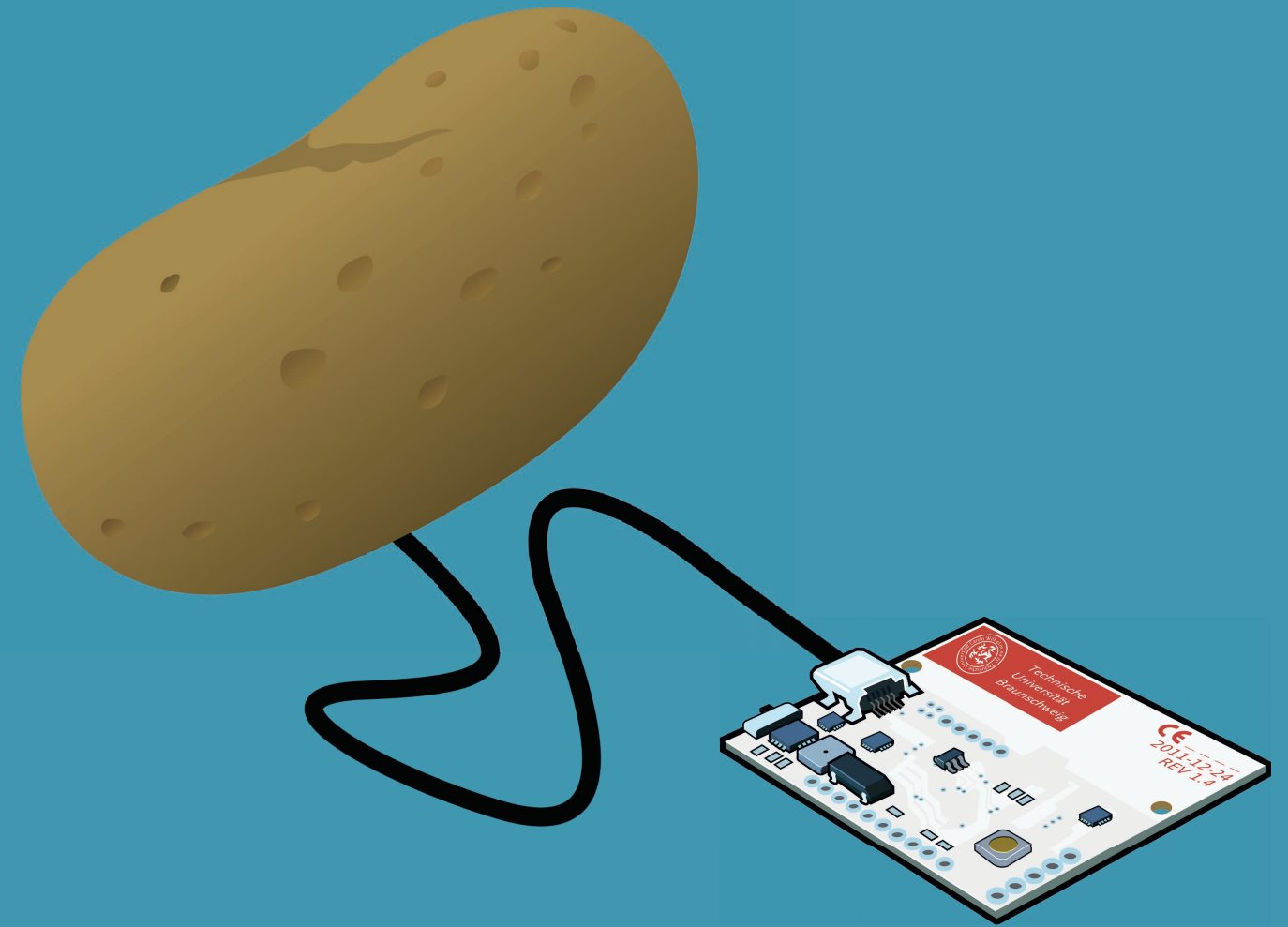


For almost 20 years the essential requirements when operating Wireless Sensor Networks (WSNs) are energy efficient sensing, processing and transmission.

However, existing works still takes a very conservative approach with large safety margins towards reliability. To reduce the power dissipation even further, undervolting is a more risk-oriented voltage scaling where components of a sensor node are operated below their recommended minimum voltage levels. Due to the exponential connection between the voltage level and the power dissipation of CMOS gates, undervolting offers huge energy-saving potentials but goes along with an increased risk of failures.

This thesis takes up this challenge and develops strategies which enable a highly efficient but dependable utilization of undervolting in WSNs.

On Reliable Undervolting in Wireless Sensor Networks



On Reliable Undervolting in Wireless Sensor Networks

Ulf Kulau



Technische
Universität
Braunschweig

On Reliable Undervolting in Wireless Sensor Networks

Von der
Carl-Friedrich-Gauß-Fakultät

der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines

Doktoringenieurs (Dr.-Ing.)

genehmigte Dissertation
(kumulative Arbeit)

von **Ulf Kulau**
geboren am **03.01.1984**
in **Rendsburg**

Eingereicht am: **02.06.2016**
Disputation am: **19.08.2016**
1. Referent: **Prof. Dr. Lars Wolf**
2. Referent: **Prof. Dr. Thiemo Voigt**

(2016)

Kurzfassung

Die langsame Entwicklung der Kapazität von Batterien auf der einen und die wachsende Anzahl von anspruchsvollen Anwendungen auf der anderen Seite erfordern Energiemanagement-Techniken auf allen Ebenen des WSN. Aus diesem Grund wurde seither viel Aufwand betrieben, um die Energieeffizienz von WSNs zu optimieren. Viele existierende Arbeiten gehen dabei jedoch sehr zurückhaltend vor und betonen stets die Stabilität, anstatt die Fehlertoleranz eines WSNs beziehungsweise der jeweiligen Anwendung auszunutzen.

Eine bekannte Methode zur Verlängerung der Laufzeit von Sensorknoten und WSNs ist die Anpassung der Versorgungsspannung. Der technische Hintergrund ist hierbei, dass beinahe alle Komponenten eines Sensorknotes auf der CMOS Technologie basieren. Da die dynamische Leistungsaufnahme von CMOS stark von der Versorgungsspannung abhängt, sorgt eine Anpassung der Spannung für eine erhöhte Energieeffizienz. Aus diesem Grund existieren diverse Ansätze, welche darauf abzielen die Spannung auf das spezifizierte Minimum zu reduzieren.

Um die Leistungsaufnahme weiter zu reduzieren, ist Undervolting ein eher risikoorientierter Ansatz, bei dem die Komponenten eines Sensorknotens mit Spannungen unterhalb ihrer Spezifikation betrieben werden. Durch den exponentiellen Zusammenhang zwischen Betriebsspannung und Verlustleistung der zugrunde liegenden CMOS-Technologie, bietet Undervolting ein enormes Energieeinsparpotential. Die erhöhte Energieeffizienz wird jedoch durch ein erhöhtes Risiko von Fehlern begleitet. In dieser Arbeit werden Strategien behandelt, welche eine effektive Nutzung von Undervolting in WSNs ermöglichen, ohne die Zuverlässigkeit der Anwendung einzuschränken.

Die Untersuchung von aktivem Undervolting auf drahtlosen Sensorknoten erfolgt dabei zunächst durch theoretische Betrachtungen und prototypische Projektstudien. Es zeigt sich, dass bei einer gleichbleibenden Spannung eine Korrelation zwischen der Temperatur und dem Auftreten von Fehlern besteht. Ideale Arbeitspunkte sind also stark Temperaturabhängig. Betrachtet man in diesem Zusammenhang WSNs im Außeneinsatz, so können Sensorknoten teilweise extremen Temperaturen ausgesetzt sein. Daher muss selbst für solch herausfordernde Umgebungsbedingungen eine zuverlässige Nutzung von aktivem Undervolting gewährleistet werden. Hierfür wurde ein Verfahren zum Lernen und Präzisieren sicherer Arbeitspunkte (überwachtes Lernen) entwickelt. Knoten sparen durch aktives Undervolting stets so viel Energie wie möglich ein und werden gleichzeitig in einem sicheren Betriebszustand gehalten. Diverse kleinere Experimente und Evaluationen betätigen dabei die zuvor getroffenen theoretischen Annahmen und zeigen die Machbarkeit und Effizienz dieses Ansatzes. Die Ergebnisse zeigen, dass Energieeinsparungen von bis zu 42% möglich sind.

Auch für Peripheriekomponenten eines Sensorknotens führt ein niedrigerer Spannungspegel zu einer geringeren Stromaufnahme. Eine regelbarer Peripheriespannungspfad ermöglicht daher Energieeinsparungen, indem die Spannung in Abhängigkeit der jeweilig aktiven Peripheriekomponente dynamisch angepasst wird. Hierfür wird ein Algorithmus vorgestellt, welcher die Mehrkosten zum Umschalten der Spannung gegenüber den Einsparungen abwägt. Während eine aktive Peripheriekomponente immer mit der empfohlenen Spannung betrieben wird, können inaktive Komponenten unterhalb ihrer Spezifikation versorgt werden. Unter Duldung dieses passiven Undervoltings kann die Nutzung von Peripheriekomponenten hocheffizient geplant werden, wodurch Energieeinsparungen von bis zu 47.9% möglich sind.

Schließlich wird mit Hilfe einer Testumgebung für 'Smart Farming' Anwendungen gezeigt, dass die entwickelten Ansätze (i) langfristig stabil sind, (ii) die Basiszuverlässigkeit von WSNs nicht beeinträchtigen und (iii) weitere Forschungsfelder im Bereich energieeffizienter aber zuverlässiger WSNs eröffnen.

Abstract

For almost 20 years the essential requirements when operating Wireless Sensor Networks (WSNs) are energy efficient sensing, processing and transmission. The limping evolution of batteries on the one hand and the growing field of challenging applications on the other necessitate advanced power-management techniques on all layers of a WSN. Hence, much effort has been put into optimizing the energy-efficiency of wireless sensor nodes, but existing works still takes a very conservative approach with large safety margins towards reliability. What these approaches do not exploit, is the fact that fault-tolerance is already built into most layers of a WSN and their applications respectively.

A well-known method to increase the lifetime of nodes and networks is voltage scaling. The background of voltage scaling techniques is that nearly all components of a sensor node are realized in CMOS technology. As the dynamic power dissipation of CMOS strongly depends on the supplied voltage level, an adaptation of the voltage level leads to an increased energy efficiency. Hence, there are several existing approaches which tend to adapt the voltage level to the specified minimum.

To reduce the power dissipation even further, undervolting is a more risk-oriented voltage scaling where components of a sensor node are operated below their recommended minimum voltage levels. Due to the exponential connection between the voltage level and the power dissipation of CMOS gates, undervolting offers huge energy-saving potentials but goes along with an increased risk of failures. This thesis takes up this challenge and develops strategies which enable a highly efficient but dependable utilization of undervolting in WSNs.

The exploration of active undervolting on WSN nodes starts with theoretical examinations and extensive measurements of a prototype implementation. It is shown that for a given voltage level there is a strong correlation between temperature and fault probability: Acceptable voltage levels strongly depend on environmental conditions. Considering outdoor deployments, WSNs can be exposed to extreme temperatures and experience sudden changes of temperature. A supervised learning strategy is presented which can learn and predict safe operating points under varying temperatures for an individual node. By dynamically adapting the voltage to the environmental conditions, nodes stay in stable operating states, while saving as much energy as possible. Several analysis and micro-experiments corroborate the theoretical assumptions and show the effectiveness of this approach. The results show that energy savings of up to 42% are achieved.

Beside the processing unit, even for peripheral devices a lower voltage level leads to a lower power dissipation. Providing an adjustable voltage path to the peripherals enables energy savings by adapting the voltage level depending on queried peripherals dynamically. For this purpose, a scheduling algorithm is presented which weights off the energy savings against the overhead for switching the voltage level. While an active peripheral is always supplied with its recommended voltage level, inactive peripherals might be operated outside their specification. Thus, passive undervolting of peripherals allows highly efficient schedules for arbitrary query-sequences. The evaluation of a prototype implementation showed that energy savings up to 47.9% are possible.

Finally with the aid of a real world testbed deployment for 'Smart Farming' applications it is shown that the developed approaches (i) can guarantee long-term stability, (ii) do not decrease the reliability of WSNs and (iii) open the door to further research on energy efficient but reliable WSNs.

List of Publications

2016

- [1] Robert Hartung, Ulf Kulau, and Lars Wolf. Distributed Energy Measurement in WSNs for Outdoor Applications. In *IEEE International Conference on Sensing, Communication and Networking Conference*, SECON 2016, London, UK, June 2016. Accepted
- [2] Ulf Kulau, Stephan Rottmann, Sebastian Schildt, Johannes van Balen, and Lars Wolf. Undervolting in Real World WSN Applications: A Long-Term Study. In *the 12th IEEE International Conference on Distributed Computing in Sensor Systems*, DCoSS 2016, Washington D.C., USA, May 2016
- [3] Ulf Kulau, Felix Büsching, and Lars Wolf. IdealVolting – Reliable Undervolting on Wireless Sensor Nodes. *ACM Transactions on Sensor Networks (TOSN)*, 12(2), April 2016

2015

- [4] Ulf Kulau, Tobias Breuer, and Lars Wolf. A Clustering-Based Characteristic Model for Unreliable Sensor Network Data. In *IEEE World Forum on Internet of Things*, WF-IoT 2015, Milan, Italy, December 2015
- [5] Ulf Kulau, Sebastian Schildt, Stephan Rottmann, and Lars Wolf. Paint it Black – Increase WSN Energy Efficiency with the Right Housing. In *the 6th ACM Workshop on Real World Wireless Sensor Networks*, RealWSN 2015, Seoul, South Korea, November 2015
- [6] Ulf Kulau, Sebastian Schildt, Stephan Rottmann, Björn Gernert, and Lars Wolf. PotatoNet – Outdoor WSN Testbed for Smart Farming Applications. Technical Report CS-2015-06, Erlangen, Germany, September 2015
- [7] Ulf Kulau, Sebastian Schildt, Stephan Rottmann, Björn Gernert, and Lars Wolf. Demo: PotatoNet – Robust Outdoor Testbed for WSNs: Experiment Like on Your Desk. Outside. In *the 10th ACM MobiCom Workshop on Challenged Networks*, CHANTS 2015, Paris, France, September 2015
- [8] Ulf Kulau, Stephan Friedrichs, and Lars Wolf. Demo: Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes. In *the 12th European Conference on Wireless Sensor Networks*, EWSN 2015, Porto, Portugal, February 2015

2014

- [9] Ulf Kulau, Felix Büsching, and Lars Wolf. Undervolting in WSNs: Theory and Practice. *Internet of Things Journal, IEEE*, 2(3):190–198, June 2015
- [10] Felix Büsching, Keno Garlich, Johannes van Balen, Yannic Schröder, Kai Homeier, Wolf-Bastian Pöttner Ulf Kulau, Stephan Rottmann, Sebastian Schildt, Georg von Zengen, and Lars Wolf. Quadrotor-based DT-WSNs for Disaster Recovery. Technical Report TR-2014-1, Potsdam, Germany, September 2014

- [11] Ulf Kulau, Sebastian Schildt, Georg von Zengen, Stephan Rottmann, and Lars Wolf. Ballistic Deployment of WSN Nodes using Model Rockets. In *the 6th Extreme Conference of Communication*, ExtremeCom 2014, Galapagos Islands, Ecuador, August 2014
- [12] Stephan Friedrichs, Ulf Kulau, and Lars Wolf. Energy-Efficient Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes. In *IEEE International Conference on Communications Workshop on Energy Efficiency in Wireless Networks & Wireless Networks for Energy Efficiency*, E2Nets 2014, Sydney, Australia, June 2014
- [13] Ulf Kulau, Felix Büsching, and Lars Wolf. Undervolting in WSNs – A Feasibility Analysis. In *IEEE World Forum on Internet of Things 2014*, WF-IoT 2014, Seoul, South Korea, March 2014

2013

- [14] Ulf Kulau, Felix Büsching, and Lars Wolf. A Node's Life: Increasing WSN Lifetime by Dynamic Voltage Scaling. In *the 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013*, DCoSS 2013, Cambridge, USA, May 2013
- [15] Ulf Kulau, Felix Büsching, Wolf-Bastian Pöttner, and Lars Wolf. Demo: A Node's Life – Increasing WSN Lifetime by Dynamic Voltage Scaling. In *the 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013*, DCoSS 2013, Cambridge, USA, May 2013

2012

- [16] Felix Büsching, Ulf Kulau, and Lars Wolf. Architecture and Evaluation of INGA - An Inexpensive Node for General Applications. In *IEEE Sensors*, Sensors 2012, Taipei, Taiwan, October 2012
- [17] Felix Büsching, Ulf Kulau, Matthias Gietzelt, and Lars Wolf. Comparison and validation of capacitive accelerometers for health care applications. *Computer Methods and Programs in Biomedicine*, 106(2), May 2012

2011

- [18] Felix Büsching, Ulf Kulau, and Lars Wolf. Demo: INGA - An Inexpensive Node for General Applications. In *the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys 2011, Seattle, WA, USA, November 2011
- [19] Ulf Kulau, Felix Büsching, Wolf-Bastian Pöttner, and Lars Wolf. INGA - Architektur eines universell einsetzbaren Sensorknotens. Technical Report TR-RI-11-313, Paderborn, Germany, September 2011

Contents

1	Introduction	1
1.1	Publications	2
1.2	Contributions	3
1.3	Outline	4
2	Terminology and Technical Background	7
2.1	Wireless Sensor Networks (WSNs) and Nodes	7
2.2	Complementary Metal-Oxide Semiconductor (CMOS)	9
2.2.1	Technology	9
2.2.2	Power Dissipation	9
2.3	Voltage Scaling Techniques	10
2.3.1	Efficiency of Voltage Conversion	11
3	Active Undervolting of MCUs in WSNs	13
3.1	Theoretical Consideration	13
3.2	Lab-based Studies	14
3.2.1	Prototype Implementation and Bench Test	14
3.2.2	Characteristics and Results	16
3.3	IdealVolting – Reliable Undervolting on every Node	17
3.3.1	Undervolting Capable Sensor Node	17
3.3.2	Iterative Self-Optimization	18
3.3.3	Supervised Learning Approach	19
3.3.4	Results	20
4	Passive Undervolting of Peripherals	23
4.1	Peripheral Voltage Scheduling Problem	23
4.2	Implementation and Results	25
5	Use of Active Undervolting in Real WSN Deployments	27
5.1	PotatoNet – Outdoor WSN Testbed	27
5.1.1	Distributed Energy Measurement	28
5.2	Long-Term Study and Results	28
5.3	Temperature-Dependent Energy Efficiency	29
6	Conclusions	31
6.1	Future Work	32
	Bibliography	33
	List of Acronyms	39
	Publications	41
	Paper 1: Undervolting in WSNs: Theory and Practice (appeared in IEEE Internet of Things Journal Volume 2, Issue: 3)	43

Paper 2: IdealVolting – Reliable Undervolting on Wireless Sensor Nodes (appeared in ACM Transactions on Sensor Networks Volume 12, Issue 2, April 2016)	53
Paper 3: Energy-Efficient Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes (appeared in Proceedings of IEEE International Conference on Communications Workshops (E2Nets 2014))	91
Paper 4: Demo Abstract: Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes (appeared in European Conference on Wireless Sensor Networks (EWSN 2015))	97
Paper 5: Paint it Black – Increase WSN Energy Efficiency with the Right Housing (appeared in ACM Workshop on Real World Wireless Sensor Networks (RealWSN 2015))	99
Paper 6: Demo: PotatoNet – Robust Outdoor Testbed for WSNs (appeared in ACM MobiCom Workshop on Challenged Networks (CHANTS 2015))	103
Paper 7: Undervolting in Real World WSN Applications: A Long-Term Study (accepted for publication in IEEE Conference on Distributed Computing in Sensor Systems (DCoSS 2016))	105
Paper 8: Distributed Energy Measurement in WSNs for Outdoor Applications (accepted for publication in IEEE Conference on Sensing, Communication and Networking (SECON 2016))	113

1 Introduction

The charm of Wireless Sensor Networks (WSNs) is their ability to allow large-scale distributed sensing without any existing infrastructure like cable-based links or a power-grid. The WSN itself is therefore composed of sensor nodes which are able to sense, process and transmit data wirelessly. In some scenarios a WSN is also equipped with actors so that objects can be sensed as well as controlled remotely. By integrating such capabilities to the Internet, the WSNs are a vital component of the upcoming Internet of Things (IoT). Recently, emerging IoT technologies such as Smart Homes, Smart Cities or Smart Agriculture heavily influenced classical WSN research by tending to real world applications. The exponentially growing interest for IoT solutions can be deduced from forecasts like [20, 21] and emphasizes the importance of this trend. For example Intel prognosticates that around 200 billion 'things' will be interconnected by the year 2020 [22]. This huge amount of sensing devices leads to several forthcoming challenges but there still remains a basic requirement – energy efficiency.

For most WSN or IoT applications a local energy source is required. Commonly, batteries are used for this purpose. In addition, nodes can be supported by energy harvesting techniques [23], but the gathered energy is often not sufficient to power the nodes properly. However, the limited capacity and limping evolution of this energy source, as well as the intention to operate WSNs for long lifetimes lead to the fact that energy is one of the scarcest resource of WSN nodes [24, 25]. In further consequence, the entire hard- and software design of WSNs has been driven by the requirement of low power. This is why a typical wireless sensor node, consisting of a processing, a sensing and a transceiver unit, has limited computational capabilities and throughput capacities. Experiences with real world WSN deployments have shown that WSNs can be rather unreliable networks. For this reason past and recent research on WSNs is focused on achieving both long lifetimes while ensuring reliability and fault tolerance by software [26]. However, many WSN software stacks and applications do not assume absolute dependability of the system but would benefit from significantly increased lifetime. Thus, in order to reduce the energy consumption of sensor nodes significantly, it is time to rethink the constraint of complete dependability of the lower layers. In a typical WSN setup the higher layers of the communication and software stack already implement measures to deal with unreliability and faults in the system.

For this reason many WSN applications would benefit from the approaches presented in this thesis. Exemplarily, this thesis deals with 'Smart Farming' applications including a testbed deployment on a trial field of a potato research station. The use of WSNs in agriculture is an upcoming application area where the features of sensor networks can be used to analyze and monitor the conditions of the soil and crops in order to optimize yield and minimize the use of fertilizer. The following three facts condense why Smart Farming was selected as the exemplary scenario for this thesis:

1. WSNs in agriculture are chiefly located in rural areas. The sensor nodes have to rely on autonomous energy sources due to the non-availability of any infrastructure.
2. Monitoring of e.g. plant growth requires long lifetimes of months or even years while the WSN is mainly deployed outdoors and therefore exposed to rough environmental conditions.
3. Smart Farming applications are less critical as they depend largely on sensing. Perfect operation of the WSN task is desired, but not a hard requirement. Comprehensive long-term sensing is more important than the rare absence of a measurement sample.

By acknowledging the fault-tolerant character of both the WSNs and the applications running on the WSN, more daring power-management techniques are motivated. Of course, efficient protocols have always been a major research topic of WSNs and lead to a plethora of publications [27]. Nevertheless, existing power management techniques have been designed conservatively and do not consider the fault-tolerant character of WSNs and applications. Especially for voltage scaling mechanisms *undervolting* – the operation of WSN nodes below their recommended voltage levels – offers a huge potential but at an increased risk of failures. In this thesis the terms *active* undervolting and *passive* undervolting describe paradigms how the tolerances of parts as well as the WSN can be utilized to enable more energy-efficient but still reliable WSN applications.

Here, active undervolting describes the violation of specified voltage levels while a Micro Controller Unit (MCU) is in its active state. The initial legitimization for this is theoretically derived by considering the temperature dependency of underlying Complementary Metal-Oxide Semiconductor (CMOS) technology. The fact that the threshold voltage of CMOS parts is strongly dependent on the environmental temperature, but a widespread temperature range of MCUs is generally covered by a fixed voltage level, is further discussed within this cumulative dissertation. In lab-based studies it is shown that not only the temperature but also individual characteristics lead to the fact that the absolute minimum voltage levels differ for every single MCU. Thus, an adaptive supervised learning approach is introduced – named *IdealVolting*. *IdealVolting* is executed on every sensor node so that a balance between reliable processing and the most energy-efficient state can be achieved for each MCU individually. As a consequence, *IdealVolting* adds heterogeneity to the WSN in terms of energy efficiency. In outdoor applications like Smart Farming, where nodes can be exposed to extreme temperature conditions, temperature-dependent energy efficiency is introduced by this undervolting scheme.

Passive undervolting instead describes the process when components that rest in an inactive operating state are set below their recommended voltage level. A rather less considered part of sensor nodes is the sensing unit, but the peripheral energy consumption could easily dominate that of an entire node. For this purpose, passive undervolting can help to increase the energy efficiency of peripherals. Like MCUs the energy efficiency of sensors also benefit from a downscaled voltage level. Therefore, this thesis describes a voltage level scheduling approach for peripherals. An optimal solution is possible by weighting the overhead for voltage switching against the profit in energy consumption when using the actual needed voltage level for operating a peripheral. This method guarantees that active peripherals are never undervolted, but relies on undervolting the inactive ones.

In any case the intention of all presented approaches is not to solely rely on the fallback of the fault-tolerance of the WSNs software stack. It will be shown how undervolting schemes can be used reliably while significant energy savings can be achieved at the same time. The feasibility, usability and functionality is shown within the scope of a real-world application and lays down the groundwork for further optimizations on several higher layers in WSNs.

1.1 Publications

This cumulative dissertation makes several contributions to the research area of WSNs by using undervolting on system level to raise the energy efficiency of sensor nodes and networks significantly. The particular contributions are contained in the following eight scientific publications both in conference proceedings and journals. Those eight publications are contained in this document and are listed (with their respective page number) in the following:

1. Ulf Kulau, Felix Büsching, and Lars Wolf. Undervolting in WSNs: Theory and Practice. *Internet of Things Journal, IEEE*, 2(3):190–198, June 2015 (on page 43)
2. Ulf Kulau, Felix Büsching, and Lars Wolf. *IdealVolting* – Reliable Undervolting on Wireless Sensor Nodes. *ACM Transactions on Sensor Networks (TOSN)*, 12(2), April 2016 (on page 53)

3. Stephan Friedrichs, Ulf Kulau, and Lars Wolf. Energy-Efficient Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes. In *IEEE International Conference on Communications Workshop on Energy Efficiency in Wireless Networks & Wireless Networks for Energy Efficiency*, E2Nets 2014, Sydney, Australia, June 2014 (on page 91)
4. Ulf Kulau, Stephan Friedrichs, and Lars Wolf. Demo: Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes. In *the 12th European Conference on Wireless Sensor Networks*, EWSN 2015, Porto, Portugal, February 2015 (on page 97)
5. Ulf Kulau, Sebastian Schildt, Stephan Rottmann, and Lars Wolf. Paint it Black – Increase WSN Energy Efficiency with the Right Housing. In *the 6th ACM Workshop on Real World Wireless Sensor Networks*, RealWSN 2015, Seoul, South Korea, November 2015 (on page 99)
6. Ulf Kulau, Sebastian Schildt, Stephan Rottmann, Björn Gernert, and Lars Wolf. Demo: PotatoNet – Robust Outdoor Testbed for WSNs: Experiment Like on Your Desk. Outside. In *the 10th ACM MobiCom Workshop on Challenged Networks*, CHANTS 2015, Paris, France, September 2015 (on page 103)
7. Ulf Kulau, Stephan Rottmann, Sebastian Schildt, Johannes van Balen, and Lars Wolf. Undervolting in Real World WSN Applications: A Long-Term Study. In *the 12th IEEE International Conference on Distributed Computing in Sensor Systems*, DCoSS 2016, Washington D.C., USA, May 2016 (on page 105)
8. Robert Hartung, Ulf Kulau, and Lars Wolf. Distributed Energy Measurement in WSNs for Outdoor Applications. In *IEEE International Conference on Sensing, Communication and Networking Conference*, SECON 2016, London, UK, June 2016. Accepted (on page 113)

1.2 Contributions

The individual contributions of this cumulative dissertation in general and of the publications listed in the previous section can be given as follows:

- **Basics about the characteristics of active undervolting on wireless sensor nodes:** The first contribution is a basic study about the fundamental characteristics of undervolting in WSNs. Beginning with theoretical considerations about the connection between temperatures and threshold voltages of CMOS gates, the general feasibility of undervolting parts of a sensor node is motivated. As manufacturers do not provide any information about the internal structure of their components (e.g., Register-Transfer Level (RTL)), it is infeasible to estimate undervolting effects in advance. Thus, a bench scale unit was implemented to characterize the influence of lower-than-recommended voltage levels, particularly on the MCU. In further considerations, the impact of undervolting on the power consumption of the wireless communication is also taken into account and examined in several experiments. The theoretical background as well as all experimental results are given in this thesis and in the related publications [9].
- **Dependable active undervolting of MCUs in WSNs:** The second contribution is a self-optimizing and supervised-learning approach to enable undervolting as a plug & play power-management mechanism for wireless sensor nodes. Different characteristics due to manufacturing variances as well as the temperature dependency of the absolute minimum voltage levels require a process that is capable to handle the individual properties of every node. For this purpose a control-loop consisting of a trustworthy validator and the undervolted, thus potentially unreliable, MCU is used. On top of that a supervised-learning strategy is presented which constantly keeps nodes in stable states even for rough environmental conditions where temperatures reach extreme limits and are highly fluctuating. The entire approach – called IdealVolting

– is described and accompanied by several micro-experiments and longitudinal studies within this thesis and in the related publication [3].

- **Optimal scheduling of peripherals voltage level:** The third contribution is an approach to increase the energy efficiency of peripheral components by applying voltage scaling. Even peripheral devices can benefit from a downscaled voltage level, but many components require a different minimum supply voltage. Hence, a dedicated voltage path for the sensing unit of a node is introduced to adapt the voltage level on demand. However, since voltage switching incurs some additional overhead it is not trivial to decide when a voltage change is beneficial. To solve this issue a fully self-optimizing approach is presented within this thesis. Using dynamic programming a scheduling algorithm weighs off the benefits due to lower voltage levels against the mentioned switching overhead. The theoretical formulation of the optimization problem, the solving algorithm itself and a practical implementation show that voltage scheduling is always optimal and outperforms current approaches. While an active peripheral is always powered by its specified voltage level, the inactive ones might be passively undervolted. This aspect is also addressed and becomes an individual decision whether reliability or energy efficiency is desired. The feasibility of this novel approach is experimentally verified and results are given in the thesis and in the related publications [12, 8].
- **Robust versatile outdoor testbed for wireless sensor nodes:** The fourth contribution is *PotatoNet* a generic outdoor testbed for WSNs in the light of an exemplary Smart Farming application. While there are a plethora of WSN testbeds, PotatoNet is unique in its combination of strengths: It can be deployed in rural areas in a wide range of climate conditions, while offering the same monitoring and troubleshooting capabilities as a desk setup. All aspects of the setup can be controlled remotely and it is virtually impossible to brick a WSN node by flashing bad firmware on it. The intention for the PotatoNet was to perform thorough evaluations of the undervolting schemes presented in this thesis. Thus, the PotatoNet is equipped with undervolting capable sensor nodes so that this testbed is the first of its kind. More details and deployment experiences are given in this thesis and in the related publications [7, 2, 1].
- **Extensive proof of the reliable utilization of undervolting in WSNs:** The fifth contribution is the validation that active undervolting scheme presented in this thesis can be used as a regular power-management technique *without* affecting the baseline reliability of real a WSN. Based on long-term evaluations the characteristics and gains are shown and discussed in detail. The experiments and their results are given in this thesis and in the related publications [2, 5].

1.3 Outline

This extended overview of the cumulative dissertation is meant to be a summary of the publications listed in section 6.1. A full list of publications by the author is given on page vii of this document. Those publications that are considered as an essential part of this work are contained in the dissertation and start on page 41. The thesis makes explicit references to those papers to highlight their context and the relation between the publications. The rest of this thesis is structured as follows:

To provide a common basis Section 2 gives an overview over the terminology used in this thesis and provides some technical background including CMOS technology, voltage scaling mechanisms and WSNs. In Section 3 active undervolting on WSN nodes is introduced. Beginning with lab-based studies the entire process of development towards a reliable self-optimizing undervolting is described. Furthermore, several analysis and measures are discussed to explore every design point. In Section 4 a voltage scheduling strategy for peripherals is introduced where parts might be passively undervolted. The derivation of a scheduling algorithm as well as a practical implementation show the effectiveness of this approach. An outdoor testbed implementation and extensive evaluations of undervolting in

real world WSN applications are part of Section 5. Finally, before the research papers are presented, a summary of this extended overview and a short outlook on future work is given in Section 6.

2 Terminology and Technical Background

For more than 20 years WSN have been an ongoing topic within academic and industrial research. Nevertheless, the terminology has diversified over years. For example Wireless Sensor Actor Networks (WSANs) turn into Cyber Physical Systems (CPS) or the integration of WSNs to the Internet is a vital part of the ever-growing IoT (cf. Section 1). Considering the energy-efficiency of WSNs a plethora of scientific contributions have been published and therefore many technical terms have been originated.

To pave common ground, this section defines the main terms that are used throughout this thesis. In addition, some technical details are addressed in greater depth, as the appended papers require basic knowledge about the power-dissipation of micro-electronic components. It should be mentioned that the given definitions and basics are neither complete nor comprehensive and are therefore limited to the scope of this thesis.

2.1 Wireless Sensor Networks (WSNs) and Nodes

A Wireless Sensor Network (WSN) is a distributed sensing system which is built of at least two nodes to enable applications such as environmental sensing or industrial process monitoring. With the ability of wireless communication, sensor nodes form a network infrastructure so that gathered data can be collaboratively forwarded to one or multiple sinks. The logical topology of the WSN depends on many factors but mainly on the spatial distribution and the communication radius of each node. In literature WSNs are frequently hypothesized with several hundreds or thousands of nodes, but in practice such large deployments are unusual and even scarce in academic research [28, 29]. The design of the nodes themselves is often driven by the specific requirement of a given application but low-power and low-cost are common principles. Thus, WSNs are rather unreliable networks, as nodes tend to be error-prone due to cheap hardware or energy constraints.

Typically, a sensor node can be described as a tiny computing platform with sensing and communication capabilities. A block diagram of the architecture of a wireless sensor node is given in the following Figure 2.1 followed by a brief description of the composed units:

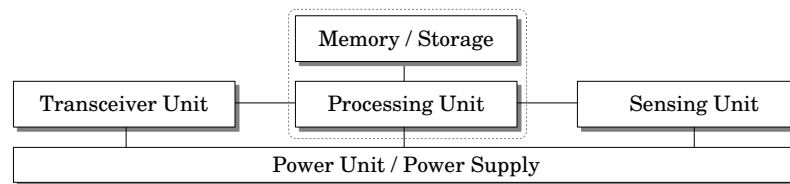


Figure 2.1: Generic composition of a wireless sensor node.

Processing Unit: This central component is responsible for controlling the entire node, data pre- and post-processing as well as for any communication-related logic (e.g., routing protocols). Commonly MCUs are used which combine a Central Processing Unit (CPU) with program memory (ROM), volatile Random-Access Memory (RAM) and various functional blocks like General Purpose Input/Outputs (GPIOs), Analog-Digital Converters (ADCs) or hardware timers. The variety of MCUs in terms of configuration and computational performance is huge. However, with regard

to the requirements of actual applications and energy constraints, their computational power is far below that of a Personal Computer (PC) or smart-phone. Instead, Ultra Low-Power (ULP) MCUs with just a few Million Instructions Per Second (MIPS) are considered to strike a balance between application-specific requirements and energy-efficiency.

Nevertheless, the MCU is active and in a high-power state for comparatively large amounts of time because it has to execute the entire software and is responsible for other units of the node. To achieve an energy-efficient usage, MCUs are usually equipped with Dynamic Power Management (DPM) in the form of several low-power sleep states. More advanced in-circuit power-management techniques like voltage scaling (cf. Section 2.3) or caching [30] are fairly rare in this performance class.

The nodes (INGA v1.5 [3] and INGA v1.6.1 [7, 2]) which are considered within this thesis and throughout the corresponding publications is based on an Atmel ATmega1284p MCU [31]. This is an 8-bit MCU with 128 kB ROM, 16 kB RAM runs at a default clock rate of $f_{cpu} = 8$ MHz.

Transceiver Unit: The transceiver unit provides wireless connectivity to a node, and thus the ability to cooperate and exchange data with other nodes. Simplistic early stream-based radios left a majority of processing to the MCU. Error-checking like Cyclic Redundancy Check (CRC), Medium Access Control (MAC) or addressing have to be implemented in software. In contrast, more recent packet-based radios implement such features in hardware and allow the transmission and reception of entire data frames. For WSN often the IEEE 802.15.4 standard [32] is used. It specifies the Physical layer (PHY) and MAC layers and a variety of compliant wireless transceivers are available from different vendors.

Due to the focus on energy-efficiency the output power of the RF signal is usually reduced to a few dBm. In further consequence, the communication range as well as the data rate is limited. Yet, while the transceiver is in transmit or receive state this unit tends to be the most power-consuming part of a node [25].

Within the appended publications of this thesis, mainly Atmel AT86RF231 and AT86RF233 transceivers were deployed [33, 34]. These IEEE 802.15.4 packet-based radios operate in the 2.4 GHz Industrial, Scientific and Medical (ISM) band and offer a line-of-sight communication range up to 250 m with the appropriate antenna [16].

Sensing Unit: A sensor is a device which translates physical parameters like temperature, pressure or humidity into equivalent, usually electrical, signals. These signals can be digitized using an ADC to enable the processing of sensed data. Nowadays the analog sensing and ADC are often fully integrated on-chip, so that sensors offer a dedicated digital interface such as Inter-Integrated Circuit (I²C) or Serial Peripheral Interface Bus (SPI). The actual application determines the set of peripherals that are placed on a node, whereas many applications require multiple sensors capturing or processing different physical quantities.

Power Supply / Power Unit: Most deployments use batteries to power the nodes whenever a permanent energy supply is not available. Due to the limited capacity of batteries and the intention to operate WSNs unattended for long periods, the power unit is a critical part of a node. Compared to the growth of computational capabilities the evolution of batteries regarding energy density is rather limping. An alternative to batteries are supercapacitors [35]. Indeed capacitors have lower energy density than batteries but offer several advantages such as faster recharge or environmental friendliness [36, 37]. Power-sources of WSN nodes are often combined with energy harvesting mechanisms to compensate the inefficiency of energy storage [23]. Beside the capacity and performance of the energy source, efficiency of voltage conversion as well as a proper power management are also of high importance.

2.2 Complementary Metal-Oxide Semiconductor (CMOS)

The most commonly used technology for Integrated Circuits (ICs) is CMOS. Modern CPUs, MCUs or other digital circuits are composed of logical gates that are constructed using this technology.

2.2.1 Technology

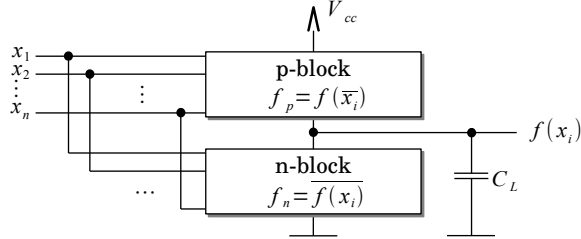


Figure 2.2: General architecture of CMOS gates.

In Figure 2.2 the general architecture of a CMOS gate is depicted. A logical function $f(x_i)$ is realized by combining a p-block ($f_p = f(\overline{x_i})$) and an n-block ($f_n = \overline{f(x_i)}$). The following example of an NOR-function shows how the logic of the blocks is derived:

$$f(x_1, x_2) = \overline{x_1 \vee x_2} \rightarrow \begin{cases} f_p = \overline{\overline{x_1} \vee \overline{x_2}} = x_1 \wedge x_2 \\ f_n = \overline{x_1 \vee x_2} = \overline{x_1} \wedge \overline{x_2} \end{cases} \quad (2.1)$$

The p-block is built of PMOS Field-Effect Transistors (FETs) and realizes f_p while an n-block is built of NMOS FETs and realizes f_n respectively. Hence, a logical CMOS gate is always self-locking and only a leakage current flows from V_{cc} to GND during static operations.

2.2.2 Power Dissipation

The power dissipation of CMOS gates is split into three parts [38].

