurement data for

creating and updating energy models. Offline approaches use either internal or external measurement equipment, and use external resources to analyze data.

online vs. offline measurement

The advantages and disadvantages of these methods are complementary: Online approaches are not intrusive and allow for dynamic models that can take user behavior and battery aging into account. However, the accuracy is low due to the usually limited measurement technology. There is a strong requirement to keep the computational load low, so that the energy savings achieved by online models is higher than the costs. However, to the best knowledge of the author of this thesis, all publications that propose online model generation do not consider the energetic overhead of this process. This might be due to the fact that all applications in this field are implemented on smart phone technologies (and bigger), where overhead measurement is challenging, and model generation is a negligible computational task in comparison to other applications, like games, GPS navigation, and web browsers.

online modeling overhead not considered

For offline models, the opposite applies: They can be more are accurate and complex, but are static and the measurement method might be intrusive.

online modeling requires sufficient computational power

For these reasons, online approaches are typically deployed on platforms with sufficient computational capabilities and ready-to-use measurement technologies. Smart phones and laptops are the most prominent examples. One such approach is *DevScope* [36]. The DevScope application queries a set of hardware states (e.g, possible CPU clock frequencies) from the underlying Android operating system. It creates a test case scenario to exercise through states and parameters synchronous to the measurement frequency of the *battery measurement unit* (BMU). With the collected BMU data, it uses polynomial regression to derive an energy model. Another example for online approaches, going a similar direction, is shown by Dong et al. with *Sesame* [25].

Pathak et al. claim that utilization based online models, like DevScope and Sesame, cannot accurately represent energy consumption, as energetic delays, like postponed energy consumptions (tail-states), distort the results [57] – a claim that was empirically verified for CPU by McCullough [53]. The same applies for all models that are generated purely from performance counters [9, 13, 68]. Instead, Pathak et al. proposed a modeling perspective from the system calls of an application. Referring to the famous *gprof* tool, they proposed *eprof* for energy profil-

ing. Similar to the approach of Tan et al. [73] discussed in Section 3.2, and Kjargaard et al. [41], it remains unclear how the energy consumption of parallel I/O operations (while task/process switching) or asynchronous I/O can be handled.

EMrise [84] is a combined measurement and simulation framework which is not limited to smartphones. It uses simulation to effectively answer the question of how many states and parameters are needed to accurately model a hardware component. It considers state transitions, but does not map function arguments or parameters, and requires manual data analysis.

The approach shown in Chapter 7 combines properties of online and offline modeling, to remove some of the issues that arise from application- or user-related modeling and CPU intensive regressions. Here, the model is created offline, but relates to actual hardware utilization locations at device driver implementations. The resulting model can be incorporated into the operating system for online purposes with low overhead. Since the model does not rely on user behavior or application system-call profiles, it removes the need for dynamic updates. It also does not rely on a battery with BMU, where many of the measured values are influenced by the aging processes of the battery.

3.4 DRIVER SYNTHESIS

For the model based synthesis of drivers, complex and feature rich solutions are proposed, by Chen et al. [23] and Wang et al. [81], among others [24, 54, 64, 65, 74] (and more). In contrast to the contribution proposed in this thesis, the synthesis approaches in the literature focus on the formal description of the device functionality to optimize resource consumption or to prove correctness.

In the proposed approach, the non-functional properties of a device are in focus. The synthesis process is restricted to creating only the energy-specific functionality, e.g., energy accounting, and to insert this functionality into a the preexisting driver.

4 ENERGY MEASUREMENT

4.1 INTRODUCTION

Modeling the energy relevant processes of an ultra-low-power system requires exact measurements of the energy consumption of all components in all of their respective states. Also, for the evaluation and comparison of models, hardware components, and software implementations, precise energy measurements are needed.

good resolution in time and value required

ULP systems, i.e. systems with an average power consumption of a few mW or less, pose high demands towards the vertical resolution of power consumption measurements. Nonetheless, due to the highly dynamic nature of computing systems, there are also challenging requirements in the horizontal (i.e., time) resolution: The sampling duration for a single measurement must be significantly less than the minimal duration of a system state under observation. This is expressed by the sample rate of a device, measured in Hertz (Hz) or samples per second (sps) ($1\,sps = 1\,\mathrm{Hz}$).

For ULP periphery, sample rates in the order of 100 kHz can be considered acceptable for tracing periphery states and sleep modes. This implies that a single, energy relevant device state has a minimum duration of 1000 CPU cycles at 50 MHz system clock rate (see Equation 4.1).

factor 2 due to Nyquist-Shannon sampling theorem

$$f_{sample} = 2 \cdot \frac{f_{CPU}}{n_{cycles}} \tag{4.1}$$

In comparison, an instruction-accurate modeling of a CPU requires a minimum sample rate of twice the CPU clock.

However, measuring electrical current both accurately and fast at the same time bears a contradiction: Power consumption measured directly at the output of a power source is always a function of the involved electrical capacitance of the *device under test* (DUT). Due to their charge accumulating nature, capacitances integrate the electrical current over time, leading to delayed and flattened waveforms of high frequency signals. That means, for accurate measurements, the current consump-

tion must be constant for an amount of time for charging all involved capacitances, thus limiting the effective sample rate.

For small currents in or below the µA region, the capacitance imposed by the wiring of a measurement instrument can already interfere. As counter-measure, any unnecessary capacitor on the power supply route (such as buffer capacitors) has to be removed, and low capacitance measurement equipment must be used.

Additionally, for energy modeling there is a requirement to synchronize measurement results with system activities. This can be done by additional digital input lines to the measurement equipment, thus enabling the DUT to signal state changes. With such signal-annotated measurements, power consumptions can easily be mapped to system states and events.

These complex measurement requirements are rarely demanded out of the ULP computing domain, which limits the available choice of lab-ready and reliable measurement equipment.

To accomplish research in the field of energy modeling, most of the available measurement devices on the market are not sufficient to the best of my knowledge. Within a reasonable price range, they either lack accuracy/resolution (simple amplifier-driven shunt meters), are too slow (high precision DC meters), do not have sufficient recording capabilities (most oscilloscopes) or lack auxiliary digital inputs (e.g. Keysight's *source measurement units* (SMUs) family).

The Keysight SMU systems, that at least fulfill the basic accuracy and timing requirements, were available for a price of approx. 17,000 € in 2018. This makes it a viable choice for serialized experiments, assuming the event-synchronization issue can be worked around. For parallel experiments, economic considerations may require a less expensive solution.

In the market of low-cost equipment for ULP development, the popular and awarded Otii Arc device is a typical example: At a price tag below € 1000, it bears a time resolution of 4 ksps [59], and thus is useful for general energy measurements and optimizations (as advertised), but insufficient for time accurate energy modeling.

This led to the development of the *MIMOSA* measurement platform, an inexpensive approach to energy measurement for ULP embedded systems.

The following section describes the fundamentals of energy measurement from a perspective of physics and electronic engineering. Section 4.4 will then discuss the principles of the *MIMOSA* measurement platform.

RELATED PUBLICATIONS

The findings presented in this chapter have partly been published in:

[17] M. Buschhoff, C. Günter, and O. Spinczyk. "MIMOSA, a Highly Sensitive and Accurate Power Measurement Technique for Low-Power Systems." In: *Real-World Wireless Sensor Networks*. Ed. by K. Langendoen, W. Hu, F. Ferrari, M. Zimmerling, and L. Mottola. Vol. 281. Lecture Notes in Electrical Engineering. Springer International Publishing, 2014, pp. 139–151. ISBN: 978-3-319-03070-8. DOI: 10.1007/978-3-319-03071-5_16.

[40] S. Kerrison, M. Buschhoff, J. Nunez-Yanez, and K. Eder. "Measuring Energy." In: *ICT - Energy Concepts for Energy Efficiency and Sustainability*. Ed. by G. Fagas, L. Gammaitoni, J. P. Gallagher, and D. J. Paul. Rijeka: InTech, 2017. Chap. 03. DOI: 10.5772/65989. URL: http://dx.doi.org/10.5772/65989.

4.1.1 *Contribution*

This chapter presents MIMOSA, a measurement technology that combines different approaches:

- Voltage-drop compensation: The measurement and amplification unit uses a feedback loop to compensate the voltage drop of a shunt resistor. This allows for better accuracy using high-ohmic shunts and the removal of further measurement amplifiers.
- Precise analog integration: MIMOSA uses analog integration circuits to obtain a continuous and precise energy observation of high frequency signals (above the sample rate).
- Synchronized digital signaling: The MIMOSA approach allows digital signaling of external events, and incorporates the external signal into the measurement data for event synchronization.

4.2 FUNDAMENTALS OF PHYSICS AND ELECTRONIC ENGINEERING

This section imparts fundamental principles of physics and electronic engineering for readers from other disciplines. It is limited to the most fundamental terms that are necessary to understand the basic concepts of the MIMOSA measurement device.

Electric charge, potential, voltage, electric and magnet fields, and current are strictly interdependent, in a way that they always occur together at the same time without a causality order. By that, it is impossible to explain these phenomena without cyclic reference to another phenomenon.

4.2.1 *Electric Charge*

Electric charge (formally Q, measured in coulomb [C], $1\,\mathrm{C} = 1\,\mathrm{A\,s}$, is a property of matter describing the force it experiences within an *electric field*. This physical force causes matter to be attracted to one side of the field. Depending on the direction of movement (towards the positive or negative side of the field), a charge is denoted positive or negative, in a way that unlike charges attract each other.

4.2.2 *Electric Field*

An electric field is the spatial structure of all electric forces affecting a section of space. It is a result of the spatial separation of electric charges that can occur as a result of several processes, including chemical reactions in batteries and mechanical work in generators.

An electric field can be represented as a vector field. For every point of an electric field, the field strength $\vec{E}$ can be measured by observing the force $\vec{F}$ that affects a charge Q resting at that point:

$$\vec{E} = \frac{\vec{F}}{Q} \tag{4.2}$$

4.2.3 *Electric Potential*

The acceleration-free movement of an electric charge Q from a point l to a reference point r within an electric field requires

energy. For a charge resting at l and being attracted towards r, this energy is considered as potential energy $E_P(Q, r, l)$ of the charge. The potential energy can be divided by the charge amount to normalize the term, so that the resulting unit, volt [V], is a charge-neutral measure that describes the electric potential of a point within an electric field towards a reference point. To create a standardization for this unit of measurement, i.e. a zero potential, a common reference point must be defined. Typically, Earth or a point in infinite distance serves this purpose.

$$V(r, l) = \frac{E_P(Q, r, l)}{Q} \tag{4.3}$$

4.2.4 *Voltage*

Voltage (formally denoted as V, sometimes as U, measured in volt [V]) denotes the difference in the *electric potential* of two spatial points. Thus, for two given points represented by their potentials V_a and V_b, there is a potential gradient $V = V_a - V_b$. Assuming that both electric potentials refer to the same reference point r, voltage describes the electric potential of two points within an electric field towards each other (with unlike signs).

$$V(a, b) = V(r, b) - V(r, a) = -(V(r, a) - V(r, b)) = -V(b, a) \tag{4.4}$$

4.2.5 *Electric Current*

The term electric current describes the movement of an electric charge Q during an interval of time t. It is caused by, and is proportional to, the potential difference (*voltage*) of two points in space. As charges cannot move freely through matter, there are further factors that influence their movement (see Section 4.2.6).

Electric current is formally denoted as I, and is measured in ampere [A], with $1\,\mathrm{A} = 1\,\mathrm{C/s}$.

$$I(t) = \frac{\mathrm{d}Q}{\mathrm{d}t} \tag{4.5}$$

4.2.6 *Electrical Resistance*

Electrical resistance, formally denoted as R and measured in ohm [Ω], with $1\,\Omega = 1\,\mathrm{V/A}$, is a property of matter that limits the movement of electric charges. Charges with infinite mobility would equalize spontaneously, so that potential gradients (voltage) would be an impossible concept. By the resistance of a material, charges can be separated and accumulated. Thus, electrical resistance is a requirement for voltage, but limits current. This is represented formally in *Ohm's Law*:

$$R = \frac{V}{I} \tag{4.6}$$

The electrical resistance value of matter mainly depends on the electron configuration and crystal structure of the material. It is influenced by temperature and the topology of the material (crystal grid orientation, cross section area) within the electric field.

Of special interest for electronic engineering are materials with very high resistance (insulators), very low resistance (conductors) and materials with variable resistance depending on environmental parameters (sensors). Semi-conductor materials (e.g. silicon, germanium) can be influenced greatly in their resistance by inducing chemical impurities into their crystal structure ("doping"). Spatial doping profiles with different materials allow for complex and dynamic electric behaviors. This enables compact micro-technology devices.

4.2.7 *Electric Energy*

Electric energy, formally denoted as E and measured in joule [J], with $1\,\mathrm{J} = 1\,\mathrm{V\,A\,s} = 1\,\mathrm{W\,s}$, is the energy required to move an electric charge Q between two points in an electric field over potential gradient V. It is identical to the potential energy as described in Section 4.2.3.

$$E \overset{4.3}{=} V \cdot Q \tag{4.7}$$

4.2.8 *Electric Power*

Electric power, formally noted as P and measured in watt [W], with $1\,\mathrm{W} = 1\,\mathrm{V\,A}$, is the *rate* at which energy is transferred in an electric system.

$$P(t) = \frac{dE}{dt} \tag{4.8}$$

For the common case of non-constant voltage and/or current, electric power can be determined for a point t in time:

$$P(t) \overset{4.7}{=} V(t) \cdot I(t) \overset{4.6}{=} I(t)^2 \cdot R = \frac{V(t)^2}{R} \tag{4.9}$$

Power is defined as the rate of energy transfer, thus energy can be calculated as:

$$E = \int_t P(t)\,\mathrm{d}t \tag{4.10}$$

4.2.9 *Capacitance*

Capacitance, formally C, measured in farad [F], with $1\,\mathrm{F} = 1\,\mathrm{C/V}$, is a measure for the change of charge that occurs within a system due to a voltage change, as given by Equation 4.11. It is dependent on physical parameters of the materials used in a system and the geometry of the system. As a component of electronic engineering, a *capacitor* is a system with a defined capacitance consisting of two electrodes separated by an insulator ("dielectric").

$$C = \frac{\mathrm{d}Q}{\mathrm{d}V} \tag{4.11}$$

Increasing the charge of a system also increases the voltage within the system, and in consequence the system gains potential energy.

capacitors have a dynamic, i.e., time dependent behavior

Equation 4.12 shows the relation between a voltage change during an interval $t = t_1 - t_0$ and the resulting energy change.

$$\begin{aligned} \Delta E_t &= \int_t Q(t)\,\mathrm{d}t = \int_t C \cdot V(t)\,\mathrm{d}t \\ &= \frac{1}{2} C \cdot (V(t_1) - V(t_0))^2 \end{aligned} \tag{4.12}$$

In a capacitive system, there is a relation between the rate of change of the voltage and the current, as shown in Equation 4.13.

$$I(t) = C \cdot \frac{\mathrm{d}V(t)}{\mathrm{d}t} \tag{4.13}$$

4.3 MEASUREMENT TECHNIQUES

The following section describes common techniques for the measurement of the transfer of electric energy in computer systems. Here, the common case of a known and constant supply voltage for the DUT is presumed,

According to Equation 4.9, power transfer can be measured by sampling the current over time. Energy transfer can then be calculated by Equation 4.10.

4.3.1 *Shunt-Based Measurement*

Shunt-based measurement is a commonly implemented method in basic measurement devices. The principle is to convert an electric current into voltage, and then measuring the voltage instead, as this is technically easier to achieve.

According to Ohm's Law (Equation 4.6), a current over a resistance can only be induced by a potential gradient ("voltage drop"), which is proportional to the current. Thus, shunt-based measurement devices route the current to observe over a low-ohmic resistor ("shunt") and measure the voltage drop at the junctions of the shunt.

Figure 4.1 shows this setup. The DUT is modeled as the variable resistor R_{DUT}. The measurement device, represented within the dashed box, has to be inserted into the power-supply line of the DUT, so that all of the current passes the shunt. For this purpose, the direct connection between one side of the voltage source and the DUT has to be opened and bridged by the measurement device.

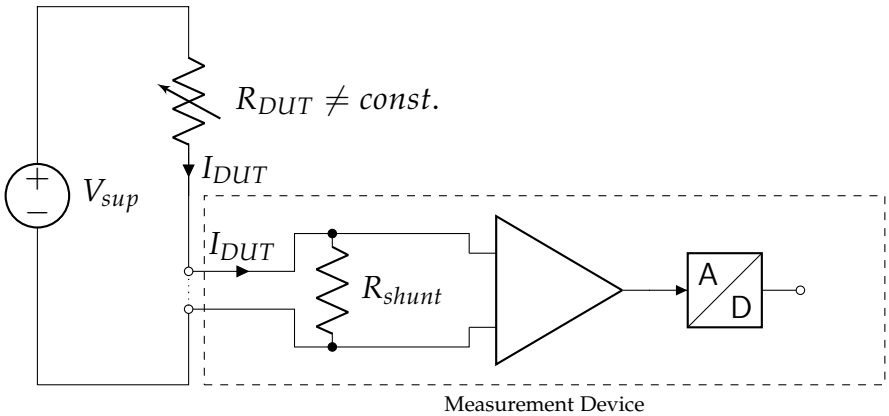


Figure 4.1: Schematic of a shunt-based measurement

It can be assumed that the DUT behaves like a variable resistor R_{DUT} [1]. The supply voltage V_{sup} will then be divided by the two serial resistors R_{shunt} and R_{DUT}. That means, the voltage V_{DUT} at the DUT will drop according to Equation 4.14.

$$\begin{aligned} V_{sup} &= V_{DUT} + V_{shunt} \\ \Leftrightarrow V_{DUT} &= V_{sup} - V_{shunt} \overset{4.6}{=} V_{sup} - [R_{shunt} \cdot I_{DUT}] \end{aligned} \tag{4.14}$$

high-ohmic shunts influence the DUT

This shows that a high-ohmic shunt resistor causes a considerable voltage drop, which is also dependent on the current consumption of the DUT. Thus, to avoid low voltage conditions, the shunt value must be as small as possible.

To ensure a minimum supply voltage of $V_{DUT,min}$ at a given maximum current $I_{DUT,max}$, the maximum shunt value $R_{shunt,max}$ is given by Equation 4.15.

$$R_{shunt,max} = \frac{V_{shunt,max}}{I_{DUT,max}} = \frac{V_{sup} - V_{DUT,min}}{I_{DUT,max}} \tag{4.15}$$

For an energy efficient device, it can be considered that V_{sup} is close to $V_{DUT,min}$. Thus, shunt-based measurement devices typically employ the lowest-ohmic shunt that is technically feasible. In other words, the influence of the measurements on the DUT supply voltage shall be as low as possible.

low-ohmic shunts reduce accuracy But, a low-ohmic shunt will also cause V_{shunt} to be low. As V_{shunt} is the actual objective of measurement, this usually bears the requirement for a measurement amplifier to feed an *analog to digital converter* (ADC). For ULP devices, with currents in the mA and µA region, there are high quality requirements for the amplifier, as all components add *static noise*[2] and *thermal noise*[3]. In consequence, when V_{shunt} is in the same order of magnitude than the noise, shunt measurements become highly inaccurate. Thus, as noise limits the vertical resolution of measurements, it is important to keep the signal-to-noise ratio high by avoiding long current paths through many components.

1 In fact the DUT may also pose a capacitive load, but at constant voltage this leads to a delayed energy transfer, which in sum still leads to correct measurements. However, high capacitive loads should be avoided to maintain time and activity correlations during the measurement.

2 Noise induced by nearby electric devices.

3 Noise created by temperatures above 0 K, when thermal movements of atoms and electrons within a crystal grid generate random currents.

As a conclusion, separating the measurement signal from noise calls for a high-ohmic shunt, as the shunt voltage is directly proportional to its resistance. But, this demand is countered by the requirement to minimize the measurement influence on the DUT.

4.3.2 *Charge Accumulation Measurement*

Another form of current measurement is achieved by accumulating charges in a capacitor. According to the charging behavior of a capacitor, its voltage raises monotonously over time, depending on the charging current (Equation 4.13).

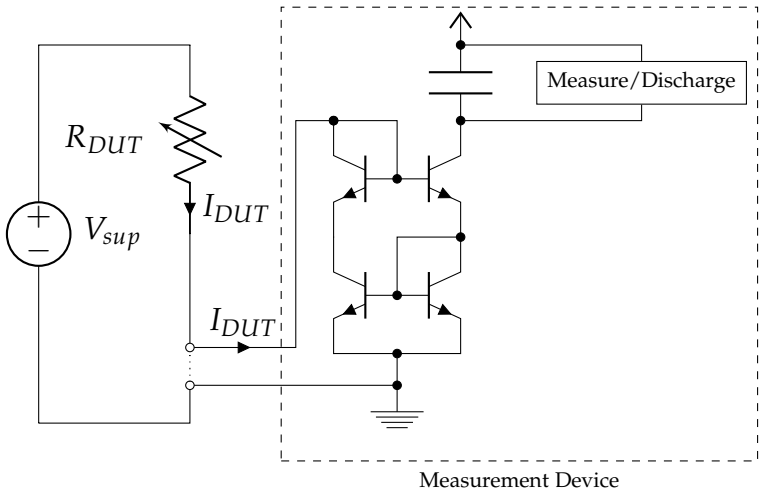


Figure 4.2: Schematic of a Wilson current mirror to charge a capacitor.

The capacitor cannot be inserted directly into the current path, because its conductivity depends on its charge-state: While an uncharged capacitor acts like a short-circuit connection, it becomes non-conductive with accumulated charge. Thus, the charged capacitor would drop the supply voltage. To avoid this, a *current mirror* circuit can be used.

Current mirrors are basically current amplifier circuits with an amplification factor of 1. It is important to understand that a current mirror has a low input impedance, so that the current on the input side is uninfluenced by the mirror. On the output side, the voltage is regulated so that the output current resembles the input current, even under high load.

Figure 4.2 shows a measurement setup utilizing a *Wilson current mirror* that consists of four transistors.

The measurement unit used to sense the capacitor charge can be implemented in three different ways:

1. The voltage of the capacitor can be sampled at a constant rate by an ADC.

2. The time needed to reach a predefined threshold voltage can be measured by using a comparator and a timer.

3. The rate of charging the capacitor can be determined by a comparator, counter and timer.

These methods have their specific advantages and disadvantages. The fixed sample rate of the first method is usually a desired behavior. Its accuracy is dominated by the quality of the ADC and by the charge of the capacitor, because it does not charge linearly. Holding a constant charge within a capacitor during the ADC sampling period adds further components and complexity.

Time measurements, as in method 2, have a well defined precision that only depends on the resolution of the used timer. Anyhow, their variable rate of measurements might be inappropriate for some applications.

Method 3 avoids these problems. It uses a small capacitor that charges and discharges quickly within the respective measurement range, and increments a counter each time the capacitor reaches a respective comparator value. The counter value is read and reset at a constant rate to derive the measurement value. By that, a constant sample time is guaranteed, while removing the requirements for a high timer resolution. The accuracy limit is dominated by the quality of the capacitor and the comparer.

In any of the three cases, the capacitor has to be discharged after each measurement, causing a further inaccuracy, as it cannot collect charges during discharge cycles.

disadvantages

The biggest drawback of using capacitive measurement is the technical implementation of a current-mirror: The used transistors must be identical in their electrical behavior on a level of precision that is barely achieved in reality. Even current mirrors manufactured on a single chip often show unacceptable tolerances for ultra-low power measurements. Additionally, the low input impedance required for measurements in the µA range is difficult to achieve. The quality of the capacitor also greatly affects the measurement quality.