1. The *leakage power* P_{lkg} occurs because the gate oxide (SiO_2) of the FET is so thin, that electrons can tunnel through it. Nowadays the problem is that on the one hand the size of FETs shrinks on the other hand the amount of FETs per chip increases. Together the leakage current sums up to a comparatively small but noticeable value. Especially applications with an extremely low duty cycle can suffer from the leakage current, as P_{lkg} becomes dominant [39].
2. The *short-circuit power* P_{srt} occurs for a short period of time when both P-Channel Metal-Oxide Semiconductor (PMOS) and N-Channel Metal-Oxide Semiconductor (NMOS) FETs are on during changes of state. Hence, P_{srt} depends on the slew rate of the signal that changes.
3. The *dynamic power dissipation* P_{dyn} is caused by discharging and charging of the parasitic output capacitor C_L (cf. Figure 2.2) [40]. This happens whenever the output signal of a gate changes. A change from logical 0 to 1 causes a charge of C_L and will therefore evoke a current I_{C_L} :

$$I_{C_L} 0 \rightarrow 1 = C_L \cdot \frac{dV_{cc}}{dt} \quad (2.2)$$

A logical 1 corresponds to V_{cc} . Assuming a continuous toggling signal at the output with a

period of T_p and a duty factor of 50%, the power dissipation for a single period is given by:

$$P_{dyn} = \frac{1}{T_p} \int_0^{T_p} I_{C_L} \cdot V_{cc} dt = \frac{1}{T_p} \int_0^{V_{cc}} C_L \cdot V_{cc} dV_{cc} = \frac{C_L \cdot V_{cc}^2}{T_p} \quad (2.3)$$

As the clock rate f is the inverse of the period T_p the dynamic power dissipation results in:

$$P_{dyn} = C_L \cdot f \cdot V_{cc}^2 \quad (2.4)$$

Beside a linear dependency on the clock rate the dynamic power dissipation shows a strong (quadratic) dependency on the supplied voltage level. This connection is the starting point for many optimizations and in particular for this thesis.

All in all it depends on the used CMOS technology and the application area how the power dissipation between P_{lkg} , P_{srt} and P_{dyn} is partitioned. For the MCUs normally used in the area of WSNs it is still common that the dynamic power dissipation P_{dyn} dominates the entire power dissipation ($P_{dyn} \gg P_{lkg} + P_{srt}$). For this reason MCUs provide several DPM mechanisms [41]. During idle times the MCU can be set to more energy-efficient sleep modes by register calls. The intention is to reduce the duty cycle and thus the time with dynamic power dissipation.

Clock gating is also a useful method to reduce the overall power consumption of an MCU. Parts of an MCU can be switched off by intercepting the supply clock [42]. Thus, the dynamic power dissipation for this part tends to zero (cf. Equation 2.3).

2.3 Voltage Scaling Techniques

As explained in the previous section, adaptation of the voltage level can offer significant energy saving potentials. Hence, there is a plethora of voltage scaling techniques but in the scope of this thesis only those related to WSNs are considered.

Dynamic Voltage Scaling (DVS) is a well-known technique to adapt both the clock rate f and the corresponding voltage level to a given workload. The connection between the clock rate and the voltage level is due to the switching delay of CMOS gates which grows with a lower voltage level. Thus, to guarantee that a switching operation finishes within $\frac{1}{f}$, a minimum voltage level is needed by a respective clock rate. The following Figure 2.3 summarizes the principle of DVS and depicts the difference to DPM mechanisms:

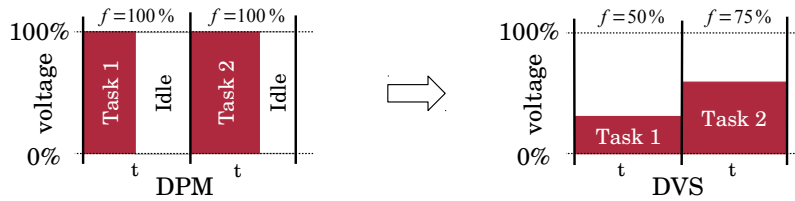


Figure 2.3: Function principle of DVS techniques compared against DPM approaches.

While MCUs using DPM run tasks to completion at a fixed clock rate and voltage level, DVS scales down the clock rate to meet a predefined deadline and adapts the voltage level accordingly. DVS has also been considered for WSN in several publications [43, 44]. A few publications proposed an implementation on the system level [14, 45]. It has been shown that voltage scaling has a positive impact on the battery lifetime of nodes and networks. The perspective of this thesis is that existing approaches still act too conservative with significant operation reserves. Considering safety margins

and the fault-tolerant character of WSNs, there is a huge potential for even further energy savings by using more daring voltage scaling.

2.3.1 Efficiency of Voltage Conversion

The ability to control the voltage level by software is the premise for active voltage scaling on sensor nodes. The power unit of a node has to be equipped with additional hardware to convert the voltage level of the power source to voltages suitable for the different components on the node. The overhead of such a *voltage scaling module* should be as low as possible. As the power unit is one part of the sensor node where sleep states are not recommended, every additional power dissipation of the voltage scaling circuitry has to be compensated by the expected yield. In general two losses have to be considered for a voltage scaling module 1. the quiescent current I_q and 2. the level of efficiency η . On the one hand Low Dropout Linear Converters (LDOs) offer an almost negligible I_q but the efficiency depends on the difference between input- and output voltage:

$$\eta_{ldo} = \frac{V_{out}}{V_{in}} \quad (2.5)$$

Thus, voltage conversion by using LDOs is inefficient for many cases where the voltage level of the power source (e.g. battery) is high compared to the needed output voltage. On the other hand step-down (buck) converters are more efficient. By rapidly switching the power source, the duty cycle defines the charge that is transferred to the actual load. Unfortunately buck converters have higher quiescent currents, so that it depends on the duty cycle of the WSN application whether a buck or an LDO is the more efficient choice.

With regard to the different efficiencies in voltage conversions, the savings in current consumption are a more meaningful metric. Thus, for the sake of fairness this thesis and the appropriated papers mainly limit to the current consumption to demonstrate the savings due to undervolting. However, with high efficient voltage conversion even more savings are possible.

3 Active Undervolting of MCUs in WSNs

The previous section has shown that the adaptation of the voltage level has a huge impact on the energy efficiency of CMOS-based components. Thus, voltage scaling mechanisms offer the potential to prolong the lifetime of WSNs or even enable energy neutral sensor nodes [46]. However, existing approaches have in common that they are designed conservatively. In terms of reliability the design of ordinary approaches is focused on node level. The hardware specifications of the particular component define the constraint for the voltage scaling. If we look at the reliability of an overall WSN instead of the reliability of a single node only, we can make the following observations:

- A WSN consists of many nodes. These nodes exhibit various individual characteristics that are less considered by the specifications (which are specified conservatively to cover the whole range of acceptable parts).
- The reliability of a single node is less important if the whole networks still assures proper execution of the WSN application task with the desired Quality of Service (QoS).
- Threshold voltages of CMOS-based components of the sensor nodes are temperature-dependent but only a fixed voltage level is specified to cover the entire temperature-range. However, the operating conditions of WSNs do not reach the stated extreme temperatures. Hence, nodes must be able to run below their specified voltage level under 'normal' conditions whereas higher temperatures imply further lowering of the voltage level.

This motivates an approach where more energy can be saved when the voltage level is further reduced, even below the recommended minimum given by the specification. In this thesis the deliberately undercutting of the recommended voltage level is defined as *active undervolting*.

The idea of undervolting is not new and was mentioned in some articles regarding general computer architectures [47]. In [48], undervolting is explicitly specified as a technique, which reduces the power dissipation of a Power7 CPU. However, up to now active undervolting has not been applied to small MCUs as part of a distributed system that is further exposed to rough environmental conditions, in particular WSNs.

The discussion in this section is primarily based on the following publications: [9, 3]. Those publications are part of this cumulative dissertation and hence are included in this document.

3.1 Theoretical Consideration

The fundamental assumptions about the feasibility of undervolting on WSN nodes and their validation are published in [9, 3] (see pages 43 and 53).

In many WSN applications nodes are powered by a battery directly. The batteries discharge and at the end of the lifetime it may happen that the voltage level drops below the recommended level for, e.g., the MCU. Thus, undervolting happens quasi accidentally but it can be observed that such undervolted nodes still operate properly for a certain amount of time. This observation about the potential feasibility of undervolting in WSNs is only a weak criterion to legitimate active undervolting.

Instead, an important aspect to be considered for this is the temperature-dependent threshold voltage (V_{th}) of CMOSs gates. Equation 3.1 shows the connection between the temperature and V_{th} at

which the CMOS gate alters its logic level (whereas V_{th0} is the threshold voltage at temperature T_0 and α is a temperature coefficient) [49, 50].

$$V_{th}(T) = V_{th0} + \alpha \cdot (T - T_0) \quad (3.1)$$

At low temperatures a higher voltage level is required whereas higher temperatures allow lower voltage levels. Unfortunately the recommended voltage levels given in the specifications of MCUs are fix and do not consider the impact of temperature. Yet, the MCUs are specified to cover a wide temperature range. For instance the temperature range of the ATmega1284p MCU used in this thesis is specified between $-55^\circ\text{C} \leq T \leq 125^\circ\text{C}$. Hence, covering the worst case scenario of environmental conditions with a fixed voltage level implies that MCUs have to have the ability to run below recommended voltage levels under 'normal' conditions.

3.2 Lab-based Studies

The largest issue when starting to analyze the behavior of undervolted MCUs is that the manufacturers do not release any information about the internal architecture such as RTL-logic. This information would help to reveal weak spots of the MCU when undervolting is applied. Moreover, to allow a continuous observation of the undervolted MCU it would be possible to derive an efficient online-testing by deducing test-vectors.

However, as no information are available, an exemplary MCU was examined by using a black-box approach. In practice the undervolted MCU and therefore the Device Under Test (DUT) is an Atmel ATmega1284p 8-Bit MCU [31]. This MCU contains the widespread AVR-core that is commonly used on many WSN nodes [16, 51] as well as on the popular Arduino platform [52]. Nevertheless, the intention of the corresponding publications [9] is to discuss the general feasibility of undervolting in WSNs and to show characteristics of CMOS based MCU when using this power management technique.

3.2.1 Prototype Implementation and Bench Test

The examination of the feasibility and general effects of undervolting on system level starts with a bench test. As this thesis is focused on a system level approach only Commercial Off-The-Shelf (COTS) parts were used for the practical implementation. In general the prototype implementation is composed of the following three parts (cf. Figure 3.1):

1. *Voltage Scaling Module*: A basic requirement is the ability to scale the voltage level by software. For this purpose there are many existing power management ICs but these fully-integrated solutions are less suitable to be used on WSN nodes. In most cases the overhead, in particular the quiescent current, is too high. Hence, the expected gain due to undervolting might be negated by an oversized power unit. Additionally, the resolution of the voltage scaling is insufficient. With the intention to operate the MCU at the lowest possible voltage level, a fine-grained voltage scaling is inevitable. Hence, a customized circuit based on a digital-potentiometer has been developed. The ohmic resistance of the potentiometer is controllable via software using the I²C-bus. Common adjustable voltage regulators can be connected to the resistor to get a variable voltage level as a function of the register settings of the potentiometer [45, 14].
2. *Undervolted MCU (DUT)*: The exemplary MCU used in this thesis and the corresponding papers is the 8-Bit ATmega1284p. The first prototype implementation was designed to perform extensive evaluations to gain a principal understanding of the possibilities and limitations when undervolting an MCU. To enable testing of several ATmega1284p, the MCUs were mounted on shields fitting on a socket of the prototype implementation. Thus MCU-instances can be

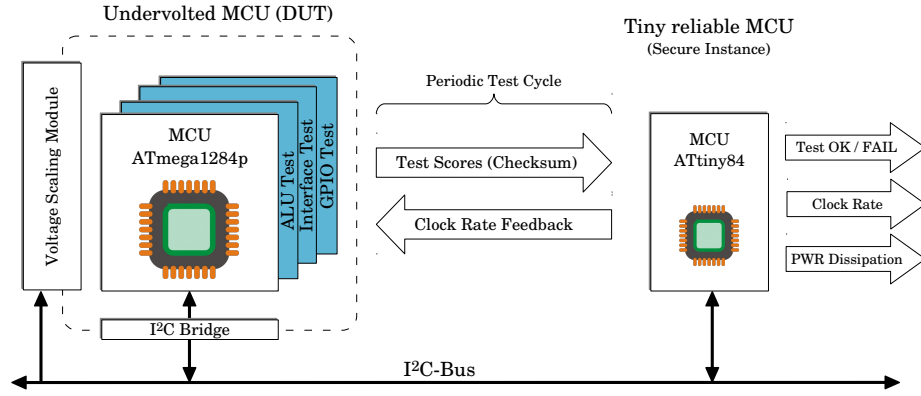


Figure 3.1: Block diagram of the prototype implementation of the bench scale test [9].

examined on the same prototype board by exchanging the shield without effort. The first batch of MCUs were ordered from three different hardware distributors to avoid that all parts originate from the same product line or production run.

3. *Tiny reliable MCU (SI)*: While being undervolted the ATmega1284p is potentially unreliable. To be able to reliably detect malfunctions of the DUT and for test automation a second tiny but reliable MCU is used. A ATtiny84 is responsible for validating test scores, among other things. In the following this co-MCU is called Secure Instance (SI). The SI is an essential part of the prototype and the subsequent IdealVolting implementation (cf. Section 3.3) relies on it.

In sum Figure 3.1 depicts a block diagram and the interaction between all parts. The common interconnection between all parts is given by an I²C-bus.

The ATmega1284p executes several tests periodically. Those tests include the GPIOs, SPI-bus, I²C-bus and mainly the Arithmetic Logical Unit (ALU). The results of the tests are then transferred to the SI which validates the test scores and generates some debug output. Beginning from the nominal voltage level of the MCU, the voltage level can be lowered stepwise. While keep on lowering the voltage level the periodic test cycle reveals failures and thus the individual characteristics of each undervolted ATmega1284p MCUs.

The test cycle is initiated by the ATmega1284p every second so that the default period equals to 1 Hz. In further consequence the SI is able to measure the time between the current I²C frame and the last frame that has been sent. Thus, the impact of undervolting on the system clock can be derived.

To gather information about the power dissipation, the SI is able to measure the current consumption by using a current-sense amplifier. However, in most cases more accurate multimeter and oscilloscopes have been used to evaluate the energy efficiency.

Clock Rate Recalibration Even without undervolting, the internal RC-oscillator of the ATmega-1284p shows a strong dependence on the supplied voltage level [31]. A lower voltage level affects the oscillator speed, so that a lowered voltage level results in a dropped clock rate. While staying within the specifications this connection might be negligible, but due to undervolting this this could become a critical issue.

The following generic result is taken from [9] and clarifies the need of a clock rate recalibration when undervolted MCUs rely on the internal RC-oscillator. Assuming a clock rate of $f_{cpu} = 4$ MHz at the nominal voltage level $V(f_{cpu}) = 1.8$ V. With an undervolting of $V(f_{cpu}) - 10\%$ the oscillation of the internal RC oscillator decreases, so that the clock rate is reduced by about 4.86% to ≈ 3.8 MHz. While a slightly slower processing speed might be acceptable, there are several reasons why it is crucial to adjust the system to a stable clock rate:

- Many time critical processes count on a fixed clock rate for ALU and peripherals such as timers and busses. For example in real-time schedules a missed deadline affects the QoS of applications and has to be avoided [53].
- In Section 2.3 the connection between minimum voltage level and the clock rate was discussed. A dropped clock rate would result in lower minimum voltage levels. For the sake of comparability a stable clock rate should be guaranteed when comparing undervolted MCUs.

To face this problem either a stabilized external clock can be provided or the internal clock rate can be recalibrated. As mentioned above, the SI is able to derive the clock rate of the MCU by measuring the time of the test cycle period. Hence, the SI can generate a binary feedback whether the clock rate should be incremented or decremented. This information is shared by using the I²C-bus so that the ATmega1284p can adjust its RC-oscillator by an internal calibration register (OSCCAL) [31]. The recalibration process was successfully implemented and evaluated in [9] so that it is possible to provide a stable clock rate even with undervolting. Later on the same mechanism was integrated into the IdealVolting process (cf. Section 3.3).

ALU Error Detection The MCU is the central component of a sensor node and controls all other parts of the node (cf. Section 2.1). For this reason every failure of this component could affect the entire node. During all experiments no failures on the busses were detected. Thus, it is very likely that peripherals (e.g. busses) are marginally affected by undervolting. Instead, the ALU is prone to fail at lower than recommended voltages.

The problem is that various failures are possible when undervolting an MCU but that they are not predictable without explicit knowledge about the internal structure of the integrated circuit (e.g., RTL-logic). However, for the further usage of undervolting as a regular power management technique it is essential to determine fundamental characteristics, e.g., the probability distribution of errors.

Performing a full test is not adequate. The bus-width of the MCU is $w = 8$ bit. Hence, testing a simple arithmetic operation with $i = 2$ inputs leads to $(2^w)^i$ test cases. Moreover, the instruction set of the AVR core comprises around 140 commands.

Alternatively an IEC60730 Class B test, which is intended to prevent unsafe operation of micro-controllers [54] could be used. However, the execution time of such a Class B test bears no relation to the expected gain.

Fortunately the AVR core is used for aerospace applications. Thus, the radiation hardness of this type of MCUs has been tested for extraterrestrial applications. In [55] the manifestation of Single Event Upsets (SEUs) while executing different algorithms is examined. It was shown that a simple matrix calculation is sufficient to detect about 86 % of ALU error injections. In assumption that undervolting leads to random failures (comparable to SEUs) inside the AVR core as well, a spot test was derived which is shown in the following Equation 3.2.

$$\text{checksum} = \det(A \cdot B) \text{ with } A, B \in \mathbb{R}^{n \times n} \quad (3.2)$$

In practice the determinant of a matrix consisting of double variables with the dimension of $n = 3$ has been used. Of course, a complete coverage of failures is impossible so that errors like a wrongly calculated program counter or hanging MCUs have to be considered as well.

3.2.2 Characteristics and Results

The first evaluations were performed in our lab with almost static environmental conditions of about 22 °C defined as normal operation condition. A detailed discussion of the results is presented in [9] whereas this section gives a brief summary of the analysis.

Basically the tests confirm that MCUs can operate below their recommended voltage levels. By using the spot-test described in the previous section, the distribution of errors when carrying the

voltage level to the extreme minimum was derived. It turns out that up to a specific operating point no failures occur. The malfunction starts to appear in a small, sharp region and its distribution is similar to the transfer characteristics of CMOS gates [40]. This evaluation was further performed at five different clock rates from $f_{cpu} = 4$ MHz to $f_{cpu} = 8$ MHz. The characteristic appearance pattern of malfunctions is always similar, yet the absolute minimum operation point depends on the individual MCU instance and on the clock rate. While one MCU running at $f_{cpu} = 4$ MHz shows the first malfunction at a percent deviation of $V(F_{cpu}) - 12.22\%$ another MCU operates stable down to a deviation of $V(f_{cpu}) - 17.22\%$. At a clock rate of $f_{cpu} = 8$ MHz a deviation from the nominal voltage level of up to 23.75% was reached.

To show the potential of energy savings the trend of current consumption when lowering the voltage level was measured. As expected, the evaluation reveals a strong connection between the voltage level and the power dissipation. For the ATmega1284p a general relation between the percental saving of the current consumption and the percental deviation from the nominal voltage level of about $\frac{6}{5}$ can be observed whereas this ratio is independent of the clock rate. Considering the energy per clock cycle the examined MCUs saved up 42.66% of energy when using undervolting.

The initial assumption that MCUs have to be tolerant against lower voltage levels was justified by the temperature dependency of the CMOS gates (cf. Section 3.1). Hence, with the aid of a climatic chamber the impact of temperatures on undervolted MCUs was tested. The results validate the trend of the temperature dependence and confirms a linear connection between the absolute minimum operation point and the environmental temperature. In general a higher temperature leads to a lower minimum operation point and vice versa. However, every node shows its individual characteristics in terms of the minimum operation point (intercept) and the temperature dependence (slope).

3.3 IdealVolting – Reliable Undervolting on every Node

The lab-based studies about the feasibility and characteristics of undervolting on WSN nodes provide the basis for an approach to dynamically manage undervolting on a sensor node. Beside the individual characteristics of nodes, the temperature dependence of minimum voltage levels has to be taken into account. IdealVolting, a supervised learning strategy deals with these variations and increases both, the lifetime and the reliability of an undervolted WSN. This section gives only a short overview of IdealVolting, the full details including extensive evaluations have been published in [3] (see page 53).

3.3.1 Undervolting Capable Sensor Node

Based on both, the experiences with the INGA [16] sensor node and the prototype bench test an undervolting capable sensor node has been designed. Hence, the basic design relies on an ATmega1284p MCU as main processing unit to study undervolting. Its supply voltage can be adjusted also including lower than specified levels. Referring to the test bench implementation a second tiny but reliable co-MCU ATtiny84 is included (SI). For wireless communication the ATmega1284p is connected to an AT86RF233 [34] IEEE 802.15.4 RF transceiver.

The sensing unit includes several peripherals and is distinguished from classical nodes as the voltage level from the peripherals is also scalable. Thus, the peripherals can also benefit from a down-scaled voltage level as shown in [14]. Later on, the ability of a scalable voltage path for peripherals is used to apply the concept of passive undervolting (cf. Section 4).

In the course of this thesis a more lightweight variant of the presented INGA v1.5 [3] sensor node was developed. Especially the sensing unit was pared down while the core components stayed the same.

A more detailed description of the node's hardware, schematics and layout is discussed in [3]. The presented node is the first wireless sensor node with undervolting capabilities. However, it should be mentioned that the implementation is rather an experimental platform than a fully optimized sensor

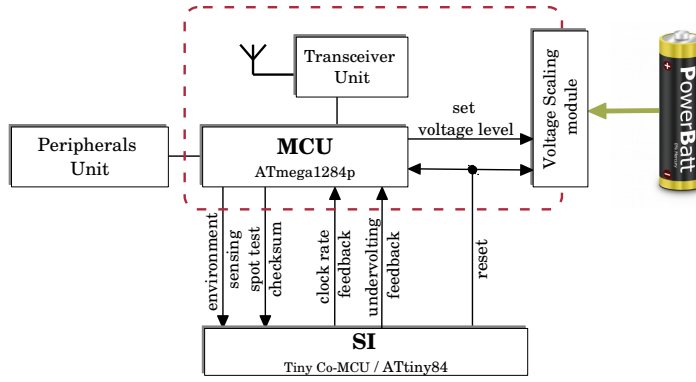


Figure 3.2: Block diagram of the SI based control loop approach [3].

node. The entire hardware relies on COTS parts only so that some overhead, e.g., due to the level translators is inevitable. A discussion about the overhead as well as an evaluation of its impact on the energy savings is also part of [3].

3.3.2 Iterative Self-Optimization

Since the minimum voltage level where the operation of a micro controller becomes unreliable is different for every MCU instance, an adaptive approach which takes care of the individual properties of every single MCU is needed. A straightforward approach would be based on extensive evaluations whose results could be used to define tables with operational voltage level parameters for certain environmental conditions for each individual MCU. However, the required large-scale evaluations would mean an immenseness overhead, when deploying a WSN. Furthermore, it is impossible to test any MCU instance to examine the absolute ratings of undervolting, whereby energy savings are wasted or nodes are failure-prone.

To realize an individual failure handling even for malicious and hanging nodes, the SI is used to build a control loop. As a result each node is able to achieve a balanced state between faultless operation and ideal power dissipation. Figure 3.2 shows a block diagram of this approach.

The I²C-bus is used to interconnect the main MCU, the SI and the voltage scaling module. During normal operation the main MCU is the I²C-bus master and implements a simple request - response protocol. Similar to the periodic test cycle of the bench test implementation, the SI periodically validates the correct operation of the undervolted main MCU. The simple algorithm starts at the recommended voltage level $V(f_{cpu})$. If no error occurred for a certain time, the voltage level is lowered. When the first failure is detected, the voltage level is increased slightly. It does not always happen that ALU errors appear first. In some cases, the MCU can hang and would therefore not be able to get back to a safe state again. However, as the recalibration of the clock rate (cf. Section 3.2.1) is also part of the protocol, the SI measures the time between the test cycles (I²C-frames). Thus, the absence of a request frame of the main MCU is detectable and indicates that the MCU might hang. Subsequently, the SI takes control of the entire sensor node as the I²C-bus interconnects the involved components. A safe state is recovered by resetting the entire node and increasing the voltage level slightly to reach a more stable point of operation again. In sum, due to the fact that the distribution of errors focuses around a sharp region, each node can reach its *ideal voltage* level individually. Moreover, the capability of the SI to act as an external watchdog makes sure, that the system can not hang in an undefined state indefinitely. A detailed description of the iterative self-optimization including the I²C protocol and several experiments validating the functionality of this approach are given in [3].

3.3.3 Supervised Learning Approach

With several evaluations it was shown that the control loop based self-optimizing undervolting can be used reliably throughout a real WSN while prolonging the lifetime of the network significantly. Certainly, the previous approach assumes mostly static environmental conditions and is therefore chiefly usable for, e.g., indoor deployments. However, even without undervolting, a classical WSN already suffers from rough environmental conditions [56, 57] such as experienced in outdoor deployments [58, 59].

With regard to undervolting the temperature is a key value when ascertaining ideal voltage levels (cf. Section 3.1). To prove that self-optimizing undervolting does not only work for almost constant conditions, a node was exposed to different temperature values. The essential statement is that the basic functionality of the automated finding of ideal voltage levels is not affected by varying temperature conditions. This is important as WSNs can be exposed to extreme difference in temperature, e.g. [60, 61]. The small size of sensor nodes and therefore the low mass leads to a fast heating-up due to, e.g., sunlight as well as a fast cooling-down. Using a simple control-loop to deal with such highly varying temperature conditions would provoke numerous failures, as the feedback whether the absolute minimum voltage level is reached is indicated by the detection of a provoked error caused by undervolting (cf. previous Section 3.3.2). Hence, at least one failure occurs whenever a new ideal voltage level is searched due to changes in temperature.

To face this challenge a supervised learning strategy is build on top of the iterative self-optimization. A sensor node can learn from its surrounding world over time by aggregating ideal voltage levels together with environmental information. For this purpose learning needs the ability to remember certain events. Using this memory allows to draw conclusions which help to improve the actions of the specific nodes continuously. In this case these memorizable events are detected failures caused by undervolting respectively the finding of ideal voltage levels. Thus, it is possible to acquire experiences which voltage level is suitable at which temperature and how the clock rate has to be adjusted for those operation points. Using a subset of the gathered data is then used to avoid that a node will run into every possible error. The trend of the data can be used to predict further system states even for unexperienced environmental conditions. Together the combination of self-optimizing undervolting and the supervised learning approach is named IdealVolting. The entire implementation runs on the SI due to the following reasons: First, the undervolted main MCU might be unreliable and therefore less suitable to collect and predict ideal operating points. Second, the software overhead on part of the main MCU should be low to avoid that IdealVolting affects the actual application processes.

Learning Phase: The learning phase starts with the finding of the ideal voltage level. This voltage level is stored with the corresponding temperature in a persistent table on the node. When the temperature changes, the node looks up the minimum voltage in its table. If an entry for the current temperature index exists, the node adopts the stored operating point. Otherwise, iterative self-optimization is used as described above to find the ideal operating point and generate a new table entry for this temperature. Thus, a node will not run into the same unsafe state twice.

In practice the storage space of the SI might be limited so that a quantization of the temperature values needs to be used to reduce the size of the table. A single entry can cover higher temperatures but when many temperatures are covered by a single ideal voltage, the characteristic curve is getting coarser and energy savings are pruned. For the used platform a resolution of 1 °C and a quantization factor of $q = 2$ strikes a balance between the memory footprint and a well-suited characteristic curve.

Prediction: Over time a node collects more and more of verified (temperature,voltage)-tuples and as a result gets more reliable. Nevertheless, every unknown environmental condition provokes at least one failure. A prediction of table entries is of enormous benefit in terms of the dependability of a node.

To shorten the learning phase a subset of ascertained ideal operating points is used to predict missing entries of the table. With regard to equation 3.1 and validated by several experiments, the connection between absolute minimum voltage levels and the temperature is linear. Hence, the linear regression can be used to estimate ideal voltage levels as a function of the temperature. For each entry it has to be guaranteed that the disturbance due to integer rounding and the linear regression does not lead to failure-prone system states.

Firstly a definition of a clear constraint when predicting the data from a fragmentary table was derived. Based on a dataset of several measurements within a climatic chamber the minimum amount of ideal states that fulfil the given constraint was calculated. The results show that the accuracy of the predictions grows with the number of ascertained ideal voltage levels that are taken into account. In practice any five entries of a table are sufficient to predict a reliable but efficient voltage curve for a node. The actual implementation acts with reserve and used a threshold of seven entries.

However, a decent prediction is only possible when there are no outliers in the amount of learned ideal voltage levels. Due to undervolting rogue results could be forwarded to the SI so that a prediction might fail. By taking the standard deviation into account, the 2σ -Method is used to check the table before prediction. Affected entries are deleted and the prediction is merely deferred, until enough valid entries are collected.

A comprehensive description of the entire implementation including the sensor node, all protocols and the self-learning approach is published in [3].

3.3.4 Results

Each step of the development of IdealVolting was validated by several micro-experiments to explore the design points. Thus, it was shown that undervolting can be used on an individual node while ensuring the intended reliability. The functionality of the protocols as well as their characteristics are therefore investigated thoroughly. Conclusive trials were performed to evaluate the presented approaches and compare their efficiency to state of the art power management techniques.

The evaluation of the learning phase is based on a real world WSN deployment on an attic. While the prediction process was disabled, the goal was to show the effectiveness of the learning phase and that ascertained states are safe, i.e., that they are a reliable basis to predict further states. In total four nodes were deployed. Three of these nodes were undervolted and the fourth node operated at regular voltage level for reference purposes. As expected, a lot of errors occurred at the beginning of the 10 days experiment. Every unknown temperature leads to an error because of the feedback for the underlying iterative self-optimization. As any experienced temperature usually adds a new table entry with a corresponding safe operating voltage, already after the first day the amount of errors drops. Afterwards only occasional errors are observed whenever a so-far unknown environmental temperature value was reached. Thus, during longer lifetimes of the network, the total amount of errors will decrease which validates the effectiveness of the learning phase. During this experiment the clock rate of the MCUs was set down to $f_{cpu} = 4$ MHz (default $f_{cpu} = 8$ MHz) and the nodes experienced a temperature range between $12^{\circ}\text{C} \leq T \leq 37^{\circ}\text{C}$. Depending on the temperature and the individual characteristics of the nodes the undervolted MCUs undercut the recommended voltage level significantly by 11.0% to 18.4%.

The test of the prediction process was performed using four nodes running the IdealVolting implementation. All nodes were deployed within a climatic chamber so that a widespread temperature range is covered within a short time. The nodes send their data to a sink located outside of the climatic chamber via IEEE 802.15.4 link. With the aid of the controllable environment of the climatic chamber the experiment was performed repeatedly. Starting from the nominal voltage level the nodes started lowering their voltage level linearly until the first ideal operating point was found. At the beginning the typical characteristic of learning phase is observed where every new temperature value evokes errors due to the searching for ideal voltage levels. However, the threshold for starting the

prediction was set to 7 collected ideal voltage levels so that the absolute amount of errors is limited clearly. After prediction the nodes were exposed to so-far unexperienced temperatures but no more errors were detected. To show resistance against temperature fluctuations, a node was heated up by using a light bulb and subsequently cooled down in a freezer. While this procedure was repeated several times, all detected errors were limited to the learning phase whose operability was not affected by the heavily varying temperature. In sum the evaluations showed that the IdealVolting process pares the total amount of errors down to a minimum and enables a reliable usage of undervolting on sensor nodes and WSNs respectively.

Beside the reliability IdealVolting keeps the nodes always at a highly energy efficient state. Unfortunately, the actual implementation comes with some overhead such as level translators or the SI. This overhead has to be outweighed by the expected gain and the question is, whether IdealVolting is still beneficial and how it compares to other power-management strategies. Hence, with regard to other state-of-the-art power-management techniques IdealVolting was compared to DPM, DVS, modDVS and pure undervolting. modDVS is an improved variant of DVS for low-power MCUs and was presented in [14]. For a varying duty cycle D the power dissipation of different MCUs was measured for each of the aforementioned power-management techniques. At $D = 100\%$ IdealVolting outperforms the current power management techniques with savings in current consumption of about 32.6%. From the classical power management techniques modDVS performs best. However, for $100\% \geq D > 10\%$ the savings in current consumption of IdealVolting is still always $> 15\%$ ($> 20\%$ for pure undervolting without any reliability scheme). Due to the overhead the gain drops at lower duty cycles and for $D < 4\%$ IdealVolting is less beneficial compared to the other mechanisms.

For a concluding and realistic long-term evaluation of IdealVolting a WSN consisting of three undervolted and one normally powered node was deployed outdoors. All nodes were powered by two AAA batteries and sent their data to a sink node via an IEEE 802.15.4 radio link. The lifespan of a node was declared to last from the first to the last received packet. As a result IdealVolting leads to an increased lifetime of the nodes up to 40.8% without affecting the quality of sensing. In parallel a single node running IdealVolting was deployed outdoors for a timespan of about 4.5 months. The learning phase was finished after only 8 hours and in the remaining runtime no more errors were detected.

4 Passive Undervolting of Peripherals

Most research about energy efficiency in WSNs focuses exclusively on the processing or transceiver unit. A rather less considered part of the sensor node is the sensing unit. However, for several applications the power dissipation of this unit might have a significant impact on the entire energy footprint of a node. Multi-sensing applications such as [62, 63, 64] draw on a comprehensive sensor set to fulfil their task. Some of these peripherals show a relatively high current consumption during their active usage. While also for peripheral devices, a lower voltage level leads to a lower energy consumption [14], usually each peripheral requires a different minimum operating voltage. This section deals with an approach applying DVS to peripherals by calculating an optimal voltage schedule for a given usage sequence of the peripheral devices. On the one hand the presented approach will always guarantee that a peripheral is powered with its nominal voltage level during active usage, to ensure correct measurement results. On the other hand peripherals resting in passive state might be undervolted to allow more efficient schedules.

The content of this section has been published in [12] (see page 91) and a demo implementation showed the usability and benefits of this approach [8] (see page 97).

4.1 Peripheral Voltage Scheduling Problem

By way of example a sensor node with three sensors (Sensor 1, Sensor 2 and Sensor 3) is considered. Each sensor requires a different minimum voltage level (v_{s1} , v_{s2} and v_{s3}). In general sensor nodes are not equipped with an adjustable voltage path supplying the sensing unit. Instead, a common voltage level (*ConstDefault*) throughout the node is used for all components (cf. Figure 4.1).

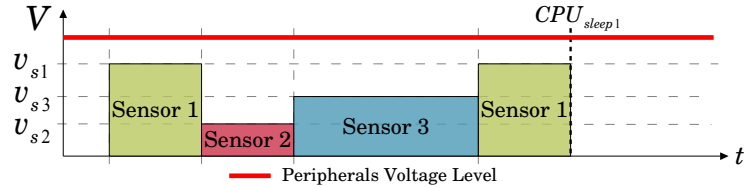


Figure 4.1: Query sequence of three sensors when a default voltage level is applied (*ConstDefault*).

Considering a voltage scaling for peripherals as integrated on the sensor node used in this thesis (cf. Section 3.3.1), the energy efficiency of peripherals can be increased by adapting the voltage level dynamically. Assuming a query-sequence of the peripheral devices on one sensor node. A naive approach is to statically set the voltage level to the maximum of the minimal required voltage level of all involved peripherals (*ConstMaxMin*). Figure 4.2 illustrates this approach.

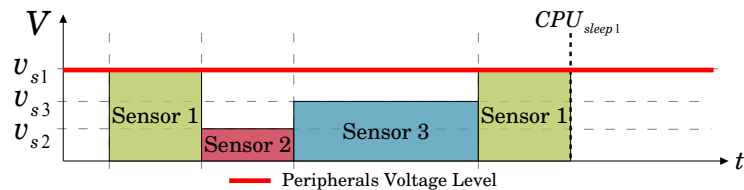


Figure 4.2: *ConstMaxMin* – Use the maximum of the minimal required voltage level.

As a result a subset of the peripherals is still supplied with a voltage level that is above the acceptable minimum, and thus energy is wasted. Hence, another trivial strategy – *AlwaysSwitch* – always adapts the voltage level to the required minimum for every query with a different voltage requirement than its predecessor (cf. Figure 4.3).

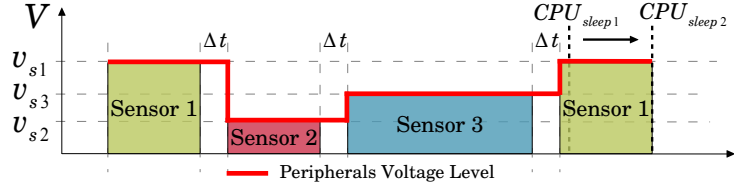


Figure 4.3: Always adapt the voltage level to the required minimum (*AlwaysSwitch*).

Unfortunately *AlwaysSwitch* ignores the fact that switching the voltage evokes a non-negligible overhead. In particular the static overhead and the additional time Δt interfacing an adjustable voltage supply prolongs the duty-cycle of a processing unit, leading to a higher energy consumption. Thus, an adequate algorithm weighing off the benefits of a downscaled voltage level against the switching overhead is needed. The complexity of such an algorithm must fit to the computational and memory constraints of a WSN node. Consider a sensor node with a set S of peripheral hardware devices. Each $s \in S$ has two attributes:

1. A minimum voltage $v_{min}(s)$ required to properly operate s .
2. The total energy consumption $e_s(v)$ of all peripherals while only s is active, as a function of the peripheral voltage v .

The energy consumption $e_s(v)$ of each $s \in S$ is given by Equation 4.1 with the time t_s necessary to query s , the peripheral voltage v , and the accumulated current $I_s(v, t)$ flowing through s as well as through the inactive peripheral hardware $S \setminus \{s\}$:

$$e_s(v) = v \int_0^{t_s} I_s(v, t) dt \quad (4.1)$$

For an amount C of energy, the switching overhead, the sensor node can adapt its peripheral voltage. The application is presented by a sequence of queries denoted by $[1, \dots, n]$, so that the energy consumption E of a voltage schedule is given by:

$$E = \sum_{i=1}^n e_{s_i}(v(i)) + \sum_{i=2}^n \begin{cases} C & \text{if } v(i-1) \neq v(i), \\ 0 & \text{otherwise.} \end{cases} \quad (4.2)$$

The goal is to minimize E . A feasible voltage schedule $v(1), \dots, v(n)$ is *optimal* when E is minimal in the set of all voltage schedules fulfilling the application's sensing requirements. A comprehensive description of the algorithm, solving the voltage scheduling problem, is given in [12]. Dynamic programming is used to efficiently solve the recursion by determining E_i before E_{i+1} . At this point Figure 4.4 visualizes the problem-solving approach of the algorithm. By way of illustration the switching overhead is defined as $C = 1$, the peripherals voltage level is $v_i = i$ and the energy consumption $e_{s_i}(v) = v_i$. It can be seen that switching the voltage level leads to a lower total energy. In this case there are two voltage schedules summing up to an optimal energy consumption. However, the actual schedule is selected by the minimum amount of switching operations.

While every queried peripheral is always supplied by its recommended voltage level, during passive operation peripherals might be operated below their minimum voltage levels. This process is called *passive undervolting*. Analog components are less affected but digital components often hold a state, that is stored and maintained even when the peripheral is idle. Usually integrated volatile memories

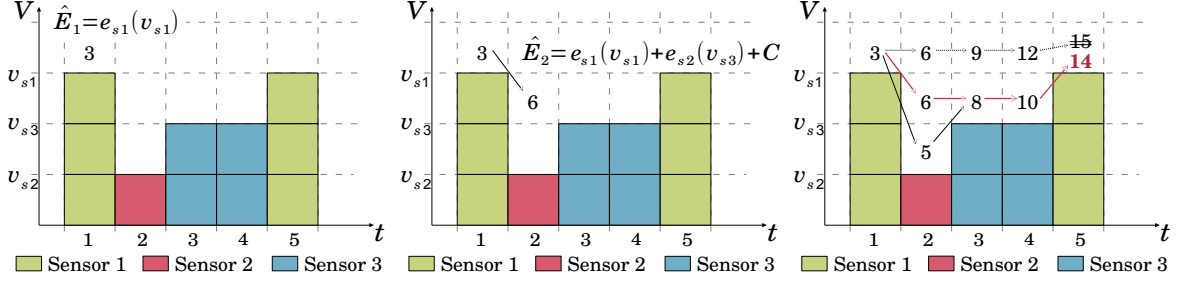


Figure 4.4: Exemplary solution of the voltage scaling problem by using dynamic programming.

such as SRAM are used for this purpose. In fact, Static Random Access Memory (SRAM) is more tolerant against lower than recommended voltage levels than other parts of a peripheral or MCU. Thus, the general feasibility of *passive undervolting* was already shown and utilized in [65]. Hence, the presented mechanism to derive optimal voltage schedules is far more efficient but also reliable.

Nevertheless, some components might provide wrong results when passive undervolting is used. To avoid such risks the algorithm can be easily extended to only use a predefined minimum voltage level. In doing so crucial components can be protected against failures and the process of calculating the schedule is even speed up. The switching costs would increase but might still provide benefits.