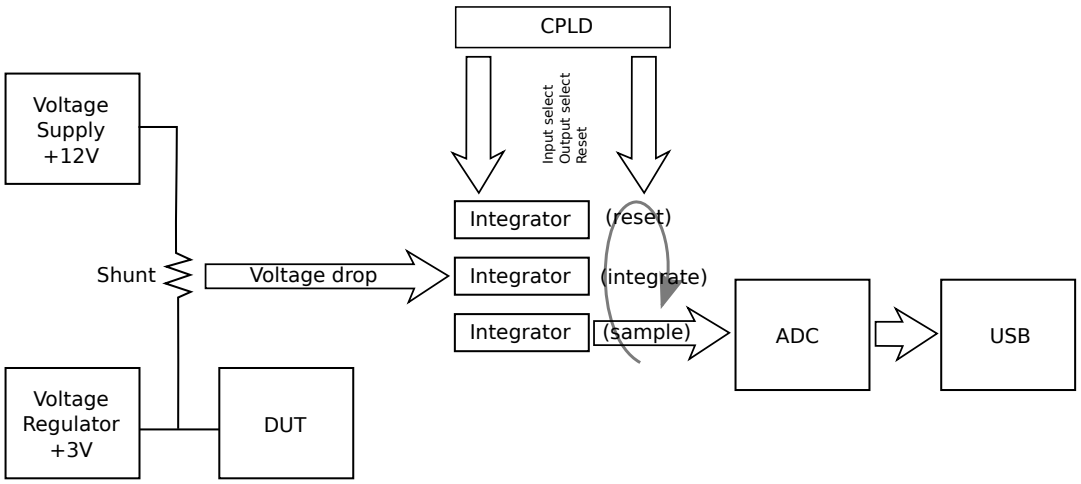


Figure 4.3: Overview of the MIMOSA system (source: [17])

4.4 THE MIMOSA APPROACH

In an endeavor to tackle the challenges described in the introduction of this chapter, a measurement device was constructed to allow for inexpensive, precise and event-synchronized measurements of energy consumption in ULP devices. We named this device "MIMOSA", an acronym for the German term "**M**essgerät zur **i**ntegrativen **M**essung **o**hne **S**pannungs**a**bfall" (measurement device for integrative measurement without voltage drop).

The goals of the MIMOSA development were:

- use of a resistor for current to voltage conversion to reduce complexity
- compensation of any voltage drop at the DUT, enabling the use of high-ohmic shunts
- analog integration of current using a capacitive integrator circuit
- continuous, gap free measurement also during sampling and capacitor discharge periods
- a sample rate of 100 kHz
- a selectable input range and accuracy
- Low material costs to enable parallel measurements with potentially many devices.

Figure 4.3 shows an overview of the measurement system. The following sections will discuss the approach in two parts. Section 4.4.1 discusses the idea of a shunt-like current to voltage conversion without voltage drop at the DUT. Section 4.4.2 explains the integrator circuits used to feed the ADC.

4.4.1 *Shunt Measurement without Voltage Drop*

To compensate the voltage drop that occurs at a shunt resistor, the MIMOSA device was designed as power supply for a DUT, in contrast to passive measurement techniques where the measurement device is included into a circuit and ideally behaves fully transparent.

The integrated power supply, shown in Figure 4.4, is a regulated voltage source with a precise reference voltage. It can always operate the DUT at its defined voltage level V_{DUT}. To achieve this, an operational amplifier is used to implement a *voltage buffer*. This circuit mirrors an input voltage (the reference voltage V_{ref}) to its output, while the current is delivered by the power supply of operational amplifier. By using a feedback loop, the output voltage is always regulated to the reference voltage, so that any voltage drop at the output is compensated by the circuit.

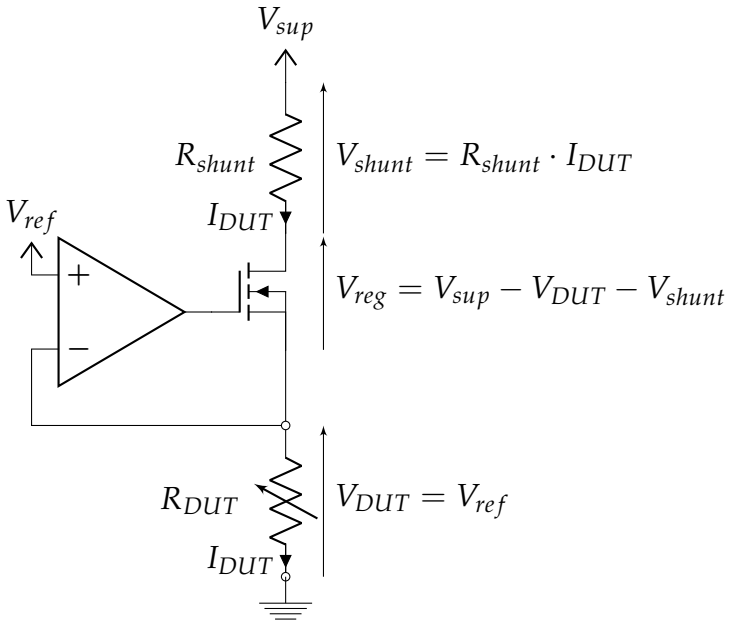


Figure 4.4: Simplified schematic diagram of the MIMOSA voltage regulator

As shown, a transistor is used to decouple the output current of the operational amplifier from the current used to drive the DUT. In this non-inverting configuration, the transistor does not interfere with the feedback loop, as the voltage at the transistor's gate contact is proportional to the voltage at the source contact. Anyhow, no considerable current can flow over the gate of the transistor. Instead, the current is delivered by the voltage supply powering MIMOSA, over R_{shunt}, and the drain-source path of the transistor.

In summary, the following was achieved:

- The only stable configuration of the operational amplifier exists at the equilibrium of the input junctions. By a negative feedback loop, the operational amplifier thus regulates the feedback loop voltage V_{DUT} to V_{ref}.
- The output of the operational amplifier has no current path to the DUT, as the gate contact of the transistor is isolated from the other contacts. The operational amplifier is not loaded.
- The input pins of an ideal operational amplifier are free of current by definition.
- As a consequence, the current through R_{shunt} equals the current through R_{DUT}, while V_{DUT} is (within limits) independent of the load current and the shunt.

The supply voltage V_{sup} of the measurement system must be greater than $V_{DUT} = V_{ref}$. Thus, the voltage drop at the shunt lies between 0 V and $V_{sup} - V_{DUT}$. Any surplus voltage is dropped by the regulating transistor (V_{reg}).

As a result, R_{shunt} can be chosen as an (inexpensive) high-ohmic precision resistor, so that V_{shunt} does not require further amplification. By that, a better signal to noise ratio can be achieved while reducing costs at the same time.

4.4.2 *Current Integration*

An important part of the MIMOSA system is the integration unit, as the measurement goal is determining the energy consumption of a DUT, whereas the circuit shown in Section 4.4.1 only delivers V_{shunt}.

Equation 4.16 is used to calculate the energy consumption of the DUT.

$$
\begin{aligned}
E_{[t_0,t_1]} &\overset{4.10}{=} \int_{t_0}^{t_1} P(t)\,\mathrm{d}t \\
&\overset{4.8}{=} \int_{t_0}^{t_1} V_{DUT} \cdot I_{DUT}(t)\,\mathrm{d}t \\
&\overset{4.6}{=} V_{DUT} \int_{t_0}^{t_1} \frac{V_{shunt}(t)}{R_{shunt}}\,\mathrm{d}t \\
&= \frac{V_{DUT}}{R_{shunt}} \int_{t_0}^{t_1} V_{shunt}(t)\,\mathrm{d}t
\end{aligned}
\tag{4.16}
$$

As shown, the energy can be calculated by integrating over samples of V_{shunt}. However, any energy peaks shorter than a sample period are potentially removed or distorted by the filtering mechanisms of the *ADC*, which usually utilizes a *sample-and-hold* circuit to sample momentary values.

An analog integrator can precisely integrate V_{shunt} for the duration of a sample period, leading to a continuous integration. By that, energy peaks shorter than a sample period will raise the next sample value proportionally to the energy consumption.

Integration and sampling is performed in three steps:

1. Integration of V_{shunt} over a complete sampling period.
2. Holding the integrated value constant for sampling during the next sampling period.
3. Resetting the integrator during the third sample period.

Thus, to achieve continuous measurements, three integrators are required which work in alternating order.

Each of the integrators is implemented as shown in Figure 4.5. The circuit is set up around a standard integrator loop consisting of the operational amplifier Op_1 and capacitor C_1. The input voltage from the shunt is fed into the loop by potentiometer P_1, which is necessary to calibrate the integrators to compensate their component-specific deviations. Five digital input signals control the state of the integrator by switching the transistors T_1-T_5.

T_1 and T_3 work together with diode D_1 to isolate the capacitor from the rest of the circuit while sampling. By connecting the capacitor to ground potential via T_5 while sampling, a stable reference potential for the ADC is achieved. The *clear* signal is used to reset the capacitor. A *GND* signal can be used to inject ground voltage into the integrator loop for calibration purposes.

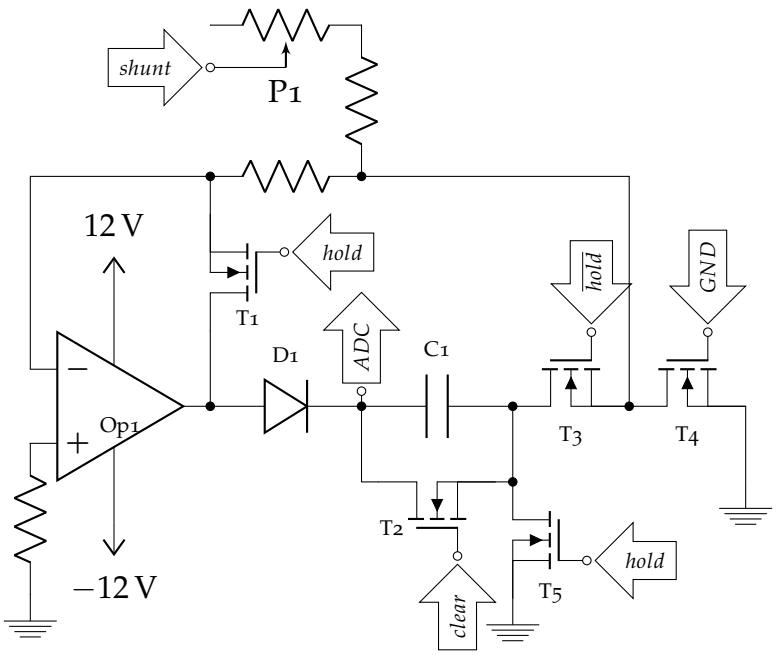


Figure 4.5: Schematic of a MIMOSA integrator.

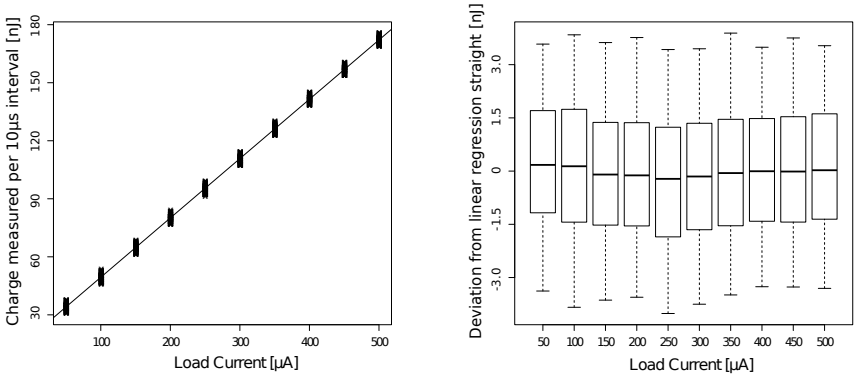
4.4.3 *MIMOSA Performance Evaluation*

To analyze the performance of MIMOSA, two basic test setups were evaluated. The first setup uses a set of 10 predefined high-precision resistors with equidistant values to load MIMOSA with direct current (DC) over its measurement range. This setup is able to show inaccuracies beyond the 0.1% error of the high-precision resistors. It will also show the measurement linearity throughout the whole measurement range.

Figure 4.6a shows the results of 3000 consecutive samples taken with a 30 kΩ shunt for each load value. The x-axis shows the resulting current at $V_{DUT} = 3\,\mathrm{V}$. The y-axis shows the energy of the cumulated charge in the 10 nF integration capacitor. This value is calculated from the sampled ADC voltage of the integrator using Equation 4.12.

The values appear as a vertical bar, because the marks are overlapping due to the small deviation of the values. The figure also shows a regression line (with minimal *root-mean-square error* (RMSE)) that can be used as an optimized model for calculating DUT energy consumption from sampled values. This model is able to compensate static and linear errors that occur due to the error characteristics of the used components.

Figure 4.6b shows the deviation from the regression line, showing that noise is below ±3.5 nJ. For the lowest energy



(a) Values of 3000 measurements (b) Signal noise

Figure 4.6: DC current measurement with MIMOSA in the µA range.

value (cumulated energy 33 nJ) the error is less than 12 %, for the greatest value (170 nJ), the error below 2.1 %.

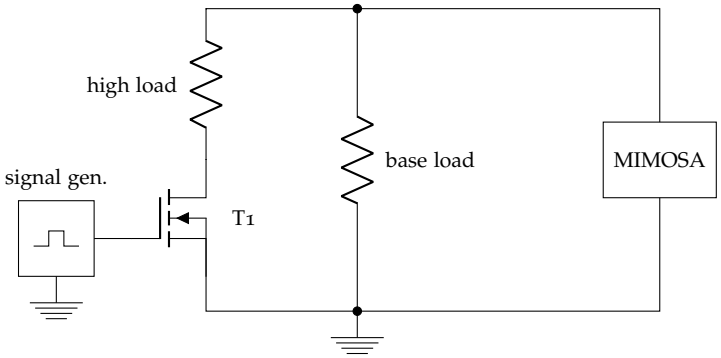
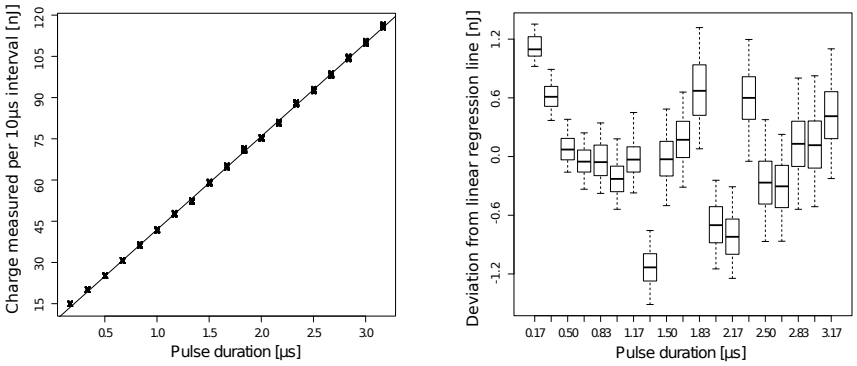


Figure 4.7: Alternating current test setup

Another evaluation setup was used to test the performance of the analog integrators. In this setup, an external circuit switches a resistor using a series of rectangular pulses at a decreasing time interval. The test setup is depicted in Figure 4.7. Here, the MIMOSA device is directly connected to two resistors, one simulating a low current base load, and another one simulating a high power consumption. The latter is added to the measurement by closing transistor T_1.

With different frequencies of the rectangular signal produced by the signal generator, the dynamic behavior of MIMOSA can be observed. By generating pulses shorter than the sample rate, the linearity and precision of the integrator circuits can be analyzed.

Figure 4.8a shows equidistant measurements for pulse durations between 166 ns (equivalent to a 6 MHz signal) and 3 µs



(a) Measured Values (b) Signal noise

Figure 4.8: Rectangular signal measurement with MIMOSA, with pulse durations below sampling duration.

(333.3 kHz) with linearly increasing energy consumption. The sampling period of MIMOSA is 10 µs (100 kHz sampling rate). Again, the regression line with minimal RMSE is plotted.

Figure 4.8b shows that the deviation from the regression line was never above ± 1.8 nJ.

4.5 CHAPTER SUMMARY

This chapter introduced basic concepts of external measurement technologies for the modeling of the energy consumption of ULP components. The modeling process requires energy measurement equipment with vertical resolutions in the µA range and horizontal resolutions of at least 100 ksps. Additionally, an event channel is required to tag or timestamp relevant software events in the measurement data stream. Measurement data acquisition over several minutes must be possible. Finally, setting up parallel measurements requires economic considerations for the purchase of multiple measurement devices.

It was shown that current market-ready measurement solutions do not suit the purpose: Oscilloscopes need additional equipment for precise current measurements, and data tagging and long-time data acquisition is often challenging in practice. Distinct energy measurement devices do not show the resolution required for modeling. All SMUs with sufficient resolution that are known to the author do not allow to tag measurement data, are available only in a price range above € 10,000, and

synchronizing multiple of these devices in practice can be challenging due to their proprietary software and data interfaces.

As a solution to this problem, MIMOSA was introduced. The MIMOSA device fulfills all measurement demands for energy modeling. With a vertical resolution of below 10 µA, a horizontal resolution of 100 ksps, analog circuits that precisely integrate sub-sample peaks at frequencies up to 6 MHz, digital data inputs, direct USB logging features and prototype costs below € 300, MIMOSA has proven to be a valuable tool for the further research shown in this thesis.

5

PHOTOVOLTAIC HARVEST MEASUREMENT

5.1 INTRODUCTION

online measurements for energy harvesting income

In Chapter 4, external measurement methods were discussed. The presented methods help creating energy models at design time ("*offline*"). While these models already bear the potential to track energy consumptions at run time (see Chapter 6), there is still a demand for continuous *online measurements*, e.g., for tracking the power generation of energy harvesters.

There are already single chip measurement solutions for computing systems on the market that support battery control and charge estimation in smart phones and mobile PCs. However, battery free ULP IoT devices pose new challenges to online measurement systems.

Computing systems powered by energy harvesters are typically equipped with rechargeable batteries or super-capacitors as energy storage. They act as dependable power supply during phases of low energy harvest. By that, an application on a well-dimensioned system can run energy hungry tasks like radio communication at any time without respect to the harvesting situation.

Systems without energy storage may suffer from varying power levels. These systems are termed *transiently-powered computing systems* [46]. They are motivated mainly by the disadvantages of rechargeable batteries listed in Section 2.2.1.

By the large-scale deployment of nodes in Industry 4.0 scenarios, economic solutions become mandatory.

Low maintenance requirements are a strong motivation for battery-free systems. Current *smart city* or *Industry 4.0* scenarios envision the deployment of hundreds of thousands of computational nodes. Managing the batteries of such a network exposes hard to solve logistic problems. In contrast, transiently-powered systems completely remove battery maintenance cycles.

Apart from maintenance, there is also a large technical and economic overhead in the hardware setup for a battery-driven node. To outline this, Figure 5.1 shows a typical sensor node design that uses a a PVC as its primary power source.

A MPPT circuit is typically used to dynamically adapt the energetic load of a PVC to the illumination, thus driving it to best

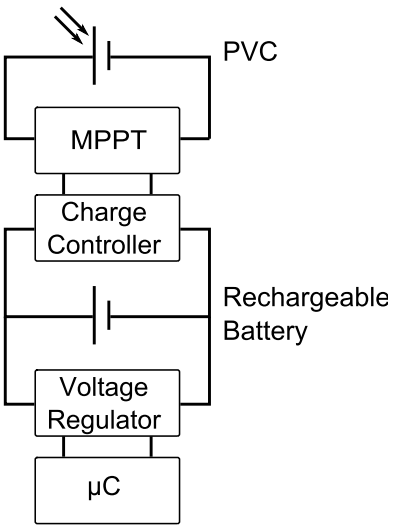


Figure 5.1: Default circuit for a solar harvester

operating conditions. Additionally, a charge controller converts the harvested energy to the operating conditions of the battery. Lastly, an optional voltage regulator converts the battery voltage to the operating conditions of an MCU.

Reducing the amount of used components to the minimums leads to *converter-less* and *energy storage-less* systems. However, even if such a setup is feasible for a particular power source, it limits some of the capabilities of the system, mainly its use for critical applications and its energetic efficiency. As there is a plethora of different technologies for energy storage, charging units, converters and MPPTs, there is also a wide design space for energy supply circuits. This makes it interesting to find the lower bounds of application requirements and simplicity, respectively material costs.

To shed some light on the prototype costs of using battery driven setups, Table 5.1 shows the component expenses for the transiently powered sensor node as proposed here, and the cost-overhead for an MPPT circuit (includes battery charger) and a thin-film battery. Excluded are the prices for the circuit board, some minor electronic components, and the PVC, as the used Solems indoor PVC is currently not available on the retail market. As shown, the battery-related hardware in the lower section of the table increases the price by $10.13.

This shows that, although the application scenarios for transiently-powered systems are limited, their advantages over battery-buffered photovoltaic systems should be exploited wherever possible.

Type	Article	Price [USD]
MCU	MSP430FR5969IRGZR	5.29
V-Reg 3V	TPS78228DDCT	0.55
Transceiver	nRF24L01	1.72
Light sensor	MAX440009	3.57
Subtotal		**11,13**
MPPT-Circuit	BQ24650RVAT	5.18
LiIon battery	PRT-13853	4.95
Total		**21,26**

Table 5.1: Basic component prices of an example sensor node, as of March 2017, either from TI online store (if available) or DigiKey. Prices were chosen for the smallest available batch size.

5.1.1 *Contribution*

This chapter proposes a system design for ULP systems that removes the power converters (MPPT, charge controller) and the rechargeable battery. To support high-energy periphery, a standard electrolyte capacitor is used as buffer for single activities. To maintain adaptive behavior, robust power-management circuits were implemented that utilize a very small set of electrical components (a resistor and/or a Zener diode, no analog to digital converters or comparators). In comparison to state-of-the-art approaches, this approach also respects peripheral energy consumption, while still having a lower part overhead for the hardware implementation.

In short, the most important contributions of this chapter are:

- Methods to reduce cost, size and weight overheads by reducing power-management hardware to the limits.
- An energy-neutral PVC system design based on these methods and its efficiency evaluation (including peripheral load), showing competitive results compared to the state of the art in energy-neutral systems with less hardware components.
- Small-scale energy buffer support, applicable for periphery with higher power consumptions than harvested.

5.2 CONCEPT

PVC powered devices use large energy buffers to bridge periods of low energy income. By that, a well dimensioned device can sustain continuous operation when deployed in a predictable environment. By the steady availability of sufficient power, these systems do not need special measures in software to adapt to the available light.

All the above assumptions change when batteries are removed: The device cannot sustain operation in dark periods, and by starting activities in low illumination, it will exhaust the available energy and thus cause system failures.

applicable only for special purposes

Removing the battery limits the intended purpose of the device. For example, it is not possible to enable pure PVC powered operation in dark periods. Such a system can only be used for non-critical applications, where delayed operation and failure during dark periods is acceptable. Examples for these systems are sensor network nodes that take daily sensor samples, and equipment that is used in illuminated areas in Industry 4.0 scenarios.

adaptation to energy income

However, a device can operate in reduced illumination, if the execution rate of energy consuming periphery operations is adapted. To achieve this, energy for at least one operation must be collected in a buffer. When the buffer is sufficiently filled, a single periphery operation can be performed. Afterwards, the system sleeps again until the next operation is possible.

As the amount of energy to store for single hardware operations (e.g., sending a radio packet) is extremely small in comparison to battery capacities, this enables a change towards a more sustainable and less expensive energy storage technology: Capacitors and Super Capacitors (see Section 2.2.2).

To estimate the charge of a capacitor, methods of charge accumulation measurement (see Section 4.3.2) are promising, because components like comparators, counters and timers are already present on most MCU platforms. A current mirror is not needed, because here, the measurement objective already is the cumulated charge in a capacitor.

In the following section, a basic capacitive measurement circuit is presented. To reduce costs for mass deployment, the circuit components then get reduced as far as possible by exploiting *parasitic mechanisms* and certain I/O-pin features of typical MCU hardware.

5.2.1 *Basic Circuit*

The starting point for the charge measurement circuit is a software-aided *voltage-to-frequency converter* (VFC), that basically resembles method 2 shown in Section 4.3.2.

an ADC cannot sense its own supply voltage

Although an ADC could be used as an alternative to a VFC for the purpose of power-management, it was avoided intentionally. ADCs need a reference voltage, and the respective voltage generator is often dependent on a stable supply voltage. Thus, using an ADC to sense its own supply voltage can lead to random results. Additionally, ADCs on the MCU should be saved for the actual sensor application.

flexible base circuit, many parts

The circuit shown here has only minimal requirements towards the used MCU, making it is more flexible than the reduced circuits shown in later steps. Figure 5.2 depicts a circuit that implements this approach.

The idea is of this approach is to charge an energy buffer (capacitor or supercap) by a PVC. The voltage of the buffer and the PVC are always equal, i.e., a depleted buffer drops the voltage of an illuminated PVC while charging, and rises the voltage when discharging. This voltage is the supply voltage V_{supp} for the rest of the circuitry.

$E = \frac{1}{2}C \cdot V^2$

The energy in the buffer correlates to its voltage (V_supp) (see Equation 4.12). There are two interesting values:

- When the buffer voltage reaches the input requirements of the voltage regulator, this will boot the ULP MCU. By low MCU power requirements, chances are good that the harvesting income grants stable operation. The PVC must be dimensioned so that this point is easily reached in mediocre light conditions.
- When the buffer reaches a pre-determined higher voltage, there is enough energy stored for a single peripheral task. As the buffer cannot exceed the nominal voltage of the PVC, the PVC has to be chosen accordingly.

Thus, the objective of the controlling circuit is to keep the energy buffer above the minimum input requirements of the voltage regulator (usually slightly above V_{DD}), and to detect buffer voltages that are high enough to perform periphery tasks.

delayed interrupt triggering over an RC circuit

To gain voltage information, the VFC circuit charges capacitor C over resistor R, which leads to a charging time that depends on their respective (constant) values and V_{supp}. If the

capacitor reaches a voltage above the threshold voltage of the IRQ pin, it will trigger an *interrupt service routine* (ISR).

The software reacts to the signal by resetting the capacitor using transistor T which is connected to an output pin. Additionally, the software can decide to do energy intensive workloads. This decision is not trivial: As any supply voltage above V_{thr} will cause interrupts, but the interrupt frequency is dependent on V_{supp}. Thus, the interrupt frequency must be counted.

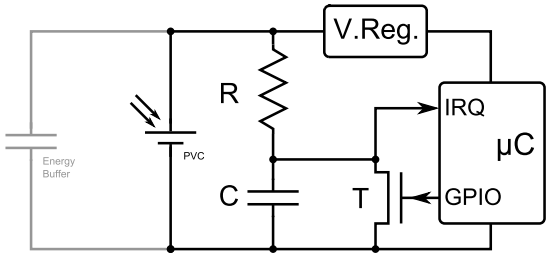


Figure 5.2: A simple adaptive WSN node with software aided voltage to frequency conversion.

The correlation of interrupt frequency and V_{supp} can identified by considerations:

With τ being a time constant as calculated in Equation 5.1, the capacitor C will be charged according to Equation 5.2. τ denotes the time that is required to raise the voltage of the capacitor by 63.2%. Thus, 5τ is needed to charge a capacitor to 99.3% or to discharge it to 0.7%. For this reason, 5τ is commonly used as charge or discharge time of a capacitor in an *RC circuit*.

If C has a low capacitance, V_{supp} can be considered constant, as it can be assumed that the rate at which a PVC significantly changes its output voltage is by magnitudes lower than the charge time of the used RC chain.

$$\tau = R \cdot C \tag{5.1}$$

$$V_c(t) = V_{supp} \cdot (1 - e^{-\frac{t}{\tau}}) \tag{5.2}$$

With V_{thr} being the threshold voltage of the IRQ pin of the MCU (i.e., the voltage at which a signal flank is detected), the time to raise a low-high flank starting from a depleted capacitor is calculated in Equation 5.3.

$$\begin{aligned} V_{thr} &= V_{supp} \cdot (1 - e^{-\frac{t_{thr}}{\tau}}) \\ \Leftrightarrow \quad e^{-\frac{t_{thr}}{\tau}} &= 1 - \frac{V_{thr}}{V_{supp}} \\ \Leftrightarrow \quad -\frac{t_{thr}}{\tau} &= ln(1 - \frac{V_{thr}}{V_{supp}}) \\ \Leftrightarrow \quad t_{thr} &= -\tau \cdot ln(\frac{V_{supp} - V_{thr}}{V_{supp}}) \end{aligned} \tag{5.3}$$

Transistor T is used to reset the capacitor. This circuit requires an interrupt service routine, triggered by the IRQ signal, to actively switch T. Assuming that this happens after t_{sw}, the interrupt frequency can be calculated as shown in Equation 5.4

$$f_{irq} = \frac{1}{t_{thr} + t_{sw}} \tag{5.4}$$

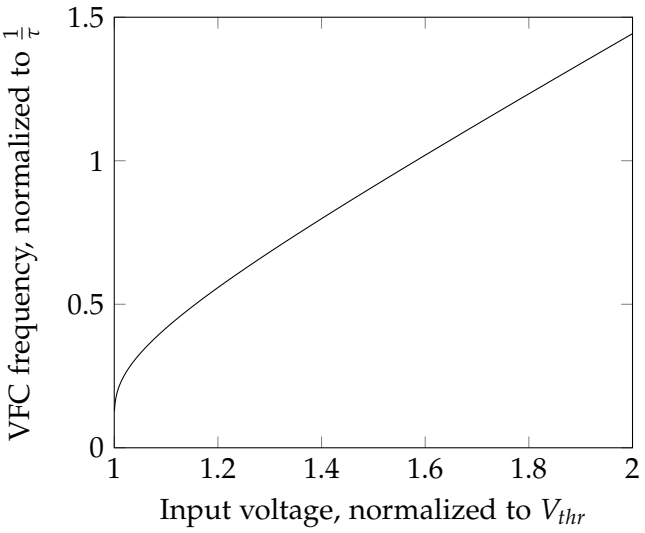


Figure 5.3: Frequency output of the VFC

Figure 5.3 shows the calculated frequency output of the VFC at $t_{sw} = 0$. The graph shows that the output frequency becomes nearly linear at voltages above $1.1 \cdot V_{thr}$.

This leads to the following methodology for choosing components for the VFC:

1. Choose a harvester with a nominal output voltage of at least the voltage regulator's minimum input voltage.
2. Determine the typical latency t_{sw} of the interrupt service routine.
3. Choose an interrupt frequency for a given voltage, e.g. at the harvester's maximum output voltage.

4. Calculate t_{thr} (Equation 5.4) and τ (Equation 5.1). V_{thr} is a parameter of the used MCU and usually ranges at 50% of its supply voltage.

5. Choose a capacitor with a capacitance multiple times greater than the input capacitance C_{int} of IRQ line, which is usually somewhere in the nF or pF range.

6. Choose R according to Equation 5.1.

5.2.2 *Removing Transistor T*

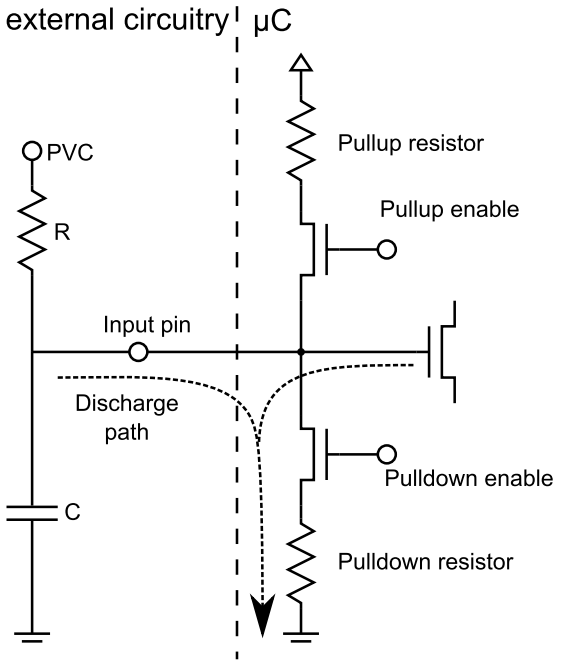


Figure 5.4: Using an internal pull-down resistor to discharge a capacitor.

The basic VFC circuit consists of a resistor, a capacitor and a transistor. The transistor is necessary to discharge the capacitor C and so to reset the circuit. As most modern MCUs have highly configurable IO pins, there is a high probability that switchable pull-up and pull-down resistors are available on the device.

These resistors serve the need to hold input lines in a defined state when not actively driven. Figure 5.4 shows the typical construction of an input line within the MCU in a simplified schematic. By activating the pull-down resistor (usually in the

range of 10 kΩ, a path to discharge the capacitor can be established without the need for an external transistor. Anyhow, this concept is a compromise, as the capacitor cannot be fully discharged and discharging requires additional time. Also, there is a current flowing from the supply through the pull-down resistor to ground, imposing an energy leak.

Depending on the value of R_{pull}, the discharge time for C, as calculated in Equation 5.5, becomes considerable. The pull-down resistor must be active for at least $t_{disch} + t_{sw}$.

$$t_{disch} = 5 \cdot R_{pull} \cdot C \tag{5.5}$$

The lowest achievable voltage at the capacitor is determined by the voltage divider created of R and R_{pull}, which calculates according to Equation 5.6. Thus, the value of R must be chosen high enough so that $V_{C,min}$ is less than the threshold voltage, or the capacitor is unable to trigger a flank.

$$V_{C,min} = \frac{R_{pull}}{R + R_{pull}} \cdot V_{supp} \tag{5.6}$$

The leakage current that occurs while clearing the capacitor is calculated in Equation 5.7 according to Ohm's law.

$$I_{leak} = \frac{V_{supp}}{R + R_{pull}} \tag{5.7}$$

5.2.3 *Removing Capacitor C*

As an input pin of an MCU is usually connected to the gate contacts of MOS transistors, it behaves like a (very small) capacitor. The internal capacitance, C_{int}, is usually within the pF to nF range. The transistor switches when enough charge was collected within the gate electrode to establish an electrical field throughout the substrate of the transistor. To switch back, this charge must be removed.

Thus, the capacitance for the VFC calculation is actually $C = C + C_{int}$, as both the external and internal capacitors are charged in parallel. C_{int} was omitted as it is negligible if the external capacitor was chosen correctly.

Alternatively, the external capacitor can be completely removed, so that only C_{int} is used. However, this might require to choose R as a high-ohmic resistor within the MΩ range to increase τ so that an acceptable interrupt frequency can be achieved.

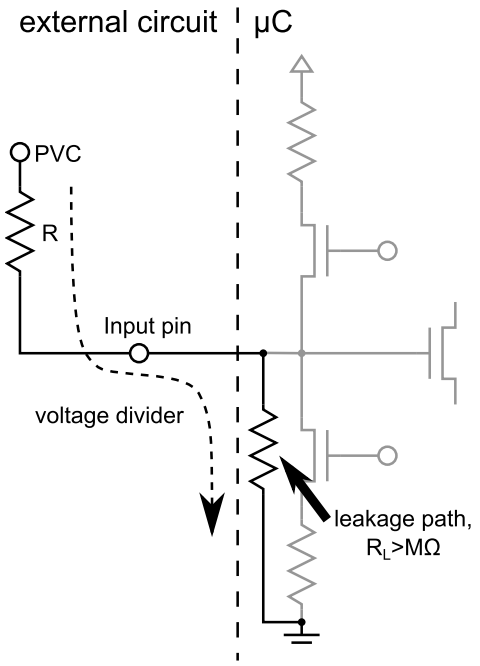


Figure 5.5: Voltage drop over R caused by leakage current when $R \sim R_L$.

However, there is an upper limit to the resistor that can be chosen here: As actual hardware is not ideal, the internal conductance of the input pin towards ground is not zero (there is a leakage current through the substrate causing a finite resistance). Figure 5.5 shows this as a fictive resistor R_L. The serialization of R and R_L form a voltage divider, and if R is in the same range as R_L, the voltage at R_L (and thus C_{int}) will drop.

This effect will not only change the resulting interrupt frequency due to the voltage drop, but might also prevent the activation of the interrupt at all if the voltage at R_L never rises above V_{thr}. In consequence, the parasitic voltage divider shown in Figure 5.5 limits the minimum interrupt frequency, as R cannot become arbitrarily high.

Another way to reduce the interrupt frequency is to reduce the charging voltage of C_{int} to a value just above V_{thr}. For this purpose, a z-diode can be used. However, it is questionable if it makes sense to substitute an external capacitor by a z-diode. This decision depends on the design goal, price, size and leakage current of the respective components.

5.2.4 *Flank Duration Measurement*

Interrupt frequency counting requires an appropriate amount of events to estimate the frequency, so that multiple interrupts must occur to derive a decision about peripheral tasks.

For MCUs equipped with appropriate timers and capture/-compare functionality, it can be beneficial to measure the flank time t_{thr} (i.e., the time from discharging until reaching V_{thr}) directly at each interrupt.

For this purpose, the VFC signal must be connected to a capture input of a timer/counter unit that is configured for an appropriate counting frequency to measure the flank time. When the VFC triggers the capture input of this unit, the current counter value is frozen and an ISR is started. The captured value can be read and the counter can be reset after discharging the capacitor.

This approach has its distinct advantages and disadvantages compared to interrupt frequency counting: The main advantage is that a single hardware interrupt already yields a measurement result, which reduces the computational overhead for interrupt handling. The main disadvantage is the requirement for a high timer resolution and the need for a capture unit.

5.3 LAB PROTOTYPE

This section introduces a microcontroller setup for lab evaluation purposes. In contrast to a full functional prototype, the prototype shown here uses simulative hardware elements to create reproducible conditions.

The prototype is based on TI's MSP430FR5969 "Launchpad" experimentation board. Its super-capacitor was deactivated by removing the respective jumpers. Likewise, the connection to its on-board programmer/debugger hardware was completely cut to remove leakage currents to this component. As a consequence, the MCU remains the only active part.

The node is intended to be driven by a PVC, but to create reproducible lab conditions with a known energy income, a PVC simulation was used. A prototype schematic is depicted in Figure 5.6.

The simulated PVC is set up to imitate the behavior of a *Solems 07/072/048* indoor cell. It delivers a nominal voltage of 4 V. A voltage regulator supplies the MCU and further periphery with 3 V. Although it is possible to drive the MCU at lower

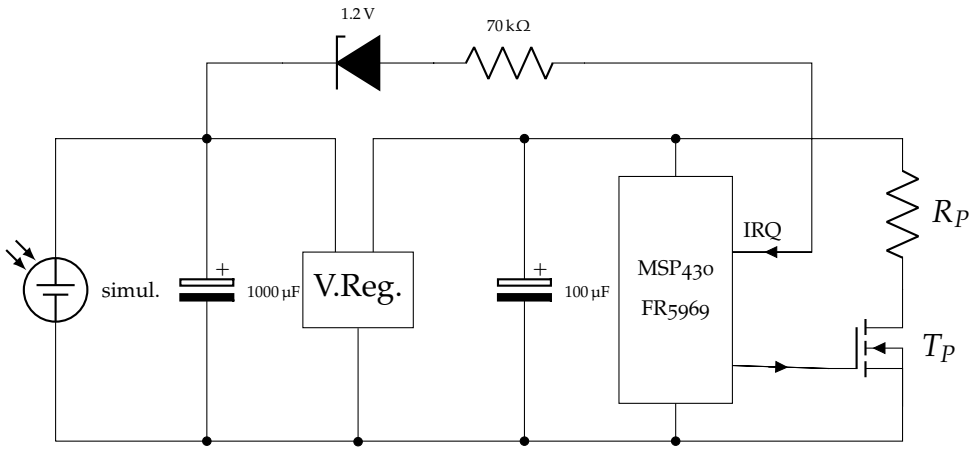


Figure 5.6: Lab prototype schematic

voltages, the intention to simulate the use of high power periphery, e.g., a radio transceiver, called for higher voltages.

The periphery current consumption is simulated by the ohmic resistor R_P that can be switched on and off by transistor T_P. The MCU controls the periphery by an I/O signal.

definition of workload

The task of the node is to drive a peripheral workload. The term *workload* is used in the following text to describe the utilization of an external periphery component, e.g. a radio transceiver, with a known energy consumption and a limited time window.

Failing power during peripheral access leads to incomplete workloads. In this case, the voltage has dropped to a level that renders the system inoperative. The objective of the prototype is to maximize the number of complete workloads, while reducing the occurrence of incomplete workloads.

Definition:Undervoltage, incomplete workload

A workload is considered as incomplete, if at any time during the active workload the regulator output voltage of 3 V drops by 10% to 2.7 V.

5.3.1 *Periphery*

The periphery is simulated by resistor R_P and transistor T_p. R_P was dimensioned so that the whole MCU circuit creates a load of 16 mA at 3 V when the MCU is active at 8 MHz.

A single workload is performed by closing T_1 for 10 ms. This leads to an energy consumption of 480 µJ.

5.3.2 *Voltage Regulator*

The voltage regulator supplies the output voltage for the digital circuit. A high efficiency device (e.g., switching regulator) must be chosen.

Some regulator types require a high switch-on current. These types are inappropriate for the purpose, as they are prone to fail starting the MCU in transient power supply setups. Generally, types that start at lower input voltages than the nominal output voltage are more suitable. These types deliver $V_{out} \approx V_{in}$ for voltages below their nominal output voltage.

Equation 5.8 is used to calculate the regulator's input current at full charge (neglecting the loss of the regulator). With 16 mA output current at 3 V, the regulator requires an input current of 12 mA at 4 V (the nominal voltage of the PVC).

$$I_{in} = \frac{V_{out}}{V_{in}} \cdot I_{out} = \frac{3\,\mathrm{V}}{4\,\mathrm{V}} \cdot 16\,\mathrm{mA} = 12\,\mathrm{mA} \tag{5.8}$$

On the output side of the voltage regulator, a small amount of charge (100 µF) is buffered, as the voltage regulator requires a capacitive load according to its data sheet.

5.3.3 *Energy Buffer*

As it is necessary to collect energy for a workload, an energy buffer is needed. It must be dimensioned correctly in order to prevent it from low voltage during active workloads.

As the minimum acceptable voltage for the workload is 2.7 V, the capacitor must be selected so that a voltage drop from 4 V to 2.7 V takes at least 10 ms for the given power consumption.

To calculate the required capacitance, the impedance of the input side of the regulator is determined by Equation 5.9. The resulting value is used for the calculation of C according to Equation 5.10, which describes the discharge curve of a capacitor. With $t = 10\,\mathrm{ms}$, $R = 333\,\Omega$, $V_0 = 4\,\mathrm{V}$ and $V(t) = 2.7\,\mathrm{V}$ this results to 764 µF.

$$R = \frac{V_{PVC}}{I} = \frac{4\,\mathrm{V}}{12\,\mathrm{mA}} = 333\,\Omega \tag{5.9}$$

$$V(t) = V_0 \cdot e^{-\frac{t}{RC}} \Rightarrow C = -\frac{t}{R \cdot ln(V(t)/V_0)} \tag{5.10}$$

A 1000 µF electrolyte capacitor was used to have some reserves for energy losses in the capacitor and the regulator, as

they are not ideal parts in practice. As capacitances of this size are available as electrolyte capacitors, it is not necessary to use a super cap with limited lifetime and voltage range.

There is one further important aspect for dimensioning the capacitance: Buffers cannot be chosen arbitrarily high, because each buffer imposes a time delay between the harvester and the VFC. If this delay becomes too large, the system might not be able to adapt its load to the maximum-power point (MPP) of the PVC. Since the energy transfer from the PVC to the buffers cannot be controlled in any way, it must be assured that energy transfers from the buffers to the system will influence the PVC output current without longer delays. Thus, increasing the buffers reduces the efficiency of the PVC.

5.3.4 *PVC Simulation*

Purpose: Reproducibility

The energy output of a PVC is highly sensitive to external parameters like the light intensity, light temperature, radiation angle, and environment temperature. This makes it hard to reproduce a changing light profile that was used during experimentation.

This problem was solved by surrogating the PVC by a circuit with a controllable behavior. The surrogate shows similar voltage and current properties as an actual PVC. It is driven by a current source with a defined output current I_{illum} that represents the illumination of the PVC. With a programmable current source, light profiles like dusk or dawn phases can be simulated.

programmable illumination sequences

To understand the concept of the surrogate, it is important to understand the electrical properties of a PVC first. Without load, a PVC still can supply a nominal voltage even in the dark by thermal electrons. The nominal voltage is a property of the space charge zone of the semiconductor materials used for the construction of the cell.

However, the amount of electrical current available at the PVC is limited by the available light. As soon as this available current is exhausted, the voltage of the PVC drops, as the space charge zone disappears. Figure 5.7 depicts the voltage vs. current behavior (and the corresponding power output) of a PVC in two different light conditions. With power being the product of voltage and current, the maximum power output can be achieved by consuming a current close to the point where the voltage drop occurs.

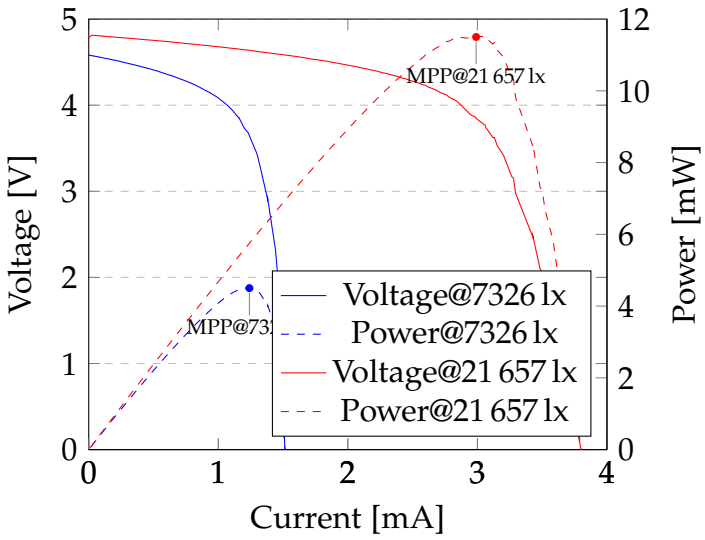


Figure 5.7: Current vs. voltage and power for two different light situations, as measured from a Solems 07/072/048 indoor PVC

Because here, electrical current is dependent on the consumer of electric energy, it is important to constantly monitor the voltage of the PVC and to drive the current consumption close to the maximum power point. For this purpose, maximum-power-point tracker (MPPT) circuits are able charge energy storage devices with excess energy.

Without energy storage (or with very small storage capacities as shown here), the consumer has to adapt its workload to the energy income.

The electrical output of a PVC can be reproduced by the circuitry shown in Figure 5.8 [22, 35, 83].

nominal voltage

This setup allows for setting the nominal voltage (that is, the voltage in zero-load conditions) by choosing the number of diodes. A silicon diode in forward configuration has a constant voltage drop ("forward voltage") of around 0.68 V. We used six diodes in series to achieve a voltage of ≈4 V. If no consumer is present at the surrogate (no current output), the current supply has to regulate its output voltage to the forward voltage to drive I_{illum} through the diodes. This defines the nominal voltage.

A device connected to the output "steals" current from the diodes. The voltage of the current source has to drop for keeping I_{illum} at the preset value, as it requires less voltage to transport electrical energy through a demanding consumer than keeping up the diodes' forward voltage. As soon as the forward voltage is undercut, the non-linear shut-down behav-

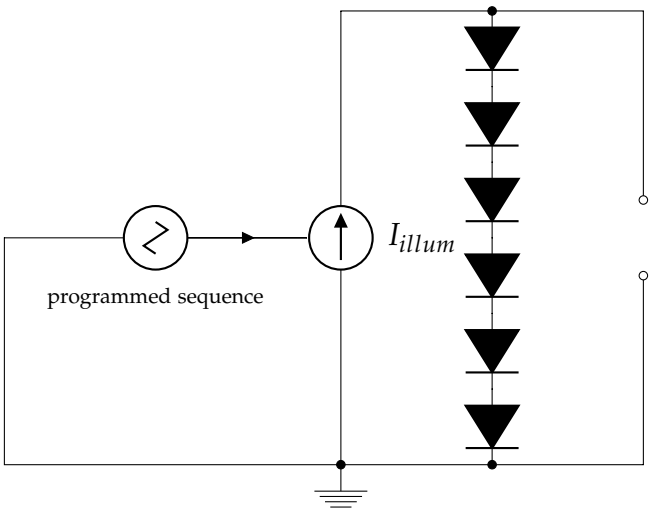


Figure 5.8: A diode series driven by a programmable current source simulates the voltage and current behavior of a photovoltaic cell.

ior of the diodes form the voltage drop that is typical also for PVCs.