Considering that the sensor set S of a WSN node is limited and that an optimal schedule only uses $v(i) \in \{v_{min} | s \in S\}$: The simplicity of the presented approach is sufficiently lightweight to be executed on a typical WSN node with limited computational resources (cf. Section 2.1).

4.2 Implementation and Results

The general idea of providing an adjustable voltage path even for peripheral components was already considered by the undervolting capable WSN node architecture in Section 3.3.1. However, based on this architecture a stand-alone prototype was developed as it offers a more convenient debugging and processing of lab tests. The exemplary sensor set S includes three different motion sensors, a pressure sensor and a non-volatile EEPROM. Figure 4.5 shows a picture of the prototype implementation during operation. A more detailed description of the hardware is given in [12].

As mentioned above, a requirement for calculating an optimal schedule is the knowledge about the energy consumption of the peripherals, in particular $e_s(v)$. In practice determining $e_s(v)$ may be arbitrarily complicated as data sheets rarely offer information and the current consumption of inactive parts has to be considered as well. For this reason the node performs a self-parametrization by using a second MCU or in case of the node presented in Section 3.3.1 the SI. Both, the active time of a peripheral t and the current consumption $I_s(v, t)$ is measured for any given s and values of v . As an optimal schedule would only use the minimum voltage levels of the peripherals $v_{min}(s)$, it is sufficient to measure $e_s(v_{min}(s'))$ for each $(s, s') \in S \times S$. The self-parametrization has to be performed only once and the results can be stored and processed as raw values which reduces the computational complexity as expensive floating-point arithmetic is avoided.

To test the voltage scheduling approach the energy consumption of the calculated switching schedule for exemplary query sequences was compared to the aforementioned trivial strategies *ConstDefault*, *ConstMaxMin* and *AlwaysSwitch*. As expected and shown in [12], in all cases the scheduling approach never consumes more energy than any other strategy. Indeed, for the given list of sequences the improvement of voltage scheduling over *ConstDefault* is between 31.5% and 47.9%. *ConstMaxMin* is more efficient than using a default voltage level. However, scheduling can save up to 20.4% and the majority of the results is above 17.0%. By neglecting the switching overhead *AlwaysSwitch* shows the best results compared to the scheduling approach. Nevertheless, in some cases

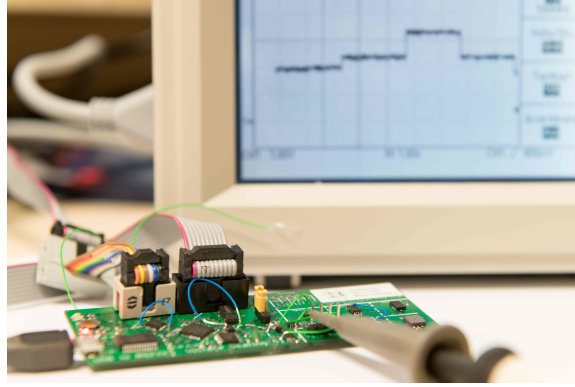


Figure 4.5: Test setup of the prototype implementation during evaluations [12, 8].

the scheduled voltage level leads to savings up to 20.2%.

During all experiments and depending on the respective query sequence, peripherals were passively undervolted. On no account a negative impact on the quality of sensing compared to the trivial strategies was detected. Thus, by allowing inactive peripherals to be powered below their recommended voltage levels, the scheduling of peripherals voltage performs best and helps to increase the energy efficiency of the entire node.

An independent review of the presented approach was possible at the demo session of the "12th European Conference on Wireless Sensor Networks" [8]. The audience was invited to assemble a custom query sequence of peripherals. This sequence was fully self-optimized and evaluated on the prototype implementation directly. Afterwards the real measurement results were displayed and confirmed the feasibility and characteristics of a scheduled peripherals voltage level.

5 Use of Active Undervolting in Real WSN Deployments

Already without a potentially daring energy management technique like undervolting, the programming, deployment and maintenance of a real world WSN is a challenge. This is aggravated by the fact that WSNs which are deployed outdoors are often affected by rough environmental conditions [60, 66, 67]. Especially the partly extreme temperatures lead to the effect that the efficiency and reliability of nodes is strongly affected [68, 56]. Using undervolting in such challenging environments might create objections but the presented IdealVolting implementation was designed to deal with harsh temperature conditions (cf. Section 3.3). Hence, by using a real WSN outdoor deployment the usability and dependability of active undervolting was evaluated extensively for an exemplary Smart Farming application. In addition, the perception that high temperatures will only affect WSNs negatively is counteracted by active undervolting and IdealVolting respectively.

For this section the discussion is mainly based on the following publications [7, 2, 1, 5] (see on pages 103, 105, 113 and 99) which contain far more details.

5.1 PotatoNet – Outdoor WSN Testbed

It is common practice to test new protocols or algorithms for WSNs with the aid of testbed implementations. Such real life examinations reveal pits and snares that have not been considered during theoretical considerations or simulations. Consequently, there is a plethora of WSN testbeds as for instance [69, 70, 71, 72]. However, existing testbeds are less suited to evaluate active undervolting properly. First, active undervolting has not been applied to WSNs before so that existing deployments are not equipped with the required hardware. In addition, errors due to undervolting WSN nodes have to be covered by hard- and software to avoid that worst-case failures affect the entire testbed functionalities. Second, due to the connection between absolute minimum voltage levels and temperatures, the most interesting use case for active undervolting are outdoor deployments. Unfortunately, testbeds are mainly located indoors. Even if testbeds are deployed outdoors [73, 60], they are designed for a special purpose rather than offering features of a generic WSN testbed, e.g., remote availability, programmability or maintainability.

Previously the term Smart Farming was motivated (cf. Section 1). Requirements like robustness or energy efficiency against predominantly fault-tolerant application tasks are well suited to evaluate the performance of active undervolting in WSN scenarios. Therefore, a cooperation with a potato research station (VSD¹) in Northern Germany was entered. The VSD provided one of their trial-fields to deploy a WSN testbed – named *PotatoNet*. As this installation was about 100 km away from the university a major design goal was availability and reliability. The worst thing that can happen to a remote testbed is loosing connectivity to the outside world. In such a case, there would be no way to diagnose or rectify problems remotely. Also, it becomes hard to assist local helpers, who are not experts in the operation of the testbed, to solve or identify hardware problems. Therefore, *PotatoNet* is designed to use two uplinks. A cellular connection is too unreliable to act as the sole uplink so that the next low-speed DSL landline was tapped by using Wi-Fi Bridges and a VDSL point-to-point link.

The basic architecture of *PotatoNet* consists of a central box providing management functionalities and distributing power to several field nodes. For cost-efficiency reasons the nodes are designed to

¹<http://www.vsd-dethlingen.de>

be cheap, while complexity and redundancy features are moved to the central component. Ethernet with passive Power over Ethernet (PoE) was used to distribute power and to manage the field nodes. A field node itself is composed of an undervolting capable WSN node (cf. Section 3.3.1) and a small embedded Linux board WRTNode². By supporting OpenWRT a large amount of standard Linux software is available for installation directly from the OpenWRT repositories. This makes programming or processing data collected from the WSN node very pleasant. Indeed, a substantial difference to other testbeds is that PotatoNet is engineered to provide as little abstractions as possible. The same standard tools as used for a local setup on your desk are available, yet remotely and outdoor.

In the first season PotatoNet was deployed outdoors for more than 9 months. During this time several WSN evaluations were performed and the testbed showed no notable malfunction.

5.1.1 Distributed Energy Measurement

A major area of interest for the presented approaches is the ability to measure the energy efficiency of the nodes and WSN respectively. However, in most setups it is hard to get a realistic picture of energy usage. It is possible to hook up a single node to expensive lab equipment such as an oscilloscope to study it in detail. This, however, does not give you an overview how a network as a whole is performing and is impractical to apply in outdoor testbeds.

Hence, PotatoNet was extended by a reliable, scalable and convenient distributed energy measurement of the field nodes which has been presented in [1]. Each field node was equipped with an MCU-based oscilloscope to measure the power dissipation of the WSN nodes over time. A high sample-rate of up to 500 kHz was desired to enable in-field debugging of protocols and algorithms. To ensure reliability an essential requirement for the measurement is correctness within a widespread temperature range. Environmental conditions of the outdoor deployment also affect the oscilloscope. It has to be ruled out that changes in power dissipation due to environmental variances are not traced back on temperature dependencies of the oscilloscope itself. Therefore the selection of COTS components was driven by estimating the total parasitic error in advance. After designing and implementing the hardware, several oscilloscopes have been evaluated in terms of accuracy and temperature invariance. The lab-tests revealed a 0.9% drift for voltage measurements and a 1.6% drift for current measurements over a temperature range of 100 K. Thus, the oscilloscopes offer trustful results even if they are used in harsh environmental conditions, e.g., in WSN outdoor deployments.

Another design goal for the distributed energy measurement was the scalability. Of course, the intention is to equip as many nodes as possible with an oscilloscope to achieve a maximum coverage of the WSN. Towards this goal, the costs for a single oscilloscope have to be fairly small. This tough requirement has been taken into account while designing the hardware, and finally results in costs below 30 EUR each. However, a large number of oscilloscopes sampling at high resolutions lead to an enormous amount of data. To avoid that the testbed's network is being congested, data are stored on SD cards on each oscilloscope. An index is kept of all measurements across the nodes on a central server. A central management server provides the user a distributed energy measurement system (DEMS). Thus, sampled data can be requested directly or a combination of command line options can be used for starting a measurement and reading the data afterwards.

5.2 Long-Term Study and Results

By using PotatoNet a long-term study was performed to show the reliability and usability of active undervolting in WSNs. Therefore, all field nodes changed periodically between IdealVolting and using the nominal voltage level. The switching of the entire WSN state took place every 144 s so that both states experienced almost the same environmental conditions. Thus, the characteristics of a

²<http://wrtnode.com>

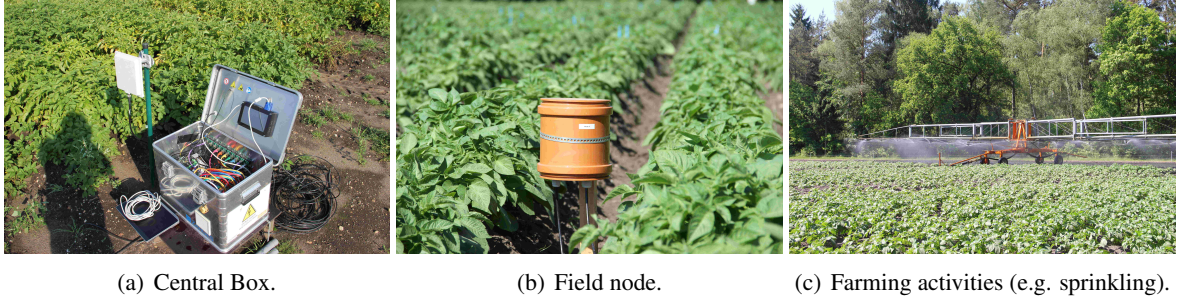


Figure 5.1: PotatoNet and its environment during the long-term study.

network running IdealVolting can be compared directly against those of a WSN with normal powered nodes.

The experiment was performed for a period of one farming season (about 4 months) where a dataset of 7.5 GB was collected. Meanwhile, the WSN experienced rough environmental conditions like direct sun exposure, rain, sprinkling and farming activities. Maximum temperature values of $-1^\circ\text{C} \leq T \leq 49^\circ\text{C}$ and a high fluctuation indicated that the WSN deployed on a field is exposed to a challenging environment. Nevertheless, IdealVolting was executed without any malfunction so that each node settled at its individual minimum voltage level which was continuously adapted depending on the ambient temperature. Thus, individual characteristics of nodes paired with a highly varying environment lead to a strong heterogeneity throughout the entire WSN.

Beside general characteristics of IdealVolting and WSNs in Smart Farming applications, the major interest was to prove the dependability of active undervolting in WSNs. Therefore, the Packet Reception Ratio (PRR) in IdealVolting mode (PRR_{iv}) was compared to the PRR in nominal voltage mode (PRR_n). Firstly the difference between the PRRs was calculated as $\Delta\text{PRR} = \text{PRR}_n - \text{PRR}_{iv}$. In all scenarios ΔPRR seems negligible which is a good indication that IdealVolting does not affect the baseline reliability of a WSN. To prove this assumption, the voltage level during IdealVolting v was correlated with ΔPRR and the correlation coefficient was derived as follows:

$$r(\Delta\text{PRR}, v) = \frac{\sum_i (\Delta\text{PRR}_i - \overline{\Delta\text{PRR}}) \cdot (v_i - \bar{v})}{\sqrt{\sum_i (\Delta\text{PRR}_i - \overline{\Delta\text{PRR}})^2 \cdot \sum_i (v_i - \bar{v})^2}} \quad (5.1)$$

$$\text{with } \overline{\Delta\text{PRR}} = \frac{\sum_i \Delta\text{PRR}_i}{n} \quad \text{and} \quad \bar{v} = \frac{\sum_i v_i}{n}$$

If undervolting impacts the PRR, the coefficient $r(\Delta\text{PRR}, v)$ would reveal a relationship. However, in any case the values for r are very low so that a statistical connection between undervolting and the PRR of the WSN can be ruled out. Furthermore, the same methodology was used to refute an effect of active undervolting on the Received Signal Strength Indicator (RSSI) and the quality of sensing.

All results and a more detailed description of this long-term study have been published in [2] which is annexed to this thesis. Moreover, the entire dataset has been made publicly available for the community.

5.3 Temperature-Dependent Energy Efficiency

While the transceiver unit suffers from hot temperatures, active undervolting leads to a lowered voltage level for processing on the node. For this reason, the question arises how wilfully influencing

the most relevant factor – the temperature – affects the energy savings of a WSN node using active undervolting. Although data sheets document an increasing power dissipation at higher temperatures due to physical CMOS effects, the strong temperature dependency of the absolute minimum voltage level might offer the opportunity for further energy savings.

Several nodes were examined within a climatic chamber to show the power dissipation of an undervolted MCU when using the IdealVoting scheme. The data were used to derive an energy model where the current consumption can be described as a function of the voltage level v and the environmental temperature T termed as $I_{cc}(v, T)$. This model was extended with the individual characteristics of several nodes, in particular the minimum voltage level as a function of the temperature $v(T)$. Together it is possible to estimate the energy consumption of WSN nodes solely by knowing the temperature T .

The model itself revealed that an increased temperature leads to an increased energy efficiency of the MCU and thus of the entire node. Obviously, actively heating a node is not sustainable, however housings of different materials with different heat capacities and thermal conductivities can change the temperature a node experiences. To quantify the potential energy savings from optimal temperature profiles we performed a long term measurement of the temperatures experienced by sensor nodes in 4 different housings (without housing, plastic enclosure, transparent glass and black painted stone with thermal capacity) and two different locations (direct sunlight, shadowy place). The experiment was performed for approximately three months from May to July and located in Northern Germany.

Unsurprisingly, nodes deployed in a shadowy place are less affected than nodes exposed to direct sunlight. However, there is a remarkable difference between the housings, especially during day times. The glass housing located in a sunny spot adds up the highest average temperature and offers therefore the biggest potential for energy savings when using active undervolting. On a single day up to $\approx 4\%$ of energy savings can be achieved between the housings. Between a hot and a cold day even $\approx 20\%$ of savings are possible. The results might not seem outstanding but it has to be considered that these savings are for free by just choosing the right design of a housing.

A more detailed description of the experiment and a deeper discussion of the results is presented in [5] which is appended to this thesis.

6 Conclusions

For many years WSNs played a significant role within academic research on distributed computing. The upcoming IoT helps to provide WSNs to a renaissance but also reveals the open issues and challenges when using WSN for real world applications. Especially outdoor deployments suffer from rough environmental conditions and the usage of location independent energy sources limits the service life of applications clearly. While a huge number of existing power-management techniques always considered the absolute dependability of WSNs, this thesis proposed the usage of *undervolting* to increase the energy efficiency of nodes and networks significantly. In fact an emerging class of WSN applications do not require a complete dependability but would benefit from an increased lifetime. Moreover, the per se fault tolerant character of WSNs is well fitted for the usage of a more daring power-management technique, in particular undervolting.

Initially the general feasibility of undervolting was motivated by theoretical considerations. The temperature dependency of CMOS gates as well as manufacturing variances has led to the assumption that micro-electronic components must be able to operate below their recommended voltage levels under certain conditions. This assumption is not only limited to the scope of this thesis, as the underlying CMOS technology is primarily used for realizing diverse MCUs. Usually manufacturers do not provide any information about the internal logic of their components. Hence, the practical examination of an undervolted MCU was performed by black-box testing. A bench scale test was implemented to evaluate the characteristics of undervolting where several Atmel ATmega1284p MCU were used. The ATmega1284p was selected representative for typical MCUs used on WSN nodes. The lab-based studies showed both the huge potential of energy savings of up to 42% and the general feasibility of undervolting on WSN nodes.

The main contribution is the reliable and effective utilization *active undervolting* in real WSNs. A sole reliance on the fault tolerance of the WSN is not a solution when applying undervolting to sensor nodes. The error-proneness would lead to potentially unstable nodes so that an adequate execution of the respective WSN application is jeopardized. For this reason the goal of the presented *IdealVolting* mechanism is to achieve the most energy efficient but dependable state for nodes individually. By using a second tiny but reliable MCU a control-loop was implemented which adapts each node to the aforementioned ideal operating state. This self-optimization was extended by a supervised-learning strategy to cover the impact of changing environmental conditions, especially temperatures. The entire development of IdealVolting was continuously evaluated by several micro-experiments to corroborate each part of the implementation. Even for challenging environments further tests showed the performance and reliability of the IdealVolting scheme and demonstrated the benefits for WSNs.

Beside the MCU also peripherals such as sensors or memories benefit from a downscaled voltage level in terms of a more energy efficient operation. Thus, within this thesis an efficient voltage scaling approach for peripherals was proposed, which assumes that inactive peripherals are allowed to be undervolted (*passive undervolting*). The crux was that on the one hand different peripherals usually require different minimum voltage levels to operate properly and on the other hand the switching of the voltage level evokes a non-negligible overhead. Therefore, the presented scheduling algorithm weighs off the benefits due to lower voltage levels against the aforementioned switching overhead. It is always ensured that queried peripherals are supplied by their specified voltage level, however, inactive components might be undervolted. While it was shown that the energy efficiency of query sequences can be increased by up to 47.9%, an impact on the quality of sensing due to passive undervolting was not observed for the used ensemble of peripherals.

To examine the effects of *active undervolting* of the node's MCU and in particular IdealVolting in

greater depth, the *PotatoNet* WSN testbed was developed. In the light of upcoming Smart Farming applications, a WSN composed of undervolting capable sensor nodes was deployed on a trial field of a potato research station. This exemplary application area was ideal to show the resilience of IdealVolting in real world applications. However, implementing, maintaining and using a testbed in a rural area with challenging environmental conditions is not trivial. Robustness and availability were two major design goals to guarantee permanent testbed capabilities. For maximum flexibility PotatoNet relies on the same tools for collecting data or programming nodes as would be used for standard desk-setups, yet the actual WSN is located faraway outdoors.

By using the PotatoNet a *long-term study* was performed to show that IdealVolting leads to a reliable usage of undervolting in WSNs. For a duration of one farming season the nodes periodically switched between IdealVolting and using the nominal voltage level. Thus, a direct comparison of an undervolted WSN against a normal powered WSN was enabled. To proof that IdealVolting does not decrease the baseline reliability of WSNs, common metrics such as PRR, RSSI or the quality of sensing were considered. The results show that in no case a correlation between undervolting and the WSN metrics were attested.

Finally, although many doubts emerged when the idea came up to use undervolting in WSNs, the thesis at hand showed that it is yet feasible and possible to achieve both – a significant *increase of energy efficiency while ensuring reliability*.

6.1 Future Work

On node level a big issue is the additional overhead of components that are needed to implement IdealVolting. Especially the usage of the SI is responsible for the pruned energy efficiency of IdealVolting at low duty-cycles. Thus, in future work the break up of the static control-loop should be intended. The SI could collect information about the reliability of the operating states. Thus, the periodic check becomes dynamical and would lower the usage of the SI. To cover long-term reliability, statistical approaches like the bathtub curve [74] could extend this process.

It is also possible to combine IdealVolting with energy harvesting techniques. Motivated by recent publications [75] the operability of the node's components can be scheduled depending on the available energy budget. Processes can be prioritised with regard to their requirement how safe they have to be executed or their deadline. Thus, IdealVolting can help to optimize the utilization of the energy storage as well as the QoS.

On network level active undervolting adds more heterogeneity. The individual characteristics of nodes and the temperature dependency of IdealVolting lead to WSNs where some nodes are more efficient than others. Classical approaches of collaborative computing [76] or routing protocols [77] could be extended to exploit this effect.

The distributed character of WSNs can further be used to utilize fault-tolerant mechanisms. For example in [78] it is planned to transfer the concept of approximate computing to WSNs. Thus, more aggressive undervolting could be considered while WSN applications can be executed on top of a highly efficient but potential unreliable network.

Bibliography

- [1] Robert Hartung, Ulf Kulau, and Lars Wolf. Distributed Energy Measurement in WSNs for Outdoor Applications. In *IEEE International Conference on Sensing, Communication and Networking Conference*, SECON 2016, London, UK, June 2016. Accepted.
- [2] Ulf Kulau, Stephan Rottmann, Sebastian Schildt, Johannes van Balen, and Lars Wolf. Undervolting in Real World WSN Applications: A Long-Term Study. In *the 12th IEEE International Conference on Distributed Computing in Sensor Systems*, DCoSS 2016, Washington D.C., USA, May 2016.
- [3] Ulf Kulau, Felix Büsching, and Lars Wolf. IdealVolting – Reliable Undervolting on Wireless Sensor Nodes. *ACM Transactions on Sensor Networks (TOSN)*, 12(2), April 2016.
- [4] Ulf Kulau, Tobias Breuer, and Lars Wolf. A Clustering-Based Characteristic Model for Unreliable Sensor Network Data. In *IEEE World Forum on Internet of Things*, WF-IoT 2015, Milan, Italy, December 2015.
- [5] Ulf Kulau, Sebastian Schildt, Stephan Rottmann, and Lars Wolf. Paint it Black – Increase WSN Energy Efficiency with the Right Housing. In *the 6th ACM Workshop on Real World Wireless Sensor Networks*, RealWSN 2015, Seoul, South Korea, November 2015.
- [6] Ulf Kulau, Sebastian Schildt, Stephan Rottmann, Björn Gernert, and Lars Wolf. PotatoNet – Outdoor WSN Testbed for Smart Farming Applications. Technical Report CS-2015-06, Erlangen, Germany, September 2015.
- [7] Ulf Kulau, Sebastian Schildt, Stephan Rottmann, Björn Gernert, and Lars Wolf. Demo: PotatoNet – Robust Outdoor Testbed for WSNs: Experiment Like on Your Desk. Outside. In *the 10th ACM MobiCom Workshop on Challenged Networks*, CHANTS 2015, Paris, France, September 2015.
- [8] Ulf Kulau, Stephan Friedrichs, and Lars Wolf. Demo: Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes. In *the 12th European Conference on Wireless Sensor Networks*, EWSN 2015, Porto, Portugal, February 2015.
- [9] Ulf Kulau, Felix Büsching, and Lars Wolf. Undervolting in WSNs: Theory and Practice. *Internet of Things Journal, IEEE*, 2(3):190–198, June 2015.
- [10] Felix Büsching, Keno Garlich, Johannes van Balen, Yannic Schröder, Kai Homeier, Wolf-Bastian Pöttner Ulf Kulau, Stephan Rottmann, Sebastian Schildt, Georg von Zengen, and Lars Wolf. Quadrotor-based DT-WSNs for Disaster Recovery. Technical Report TR-2014-1, Potsdam, Germany, September 2014.
- [11] Ulf Kulau, Sebastian Schildt, Georg von Zengen, Stephan Rottmann, and Lars Wolf. Ballistic Deployment of WSN Nodes using Model Rockets. In *the 6th Extreme Conference of Communication*, ExtremeCom 2014, Galapagos Islands, Ecuador, August 2014.
- [12] Stephan Friedrichs, Ulf Kulau, and Lars Wolf. Energy-Efficient Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes. In *IEEE International Conference on Communications Workshop on Energy Efficiency in Wireless Networks & Wireless Networks for Energy Efficiency*, E2Nets 2014, Sydney, Australia, June 2014.

- [13] Ulf Kulau, Felix Büsching, and Lars Wolf. Undervolting in WSNs – A Feasibility Analysis. In *IEEE World Forum on Internet of Things 2014*, WF-IoT 2014, Seoul, South Korea, March 2014.
- [14] Ulf Kulau, Felix Büsching, and Lars Wolf. A Node’s Life: Increasing WSN Lifetime by Dynamic Voltage Scaling. In *the 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013*, DCoSS 2013, Cambridge, USA, May 2013.
- [15] Ulf Kulau, Felix Büsching, Wolf-Bastian Pöttner, and Lars Wolf. Demo: A Node’s Life – Increasing WSN Lifetime by Dynamic Voltage Scaling. In *the 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013*, DCoSS 2013, Cambridge, USA, May 2013.
- [16] Felix Büsching, Ulf Kulau, and Lars Wolf. Architecture and Evaluation of INGA - An Inexpensive Node for General Applications. In *IEEE Sensors*, Sensors 2012, Taipei, Taiwan, October 2012.
- [17] Felix Büsching, Ulf Kulau, Matthias Gietzelt, and Lars Wolf. Comparison and validation of capacitive accelerometers for health care applications. *Computer Methods and Programs in Biomedicine*, 106(2), May 2012.
- [18] Felix Büsching, Ulf Kulau, and Lars Wolf. Demo: INGA - An Inexpensive Node for General Applications. In *the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys 2011, Seattle, WA, USA, November 2011.
- [19] Ulf Kulau, Felix Büsching, Wolf-Bastian Pöttner, and Lars Wolf. INGA - Architektur eines universell einsetzbaren Sensorknotens. Technical Report TR-RI-11-313, Paderborn, Germany, September 2011.
- [20] Shanzhi Chen, Hui Xu, Dake Liu, Bo Hu, and Hucheng Wang. A vision of IoT: Applications, challenges, and opportunities with china perspective. *Internet of Things Journal, IEEE*, 1(4):349–359, 2014.
- [21] Dave Evans. The Internet of Things: How the Next Evolution of the Internet is Changing Everything. *White Paper, Cisco Internet Business Solutions Group (IBSG)*, 2011. Retrieved from: http://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf.
- [22] Intel Cooperation. Intel IoT Gateway. *Product Brief*, 2014. Retrieved from: <http://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/gateway-solutions-iot-brief.pdf>.
- [23] Sujesha Sudevalayam and Purushottam Kulkarni. Energy harvesting sensor nodes: Survey and implications. *Communications Surveys & Tutorials, IEEE*, 13(3):443–461, 2011.
- [24] Fred Schlachter. No Moore’s Law for batteries. *Proceedings of the National Academy of Sciences*, 110(14):5273–5273, 2013.
- [25] Holger Karl and Andreas Willig. *Protocols and architectures for wireless sensor networks*. John Wiley & Sons, 2007.
- [26] Muhammad Adeel Mahmood, Winston KG Seah, and Ian Welch. Reliability in wireless sensor networks: A survey and challenges ahead. *Computer Networks*, 79:166–187, 2015.
- [27] Tifenn Rault, Abdelmadjid Bouabdallah, and Yacine Challal. Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104–122, 2014.

- [28] Cédric Adjih, Emmanuel Baccelli, Eric Fleury, Gaetan Harter, Nathalie Mitton, Thomas Noel, Roger Pissard-Gibollet, Frédéric Saint-Marcel, Guillaume Schreiner, Julien Vandaele, and Thomas Watteyne. FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed. In *IEEE World Forum on Internet of Things, WF-IoT 2015*, 2015.
- [29] Abhishek Arora, Rajiv Ramnath, Emre Ertin, Pradeep Sinha, Sandip Bapat, Vinayak Naik, Vinod Kulathumani, Hongwei Zhang, Hui Cao, Mukundan Sridharan, et al. Exscal: Elements of an extreme scale wireless sensor network. In *the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 102–108. IEEE, 2005.
- [30] Nathan Ickes, Yildiz Sinangil, Francesco Pappalardo, Elio Guidetti, and Anantha P Chandrakasan. A 10 pJ/cycle ultra-low-voltage 32-bit microprocessor system-on-chip. In *the 37th IEEE European Solid-State Circuits Conference, ESSCIRC 2011*, 2011.
- [31] Atmel. Datasheet: ATmega 1284/1284P, Rev. Atmel-8272G-AVR-01/2015, 2015. Retrieved from: <http://www.atmel.com/devices/atmega1284p.aspx>.
- [32] IEEE Standard for Local and metropolitan area networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006)*, pages 1–314, September 2011.
- [33] Atmel. Datasheet: AT86RF231, Low Power 2.4 GHz Transceiver for ZigBee, IEEE 802.15.4, 6LoWPAN, RF4CE, SP100, WirelessHART and ISM Applications, Rev.8111C - 09/09, 2009. Retrieved from: http://www.atmel.com/dyn/resources/prod_documents/doc8111.pdf.
- [34] Atmel. Datasheet: AT86RF233, Low Power, 2.4GHz Transceiver for ZigBee, RF4CE, IEEE 802.15.4, 6LoWPAN, and ISM Applications, Rev. 8315E - 07/14, 2014. Retrieved from: <http://www.atmel.com/devices/AT86RF233.aspx>.
- [35] Hai Chen, Bingqing Wei, and Dongsheng Ma. Energy storage and management system with carbon nanotube supercapacitor and multidirectional power delivery capability for autonomous wireless sensor nodes. *IEEE Transactions on Power Electronics*, 25(12):2897–2909, 2010.
- [36] T. Sels, C. Dragu, T. Van Craenenbroeck, and R. Belmans. Overview of new energy storage systems for an improved power quality and load managing on distribution level. In *the 16th International Conference and Exhibition on Electricity Distribution*, volume 4 of *CIGRE 2001*, pages 5 pp. vol.4–, 2001. Part 1: Contributions (IEE Conf. Publ No. 482).
- [37] Cooper Bussmann. High Power Supercapacitors Enable New Power Applications. . .and Replace Batteries, 2008. Retrieved from: http://www.cooperindustries.com/content/dam/public/bussmann/Electronics/Resources/Brochures/Supercapacitors_for_Replacing_Batteries.pdf.
- [38] Stephen Bo Furber. *ARM system-on-chip architecture*. Pearson Education, 2000.
- [39] Nam Sung Kim, Todd Austin, David Baauw, Trevor Mudge, Krisztián Flautner, Jie S Hu, Mary Jane Irwin, Mahmut Kandemir, and Vijaykrishnan Narayanan. Leakage current: Moore’s law meets static power. *computer, IEEE*, 36(12):68–75, 2003.
- [40] Ulrich Tietze and Christoph Schenk. *Electronic Circuits: Handbook for Design and Application*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
- [41] Luca Benini, Alessandro Bogliolo, and Giovanni De Micheli. A survey of design techniques for system-level dynamic power management. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8(3):299–316, 2000.

- [42] Hai Li, Swarup Bhunia, Yiran Chen, TN Vijaykumar, and Kaushik Roy. Deterministic clock gating for microprocessor power reduction. In *the 9th IEEE International Symposium on High-Performance Computer Architecture*, HPCA 2003, pages 113–122, 2003.
- [43] Wei Tuning, Yu Sijia, and Wei Hailong. A dynamic voltage scaling algorithm for wireless sensor networks. In *the 3rd IEEE International Conference on Advanced Computer Theory and Engineering*, ICACTE 2010, pages V1–554 –V1–557, August 2010.
- [44] T. Hamachiyo, Y. Yokota, and E. Okubo. A Cooperative Power-Saving Technique Using DVS and DMS Based on Load Prediction in Sensor Networks. In *the 4th IEEE International Conference on Sensor Technologies and Applications*, SENSORCOMM 2010, pages 7 –12, July 2010.
- [45] Leander B Hörmann, Philipp Glatz, Christian Steger, and Reinhold Weiss. Energy Efficient Supply of WSN Nodes using Component-Aware Dynamic Voltage Scaling. In *the 11th European Wireless Conference 2011-Sustainable Wireless Technologies*, European Wireless, April 2011.
- [46] Aman Kansal, Jason Hsu, Sadaf Zahedi, and Mani B Srivastava. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)*, 6(4):32, 2007.
- [47] John Sartori and Rakesh Kumar. Overscaling-friendly Timing Speculation Architectures. In *the 20th Symposium on Great Lakes Symposium on VLSI*, GLSVLSI 2010, pages 209–214. ACM, 2010.
- [48] M. Floyd, M. Allen-Ware, K. Rajamani, B. Brock, C. Lefurgy, A.J. Drake, L. Pesantez, T. Gloekler, J.A. Tierno, P. Bose, and A. Buyuktosunoglu. Introducing the Adaptive Energy Management Features of the Power7 Chip. *Micro, IEEE*, 31(2):60–75, March 2011.
- [49] Neil Weste and David Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*. Addison-Wesley Publishing Company, 2010.
- [50] R Jacob Baker. *CMOS: Circuit Design, Layout, and Simulation*. John Wiley & Sons, 2008.
- [51] Crossbow. Datasheet: Micaz, rev. a, 2012. Retrieved from: http://www.openautomation.net/uploads/productos/micaz_datasheet.pdf.
- [52] Charalampos Doukas. *Building Internet of Things with the ARDUINO*. CreateSpace Independent Publishing Platform, 2012.
- [53] T. O’Donovan, J. Brown, F. Büsching, A. Cardoso, J. Cecelio, J. do O, P. Furtado, P. Gil, A. Jugel, W-B. Pöttner, U. Roedig, J. sa Silva, R. Silva, C.J. Sreenan, V. Vassiliou, L. Wolf T. Voigt and, and Z. Zinonos. The GINSENG System for Wireless Monitoring and Control: Design and Deployment Experiences. *ACM Transactions on Sensor Networks (TOSN)*, 10(1), November 2013.
- [54] Atmel. AVR998: Guide to IEC60730 Class B compliance with AVR microcontrollers, Rev. 7715B–AVR–04/08, 2008. Retrieved from: <http://www.atmel.com/images/doc7715.pdf>.
- [55] A. Rohani and H.-R. Zarandi. An Analysis of Fault Effects and Propagations in AVR Microcontroller ATmega103(L). In *IEEE International Conference on Availability, Reliability and Security*, ARES 2009, pages 166–172, 2009.
- [56] Carlo Alberto Boano, Hjalmar Wennerström, Marco Antonio Zúñiga, James Brown, Chamath Keppitiyagama, Felix Jonathan Oppermann, Utz Roedig, Lars-Åke Nordén, Thiemo Voigt, and

- Kay Römer. Hot Packets: A Systematic Evaluation of the Effect of Temperature on Low Power Wireless Transceivers. In *the 5th Extreme Conference on Communication*, ExtremeCom, pages 7–12. ACM, August 2013.
- [57] Florian Schmidt, Matteo Ceriotti, Niklas Hauser, and Klaus Wehrle. If You Can't Take The Heat: Temperature Effects On Low-Power Wireless Networks And How To Mitigate Them. In *the 12th European Conference on Wireless Sensor Networks*, EWSN 2015. Springer, February 2015.
 - [58] Carlo Alberto Boano, James Brown, Zhitao He, Utz Roedig, and Thiemo Voigt. Low-Power Radio Communication in Industrial Outdoor Deployments: The Impact of Weather Conditions and ATEX-Compliance. In *the 1st International Conference on Sensor Applications, Experimentation, and Logistics*, SENSAPPEAL 2009, pages 159–176, September 2009.
 - [59] Carlo Alberto Boano, Marco Antonio Zuniga, James Brown, Utz Roedig, Chamath Keppitiyagama, and Kay Roemer. TempLab: a testbed infrastructure to study the impact of temperature on wireless sensor networks. In *the 13th International Conference on Information Processing in Sensor Networks*, IPSN 2014. ACM, 2014.
 - [60] J. Beutel, B. Buchli, F. Ferrari, M. Keller, and M. Zimmerling. X-SENSE: Sensing in extreme environments. In *Design, Automation Test in Europe Conference Exhibition*, DATE 2011, pages 1–6, March 2011.
 - [61] Kenneth Bannister, Gianni Giorgetti, and Sandeep Gupta. Wireless sensor networking for hot applications: Effects of temperature on signal strength, data collection and localization. In *the 5th Workshop on Embedded Networked Sensors*, HotEmNets 2008. Citeseer, 2008.
 - [62] Tao Lin, Lingran Li, and Gérard Lachapelle. Multiple sensors integration for pedestrian indoor navigation. In *IEEE International Conference on Indoor Positioning and Indoor Navigation*, IPIN 2015, pages 1–9, 2015.
 - [63] Ramasamy Kannan and Ankur Garg. Adaptive sensor fusion technology for mobile and wearable applications. In *IEEE Sensors*, pages 1–4, 2015.
 - [64] Marco Mancuso and Franco Bustaffa. A wireless sensors network for monitoring environmental variables in a tomato greenhouse. In *IEEE International Workshop on Factory Communication Systems*, pages 107–110, 2006.
 - [65] Hrishikesh Jayakumar, Arnab Raha, and Vijay Raghunathan. Hypnos: an ultra-low power sleep mode with SRAM data retention for embedded microcontrollers. In *the International Conference on Hardware/Software Codesign and System Synthesis*, page 11. ACM, 2014.
 - [66] Miguel Navarro, Tyler W. Davis, Yao Liang, and Xu Liang. A study of long-term wsn deployment for environmental monitoring. In *the 24th International IEEE Symposium on Personal Indoor and Mobile Radio Communications*, PIMRC 2013, Sept 2013.
 - [67] Carlo Alberto Boano, Nicolas Tsiftes, Thiemo Voigt, James Brown, and Utz Roedig. The impact of temperature on outdoor industrial sensor network applications. *IEEE Transactions on Industrial Informatics*, 6(3):451–459, 2010.
 - [68] Hjalmar Wennerstrom, Frederik Hermans, Olof Rensfelt, Christian Rohner, and Lars-Ake Norden. A long-term study of correlations between meteorological conditions and 802.15.4 link performance. In *the 10th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, SECON 2013, pages 221–229, 2013.

- [69] Ioannis Chatzigiannakis, Stefan Fischer, Christos Koninis, Georgios Mylonas, and Dennis Pfisterer. WISEBED: An Open Large-Scale Wireless Sensor Network Testbed. In Nikos Komninos, editor, *Sensor Applications, Experimentation, and Logistics*, volume 29 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 68–87. Springer Berlin Heidelberg, 2010.
- [70] P. Hurni, M. Anwander, G. Wagenknecht, T. Staub, and T. Braun. TARWIS - A testbed management architecture for wireless sensor network testbeds. In *the 7th IEEE International Conference on Network and Service Management*, CNSM 2011, October 2011.
- [71] Jang-Ping Sheu, Chia-Jen Chang, Chung-Yueh Sun, and Wei-Kai Hu. WSNTB: A testbed for heterogeneous wireless sensor networks. In *the 1st IEEE International Conference on Ubi-Media Computing*, July 2008.
- [72] G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: a wireless sensor network testbed. In *the 4th International Symposium on Information Processing in Sensor Networks*, IPSN 2005. ACM, April 2005.
- [73] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In *the 20th IEEE International Parallel and Distributed Processing Symposium*, IPDPS 2006, page 8, April 2006.
- [74] J. R. English, Li Yan, and T. L. Landers. A modified bathtub curve with latent failures. In *IEEE Reliability and Maintainability Symposium*, pages 217–222, January 1995.
- [75] Josiah Hester, Lanny Sitanayah, and Jacob Sorber. Tragedy of the Coulombs: Federating Energy Storage for Tiny, Intermittently-Powered Sensors. In *the 13th ACM Conference on Embedded Networked Sensor Systems*, SenSys 2010, pages 5–16. ACM, 2015.
- [76] Yuan Tian, Eylem Ekici, and Füsün Özgüner. Energy-constrained task mapping and scheduling in wireless sensor networks. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference*, MASS 2005, pages 8–pp. IEEE, 2005.
- [77] Miguel Navarro and Yao Liang. Efficient and Balanced Routing in Energy-Constrained Wireless Sensor Networks for Data Collection. In *the International Conference on Embedded Wireless Systems and Networks*, EWSN 2016, pages 101–113. Junction Publishing, 2016.
- [78] Thiemo Voigt, Magnus Sjölander, and Frederik Hermans. Poster: Approximation: A New Paradigm Also for Wireless Sensing. In *the International Conference on Embedded Wireless Systems and Networks*, EWSN 2016, pages 221–222. Junction Publishing, 2016.

List of Acronyms

ADC	Analog-Digital Converter
ALU	Arithmetic Logical Unit
CMOS	Complementary Metal-Oxide Semiconductor
COTS	Commercial Off-The-Shelf
CPS	Cyber Physical Systems
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DPM	Dynamic Power Management
DUT	Device Under Test
DVS	Dynamic Voltage Scaling
EEPROM	Electrically Erasable Programmable Read-Only Memory
FET	Field-Effect Transistor
GPIO	General Purpose Input/Output
I²C	Inter-Integrated Circuit
IC	Integrated Circuit
IoT	Internet of Things
ISM	Industrial, Scientific and Medical
LDO	Low Dropout Linear Converter
MAC	Medium Access Control
MCU	Micro Controller Unit
MIPS	Million Instructions Per Second
NMOS	N-Channel Metal-Oxide Semiconductor
PC	Personal Computer
PHY	Physical layer
PMOS	P-Channel Metal-Oxide Semiconductor
PoE	Power over Ethernet
PRR	Packet Reception Ratio

Bibliography

QoS	Quality of Service
RAM	Random-Access Memory
RF	Radio Frequency
ROM	Read-Only Memory
RSSI	Received Signal Strength Indicator
RTL	Register-Transfer Level
SEU	Single Event Upset
SI	Secure Instance
SPI	Serial Peripheral Interface Bus
SRAM	Static Random Access Memory
ULP	Ultra Low-Power
WSAN	Wireless Sensor Actor Network
WSN	Wireless Sensor Network

Publications

Undervolting in WSNs – Theory and Practice

Ulf Kulau, Felix Büsching and Lars Wolf

Institute of Operating Systems and Computer Networks

TU Braunschweig

Email: [kulau|buesching|wolf]@ibr.cs.tu-bs.de

Abstract—Adapting the voltage level of a processing unit to the actual system load leads to an increased energy efficiency. The aim is to extend the battery lifetime, but existing voltage scaling approaches still act with reserve. To reduce the energy consumption even further, it is possible to power the electric circuits below the specified voltage levels. Considering the processing unit of Wireless Sensor Nodes, this ‘undervolting’ saves up to 42% of energy per clock cycle and, hence, would lead to a significant longer lifetime of nodes and networks. Contrariwise, operating processors or nodes outside their specifications adds some extra incertitude to the system. In this paper, we analyze the effects of undervolting for a typical wireless sensor node in theory and practice. Moreover we give a detailed insight into the dependencies and used a prototype implementation to characterize the influence of lower-than-recommended voltage levels on the MCU. In addition, the impact of different temperatures is considered as well as the behaviour of an undervolted transceiver unit and, therefore, the effects on the wireless communication. While classical computer applications may contain too many hazards to outweigh the improved energy consumption when using undervolting, we show that it is particularly suitable for Wireless Sensor Networks (WSNs) with a huge potential of saving energy and the opportunity of novel power management approaches on every layer.

Index Terms—Wireless sensor networks, Undervolting, Voltage Scaling, Energy Efficiency, Prototype

I. INTRODUCTION

THE optimization of the energy efficiency of distributed sensing systems plays an important role in the realization and acceptance of prospective WSNs applications. Hence, there are many different approaches that deal with this topic. In this context, [1] shows that lowering the voltage level has an significant impact on the energy efficiency of sensor nodes.

A more common approach is Dynamic Voltage Scaling (DVS) which adapts the voltage level of the node to the actual system load to save energy. The background of DVS is that nearly all components of a sensor node are realized in Complementary Metal-Oxide Semiconductor (CMOS) technology. One of the advantage of CMOS is that the major part of the power dissipation belongs to the dynamic power consumption P_{dyn} . During static operations ($f_{cpu} = 0Hz$) only a negligible leakage current occurs. Furthermore, the dynamic power consumption has a linear dependency on the clock rate (f_{cpu}) and a quadratic dependency on the voltage level (V). Equation 1 shows this relation, where C_L is a parasitic capacity which

depends on the quality of the manufacturing process.

$$P_{dyn} = C_L \cdot f_{cpu} \cdot V^2 \quad (1)$$

Unfortunately, the switching delay of CMOS gates grows at lower voltage levels V . Hence, a certain clock rate (f_{cpu}) needs a specific minimum voltage level (V) to guarantee that a switching operation finishes within $\frac{1}{f_{cpu}}$. DVS uses these relationships and adapts both, the clock rate (f_{cpu}) and the corresponding voltage level ($V(f_{cpu})$) to the actual needed system load. Recently, the advantages of DVS in the application area of WSNs have been studied by several groups, e.g., [2], [3], [4]. Furthermore, in [5] a prototype implementation of a DVS capable wireless sensor node is presented. With regard to equation 1, all these approaches have in common that the energy efficiency of a sensor node is increased – in comparison to a fixed voltage source.

All these strategies avoid unsafe areas of operation by satisfying the specifications for adjusting the voltage level $V(f_{cpu})$ as stated in the respective datasheets. Nevertheless, during some experiments with DVS in wireless sensor nodes, we observed that micro-controller systems are able to work below their recommended voltage levels as well. So, the question came up: What happens if we ignore the specified operating area respectively how far can we go outside it?

An important aspect to be considered for this is that the voltage levels given in the specifications cover the widespread temperature range of microelectronic components. Yet, the threshold voltage (V_{th}) is temperature dependent. Looking at this temperature dependency of CMOS gates in equation 2, a micro-controller has to have the ability to run below $V(f_{cpu})$ under *normal* conditions (whereas V_{th0} is the threshold voltage at the temperature T_0 , α is a temperature coefficient) [6].

$$V_{th}(T) = V_{th0} + \alpha \cdot (T - T_0) \quad (2)$$

It might be argued that we inherently accept a decreased reliability of the sensor nodes, since undervolting may have an impact on the correct execution of a program or the proper functioning of a component. However, if the malfunction risk is limited, we stay very well within the typical WSN assumptions, i.e., that not a single node is of importance but the whole WSN must function properly and, thus, algorithms [7], MAC protocols [8] and routing algorithms [9] for WSNs are designed to be fault tolerant. For such cases, a certain number of nodes can become temporary inoperative without the malfunction of the whole network. Thus, undervolting may just be another cause for known symptoms – with a huge potential to save energy. As we demonstrate in this paper, the

potential for energy demand reduction is indeed significant and can be in the order of 30 %.

The paper is structured as follows: In the next section, general related work regarding undervolting is discussed. Afterwards, we present preliminary studies to analyze the requirements for a prototype implementation. In section IV a testbench for the extensive evaluation is presented and the impact of undervolting on an micro controller unit (MCU) regarding to the gain of energy efficiency, the reliability and the system clock is shown. Then, effects on the wireless communication are presented in section V. Finally, in Section VII we conclude the paper.

II. RELATED WORK

While several DVS approaches for WSNs are presented in various publications [2], [3], [4], [5], as far as we know *active undervolting* has not been studied for WSN so far. In some articles regarding general computer architectures (e.g. [10]) undervolting is mentioned as a possibility to reduce the power consumption of a processing unit. In [11] undervolting is explicitly specified as a technique which reduces the power dissipation of a Power7 CPU. It is shown that the entire system's power consumption drops by 15.8% up to 26.4%, without affecting the computational performance. This causal way of dealing with a possibly higher error rate is also mentioned in [12]. Through 'probabilistic computing' it was possible to increase the energy efficiency of CMOS gates up to 300%. Yet, a more application-oriented utilization of undervolting can be found in the end-user community of Personal Computers (PCs). The over-clocking scene aims to increase the performance of commercial off-the-shelf (COTS) CPUs. For that, the clock rate is set manually above the level suitable according to the voltage level specifications. To detect errors, a rather vague stress test (e.g. calculation of prime numbers) is used to check the stability of the CPU. Nevertheless, for the reliable operation of classical computer applications the usage of undervolting may contain too many hazards to outweigh the improved energy consumption. However, for several WSN scenarios the situation is different since there suddenly appearing or failing nodes may be the rule rather than the exception. This implies that self-stabilizing algorithms [13] or self-repairing routing schemes [9] are necessary because of the underlying unreliable sensor network. Nodes could be lost, damaged, or (e.g., due to the use of cheap hardware components) generate unpredictable failures – and all these have to be covered.

One of the possible sources of errors in WSNs is already undervolting, but this occurs only as a side-effect whenever a sensor node is powered through a battery directly which might be nearly exhausted. For example the WSN of [14] includes the common sensor node platform TelosB [15]. The nodes are directly powered by two Mignon (AA) batteries so that the system voltage depends on the batteries' state. At the end of the batteries lifetime, it will happen that the voltage level drops below the recommended voltage for the micro-controller's respective clock speed. The sensor node operates undervolted, but it still operates. Moreover, WSNs may be exposed to highly

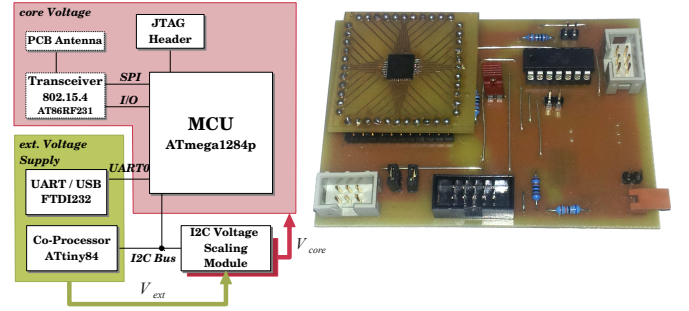


Figure 1. Block diagram and picture of the prototype implementation.

varying environmental conditions [16] which have a significant impact on the reliability of the network [17]. One reason relies on the temperature dependency of the absolute minimum voltage level (cf. equation 2). Hence, the used methods to allow active undervolting on sensor nodes can be used vice versa to increase the robustness against rough environmental conditions.

In summary, to the best of our knowledge, there is no related work which considers undervolting as a regular technique to increase the energy efficiency of small embedded devices like sensor nodes and ensures the operation by adaptive methods.

III. PROTOTYPING AND PRELIMINARY STUDIES

In general, the architecture of a wireless sensor node can be divided into three major parts: Processing unit, transceiver unit, and sensing unit. Mainly all components are based on CMOS technology. Thus, with regard to equation 1, a downscaled voltage level has a positive effect on the energy efficiency of all parts. Nevertheless, in this section we focus on the processing unit only. The *processing unit* is the central component which executes the software and is responsible to control all other parts of the node. Due to the low power constraints in WSNs, usually a small ultra low power MCU is used for this purpose. In this paper, we use the common ATmega1284p 8-Bit micro-controller [18] for the analysis of undervolting. This MCU is used in existing sensor node platforms such as [19], [20].

A. Hardware Prototype

Based on the prototype presented in [5] we prepared an undervolting capable testbed. Figure 1 shows a block diagram and a picture of the implementation, whereby we borrowed the basic design and the voltage scaling module. In addition, a second tiny micro-controller has been added, which is called the Secure Instance (SI). The ATmega1284p is undervolted and, thus, might not be reliable. Hence, the tiny co-micro-controller can be used to validate checksums or help to recalibrate the clock rate (cf. section IV).

B. Clock Rate issues through Undervolting

When the system clock of the ATmega1284p is provided by the internal RC-oscillator, a stable clock rate can not be guaranteed while using undervolting. The datasheet [18] shows

a significant dependency between the oscillator speed and the voltage level even for recommended voltage levels. Thus, one question is whether we will have to recalibrate the clock rate if we want to use undervolting. Initially we measured the deviation from a claimed clock rate (f_{cpu}) as a function of the deviation of the nominal voltage level with an oscilloscope.

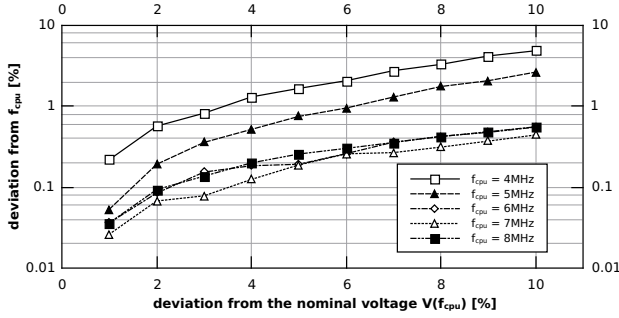


Figure 2. Clock rate: impact of undervolting and recalibration speed.

Figure 2 shows the impact on different clock rates; it can be seen that the deviation is significant. For example, we assume a clock rate of $f_{cpu} = 4$ MHz at the nominal voltage level $V(f_{cpu})$. With an undervolting of $V(f_{cpu}) - 10\%$ the oscillation of the internal RC oscillator decreases, so that the clock rate is reduced by about 4.86%.

In first instance a lower clock rate has minor effects on pure arithmetic logical unit (ALU) operations. Provided that no other errors occur through the undervolting, only the execution time rises. On the other hand, time critical processes like Operating System (OS) timer (e.g. within Contiki OS [21]) or asynchronous communication (e.g. Universal Asynchronous Receiver/Transmitter (UART)) are not reliable anymore. For example, the error rate of a UART communication grows. Under normal operation with $f_{cpu} = 4$ MHz, $V = V(f_{cpu})$ and a baud rate of 9600 BAUD an error of $\varepsilon_{UART} < 0.2\%$ is negligible. The measured worst case of $f_{cpu} = 3.4655$ MHz at $V = V(4 \text{ MHz}) - 0.28V$ ends up in a communication error of $\varepsilon_{UART} < 16\%$ [18].

To demonstrate a more practical issue related to the wireless communication, the data rate of an undervolting sensor node was measured. For this purpose the node runs at a nominal clock rate $f_{cpu} = 4$ MHz and uses Contiki OS [21] with the RIME communication stack. The packet size was set to 90 Byte.

Figure 3 shows the result of this measurement. Firstly, it should be noticed that during this evaluation no impact on the packet loss was recognized. However, it can be seen that a lower voltage level forces a lower data rate. The main reason for this effect is also related to the drop of the MCU's clock rate. Both the calculation of the communication stack and the low level interaction between the MCU and the transceiver via Serial Peripheral Interface Bus (SPI) is slowed down. Although the wireless communication is not affected in principle, other issues like a higher energy consumption due to longer up-times of the transceiver unit could arise. All in all the clock rate of the MCU affects many parts of the sensor node. Hence, to guarantee a reliable use of a node while using undervolting,

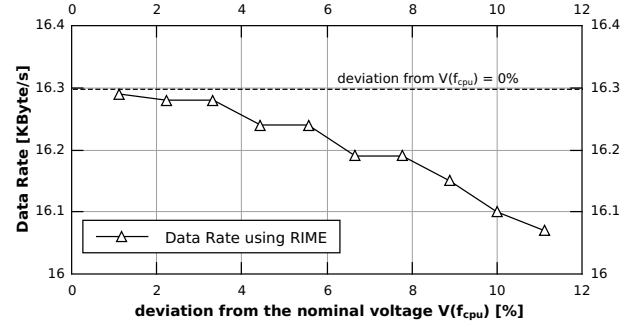


Figure 3. Drop of the data rate (RIME) due to lower clock rates when lowering the voltage level

it is necessary to recalibrate the clock rate.

IV. TESTBENCH

With the aid of the hardware prototype presented in section III-A we implemented a testbench to allow an extensive evaluation of undervolting on wireless sensor nodes. A block diagram of this implementation is given in Figure 4 which shows the general interaction between all components.

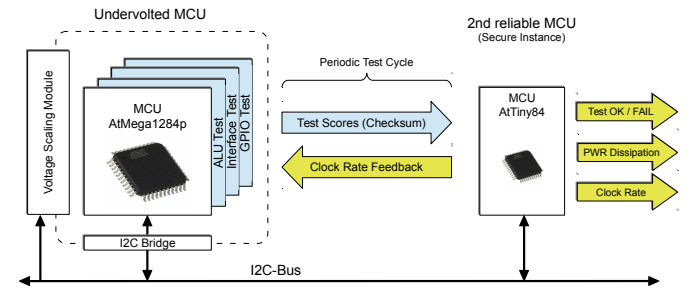


Figure 4. Testbench implementation to evaluate the effects of undervolting on sensor nodes

The interconnection between the voltage scaling module, the MCU (device under test) and the second reliable MCU (SI) is realized via an Inter-Integrated Circuit (I²C)-bus. At the beginning of every test cycle the undervolting MCU executes some tests. A more detailed description about these test (ALU, GPIO, interfaces) will be given within the following sections IV-A and IV-B. Afterwards, the test scores are sent to the SI via I²C frame. In turn, the SI validates the test scores and displays whether a failure caused by undervolting occurred or not. As the counter-check of test scores takes place periodically, the SI is able to measure the time between the current I²C frame and the last frame that has been sent. Thus, the SI has an indirect information about the actual system clock. Moreover the SI is able to measure the current consumption by using a shunt and a difference amplifier. However, in addition a more accurate multimeter and oscilloscope were used to examine the power dissipation during all evaluations.

A. Recalibrating the Clock

The need to recalibrate the clock rate of an undervolting MCU was already demonstrated in section III-B. As mentioned

above, the basic idea is to measure the time between two frames of the periodic test cycle (cf. Figure 4). A dropped clock rate would lead to a prolonged period which gives an indirect information about the deviation of the actual clock rate from the nominal one. Based on this deviation, the SI can decide whether the clock rate has to be adapted or not. Therefore, a binary information is sent to the MCU which forces the MCU to increment or decrement the internal RC-oscillator calibration register (OSCCAL). This linear recalibration approach is shown in Figure 5 at a nominal clock rate of $f_{cpu} = 4$ MHz.

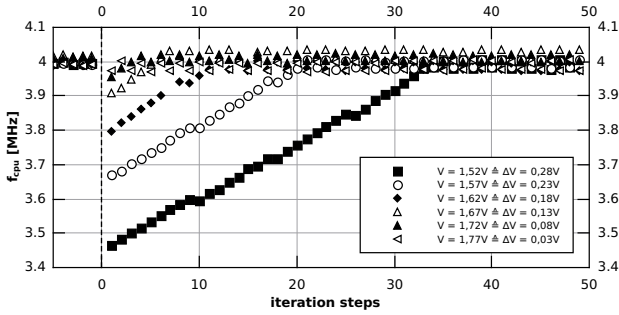


Figure 5. Clock rate: iterative recalibration of the clock rate.

We can see that it is generally possible to generate a stable clock rate even with undervolting. This result is important because it is a major requirement that undervolting should not influence the further usage of the MCU. Of course, the linear recalibration approach could be improved, so that the claimed clock rate could be reached in less iteration steps.

B. Undervolting on MCUs - Characteristics

For the first tests we ignore a dedicated view on the actual temperature and define the room temperature $T_{lab} \approx 22^\circ\text{C}$ as the normal operation condition. We ordered several ATmega1284p from three different distributors to minimize the risk of analyzing parts from the same product line. Based on the results of several tests, we show the behavior of an undervolted MCU in this section.

Various failures are possible when using undervolting on an MCU and they are not predictable without explicit knowledge about the internal structure of the integrated circuit (e.g. VHDL models). On a wireless sensor node the MCU controls all other parts so that a failure will effect the whole system. Accordingly, the question is how low the voltage level V can drop below the recommended $V(f_{cpu})$ without any recognizable effects on the practical usage of the MCU. Furthermore, for further generic solutions it is necessary to know, if every ATmega1284p has an individual probability distribution of failures or whether all are similar.

A full test of every component of the MCU is almost impossible. For example, the test of only one ALU operation (e.g. add) with $n = 2$ input variables of $m = 8\text{Bit}$ each would need a test vector of $(2^m)^n = 2^{16}$ cases. Another possibility would be an IEC60730 Class B test, which is intended to prevent unsafe operation of micro-controllers [22]. However,

with the intention of further usage of undervolting as a regular power management technique, the execution time of such a Class B test bears no relation to the expected gain. For this reason we strike a balance between the reliability and the overhead of fault detection.

A scope where the online detection of errors plays a significant role is the aerospace. In [23] an adequate method to detect single event upsets on AVR core based microprocessors is presented. It turns out that the ALU is obviously the most critical point where fault injections are manifested. Hence, in assumption that undervolting will inject unpredictable failures to the AVR core too, the spot test is mainly focused on the ALU. Moreover, [23] examined different algorithms with regard to the detectability and occurrence of errors. It was shown that a simple matrix calculation can be used to manifest about 86% of ALU errors. Based on this results we designed our spot test as described in the following equation 3.

$$\text{checksum} = \det(A \cdot B) \quad \text{with } A, B \in \mathbb{R}^{n \times n} \quad (3)$$

Nevertheless, it should be noticed, that an absolute failure detection is not possible. The fault tolerant behavior of a WSN should be able to handle sporadic failures even if we are not able to detect them here (cf. sections I, II). The peripherals tests are restricted to the I²C, SPI, UART and General Purpose Input/Outputs (GPIOs). These checks are also only spot tests and should show that peripherals are generally able to run below the recommended voltage level. As described above, the clock rate was recalibrated continuously during the tests of the peripheral's functionality.

During all tests with undervolting up to a certain level no error of the peripherals occurs. We can observe that an operating point exists, where the ALU execution breaks down. The Figures 6(a), 6(b) and 6(c) give the results of the functionality analysis for three different instances of the MCUs. From this we can derive some statements which help for the further considerations.

- In general it is possible to operate an MCU with undervolting. Up to a specific operating point, the probability that a failure occurs is very low.
- Malfunction starts to appear in a small, sharp region around this point. The characteristic curve for the output signal as a function of the input voltage correlates with the measured results [6].
- Although we use the same kind of micro-controller (ATmega1284p), the absolute minimum voltage level of the three MCU instances differs. On MCU1 with $f_{cpu} = 4\text{MHz}$ the first errors occur at a deviation of $\delta_V = 12.22\%$ from $V(f_{cpu})$. At this point MCU2 and MCU3 run stable without any failures.
- The possible deviation from $V(f_{cpu})$, i.e., where no errors occur, grows with higher clock rates.

1) *Power Consumption:* The overall goal of this paper is to explore the feasibility and potential of undervolting to increase the average energy efficiency of a sensor node. Hence, the impact of undervolting on the power dissipation of the processing unit is a matter of particular interest. Since our approach operates outside the datasheet specifications, we have

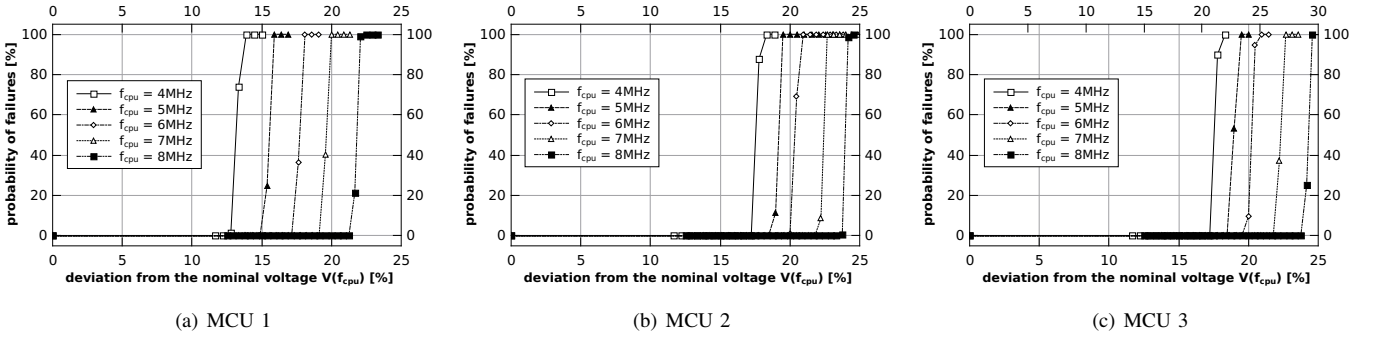


Figure 6. Functionality analysis for three different instances of the MCUs

no information about the typical characteristics how the current consumption reacts on undervolting. Thus, we measured the current consumption of three MCUs at different clock rates f_{cpu} as a function of the deviation (in percent) from the recommended voltage level $V(f_{cpu})$.

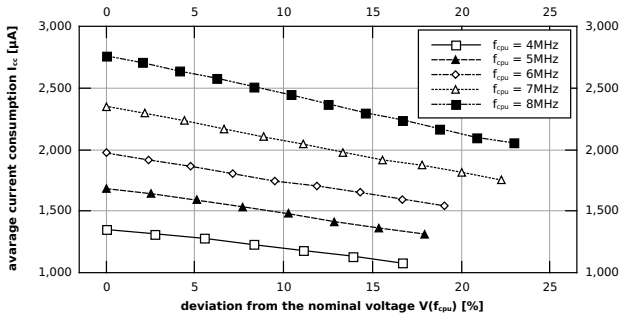


Figure 7. Current consumption vs voltage (mean value).

Figure 7 shows the average results of the measurements. As expected, it can be seen that the current consumption of an MCU decreases with lower voltage levels. In addition, the averaged, minimum and maximum standard deviation for this measurement is given in Table I.

Table I
AVERAGED, MINIMUM AND MAXIMUM STANDARD DEVIATION OF THE CURRENT CONSUMPTION OF THE TESTED MCUS

$f_{cpu}[MHz]$	4	5	6	7	8
$mean(\sigma_c)[\mu A]$	10.913	15.333	17.807	16.458	27.828
$max(\sigma_c)[\mu A]$	22.489	27.016	21.404	19.096	30.920
$min(\sigma_c)[\mu A]$	4.832	5.915	11.820	13.851	25.261

It can be seen that the value of the current consumption of MCUs is almost equal for a given supply voltage. From this it follows that undervolting will add heterogeneity in terms of the energy efficiency to a network. Due to individual minimum voltage levels, the energy efficiency of nodes of a WSN becomes individual as well (cf. Section VI).

Independent of the clock rate f_{cpu} , the relation between the percental deviation δ_v and the percental saving of the current consumption δ_i can be described as follows:

$$\delta_i \approx \frac{6}{5} \cdot \delta_v \quad (4)$$

A huge advantage of undervolting is that it has no other effects on the usage of an MCU. From this it follows, that we do not have any conflicts when using additional energy-aware concepts like dynamic power management (DPM), DVS, or low power listening (LPL). With regard to DVS, Figure 8 shows the energy per clock cycle as a function of δ_v . This

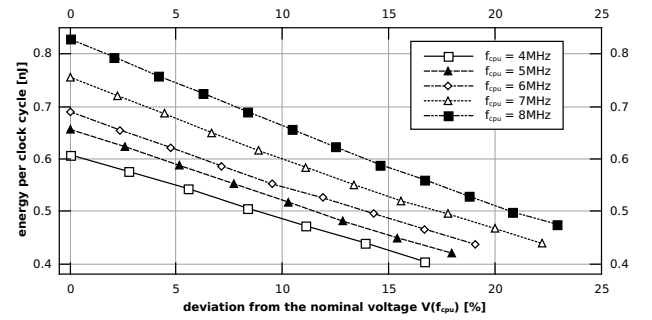


Figure 8. Energy per clock cycle vs voltage (mean value).

information about the relation between energy, voltage, and clock rate is important because this influences the way how a DVS should be implemented. Basically there are two different strategies:

- 1) A task is executed at the highest clock rate to switch to the energy-aware sleep state as soon as possible.
- 2) The clock rate is adapted so that a task meets a deadline exactly.

In this case, i.e., for the measurement results we have, the second option would be the best choice. The reason is that the curves 'energy per clock cycle' of the different clock rates (cf. Figure 8) do not show an intersection.

Table II
ENERGY SAVINGS WHEN USING UNDERVOLTING ON ATMEGA1284P COMPARED TO THE RECOMMENDED VOLTAGE LEVEL $V(f_{cpu})$.

$f_{cpu}[MHz]$	4	5	6	7	8
$E_{min}[pJ]$	404.0	420.2	437.0	466.8	475.0
$mean(\delta_e)[\%]$	16.60	18.30	19.06	20.02	22.27
$max(\delta_e)[\%]$	33.52	35.96	36.67	38.23	42.66

Table II summarizes the significant energy saving potentials of undervolting by stating maximum percental energy saving

$\max(\delta_e)$, average percental energy saving $\text{mean}(\delta_e)$, minimum energy per clock cycle $E_{\min}$.

2) *Influence of Temperature*: The initial idea of undervolting for WSNs belongs to the presumption, that a MCU has to be tolerant against extreme temperatures, but this is not always needed. Hence, for the previous observations we assumed a 'normal' operating environment. The tests were done indoor with a moderate temperature of $T_{\text{lab}} \approx 22^\circ\text{C}$. However, some application areas of WSNs have extreme temperature requirements, e.g., cold temperatures [24].

If a sensor node runs with undervolting and if the environmental temperature changes, the first question is in which direction we have to adapt the voltage level to avoid an increase of failures. To study the effects of temperature changes, we performed several experiments. For this, the MCUs under test are heated up and cooled down with a peltier element. As expected, we observed that a higher temperature T allows lower supply voltages. For example, the break-even point for MCU1 at $f_{\text{cpu}} = 4\text{MHz}$ moves from $\delta_V = 12.22\%$ ($T_{\text{lab}} \approx 22^\circ\text{C}$) up to $\delta_V = 15\%$ ($T_{\text{lab}} \approx 50^\circ\text{C}$).

Afterwards a climatic chamber was used to get more detailed information about the relation between the temperature and the break even point. The results in Figure 9 validate the trend of the temperature dependency.

- A lower temperature increases the risk to get failures. The voltage level has to raise.
- A higher temperature involves the chance to increase the energy efficiency. The voltage level can be reduced to save more energy.

However, a deliberate temperature increase with the intention to save more energy through more extensive undervolting is not recommended. The aging of components accelerates with higher temperatures [25].

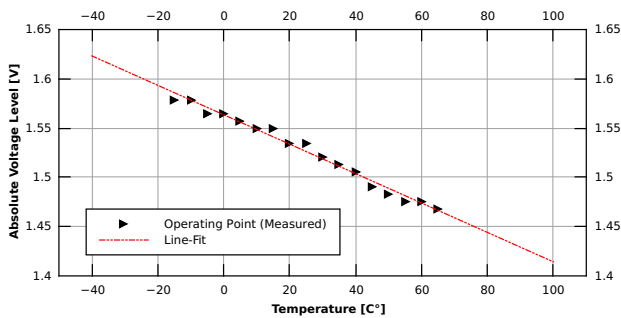


Figure 9. Minimum Voltage Level vs. temperature dependency.

3) *Limitations and Interim Conclusions*: Our approach is based on the ATmega1284p micro-controller. An obvious question is how other processing units behave. Indeed, the usage of undervolting is not limited to the used micro-controller. Referring to equation 2, the relation between the minimum threshold voltage and the temperature is a general effect of all parts based on CMOS technology. The main variation between the used and other MCUs is the specific architecture. Thus, the critical path may differ where a failure caused by undervolting is manifested. With adequate methods to detect these failures like [26], [27], other MCUs are also

able to run at more energy efficient but reliable states. We performed some short tests with an msp430 MCU [28] of the TelosB [15] sensor node. These tests showed that also this MCU can run below its recommended voltage level.

Another question is, how other parts of a sensor node (sensors, actors) react on undervolting. With regard to the sensing quality an undervolting of sensors is not recommended. A sensor converts the physical world into electrical signals. These information is converted via an analog-digital converter which makes the data utilizable for the processing unit. To avoid that the analog section of the sensors could suffer from undervolting, our approach is bound to the core functionality of a sensor node: The processing and transceiver units. Therefore, sensing errors are not caused by a sensor itself.

A mainstay of our argumentation to use undervolting on wireless sensor nodes is that WSNs are fault tolerant per se. Moreover, as described above the sensing unit will deliver true values about the environmental state. Anyway, the undervolted MCU can effect calculation errors or bit flips. Within a redundant network topology those failures can be detected and fixed. To reduce the probability of failures a software redundancy (oversampling) can be used. In addition statistics about how trustful a sensor node runs with undervolting can help to decide which is the best strategy in such cases.

The overhead for implementing undervolting on a sensor node is manageable. The potential energy savings grant additional hardware components like the SI. Moreover, the computational requirements of the SI are less and its usage is periodic with low duty cycles.

V. EFFECTS ON WIRELESS COMMUNICATION

Undervolting works for an MCU, but what if we include the transceiver unit into the same voltage path? The ability to communicate with other nodes is very important for every wireless sensor node. Thus, if undervolting would have a negative impact on this, the consequence could be a collapse of the WSN structure due to undervolting. However, temporary appearing and disappearing nodes can be handled through tolerant routing protocols like [29]. A real problem occurs when a node ends up isolated from the rest of the network. The isolated node would not be able to decide whether undervolting is responsible for its inability to communicate or just no messages are transceived and the transceiver unit is still fine. Two simple but inefficient approaches would avoid this issue:

- The transceiver unit is connected to a separated voltage path V_{rtx} . The voltage level of V_{rtx} is set to the requested value of the transceiver.
- The voltage level is temporary set to the recommended level of the transceiver unit.

A more efficient but daring solution would be:

- The transceiver unit is potentially undervolted too and the appearance of failures indicates the edge of functionality (cf. section IV-B).

With regard to the last alternative and if the sensor node uses, as it is common, an Automatic Repeat reQuest (ARQ)

mechanism for reliable data transfer such as provided by the IEEE 802.15.4 standard for wireless communication, the detection of failures when transmitting data is already covered. Then a sensor node can take an increasing number of retransmissions as a sign that undervolting effects the transmission of data. Thus, the node can react itself and adapt the voltage level to higher values. Contrariwise, when a node receives no data it can not decide whether other nodes do not send any data or undervolting prohibits the reception. Hence, a WSN heartbeat could be sent periodically to all nodes. In scenarios without moving nodes, the absence of a heartbeat indicates an error. Based on information such as provided by the heartbeats or the ARQ mechanisms, a heuristic can be derived which decides if the voltage level has to be increased or if the actual failures are caused by other reasons.

To determine the behavior of the transceiver unit if we use undervolting, we measured the probability that a communication error occurs as a function of the voltage level. We used the AT86RF231 [30], a fully integrated transceiver unit which has a minimum recommended voltage level of $V_{rtx} = 1.8V$. For these measurements, the node uses Contiki OS [21] with the RIME communication stack. To detect failures during the communication, the undervolted sensor node stays in idle listening until it receives a request message from a second node. This message is answered subsequently. In practice both, the reception and the transmission of data break down at once. The result is shown in Figure 10.

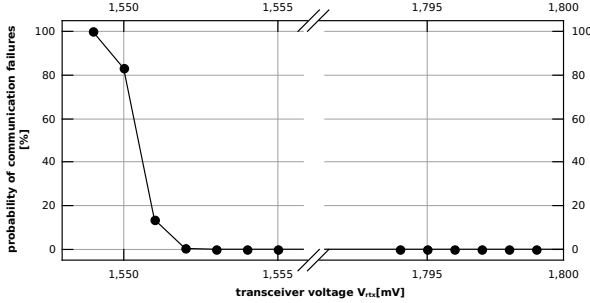


Figure 10. Undervolting induced communication errors.

It is well-known that the usage of the transceiver is one of the most power consuming process on a sensor node. Therefore, the influence of undervolting on the transceiver's current consumption is of particular interest. As the transceiver has a fixed transmit and reception power, the current consumption is less bound to the voltage level. The datasheet [30] underlines this trend of a weak relation between V_{rtx} and the current consumption. Nevertheless, the transceiver unit is not the only component of a sensor node which is involved in the wireless communication. For this reason the power dissipation of the transceiver plus the processing unit was measured while sending 90 Byte payload via RIME communication stack. Thus, the costs per transmitted byte at different deviations of the nominal voltage level can be evaluated.

Obviously, Figure 11 shows that the usage of undervolting reduces the costs per transmitted bit. The impact while running the MCU at $f_{cpu} = 4$ MHz is even higher because of the abso-

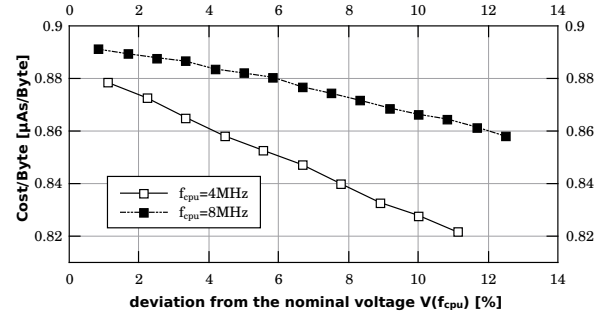


Figure 11. Transmit costs per byte when using Contiki OS rime stack and undervolting.

lute minimum voltage level at this clock rate. The transceiver unit is provided with the same voltage level as the processing unit (cf. Section III-A). As the absolute minimum voltage level of the used MCU at $f_{cpu} = 8$ MHz is $\geq 1.8V$, it will always meet the specifications of the minimum voltage level of the transceiver. However, an MCU running at $f_{cpu} = 4$ MHz will force the transceiver to be undervolted too. Moreover, during preliminary studies we also analyzed the relation between the voltage level and different traffic loads. It turns out that the impact of the voltage level remains almost steady regardless of the packet size. Table III summarizes the gain of the cost per bit when lowering the voltage level by $\Delta V = 0.6V$. For this evaluation Contiki OS and the RIME stack were used while the MCU was running at $f_{cpu} = 8$ MHz.

Table III
GAIN OF THE COST PER BIT WHEN REDUCING THE SUPPLY VOLTAGE BY $\Delta V = 0.6V$ AT DIFFERENT PACKET SIZES

Packet Size [Byte]	10	30	50	70	90
gain δ [%]	11.25	10.53	9.90	10.04	9.99

On the other hand, reducing the voltage level below the specifications might effect the link quality of the transceiver unit. According to [30] the transmission power decreases with lower voltage levels. Yet, during our measurements we do not observe an abnormal behavior of received signal strength indicator (RSSI) and link quality indicator (LQI) (distance $\approx 1m$) due to undervolting.

Overall we can conclude that, in general, it is possible to drive also the transceiver unit with undervolting. However, so far we have a limited set of experiments with that only.

VI. BENEFITS FOR MULTI-HOP WIRELESS SENSOR NETWORKS

Assuming a whole WSN consists of undervolting capable sensor nodes – what are the effects for this network when using this power management technique?

For this purpose four more MCUs were analyzed regarding their absolute minimum voltage level. The results for a total of 7 MCUs at $f_{cpu} = 4$ MHz and $f_{cpu} = 8$ MHz are given in Figure 12.

Obviously, with regard to the energy efficiency, undervolting adds heterogeneity to the network. When dealing with constant

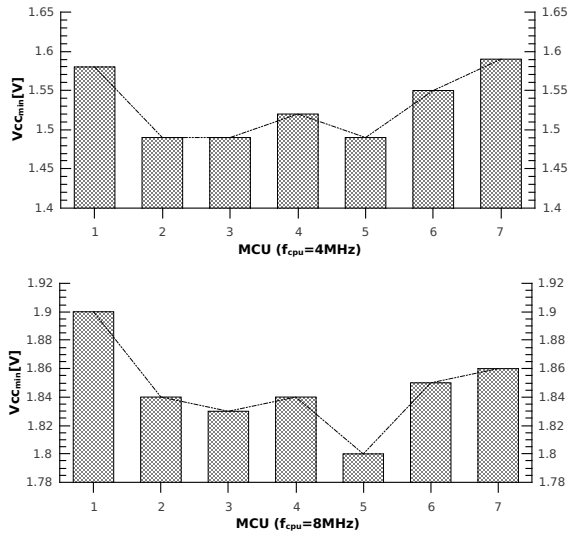


Figure 12. Undervolting adds heterogeneity. Absolute minimum voltage levels of 7 MCUs

supply voltages, the energy consumption of the nodes is almost comparable. However, due to the adaptation to individual optimal operating points, some nodes will become more efficient than others. To maximize the lifetime of WSNs – like in [31] and [32] – the rearrangement of network structures could also take the effectiveness of the nodes into account.

Moreover, the potentially highly varying environmental conditions of a WSN should be considered, e.g., sensor nodes could be exposed to a temperature differences up to 65°C during a single day [17]. The ability of nodes to run at lower voltage levels due to higher temperatures was already shown in Section IV-B2. Thus, the energy efficiency of a sensor node, respectively the WSN in general, becomes dependable on its surrounding (temperature) conditions. In future work this effect could be used to implement more advanced energy harvesting methods.

VII. CONCLUSION

In this study we have shown that undervolting saves up to 42% of the energy per clock cycle compared to the minimum recommended voltage level of the used MCU. Hence, the lifetime of nodes and networks enhances significantly. With regard to the various application area of WSNs we considered the influence of different temperatures on the concept of undervolting as well. The strong temperature dependency of the absolute minimum voltage level and the individual capabilities of every part adds a strong heterogeneity to the network. Thus, existing concepts for network structuring, load balancing or routing can be optimized on every layer. The presented results show that undervolting is a new chance to increase the energy efficiency of WSNs. The already existing fault tolerance of WSNs offers the ideal basis for this power management technique. In conclusion, the lifetime of real-world WSNs and applications could be improved heavily. The individual applicability for undervolting depends on different factors, thus, instead of generic static approaches, dynamic and adaptable algorithms

should be applied. Currently, we are working on a self-learning algorithm that is capable of predicting future system states by, e.g., taking the trend of environmental temperatures into account. Additionally, a large scale real world setup is planned and designed to analyze the capabilities of undervolting in outdoor WSNs.