To determine the available energy for any I_{illum} setting (light situation), the V/I curves for multiple I_{illum} were measured and the respective maximum power points were determined. The results are shown in Figure 5.9. The curve function was estimated by an exponential regression as shown in Equation 5.11 with a root mean square error (RMSE) of $7.39 \cdot 10^{-8} W^2$.

$$P_{max}(i) = a \cdot e^{bi} + c \tag{5.11}$$

When experimenting with changing light conditions (by changing I_{illum}), the maximum available energy can be determined by Equation 5.12. This is important for calculating the efficiency of an approach, described by the ratio of the actually used energy and the available energy.

$$E_{avail} = \int P_{max}(I_{illum}(t))\,\mathrm{d}t \tag{5.12}$$

available energy: 11.7 J

For the evaluation experiment, the PVC simulation was set up to deliver a simulated light profile with a maximum PVC output of 20 mA, as shown in Figure 5.10. The profile raises the available current (I_{illum}) from 0 to 20 mA during a 2 minute dawn phase. This current is held for 1 minute, and then a 2 minute dusk phase lowers the current back to 0 mA.

The available energy in this experiment setup is 11.7 J.

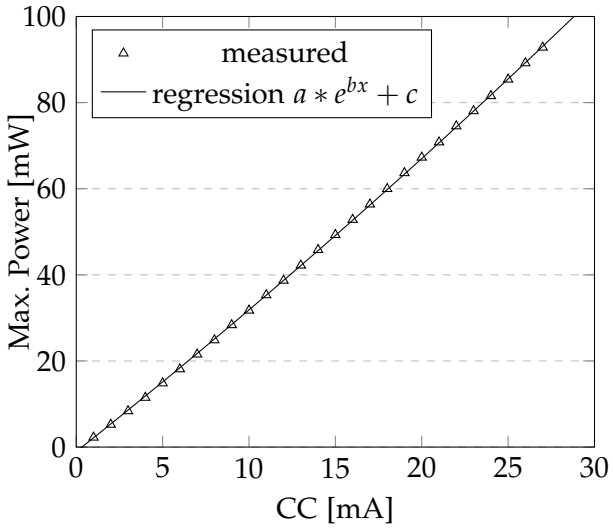


Figure 5.9: Maximum power point measurements and regression model for the simulated PVC, with $a = 0.4775, b = 6.6633, c = -0.4786$

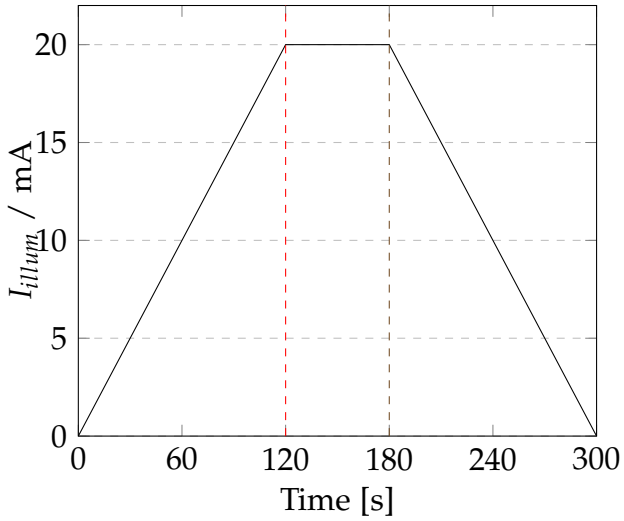


Figure 5.10: Simulated light profile

5.3.5 *Software*

The flank time of the VFC signal is measured by using the capture/counter functionality of the MCU. This peripheral feature allows to automatically capture the value of a hardware counter into a shadow register, where it can later be accessed by software. Capturing is triggered by an external signal on an I/O pin.

For the counter, a 1 MHz on-chip clock source was used. The value of the capture register is proportional to the flank time (more exactly, it is the number of clock cycles until the threshold voltage was reached).

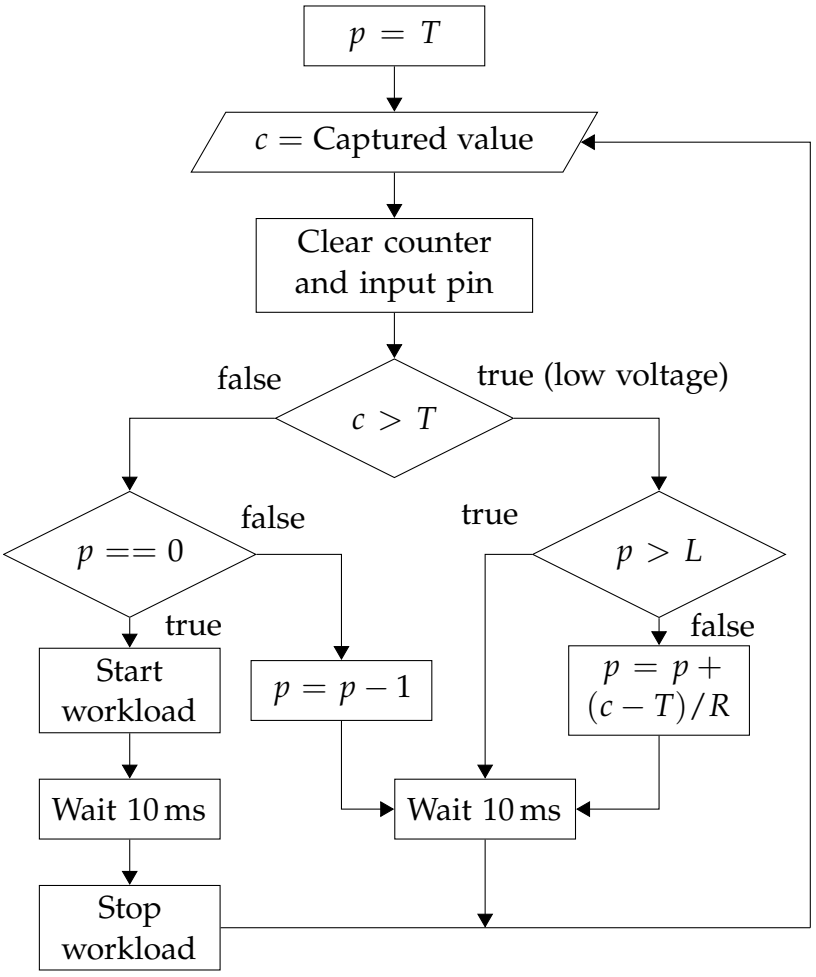


Figure 5.11: Flow chart of the charge regulator.

The algorithm used for controlling the workload segregates time into 10 ms time slices. For every time slice, it decides whether to start a workload (active workload transistor for 10 ms) or to pause for that time, as can be seen in Figure 5.11. It

uses a threshold value T to distinguish if the captured counter value, c, denotes a low voltage condition.

A "good voltage" condition does not necessarily mean that a workload can be started. Due to the charging curve of a capacitor ($E = CV^2$), it charges the most energy near the supply voltage. By that, the measurement resolution for energy decreases with rising voltage.

To ensure sufficient energy supply before starting energy intensive tasks, the amount of time slices required for fully charging is estimated. A pause counter (p) is increased when the captured timer value is in the low-voltage area. As it has a higher resolution in this range, the increment of p can be calculated in dependency of the measured voltage (respectively the captured value c). A conversion constant R is used for this purpose. p can be incremented multiple times, once on every consecutive low-voltage time slice.

The idea here is to use p for pausing the workloads while charging the capacitor. Low voltages increment p fast, and multiple, consequent low-voltage conditions occur on low energy harvest, increasing p even more. Thus, slow voltage increments lead to long charging pauses even after good voltage conditions are met.

In contrast, at high energy yield, low voltage conditions are left quickly, as the capacitor voltage ramps up quicker. In this situation, short pauses help to avoid energy loss by pausing while the capacitor is fully charged.

To avoid p being excessively incremented when the system stays in low-voltage conditions for a prolonged period of time, e.g. by a lack of illumination, further increments are omitted when a limit of L is reached.

In good voltage conditions, p is decremented each time slice until it reaches 0. This gives the capacitor enough time to charge. Then, a workload is started. The algorithm starts multiple consequent workloads if there is enough energy harvested to keep the capacitor above the good-voltage threshold.

The behavior of the algorithm can be tuned by three parameters: A *threshold* value T that defines an upper limit for the capture value to be considered as *good*, a *recover* value R that defines the increase in pause time if bad voltage conditions occur, and a *limit* value L that limits the pause time value.

The voltage-good threshold T can be set to the counter value that is measured when the buffer voltage reaches the supply requirements for the MCU.

L is the upper limit of idle time slices after reaching the good-voltage threshold. This value can be arbitrarily high, but must at least avoid integer overflows. Also, there can be a demand to further limit L, as it increases reaction latency after dark periods. The lower limit to L is determined by an assumed low-light scenario with just enough power income to continuously charge the capacitor (i.e., more power income than consumed by MCU operation without periphery). L must be higher than the number of time slices that is required in this situation for fully charging the capacitor to the PVC nominal voltage after the power-good threshold was reached.

$0.7 \cdot V_{supp}$ is the voltage at 50 % charge

R is a recovery factor that increases p depending on the measured buffer charge. The value of R depends on the counter frequency, buffer capacitance and PVC properties. As an estimate, let X_{50} be the counter value at $0.7 \cdot V_{supp}$, τ the capacitive (RC chain) time constant, and t_s the duration of a time slice, then chose R as in Equation 5.13

$$R = X_{50} \cdot (3\tau / t_s) \tag{5.13}$$

5.4 EVALUATION

To evaluate the quality of the prototype, two efficiency measures are used.

MPP EFFICIENCY: This measure describes, how efficient the available output energy of the PVC was used. To be efficient, a PVC must be kept near the MPP, which is dependent on the illumination. Otherwise, potentially available energy is wasted. An MPPT device is usually responsible to conduct exhaustive energy to a battery charger. As an MPPT unit was omitted, any surplus of energy should be used to charge the capacitor and to increase the peripheral workload frequency. The MPP efficiency is a measure for this aspect of the prototype. It is calculated as in shown in Equation 5.14.

PM EFFICIENCY: The power management efficiency describes, how much of the available energy was used for complete workloads. Energy can be wasted by incomplete workloads, by inefficient implementations, and by bad power management decisions. As the used energy per workload is constant, a workload count N can be used to determine the efficiency, as shown in Equation 5.15.

With $E_{avail} = 11.7\,\mathrm{J}$, given by the PVC surrogate setup, and a workload energy of 480 µJ (by a workload activation of 10 ms), the upper limit of completed workloads N_{max} is 23636, as calculated in Equation 5.16.

$$\eta_{MPP} = \frac{E_{used}}{E_{avail}} \tag{5.14}$$

$$\eta_{PM} = \frac{E_{comp}}{E_{avail}} = \frac{N_{comp}}{N_{max}} \tag{5.15}$$

$$N_{max} = \frac{E_{avail}}{E_{workload}} = \frac{11.7\,\mathrm{J}}{480\,\mu\mathrm{J}} = 23636 \tag{5.16}$$

5.4.1 *Evaluation Setup*

To determine E_{used}, energy measurements were performed at the output of the PVC simulator. Figure 5.12 shows the installation of a voltmeter (V_{MPP}) and an ampere meter (I_{MPP}) for this purpose. To count N_{comp}, the number of completed workloads, a flank detector was installed at the transistor gate of the artificial periphery load. In combination with volt meter V_{supp}, complete and incomplete workloads can be distinguished, with the latter being any workload with voltage drops below 2.7 V during 10 ms after flank detection.

While running the light profile given in Section 5.3.4, all meter values were continuously captured, and consumed power and energy (E_{used}) was calculated.

The create a base for comparisons and discussion, the experiment was evaluated with different software setups:

adaptive: This setup uses the adaptive behavior described in the above sections.

static-100: This setup continuously starts workloads every 10 ms, trying to utilize the system at 100% load.

static-50: This setup continuously starts workloads every 20 ms, trying to achieve an utilization of 50%.

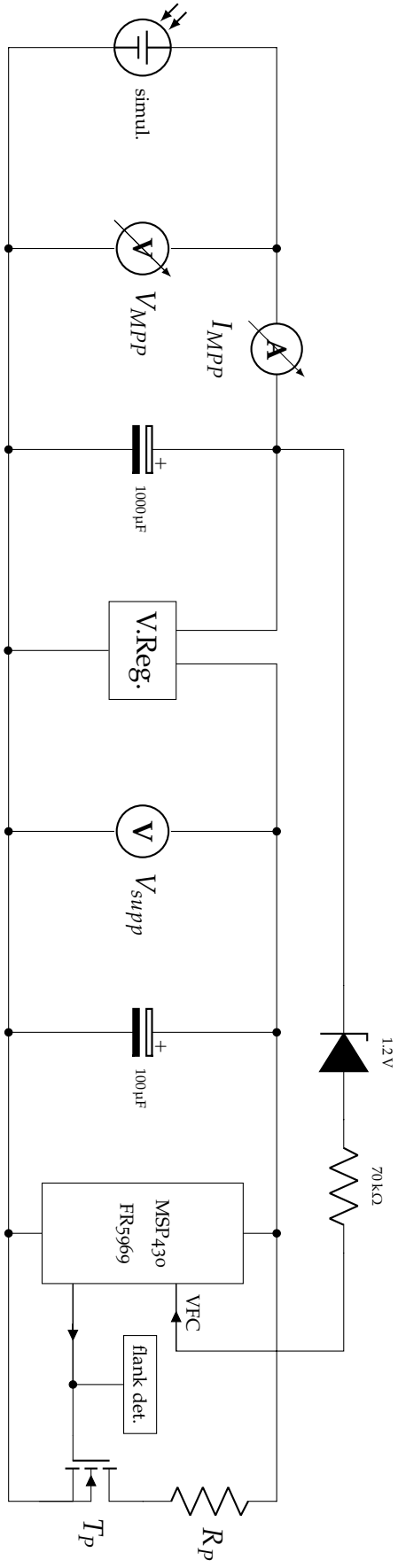


Figure 5.12: Lab prototype schematic, extended by volt and ampere meters and a flank detector.

		adaptive	static-100	static-50
complete	dawn	4683	2524	3512
	dusk	5840	3170	3573
	day	5852	5975	2971
	sum	16375	11669	10056
incomplete	dawn	22	8332	2210
	dusk	40	8303	2197
	day	0	0	0
	sum	62	16635	4407
	sum	16437	28304	14436

Table 5.2: Workload counters

	adaptive	static-100	static-50
E_{used}	10.59 J	10.29 J	7.53 J
η_{MPP}	0.905	0.879	0.644
η_{PM}	0.693	0.493	0.425

Table 5.3: Energy consumption and efficiency

5.4.2 *Results*

To evaluate the performance of this approach, the results were split into the three daylight phases (dawn, day, dusk). This revealed the adaptation behavior during changing conditions (dawn, dusk) and constant maximum utilization during the day phase, where the energetic demands of the system are fully satisfied.

Table 5.2 lists the workload counters for all three experiments. The adaptive solution is able to generate 40.3% more complete workloads than static-100, and even 62.8% more than static-50 over the time of the experiment.

The results show clearly that statically scheduled workloads can only perform in static environments: During daytime, the *static-100* performs best, actually 2% better than the adaptive solution. Anyhow, *static-100* shows the worst results for dusk and dawn periods.

In comparison, *static-50* performs better than *static-100* during the periods of changing light, but per definition can only work 50% of the available time during day time.

Even more convincing is the reduction of incomplete workloads: *Adaptive* generates just 62 incomplete workloads, while *static-100* produces 16635 and *static-50* produces 4407 low-voltage conditions. Thus, the adaptive approach heavily reduced the start of high-energy operations that can cause system reboots.

Taking the theoretical maximum of 23636 complete workloads as a measure, *adaptive* reaches an efficiency of 69.3% (vs. 49.3% of the second best result, *static-100*).

When looking at the energy consumption in Table 5.3, adaptive was able to make use of 90.4% of the available energy. As stated before, this is not a good measure for overall energy efficiency, but a measure for the ability to track the maximum power point of a PVC.

Surprisingly, maximum power consumption during the whole experiment already yields acceptable η_{MPP} results, as *static-100* performs well here at 88%. Anyhow, the energy taken by *static-100* during the dawn and dusk phases was mainly used to produce incomplete workloads caused by low-voltage conditions.

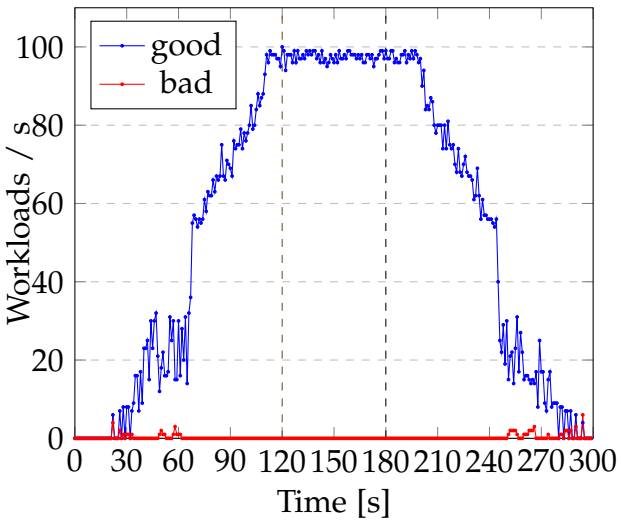
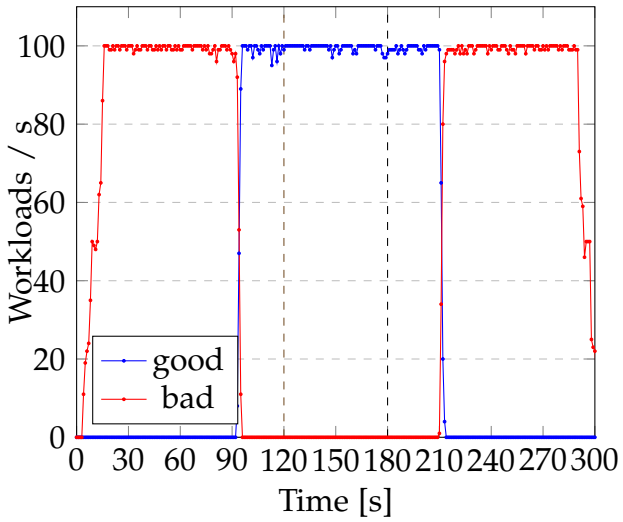
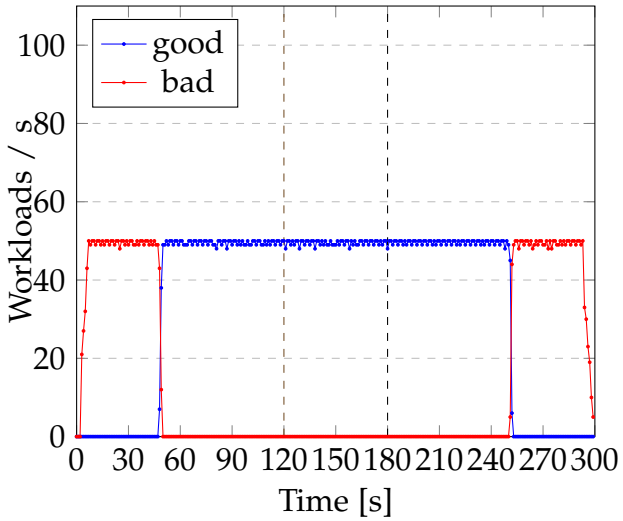


Figure 5.13: Workload behavior of the adaptive solution.

When analyzing the behavior of the three solutions over time, it can be observed that *adaptive* (Figure 5.13) continuously adapts through the dusk/dawn phases (with a little delay at the start of the dawn phase), and that it is able to reach its



(a) Workload behavior of static-100



(b) Workload behavior of static-50

Figure 5.14: Workload behavior of the static solutions.

utilization maximum at the day phase. Incomplete workloads occur rarely, and this only happens in the lower halves of the dawn/dusk phases.

As expected, *static-100* (Figure 5.14a) and *static-50* (Figure 5.14b) drain too much energy in the dawn and dusk phases, and can only create incomplete workloads until a certain energy-income is reached. Both solutions are able to reach their respective utilization maximum at the day phase. As expected, *static-50* will stay longer in its maximum utilization period, as it does not overload the system in the high-power halves of the dawn and dusk phases.

	Balsamo et al. [6]	*Brunelli et al. [14]*	*Lee et al. [43]*	*Gomez et al. [33]*	*This approach*
no hw-MPPT	n/a	-	+	-	+
no V-conv.	n/a	-	+	n/a	-
storage type	none	supercap	none	optional	electrolyte capacitor
periphery	-	n/a	-	unknown	+
η_{MPP}	n/a	80%	n/a	n/a	90.5%
η_{PM}	n/a	n/a	n/a	68.82%	69.3%
add. parts	comparat. ref.voltage-source	$\leq$€50	comparat. flip-flops, ref.voltage	buck-boost-converter, "control circuit"	$\leq$€1, (resistor, diode, el. capacitor)

Table 5.4: Comparison of different solutions for transiently powered systems.

A direct comparison with state-of-the-art solutions proposed in recent papers is given in Table 5.4. As far as comparisons are applicable, other shown approaches either have no efficiency measures specified or show lower efficiency. At the same time, the approach shown in this chapter has a minimal hardware overhead, as it only requires a DC-DC converter (the voltage regulator), and a few standard components (resistor, capacitors, diode).

5.5 REAL-COMPONENTS EXPERIMENT

This section describes the deployment of two sensor nodes using an actual PVC (Solems 07/072/048) as power source, and a radio transceiver (Nordic Semiconductor nRF24L01) as load. Additionally, each node is equipped with a MAX44009 ambient light sensor. The nodes are based on TI EXP-MSP430FR5969 Launchpads.

5.5.1 *Setup*

To test the systems under indoor conditions, the PVCs and light sensors were placed under an artificial light source. The light source then was randomly dimmed up and down to generate different light conditions. Low light conditions were kept up for longer periods of time, so that there was a chance to count a sufficient of workloads under low energy conditions for an evaluation.

Both nodes have the workload task to acquire an ambient light value from the sensor and to transmit it on a radio channel. One node starts workloads continuously in a loop, as fast as possible. The other one uses the adaptive approach and requires just two additional electronic components: The z-diode and resistor already used for the testbed setup in Section 5.4.1.

Receivers were placed in line of sight with the transmitters at a distance of 6 m. The maximum achievable range at best illumination was 7 m. On the receiver side, all transmissions on both radio channels were logged.

The dim light causes low voltage conditions on the static setup. The output power of the respective transmitter is expected to fade away in low light conditions, leading to a reduced transmission radius until the transceiver finally fails. Eventually, the MCU is expected to fail due to low voltage supply.

For the adaptive approach, the voltage and transmission range is expected to be stable, while the number of radio packets sent will adapt to the light conditions.

5.5.2 *Results*

While this setup is not considered to deliver reproducible or reliable results, it supports the results shown in Section 5.4 and shows that this approach is feasible for actual sensor setups.

As shown in Figure 5.15, the adaptive solution yields a significant higher network packet count under low and middle light conditions. Only at good light conditions above 5000 lx, both solutions show a similar behavior. In totals, the adaptive node was able to transmit 2316 radio packets, while static was able to send 982 radio packets (42.4%). The packet count in high ambient light conditions seems unexpectedly low for both experiments. This is due to the fact light intensity was not uniformly distributed over time.

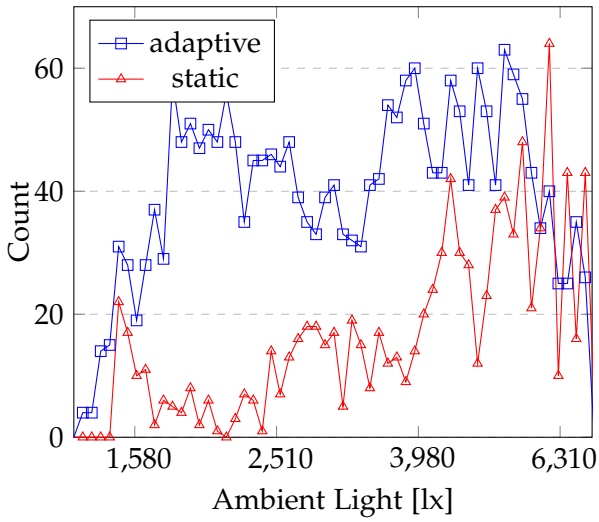


Figure 5.15: Parallel light measurements of a randomly changing light source as transmitted by both sensor nodes. The measured values were counted into 60 bins.

5.6 CHAPTER SUMMARY

This chapter proposed a methodology for creating an energy-neutral node by removing the rechargeable battery and related circuits, such as maximum power point trackers and battery charge controllers, for applications that have low constraints on their dependability.

With an energy efficiency of ≈70%, this approach is at par with other state-of-the-art approaches in energy-neutral system design, while also scaling the utilization of peripheral hardware (instead of just scaling CPU performance) and still achieving a lower part count and lower costs.

To achieve this within acceptable service quality bounds, an inexpensive hardware approach was introduced to estimate the energy that is available in a capacitor which is charged from an energy harvester, and to adapt the software behavior accordingly.

It was shown in detail that the required hardware parts for conserving and adapting to the available energy can be reduced to a standard capacitor for collecting energy, and a single resistor on an interrupt pin of the MCU to build up a voltage to frequency converter in conjunction with the internal capacitance of the interrupt pin.

In a prototype implementation with a single resistor and a z-diode connected to a basic MCU setup, a comparison to stati-

cally scheduled solutions was made. To this end, a testbed was build to simulate the electrical behavior of a photovoltaic cell during different periods of the day (dawn, daylight, dusk). The goal of the systems under test was to maximize the peripheral workload. Here, it was shown that the presented adaptive approach outperforms statically scheduled approaches. During the dawn and dusk phases, this approach generates between 33% and 63% more complete workloads compared to the best control experiment. During the day phase it generates just 2% less compared to the best static approach in this phase (which behaves worst during dawn and dusk).

Also, the number of low-voltage situations occurring during the experiments was counted. The adaptive approach shows a sum of 62 incomplete workloads during the whole experiment, in comparison to 4407 workloads for the best statically timed competitor and 16635 for the worst.

69% of the available energy was used to complete workloads, whereas the best static competitor only used 49% of the energy for this purpose.

To check if real deployments can benefit from the approach presented here, a more realistic indoor scenario was set up that used PVCs, light sensors and wireless transceivers on two concurring sensor nodes. It was shown that this approach was beneficial to a light sensing node in low and medium light situations. During the experiment, it was possible to transmit 2316 healthy network packets to a radio receiver. With a statically scheduled approach, only 982 packets were transmitted.

In conclusion, this approach retains the energy efficiency of state-of-the-art approaches, while heavily reducing hardware costs.

6

ENERGY MODELING OF EMBEDDED PERIPHERY

> “ Toolchains have been long focused on improving execution time, and developers are left assured that a compiler will do the best possible job on optimizing their code for both time and energy. The IoT energy challenge is an opportunity to start treating energy consumption as a first class citizen while developing software. ”
>
> *K. Georgiou, 2018 [32]*

6.1 INTRODUCTION

As shown in the previous chapters, the reduction of energy consumption is an important challenge in Industry 4.0, IoT, sensor networks and ubiquitous system scenarios. As these systems are usually not constructed as computational systems with a high algorithmic load, their sensor, actuator, and transmitter periphery becomes an important objective of power management.

However, to facilitate efficient use of energy, power management solutions need to be *aware* of energy income and consumption. Energy models help to map energy consumption to periphery tasks, system states, and software routines, and by that enable accurate predictions and intelligent decisions about periphery usage.

Definition: Energy awareness

The term *awareness*, as used in this chapter, denotes the ability of software modules to know and describe their own non-functional properties to other software layers. This is a notable contrast to the widespread understanding of “energy awareness” as a synonym for “energy saving”.

RELATED PUBLICATIONS

The findings presented in this chapter have partly been published in:

[18] M. Buschhoff, R. Falkenberg, and O. Spinczyk. "Energy-Aware Device Drivers for Embedded Operating Systems." In: *SIGBED Rev.* (2019). to appear.

[19] M. Buschhoff, D. Friesel, and O. Spinczyk. "Energy Models in the Loop." In: *Procedia Comput. Sci.* 130.C (May 2018), pp. 1063–1068. ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.04.154. URL: https://doi.org/10.1016/j.procs.2018.04.154.

6.1.1 *Contribution*

This chapter will present a modeling scheme for periphery devices and adapt the scheme for modeling the average power consumption of different CPU modes. The resulting models yield a high accuracy can be efficiently used to generate periphery driver and scheduling support for the accounting of consumed energy.

The resulting device drivers will be used to automate the process of model refinement and driver generation in Chapter 7.

The approaches shown in this chapter were developed as sensor network nodes running on 16 bit MCUs. As the acceptable model complexity is limited, these approaches do not scale for devices with highly complex interfaces or functionalities, e.g., GPUs.

6.2 CONCEPT OF MODEL AND DEVICE DRIVER COHERENCE

A peripheral device has functional and non-functional properties. Functional properties define the purpose of the device and are available through hardware interfaces. Non-functional properties are a consequence of the hardware implementation, but do not (intentionally) serve the purpose. E.g., energy consumption, heat dissipation, size, weight, and latency are non-functional device properties.

When starting functional activities of periphery devices, they can have either a constant or a variable time component. A functionality with a variable time component may require a significant amount of time to complete. Often, completion is signaled by IRQs to enable efficient processing. A functionality with a constant time component may also signal completion, but can also rely on CPU timing. For many device functionalities, e.g.,

for setting configuration registers, the required time is short enough to be considered complete immediately.

As energy consumption is a consequence of functionality and time, it is reasonable to model variable-time functionalities by their average power consumption, while modeling constant-time functionalities by their average energy consumption.

A hardware model of such a component can consist of a FSM. All variable-time functionalities form the set of states of the FSM. The constant time functionalities and completion signals form transitions. Further transitions can be added to switch states as necessary.

A hardware driver creates a software interface to the hardware functionalities. It makes sure that all required functionalities become available as software routines. Additionally, there must be software routines to handle the diverse interrupt signals of the peripheral hardware.

This means, independently of the knowledge of an FSM model, a device driver developer incorporates all routines that resemble the transitions of such a model.

coherence of model and driver

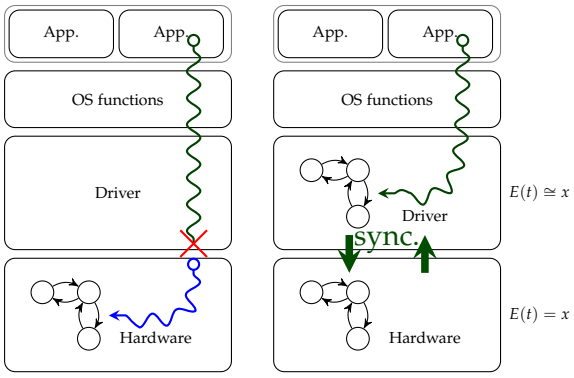
A hardware model and a software driver implemented independently are called *incoherent* in the further text, as the *syntax* of the software interface does not necessarily enable a mapping to a hardware model. This creates a *semantic gap*, as it is hard to transfer new or changed model properties (e.g., power consumption annotation) between model and code. Figure 6.1 illustrates this on the left side.

To achieve a *coherent* implementation, there must be a mapping between hardware events (transitions) and code locations. If this mapping occurs either as code annotations, automaton annotations or in a separate mapping structure is implementation specific. In the following text, model annotations to code locations are used.

Coherence closes the semantic gap, so that information can be transferred from model to code algorithmically.

6.3 AUTOMATA MODELS

The foundation of the proposed driver concept is an automaton model that describes a hardware component as a state machine. The approach shown here is restricted to variable-time functionalities that use IRQ signaling schemes, and immediate constant-time operations. For components requiring explicit timings, automata extensions like PTA models [4, 8] can be



(a) Incoherent driver

(b) Coherent driver

Figure 6.1: The classic implementation (left) showing the lack of coherence between the hardware and the software layer, while the right side closes the semantic gap and allows for synchronization of model data and implementation.

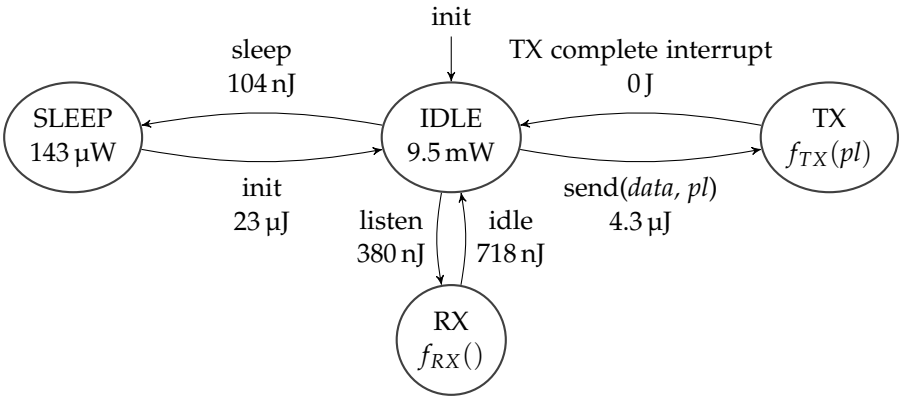


Figure 6.2: Partial model for a CC1200 transceiver. f_{RX} and f_{TX} are parameter specific and can be measured for a static device configuration (see Chapter 7).

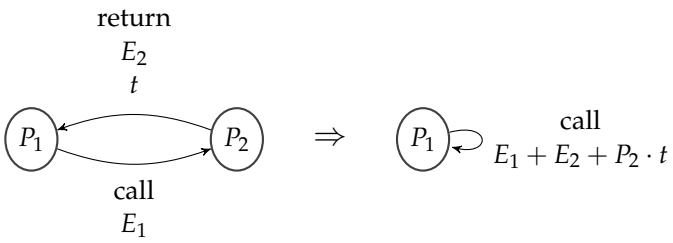


Figure 6.3: Removing a timed transition

used, but require a more complex software implementation and system timers.

Figure 6.2 shows an example of an automaton with energy annotations. As shown, transitions between $SLEEP$, $IDLE$, TX, and RX are handled explicitly, while the return path from TX to $IDLE$ is handled by an ISR.

The automata model uses three types of transitions:

1. *Software-triggered transitions* can be assigned to a code location, typically a subroutine/function, and may include parameters or other data required for utilizing the hardware.

2. *Interrupt-triggered transitions* can be assigned to interrupt service routines.

3. *Time-triggered transitions* can be assigned to a code location, typically a *stub* or the end of a subroutine/function. Time-triggered transitions are used when state changes occur within the device at a constant time after entering a state. These transitions can often be replaced, as constant timing leads to constant energy consumption (see below).

To reduce the model size, intermittent states that automatically transit back to the originating state by using a time-triggered transition can be replaced by a transition back to the originator if the energy consumption of the state is constant (Figure 6.3). This is a prevalent case for devices with standard actions, like single-value measurements or transmissions of a constant amount of data.

Formally, the used automaton is a tuple $\mathcal{A} = \langle S, S_0, \Sigma, L, T, N, E, P \rangle$ with the following components:

- S, the finite set of states
- $S_0 \in S$, the initial state
- Σ, the input alphabet (hardware events)
- L, the finite set of code locations
- $T : S \times \Sigma \to S$, the transition function
- $N : S \times \Sigma \to C$, code location assignments
- $E : S \times \Sigma \to \mathbb{R}_{\geq 0}$, the transition energy (in joules)

- $P : S \to \mathbb{R}_{\geq 0}$, the power consumption of states (in watts)

Optionally, the tuple can be extended by components of timed automata: $\mathcal{T} = \langle C, G, R \rangle$

- C is the finite set of clocks
- $G : S \times \Sigma \to \mathcal{B}(C)$ is the clock guard assignment, with $\mathcal{B}(C)$ being a set of constraints on clocks
- $R : S \times \Sigma \to \mathcal{P}(C)$ is the clock reset assignment, with $\mathcal{P}(C)$ being the power set of C

6.3.1 *Modeling CPU Energy Consumption*

Accurate energy models for CPUs are hardly available due to the complexity of these devices. However, for small scale (8 and 16 bit low power, single-core MCUs), reasonable results can be achieved by modeling the active mode and sleep modes as FSM. This allows to use the same modeling scheme for the CPU and for periphery devices.

To create coherence, code locations must be identified where mode changes occur. These are either imperative mode changes, or they may occur as a side-effect. Multiple imperative mode changes have to be refactored so that a single code location exists for model assignment. These transitions typically occur when the scheduler idles, i.e. when no process or task is currently runnable.

The MCU usually returns from sleep modes by interrupt requests, either issued by an internal timer or by a peripheral device. In many operating systems, all interrupt requests are handled by a central handler function that distributes the control flow to designated device handlers, so that a single location typically exists.

Depending on the used programming language, CPU related management functions can be segregated into a distinct class or module to resemble the structure of a driver. This eases the process of instrumentation, as CPU related code locations can be treated in the same way as periphery related locations. As an example, imperative transitions to idle and a general IRQ handler stub can be created in a CPU class, just as device driver specific functions and IRQ handlers would reside in a driver class.

6.4 DRIVER INSTRUMENTATION

In the following, the term *instrumentation* is used to describe the process of extending driver implementations by model specific interfaces and functionalities.

In the approach discussed here, the Kratos operating system [16] was used. Kratos is written in C++ and employs the aspect-oriented language extension *AspectC++* [69, 70]. Device drivers are encapsulated each in an own class, and public methods form the *application programming interface* (API) of the driver. Additionally, drivers inherit virtual functions for interrupt handling. Similarly, CPU related interfaces are encapsulated in a CPU class.

AspectC++ uses an own language called *pointcut expression* to identify code locations by class names, inheritance schemes, class methods, function arguments, attributes and/or call patterns. Thus, a pointcut expression denotes a set of code locations which are called *joinpoints*.

An *advice* assigns code to insert or replace at joinpoints as denominated by a pointcut expression. An *aspect* is a set of related advices.

This process of insertion or replacement is called *code weaving*, and occurs transparently at build time before the actual C++ compiler is run. Using AspectC++ shows negligible resource overhead when used for highly resource-constrained systems [45].

AspectC++ helps to keep up the *separation of concerns* paradigm: For each transition, the assigned code location can be used to synthesize a transition advice, which inserts model specific instrumentation code into the driver. The original source code is never touched. Instead, a set of additional code advices is generated from the model "on the fly" and inserted when building the code.

For the following analysis, two levels of instrumentation were used: Data instrumentation applies static energy and power consumption data to the driver, along with respective access functions and tracking of the current FSM state.

Accounting instrumentation extends the driver further and incorporates energy accounting, i.e., time tracking code for all device states, and energy calculations. The calculation results become available to power management modules on higher system levels.

6.5 EVALUATION

In this section, the overheads of instrumented drivers are compared to their uninstrumented counterparts within the Kratos operating system.

To show the feasibility of energy accounting on ULP systems using the proposed method, code size, memory consumption, and energy overhead are examined.

As a quality measure, the accuracy of the counting approach is evaluated by comparing accounted energy consumption to external measurements.

6.5.1 *Data Instrumentation: Data Size*

Data instrumentation inserts static energy and power information to drivers. The information is stored in data arrays. To allow a basic assignment between driver functionalities and energy model, the instrumentation aspect extends device drivers by state-tracking code. By that, the current FSM state of the driver is accessible as a number representing an array-index of the power consumption array.

Static energy data can be represented as one value per transition plus one value per state. For the evaluation setup, 32 bit unsigned integers were used to represent power values in nJ (for transitions) and energy values in nW (for states).

The static data overhead for a driver is represented in Equation 6.1, with S being the number of states and T the number of transitions.

$$M_{driver}^{text} = (S + T) \cdot 4\,B \tag{6.1}$$

6.5.2 *Data Instrumentation: Memory and Code Size*

To show a practical example for the code size of our approach, an existing driver in the operating system was chosen that was thoroughly tested before: The *serial peripheral interface* (SPI) driver is a base communication driver for many devices wired in actually existing setups. The driver model consists of three states and three transitions.

The resulting segment sizes were analyzed and are shown in Table 6.1 for both implementations. The additional code size in the text segment is 52 bytes. According to Equation 6.1, the driver needs 24 B. The remaining 28 B are consumed by the ad-

Implementation	text	bss	sum
original	8140 B	1888 B	10028 B
instrumented	8192 B	1890 B	10082 B

Table 6.1: Segment sizes in bytes of the compiled .elf files for an application using SPI with original and instrumented driver.

ditional code in the transition functions, which is responsible for maintaining a variable to store the current state, and by the getter functions of the energy-aware API.

The bss segment has grown by two bytes, because it bears the variable that keeps the current state.

There was no need for additional stack space when a transition was called. Only a small amount of time (a few cycles, depending on compiler optimizations) for storing the current state in the state variable was consumed.

In conclusion, the additional amount of resources required to insert basic energy-awareness functions into a driver are negligible.

6.5.3 *Energy Accounting: Memory and Code Size*

As energy accounting incorporates more complex behavior into a device driver, the static memory overhead in the text and the BSS segments was analyzed. The text segment represents the increase in code size of the executable binary, while the BSS segment is used for storing variables.

The increase in code size is a result of the following algorithm at the beginning of each transition:

1. **load** current time (ticks)
2. **load** stored time stamp (ticks)
3. **substract** stored time stamp from current time
4. **store** current time as new time stamp
5. **load** the previous state from the state variable
6. **load** the transition power consumptions for the previous state from the data array
7. **multiply** the power consumption and the result from step 3

8. **add** result to driver energy consumed
9. **load** the energy costs for the transition from the data array
10. **add** transition energy cost to driver energy consumed.
11. **store** the succeeding state number in the state variable

Thus, one time-stamp and one energy variable (64 bit each) is additionally required per driver. As shown before, a 32 bit integer variable is used to save the current state.

Depending on the used MCU hardware, compiler optimizations, driver and model structure, and the size number of transition functions, the relative overhead varies.

To gain a practical impression, the memory segments of compiled drivers for the MSP430 architecture with and without accounting instrumentation was observed. Table 6.2 shows the text and BSS segment sizes for the CPU driver and a display driver (Sharp-96 LCD).

Configuration	text	BSS	sum
Display accounting	624 B	18 B	642 B
CPU accounting	588 B	18 B	606 B
Both	1160 B	36 B	1196 B

Table 6.2: Incremental memory requirements for different energy accounting configurations.

From these results, the text segment overhead of the accounting instrumentation can be roughly estimated as follows:

$$M_{acct}^{text} = n \cdot 470\,B + 22\,B \cdot \sum_{i=1}^{n} E_i \tag{6.2}$$

Whereas n is the number of drivers and E_i is the number of transitions of a distinct driver. Of course, these numbers can only give an impression, since the actual costs for each operation vary by all implementation details

The overhead of the accounting is mainly a result of the compiler-generated large data word operations (32 and 64 bit on a 16-bit machine), that are necessary to calculate time and energy values within reasonable accuracy bounds.

6.5.4 *Accuracy*

As a testbed for achievable accuracy, a Texas Instruments EXP-MSP430FR5969 "Launchpad" was used. According to existing driver implementations and data sheets, FSM models were created for CPU, a Sharp-96 display, and a TI CC1200EMK-868-930 transceiver. The models were annotated with average energy values gathered during a process of driving each component through their respective FSM states, while logging the state transitions and energy consumptions using the MIMOSA platform (see Chapter 4).

The used drivers had the following functionalities:

MSP430FR5969: The CPU, driven at 16 MHz, was modeled in two states: Idle and active, as described in Section 6.3.1.

SHARP-96: A fully functional driver for the 96x96px dot matrix display, including initialization, hardware screen clearing, and writing pixel data. Font display and scrolling is handled in the application layer.

CC1200EMK-868-930: The driver was configured for transmission only, and all radio parameters were fixed, so that the model was reduced to an idle and a transmit (TX) state.

For the evaluation, three applications were implemented for utilizing periphery and CPU. While the applications were running, the energy consumption of all components was continuously logged. Additionally, calculated energy accounting results were frequently transferred over a cabled serial (UART) connection. During these data transfers, all energy measurements and accountings were paused.

Table 6.3 shows the minimum and maximum relative error and accuracy of all models for the components. The relative error is calculated as shown in Equation 6.3. The (complementary) accuracy measure is: $1 - \varepsilon_{total}(n)$.

$$\varepsilon_{total}(t) = \left| \frac{p(t)}{m(t)} - 1 \right| \tag{6.3}$$

Application 1: Display initialization

This application triggers the base functionalities of the display: Turning the device on and staying idle for a while, followed by

	error		accuracy	
Component	min	max	min	max
CPU	0.01%	5.9%	94.1%	99.99%
Display	2.9%	9.8%	90.2%	92.1%
CPU+Display	0.1%	5.6%	94.4%	99.9%
CC1200	0.00%	2.8%	97.2%	100.0%

Table 6.3: Minimum and maximum cumulative error and accuracy during both experiments.

a continuous write of pixel data. The application ends by issuing a clear-screen and power-off cycle. The application required 27 s to run.

The results of the measurements and accounting are shown in 6.4 and 6.5. The different phases of the application are clearly visible. Figure 6.6 compares measured and accounted values. Figure 6.6 shows the relative accuracy of the accounting for this application, according to Equation 6.3.

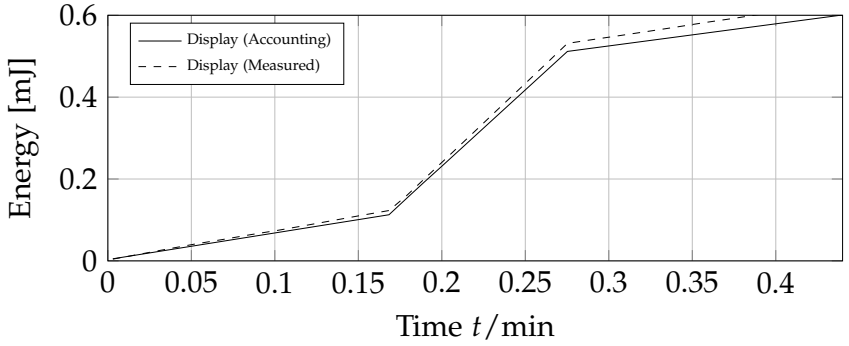


Figure 6.4: Application 1: Display energy consumption for initialization, continuous write, screen clearing and power-off.

Application 2: Display output

To evaluate extensive display usage patterns, a `print` function was implemented that is able to copy characters from a bitmap font to the display memory. The print function also implements cursor management and scrolling. The application uses it for continuous text output, thus causing extensive data transfers to the display. Each write cycle is followed by an idle phase that lets the display hold the still image. The idle phase was

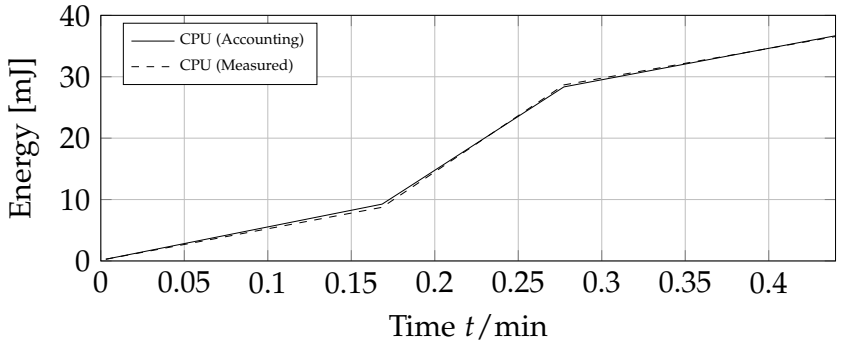


Figure 6.5: Application 1: CPU energy consumption

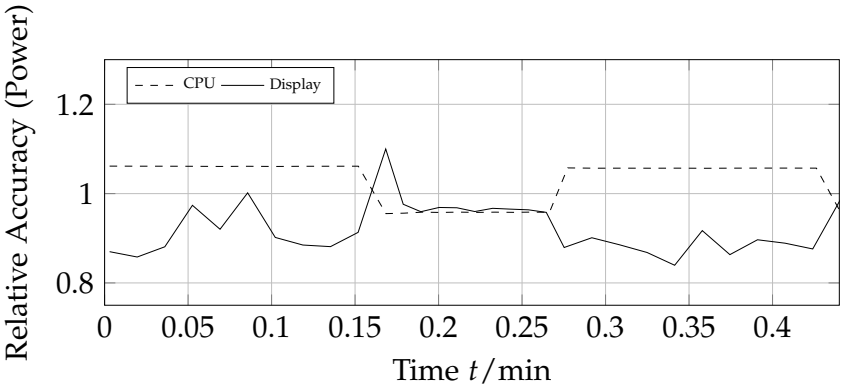


Figure 6.6: Application 1: Model accuracy

continuously increased from 1 ms to 1 s in each iteration. The application stops after 30 min.