REFERENCES

- [1] U. Kulau, F. Büsching, and L. Wolf, “Undervolting in WSNs – A Feasibility Analysis,” in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, March 2014, pp. 553–558.
- [2] W. Tuming, Y. Sijia, and W. Hailong, “A dynamic voltage scaling algorithm for wireless sensor networks,” in *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, vol. 1, aug. 2010, pp. V1–554 –V1–557.
- [3] T. Hamachiyo, Y. Yokota, and E. Okubo, “A Cooperative Power-Saving Technique Using DVS and DMS Based on Load Prediction in Sensor Networks,” in *Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on*, July 2010, pp. 7 –12.
- [4] L. B. Hoermann, P. M. Glatz, C. Steger, and R. Weiss, “Energy Efficient Supply of WSN Nodes using Component-Aware Dynamic Voltage Scaling,” *Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), 11th European*, pp. 1 –8, April 2011.
- [5] U. Kulau, F. Büsching, and L. Wolf, “A Node’s Life: Increasing WSN Lifetime by Dynamic Voltage Scaling,” in *Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on*, 2013, pp. 241–248.
- [6] R. J. Baker, *CMOS: Circuit Design, Layout, and Simulation*, 3rd ed. IEEE Press, 2010.
- [7] J. Suomela, “Survey of local algorithms,” *ACM Comput. Surv.*, vol. 45, no. 2, pp. 24:1–24:40, Mar. 2013. [Online]. Available: <http://doi.acm.org/10.1145/2431211.2431223>
- [8] W. Lee, A. Datta, and R. Cardell-Oliver, “FlexiMAC: A flexible TDMA-based MAC protocol for fault-tolerant and energy-efficient wireless sensor networks,” in *Networks, 2006. ICON ’06. 14th IEEE International Conference on*, vol. 2, 2006, pp. 1–6.
- [9] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, “RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks,” RFC 6550 (Proposed Standard), Mar. 2012. [Online]. Available: <http://www.ietf.org/rfc/rfc6550.txt>
- [10] P. Bouvry, H. González-Vélez, and J. Kolodziej, *Intelligent Decision Systems in Large-Scale Distributed Environments*, ser. Studies in Computational Intelligence. Springer, 2011, vol. 362.
- [11] M. Floyd, M. Allen-Ware, K. Rajamani, B. Brock, C. Lefurgy, A. Drake, L. Pesanté, T. Gloekler, J. Tierno, P. Bose, and A. Buyuktosunoglu, “Introducing the Adaptive Energy Management Features of the Power7 Chip,” *Micro, IEEE*, vol. 31, no. 2, pp. 60–75, March–April.
- [12] G. Anthes, “Inexact design: beyond fault-tolerance,” *Commun. ACM*, vol. 56, no. 4, pp. 18–20, Apr. 2013. [Online]. Available: <http://doi.acm.org/10.1145/2436256.2436262>
- [13] J. Ben-Othman, K. Bessaoud, A. Bui, and L. Pilard, “Self-stabilizing algorithm for energy saving in Wireless Sensor Networks,” in *Computers and Communications (ISCC), 2011 IEEE Symposium on*, 2011, pp. 68–73.
- [14] L. Mo, Y. He, Y. Liu, J. Zhao, S.-J. Tang, X.-Y. Li, and G. Dai, “Canopy closure estimates with GreenOrbs: sustainable sensing in the forest,” in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys ’09. New York, NY, USA: ACM, 2009, pp. 99–112. [Online]. Available: <http://doi.acm.org/10.1145/1644038.1644049>
- [15] J. Polastre, R. Szewczyk, and D. Culler, “Telos: enabling ultra-low power wireless research,” in *IPSN ’05: Proceedings of the 4th international symposium on Information processing in sensor networks*. Piscataway, NJ, USA: IEEE Press, 2005, p. 48.
- [16] I. Talzi, A. Hasler, S. Gruber, and C. Tschudin, “PermaSense: investigating permafrost with a WSN in the Swiss Alps,” in *Proceedings of the 4th workshop on Embedded networked sensors*, ser. EmNets ’07. New York, NY, USA: ACM, 2007, pp. 8–12. [Online]. Available: <http://doi.acm.org/10.1145/1278972.1278974>
- [17] C. A. Boano, H. Wennerström, M. A. Zúñiga, J. Brown, C. Keppitayagama, F. J. Oppermann, U. Roedig, L.-Å. Nordén, T. Voigt, and K. Römer, “Hot Packets: A systematic evaluation of the effect of temperature on low power wireless transceivers,” in *Proceedings of the*

- 5th *Extreme Conference on Communication (ExtremeCom)*. ACM, Aug. 2013, pp. 7–12.
- [18] Atmel, “ATmega164A/164PA/324A/324PA/644A/ 644PA/1284/1284P,” 2010. [Online]. Available: http://www.atmel.com/dyn/resources/prod_documents/doc8272.pdf
- [19] F. Büsching, U. Kulau, and L. Wolf, “Architecture and Evaluation of INGA - An Inexpensive Node for General Applications,” in *Sensors, 2012 IEEE*. Taipei, Taiwan: IEEE, oct. 2012, pp. 842–845.
- [20] R. Mangharam, A. Rowe, and R. Rajkumar, “FireFly: a cross-layer platform for real-time embedded wireless networks,” *Real-Time Systems*, vol. 37, pp. 183–231, 2007, 10.1007/s11241-007-9028-z. [Online]. Available: <http://dx.doi.org/10.1007/s11241-007-9028-z>
- [21] A. Dunkels, B. Gronvall, and T. Voigt, “Contiki - a lightweight and flexible operating system for tiny networked sensors,” in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, nov. 2004, pp. 455 – 462.
- [22] Atmel, “AVR998: Guide to IEC60730 Class B compliance with AVR microcontrollers,” 2008.
- [23] A. Rohani and H.-R. Zarandi, “An Analysis of Fault Effects and Propagations in AVR Microcontroller ATmega103(L),” in *Availability, Reliability and Security, 2009. ARES '09. International Conference on*, 2009, pp. 166–172.
- [24] M. Doering, S. Rottmann, and L. Wolf, “Demonstration of a Snowpack Monitoring System based on Inexpensive Sensor Nodes and solar-powered Backhaul Links,” in *Proceedings of the 4th Extreme Conference of Communication (ExtremeCom 2012), Zurich, Switzerland*, 3 2012.
- [25] F. Reynolds, “Thermally accelerated aging of semiconductor components,” *Proceedings of the IEEE*, vol. 62, no. 2, pp. 212–222, Feb.1974.
- [26] A. Merentitis, N. Kranitis, A. Paschalis, and D. Gizopoulos, “Low Energy Online Self-Test of Embedded Processors in Dependable WSN Nodes,” *Dependable and Secure Computing, IEEE Transactions on*, vol. 9, no. 1, pp. 86–100, Jan 2012.
- [27] A. Benso, A. Bosio, P. Prinetto, and A. Savino, “An on-line software-based self-test framework for microprocessor cores,” in *Design and Test of Integrated Systems in Nanoscale Technology, 2006. DTIS 2006. International Conference on*, Sept 2006, pp. 394–399.
- [28] Texas Instruments, “MSP430F15x, MSP430F16x, MSP430F161x Mixed signal microcontroller,” 2002. [Online]. Available: <http://www.ti.com/lit/ds/symlink/msp430f1611.pdf>
- [29] W.-B. Pöttner, O. Wellnitz, and L. Wolf, “QoS-AODV6E: An Energy-Balancing QoS Routing Scheme for WSNs,” in *Proceedings of the Seventh International Conference on Networked Sensing Systems (INSS2010)*, Kassel, Germany, June 2010. [Online]. Available: <http://www.ibr.cs.tu-bs.de/papers/poettner-inss2010-qosaodv6e.pdf>
- [30] Atmel, “AT86RF231, Low Power 2.4 GHz Transceiver for ZigBee, IEEE 802.15.4, 6LoWPAN, RF4CE, SPI00, WirelessHART and ISM Applications,” 2009. [Online]. Available: http://www.atmel.com/dyn/resources/prod_documents/doc8111.pdf
- [31] F. Büsching, W. Pöttner, D. Brökelmann, G. von Zengen, R. Hartung, K. Hinz, and L. Wolf, “A demonstrator of the ginseng-approach to performance and closed loop control in wsn,” in *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, june 2012, pp. 1 –2.
- [32] R. Elhabyan and M. Yagoub, “Weighted tree based routing and clustering protocol for wsn,” in *Electrical and Computer Engineering (CCECE), 2013 26th Annual IEEE Canadian Conference on*, May 2013, pp. 1–6.



Ulf Kulau received his bachelors degree in engineering in 2010 and his masters degree in computer science in 2012 from TU Braunschweig. He is currently pursuing the Ph.D. degree at TU Braunschweig. His current research interests include Wireless Sensor Networks, especially the design of micro-controller systems (like wireless sensor nodes) and energy management concepts coupled with robustness of WSNs in rough environments.



Felix Büsching is a research assistant at the Institute of Operating Systems and Computer Networks (IBR) at Technische Universität Braunschweig, Germany. He received his diploma degree in engineering from TU Braunschweig, in 2007. In 2013 he received his Ph.D. degree in computer science at TU Braunschweig. His current research interests comprise Wireless Sensor Networks, Mobile Communications, and Ambient Assisted Living.



Lars Wolf received the diploma degree in 1991 and the doctoral degree in 1995, both in computer science. From 1991 to 1996 he worked at IBM's European Networking Center in Heidelberg, Germany, on multimedia transport, resource management, quality of service, media servers and multimedia applications. In 1996 he joined the Technische Universität Darmstadt as assistant professor. There he led a research group working on multimedia networking and quality of service in distributed multimedia systems. Dr. Wolf joined Universität Karlsruhe (TH), Germany, in 1999 where he was associated professor in the computer science department and alternate director of the computer center. Since spring 2002 Lars Wolf is full professor for computer science at the Technische Universität Braunschweig where he is head of the Institute of Operating Systems and Computer Networks.

IdealVolting: Reliable Undervolting on Wireless Sensor Nodes

ULF KULAU, FELIX BÜSCHING, and LARS WOLF, Technische Universität Braunschweig

The energy consumption of Wireless Sensor Networks (WSNs) correlates with the voltage level at which the nodes are powered—a lowered voltage leads to a prolonged lifetime of nodes and networks. Operating nodes at voltage levels below the recommendation—so-called undervolting—saves energy but is accompanied by an increased risk of failures.

In this article, we show that a WSN can still work reliably, even if the voltage recommendations are violated. We show that there is a correlation between temperature and error-proneness at the same voltage level and that ideal voltage levels depend on environmental conditions. Especially in outdoor deployments, temperatures fluctuate often and heavily. Additionally, we show that these ideal voltage levels are different for each individual node. To deal with these individual variations and varying temperatures, we present a supervised learning strategy that is able to keep the nodes in uncritical states even if environmental conditions are constantly and heavily changing while saving as much energy as possible by constantly adapting the voltage level of these nodes to an individually ideal level. All measurements were performed on sensor node prototypes that are also presented in this article. In detailed evaluations, it is shown (i) that a single node will never run in the same unsafe state twice, (ii) that only five known bias points are needed to predict nearly all temperature dependencies for an individual node, and, thus, (iii) that a system of undervolted nodes can be as reliable as a conventionally powered network while prolonging the lifetime by more than 40%.

CCS Concepts: • **Hardware** → **Power and energy**; **System-level fault tolerance**; • *Networks* → *Sensor networks*; • *Computer systems organization* → *Reliability*

Additional Key Words and Phrases: Wireless sensor networks, undervolting, voltage scaling, supervised learning, energy efficiency, robustness

ACM Reference Format:

Ulf Kulau, Felix Büsching, and Lars Wolf. 2016. IdealVolting: Reliable undervolting on wireless sensor nodes. ACM Trans. Sen. Netw. 12, 2, Article 11 (April 2016), 38 pages.
DOI: <http://dx.doi.org/10.1145/2885500>

1. INTRODUCTION

WSNs are becoming increasingly popular and the application areas seem to be nearly unlimited: healthcare, home automation, wildlife monitoring, surveillance, etc., just to mention a few. While indoor deployments of WSNs could be powered by an existing grid, most outdoor deployments run on batteries. Even if outdoor deployments are supported by some kind of energy harvesting, the capacity of such devices is usually fitted to the actual needs of the specific nodes in the envisaged use case [Doering et al. 2011]. Thus, energy supply and power consumption are crucial parts in nearly every outdoor deployment of WSNs.

There are several strategies to reduce the power consumption of nodes and networks. Implementing sophisticated duty cycling schedules is a very promising and often

Authors' addresses: U. Kulau, F. Büsching, and L. Wolf, Technische Universität Braunschweig, Institute for Operating Systems and Computer Networks, Mühlenpfordtstraße 23, 38106 Braunschweig, Germany; emails: {kulau, buesching, wolf}@ibr.cs.tu-bs.de.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.

© 2016 ACM 1550-4859/2016/04-ART11 \$15.00

DOI: <http://dx.doi.org/10.1145/2885500>

practised policy; during the sleep phases of processor, sensors, and radio, the energy consumption is almost negligible. So, the major amount of energy is consumed during the active phase of the node [Min et al. 2014]. When energy saving by duty cycling is exhausted or, if duty cycling is not usable (e.g., as for some monitoring applications), the energy consumption in the active phases has to be reduced.

Voltage scaling can be helpful in such situations. For most components on a node, the energy consumption is directly related to the voltage level. Thus, adapting—usually reducing—the voltage level to the lowest possible level will save energy [Gonzalez et al. 1997; Pouwelse et al. 2001]. The dynamic adaption of the voltage level does not need to be compulsorily performed for the whole node; it can also be adjusted component-by-component or for different voltage paths [Friedrichs et al. 2014].

The specific power level at which the individual components are typically running is usually derived from datasheets or manufacturer’s information. For microprocessors, often a range of voltage levels for a given clock speed is specified. Higher frequencies require a higher voltage level. The reason for this is that the gate switching speed of Complementary Metal-Oxide Semiconductor (CMOS) needs a specific minimum voltage. So, when reducing the clock speed to save energy, it is possible to reduce the voltage level as well, which will save additional energy.

More energy could be saved if the voltage level could be further reduced. Undercutting the specified voltage area is called *undervolting*. When running on batteries, undervolting is performed at the end of the battery lifetime (quasi-accidentally). If a battery discharges, the voltage level drops, and it can be observed that even if the voltage level has dropped under the specified level for the processor, in many cases, the node will still work for a certain amount of time, at least until the voltage is no longer sufficient. In addition to the clock speed, the operating voltage of CMOS-based circuits is also temperature dependent. At lower temperatures, a higher voltage level is needed; at higher temperatures a lower voltage is sufficient. However, although micro controller units (MCUs) can be exposed to rough environments, manufacturers specify only a fixed voltage level to cover the widespread temperature range. In other words, MCUs must be able to operate with the recommended voltage level under several temperature conditions. For this reason, we analyzed the general effects of undervolting on a wireless sensor node in our preliminary studies [Kulau et al. 2015]. The crux is that the enormous potential of energy savings are contradictory to an increased probability of errors when lowering the voltage level too far.

Thus, a proper voltage has to be found where the voltage level is as low as possible but the node is still working correctly and reliably. But such an appropriate voltage is influenced by many factors: Not only the environment and the temperature have an impact on undervolting opportunities, but also the manufacturing process of the integrated circuits leads to differences in the required voltage level. So, the ideal voltage level not only differs for diverse scenarios; it can additionally be different for each individual node.

Thus, this article proposes a self-optimizing approach which is able to deal with these differences and get the best power saving out of each individual node. Nevertheless, considering highly varying environmental conditions when adapting the voltage level to the border of correct functionality might still lead to failures. Thus, the self-optimization is supplemented by a supervised learning algorithm that is capable of predicting future system states by taking environmental changes into account. Both the individual adaption together with the supervised learning process is called *IdealVolting*. Independent of any use case, the sensor nodes running IdealVolting are able to achieve reliable and highly energy-efficient states at all times.

In the following section, we present the technical background for voltage scaling and the relationships among power, voltage, clock rate, and temperature. In Section 3, we introduce the hardware of an adapted wireless sensor node that is capable of dynamic

voltage scaling and therefore allows adaptive undervolting. In Section 4, we present algorithms, implementations, and evaluation for lowering the voltage levels to individual *ideal* levels. Changing environmental conditions for indoor and outdoor deployments is considered in Section 5; Section 6 explains the supervised learning approach in detail. The implementation of the whole system is described in Section 7 and evaluated in detail in Section 8. After discussing the related work in Section 9, summary, conclusion, and outlook are given in Section 10.

2. TECHNICAL BACKGROUND

If we take a closer look at the typical hardware architecture of wireless sensor nodes or other embedded ultra low power (ULP) micro controller systems, it is common practice that most of the used integrated circuits are realized in CMOS technology. One of the advantages of CMOS is that the major part of the power dissipation belongs to the dynamic power consumption P_{dyn} . During static operations ($f_{cpu} = 0Hz$), only a leakage current occurs. For some applications with extra long sleep periods, this leakage current might be an issue and has to be mitigated [Jayakumar et al. 2014]. With a growing number of CMOS gates and a shrinking feature size, the accumulative sub-threshold leakage current might become a problem, too [Kim et al. 2003]. However, in this work, we focus on ULP MCUs where the static power consumption is almost negligible compared to the dynamic power consumption.

The dynamic power consumption itself has a linear dependency on the clock rate (f_{cpu}) and a quadratic dependency on the voltage level (V). Equation 1 shows this relation, where C_L is a parasitic capacity that depends on the quality of the manufacturing process:

$$P_{dyn} = C_L \cdot f_{cpu} \cdot V^2. \quad (1)$$

Unfortunately, the switching delay of CMOS gates grows at lower voltage levels V . Hence, a certain clock rate (f_{cpu}) needs a specific minimum voltage level (V) to guarantee that a switching operation finishes within $\frac{1}{f_{cpu}}$. Dynamic Voltage Scaling (DVS) uses these relationships and adapts both the clock rate (f_{cpu}) and the corresponding voltage level ($V(f_{cpu})$) to the actually needed system load. Recently, the advantages of DVS in the application area of WSNs have been studied by several groups such as Tuming et al. [2010], Hamachiyo et al. [2010], and Hoermann et al. [2011]. With regard to Equation 1, all these approaches have in common that the energy efficiency of a sensor node is increased in comparison to a fixed voltage source.

These strategies avoid unsafe areas of operation by satisfying the specifications for adjusting the voltage level $V(f_{cpu})$ as stated in the respective datasheets. Nevertheless, during experiments with DVS in wireless sensor nodes, we observed that microcontroller systems are able to work below their recommended voltage levels as well.

The important aspect to be considered for this is that the voltage levels given in the specifications cover a wide temperature range of microelectronic components. Yet the threshold voltage (V_{th}) is temperature dependent. Looking at this temperature dependency of CMOS gates in Equation (2), a microcontroller has to have the ability to run below $V(f_{cpu})$ under *normal* specific conditions (whereas V_{th0} is the threshold voltage at the temperature T_0 , α is a temperature coefficient) [Baker 2011]:

$$V_{th}(T) = V_{th0} + \alpha \cdot (T - T_0). \quad (2)$$

In sum, the datasheets give the safe operating voltage at a given frequency for a wide range of temperatures. As a consequence, there is a scope for energy saving if the operating voltage is adapted according to actual temperature values.

It might be argued that we inherently introduce a decreased reliability for the sensor nodes since undervolting may have an impact on the correct execution of a program

or the proper functioning of a component. Of course, any failure caused by lower than recommended voltage levels could have a significant impact on applications. Hence, the goal of this work is to run undervolted MCUs (nearly) as reliably as those supplied with the specified voltage levels.

Although unpredictable errors due to undervolting may still occur, if this malfunction risk is limited, we stay very well within the typical WSN assumptions. A single node is not of importance, but the whole WSN must function properly. Thus, WSNs should be designed to be fault tolerant because of, for example, rough operational areas, cheap hardware, or even passive undervolting when the voltage of discharging batteries drops below the recommended levels.

This implies that self-stabilizing algorithms [Ben-Othman et al. 2011] or self-repairing routing schemes [Winter et al. 2012] are already needed because of the underlying unreliable sensor network. Hence, in exchange for significantly increased energy efficiency, unexpected failures due to undervolting do not strongly harm most WSN applications.

So considering Anthes [2013], an absolute coverage of failures caused by undervolting is not needed by every application whenever occurring failures are tolerable. Nevertheless, although failures might be covered by the fault-tolerance mechanisms of WSNs, there might be other cases where a balance between the benefit of and the risk due to undervolting has to be set for each specific case. For instance, applications and system components like routing protocols spend a lot of energy to build network state. This state might be lost due to resets or other errors that could once again lead to increased energy consumption. Once more, it is important to mention that the goal of the presented approach is to avoid unstable states or incorrect execution of programs whenever possible. The fault-tolerance of WSN is just a backup mechanism to avoid unpredictable failures caused by undervolting being propagated to the application layer.

3. UNDERVOLTING CAPABLE SENSOR NODE

Using undervolting to reduce the energy consumption of processing units is not a new idea. Undervolting is unintentionally done at the end of the battery lifespan of every battery-powered node.

Indeed, ongoing research aims to integrate active undervolting at the chip level [Das et al. 2009], but this approach is far from being production ready. Hence, to demonstrate the feasibility and to transfer the presented concepts of reliable undervolting to a real wireless sensor node, there was the need to develop an undervolting-capable sensor node. Moreover, in assuming that undervolting on a chip level will be available on commercial off-the-shelf (COTS) processing units in upcoming years, this sensor node can be used to analyze the effects of such hardware devices in advance.

In Kulau et al. [2015], we presented a basic architecture to perform extensive evaluations of undervolting on WSNs. Here, this design was adapted and extended. As a result, this prototype is a full-fledged WSN with the capability of undervolting and a comprehensive software support of common WSN operating systems (Tiny OS [Levis et al. 2005], Contiki OS [Dunkels et al. 2004] and Arduino^{1,2}).

This section gives a detailed description of the nodes's architecture and points out the pitfalls when adding scalable voltage levels to a system.

3.1. Voltage Scaling Module

One prerequisite to achieving active undervolting on a WSN is the ability to adjust the voltage level by software. There are many fully integrated power-management

¹<http://www.arduino.cc>.

²[git://git.ibr.cs.tu-bs.de/project-cm-2012-inga-arduino.git](https://git.ibr.cs.tu-bs.de/project-cm-2012-inga-arduino.git).

Integrated Circuits (ICs) which are able to fulfil this requirement. However, these ICs are not suitable for small embedded devices like WSNs as the quiescent current of those parts is too high.

A more lightweight solution for a voltage scaling module that is very well suited for undervolting in WSNs is based on a digital potentiometer. The ohmic resistor of the potentiometer is controllable by software via the Inter-Integrated Circuit (I²C) bus. Common adjustable voltage regulators can be connected to the ohmic resistor to get a variable voltage level as a function of the register settings of the potentiometer [Hoermann et al. 2011; Kulau et al. 2013]. Furthermore, compared to existing power-management ICs, the usage of digital potentiometers allows for a more fine-grained shading of the voltage level. Thus, the individual absolute minimum voltage level (V_{abs}) can be detected with a much higher resolution (cf. Section 4).

A rather general question is what kind of voltage regulator should be used. With regard to the efficiency of voltage conversion, a step-down converter (buck) is more efficient than a linear dropout converter (LDO). On the other hand, a buck has higher quiescent current. In practice, as showed in Kulau et al. [2013], it depends on the duty cycle whether a buck or an LDO should be used. For this reason, we designed both voltage regulators on the layout but used an LDO for the first batch of prototypes.

A scalable voltage path is essential to provide undervolting on the processing unit. Additionally, a second independent and scalable voltage path for peripherals is available in our system. This allows us to adapt the peripherals' voltage to ensure that peripherals are always supplied with their recommended voltage levels. In addition, as shown in Friedrichs et al. [2014], an optimally scheduled peripherals voltage level increases energy efficiency significantly. Thus, this energy management technique can be used contemporaneously with undervolting the core components (processing and transceiver unit) of the sensor node.

All in all, we have three different voltage levels on the Printed Circuit Board (PCB). The core voltage V_{core} is connected to the MCU and the transceiver unit. This voltage path is used for undervolting, whereas the peripheral voltage V_{per} offers the supply for all sensing and memory components on the node. The external voltage V_{ext} supplies the voltage scaling module, which generates V_{core} as well as V_{per} .

3.2. Sensor Node's Architecture

In general, the architecture of the sensor node follows the classical design and can therefore be divided into three major parts: Processing unit, transceiver unit, and sensing unit. The ATmega1284p MCU forms the main processing unit, which is a commonly used MCU on various sensor nodes, such as micaZ [Crossbow 2012], INGA [Büsching et al. 2012] or Raven [Atmel 2012]. For wireless communication via IEEE 802.15.4, the processing unit is connected to an AT86RF233 radio transceiver. In contrast to standard architectures, this node is additionally equipped with an ATtiny84 MCU.

The ATmega1284p acts as the main processing unit, which is potentially undervolted and thus might not be reliable. Hence, a trustable reference is needed to observe and control the active undervolting. Using an MCU instead of static hardware devices offers the benefit of facilitating more sophisticated methods like supervised learning algorithms (cf. Section 6). This second co-MCU is called a Secure Instance (SI), which is an essential part of the undervolting-capable sensor node.

A block diagram of the whole architecture of this node is given in Figure 1. As mentioned earlier, the voltage levels are partitioned in three areas, V_{ext} , V_{core} , and V_{per} . These areas are not fully isolated from each other because the components are interconnected through buses (I²C, Serial Peripheral Interface Bus (SPI), Universal Asynchronous Receiver/Transmitter (UART)) and other control lines. Thus, a problem occurs whenever the voltage paths have different voltage levels $V_{ext} \neq V_{core} \neq V_{per}$.

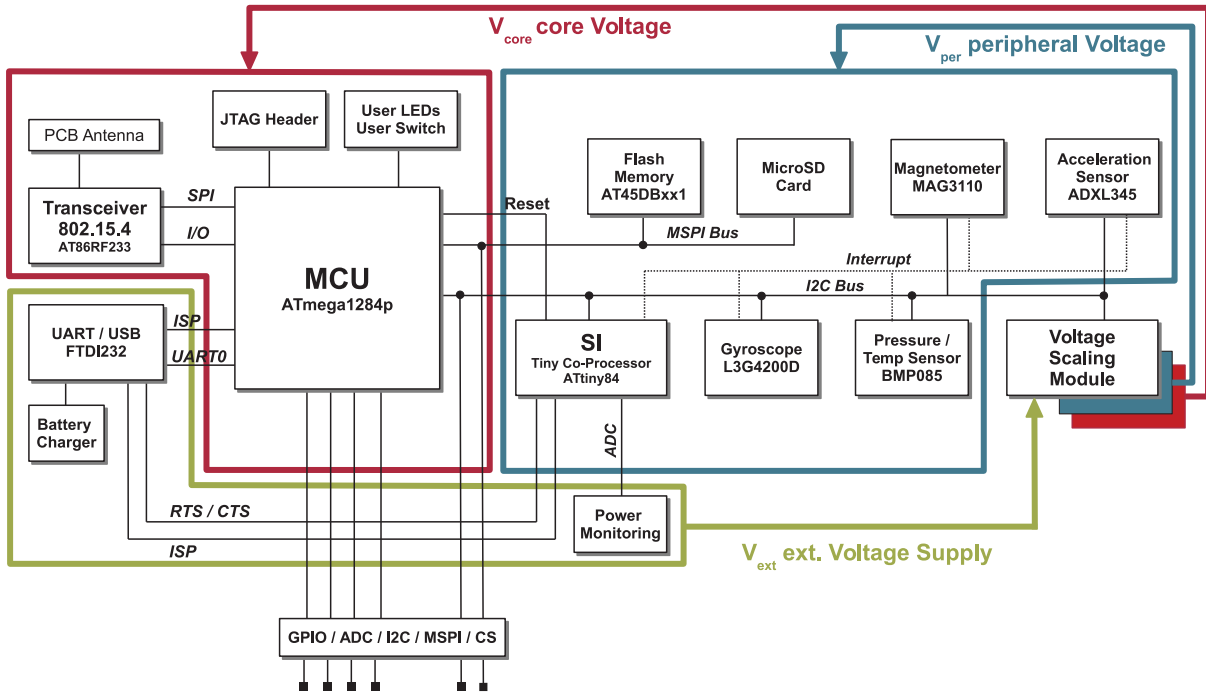


Fig. 1. Block diagram of the the prototype.

The potential difference between interface and supply voltages could lead to damage on several components of the sensor node. For this reason, we integrated voltage-level translators (Fairchild FXMA2104) at every point of contact between the different voltage paths. The additional overhead of one active level translator is $<5\mu A$ each. Compared to the sleep current of common MCU or transceiver units, this additional current consumption has a negative impact on the lifetime of WSN applications with very low duty cycles. However, this first undervolting-capable WSN is chiefly used to analyze the effects of lower than recommended voltage levels. Furthermore, the level translators are needed to design such a node even with current COTS components. Of course, for further generations of undervolting-capable sensor nodes, these level translators could be redundant, assuming that core voltage levels of ICs will be independent from the I/O-levels.

In addition to the requirements to implement controlled undervolting, the sensor node is equipped with the following features:

- Two external memory devices:* The flash memory (Atmel AT45DB321) has a capacity of 32MBit and the advantage that data can be stored very fast. An integrated dual buffer avoids idle times while a page is written to the flash. Alternatively, a microSD card can be used to store larger amounts of data.
- 9 degrees of freedom sensing:* To be able to use the node for motion sensing in ongoing research projects, such as for fall detection in elderly health care or as a flight control unit for quadcopters, a 3-axis accelerometer (Analog Devices ADXL345), a 3-axis gyroscope (ST Microelectronics L3G4200D), and a 3-axis magnetometer (Freescale Xtrinsic MAG3110) are connected to the I²C bus.
- Concurrent power monitoring:* The measurement of the current consumption and external voltage level is part of the SI. Thus, a concurrent observation of the power dissipation is possible.

As mentioned earlier, the SI is realized through the ATiny84 MCU. Thus, the sensor node is equipped with two MCUs, - leading to the rather practical issue of how to



Fig. 2. Prototype of the undervolting-capable wireless sensor node with two independent and scalable voltage paths.

Table I. Additional Overhead to Enable a Controllable Undervolting

Device	Constant Overhead	Typical Overhead	
		Active [μA]	Sleep [μA]
SI		400	0.1
Voltage Scaling Module	✓	3.6	–
I ² C Level Translator	✓	5	–

program the devices. The usage of an external programmer would be the easiest way. Nevertheless, the connection between the programmer and the respective MCU would have to be plugged in manually. Hence, this solution would not be sufficient for a smooth progress of work for just programming a sensor node. For this reason, the programming interfaces of both MCUs are connected to the FTDI232 USB bridge (cf. Figure 1). The FTDI232 offers a bit-banging mode which makes it possible to emulate the programming protocol of the MCUs. To avoid collisions, the programming interfaces of the MCUs are isolated via tri-state buffers that are controlled by the FTDI232 as well.

Prototype Implementation. According to the presented architecture, the schematics and a PCB layout were designed. Afterward, ten prototypes were produced to enable this work. The complexity of the sensor node forces a four-layered PCB, but the manufacturing costs are comparable to an ordinary two-layered board [Büsching et al. 2012]. The presented hardware design, as well as the extensive software support (Contiki OS, Tiny OS, Arduino), is completely under open source licences.³ A picture of the node can be seen in Figure 2.

3.3. Additional Overhead

In order to allow for further mechanisms (e.g., to control undervolted nodes, achieve energy efficient states and guarantee reliability), the trustable SI is inevitable. However, the integration of an additional component entails an additional overhead, as already discussed, in connection with the need of level translators. Table I summarizes the needed components and their typical overhead. Other peripherals, such as sensors, transceivers, or memories, are not considered because their composition is independent from further course of action.

It can be seen that a constant overhead of $\approx 9\mu A$ is relatively high compared to the sleep modes of current MCUs. Thus, especially for applications with very low duty cycles, this overhead could have a negative impact on the overall energy consumption of a node. In Section 8.6, this issue is taken up and discussed in detail. It should be

³<http://www.ibr.cs.tu-bs.de/projects/inga/>.

emphasized that the presented undervolting capable node is an experimental platform rather than a fully optimized sensor node. In our architecture, we rely on COTS parts only; thus, some overhead (e.g., due to the level translator) is (sadly) inevitable. The usage of an Application-Specific Integrated Circuit (ASIC) could reduce the constant overhead significantly, but its design is out of the scope of this work. Nevertheless, compared to the constant overhead, the SI has much higher power dissipation during its active operation. So, even for this experimental platform, this extra effort prunes the huge potential of undervolting. However, the introduction of additional hardware might be permitted with regard to expected significant energy savings of up to 42% [Kulau et al. 2015]. In the following, an estimation of the expected gain of undervolting including the additional overhead is derived.

Actually, what matters most is the duty cycle of the SI because the current consumption of inactive MCUs is almost negligible. Assuming that the SI executes a periodic process to observe the accurate behavior of the main MCU, it must not stay in active mode all the time. Furthermore, a lower computational performance of the SI results in a low power dissipation during active times. Hence, the software implementation of the SI is driven by the challenge of energy-efficient programming. With reference to the prototype implementation in Section 3.2, an Atmel ATtiny84 is used as SI, which has a current consumption of $I_{SI_{active}} \approx 400\mu\text{A}$ at $f_{cpu} = 1\text{MHz}$. Yet, in sleep state, only $I_{SI_{sleep}} \approx 100\mu\text{A}$ are necessary. Based on these data and together with the applied voltage level V_{per} , the energy consumption of the SI is given as a function of the duty cycle λ (t time):

$$E_{SI}(\lambda) = V_{per} (\lambda \cdot t \cdot I_{SI_{active}} + (1 - \lambda) \cdot t \cdot I_{SI_{sleep}}). \quad (3)$$

Furthermore, the constant overhead E_{co} has to be considered; this is based on the additional current consumption of the voltage scaling module, the I²C level translator (cf. Table I).

The energy-saving effect due to undervolting is related to the main MCU. As given by Equation (1) the dynamic power dissipation of the CMOS-based MCU shows a quadratic dependency on the voltage level and a linear dependency on the clock rate. Thus, the yield of undervolting is mainly part of the active power dissipation, so that a duty cycle is not considered at this point. Analogous to Equation (3) the active energy consumption of the main MCU is given by:

$$E_{cpu}(V_{core}, f_{cpu}) = V_{core} \cdot t \cdot I_{MCU_{active}}(V_{core}, f_{cpu}). \quad (4)$$

According to the specified minimal voltage levels at a given clock rate $V(f_{cpu})$, the energy consumption of the main MCU is given by $E_{cpu}(V(f_{cpu}), f_{cpu})$. When using undervolting, the nominal voltage level is wilfully set below the specifications ($V(f_{cpu}) - \Delta v$). Hence, the energy consumption of the main MCU while using undervolting is given by $E_{cpu}(V(f_{cpu}) - \Delta v, f_{cpu})$. Considering the additional overhead of the SI $E_{SI}(\lambda)$, the expected energy saving can be calculated as follows:

$$\gamma(f_{cpu}, \Delta v, \lambda) = \frac{E_{cpu}(V(f_{cpu}), f_{cpu})}{E_{cpu}(V(f_{cpu}) - \Delta v, f_{cpu}) + E_{SI}(\lambda) + E_{co}}. \quad (5)$$

In general, information about the behavior of undervolted MCUs is not available in the datasheets. Nevertheless, numeric results about the impact of undervolting on the energy efficiency of the ATmega1284 processing unit are available from our preliminary studies in Kulau et al. [2015], and we can derive $E_{cpu}(V(f_{cpu}) - \Delta v, f_{cpu})$. Thus, together with the data provided by the datasheets of all involved components and Equation (5), it is possible to rate the expected gain of the whole system.

The results for different clock rates f_{cpu} are given in Figures 3(a)–(c). It should be mentioned that for the used type of MCU, the deviation from $V(f_{cpu})$ (i.e., where

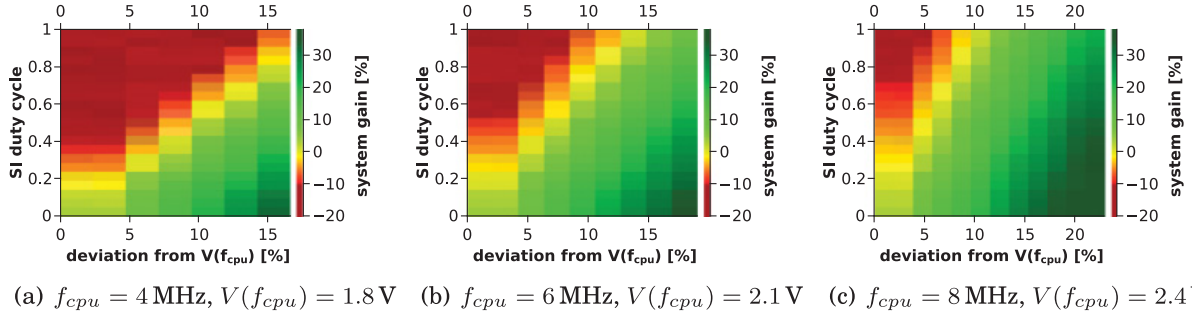


Fig. 3. Expected energy savings γ for the overall system including SI. The duty cycle of the SI (ATtiny84) may prune the yield of only slightly undervolted MCUs. However, in general, the overhead of the SI is easily outweighed.

no errors occur) grows with higher clock rates. Typical deviations from the nominal voltage level are between $12\% \leq V_{f_{cpu}} \leq 24\%$ [Kulau et al. 2015]. Hence, a large part of the overall system gain depends on the duty cycle of the SI. Although the expected energy savings of the undervolted MCU give ample scope to legitimate using the SI, the additional overhead could also imply negative effects on power consumption if the SI duty cycle is increased above a certain level.

In other words, if the MCU is checked by the SI less often, this leads to a higher gain related to energy efficiency. Contrariwise, the possibility of detecting failures drops with a lower check rate. Therefore, in the first instance, we need to trade reliability for energy efficiency. In the long term, an adaptation of the check rate according to the actual context is desirable. This is another task for the supervised learning process (cf. Section 6).

4. AUTOMATED DETECTION OF IDEAL VOLTAGE LEVELS

Due to tolerances in the manufacturing process, the voltage level where the operation of a micro controller becomes unreliable is different for every single MCU, even of the same type [Kulau et al. 2015]. Therefore, we need an adaptive approach that considers the individual properties of every single MCU. This is emphasized by the limited available information about the coherence between the underlying CMOS gates and their temperature dependencies. A straightforward approach would be based on extensive evaluations using many wireless sensor nodes. The results could be used to define look-up tables with operational voltage-level parameters for certain environmental conditions. In this way, one or several safe operating areas for undervolting could be defined. However, such an approach would not only come with a large overhead due to the necessary extensive evaluation, it would also be neither optimal nor reliable.

First, energy savings would still be less than the achievable optimum unless every single instance is tested under all operational conditions. If some sensor nodes of the WSN show a higher tolerance for undervolting than the tested nodes and would be able to run below the defined minimum voltage level, the potential of energy conservation is not exhausted.

Second, the same arguments apply for unreliability: If a node is less tolerant than the tested nodes, this will lead to an unreliable system. Furthermore, reliable handling of failures would not be possible. Even if the self-defined safe operating area is designed conservatively, random errors can always occur when using undervolting, especially when not all environmental conditions (such as temperature ranges) can be studied beforehand. For example, suppose that the calculation of the program counter goes wrong; the MCU could be resetting constantly and would not be able to increase its own voltage level again.

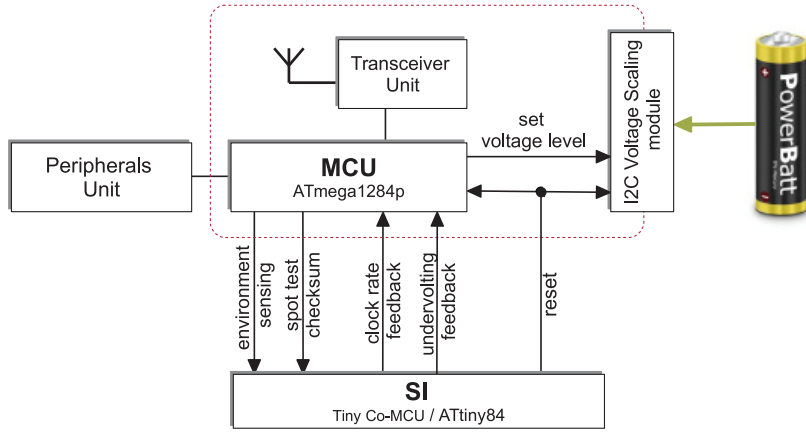


Fig. 4. Co-processor-based system architecture for closed-loop control.

Thus, proper failure handling mechanisms should be provided to avoid such deadlock situations. Moreover, an independent estimation of errors caused by undervolting is necessary to decide whether an MCU has reached its absolute minimum voltage level V_{abs} or not.

4.1. Closed-Loop Control Approach

A simple solution for this challenge is the implementation of a closed-loop control to find ideal voltage levels by successive approximation. The basic idea is to lower the voltage level step by step and to reveal the first appearance of transient malfunction in the MCU. Once a failure caused by undervolting is detected, the absolute minimum voltage level V_{abs} is reached. Immediately, the voltage level is increased slightly to run the MCU at the most energy-efficient but stable state (now referred to as the *ideal voltage level*, V_{opt}). The advantage of a control loop is that a unique sensor node can achieve a balanced state between faultless execution and optimum power dissipation. Moreover, this adaptation takes place regardless of external temperature conditions or individual characteristics that may influence the ideal voltage level, being taken care of by the control loop.

The reliable SI can be used to realize such closed-loop control. Figure 4 shows a block diagram of this approach.

4.1.1. MCU Spot Test. First, we must decide how the MCU can be checked to see whether the current voltage state is ideal or not. The basic idea is that the MCU performs certain periodic test calculations. The SI inspects the validity of the MCU results. A periodic full test of every component of the MCU is almost impossible. For example, the test of only one ALU operation (e.g., add) with $n = 2$ input variables of $m = 8\text{Bit}$ each would need a test vector of $(2^m)^n = 2^{16}$ cases. Another possibility would be an *IEC60730 Class B* test, which is intended to prevent unsafe operation of MCUs [Atmel 2008]. However, with the intention of further undervolting as a regular power management technique and the paradigm of energy-efficient programming (cf. Section 3.3), the execution time of such a Class B test bears no relation to the expected gain. On the one hand, the Class B test is very fine-grained and would check a major part of the MCU, which leads to high detection rate of possible failures. On the other hand, due to long test execution time, the actual application might not be served, which would affect the application's process flow. For this reason, a simplified spot test strikes a balance between reliability and the overhead of fault detection. The online detection of errors plays a significant role in several application areas (e.g., aerospace applications). In Rohani and Zarandi [2009], an adequate method to detect

Single Event Upsets (SEUs) on AVR core-based microprocessors is presented. However, failures caused by ion radiation (SEU) or due to insufficient voltage levels are almost comparable. The system state of a transistor is changed in both cases, which leads to a bit-flip. It turns out that the arithmetic logical unit (ALU) is the most critical point where fault injections are manifested. With the assumption that undervolting will inject transient failures into the AVR core, too, the spot test is mainly looking at the ALU.

Moreover, Rohani and Zarandi [2009] examined different algorithms with regard to the detectability and occurrence of errors. It was shown that a simple matrix calculation can be used to manifest about 86% of ALU errors. Based on these results, we designed our spot test as a determinant calculation of a matrix multiplication:

$$\text{checksum} = \det(A \cdot B) \text{ with } A, B \in \mathbb{R}^{n \times n}. \quad (6)$$

In Kulau et al. [2015], we used a variant of this test and showed that errors caused by undervolting can be detected sufficiently. We therefore omit details in this article and refer the reader to Kulau et al. [2015]. For this purpose, the calculation of the checksum is executed on the device under test, which is the undervolted MCU.

Since the MCU might be unreliable due to undervolting, the validation of the calculated checksum must be done by the SI.

4.1.2. Secure Instance. The SI is realized through a second tiny microcontroller. In contrast to the main processing unit, this MCU is always supplied with an adequate voltage level to guarantee a trusted base for processing the test scores of the main MCU. However, other parts also can be used if they are able to generate a binary feedback of the test scores or the calculated checksum of the main MCU (e.g., shift registers). The advantage of using a second MCU is that the design space of a software-based SI is much higher.

For the basic implementation of the closed-loop control approach, a very limited computational performance is required. Mainly, the SI has to countercheck if any failure occurs. The tasks of the SI vary from (i) forcing the main MCU to adapt the clock rate and optimize the voltage level, up to (ii) resetting the whole sensor node if the MCU hangs completely (as described in Section 4.2)

However, how the SI optimizes undervolting can be more complex. The supervised learning process described in Section 6 is also executed on the SI.

With regard to already existing fault tolerance concepts for distributed systems, an SI is also helpful. For example, the implementation of resource-efficient byzantine fault tolerance is possible because the SI can be used as the trusted computing base [Kapitza et al. 2012].

4.2. Iterative Self-Optimization: Protocol

At no time would it be predictable which part of the ALU is infected by undervolting and which consequential errors might be caused. For example, a wrongly calculated Program Counter (PC) could also lead to software resets or, in the worst case, to a hanging MCU (deadlock). For this reason, a protocol to find ideal voltage levels individually has to guarantee both adequate failure handling as well a trouble-free application of the sensor node itself. In general, iterative self-optimization is based on a very simple periodic request-response procedure. The undervolted MCU initiates a request to obtain information on how to adjust the voltage level and clock rate. For this purpose, the MCU sends information about its current status to the SI. Subsequently, the SI sends a response frame back to the MCU to advise how the MCU should update the voltage level and clock rate adaptation for the next test cycle.

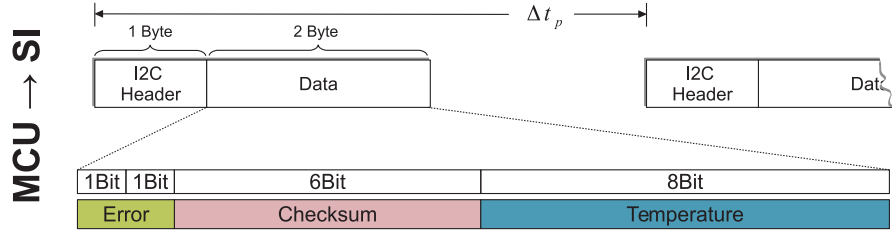


Fig. 5. Request Frame. Information given from the undervolted MCU to the SI.

Table II. Possible WSN Failure States of Sensor Node within the Periodic Data Frame

State		Description
0	0	No failure
1	0	WSN heartbeat timeout
0	1	Remarkable retransmission rate
1	1	Reserved

Although the MCU is the unreliable part, it initiates the request. However, due to the MCU's periodic request, the SI is able to detect the worst case of a deadlocked MCU as well. An indicator for such an error is the absence of a request frame.

In the following, a more detailed description of self-optimization based on the request-response approach is given.

Figure 5 illustrates the *request frame* that is sent from the undervolted MCU to the SI. The request data frame contains information about the condition of wireless communication. According to Kulau et al. [2015], depending on the selected clock rate, undervolting may additionally infect the RF-section of a sensor node. Hence, we need information on whether a failure on the wireless connection of the node occurred. Therefore, the data frame has a 2-bit wide field to indicate *error* states. The definition of these states is given in Table II.

However, an SI does not have to react promptly on every WSN failure notification. The WSN heartbeat timeout or an increasing retransmission rate could be caused by other external events as well (e.g., interference, moving nodes, etc.). At this point, it should be mentioned that WSN failure states are not implemented in this work. Undervolting the transceiver unit would go beyond the scope of this work. Thus, whenever we use the transceiver unit, we take care that the core voltage level is always above the minimum recommended voltage level $V = 1.8V$.

The ALU is the first component of the MCU that may suffer from a lowered voltage level. To detect calculation errors, we use the test method described in Section 4.1.1 (matrix multiplication). The *checksum* contains the result of this test performed on the MCU and sent to the SI. The SI compares the value of the checksum with its own calculation results and checks whether it is valid. In this case, the checksum is 6 bits wide so that the result of the calculation has to be $-2^5 \leq \text{checksum} \leq 2^5 - 1$. According to preliminary studies [Kulau et al. 2015], this number range together with the presented test method is sufficient to detect ALU failures caused by undervolting.

The *temperature* value in the second byte of the data frame is at this point the first optional information sent to the SI. For the concept of closed-loop control (cf. Figure 4), information on whether a failure occurs is entirely sufficient. Failures provoked through environmental changes (e.g., drop of temperature) are covered anyway, because the SI has no knowledge of the circumstance that caused the failure. Nevertheless, information about temperature can help to avoid failures in the future. The correlation between temperature and the appearance of failures can be used by the SI to derive a heuristic, which is part of the supervised learning implementation (cf. Section 6).

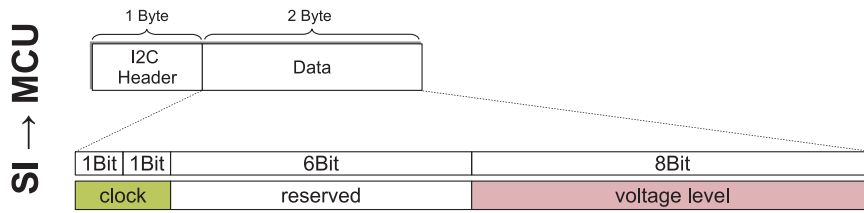


Fig. 6. Response data frame from the SI to the MCU.

Because the interconnection of our approach is realized through the I²C bus, the data frame comes with a communication overhead of only 1 byte. In the first instance, the data frame itself needs only 2 bytes, but later it will be extended for the implementation of supervised learning. The frame should always be fairly short to minimize its influence on the timings of the actual application. Furthermore, a larger data frame would force the SI to stay active over a longer period of time due to a prolonged data transfer. However, in high speed mode ($SCL = 400\text{kHz}$) the transmission of a byte lasts only $\approx 20\mu\text{s}$.

A more indirect but important piece of information is the time between two data frames Δt . In Kulau et al. [2015], it was mentioned that the voltage level influences the clock rate of the MCU if an internal RC-oscillator is used for clock generation. Provided that the main clock of the MCU is responsible for triggering the periodic data frame, Δt gives indirect information about the actual clock rate f_{cpu} . Hence, the SI can send binary feedback if f_{cpu} has to be increased, decreased, or held constant. Of course, the SI can generate a more complex response to accelerate recalibration to the desired clock rate. However, as evaluated in Kulau et al. [2015], a simple feedback-based calibration is sufficient.

Additionally, Δt indicates whether the MCU is still working. We observed that a failure of the ALU test does not imply that the MCU fails completely. After the reception of false checksum values, the SI advises the MCU to increase its voltage level to prevent further failures. However, a wrong ALU calculation may also infect the PC result. It could happen that the PC jumps to an unused memory address and, as a consequence, the MCU hangs. In that case, an internal MCU watchdog would not be a solution because a trigger by the watchdog would only force a reset without resetting the voltage level. This would be the worst case because the sensor node is not able to return to a safe operating state. With our system design, the SI is able to solve this issue in the following manner.

Whenever $\Delta t > t_{timeout}$, the SI recognizes that the MCU is no longer available. The SI acts like an external watchdog and resets the sensor node in three steps:

- (1) The SI pulls the reset line of the MCU to ensure that the MCU does not effect the I²C bus (Tri-State).
- (2) The SI becomes an I²C master device to control the voltage scaling module. The voltage level is set back to $V(f_{cpu})$.
- (3) The SI becomes a I²C slave device again and releases the reset line of the MCU.

Thus, even if the worst case of undervolting happens, this does not lead to a complete outage. A node is just not available for $\sim t_{timeout}$. For the other nodes in the WSN, this temporary breakdown of a sensor node equals a suddenly disappearing and reappearing node. Under the assumption that WSNs are able to handle structural changes in the network per se, this does not lead to any significant problems. Algorithms [Suomela 2013], MAC protocols [Lee et al. 2006], and routing algorithms [Winter et al. 2012] for WSNs are designed to be fault tolerant.

Table III. Binary Feedback to the MCU: How to Modify the Clock Rate Calibration

State		Description
0	0	No recalibration
1	0	Increase clock rate
0	1	Decrease clock rate
1	1	Reset calibration register to default value

In addition to the connection between the SI and the MCU, the I²C bus may be shared with other peripheral components like sensors, actuators, or memories. Hence, the correct functionality of this bus is required. Failures may lead to lost or modified data transfers while interacting with the peripherals. Another effect when using the I²C bus as the interconnection to the SI is that a malfunction of this interface through undervolting might influence the checksum result of the data frame. However, in the first instance, we are only interested *if* a failure occurs. Whether it was a wrong ALU calculation or an I²C bus error is not important. Indeed, over time, it will be possible to observe different sources of errors, and it would be wasteful if this information is not used for further optimization. Yet, the SI is able to decide how to adapt the voltage level and clock rate. If any failure occurs, the voltage level has to be increased immediately. Otherwise, the voltage level can be decreased gradually or hold constant depending on the further usage of the proposed protocol. The SI propagates this information to the MCU by a response frame as follows:

The first two bits contain the binary feedback for the recalibration of the clock rate (*clock rate adjustment*). Table III describes the 2-bit feedback to the MCU.

Like many other MCUs, the ATmega1284p MCU offers a register to calibrate the internal RC-oscillator (called the OSCCAL register). For this implementation, the OSCCAL register is fed by binary feedback to mitigate the impact of undervolting on the clock rate f_{cpu} . It would be possible to improve the recovery of f_{cpu} further by giving more detailed information. However, for that, the response data frame would have to be spread to more bits. For example, we could send the entire value for a calibration register back to the MCU; this is part of the supervised learning process. Nevertheless, in this case, a linear recalibration is sufficient.

The second byte of the response data frame indicates the change of the actual voltage level. With regard to the later implementation, we assume that the voltage level is defined by an 8-bit register. A too-fast drop of the voltage level could have a negative influence on the clock rate and could cause errors. Thus, it is recommended to lower the voltage level only stepwise. After reception of the voltage level adaptation, the MCU takes the new register value and forwards it to the I²C voltage scaling module.

Why does the MCU and not the SI control the voltage scaling module? All relevant parts are connected to the I²C bus, so, in general and as described above, the SI would be able to access the voltage level. The major reason is that the SI is nothing more than a reliable oracle. Therefore, the SI should be passive and only intervene when the sensor node is in a deadlock. To control the voltage scaling module, the SI has to become an I²C master. Thus, the I²C bus is shared by two master devices. This means that the bus lead has to be arbitrated. In the worst case, the SI would never get the permission to write on the bus, or it could block overlaying applications from using the I²C bus. For this reason, the MCU is responsible for all master I²C transactions except for deadlocks (as explained earlier).

4.3. Finding Ideal Voltage Levels: Functionality Tests

It is essential to know whether the proposed automated finding of ideal voltage levels V_{opt} is a firm base for further implementation. Thus, before deriving more sophisticated

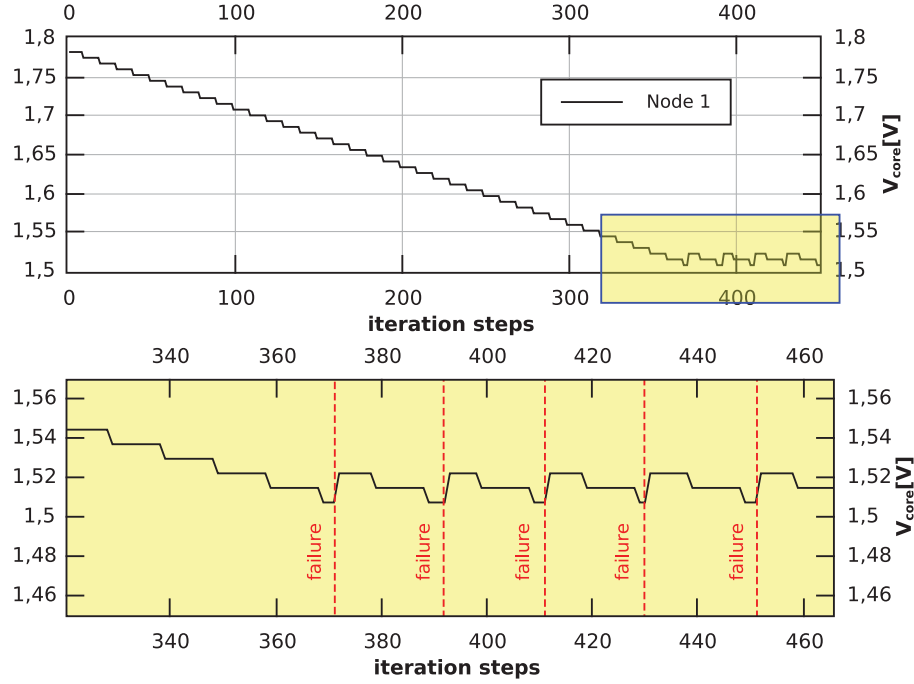


Fig. 7. Iterative self-optimized undervolting, general functionality. At the dotted vertical lines, an error is detected.

algorithms to achieve energy efficiency, reliability, and robustness, the functionality of iterative self-optimization has to be tested. Moreover, with regard to the lifetime of sensor nodes, it would be useful to know how undervolting performs in a real WSN to estimated the expected gain.

For this interim evaluation, the undervolting-capable sensor nodes presented in Section 3 were used. At this point, all experiments were performed in a non-varying laboratory condition at $T_{lab} \approx 22^\circ C$. However, because temperature has a strong impact on the threshold voltage of CMOS gates, the general reliability of finding ideal voltage levels at different temperatures is considered as well.

The nodes were all running at 4MHz (nominal minimum voltage 1.8V) because this is the most interesting and challenging use case for the specific MCU [Kulau et al. 2013]. For long-term evaluation, the nodes were set to the MCU's default clock rate of 8MHz (nominal minimum voltage $V(f_{cpu} = 8MHz) = 2.4V$).

The general functionality of self-optimization can be seen in Figure 7. For these measurements, the voltage level has been decreased gradually. Starting at $\approx 1.78V$, the algorithm lowers the voltage every 10 iteration steps; thus, in this case, every tenth second. The lower part of Figure 7 shows a magnification of the lower right part of the upper curve. Any time an error occurs, the absolute minimum voltage level V_{abs} is reached. Subsequently, the voltage is raised by two intervals and then lowered again to show the repeatability of finding ideal voltage levels. Obviously, the cyclic algorithm of readapting the voltage level continuously is not ideal because unnecessary failures are provoked. However, the evaluation shows the general functionality of self-optimizing undervolting and the reliable detection of failure states using this approach.

4.3.1. Iterative Self-Optimization. We tested the iterative self-optimization with four different prototypes. Figure 8 shows that they all behave similarly, but, due to individual differences, the ideal voltage level is different for every node. For the four tested nodes at $T_{lab} \approx 22^\circ C$, the lowest voltage at which accurate functionality is achieved is

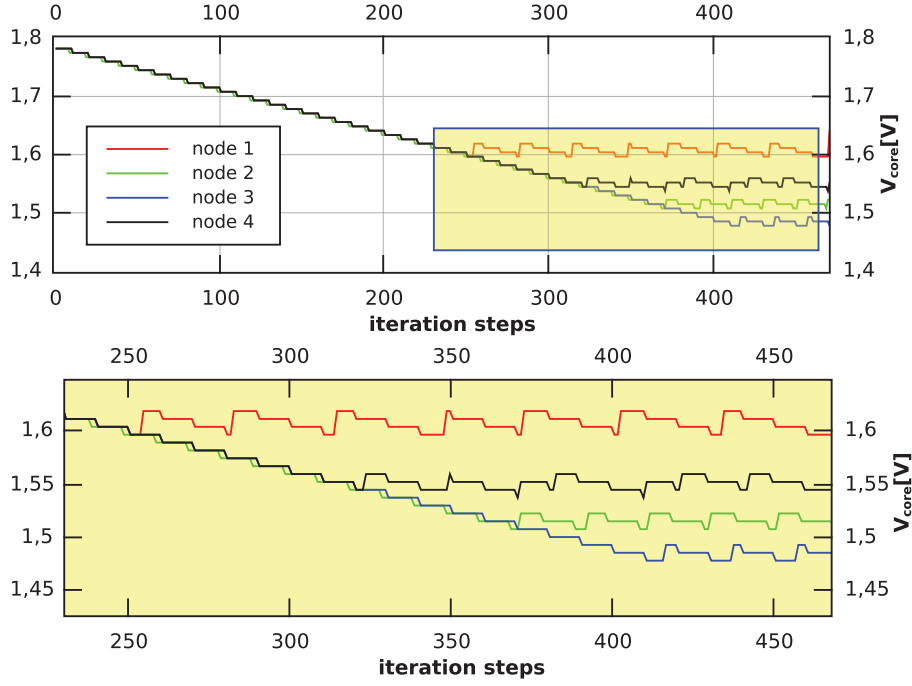
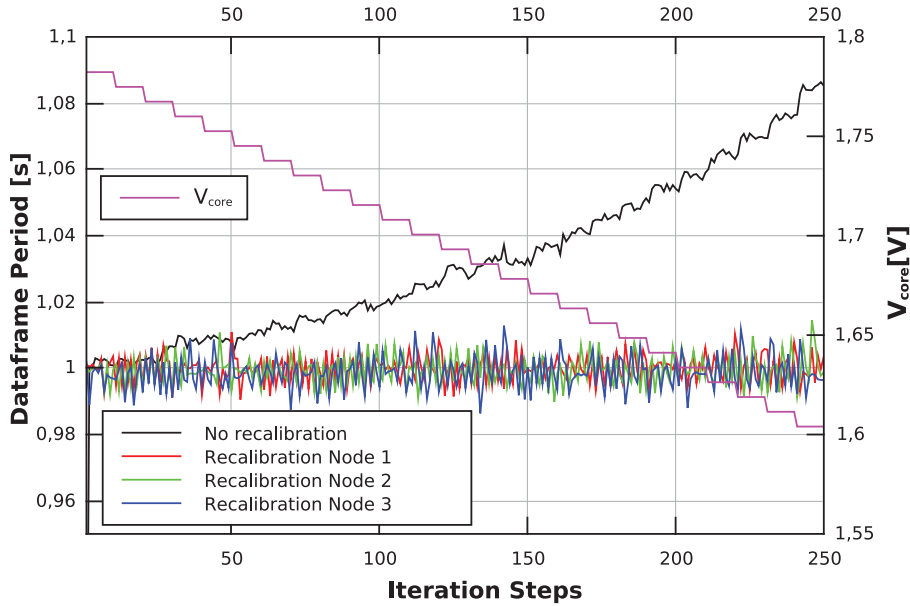


Fig. 8. Iterative self-optimized undervolting of four different nodes.


 Fig. 9. A continuous recalibration of f_{cpu} ensures a stable clock rate.

between $V_{min} = 1.48V$ and $V_{max} = 1.60V$, with a mean of $V_{mean} = 1.53V$ and an average standard deviation $\bar{\sigma} = 0.037V$.

4.3.2. Clock Rate Stabilization. The internal oscillator of an MCU is voltage dependent, as mentioned in Section 4.2 and in Kulau et al. [2015]. Figure 9 shows that without a recalibration (black line), the period of a data frame would increase during decreasing system voltage. Because this would lead to evident errors (e.g., at UART transmissions), the clock rate has to be recalibrated. The functionality of the protocol-based recalibration of three nodes can also be seen in Figure 9; the accuracy of the calibrated

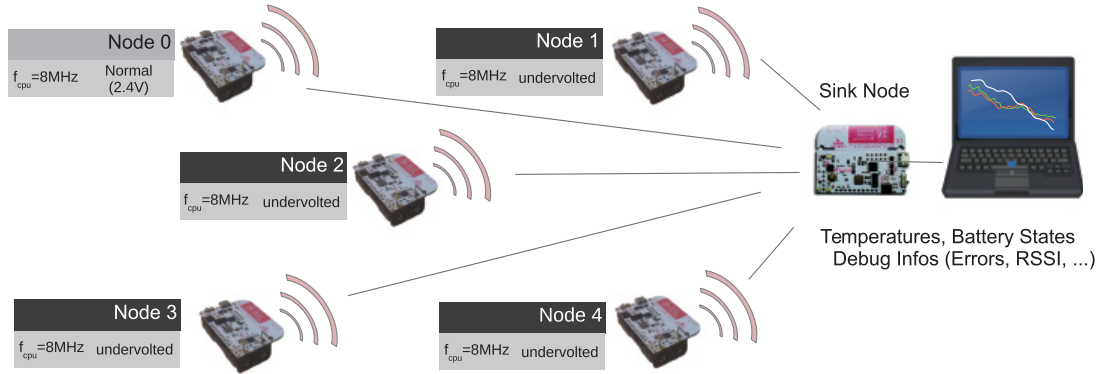


Fig. 10. Long-term evaluation with four undervolted nodes.

frequency depends on the actual voltage level and individual effects. The accuracy of our implementation is higher than 98.5% for every node.

4.3.3. Increased Energy Efficiency – Long-Term Evaluation. The goal of this evaluation is to show the huge energy saving potential of undervolting. Moreover, it should show that an undervolted system does not only work in short-term laboratory settings.

We set up a simple scenario consisting of five nodes: four undervolted nodes and one normal node without undervolting for comparison purposes. All nodes were equipped with two AAA batteries each and placed in an office to measure the environmental temperature. All nodes sent their data to a sink node via an IEEE 802.15.4 radio link.

On four nodes, self-optimizing undervolting was used to discover an ideal voltage level. After determining this operating point, the voltage level would only have to readapt if and when a failure occurs. However, reliable operation is continuously controlled by the SI. Therefore, the SI verifies the checksum and recalibrates the clock rate with a static period of $t_{interval} = 1s$ as described in Section 4.2. As mentioned, for this evaluation, all nodes were set to the MCU’s default clock rate of $f_{cpu} = 8MHz$. The normal node was set to the minimum recommended voltage level of $V(f_{cpu} = 8MHz) = 2.4V$ [Atmel 2010]. Hence, we compared our undervolting approach with the hitherto ideal operation point.

Figure 10 shows the setup of this evaluation. An ADC of the SI was used to sample and send the battery voltage levels together with the payload data of the temperature measurement application. The resulting discharge curves of the batteries are shown in Figure 11. We declared the lifespan of a node to last from the first to the last received packet. In case of the “normal” node, the last voltage values below the recommended 2.4V were also counted in the lifetime. The undervolted nodes settled at a voltage level of $1.834V \leq V \leq 1.891V$, which corresponds to a deviation from the nominal voltage of $21.2\% \leq \Delta V \leq 23.58\%$.

The results show that undervolting increases the lifetime up to 42.27%, which is comparable to the short tests done in preliminary studies [Kulau et al. 2015]. A more detailed view of the lifetime improvements is given in Table IV.

A previous mentioned advantage of undervolting is that it does not influence the further usage of the MCU. For this reason, it is essential to know whether undervolting has a negative impact on the overlaying application. In this case, the exemplary application was measuring the temperature of an office. Hence, temperature was sensed and continuously compensated for by using calibration coefficients and a predefined algorithm of the temperature sensor [Bosch-Sensortec 2009] itself. Figure 12 shows that undervolting does not influence the measurement in respect to the correct execution of the compensation algorithm. Compared to the normal node, the highest deviation is

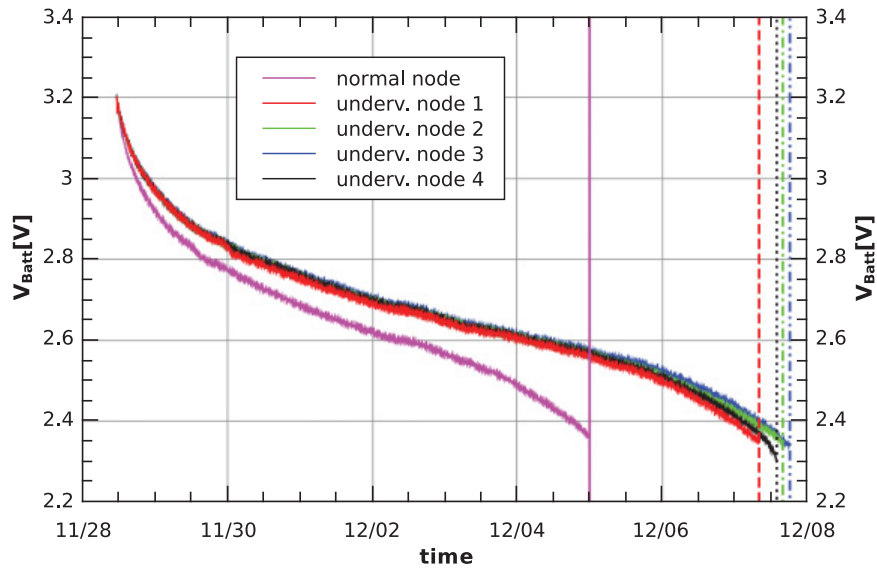


Fig. 11. Battery discharge curves of the long-term evaluation. The usage of self-optimizing undervolting prolongs the lifetime of the nodes significantly.

Table IV. Prolongation of the Lifetime Compared to a DVS-Capable Sensor Node

Node	1	2	3	4
Improvement [%]	35.76	40.89	42.27	39.61

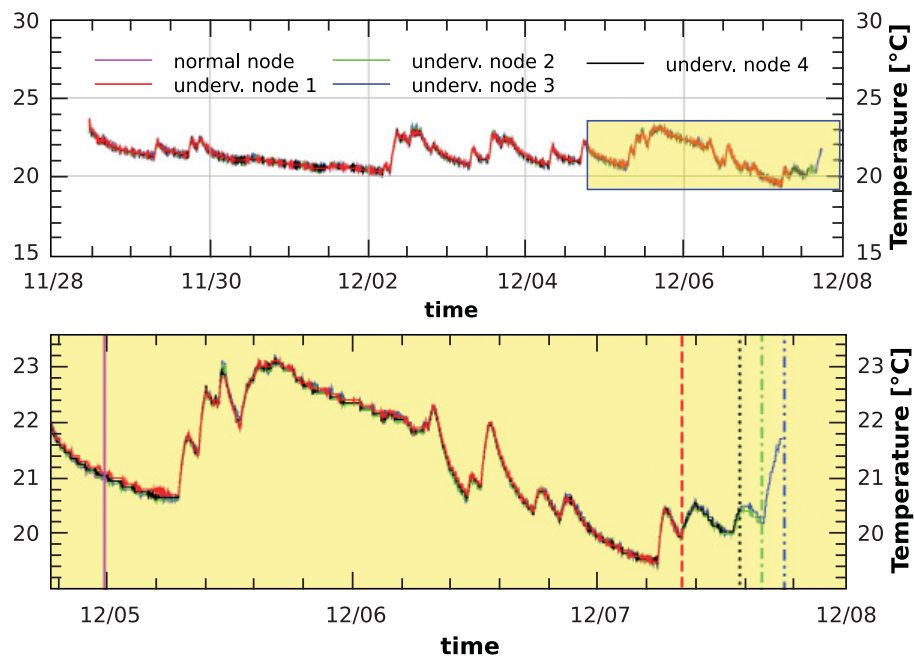


Fig. 12. Aggregated temperature values at the sink node of the long-term evaluation. No significant impact is seen on the overlying application using the undervolting approach.

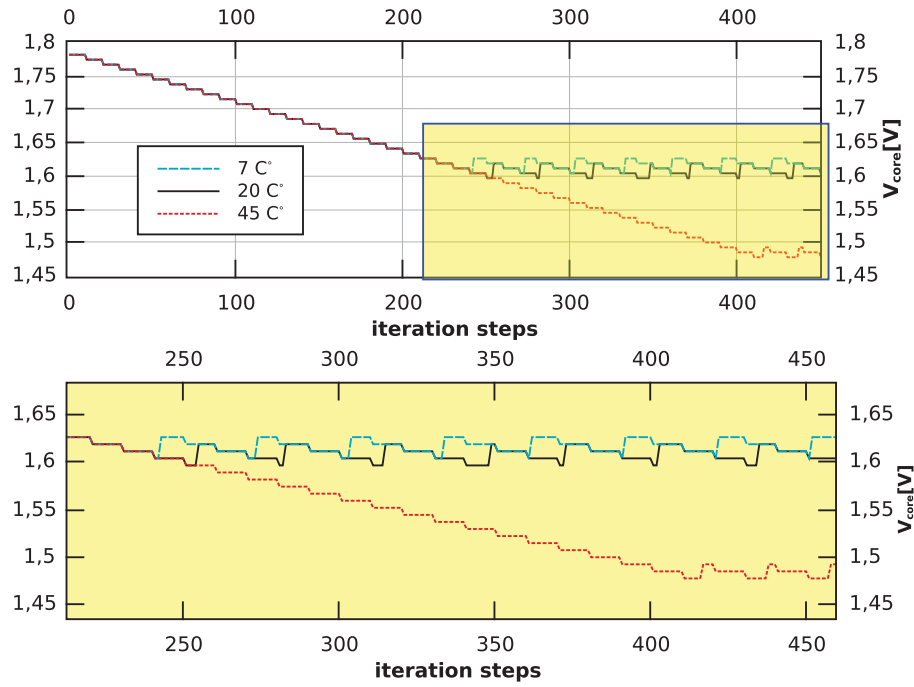


Fig. 13. The ideal voltage level shows a significant temperature dependency. Note: Different environmental conditions do not have any impact on the reliable finding of ideal voltage levels.

less than 2%, which is not attributable to undervolting because the declared accuracy of the sensors is already $\pm 3\%$.

During the whole evaluation, no abnormal behavior of the Received Signal Strength Indicator (RSSI) or other restraints of link quality was observed. Of course, in this case, a minimum voltage level of 1.834V is higher than the specified minimum supply voltage of the transceiver. Thus, potential communication failures would only have been attributed to bus errors.

4.3.4. Influence of Temperature. Up to now, the presented approach deals with almost static environmental conditions. However, in Section 2, the use of undervolting was motivated by the strong connection between the threshold voltage of CMOS parts and the external temperature. From this, it follows that, in addition to individual differences between the sensor node's hardware, the ideal voltage level shows a significant temperature dependency, as shown in our preliminary studies [Kulau et al. 2015].

So far, it is known that a higher temperature leads to a lower ideal voltage level and vice versa. Thus, active undervolting can compensate the effect that the current consumption of MCUs grow with higher temperatures. Moreover, higher temperatures lead to more energy-efficient MCU states.

Nevertheless, while the self-optimizing approach performs well under constant environmental conditions, it is important to know whether ideal voltage levels can be adjusted using this approach. Therefore, a node was exposed to different temperatures while performing the self-optimization process. Figure 13 shows results of this measurement using the same node at three different temperatures.

This specific node still works with a minimal voltage of 1.48V at a temperature of $T_{hot} \approx 45^\circ\text{C}$. At a temperature of $T_{cold} \approx 7^\circ\text{C}$, accurate functionality is only given if the voltage level is not below 1.62V. However, the basic functionality of the automated finding of ideal voltage levels is not influenced by different temperatures. Thus, it is possible to adapt the voltage level to the ideal state even for various environmental conditions, which is essential for the usage of undervolting in real WSN applications.

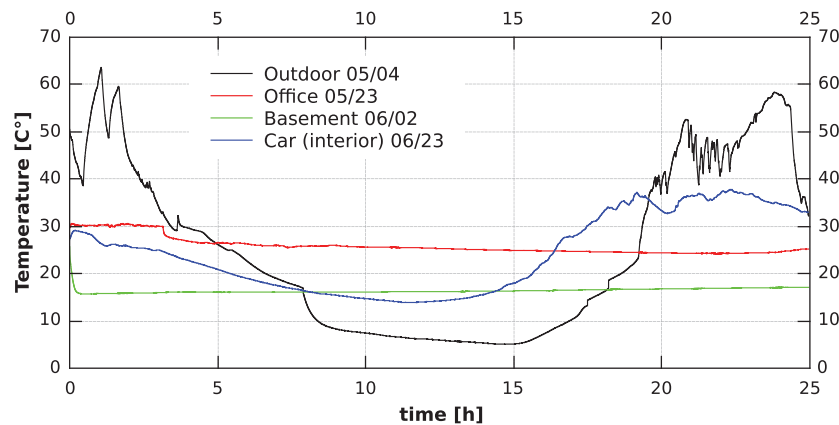


Fig. 14. Measurement of the temperature value (die temperature) of some exemplary WSN application areas.

5. CONSIDERING ENVIRONMENTAL CONDITIONS

Even without an allegedly daring energy management technique like undervolting, a classical WSN already struggles with outdoor deployments [Boano et al. 2013, 2014; Beutel et al. 2011]. Highly varying temperature especially leads to decreased efficiency in the entire WSN. For example, temperature fluctuations have a strong impact on clock drift, which might effect time synchronizations. Considering the already required clock recalibration with the aid of the SI (cf. Section 4.3.2), this issue can be mitigated as a side effect when checking the main MCU periodically. Indeed, it should be mentioned that the SI needs a temperature-stable reference clock to generate usable feedback for main clock recalibration.

In addition to this effect, the probability of hanging in resetting nodes is also higher so that running a WSN for quasi “normal” conditions already becomes a challenge. However, with the background of CMOS gate temperature dependency and the concept of active undervolting with the aid of a control loop, it could be possible to achieve both a more energy-efficient and robust WSN even in challenging environments.

To estimate how deployed sensor nodes are stressed by their environmental conditions, we measured the die temperature of an IC on the undervolting-capable sensor nodes (cf. Section 3) of some exemplary application areas (outdoor, basement, office, car). The measurements took place in central Europe with a sample rate of 0.2Hz. For this evaluation, we did not set the voltage level below the specifications.

Figure 14 shows the absolute temperature values for a period of approximately one day where the start time of the measurements is about 4 p.m. Whereas nodes inside a building are less exposed to varying temperatures, outdoor deployments experience a wide temperature range even in a single day, conceivably between 5°C and 65°C. Indeed, when nodes are exposed to sunlight, the absorption of the IR-radiation leads to temperature values much higher than the actual air temperature. Long-term evaluations showed that temperatures up to 83°C are reached [Boano et al. 2013].

The measurements in Figure 15 also show that outdoor nodes are exposed to a higher variation in temperatures compared to indoor deployments. Even within a few minutes the temperature varies between several degrees. With regard to Section 4.3, these varying environmental conditions may provoke failures or inefficient energy usage. Feedback on whether the absolute minimum voltage level V_{abs} is reached is indicated by the detection of a provoked error caused by undervolting. Hence, at least one failure occurs whenever a new ideal voltage level is searched due to changes in temperature. It is possible that the disadvantage of potential errors could be higher than the energy savings under specific environmental conditions.

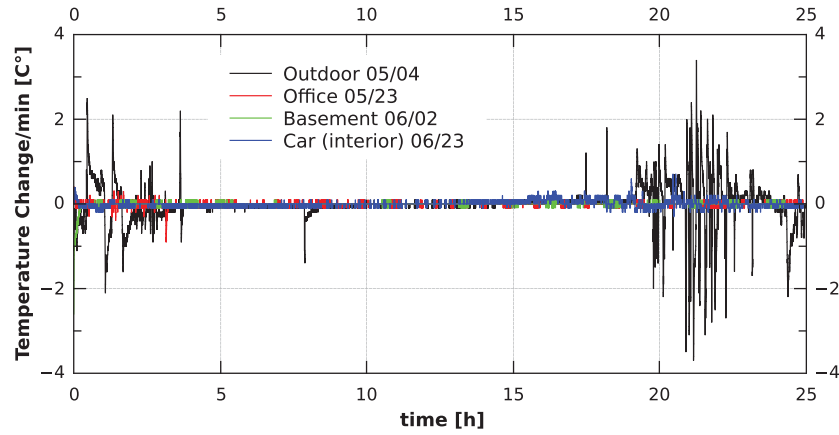


Fig. 15. Temperature change within one minute.

Thus, using the strong temperature-dependent concept of undervolting becomes a challenge. Dropping temperatures especially could lead to unstable system states when setting the voltage level to the edge of functionality. Conversely, in anticipation of such high temperatures, the voltage level of nodes can be lowered even further. All in all, assuming that the ideal voltage level is bounded by the actual temperature, the energy efficiency of a sensor node will depend on its current context.

Another potential optimization, when the characteristics of the environmental peculiarities are known, is to adapt the period of the control loop dynamically. In Section 3.3, the overhead due to additional hardware complexity (SI usage, level translator) was mentioned. A more stable environment leads to longer intervals of the control loop and therefore to fewer duty cycles. On the other hand, a stressful outdoor usage would cause a shorter period. Here, the update interval is set to a static value of 1s (highest). Experience with deployed and undervolted WSNs will provide information that may be used in future work to model a dynamic interval or a reactive adaptation.

However, to achieve reliable undervolting on the sensor nodes, the simple control loop has to be extended. Otherwise the hitherto proposed control of active undervolting would only be useful for mostly homogeneous application areas. The next section introduces supervised learning methods to extend the applicability substantially.

6. SUPERVISED LEARNING

By aggregating ideal voltage levels together with environmental information, a sensor node can learn from its surrounding world over time. However, learning first needs the ability to remember certain events. Beyond that, this memory can be used to draw conclusions that help to improve the actions of the specific nodes continuously.

In our system, these memorizable events are detected failures caused by undervolting while finding ideal voltage levels. Thus, it is possible to determine which voltage level is suitable at which temperature and how the clock rate has to be adjusted for those operation points. Moreover, a learning approach means that a sensor node should not run into every possible failure. Hence, in Section 6.2, a subset of operating points is used to predict further system states. The whole supervised learning process is part of the SI.