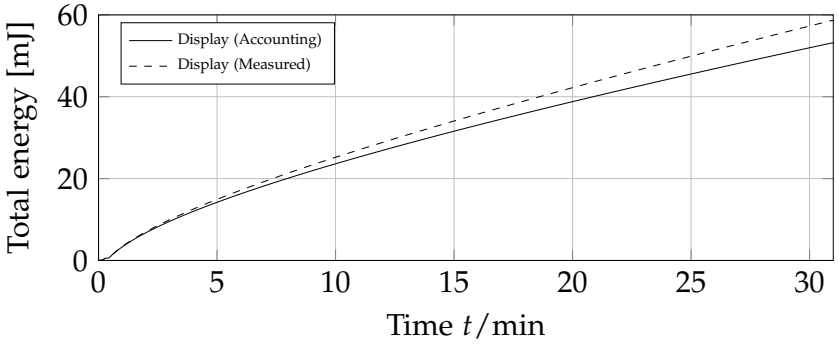


Figure 6.7: Application 2: Display energy consumption at repeated heavy duty write cycles alternated by increasing hold times.

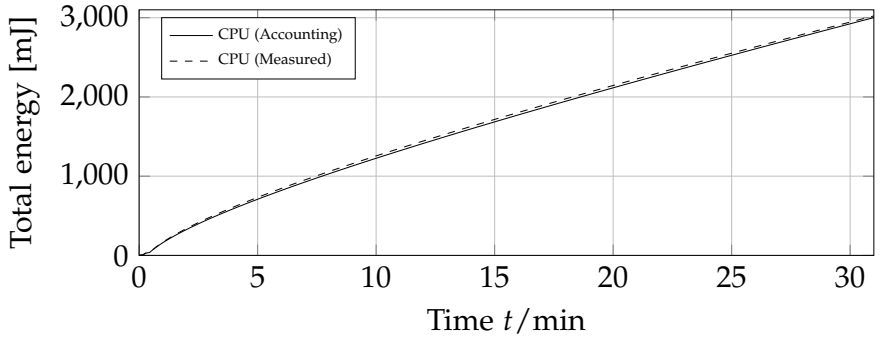


Figure 6.8: Application 2: CPU energy consumption

Application 3: Radio transmissions

With this application, an evaluation of different data transmission patterns was performed. It sends clusters of data packets using the CC1200. Each cluster contains packets of varying size, starting from 8 bytes payload, and increasing to 48 bytes, then decreasing to 8 bytes again. The idle time between the packets of a cluster increases, from 1 ms to 29 ms. A cluster is sent three times consequently. The whole experiment took about 2.3 min.

Model accuracy: Conclusion

The comparison of energy accounting and energy measurements showed different accuracies for different devices: The energy consumption of a CC1200 transceiver could be accounted

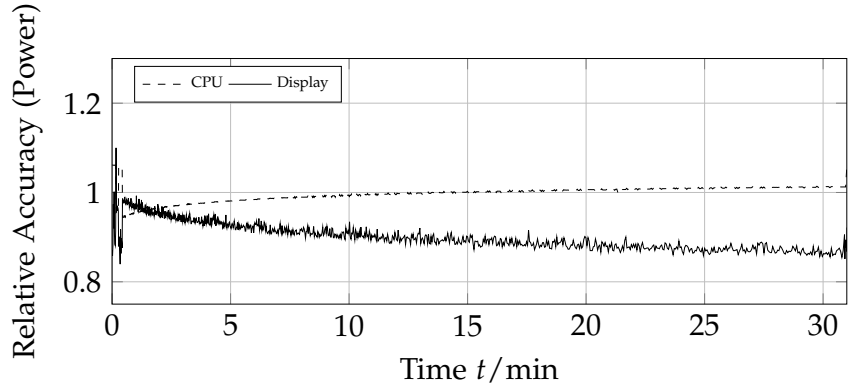


Figure 6.9: Application 2: Model accuracy

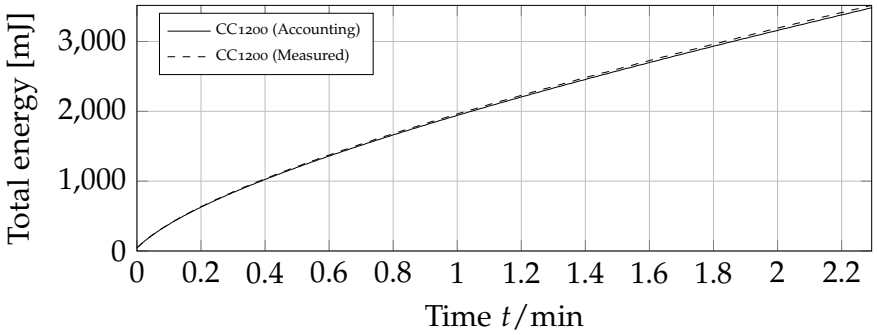


Figure 6.10: Application 3: CC1200 energy consumption

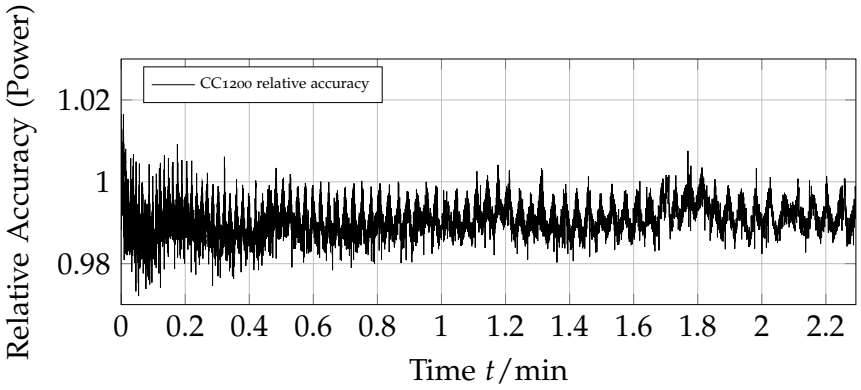


Figure 6.11: Application 3: Model accuracy

with a maximum error of below 3 %. Similar results could be achieved for the CPU energy consumption: Depending on the task, the error was below 6 % in the worst case. Only the Sharp-96 display – due to its very low power consumption and the resulting measurement/modeling error – continuously yielded a higher error of 9.8 %.

6.5.5 *Time and Energy Overhead of the Accounting*

The energy consumption of power-management implementations is an essential measure for its usability: To become beneficial, decisions of the power management must enable higher energy savings than the implied overhead.

To measure the time overhead imposed by energy accounting, the internal timers of the CPU and the external energy measurement equipment was used. Certain events, like the start of a transition, were signaled to the external measurement equipment on an I/O line of the MCU. Both internal and external measurements showed an additional CPU utilization per transition of 58 µs on a 16 MHz driven MSP430-FR5969, or 928 CPU cycles, which consumed an average energy of 138 nJ.

The presented applications show both good and bad examples for imposed overheads: The implementation of the Sharp-96 display driver was deliberately chosen to yield extremely high overheads by accounting every pixel-line written on the display. Here, so many driver calls were performed that not only the energy accounting, but also the driver invocation itself became inefficient. Figures 6.4, 6.5, 6.7, and 6.8 show that the CPU, i.e. the component where the accounting is running, is the most energy-hungry device in the system, while the accounted periphery itself requires only 1/50 of the energy. This scenario was chosen, because is delivers the most reliable information about the feasible accuracy of the model: The low energy consumption of the display ranges at the lower limits of the measurement scale, and the vast number of calls creates statistical stability.

Table 6.4 shows the energy consumption of the accounting instrumentation was ≈1400 mJ, thus ranging at ≈46% of the whole system's energy consumption of 3059 mJ.

In an actual system implementation, a whole-screen update call is more efficient, as it performs one energy accounting iteration for all 96 pixel lines. This would range the efficiency of the implementation into reasonable dimensions.

The third application made 14268 transition calls to the radio driver to transmit 7134 packages. By that, the overhead of the accounting implementation summarizes to 1.9 mJ. This is 0.04% of the overall energy consumption of 4.07 J during the run time. In this situation, the driver instrumentation can be considered beneficial, assuming that the information gathered by the accounting leads to intelligent run-time decisions on a higher level power-management system.

	transitions	driver overhead	overall consumption	%
CC1200	14268	1.9 mJ	4070 mJ	0.04
Display	$\approx 10^6$	1400 mJ	3059 mJ	45.77

Table 6.4: Energy consumptions and overheads of both experiments

6.6 CHAPTER SUMMARY

This chapter presented a concept of modeling the energy consumption of peripheral devices and the CPU of a ULP system. The modeling concept is based on finite-state machines with additional cost annotations. To correlate device activities with device-driver source code, a concept of model-to-code coherence was presented.

The resulting models can be used to extend existing device drivers by model-instrumentation code. This code can serve different purposes, from implementing an energy-consumption database, to online accounting of device energy consumptions, over to measurement specific code, as will be shown in the next chapter.

In a set of evaluation scenarios, AspectC++ was used to instrument different device drivers of the Kratos operating system by energy-consumption data and by accounting functionalities. The instrumented drivers were used to measure the imposed overheads in code size, memory usage, run-time and energy consumption. Additionally, the model accuracy was determined by a comparison of energy accounting results and parallel energy measurements.

The results showed that the presented scheme has low overheads, but the total number of driver calls has to be considered when instrumenting for energy accounting: The lowest over-

heads were measured for device functions that consume more energy than the CPU. For functions with lower device energy consumptions than that of the CPU during a driver call, the accounting can become malicious, because the accounting results are calculated by the CPU. Here, the instrumentation should be avoided.

The model concept yields a fair accuracy. In all experiments, the maximum error never exceeded 10 % for a device at the lower limits of the measurement range. The best model, for the CC1200 transceiver, never exceeded an error of 3 %.

7 AUTOMATED MODEL ANNOTATION

7.1 INTRODUCTION

In Chapter 6, a method was proposed to describe the energy consumption of CPU and periphery devices of an ULP system as cost-annotated FSM models.

Such a FSM model is derived from two sources: First, a description of the device functionality and interface, typically the device data sheet or a pre-existing driver, to derive the structure of the FSM. Second, measured energy values to annotate costs.

However, generating reproducible energy consumption values is a tedious endeavor: Next to the complex measurement setup, it requires an application that triggers all transitions of a FSM, and additionally signals the transitioning to external equipment for synchronizing the measurement results.

After measurements are completed, the resulting data must be analyzed. The result of the analysis must resemble time-dependent power consumptions of the FSM states, and energy values for the transitions.

All these steps require a high amount of expertise and time. Thus, automating the measurement and analysis process is essential to create reproducible cost annotations with reasonable effort.

Figure 7.1 shows the whole process, from implementing drivers and measurement applications, over instrumentation, deployment and measurement, to data analysis and cost annotation. As shown, most steps leading to an annotated model require manual work.

However, the steps of creating a measurement application, compiling, deploying, analyzing the results and (re-) annotating the model will be automated in the following sections. This leads to a measurement system, where only the tasks of developing the device drivers (without respect to the model) and the coherent model structure (without cost annotations) are necessary manual work.

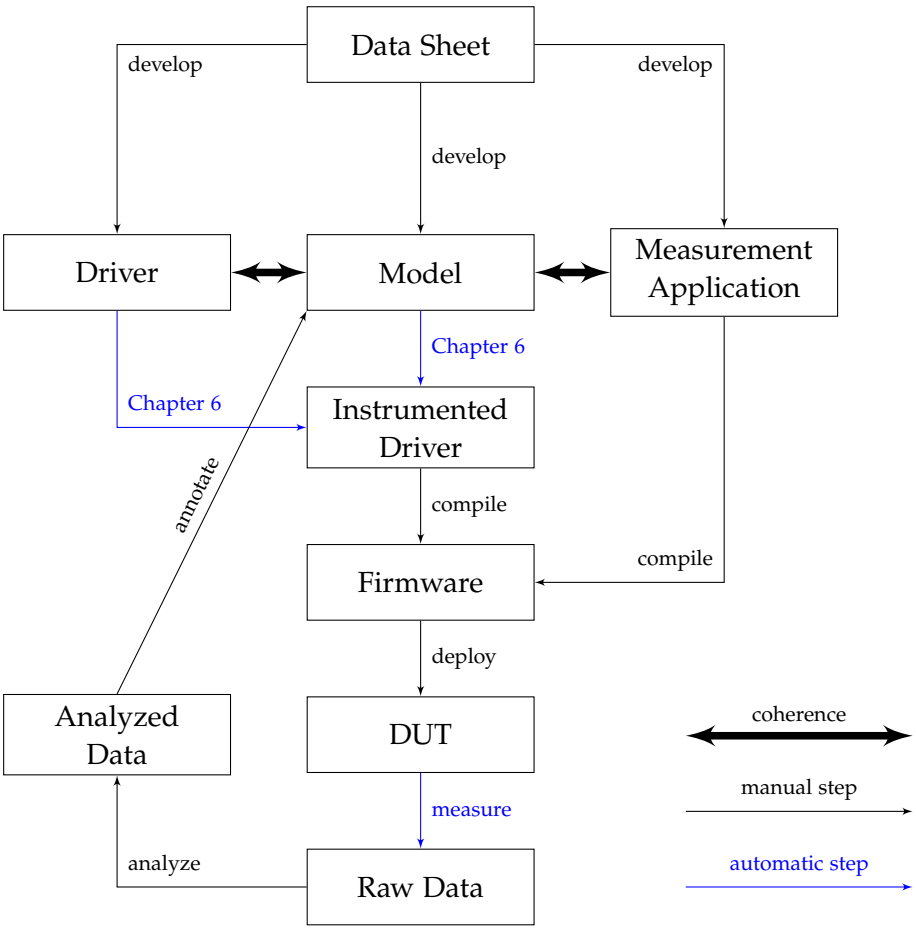


Figure 7.1: Process of manually generating cost annotations for energy models.

As the compilation and deployment steps are usually trivial to automate by scripting, they will not be addressed any further in this thesis.

RELATED PUBLICATIONS

The findings presented in this chapter have partly been published in:

[18] M. Buschhoff, R. Falkenberg, and O. Spinczyk. "Energy-Aware Device Drivers for Embedded Operating Systems." In: *SIGBED Rev.* (2019). to appear.

[19] M. Buschhoff, D. Friesel, and O. Spinczyk. "Energy Models in the Loop." In: *Procedia Comput. Sci.* 130.C (May 2018), pp. 1063–1068. ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.04.154. URL: https://doi.org/10.1016/j.procs.2018.04.154.

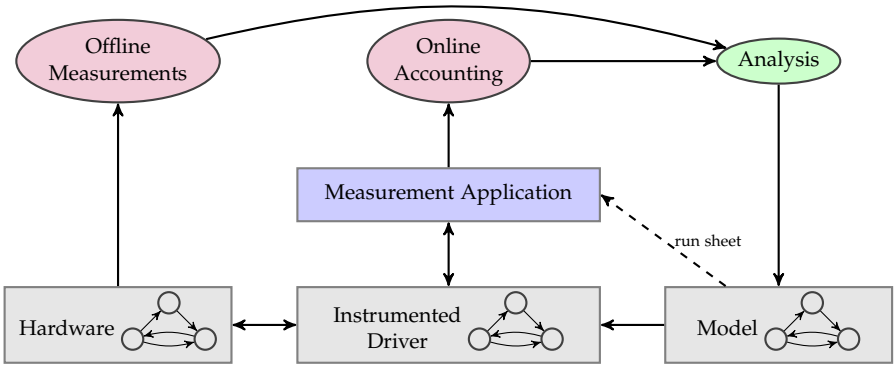


Figure 7.2: Measurement loop. Solid lines resemble automatic steps, dashed lines allow for manual intervention.

[31] D. Friesel, M. Buschhoff, and O. Spinczyk. "Parameter-Aware Energy Models for Embedded-System Peripherals." In: *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*. June 2018, pp. 1–4. DOI: 10.1109/SIES.2018.8442096. URL: https://ieeexplore.ieee.org/document/8442096.

7.1.1 *Contribution*

This chapter presents a system design of an automated measurement and analysis tool chain for the creation of energy models and the instrumentation of device drivers.

The most important contributions of this chapter are:

- The synthesis of a measurement application that explores the FSM model of a peripheral device.
- The assembly of a measurement loop for measuring the energy consumption of a system under test.
- The automated analysis and refinement of cost annotations, and the automated re-instrumentation of drivers to achieve measurement and refinement cycles.

7.2 SEMI-AUTOMATED APPLICATION SYNTHESIS

The synthesis of the measurement application is based on a *run sheet* for the automaton model: It contains a list of transitions to be executed in the given order, grouped in *experiment runs*.

Each experiment run is started by a driver initialization and iterates over a path of the model.

The run sheet can be generated automatically when the following assumptions apply: First, there is a transition (and driver function) available, leading from any state to an initialization state, i.e., the hardware can always be reset or initialized. Second, all states can be reached after initialization.

In this case, all non-cyclic paths from initialization to any state can be found by a cycle free depth-first search.

In the set of found paths for all states, any path that is contained as sub-path within another path can be removed, as the energy model is independent of a transition history. Every remaining path forms an experiment run.

It is advisable to check and manually adapt the run sheet as necessary, especially if hardware requirements might impose restrictions that are not part of the model.

To synchronize the measurement equipment with transitions, the instrumentation aspects are extended to insert code for online-communication with external equipment via I/O pins of the MCU.

As shown in Figure 7.2, the instrumentation code can log energy accounting values, if cost-annotations were already inserted into the model. In this case, the analysis tools compare measurement and accounting results to validate the model and to create a measure for model quality.

After one loop iteration, a valid and stable model should exist. However, if there are large deviations when repeating the loop, this is an indicator for either structural faults in the model or an indeterministic hardware setup.

7.3 AUTOMATED ANALYSIS: STATIC MODEL

In its basic functionality, the analysis tool reads measurement values from the measurement platform and relates them to transitions and states. This assumes constant *power* consumption in the states and constant *energy* consumption in the transitions, thus leading to a static model that ignores function parameters of driver calls. This assumption implies that function parameters to device driver calls only influence the *duration* of a state (and with that, also its energy consumption), but not its power consumption.

Static modeling was used in Chapter 6. This form of modeling is highly application specific, as it does not allow for recon-

figuration of energy relevant hardware parameters. However, it shows low overhead in CPU resources when used for energy accounting.

To gain a static model, several values are continuously calculated and logged for each step in the run sheet during the measurement cycle, e.g., the duration of the former state, the duration of the transition, the average power consumption of the former transition, and the energy consumption of the transition.

The resulting log is analyzed after the whole run sheet was processed, and averaged power and energy consumptions, and further statistical measures (minimum and maximum values, standard deviations, etc.), are calculated for each state and transition. The toolchain finally creates a new model, consisting of the structural model that was used as input, extended by cost annotations. This model can be used to re-iterate the process for verification or accuracy measurement (by comparison of accounted values vs. measured values).

Finally, the generated model can be used for the intended purpose by instrumenting the device driver as required.

7.4 AUTOMATED ANALYSIS: DYNAMIC MODEL

The work presented in the following section is mainly contributed by Daniel Friesel [31] in our collaborative research, and without any claims to his (ongoing) work in this field, a brief overview is given here for completeness.

A dynamic approach extends the static model with function parameters of device driver method calls, and uses regression methods to determine the parameter influence. This makes these models more flexible (and less application specific) than static models, but leads to considerably higher overheads. However, dynamic models can be used to derive static models for a specific configuration.

To model the energetic influence of driver function arguments, the cost annotation scheme of the FSM model must be extended to represent a cost function instead of a numeric constant, and the run sheet must be extended for parameter ranges. The synthesized measurement application iterates through the power set of these ranges. The resulting data is analyzed in three steps:

1. **Identification:** In this phase, all arguments influencing the energy consumption are selected. For this purpose,

reference measurements are taken while keeping the observed argument constant. Afterwards, the observed argument is varied throughout several iterations. If the varied argument leads to an energy consumption that is out of the noise bounds of the reference measurements, the observed argument is considered as influencing.

2. **Single-argument regression:** The cost function for a single argument is identified by using eight regression methods (linear, logarithmic, shifted logarithmic, exponential, square, fraction, root, one-bit count). The function $f(\vec{p})$ with the lowest *sum of squared residuals* (SSR) is chosen for the next step.

3. **Multi-argument regression:** All identified functions $f(\vec{p}) \in F$ are combined within a compound function of the form shown in Equation 7.1. The coefficients for this functions are determined using regression while minimizing the error. The result is the assumed cost function.

$$g(\vec{p}) = \sum_{F' \in \mathcal{P}(F)} \left(a_{F'} \cdot \prod_{f \in F'} f(\vec{p}) \right) \tag{7.1}$$

7.5 EVALUATION

This section will discuss the results of automated dynamic modeling approaches in comparison to static modeling by using a dynamic scenario. A discussion of model accuracy for a purely static configuration is found in Chapter 6.

The automatic measurement and analysis of data was evaluated by creating energy models of two radio transceivers (TI CC1200, nRF24L01+), a low-power I^2C temperature sensor (LM75B), and a synthetic peripheral with programmable power consumption behavior.

Both transceivers contain an IDLE, RX, TX and SLEEP state. For the evaluation of the dynamic model approach, three adjustable parameters were present: Transmission power, bit rate and transmitted data length.

Table 7.1 shows the determined influence of parameters on model attributes and the model error both for static and dynamic model attributes. The model error was assessed using 200 Monte-Carlo cross-validation runs. Data was split into 2/3

model attribute	influencing			error	
	data rate	payload length	tx power	static	model
CC1200 TX power	✓	✓	✓	12 %	0.7 %
CC1200 TX time	✓	✓	–	87 %	0.1 %
CC1200 RX power	✓	✓	–	0.2 %	<0.1 %
nRF24 TX power	✓	–	✓	34 %	1.2 %
nRF24 TX time	✓	–	–	16 %	<0.1 %
nRF24 RX power	✓	–	–	2.2 %	<0.1 %

Table 7.1: Symmetric model error of static and parameter-aware (right) model attributes for CC1200 and nRF24 transceivers in Monte-Carlo cross validation. Parameter influence is shown in the middle.

for training and 1/3 for validation. Figure 6.2 shows an automatically measured PTA model for the CC1200 transceiver.

static error deviation from Chapter 6

The presented error for the static model widely deviates from the given error in Chapter 6, where the error was less than 3%, because fixed parameters for bit rate and transmission power where used in that chapter. In contrast, this chapter uses variable function parameters to compare static and dynamic modeling.

The results show that parameters significantly influence energy consumption, occasionally in unexpected ways. For example, it was expected that the power consumption during TX is constant for the CC1200, and payload length would only influence the energy consumption through TX duration. Actually, even the power depends on the payload length. It turned out that the CC1200 has a fixed preamble with separately set transmission power, so the preamble/payload duration ratio (and hence TX power) also depends on the payload size. In contrast, the nRF24 transmissions use a fixed packet length by default, so the energy consumption of a packet transmission is completely independent of payload length.

With the synthetic peripheral, several functions for state power consumption were tested. The used function was reliably detected within less than 0.7 % model error. Even in a pessimistic parameter-aware cross-validation setting (i.e., the parameter combinations of training and validation set are mutually exclusive, which is rarely the case in real-world usage), correct functions were determined in at least 90 % of cases, and model error did not exceed 1.4 %.

For most parameter-independent transceiver states and the temperature sensor, model errors below 0.8 % were observed for state power consumption. The only exceptions were the CC1200 SLEEP state, showing random deviations of 7 % independent of the parameter settings, and few ultra-low-power states which suffered from limited accuracy of the available measurement equipment. Absolute errors were below 1.2 µW here.

Modeled transition energy showed errors of 1 to 10 % (5 µJ) and transition duration up to 2.5 %. Only errors for transitions longer than 100 µs were correctly measured, as the horizontal accuracy of the measurement equipment was limited to this. Assuming an average of two transitions per second, overall model error was below 1.5 %.

7.6 CHAPTER SUMMARY

This chapter presented an approach to automate large parts of creating energy model annotations for periphery components. By using the methods presented in Chapter 6, an instrumentation mechanism was presented that is able to signal state transitions to external measurement equipment. With a structural (unannotated) FSM model and an ordered list of transitions as input, an application was synthesized on top of the Kratos operating system to drive a target node through all relevant states and transitions.

This combination enables the automated measurement and data analysis with external equipment. Static, averaged cost annotations (as used in Chapter 6) can be derived directly from this data.

In current, ongoing research, regression methods are used to also incorporate dependencies to device-driver function parameters for creating dynamic models. The evaluation section shows first results for model accuracy of dynamic models. However, dynamic models impose higher overheads than static models, but they can be used to derive static models for a given hardware configuration.

8

PRACTICAL APPLICATIONS

This chapter presents two practical applications of the concepts discussed in this thesis.

RELATED PUBLICATIONS

The findings presented in this chapter have partly been published in:

[28] R. Falkenberg, M. Masoudinejad, M. Buschhoff, A. K. Ramachandran Venkatapathy, D. Friesel, M. ten Hompel, O. Spinczyk, and C. Wietfeld. "PhyNetLab: An IoT-Based Warehouse Testbed." In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*. Sept. 2017.

[52] M. Masoudinejad, A. K. Ramachandran Venkatapathy, D. Tondorf, D. Heinrich, R. Falkenberg, and M. Buschhoff. "Machine Learning Based Indoor Localisation Using Environmental Data in PhyNetLab Warehouse." In: *Proceedings of the European Conference on Smart Objects, Systems and Technologies 2018*. Smart Systech 2018. Dresden, Germany, 2018.

8.1 SOLAR DOORPLATE

8.1.1 *Project Group*

The Solar Doorplate (SDP) was established as a student project group led by Olaf Spinczyk, Alexander Lochmann and me, tasking students to design hardware and software for an energy autonomous, radio connected door plate [76]. Twelve students worked on the project for one year (Jan Böcker, Timo Cramer, Daniel Friesel, Sebastian Lukas Hauer, Michael Müller, Todor Nikolov, Benedikt Ruthenberg, Daniel Smit, Merlin Stampa, Martin Strzelczyk, David Tondorf, Franz-Bernhard Tuneke).

The focused application of the door plates is the display of the occupation of seminar rooms and of up-to date information about the presence of staff employees. For that, the door plates have to be updated frequently via radio.

8.1.2 *Node Hardware*

Figure 8.1 shows a single door plate that consists of an MSP430 based controller platform, equipped with a flexible PVC tape, a light intensity sensor and an ePaper display. The devices use super capacitors as energy buffers.

The first prototype was implemented on the pre-existing PhyNode platform (see Section 8.2), and used the existing LCD screen. Then, the feature-rich PhyNode platform was consequently stripped down to the minimum necessary components.

After extensive energy measurements with Mimosa (see Chapter 4), the display, PVC and radio receiver were exchanged to better suit the purpose of a door plate. Figure 8.2 shows the final set of circuit boards. The device is constructed in a way that allows the boards to be folded behind the display.

8.1.3 *Network*

The *Solar Doorplate* (SDP) network uses a wired hub as central node. The hub consists of a *Raspberry PI* micro PC with Linux as operating system. It employs a web server to deliver an interactive web page as user interface.

All devices of the network are equipped with NRF24 transceivers that work in the 2.4 GHz frequency band. To limit the listening phases of the PVC powered devices, the hub assigns time slots to each node. The time slots are organized on a minute raster, so that devices can sleep several minutes between radio phases. Time synchronization occurs by frequent radio broadcasts from the hub.

8.1.4 *Energy Management*

To achieve energy state information, each SDP uses two methods:

First, it estimates the energy income by using a light intensity sensor. To achieve this, a model of sensed light values and energy income was created offline using data from extensive measurements.

Second, the Solar Doorplate platform uses energy accounting to calculate consumed energy for display updates, radio transmissions and active CPU phases. This is supported by the Kratos operating system which was used for all measurements and evaluations in the recent chapters of this thesis. Kratos is

Figure 8.1: A Solar Dooplate

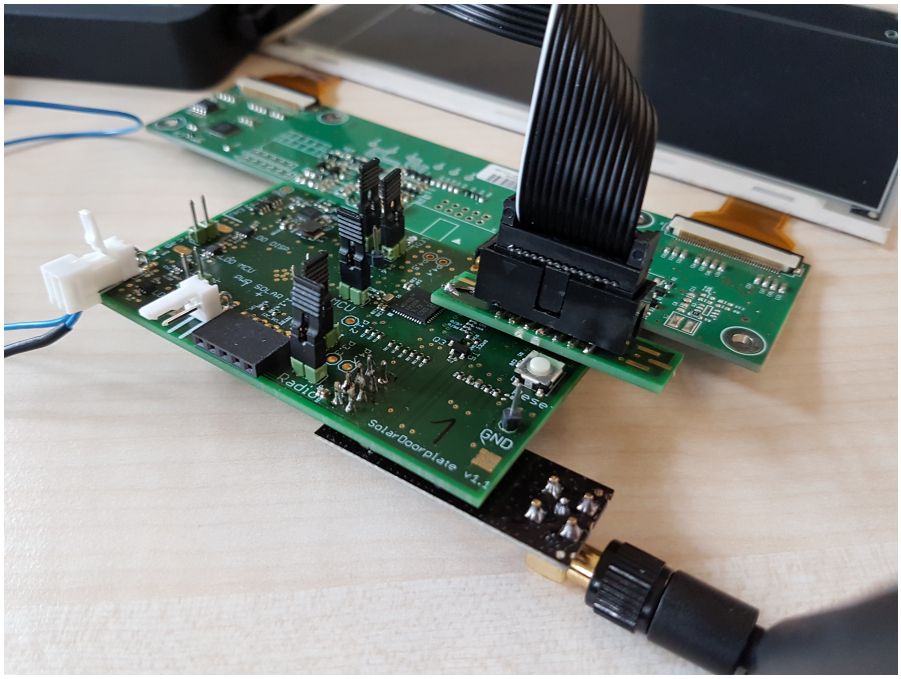


Figure 8.2: The Solar Doorplate circuit board stack: NRF24 transceiver (bottom), Solar Doorplate (middle), display controller (top), magnetic debug connector (ribbon cable)

an extension of the OO-StuBS operating system code, which was created and used for teaching, so that the project-group students were able to use a software platform already known to them.

Finally, the SDPs use the gathered values to calculate the energy buffer charge. The charge level is then sent to the hub during their transmission periods.

This allows the hub to schedule display updates for each door plate according to the individual energy level. The main objective is to avoid interrupted updates of the ePaper display. ePaper updates take several seconds and require a high amount of energy. Energy failures during this time would lead to unreadable, random display images.

8.1.5 *Challenges*

The development of the SDP platform issued several challenges to the state of the art in energy-aware embedded-system engineering. First of all, energy income and use had to be thoroughly balanced. To achieve this, a plethora of different peripheral devices like displays, radio transceivers and PVCs was extensively examined. As an example, to determine the correct PVC size, type, and angle that allows for indoor-use with neon tube illumination, income measurements under changing light intensity, color and angle were performed over several weeks using a measurement device crafted only for this purpose.

The high measurement effort that was necessary for all hardware components intensified the demand for low-cost income measurements and automated measurement methods within the Kratos tool chain. This led to the solutions presented in Chapters 5 and 7.

Finally, the project group members had to create an energy efficient network protocol and an energy policy for the hub. Only the accurate knowledge of the energy consumption in each state of the used transceiver hardware enabled the optimization for both energy and latency.

8.2 PHYNETLAB

PhyNetLab is an endeavor of the Fraunhofer IML institute in Dortmund to create PVC powered cyber-physical sensor networks for to support logistical processes in Industry 4.0 scenarios.

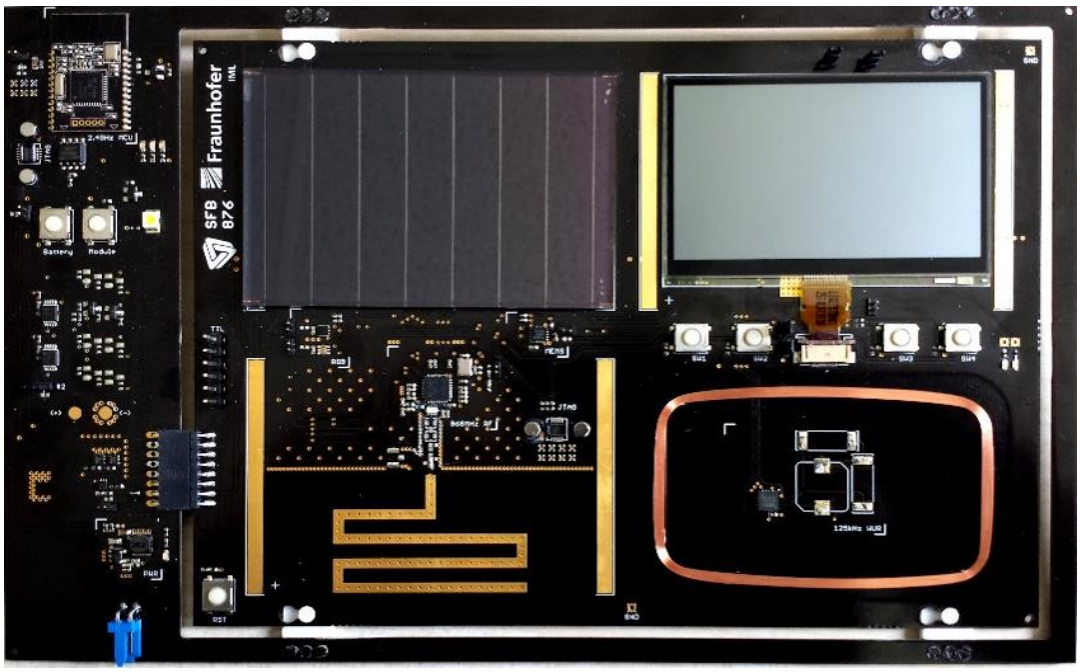


Figure 8.3: A fully equipped PhyNode, consisting of outer frame (MNB) and inner board (SSB). Visible is the PVC (top left), LCD display (top right), wakeup receiver loop (bottom right) and transceiver with circuit-board antenna (bottom left).

PhyNetLab consist of a number of computational nodes, the *PhyNodes*, with different hardware configurations. Each node consists of two circuit boards: A central *swappable slave board* (SSB), which contains all components of the actual node, and an *master network board* (MNB), which allows to control and observe the SSB during experiments. The circuit boards are shown in Figure 8.3.

8.2.1 *Hardware*

A PhyNode SSB consist of a variety of partly optional components, with the most important listed in Table 8.1.

The MNB is powered by an own battery and uses a ZigBee channel to provide health-data, experiment meta-data and firmware management for the PhyNode. It is connected to the slave board by an 8-pin connector. Special care has been taken to decouple the energy consumption of the SSB from the MNB.

CPU	Texas Instruments MSP430FR-5969
PVC	Solems 07/072/048 (4 V, indoor use)
Power management	BQ25570 power point tracker and battery charger, power management IC with programmable output voltages
Energy buffer	A variety of optional buffers, depending on board version (lithium batteries of different capacities, super capacitors)
Sensors	Ambient light intensity (incl. infrared), ambient RGB colour, two temperature sensors, 3-axis accelerometer
Display	(optional) Sharp 400 x 240 px memory LCD
User interaction	4 pushbuttons
Communication	Texas Instruments CC1200 sub 1 GHz ISM band transceiver
Other	(optional) 125 kHz wakeup receiver

Table 8.1: The most important PhyNode SSB components.

Figure 8.4: PhyNetLab shelf setup for a machine learning student competition.

8.2.2 *Software*

PhyNodes can be run by any binary compatible firmware. For contributions of the author's scientific work group, the Kratos operating system was used [28, 52], deploying energy aware concepts and device drivers as discussed within this thesis.

During the 2018 Summer School event of the collaborative research center SFB 876 of the DFG, the PhyNetLab hosted the environment for a student competition: The participants were challenged to analyze formerly collected sensor data of 20 PhyNodes, and to develop an algorithm and PhyNode implementation to localize a PhyNode's position within a grid of discrete shelf locations. The shelf setup is shown in Figure 8.4.

The given sensor data included light and color values, temperatures, accelerometer data, and *received-signal-strength indicator* (RSSI) data from signals transmitted by three positional beacons. To normalize the values to daily environmental variations, two PhyNodes were kept as reference in fixed positions at the far ends of the aisle, and their sensor values were continuously available via radio transmissions during the competition.

The students implemented their solution to self-localization within a preconfigured Kratos development environment. By that they had access to all sensors using the instrumented Kratos device drivers. The given radio protocols allowed them to read the data of the reference nodes.

Energy accounting values were used as a performance measure.

The software framework offered a function to transmit the final localization result via radio. The result data included energy values from the energy accounting device drivers. It thus became possible to use energy consumption as a second performance measure for the participant ranking.

The reference solution for the given scenario was published [52].

9

SUMMARY AND CONCLUSION

This chapter discusses the research questions and the thesis statement presented in Chapter 1.

9.1 RESEARCH QUESTIONS

9.1.1 *Measurement and Modeling*

> “What are reasonable methods to measure and model energy consumptions of components of ULP embedded systems? What is the possible accuracy of such models? Can they be applied to operating system software to account energy consumption, and what is the energetic overhead of this?”
>
> *Research questions, Chapter 1*

The measurement of energy consumption of ultra low power system components has shown a nontrivial set of problems: A practical measurement setup requires a high resolution in both horizontal (time) and vertical (power) direction. As long-time measurements over several minutes are required, energy measurement is also memory intensive. Additionally, the measurements need to be synchronized and annotated with hardware and software events of the measurement platform under test.

Market ready solutions to this do partially exist, in form of oscilloscopes (usually with a very limited set of input channels and special requirements for long time measurements), specialized measurement equipment (usually designed for a different purpose, like battery modeling, often lacking digital inputs), or simple shunt based solution with inappropriate precision. At the time being, the most suitable device for the purpose of ultra low power measurement range in a price segment above €10,000, and still require compromises. The price range limits the possibilities of measuring many components in parallel.

The findings in Chapter 4 show that accurate energy measuring for the given purpose is possible to achieve and inexpensive, with prototype costs below €200. However, this requires re-thinking the approach to measurement: The proposed MI-

MOSA approach is a *source measurement unit*, replacing the original power source. It compensates the drop of voltage occurring at a measurement shunt by an active voltage stabilizer component, and uses analog integration circuits to improve average accuracy while reducing the amount of measurement data.

With precise measurements, energy annotated finite state machine models were presented in Chapter 6. Transitions of these models can be assigned to code locations of existing device drivers, and by aspect oriented programming and code synthesis, energy accounting drivers were instantiated. This method showed that the energy models can reach an accuracy of over 90% (over 98% for some devices) in comparison to energy measurements. If the model is applied thoroughly, the overheads of this method are negligible, especially for the largest power consumers of a system. However, with excessive call patterns and devices with lower energy consumption than that of the CPU, the energy overheads of the accounting device drivers can become dominant.

9.1.2 *In-Situ Measurement*

> “What are efficient and inexpensive online-measurement methods for transiently powered systems? How can they be used to adapt the system load to energy income? Does this have a positive impact on the system’s up time?”
>
> *Research questions, Chapter 1*

The market for power measurement circuitry offers a variety of solutions to online power measurement and management. Among them, there is a plethora of battery charging units, voltage converters, energy meters, harvesting solutions and smart batteries.

However, these components add to price, weight and size of a system. The impact on the deployment of mobile Industry 4.0 devices can be enormous.

The author of this thesis worked in the collaborative research center SFB-876, project A4, which was motivated by the vision of deploying sensor network nodes on several hundred thousands of warehouse containers. Here, the utilization of battery driven nodes reveals new problem domains: Economy, ecology, maintenance durations and maintenance costs now become limiting factors for the installation of such an infrastructure. These

problems are not typical for the research fields of computer science and electronic engineering, but need to be solved there.

A core technology that enables battery-free, energy-neutral system designs is the in-situ measurement of harvested energy to facilitate the adaptation of energy consumption. Chapter 5 gives an overview of battery-powered system designs in the state of the art, showing that existing solutions, although being market ready and working, are ignorant about economic, ecologic and maintenance issues. An own solution is proposed in that chapter, to completely remove the battery and harvester components and to achieve energy measurements with basic electronic components, while yielding considerable financial savings and removing the ecologic impact of batteries. The shown approach utilizes parasitic effects on the MCU and uses on-chip timer/counter for flank time measurements. Two efficiency measures were empirically evaluated by an experimentation setup, showing that the efficiency of the proposed solution is in the same range as the state of the art in energy neutral system design, at lower costs and with reduced component count.

9.1.3 *Automatic Model Generation*

> “Can energy measurement and modeling be automated to become easily applicable for application developers? Can energy aware software be synthesized from these models?”
>
> *Research questions, Chapter 1*

One of the main obstacles in using online energy models is their limited availability. Creating online models for each component is a daunting task that requires understanding of measurement equipment, a structural or functional model derived from a data sheet, and specialized measurement/benchmarking applications. These applications are necessary to utilize the hardware of a system under test for facilitating time synchronization of measured data and hardware events. After this complex process, all collected data has to be analyzed and incorporated into the model to finally add data structures and algorithms into software that reflect the model.

Additionally, the combination of large hardware configuration spaces and the highly limited resources of ultra-low-power systems pose another challenge: A “one size fits all” approach does not exist, so the model and its implementation must be re-

duced to a reasonable subset of the whole configuration space for a specific application.

With the high complexity of this process, it seems improbable that developers will be able to obtain a comprehensive database of manually crafted online models for their applications. Here, automatic model generation systems can help to ease the process, but to the best knowledge of the author, there is no current state of the art for a combination of external measurements, code analysis and flexible software instrumentation. The high specialization of research in the respective fields leads to isolated, highly complex partial solutions that are hard to combine into a crosscutting approach to automatic modeling and instrumentation.

The findings in Chapters 4 and 6 prepared for the automated approach shown in Chapter 7. Here, the structural automaton models (still manually derived from data sheets or device driver implementations) are constructed to be *coherent* to existing device drivers. This means that model entities can be mapped to code locations in the driver source code. These models allow for synthesizing benchmarking applications that drive the hardware components into each energetic state of the structural model, while signaling state changes to external measurement equipment.

With such an application running on the target system, automated energy measurements with the MIMOSA system can be performed. Afterwards, the collected data can be analyzed automatically: Each data point can be assigned to hardware states, and by statistical methods and methods of machine learning, all data points can be cumulated into cost models of scalable complexity: From simple (low overhead) averaged energy costs for hardware states and transitions, to complex, detailed models with high calculation overhead that take the interdependencies of device driver function call arguments into account. That means, cost annotations can be generated within minutes, while requiring low-cost measurement equipment.

The final step is the automated integration of the cost-annotated model into the device driver. Here, aspect oriented programming is used to insert ("weave") source code into the coherent device driver. This instrumentation code can be set up for different purposes. During the automated measurements, this mechanism is used to insert signaling code to external periphery on driver calls. For final deployment, online energy accounting code was inserted. However, the mechanisms used

in this thesis allow for any custom instrumentation, like logging driver calls plus energy data, or energy based driver access schemes for power management.

9.2 CONCLUSION

> “System software that utilizes precise energy models to become aware of the electrical energy consumption of device components enables efficient power management solutions. This results in decreased recharge cycles of existing battery powered solutions. In combination with energy harvesting technologies, it enables new battery free applications.”
>
> *Thesis Statement, Chapter 1*

The findings presented in this thesis led to a set of systems used in research and teaching at the TU-Dortmund university. Chapter 8 is a showcase of some of these systems: A radio network capable digital door tag, that is solely powered by photovoltaic cells, and a whole cyber-physical system testbed that allows experiments respecting energy distribution in a large network of sensor nodes. Chapter 5 demonstrates a low-cost, battery free sensor network node for illumination sensing.

The research presented here implies two levels of power management: Low-level approaches use clock-gating, voltage/frequency scaling and power switching for idle components. The mechanisms and their impact are well researched, and a plethora of sophisticated management functions is already transparently implemented in hardware. Thus, the objective for my research was to create a foundation to high-level power-management solutions. These solutions are problem or application specific, as they exploit the limits of availability, response time, transmission range, and other measures of quality.

However, high-level decisions require valid information about system state, currently available power, buffered energy, and device specific energy consumptions. This led to two lines of approach: First, online measurement methods to estimate energy buffer charges and *energy income* at the lowest costs and highest efficiency possible. Second, the estimation of *energy expenses* by continuous tracking of device driver calls and sleep modes. Using low-overhead energy models at the device driver level, all information demands about system state, as-

sumed/projected energy consumptions and actual energy consumptions can be satisfied.

BIBLIOGRAPHY

[1] "A Conversation with Steve Furber." In: *Queue* 8.2 (Feb. 2010), pp. 1–8. ISSN: 1542-7730. DOI: 10.1145/1716383.1716385. URL: http://doi.acm.org/10.1145/1716383.1716385.

[2] Z. Abbas and M. Olivieri. "Impact of technology scaling on leakage power in nano-scale bulk CMOS digital standard cells." In: *Microelectronics Journal* 45.2 (2014), pp. 179–195. ISSN: 0026-2692. DOI: https://doi.org/10.1016/j.mejo.2013.10.013. URL: http://www.sciencedirect.com/science/article/pii/S002626921300253X.

[3] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. "Wireless sensor networks: a survey." In: *Computer Networks* 38.4 (2002), pp. 393–422. ISSN: 1389-1286. DOI: http://doi.org/10.1016/S1389-1286(01)00302-4. URL: http://www.sciencedirect.com/science/article/pii/S1389128601003024.

[4] R. Alur, S. La Torre, and G. Pappas. "Optimal paths in weighted timed automata." In: *Hybrid systems: computation and control*. Springer, 2001, pp. 49–62.

[5] ARM Ltd. [GB]. *Cortex-M*. May 2019. URL: https://developer.arm.com/ip-products/processors/cortex-m.

[6] D. Balsamo, A. Das, A. S. Weddell, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini. "Graceful Performance Modulation for Power-Neutral Transient Computing Systems." In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35.5 (May 2016), pp. 738–749. ISSN: 0278-0070. DOI: 10.1109/TCAD.2016.2527713.

[7] D. Balsamo, A. S. Weddell, G. V. Merrett, B. M. Al-Hashimi, D. Brunelli, and L. Benini. "Hibernus: Sustaining Computation During Intermittent Supply for Energy-Harvesting Systems." In: *IEEE Embedded Systems Letters* 7.1 (Mar. 2015), pp. 15–18. ISSN: 1943-0663. DOI: 10.1109/LES.2014.2371494.

[8] G. Behrmann, A. Fehnker, T. Hune, K. Larsen, P. Pettersson, J. Romijn, and F. Vaandrager. *Minimum-cost reachability for priced time automata*. Springer, 2001.

[9] F. Bellosa. "The benefits of event: driven energy accounting in power-sensitive systems." In: *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*. ACM. 2000, pp. 37–42.

[10] L. Benini, A. Bogliolo, and G. D. Micheli. "A survey of design techniques for system-level dynamic power management." In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 8.3 (June 2000), pp. 299–316. ISSN: 1063-8210. DOI: 10.1109/92.845896.