6.1. Learning Phase

Figure 16 shows the components and flowchart of the learning approach to adapt voltage level and clock rate as a function of temperature. The initial state of the learning phase starts by searching the ideal voltage level for the current environmental condition $V_{opt}(T)$. This process was described and evaluated in Section 4.

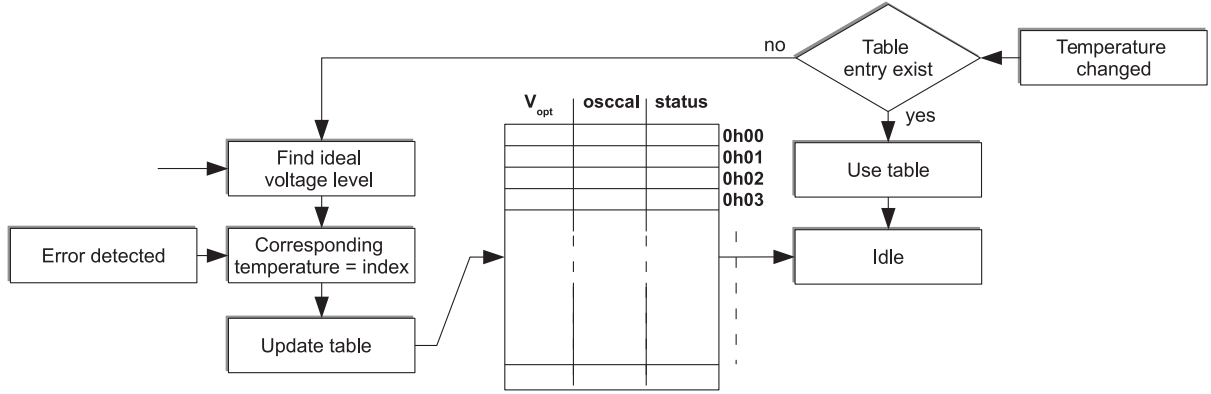


Fig. 16. Principle of the supervised learning approach for undervolting nodes. Assured states (ideal voltage level and recalibration register) are stored in a table.

After finding $V_{opt}(T)$, the temperature is used as the input for a function $x(T)$ to calculate the table index of the corresponding settings. The quantization of the temperature values leads to a reduced memory footprint of the table. The allocation of how many temperatures are covered by one ideal voltage level depends on the specific MCU. It is easy to condense the table in this way when the detected voltage level is used to cover higher temperatures only. However, the increments of the characteristic curve become coarser and energy might be wasted. Two temperature values are assigned to one voltage level for our platform (quantization factor $q = 2$). Equation (7) gives the used function. T_{offset} is used to cover negative temperatures:

$$x(T) = \left\lfloor \frac{T + T_{offset}}{2} \right\rfloor. \quad (7)$$

A table entry consists of the triple (*ideal voltage level*, *OSCCAL recalibration register*, *Status*) with the following information:

Ideal Voltage Level. This is the ascertained V_{opt} for the current temperature T and is defined as follows:

$$V_{opt}(T) = V_{abs}(T) + V_{mrg}. \quad (8)$$

The absolute minimum voltage level V_{abs} is reached whenever the first error is detected. Running the MCU at V_{abs} must not mean that further execution completely fails. Nevertheless, the probability of occurring failures is considerably increased. Hence, V_{abs} is adjusted by a safety margin V_{mrg} . However, this supervised learning approach would not fail even if V_{mrg} is set to zero; indeed, it allows different strategies. The disadvantage of such a setting compared to conservative settings of V_{mrg} is an increased amount of errors within the learning phase. On the other hand, an oversized margin will curb the possible energy savings. In practice, we set $V_{mrg} \approx 15\text{mV}$ as is a sufficient tradeoff between reliability and energy efficiency.

OSCCAL Recalibration Register. As mentioned earlier, the clock rate of the MCU is influenced by undervolting. To recalibrate the main clock, the MCU ATmega1284p [Atmel 2010] offers an OSCCAL that is accessible by software during runtime. Because the recalibration is done continuously, the OSCCAL register is always up-to-date. Hence, when finding a stable state, OSCCAL can be associated directly.

Why should the value of OSCCAL be stored when recalibration is done continuously: In connection with energy efficiency, the SI adds a non-negligible overhead that has to be exceeded by the benefit of undervolting, as described in Section 3.3. As soon as a node has learned enough table entries, the periodic execution of the control loop is

unnecessary. After sending the information of ideal states as a function of the environmental temperature to the main MCU, the SI can be switched off to save even more energy (cf. Section 3.3).

Status. The status entry simply holds information for the processing. In addition to the information on whether an ideal state for a given temperature exists, it indicates whether this state is ascertained or predicted. For further use cases, it can also indicate the reliability of an entry by, for example, taking into account the average duration without any error.

Considering DVS techniques, it would further be possible to expand the table by the dimension of the clock rate f_{cpu} . However, due to limited memory capabilities of the SI, we focused here on a fixed clock rate to optimize dynamic power management (DPM) by IdealVolting.

6.2. Prediction

The goal of the second phase is to derive a trend that can be used to predict ideal voltage levels even for unknown environmental conditions. With regard to the reliability of the node, a prediction of ideal voltage levels would be of enormous benefit. Assuming that *only* the learning phase would be used, every new temperature would cause an error before the corresponding V_{opt} is stored. Hence, to shorten the learning phase, some ascertained table entries are used to predict the missing ideal voltage levels.

Considering Equation (2), the threshold voltage of a CMOS-based MCU depends on the temperature coefficient α . Because a coefficient is equivalent to a factor, we assume a linear dependency of V_{opt} and the temperature T . Measured results in a climatic chamber [Kulau et al. 2015] and previous long-term measurements of undervolting confirm a linear connection between voltage level and temperature. A higher temperature leads to a lower ideal voltage level and vice versa.

However, the straight proportional dependency of V_{opt} on the temperature simplifies the prediction. A special case of regression analysis, linear regression, can be used to estimate the slope m and the y-intercept b . Together with the function $x(T)$ of the table described in Section 6.1, ideal voltage levels can be given as a function of the temperature T (disturbance variable ϵ_T):

$$V_{opt}(T) = \lceil m \cdot x(T) + b \rceil + \epsilon_T. \quad (9)$$

Consider n as the number of ascertained ideal voltage levels V_{opt_d} stored at table index x_d . Then, the slope m is calculated by the following equation:

$$m = \frac{\sum_{d=1}^n x_d V_{opt_d} - \frac{(\sum_{d=1}^n x_d)(\sum_{d=1}^n V_{opt_d})}{n}}{\sum_{d=1}^n x_d^2 - \frac{(\sum_{d=1}^n x_d)^2}{n}}. \quad (10)$$

By using m , the intercept b is given as:

$$b = \frac{\sum_{d=1}^n V_{opt_d}}{n} - \frac{\sum_{d=1}^n x_d}{n} \cdot m. \quad (11)$$

Because the voltage level of the I²C voltage scaling module is set by an 8-bit register, the calculated result of $V_{opt}(T)$ is rounded to the corresponding integer value (cf. Equation (9)). Both rounding and linear regression show that a disturbance ϵ_T is part of the calculated optimum voltage level. On the one hand, $\epsilon_T > 0$ would have a slightly negative effect on energy efficiency. On the other, $\epsilon_T < 0$ could jeopardize the postulated reliability of this approach. To ensure a safe execution of applications, all ϵ_T have to be lower than the safety margin V_{mrg} of Equation (8):

$$|\epsilon_T| < V_{mrg} \quad \forall V_{opt}(T). \quad (12)$$

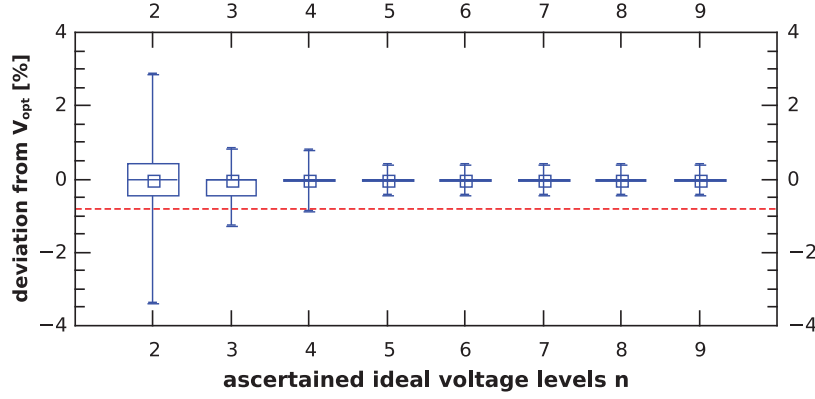


Fig. 17. Averaged deviation of the predicted optimum voltage levels from the measured ones. Deviations below the dashed line could lead to unreliable execution.

How many ideal voltage levels n have to be found before a prediction fulfills Equation (12). For this purpose, we draw on our previous studies [Kulau et al. 2015], where ideal voltage levels were ascertained by measurements in a climatic chamber. This reference measurement was used to verify the prediction of ideal levels. First, we defined every permutation of a subset of ideal levels with different sizes n . Based on these subsets, we executed the prediction and compared the results with the reference measurement. The deviations from the predicted values compared to the measured ones is shown in Figure 17.

Figure 17 shows that the accuracy of the predictions grows with the number of ascertained ideal voltage levels that are taken into account. This dependence was expected. Fortunately, only $n = 5$ points are necessary to predict an almost reliable and energy-efficient characteristic voltage curve $V_{opt}(T)$. Initially for our implementation, we conservatively set the threshold for the start of prediction to $n \geq 7$.

However, a decent prediction is only possible on the assumption that the amount of the ascertained ideal voltage levels is free of outliers. Especially when dealing with undervolting, the values listed in the table of the learning phase should be validated before starting the prediction. An adequate procedure is to take the standard deviation into account, for example by using the 2σ -method:

$$V_{opt_d} \text{ is outlier} \Leftrightarrow \left| \frac{V_{opt_d} - \overline{V_{opt_d}}}{\sigma} \right| > 2 \quad (13)$$

$$\text{with } \overline{V_{opt_d}} = \frac{\sum_{d=1}^n V_{opt_d}}{n}$$

$$\text{and } \sigma = \sqrt{\frac{1}{n-1} \cdot \sum_{d=1}^n (V_{opt_d} - \overline{V_{opt_d}})^2}$$

When an outlier is detected, there are several ways to solve this issue. Our implementation deletes the corrupted entry and decreases n . Thus, the prediction is deferred. Another possibility could be to repair the outlier by using, for example, Equation (9). One solution is to substitute the entry with the mean value of neighboring entries. However, interpolated values would add extra uncertainty to the prediction.

In addition to the voltage level, a table entry contains the register value OSCCAL for recalibration of the clock rate. The process of recalibration is estimated as a linear function, too. Hence, we use the same methods just described to predict the missing OSCCAL values.

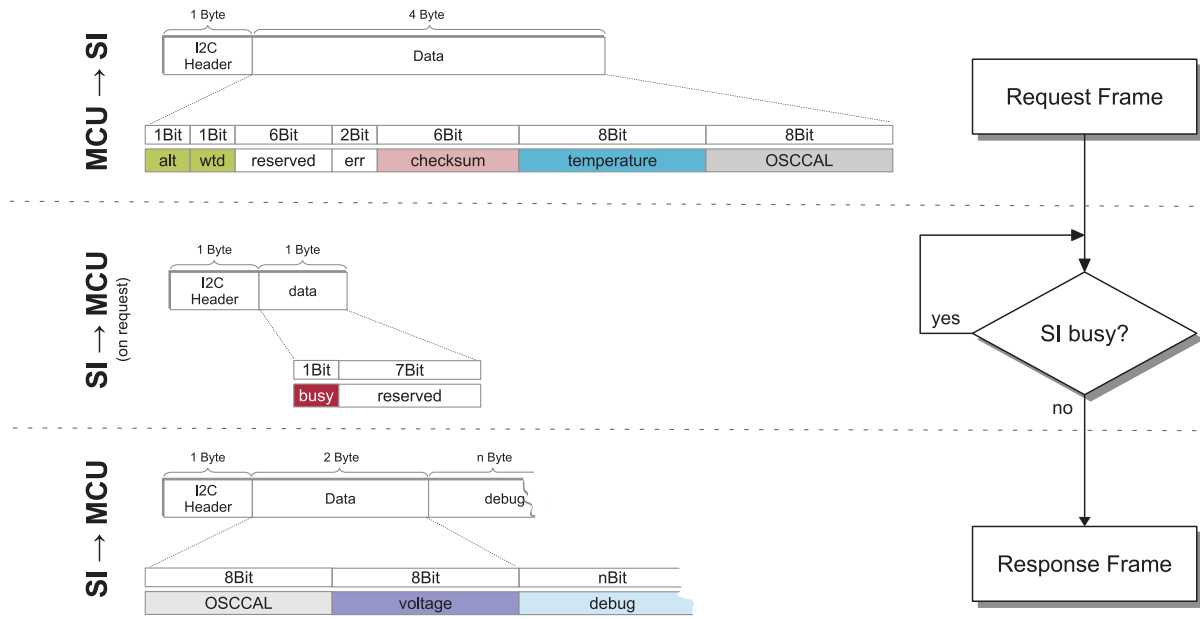


Fig. 18. Implementation of the Control Loop – I²C communication protocol between the main MCU and the SI.

7. IMPLEMENTATION

As mentioned, the major part of the software implementation of active undervolting is part of the SI. One advantage of the request-response mechanism is that the software overhead on part of the main MCU is low. For example, a simple concurrent Contiki [Dunkels et al. 2004] process is absolutely sufficient and influences application processes only slightly. However, compared to the simple communication protocol of the automated detection of ideal voltage levels (cf. Section 4.2), a bit more effort to achieve the supervised learning method is needed because the exchanged frames are more complex. Figure 18 shows the principle of this communication.

A request frame from the main MCU to the SI (MCU → SI) contains four bytes. In the first instance, the information of a request frame helps the SI to decide whether the MCU runs stable or not. An alternating bit (alt) is part of the first byte. If the SI receives the same bit twice, something probably went wrong. An example for this could be that undervolting leads to a software reset by calculating the PC as zero. Other bits of the first byte are used to control the functionality of the SI, like enabling or disabling the external watchdog (wtd). The second byte contains the result of the MCU spot test whereby the SI checks for calculation errors. Of course, information about the current temperature is part of the request frame because the supervised learning approach is based on the correlation between the minimum voltage level and the corresponding temperature. The value of the recalibration register (OSCCAL) is sent to the SI to countercheck the current clock rate settings.

The information of a request frame is used to indicate ideal voltage levels at different environmental conditions and store those working points in a table. So, in general, the SI implements the flowchart presented in Figure 16. The program paradigm of the SI is automata-based programming. Thus, the software implementation as a state machine allows the verification of functionality and ensures determinism. Moreover, with regard to the prediction of further operating points (cf. Section 6.2), a state machine is easy to expand by simply adding transitions.

During calculations, the SI sets a lock bit which indicates that the SI is busy because the calculation has not finished yet. Subsequently, the new values for setting the voltage

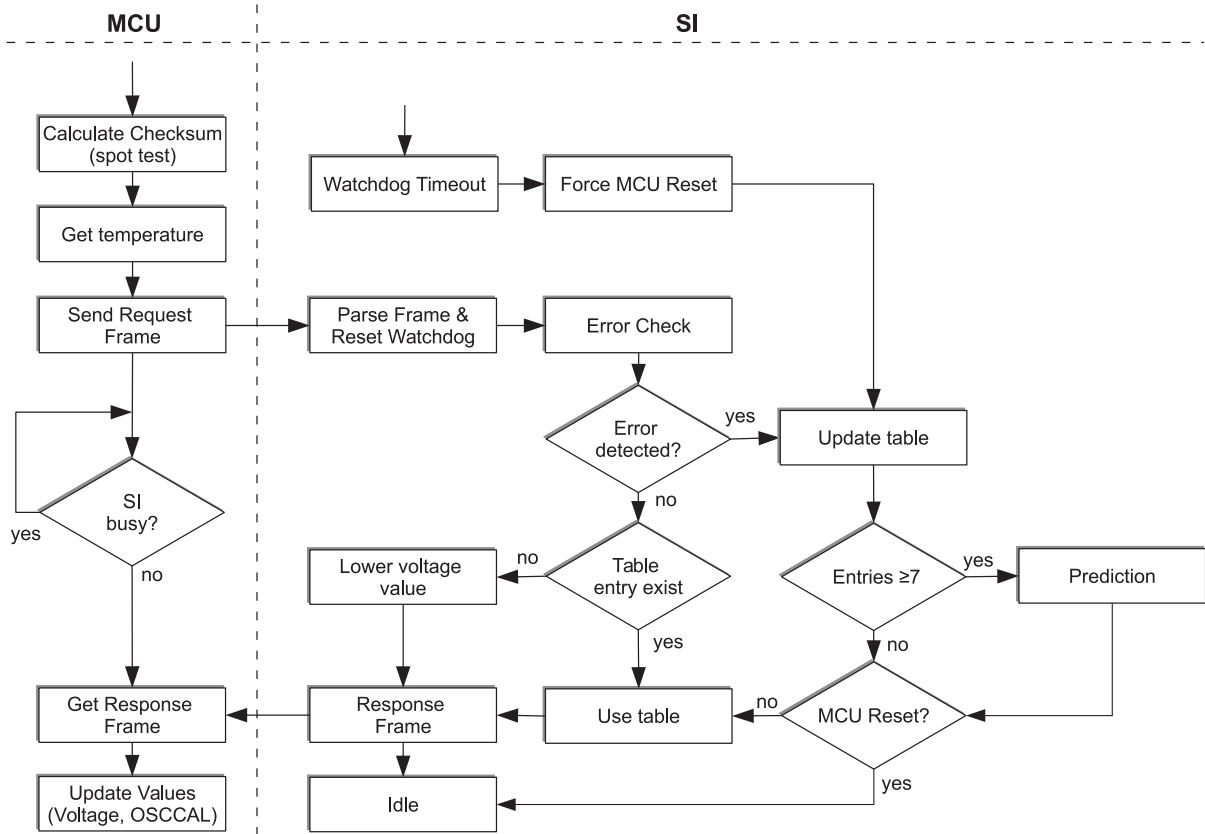


Fig. 19. Summary of the operation of the control loop to self-optimize the ideal voltage level adaptation. Note that the given flowchart is solely a rough draft of the implementation.

level and recalibration register are offered in the response. In addition, the SI could offer debug information like logged failures, fill level of table, or the time of the last control loop cycle. Afterward, the SI awaits the next request frame, which is triggered by the main MCU. Whenever a request frame is missing, the SI forces the main MCU to be reset. Thus, a deadlock situation caused by undervolting or other influences is avoided by the external watchdog (cf. Section 4.2).

To summarize the operation of IdealVolting, Figure 19 gives an overview of how the adaptation of ideal voltage levels is implemented. The duty-cycle of the SI implementation is 0.58%.

8. EVALUATION

Evaluations demonstrate the overall functionality, effectiveness, reliability, and usability of the presented approach.

Here, we are less focused on the energy-saving effects of undervolting than in Kulau et al. [2015], and the evaluation in Section 4.3.3 already showed the enormous potential energy savings of undervolting. Nevertheless, a comparative evaluation of undervolting as provided by IdealVolting against other state-of-the-art energy management techniques is also part of this section.

8.1. Learning Phase: Long-Term Observations

In Section 4.3.3, a long-term evaluation showed that a stable WSN application is ensured even though the nodes are set below the manufacturer's recommended voltage level. However, this test took place within an office building and the nodes were less stressed by environmental changes. Hence, an evaluation is needed to show the

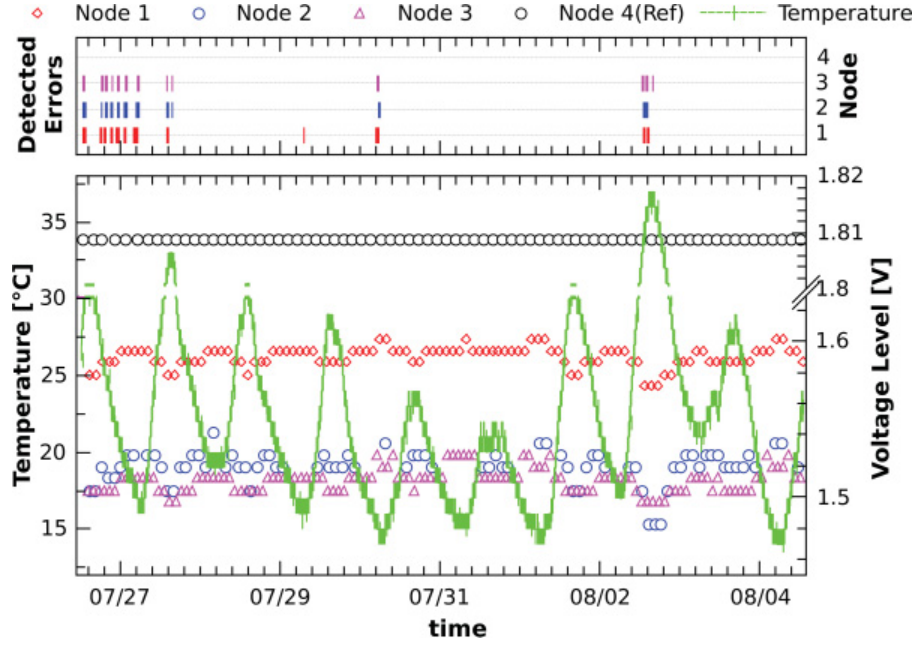


Fig. 20. Temperature measurement and voltage levels of 10 days of experiment in an attic with three supervised learning undervolted nodes at $f_{cpu} = 4\text{MHz}$ (without prediction process).

long-term stability of undervolted nodes deployed in challenging environments, especially under realistic temperature changes (cf. Section 5).

During this evaluation, the prediction process is disabled. The goal is to show the principle of the learning phase and that ascertained states are safe (i.e., that they are a reliable basis with which to predict further states). Overall, we used four nodes and deployed them in an attic in summer. Three of these nodes were undervolted; the fourth node used a regular voltage level for reference purposes. During this evaluation, the clock rates of the nodes were set to $f_{cpu} = 4\text{MHz}$, which leads to a minimum recommended voltage level of $V(f_{cpu} = 4\text{MHz}) = 1.8\text{V}$ according to the datasheet.

Figure 20 shows that this level is significantly undershot by all undervolted nodes. Thus, since the voltage level has a quadratic influence on the power dissipation of CMOS gates [Tietze and Schenk 2007], all undervolted nodes operated with notably increased energy efficiency.

As noted, during this 10-day evaluation the prediction process was disabled. While the nodes execute the learning phase, every unknown temperature leads to an error because of the feedback for readapting the voltage level. Especially at the beginning, the amount of detected errors is considerably increased (as can be seen in the upper part of Figure 20). Every new temperature value adds an entry to the table of a node. From this, it follows that after the first day cycle the density of errors drops as stable states are already ascertained for most recurring environmental temperatures. Later, sporadic errors occur whenever the nodes are exposed to hitherto unknown temperature values. Thus, with regard to lifetimes of several months, the total amount of errors will decrease even further. Yet, some errors may still be provoked whenever an ideal voltage level is searched for a new temperature.

8.2. Function Test of IdealVolting

To limit the absolute amount of errors, in Section 6.2 the prediction process was presented. The learning phase together with the prediction process allows for the reliable usage of undervolting (IdealVolting). For this evaluation, we used a climatic chamber

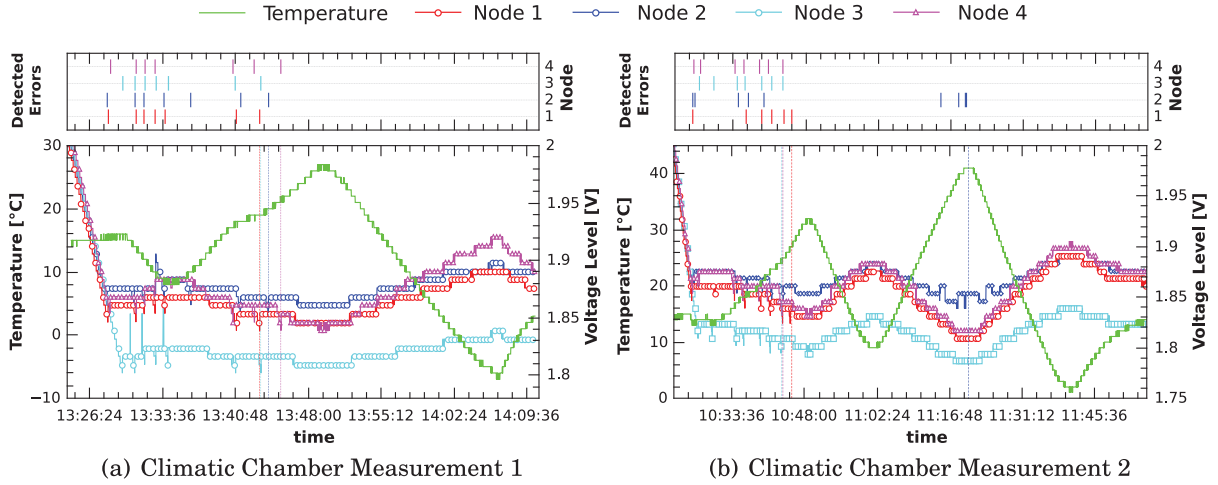


Fig. 21. Two independent measurements in a climatic chamber with four nodes using the supervised learning approach. The temperature values were measured by the nodes themselves, but they are also validated against a non-undervolting reference node.

to achieve widespread temperatures within a short time compared to real-world deployments. In this case, we used four nodes running the presented implementation of the supervised learning approach plus prediction. Every node executes an exemplary application that samples the temperature of a pressure sensor and continuously compensates the raw values by a predefined algorithm. Application data and information about the state of the supervised learning process were sent to a sink node via an IEEE 802.15.4 wireless link. The clock rates of the nodes were set to $f_{cpu} = 8\text{MHz}$, which leads to a minimum recommended voltage level of $V(f_{cpu} = 8\text{MHz}) = 2.4\text{V}$ according to the datasheet.

Figures 21(a) and (b) show the results of two independent measurements. At the beginning, the voltage level is lowered from $V = 2.4\text{V}$ linearly until the first unstable state is found. Just as in the previous evaluation, first every new temperature value evokes errors due to the searching for ideal voltage levels. However, in Figure 21(a) it can be seen that after about seven detected errors, which corresponds to $n = 7$ ascertained ideal voltage levels $V_{opt}(T)$, the prediction is executed (vertical, dashed lines). Although the nodes are exposed to unknown temperature conditions afterward, no more errors are detected due to prediction. The same result is obtained by the second measurement (cf. Figure 21(b)). Conspicuous is the deferred execution of the prediction of node 2: Compared to the other nodes, node 2 was not able to ascertain the same amount of table entries in the same period of time. However, the missing states for prediction are collected later. Afterward, node 2 is as reliable as the other nodes.

Both measurements show that the voltage level of each node is individually adapted to the environmental temperature. A higher temperature leads to a lower voltage level and vice versa. Thus, the general effect of increased CMOS gate current consumption at higher temperatures is compensated for; indeed, in this case, the node's energy efficiency is even increased. For measurement 1, Figure 22 depicts the characteristic voltage curves—either ascertained or predicted—as a function of the temperature for all four nodes. It can be seen that every node has its individual characteristic regarding minimum voltage level (intercept) and temperature dependency (slope). From the characteristic voltage curves, it follows that, according to the energy efficiency within a WSN, using IdealVolting will add more heterogeneity. For example, in further applications, the individual energy footprint of nodes could be used to optimize load balancing or routing schemes.

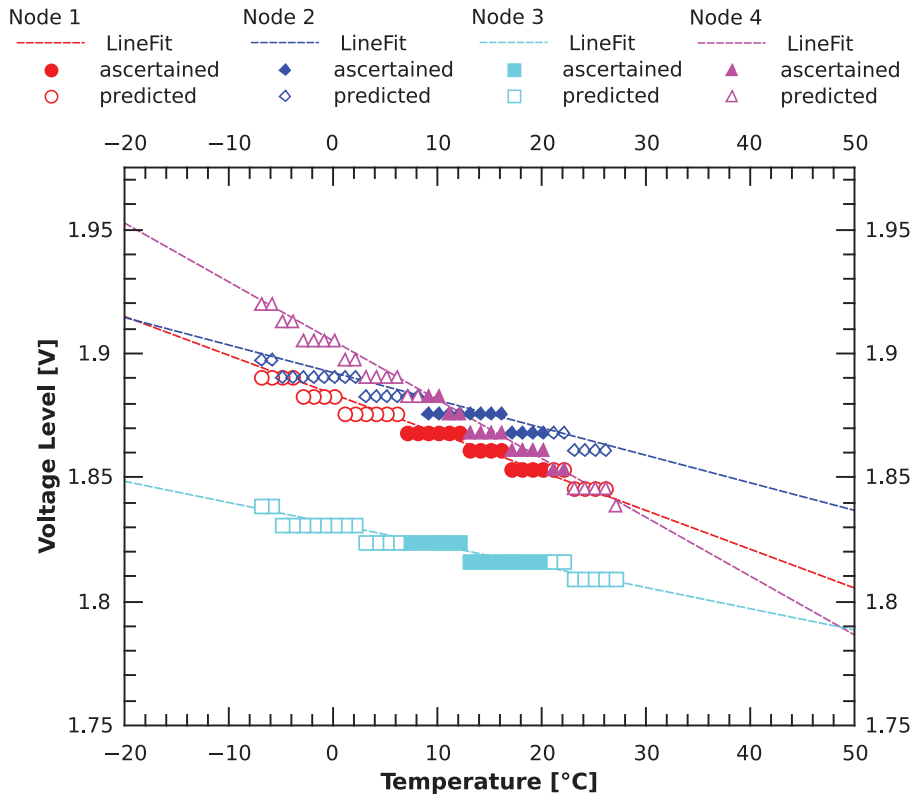


Fig. 22. Characteristic voltage curves. Ideal voltage levels as a function of temperature. Ascertained and predicted states of the first climatic chamber measurement.

Table V. Mean Deviation and Standard Deviation of the Temperature Measurement Compared to the Reference Node

	Node 1	Node 2	Node 3	Node 4
mean deviation [%]	2.264	0.6255	0.521	0.966
standard deviation σ [%]	3.219	2.558	3.1584	2.775

8.3. Quality of Sensing: Varying Conditions

The exemplary overlaying application of the previous evaluation was to sample, compensate, and send the temperature value of a sensor to a sink node. Once again, to show that the presented approach does not have a negative impact on the quality of sensing, we compared the received temperature values with those of a reference node. Table V gives the deviation of all values of climatic chamber measurement 1. The average standard deviation is about 3%. In this case, this equals a total deviation of $\Delta T \approx 0.77^\circ\text{C}$. Considering the declared accuracy of the sensors ($\pm 3\%$), it is hardly attributable to undervolting.

8.4. Robustness

To test the robustness of the IdealVolting implementation, we used a light bulb as a heat source and a freezer to cool down a node abruptly. The process of supervised learning and prediction is the same as described earlier. Figure 23 illustrates this short test and the outcome.

To simulate a challenging environment, we exposed the node to sporadic varying temperatures. The different colors indicate how the current voltage level was selected. The node starts with the iterative search of ideal voltage levels. The searching is increasingly replaced by ascertained ideal voltage levels, and, after 1400s the prediction

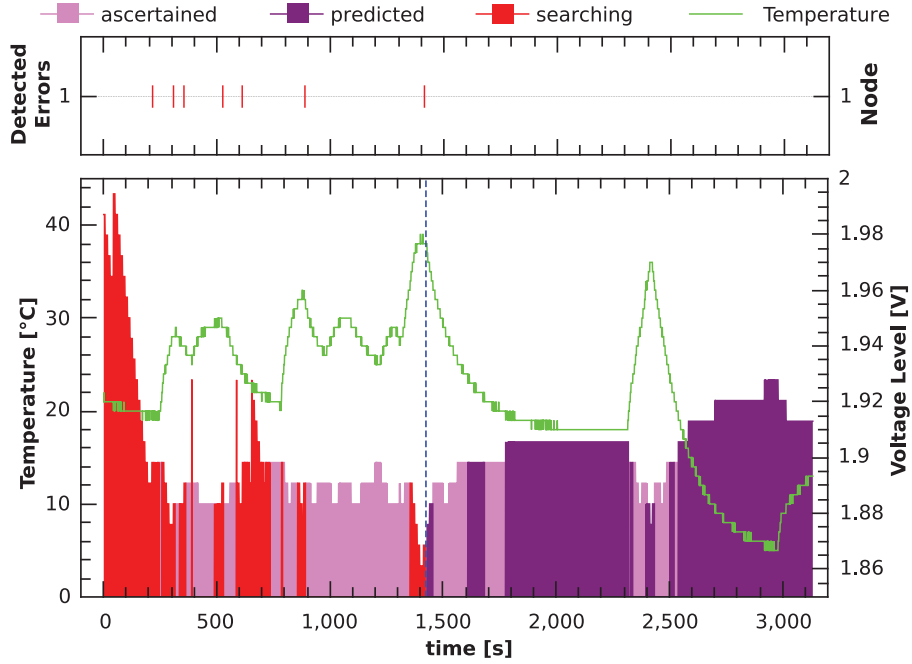


Fig. 23. Testing the robustness of IdealVolting by using a lightbulb as heat source and a freezer to cool down the node abruptly.

took place (dashed line). After prediction, the node was alternately placed in a freezer or under a lightbulb to generate a sudden drop or rise of temperatures. However, after the prediction, no error was detected, which allows IdealVolting to be used in rough environmental conditions as described in Section 5.

8.5. Reliability: Long-term Outdoor Evaluation of IdealVolting

To summarize long-term evaluation, one node from the previous evaluation was deployed outdoors in central Europe to measure the environmental temperature. The collected temperature values and various status information were sent to a sink node via an IEEE 802.15.4 wireless link. The sink node was placed indoors. The clock rate of the nodes was set to $f_{cpu} = 8\text{MHz}$.

In addition to the measurement task, the outdoor node executed the proposed IdealVolting scheme, which includes the learning phase, the prediction of further system states, and thus the continuous adaption to energy-efficient but reliable system states at any environmental condition. The overall duration of this evaluation was 135 days or ≈ 4.5 months.

8.5.1. Learning and Prediction. Figure 24 depicts the learning phase, in which the general behavior is similar to the evaluation in Section 8.2. The node lowers the voltage level from $V(f_{cpu} = 8\text{MHz}) = 2.4\text{V}$ linearly until the first ideal voltage level is found. Depending on its environmental condition, the node collects more and more table entries until the prediction process takes place (dashed line). In practice, after almost 8 hours the node was able to calculate its characteristic voltage curve, which prevents errors for further system states.

Of the total amount of $n = 7$ errors throughout the long-term evaluation, all of them occurred within the learning phase because no further errors were detected during the remaining evaluation. In relation to the overall duration of 3,241 hours, a timespan of 8 hours with possibly sporadic errors seems to be tolerable or even negligible for many applications. Moreover, as an extension to the presented approach, the parameters of the characteristic curve were stored in the Electrically Erasable Programmable

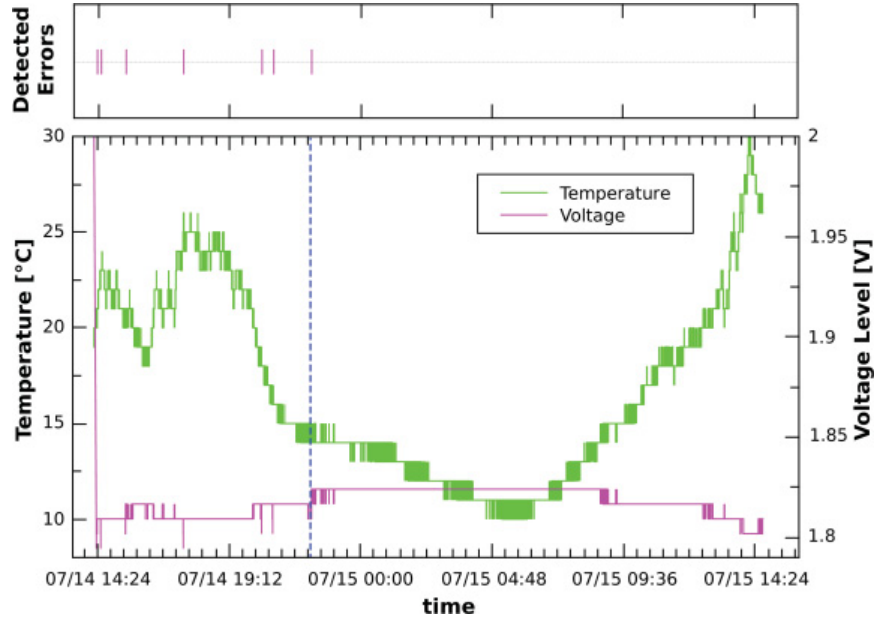


Fig. 24. Learning phase of the long-term evaluation.

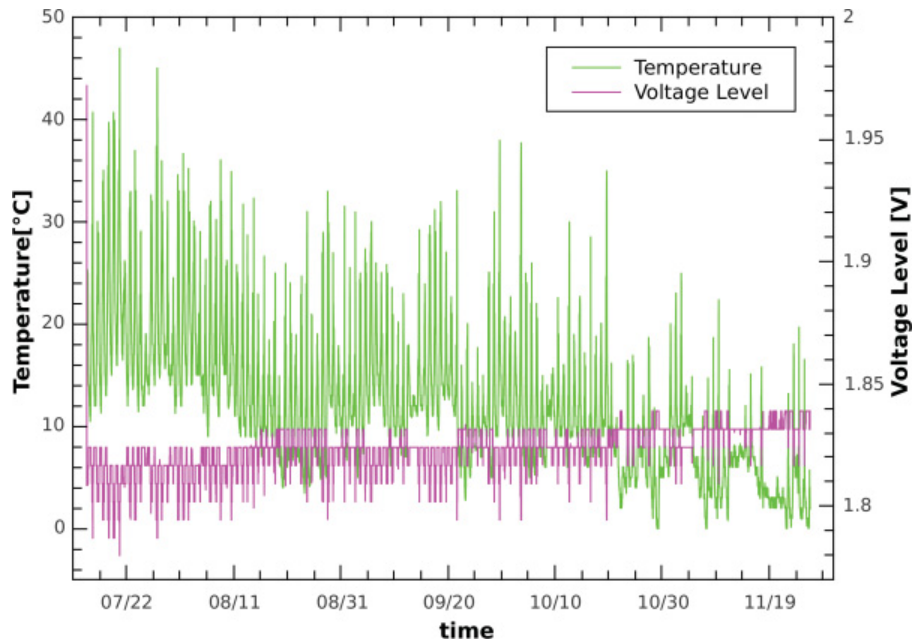


Fig. 25. Long-term evaluation of almost 4 months.

Read-Only Memory (EEPROM) of the SI. Hence, when reusing the node for further applications, it may even skip the learning phase.

8.5.2. Long-term Results. Figure 25 shows the temperature and voltage level of the entire evaluation span. It can be seen that an outdoor node is exposed to a challenging environment (see Section 5). However, IdealVolting acts inversely proportional to the temperature and thus avoids unreliable system states.

Table VI summarizes notable facts about this evaluation after the prediction process has been performed.

Table VI. Summary of the Long-term Evaluation (After Prediction)

	Overall			Single Day		# Test Cycles
	max	min	mean	max Δ	mean Δ	11646372
Temperature [°C]	47.0	-1.0	11.74	32.0	15.93	# Detected Errors
Voltage Level [V]	1.839	1.779	1.823	0.03706	0.0199	

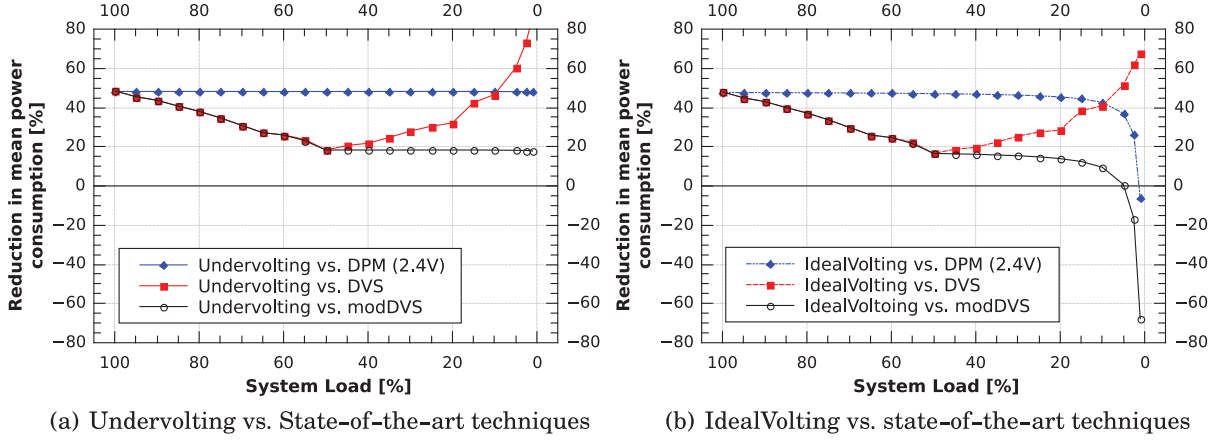


Fig. 26. Comparison of undervolting comparing IdealVolting against state-of-the-art power management techniques.