[11] C. Bergonzini, D. Brunelli, and L. Benini. "Algorithms for harvested energy prediction in batteryless wireless sensor networks." In: *2009 3rd International Workshop on Advances in sensors and Interfaces*. June 2009, pp. 144–149. DOI: 10.1109/IWASI.2009.5184785.

[12] N. Berthier, F. Maraninchi, and L. Mounier. "Synchronous Programming of Device Drivers for Global Resource Control in Embedded Operating Systems." In: *ACM Transactions on Embedded Computing Systems (TECS)* 12.1s (Mar. 2013), 39:1–39:26. ISSN: 1539-9087. DOI: 10.1145/2435227.2435235. URL: http://doi.acm.org/10.1145/2435227.2435235.

[13] W. L. Bircher and L. K. John. "Complete system power estimation: A trickle-down approach based on performance events." In: *2007 IEEE International Symposium on Performance Analysis of Systems & Software*. IEEE. 2007, pp. 158–168.

[14] D. Brunelli, C. Moser, L. Thiele, and L. Benini. "Design of a Solar-Harvesting Circuit for Batteryless Embedded Systems." In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 56.11 (Nov. 2009), pp. 2519–2528. ISSN: 1549-8328. DOI: 10.1109/TCSI.2009.2015690.

[15] B. Buchli, P. Kumar, and L. Thiele. "Optimal Power Management With Guaranteed Minimum Energy Utilization For Solar Energy Harvesting Systems." In: *Proceedings of the 11th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*. Fortaleza, Brazil: IEEE, June 2015.

[16] M. Buschhoff. "KRATOS - A Resource Aware, Tailored Operating System." In: *Technical report for Collaborative Research Center SFB 876 - Graduate School*. Ed. by K. Morik and W. Rhode. 10. Dec. 2014.

[17] M. Buschhoff, C. Günter, and O. Spinczyk. "MIMOSA, a Highly Sensitive and Accurate Power Measurement Technique for Low-Power Systems." In: *Real-World Wireless Sensor Networks*. Ed. by K. Langendoen, W. Hu, F. Ferrari, M. Zimmerling, and L. Mottola. Vol. 281. Lecture Notes in Electrical Engineering. Springer International Publishing, 2014, pp. 139–151. ISBN: 978-3-319-03070-8. DOI: 10.1007/978-3-319-03071-5_16.

[18] M. Buschhoff, R. Falkenberg, and O. Spinczyk. "Energy-Aware Device Drivers for Embedded Operating Systems." In: *SIGBED Rev.* (2019). to appear.

[19] M. Buschhoff, D. Friesel, and O. Spinczyk. "Energy Models in the Loop." In: *Procedia Comput. Sci.* 130.C (May 2018), pp. 1063–1068. ISSN: 1877-0509. DOI: 10.1016/j.procs.2018.04.154. URL: https://doi.org/10.1016/j.procs.2018.04.154.

[20] M. Buschhoff, C. Günter, and O. Spinczyk. "A unified approach for online and offline estimation of sensor platform energy consumption." In: *Proceedings of the 8th International Wireless Communications and Mobile Computing Conference (IWCMC '12)*. Limassol, Cyprus, Aug. 2012, pp. 1154–1158. DOI: 10.1109/IWCMC.2012.6314369.

[21] M. Buschhoff, J. Streicher, B. Dusza, C. Wietfeld, and O. Spinczyk. "MobiSIM: A Simulation Library for Resource Prediction of Smartphones and Wireless Sensor Networks." In: *Proceedings of the 46th Annual Simulation Symposium*. ANSS '13. San Diego, California: Society for Computer Simulation International, 2013.

[22] R. C. Campbell. "A Circuit-based Photovoltaic Array Model for Power System Studies." In: *2007 39th North American Power Symposium*. Sept. 2007, pp. 97–101. DOI: 10.1109/NAPS.2007.4402293.

[23] H. Chen, G. Godet-Bar, F. Rousseau, and F. Petrot. "Me3D: A model-driven methodology expediting embedded device driver development." In: *22nd IEEE International Symposium on Rapid System Prototyping (RSP)*. May 2011, pp. 171–177. DOI: 10.1109/RSP.2011.5929992.

[24] P. Chou, R. Ortega, and G. Borriello. "Synthesis of the hardware/software interface in microcontroller-based systems." In: *1992 IEEE/ACM International Conference on Computer-Aided Design*. Nov. 1992, pp. 488–495. DOI: 10.1109/ICCAD.1992.279322.

[25] M. Dong and L. Zhong. "Self-constructive high-rate system energy modeling for battery-powered mobile systems." In: *Proceedings of the 9th international conference on Mobile systems, applications, and services*. ACM. 2011, pp. 335–348.

[26] P. Dutta, M. Feldmeier, J. Paradiso, and D. Culler. "Energy Metering for Free: Augmenting Switching Regulators for Real-Time Monitoring." In: *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*. IPSN '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 283–294. ISBN: 978-0-7695-3157-1. DOI: 10.1109/IPSN.2008.58. URL: https://doi.org/10.1109/IPSN.2008.58.

[27] G. Fagas, L. Gammaitoni, D. Paul., and G. A. Berini, eds. *ICT - Energy - Concepts Towards Zero - Power Information and Communication Technology*. location: InTech. ISBN: 978-953-51-1218-1. DOI: 10.5772/55410.

[28] R. Falkenberg, M. Masoudinejad, M. Buschhoff, A. K. Ramachandran Venkatapathy, D. Friesel, M. ten Hompel, O. Spinczyk, and C. Wietfeld. "PhyNetLab: An IoT-Based Warehouse Testbed." In: *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*. Sept. 2017.

[29] D. Flynn, R. Aitken, A. Gibbons, and K. Shi. *Low Power Methodology Manual: For System-on-Chip Design*. Integrated Circuits and Systems. Springer US, 2007. ISBN: 9780387718194. URL: https://books.google.de/books?id=C4yy1NF0QwUC.

[30] D. Friesel, M. Buschhoff, and O. Spinczyk. "Annotations in Operating Systems with Custom AspectC++ Attributes." In: *Proceedings of the 9th Workshop on Programming Languages and Operating Systems (PLOS '17)*. PLOS'17. Shanghai, China: ACM, 2017, pp. 36–42. ISBN: 978-1-4503-5153-9. DOI: 10.1145/3144555.3144561. URL: http://doi.acm.org/10.1145/3144555.3144561.

[31] D. Friesel, M. Buschhoff, and O. Spinczyk. "Parameter-Aware Energy Models for Embedded-System Peripherals." In: *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*. June 2018, pp. 1–4. DOI: 10.1109/SIES.2018.8442096. URL: https://ieeexplore.ieee.org/document/8442096.

[32] K. Georgiou, S. Xavier-de-Souza, and K. Eder. "The IoT Energy Challenge: A Software Perspective." In: *IEEE Embedded Systems Letters* 10.3 (Sept. 2018), pp. 53–56. ISSN: 1943-0663. DOI: 10.1109/LES.2017.2741419.

[33] A. Gomez, L. Sigrist, M. Magno, L. Benini, and L. Thiele. "Dynamic Energy Burst Scaling for Transiently Powered Systems." In: *Proceedings of the 2016 Conference on Design, Automation & Test in Europe*. DATE '16. Dresden, Germany: EDA Consortium, 2016, pp. 349–354. ISBN: 978-3-9815370-6-2. URL: http://dl.acm.org/citation.cfm?id=2971808.2971888.

[34] H. Gualous, R. Gallay, G. Alcicek, B. Tala-Ighil, A. Oukaour, B. Boudart, and P. Makany. "Supercapacitor ageing at constant temperature and constant voltage and thermal shock." In: *Microelectronics Reliability* 50.9 (2010). 21st European Symposium on the Reliability of Electron Devices, Failure Physics and Analysis, pp. 1783–1788. ISSN: 0026-2714. DOI: https://doi.org/10.1016/j.microrel.2010.07.144. URL: http://www.sciencedirect.com/science/article/pii/S0026271410004178.

[35] M. Jazayeri, S. Uysal, and K. Jazayeri. "A simple MATLAB/Simulink simulation for PV modules based on one-diode model." In: *2013 High Capacity Optical Networks and Emerging/Enabling Technologies*. Dec. 2013, pp. 44–50. DOI: 10.1109/HONET.2013.6729755.

[36] W. Jung, C. Kang, C. Yoon, D. Kim, and H. Cha. "DevScope: A Nonintrusive and Online Power Analysis Tool for Smartphone Hardware Components." In: *Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. CODES+ISSS '12. Tampere, Finland: ACM, 2012, pp. 353–362. ISBN: 978-1-4503-1426-8. DOI: 10.1145/2380445.2380502.

[37] S. Käbitz, J. B. Gerschler, M. Ecker, Y. Yurdagel, B. Emmermacher, D. André, T. Mitsch, and D. U. Sauer. "Cycle and calendar life study of a

graphite | LiNi1/3Mn1/3Co1/3O2 Li-ion high energy system. Part A: Full cell characterization." In: *Journal of Power Sources* 239 (2013), pp. 572–583. ISSN: 0378-7753. DOI: https://doi.org/10.1016/j.jpowsour.2013.03.045. URL: http://www.sciencedirect.com/science/article/pii/S0378775313004369.

[38] S. Kellner and F. Bellosa. "Energy accounting support in tinyos." In: *PIK-Praxis der Informationsverarbeitung und Kommunikation* 32.2 (2009), pp. 105–109.

[39] S. Kellner, M. Pink, D. Meier, and E.-O. Blass. "Towards a Realistic Energy Model for Wireless Sensor Networks." In: *Fifth Annual Conference on Wireless on Demand Network Systems and Services*. Jan. 2008, pp. 97–100. DOI: 10.1109/WONS.2008.4459362.

[40] S. Kerrison, M. Buschhoff, J. Nunez-Yanez, and K. Eder. "Measuring Energy." In: *ICT - Energy Concepts for Energy Efficiency and Sustainability*. Ed. by G. Fagas, L. Gammaitoni, J. P. Gallagher, and D. J. Paul. Rijeka: InTech, 2017. Chap. 03. DOI: 10.5772/65989. URL: http://dx.doi.org/10.5772/65989.

[41] M. B. Kjærgaard and H. Blunck. "Unsupervised Power Profiling for Mobile Devices." In: *Mobile and Ubiquitous Systems: Computing, Networking, and Services: 8th International ICST Conference, MobiQuitous 2011, Revised Selected Papers*. Ed. by A. Puiatti and T. Gu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 138–149. ISBN: 978-3-642-30973-1. DOI: 10.1007/978-3-642-30973-1_12.

[42] K. Kordesch, J. Gsellmann, M. Peri, K. Tomantschger, and R. Chemelli. "The rechargeability of manganese dioxide in alkaline electrolyte." In: *Electrochimica Acta* 26.10 (1981), pp. 1495–1504. ISSN: 0013-4686. DOI: https://doi.org/10.1016/0013-4686(81)90021-9. URL: http://www.sciencedirect.com/science/article/pii/0013468681900219.

[43] H. G. Lee and N. Chang. "Powering the IoT: Storage-less and converter-less energy harvesting." In: *The 20th Asia and South Pacific Design Automation Conference*. Jan. 2015, pp. 124–129. DOI: 10.1109/ASPDAC.2015.7058992.

[44] Q. Liu and C. Jung. "Lightweight hardware support for transparent consistency-aware checkpointing in intermittent energy-harvesting systems." In: *2016 5th Non-Volatile Memory Systems and Applications Symposium (NVMSA)*. Aug. 2016, pp. 1–6. DOI: 10.1109/NVMSA.2016.7547183.

[45] D. Lohmann, W. Hofer, W. Schröder-Preikschat, J. Streicher, and O. Spinczyk. "CiAO: An Aspect-Oriented Operating-System Family for Resource-Constrained Embedded Systems." In: *Proceedings of the 2009 USENIX Annual Technical Conference (ATC '09)*. Berkeley, CA, USA: USENIX Association, June 2009, pp. 215–228.

[46] K. Ma, Y. Zheng, S. Li, K. Swaminathan, X. Li, Y. Liu, J. Sampson, Y. Xie, and V. Narayanan. "Architecture exploration for ambient energy harvesting nonvolatile processors." In: *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*. Feb. 2015, pp. 526–537. DOI: 10.1109/HPCA.2015.7056060.

[47] Madden, Samuel R. *TinyDB: A Declarative Database for Sensor Networks*. URL: http://telegraph.cs.berkeley.edu/tinydb/.

[48] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. "TinyDB: An Acquisitional Query Processing System for Sensor Networks." In: *ACM Trans. Database Syst.* 30.1 (Mar. 2005), pp. 122–173. ISSN: 0362-5915. DOI: 10.1145/1061318.1061322. URL: http://doi.acm.org/10.1145/1061318.1061322.

[49] M. Magno, D. Porcarelli, D. Brunelli, and L. Benini. "InfiniTime: A multi-sensor energy neutral wearable bracelet." In: *International Green Computing Conference*. Nov. 2014, pp. 1–8. DOI: 10.1109/IGCC.2014.7039180.

[50] B. Martinez, M. Montón, I. Vilajosana, and J. D. Prades. "The Power of Models: Modeling Power Consumption for IoT Devices." In: *IEEE Sensors Journal* 15.10 (Oct. 2015), pp. 5777–5789. ISSN: 1530-437X. DOI: 10.1109/JSEN.2015.2445094.

[51] M. Masoudinejad, J. Emmerich, D. Kossmann, A. Riesner, M. Roidl, and M. ten Hompel. "Development of a measurement platform for indoor photovoltaic energy harvesting in materials handling applications." In: *IREC2015 The Sixth International Renewable Energy Congress*. Mar. 2015, pp. 1–6. DOI: 10.1109/IREC.2015.7110938.

[52] M. Masoudinejad, A. K. Ramachandran Venkatapathy, D. Tondorf, D. Heinrich, R. Falkenberg, and M. Buschhoff. "Machine Learning Based Indoor Localisation Using Environmental Data in PhyNetLab Warehouse." In: *Proceedings of the European Conference on Smart Objects, Systems and Technologies 2018*. Smart Systech 2018. Dresden, Germany, 2018.

[53] J. C. McCullough, Y. Agarwal, J. Chandrashekar, S. Kuppuswamy, A. C. Snoeren, and R. K. Gupta. "Evaluating the effectiveness of model-based power characterization." In: *USENIX Annual Technical Conf*. Vol. 20. 2011.

[54] F. Mérillon, L. Réveillère, C. Consel, R. Marlet, and G. Muller. "Devil: An IDL for hardware programming." In: *Proceedings of the 4th conference on Symposium on Operating System Design & Implementation-Volume 4*. USENIX Association. 2000, pp. 2–2.

[55] H. J. Osten, J. P. Liu, P. Gaworzewski, E. Bugiel, and P. Zaumseil. "High-k gate dielectrics with ultra-low leakage current based on praseodymium oxide." In: *International Electron Devices Meeting 2000. Technical Digest. IEDM (Cat. No.00CH37138)*. Dec. 2000, pp. 653–656. DOI: 10.1109/IEDM.2000.904404.

[56] J. Pallister, S. Kerrison, J. Morse, and K. Eder. "Data dependent energy modelling: A worst case perspective." In: *Computing Research Repository, arXiv* (2015).

[57] A. Pathak, Y. C. Hu, M. Zhang, P. Bahl, and Y.-M. Wang. "Fine-grained power modeling for smartphones using system call tracing." In: *Proceedings of the sixth conference on Computer systems*. ACM. 2011, pp. 153–168.

[58] P. Pillai and K. G. Shin. "Real-time dynamic voltage scaling for low-power embedded operating systems." In: *ACM SIGOPS Operating Systems Review*. Vol. 35. 5. ACM. 2001, pp. 89–102.

[59] QOITECH AB. *Otii Arc Technical Specification*. URL: https://www.qoitech.com/products/techspec.

[60] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman, and M. Srivastava. "Design Considerations for Solar Energy Harvesting Wireless Embedded Systems." In: *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*. IPSN '05. Los Angeles, California: IEEE

Press, 2005. ISBN: 0-7803-9202-7. URL: http://dl.acm.org/citation.cfm?id=1147685.1147764.

[61] B. Ransford, S. Clark, M. Salajegheh, and K. Fu. "Getting Things Done on Computational RFIDs with Energy-Aware Checkpointing and Voltage-Aware Scheduling." In: *HotPower* 8 (2008), pp. 5–5.

[62] A. Rodriguez Arreola, D. Balsamo, A. Das, A. W, D. Brunelli, B. Al-Hashimi, and G. Merrett. "Approaches to Transient Computing for Energy Harvesting Systems: A Quantitative Evaluation." In: *Proceedings of the 3rd International Workshop on Energy Harvesting & Energy Neutral Sensing Systems*. ENSsys '15. Seoul, South Korea: ACM, 2015, pp. 3–8. ISBN: 978-1-4503-3837-0. DOI: 10.1145/2820645.2820652. URL: http://doi.acm.org/10.1145/2820645.2820652.

[63] T. Ruan, Z. J. Chew, and M. Zhu. "Energy-Aware Approaches for Energy Harvesting Powered Wireless Sensor Nodes." In: *IEEE Sensors Journal* 17.7 (Apr. 2017), pp. 2165–2173. ISSN: 1530-437X. DOI: 10.1109/JSEN.2017.2665680.

[64] L. Ryzhyk, P. Chubb, I. Kuz, E. Le Sueur, and G. Heiser. "Automatic Device Driver Synthesis with Termite." In: *Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles*. SOSP '09. Big Sky, Montana, USA: ACM, 2009, pp. 73–86. ISBN: 978-1-60558-752-3. DOI: 10.1145/1629575.1629583. URL: http://doi.acm.org/10.1145/1629575.1629583.

[65] L. Ryzhyk, A. Walker, J. Keys, A. Legg, A. Raghunath, M. Stumm, and M. Vij. "User-guided device driver synthesis." In: *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*. 2014, pp. 661–676.

[66] C. Seceleanu, A. Vulgarakis, and P. Pettersson. "REMES: A Resource Model for Embedded Systems." In: *14th IEEE International Conference on Engineering of Complex Computer Systems*. June 2009, pp. 84–94. DOI: 10.1109/ICECCS.2009.49.

[67] P. SIMON and Y. GOGOTSI. "Materials for electrochemical capacitors." In: *Nanoscience and Technology*, pp. 320–329. DOI: 10.1142/9789814287005_0033. eprint: https://www.worldscientific.com/doi/pdf/10.1142/9789814287005_0033. URL: https://

www . worldscientific . com / doi / abs / 10 . 1142 / 9789814287005_0033.

[68] D. C. Snowdon, E. Le Sueur, S. M. Petters, and G. Heiser. "Koala: A platform for OS-level power management." In: *Proceedings of the 4th ACM European conference on Computer systems*. ACM. 2009, pp. 289–302.

[69] O. Spinczyk, A. Gal, and W. Schröder-Preikschat. "AspectC++: An Aspect-Oriented Extension to C++." In: *Proceedings of the 40th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Pacific '02)*. Sydney, Australia, Feb. 2002, pp. 53–60.

[70] O. Spinczyk and D. Lohmann. "The Design and Implementation of AspectC++." In: *Knowledge-Based Systems, Special Issue on Techniques to Produce Intelligent Secure Software* 20.7 (2007), pp. 636–651. DOI: 10 . 1016 / j . knosys . 2007.05.004.

[71] S. Steinke, M. Knauer, L. Wehmeyer, and P. Marwedel. "An Accurate and Fine Grain Instruction-Level Energy Model Supporting Software Optimizations." In: *PATMOS*. Yverdon (Switzerland), Sept. 2001.

[72] S. Sudevalayam and P. Kulkarni. "Energy Harvesting Sensor Nodes: Survey and Implications." In: *IEEE Communications Surveys Tutorials* 13.3 (Third 2011), pp. 443–461. ISSN: 1553-877X. DOI: 10.1109/SURV.2011.060710.00094.

[73] T. Tan, A. Raghunathan, and N. Jha. "Embedded operating system energy analysis and macro-modeling." In: *IEEE International Conference on Computer Design: VLSI in Computers and Processors.* 2002, pp. 515–522. DOI: 10.1109/ICCD.2002.1106822.

[74] J. Tanguy, J. Bechennec, M. Briday, S. Dube, and O. Roux. "Device driver synthesis for embedded systems." In: *IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA)*. Sept. 2013, pp. 1–8. DOI: 10.1109/ETFA.2013.6647951.

[75] Texas Instruments. *MSP430 Ultra-Low-Power MCUs*. May 2019. URL: http : / / www . ti . com / microcontrollers / msp430-ultra-low-power-mcus/overview.html.

[76] The Solar Doorplate Project Group. *Solar Doorplate- Energiegewahre Systemsoftware für Ubiquitäre Systeme*. May 2019. URL: https : / / ess . cs . tu - dortmund . de / DE / Teaching/PGs/solardoorplate/.

[77] TinyOS. *The TinyOS Homepage*. May 2019. URL: http://www.tinyos.net.

[78] A. Wang, B. H. Calhoun, and A. P. Chandrakasan. *Sub-threshold design for ultra low-power systems*. Vol. 95. Springer, 2006. DOI: https://doi.org/10.1007/978-0-387-34501-7.

[79] J. Wang, P. Liu, J. Hicks-Garner, E. Sherman, S. Soukiazian, M. Verbrugge, H. Tataria, J. Musser, and P. Finamore. "Cycle-life model for graphite-LiFePO4 cells." In: *Journal of Power Sources* 196.8 (2011), pp. 3942–3948. ISSN: 0378-7753. DOI: https://doi.org/10.1016/j.jpowsour.2010.11.134. URL: http://www.sciencedirect.com/science/article/pii/S0378775310021269.

[80] O. Wang and W. Yang. "Energy Consumption Model for Power Management in Wireless Sensor Networks." In: *SECON '07. 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks.* June 2007, pp. 142–151. DOI: 10.1109/SAHCN.2007.4292826.

[81] S. Wang and S. Malik. "Synthesizing operating system based device drivers in embedded systems." In: *Hardware/Software Codesign and System Synthesis, 2003. First IEEE/ACM/IFIP International Conference on*. Oct. 2003, pp. 37–44. DOI: 10.1109/CODESS.2003.1275253.

[82] A. Weder. "An Energy Model of the Ultra-Low-Power Transceiver nRF24L01 for Wireless Body Sensor Networks." In: *Second International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN)*. July 2010, pp. 118–123. DOI: 10.1109/CICSyN.2010.24.

[83] W. Xiao, W. G. Dunford, and A. Capel. "A novel modeling method for photovoltaic cells." In: *2004 IEEE 35th Annual Power Electronics Specialists Conference (IEEE Cat. No.04CH37551)*. Vol. 3. June 2004, 1950–1956 Vol.3. DOI: 10.1109/PESC.2004.1355416.

[84] N. Zhu and A. V. Vasilakos. "A generic framework for energy evaluation on wireless sensor networks." In: *Wireless Networks* 22.4 (2016), pp. 1199–1220. ISSN: 1572-8196. DOI: 10.1007/s11276-015-1033-x.

ACRONYMS

ADC analog to digital converter.
API application programming interface.

BMU battery measurement unit.

CPU central processing unit.

DC direct current.
DFS dynamic frequency scaling.
DUT device under test.
DVFS dynamic voltage and frequency scaling.

FSM finite-state machine.

GPU graphics-processing unit.

IoT Internet of Things.
IRQ interrupt request.
ISR interrupt service routine.

LCD liquid-crystal display.
LED light-emitting diode.

MCU microcontroller unit.
MNB master network board.
MOS metal-oxide semiconductor.
MPP maximum-power point.
MPPT maximum-power-point tracker.

PTA priced timed automata.
PVC photovoltaic cell.

RFID radio-frequency identification.
RMSE root-mean-square error.
RSSI received-signal-strength indicator.

SDP Solar Doorplate.
SMU source measurement unit.
SPI serial peripheral interface.
SSB swappable slave board.

TDM time-division multiplex.

TPS transiently-powered system.

ULP ultra-low power.

VFC voltage-to-frequency converter.

LIST OF FIGURES

LIST OF TABLES