Compared to the outdoor measurement in Section 5, the maximum difference in temperature of 32°C is rather small because the node was placed in a shady spot without direct sunlight.

Indeed, the total amount of errors is still non-zero. As described earlier, the provoked errors are necessary to derive the characteristic curve of the node. Yet, assuming that a newly installed WSN often needs time to run stable anyway, the errors that occur during the learning phase are almost negligible. Moreover, whenever no failure is tolerable, it is possible to pre-set the nodes in the lab because the characteristic curve is stored in the EEPROM after prediction. Hence, the node can skip the transiently unreliable learning phase if needed.

8.6. System Gain in Power Dissipation

In Kulau et al. [2015], we showed the enormous energy savings when setting the voltage level below the recommended specifications. Unfortunately, the presented IdealVolting approach of a self-optimizing and reliable undervolting comes with some overhead that has to be outweighed by the expected gain. In Sections 3.2 and 3.3, this issue was discussed. Here, the question is whether IdealVolting is still beneficial and how it compares to other power-management strategies.

With regard to other state-of-the-art power management techniques, this evaluation compares IdealVolting to DPM, DVS, modDVS, and pure undervolting. modDVS is an improved variant of DVS for low-power MCUs (see Kulau et al. [2013] for more details about modDVS).

For a varying system load, the power dissipation of three different MCUs was measured for each of the aforementioned power management techniques. The comparison of the gain of pure undervolting using IdealVolting against DPM, DVS, and modDVS was therefore always based on the average power consumption of the three MCU instances. The deviation from nominal voltage level for undervolting compared to IdealVolting was $\Delta V = 22.9\%$. The results are given in Figures 26(a) and (b).

Due to additional overhead when using IdealVolting (cf. Section 3.3), the gain drops at low duty cycles. Hence, IdealVolting is mainly beneficial the longer a node persists

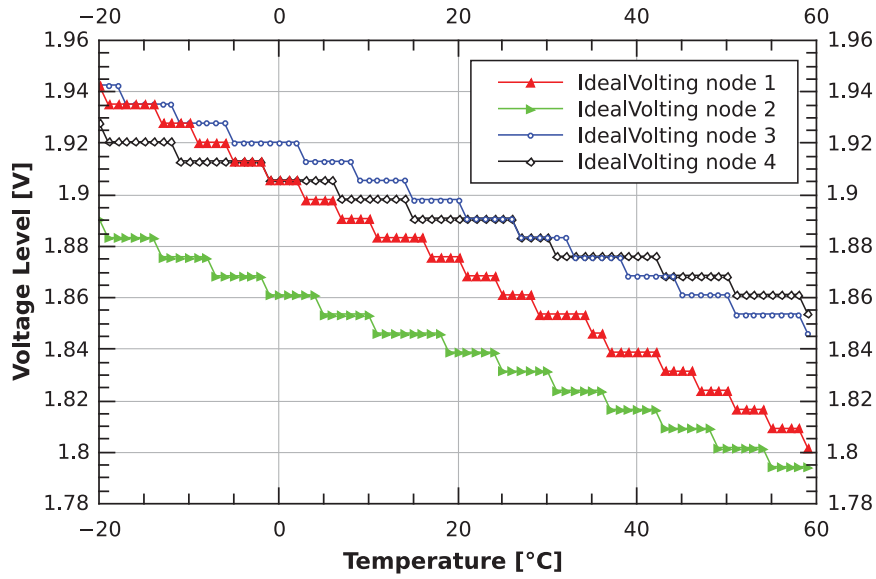


Fig. 27. Characteristic voltage curves of the nodes that used IdealVolting. Every node shows an individual characteristic.

in active mode. WSN applications with extraordinary sleep periods (e.g., Chen et al. [2013]), would not gain from the current IdealVolting implementation. Nevertheless, the presented approach can be combined with existing power management techniques for such scenarios [Jayakumar et al. 2014]. During power-down sleep states, the periodic check of the MCU is needless. Unused components like the SI and the level translator can be totally switched off.

8.7. Increased Energy Efficiency: Long-Term Outdoor Evaluation of IdealVolting

In Section 4.3.3, we performed a long-term evaluation of an exemplary WSN, showing how the self-optimizing control loop ascertained ideal voltage levels. However, the evaluation took place within an office environment so that no voltage adaptation due to temperature fluctuations was needed.

With IdealVolting, we showed that reliable undervolting is also possible for harsh environmental conditions. To show how IdealVolting performs against the naive approach of Section 4.3.3, we repeated this evaluation but (i) we deployed the nodes outdoors, and (ii) we used the IdealVolting scheme. The application task and general node settings are the same for this application; for more details about the setup of this evaluation, refer to Section 4.3.3.

The characteristic voltage curves for the nodes running IdealVolting are given in Figure 27. As expected, every single node shows its own characteristic in slope and intercept.

The battery discharge curves are given in Figure 28. It can be seen that the results are similar to those of the indoor deployment. All nodes ran stable even for varying environmental conditions, whereas lifetime increased significantly.

The percentage improvement of the lifetime due to IdealVolting is summarized in Table VII. With IdealVolting, the lifetime increases up to 40.78% compared to a node powered at the recommended voltage level of $V(f_{cpu} = 8\text{MHz}) = 2.4\text{V}$.

For the sake of completeness, Figure 29 shows the measured temperature profiles. It should be mentioned that the fluctuation of the evaluations' environmental condition was moderate. The absolute temperatures stayed at $9.2^\circ\text{C} \leq T_{eval} \leq 31.3^\circ\text{C}$.

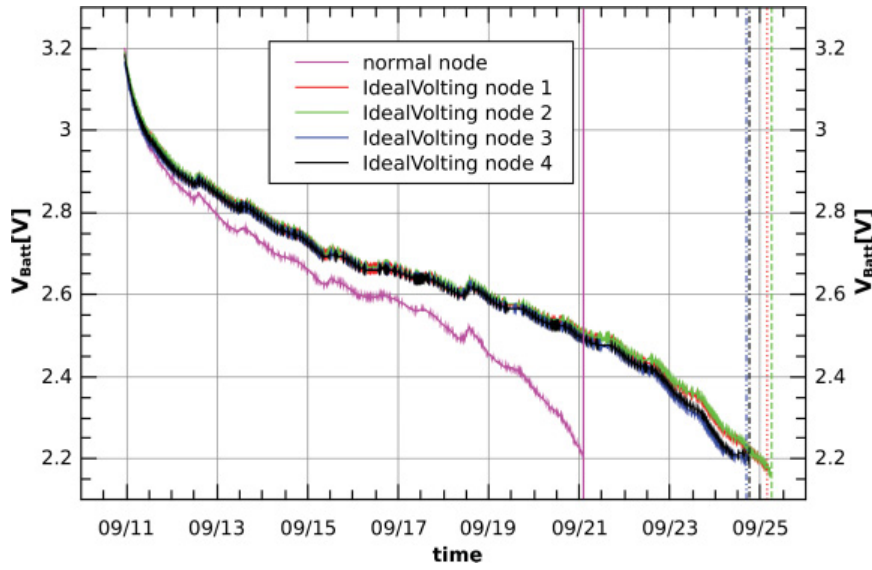


Fig. 28. Battery discharge curves of the long-term evaluation.

Table VII. Prolongation of the Lifetime Compared to a Normal Sensor Node ($V = 2.4V$)

Node	1	2	3	4
Improvement [%]	40.15	40.87	35.43	36.31

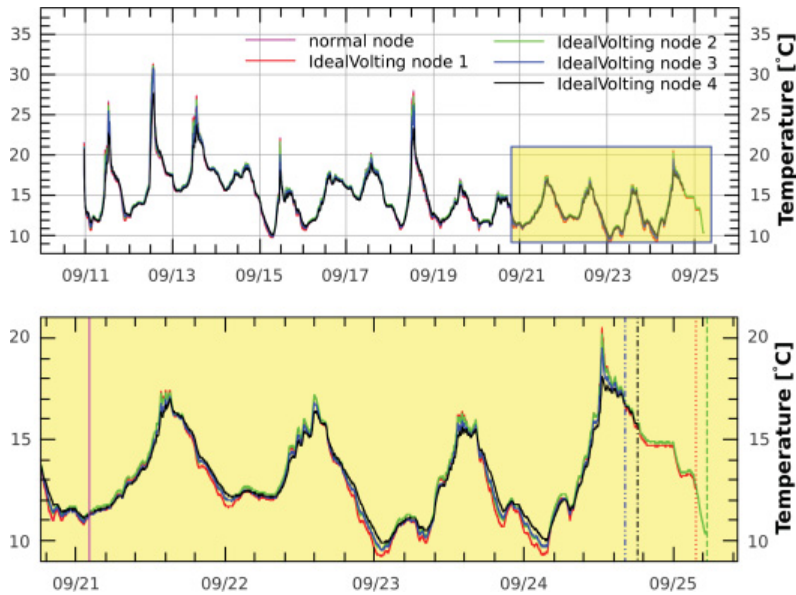


Fig. 29. Aggregated temperature values at the sink node of the long-term evaluation of IdealVolting outdoors.

9. RELATED WORK AND PRELIMINARY STUDIES

One recurring challenge for several WSN application scenarios is the absence of a sufficient and location-independent energy source. Since the improvement of battery capacity, unluckily, does not follow Moore's Law, other ways must be found to enhance the lifetime of sensor nodes and networks [Christopher Mims 2011]. Talzi et al. [2007] and Liu et al. [2004] pointed out that, especially for outdoor deployments, energy efficiency is a key factor when setting up a maintainable WSN.

In addition to many other approaches, adjusting the voltage level to achieve a more energy-efficient WSN is a commonly studied technique. For example, passive methods like that presented by Cho et al. [2007] aim to increase the average MIPS of a MCU by adapting the clock rate f_{cpu} of a system to the current battery state. Active methods adapt both the clock rate f_{cpu} and the voltage level V to the actual system load. Implementations of such DVS approaches are common practice and available even for WSNs. Kulau et al. [2013], Hoermann et al. [2011], and Tuming et al. [2010] show the tremendous benefit of a downscaled voltage level. However, existing DVS approaches still act conservatively and avoid lowering the voltage level below the recommended $V(f_{cpu})$.

Using undervolting to reduce the energy consumption of processing units is not a new idea. In connection with DVS, several articles [Wang et al. 2011; Lee and Zomaya 2010] quote undervolting as an energy-saving technique but without going into details. However, the explicit use of lower-than-recommended voltage levels takes place at the chip level [Kuroda et al. 1998]. Implementations like that in Lefurgy et al. [2013] formulate a legitimization of undervolting similar to that given in the introduction. The recommended operating voltage of MCUs is often selected to cover worst-case scenarios. Thus, energy is wasted when MCUs are used under moderate environmental conditions. One advantage of on-chip solutions is that they allow users to detect failures on-the-fly, which allows one to roll back and replay corrupted operations [Das et al. 2009].

In contrast, this article is focused on a system-level approach based on COTS ultra-low power MCUs as used in common sensor node architectures [Büsching et al. 2012; Moteiv Corporation 2006; Crossbow 2012].

However, for implementing active undervolting, the manufacturers of MCUs offer insufficient information about internal structures. Nevertheless, against the backdrop that COTS microcontrollers must be able to be undervolted, we made an experimental feasibility analysis [Kulau et al. 2015]. The results show a significant potential for energy savings and a sufficient way to detect errors caused by undervolting.

In summary, to the best of our knowledge, there is no related work that considers undervolting as a regular technique to increase the energy efficiency of small embedded devices like sensor nodes and ensures their reliable operation by providing adaptive methods. Moreover, the presented hardware platform is the first undervolting-capable sensor node available to investigate the effects of ideal voltage levels of real-world deployed sensor nodes.

10. DISCUSSION AND CONCLUSION

All in all, many WSN use cases will benefit from IdealVolting. If there is a standard use case for WSNs, then it is to record data in the field, lightly process it, and transmit it. Beyond that, in some applications, the MCU is used for post-processing, such as data compression [Kimura and Latifi 2005] or encryption [Chan and Perrig 2003]. Hence, the workload of the MCU of a sensor node differs and depends on the specific WSN application.

Up to now, we showed that the current IdealVolting implementation performs best for higher workloads above $\geq 4\%$ duty cycle. Nevertheless, we are working on further optimizations of IdealVolting to increase energy efficiency even for low duty cycles.

Energy efficiency is also crucial for nearly every outdoor deployment of WSNs. When established strategies like DPM or DVS are exhausted and a node's voltage level is already adapted to the needed minimum as specified by the manufacturer, IdealVolting gives a significant opportunity for further savings. Yet, simply decreasing the voltage level under the recommended minimum is not always a good idea. The level at which a node or an MCU still works properly depends on the clock speed, the temperature, and

its individual properties due to deviations in the manufacturing process. Out of these three factors, only the clock speed is controllable by the user. Therefore, a static “ideal” voltage level computed once and for every node in a network is not possible.

Thus, we presented a supervised learning strategy for undervolting in WSNs that covers the environmental conditions and the individual peculiarities of each individual node. This allows us to sustain the reliability of the network while the energy consumption is significantly lowered and the lifetime is prolonged.

We showed that the presented concept is qualitatively workable and that a quantitative gain of up to 42% can be reached considering the energy demand of nodes. Moreover, we demonstrated that the system can successfully learn the impact of different environmental conditions and effectively adapt to new situations.

Future work will include the optimization of the SI utilization and further evaluations of IdealVolting in experimental WSN setups. To analyze the effects and impact of IdealVolting on challenging WSNs applications, we are currently working on an outdoor testbed to achieve both reliability and energy efficiency.

REFERENCES

- Gary Anthes. 2013. Inexact design: Beyond fault-tolerance. *Communications of the ACM* 56, 4 (2013), 18–20.
- Atmel. 2008. AVR998: Guide to IEC60730 Class B compliance with AVR microcontrollers. (2008). Retrieved from <http://www.atmel.com/images/doc7715.pdf>.
- Atmel. 2010. Datasheet: ATmega 1284/1284P, Rev. 8272A–AVR–01/10. Retrieved from http://www.atmel.com/dyn/resources/prod_documents/doc8272.pdf.
- Atmel. 2012. Datasheet: AVR Raven, Rev. 8117E–AVR–07/12. Retrieved from <http://www.atmel.com/Images/doc8117.pdf>.
- R. Jacob Baker. 2011. *CMOS: Circuit Design, Layout, and Simulation*. Vol. 18. John Wiley & Sons.
- Jalel Ben-Othman, Karim Bessaoud, Alain Bui, and Laurence Pilard. 2011. Self-stabilizing algorithm for energy saving in wireless sensor networks. In *Proceedings of the Symposium on Computers and Communications (ISCC'11)*. IEEE, 68–73.
- Jan Beutel, Bernhard Buchli, Federico Ferrari, Matthias Keller, Marco Zimmerling, and Lothar Thiele. 2011. X-sense: Sensing in extreme environments. In *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE'11)*. IEEE, 1–6.
- Carlo Alberto Boano, Hjalmar Wennerström, Marco Antonio Zúñiga, James Brown, Chamath Keppitiyagama, Felix Jonathan Oppermann, Utz Roedig, Lars-Åke Nordén, Thiemo Voigt, and Kay Römer. 2013. Hot packets: A systematic evaluation of the effect of temperature on low power wireless transceivers. In *Proceedings of the 5th Extreme Conference on Communication (ExtremeCom)*. ACM, 7–12.
- Carlo Alberto Boano, Marco Antonio Zuniga, James Brown, Utz Roedig, Chamath Keppitiyagama, and Kay Roemer. 2014. TempLab: A testbed infrastructure to study the impact of temperature on wireless sensor networks. In *Proceedings of the 13th International Conference on Information Processing in Sensor Networks (IPSN'14)*. ACM, 95–106.
- Bosch-Sensortec. 2009. Datasheet: BMP085 Digital Pressure Sensor, BST-BMP085-DS000-05, Rev. 1.2. (2009).
- Felix Büsching, Ulf Kulau, and Lars Wolf. 2012. Architecture and evaluation of INGA - an inexpensive node for general applications. In *IEEE Sensors 2012*. IEEE, 842–845.
- Haowen Chan and A. Perrig. 2003. Security and privacy in sensor networks. *Computer* 36, 10 (Oct 2003), 103–105. DOI:<http://dx.doi.org/10.1109/MC.2003.1236475>
- Dan Chen, Zhixin Liu, Lizhe Wang, Minggang Dou, Jingying Chen, and Hui Li. 2013. Natural disaster monitoring with wireless sensor networks: A case study of data-intensive applications upon low-cost scalable systems. *Mobile Networks and Applications* 18, 5 (2013), 651–663.
- Youngjin Cho, Younghyun Kim, and Naehyuck Chang. 2007. PVS: Passive voltage scaling for wireless sensor networks. In *Proceedings of the 2007 International Symposium on Low Power Electronics and Design (ISLPED'07)*. ACM, 135–140.
- Christopher Mims. 2011. MIT Technology Review: Why Your Battery Life is Terrible, in One Handy Chart. Retrieved from <http://www.technologyreview.com/view/424996/why-your-battery-life-is-terrible-in-one-handy-chart/>.

- Crossbow. 2012. Datasheet: MICAz, Rev. A. Retrieved from http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf.
- Shidhartha Das, Carlos Tokunaga, Sanjay Pant, Wei-Hsiang Ma, Sudharsen Kalaiselvan, Kevin Lai, David M. Bull, and David T. Blaauw. 2009. RazorII: In situ error detection and correction for PVT and SER tolerance. *IEEE Journal of Solid-State Circuits* 44, 1 (2009), 32–48.
- Michael Doering, Stephan Rottmann, and Lars Wolf. 2011. Design and implementation of a low-power energy management module with emergency reserve for solar powered DTN-nodes. In *Proceedings of the 3rd Extreme Conference of Communication (ExtremeCom 2011)*, Manaus, Brazil.
- Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. 2004. Contiki-a lightweight and flexible operating system for tiny networked sensors. In *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, 2004*. IEEE, 455–462.
- Stephan Friedrichs, Ulf Kulau, and Lars C. Wolf. 2014. Energy-efficient voltage scheduling of peripheral components on wireless sensor nodes. In *Proceedings of the IEEE International Conference on Communications (ICC'14) - Workshop on Energy Efficiency in Wireless Networks & Wireless Networks for Energy Efficiency (E2Nets)*. IEEE, Australia, Sydney.
- Ricardo Gonzalez, Benjamin M. Gordon, and Mark A. Horowitz. 1997. Supply and threshold voltage scaling for low power CMOS. *IEEE Journal of Solid-State Circuits* 32, 8 (1997), 1210–1216.
- Takahiro Hamachiyo, Yusuke Yokota, and Eiji Okubo. 2010. A cooperative power-saving technique using dvs and dms based on load prediction in sensor networks. In *Proceedings of the 4th International Conference on Sensor Technologies and Applications (SENSORCOMM'10)*. IEEE, 7–12.
- Leander B. Hoermann, Philipp M. Glatz, Christian Steger, and Reinhold Weiss. 2011. Energy efficient supply of WSN nodes using component-aware dynamic voltage scaling. *Proceedings of the 11th European Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless)*. 1–8.
- Hrishikesh Jayakumar, Arnab Raha, and Vijay Raghunathan. 2014. Hypnos: An ultra-low power sleep mode with SRAM data retention for embedded microcontrollers. In *Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis (CODES'14)*. ACM, New York, NY, Article 11, 10 pages. DOI : <http://dx.doi.org/10.1145/2656075.2656089>
- Rüdiger Kapitza, Johannes Behl, Christian Cachin, Tobias Distler, Simon Kuhnle, Seyed Vahid Mohammadi, Wolfgang Schröder-Preikschat, and Klaus Stengel. 2012. CheapBFT: Resource-efficient byzantine fault tolerance. In *Proceedings of the EuroSys 2012 Conference (EuroSys'12)*. ACM, 295–308.
- Nam Sung Kim, Todd Austin, David Baauw, Trevor Mudge, Krisztián Flautner, Jie S. Hu, Mary Jane Irwin, Mahmut Kandemir, and Vijaykrishnan Narayanan. 2003. Leakage current: Moore's law meets static power. *Computer* 36, 12 (2003), 68–75.
- N. Kimura and S. Latifi. 2005. A survey on data compression in wireless sensor networks. In *Proceedings of the International Conference on Information Technology: Coding and Computing, 2005 (ITCC'05)*. Vol. 2. 8–13 Vol. 2. DOI : <http://dx.doi.org/10.1109/ITCC.2005.43>
- Ulf Kulau, Felix Büsching, and Lars. 2013. A node's life: Increasing WSN lifetime by dynamic voltage scaling. In *Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE*. 241–248.
- Ulf Kulau, Felix Büsching, and Lars Wolf. 2015. Undervolting in WSNs: Theory and practice. *IEEE Internet of Things Journal* 2, 3 (June 2015), 190–198.
- Tadahiro Kuroda, Kojiro Suzuki, Shinji Mita, Tetsuya Fujita, Fumiyuki Yamane, Fumihiko Sano, Akihiko Chiba, Yoshinori Watanabe, Koji Matsuda, Takeo Maeda, and others. 1998. Variable supply-voltage scheme for low-power high-speed CMOS digital design. *IEEE Journal of Solid-State Circuits* 33, 3 (1998), 454–462.
- W. Louis Lee, Amitava Datta, and Rachel Cardell-Oliver. 2006. FlexiMAC: A flexible TDMA-based MAC protocol for fault-tolerant and energy-efficient wireless sensor networks. In *Proceedings of the 14th IEEE International Conference on Networks (ICON'06)*, Vol. 2. IEEE, 1–6.
- Young Choon Lee and Albert Y. Zomaya. 2010. Energy efficient resource allocation in large scale distributed systems. In *Proceedings of the 9th International Symposium on Distributed Computing and Applications to Business Engineering and Science (DCABES)*. IEEE, 580–583.
- Charles R. Lefurgy, Alan J. Drake, Michael S. Floyd, Malcolm S. Allen-Ware, Bishop Brock, Jose A. Tierno, John B. Carter, and Robert W. Berry. 2013. Active guardband management in power7+ to save energy and maintain reliability. *IEEE Micro* 33, 4 (2013), 35–45.
- Philip Levis, Sam Madden, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer, and others. 2005. TinyOS: An operating system for sensor networks. In *Ambient Intelligence*. Springer, 115–148.
- Ting Liu, Christopher M. Sadler, Pei Zhang, and Margaret Martonosi. 2004. Implementing software on resource-constrained mobile sensors: Experiences with impala and zebranet. In *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services (MobiSys'04)*. ACM, 256–269.

- Alexander W. Min, Ren Wang, James Tsai, and Tsung-Yuan Charlie Tai. 2014. Joint optimization of DVFS and low-power sleep-state selection for mobile platforms. In *IEEE International Conference on Communications (ICC)*, 2014. 3541–3546.
- Moteiv Corporation. 2006. Datasheet: Tmote Sky. Retrieved from http://www.snm.ethz.ch/pub/uploads/Projects/tmote_sky_datasheet.pdf.
- Johan Pouwelse, Koen Langendoen, and Henk Sips. 2001. Dynamic voltage scaling on a low-power microprocessor. In *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking (MobiCom'01)*. ACM, 251–259.
- Alireza Rohani and Hamid R. Zarandi. 2009. An analysis of fault effects and propagations in AVR microcontroller ATmega103 (L). In *Proceedings of the International Conference on Availability, Reliability and Security (ARES'09)*, 2009. IEEE, 166–172.
- Jukka Suomela. 2013. Survey of local algorithms. *ACM Computing Surveys (CSUR)* 45, 2 (2013), 24.
- Igor Talzi, Andreas Hasler, Stephan Gruber, and Christian Tschudin. 2007. PermaSense: Investigating permafrost with a WSN in the swiss alps. In *Proceedings of the 4th Workshop on Embedded Networked Sensors (EmNets'07)*. ACM, 8–12.
- Ulrich Tietze and Christoph Schenk. 2007. *Electronic Circuits: Handbook for Design and Application*. Springer-Verlag.
- Wei Tuning, Yu Sijia, and Wei Hailong. 2010. A dynamic voltage scaling algorithm for wireless sensor networks. In *Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE'10)*, Vol. 1. 554–557.
- Jun Wang, Ling Feng, Wenwei Xue, and Zhanjiang Song. 2011. A survey on energy-efficient data management. *ACM SIGMOD Record* 40, 2 (2011), 17–23.
- Tim Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander. 2012. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550 (Proposed Standard).

Received March 2015; revised October 2015; accepted January 2016

Energy-Efficient Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes

Stephan Friedrichs, Ulf Kulau and Lars Wolf

Technische Universität Braunschweig

Institute of Operating Systems and Computer Networks

Mühlenpfordtstraße 23

D-38106 Braunschweig

Email: {sfriedr, kulau, wolf}@ibr.cs.tu-bs.de

Abstract—Much effort has been put into optimizing the energy-efficiency of wireless sensor nodes, but existing work exclusively focuses on the transceiver and the processing unit. Nevertheless, the peripheral energy consumption may dominate that of the entire node. We introduce the concept of a dynamically scalable peripheral voltage supply: Even for peripheral devices, a lower voltage level leads to a lower energy consumption. Each peripheral requires a different minimum operating voltage, but switching the voltage level consumes energy as well. We combine theory and practice to present an algorithm weighing off the benefits of a downscaled voltage level against the switching overhead, i.e., for calculating an optimal peripheral voltage schedule. Our approach is capable of self-parameterization and has been implemented and tested on a prototype, saving up to 47 % of peripheral energy as compared to existing solutions.

I. INTRODUCTION

Compared to the growth of computational capabilities (Moore's law), the evolution of adequate mobile energy sources is limping. Hence, a significant amount of research aims at improving the lifetime of mobile applications like wireless sensor networks (WSNs). In this context, Dynamic Voltage Scaling (DVS) is a well-known technique, which depends on the ability of a system to adapt the voltage level to its workload. As the dynamic power consumption of CMOS gates shows a quadratic dependency on the voltage level, DVS helps to significantly improve the energy efficiency of micro-electronic systems [1], which has been exploited in several existing approaches [2], [3], [4], [5], leading to an increase of WSN lifetime. Nevertheless, existing work exclusively focuses on the micro-controller unit and does not consider peripherals like memory devices, sensors, or actuators. However, for some applications, a set of various sensors is needed [6]. Thus, the active power consumption of the sensing unit can easily exceed that of the micro-controller unit. For example, the current consumption of a typical gyroscope [7] is about three times higher than those of the msp430 [8] micro-controller.

Hence, the reduction of peripheral power consumption has a significant impact on the overall energy efficiency of a sensor node. Typical sensor node implementations lose much energy to a fixed, predefined, global peripheral voltage level. In this paper, we seize the concept of scalable peripheral voltage to optimize the overall power consumption and thus battery life.

For this purpose, we assume an a-priori known usage sequence of the peripheral devices on one sensor node, e.g. one that is periodically repeated, for instance: “read sensor 1, read

sensor 2, write to external memory, read sensor 1, ...”. Every peripheral device requires a minimum peripheral voltage level and a lower voltage implies lower energy consumption [5]; but switching the voltage level requires energy as well.

The current best practice is to statically set the voltage level to the maximum of the minimum required voltage level of all peripherals and to not change it dynamically.

Our novel approach addresses the trade-off between the energy gain of a temporarily reduced peripheral voltage and the corresponding switching overhead. We have developed and implemented an algorithm calculating optimal voltage schedules for arbitrary peripheral usage sequences. Yet, the overhead for calculating the schedule is negligible, as the optimal sequence of switching the voltage level is calculated only once during startup.

We address the rather practical issue of measuring the parameters of the involved peripheral components in our implementation as well. The developed prototype is able to self-parameterize every needed variable to run the algorithm. Hence, a fully self-optimizing and generic solution is presented.

The outline of the paper is as follows: Related work is discussed in the next section. Section III formalizes the peripheral voltage scheduling problem and introduces notation. Sections IV and V present our scheduling algorithm and its practical implementation. Our approach is evaluated in Section VI, while Section VII concludes this paper.

II. RELATED WORK

Typical DVS approaches focus on the processing unit only [9], [3]. However, our previous work [5] considers the effects of various voltage levels on the peripherals as well. Even peripherals like Micro-Electro-Mechanical Systems (MEMS), which mainly consist of non-CMOS parts [10], have a reduced energy consumption when exposed to a downscaled voltage level. This result leads to the idea of adapting the peripheral voltage level to the actual sensing requests instead of the system load.

[5] introduces an exclusive scalable voltage path for the peripheral components. Thus, together with a non-negligible switching overhead, the voltage level of the sensors and memory devices can be adapted. In this context, the scheduling problem is to find the best trade-off between reducing the

energy consumption by temporarily lowering the voltage level and the inferred switching costs.

Existing algorithms [11], [12] for voltage scheduling on low-performance sensor nodes are not sufficient for this purpose. In conclusion, they focus on the processing unit only or neglect the energy costs for adapting the voltage level.

To the best of our knowledge, there is no related work addressing the scheduling problem of finding an optimal voltage adaption scheme.

III. THE PERIPHERAL VOLTAGE SCHEDULING PROBLEM

We consider a wireless sensor node with a set of peripheral hardware, typically sensors and external memory. Each peripheral hardware device requires a minimum voltage to be properly operated. To the best of our knowledge, the best practice up to date is to statically configure the lowest peripheral voltage conform to all peripheral devices' voltage requirements. This can be very inefficient, because most hardware consumes more energy when exposed to higher voltage. In this work, we seek to exploit a sensor node's mechanism to dynamically switch the peripheral voltage.

The crux is that switching the voltage does not come for free. If it would, one could simply operate every peripheral device with its minimum required voltage. But switching the voltage consumes energy as well: The additional time interfacing a scalable voltage supply prolongs the duty-cycle of a processing unit, leading to a higher energy consumption. We present a measurement for the switching overhead in Section V.

In this work, we provide an algorithm for peripheral voltage scheduling that weighs off the energetic benefits of switching to a lower peripheral voltage against the switching overhead without violating the minimum voltage requirements of active hardware.

We achieve this by exploiting two observations due to preliminary measurements.

- 1) Most peripheral hardware devices can be safely undervolted while they are inactive.
- 2) The energy consumption of each peripheral device increases when the peripheral voltage is increased.

Both observations have been verified in experiments for a wide variety of hardware.

A. Peripheral Energy Consumption

Consider a sensor node with a set S of peripheral hardware devices. In order to assess the benefits of switching to a lower peripheral voltage before using $s \in S$, we need to know how much energy is consumed when querying s using the peripheral voltage v . We denote that relation by the function $e_s(v)$, which may be arbitrarily complicated, see Figure 1. $e_s(v)$ depends on the time t_s necessary to query s , the peripheral voltage v , and the accumulated current $I_s(v, t)$ flowing through s as well as through the inactive peripheral hardware $S \setminus \{s\}$:

$$e_s(v) = v \int_0^{t_s} I_s(v, t) dt \quad (1)$$

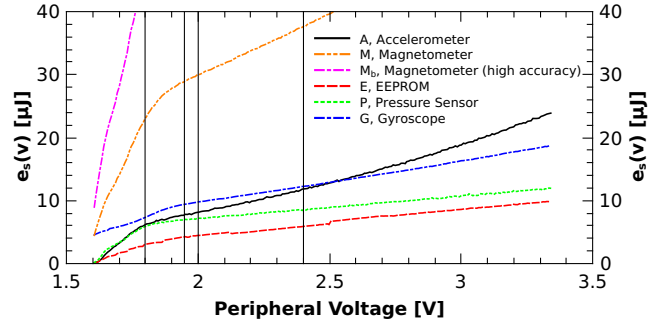


Figure 1. Active and passive energy consumption of the peripheral set of our prototype, compare Table I. The vertical bars at 1.800 V, 1.950 V, 2.000 V, and 2.400 V denote the minimum voltages of EEPROM and Pressure Sensor, Magnetometer, Accelerometer, and Gyroscope, respectively.

Moreover, some peripherals support various operating modes, resulting in a different energy consumption as well. As an example in Figure 1 the magnetometer can be used in standard mode M , and with higher accuracy M_b , leading to higher energy consumption $e_{M_b}(v)$. We simply model such varieties as different peripherals $s \in S$.

Instead of attempting to explicitly model $e_s(v)$, we consider it a black box and only assume that it is a monotonically increasing function. In the following subsection, we argue that only a very small set of values for v is of interest for our scheduling problem (only the minimal voltages for the peripherals). That allows us to (1) measure all relevant values of $e_s(v)$, of which there are no more than $|S|^2$, in a self-parameterization phase, compare Section V-A, (2) then run the scheduling algorithm using those values, and (3) run the schedule.

B. Notation

We formalize the peripheral voltage scheduling problem motivated above, using the following model.

Consider a sensor node with a set S of peripheral hardware devices. Each $s \in S$ has two attributes: (1) a minimum voltage $v_{\min}(s)$ required to properly operate s , and (2) the energy consumption $e_s(v)$ of all peripherals while only s is active, depending on the peripheral voltage v , see above. Throughout this work, we assume $e_s(v)$ to be a monotonically increasing function, i.e., that a reduction of the peripheral voltage never results in an increased energy consumption. For a constant amount C of energy, the *switching overhead*, the sensor node can adapt its peripheral voltage, refer to Section V for details.

The sensor node is presented a sequence of *queries* denoted by $[1, \dots, n]$ to fulfill its sensing and operation tasks. Query i operates the peripheral device $s_i \in S$, thus requiring a minimum peripheral voltage of $v_{\min}(s_i)$ and resulting in an energy consumption of $e_{s_i}(v)$. Note that $s_i = s_j$ for $i \neq j$ is quite common, because the typical application repeatedly operates a handful of sensors over a very long running time; in other words, we usually have $n \gg |S|$. A sequence of peripheral voltages $v(1), \dots, v(n)$ is called *voltage schedule* for the queries $[1, \dots, n]$, and we call a pair of query and voltage $(i, v(i))$ a *configuration*. $(i, v(i))$ is *feasible* if $v(i) \geq v_{\min}(s_i)$ and a voltage schedule is feasible if all its configurations are

feasible. The energy consumption E of a voltage schedule is:

$$E = \sum_{i=1}^n e_{s_i}(v(i)) + \sum_{i=2}^n \begin{cases} C & \text{if } v(i-1) \neq v(i), \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Our goal is to minimize E . We call a voltage schedule *optimal* if E is minimal. It follows from the monotonicity of $e_s(v)$ that an optimal schedule only uses $v(i) \in \{v_{\min}(s) \mid s \in S\} = \{V_1, \dots, V_m\}$ with $V_1 < \dots < V_m$. As motivated above, there are at most $|S|^2$ device-voltage combinations whose energy consumption can be measured in a self-parameterization phase, compare Section V-A.

Note that the current best practice, i.e., not adapting the peripheral voltage, is equivalent to a trivial schedule that simply keeps the minimum peripheral voltage high enough for all peripheral devices: $v(1) = \dots = v(n) = \max_{s \in S} v_{\min}(s)$.

IV. ALGORITHM

Let us, for the configuration (i, V_j) , determine the minimum amount of energy $E_{i,j}$ necessary to reach it using a feasible schedule $v(1), \dots, v(i)$ while assuming an infinite energy consumption for infeasible configurations. For the first query, we have:

$$E_{1,j} = \begin{cases} \infty & \text{if } V_j < v_{\min}(s_1), \\ e_{s_1}(V_j) & \text{otherwise.} \end{cases} \quad (3)$$

For $2 \leq i \leq n$, there is the mandatory energy consumption $e_{s_i}(V_j)$ to answer the query i itself, as well as the accumulated costs for traversing $i-1$ preceding configurations. There are two ways to reach the configuration (i, V_j) with an optimal energy consumption: Either the peripheral voltage from the previous query is kept, or it is changed. The former case yields an additional energy consumption of $E_{i-1,j}$. In the latter case we require the minimum amount of energy $\hat{E}_{i-1}$ to reach the cheapest feasible predecessor configuration and the additional costs C for switching the voltage, where $\hat{E}_{i-1} = E_{i-1,\hat{j}}$ with $\hat{j} := \arg \min_j E_{i-1,j}$. This yields, for $2 \leq i \leq n$:

$$E_{i,j} = \begin{cases} \infty & (i, V_j) \text{ is infeasible,} \\ e_{s_i}(V_j) + E_{i-1,j} & E_{i-1,j} < \hat{E}_{i-1} + C, \\ e_{s_i}(V_j) + \hat{E}_{i-1} + C & \text{otherwise.} \end{cases} \quad (4)$$

Equations (3) and (4) directly translate into Algorithm 1, see Figure 3. It uses dynamic programming to efficiently solve the recursion by determining $E_{i,\cdot}$ before $E_{i+1,\cdot}$; the optimal overall schedule is that ending in the configuration (n, V_j) , where $E_{n,j} = \hat{E}_n$ is minimal and the schedule $v(1), \dots, v(n)$ is determined by following the backward references stored in P , compare Figure 2.

A. Algorithm Extensions

Algorithm 1 processes finite query sequences of fixed order. In the remaining part of this section, we outline how to extend it to determine optimal schedules for periodically repeating queries, how the order of queries can be modified to find more energy-efficient schedules, and how to integrate global voltage limits into Algorithm 1.

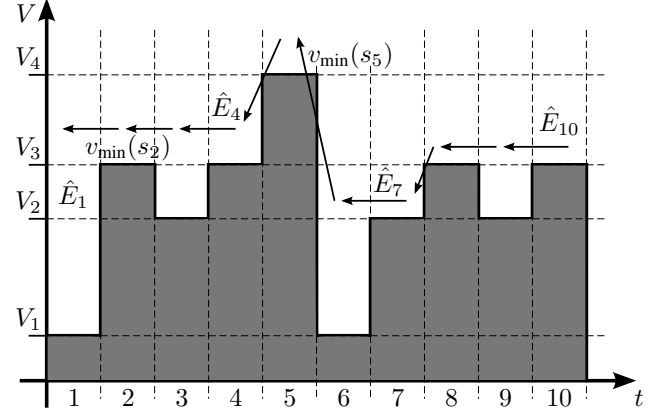


Figure 2. Using dynamic programming to determine an optimal schedule. Here: $(v(1), \dots, v(10)) = (V_3, V_3, V_3, V_3, V_4, V_2, V_2, V_3, V_3, V_3)$. Note that different queries with the same $v_{\min}(s_i)$, like 3, 7, and 9 may originate from the same sensor $s \in S$.

```

1: for  $j \leftarrow 1, \dots, m$  do
2:    $E_{1,j} \leftarrow \begin{cases} \infty & \text{if } V_j < v_{\min}(s_1) \\ e_{s_1}(V_j) & \text{otherwise} \end{cases}$ 
3:    $P_{1,j} \leftarrow \perp$ 
4: end for
5: for  $i \leftarrow 2, \dots, n$  do
6:    $\hat{j} \leftarrow \arg \min_{j=1, \dots, m} E_{i-1,j}$ 
7:   for  $j \leftarrow 1, \dots, m$  do
8:     if  $V_j < v_{\min}(s_i)$  then
9:        $E_{i,j} \leftarrow \infty$ 
10:       $P_{i,j} \leftarrow \perp$ 
11:     else if  $E_{i-1,j} < E_{i-1,\hat{j}} + C$  then is a maximum
12:       of
13:          $E_{i,j} \leftarrow e_{s_i}(V_j) + E_{i-1,j}$ 
14:          $P_{i,j} \leftarrow j$ 
15:       else
16:          $E_{i,j} \leftarrow e_{s_i}(V_j) + E_{i-1,\hat{j}} + C$ 
17:          $P_{i,j} \leftarrow \hat{j}$ 
18:       end if
19:     end for
20:   end for

```

Figure 3. Algorithm 1. Determining an optimal schedule using dynamic programming.

1) *Periodic Schedules:* The most common scenario is not to have a finite sequence of queries, but one that is repeated indefinitely, i.e., until battery depletion. So instead of a query sequence $[1, \dots, n]$, we have $[1, \dots, n, 1, \dots, n, \dots]$. Algorithm 1 can calculate an optimal schedule for that case as well: Suppose that query i has the strongest voltage requirement of all queries, i.e., it maximizes $v_{\min}(s_i) =: V$. Then any feasible schedule must enter the configuration (i, V) whenever i occurs in the query sequence. We use (i, V) as a fixed point. It is easy to check that an optimal voltage schedule is obtained by providing Algorithm 1 with the query sequence $[1, \dots, i, \dots, n, 1, \dots, i-1, i]$ and then periodically repeating the schedule for the underlined queries.

2) *Optimal Ordering of Queries:* Occasionally, queries do not have to be served in a specific order, e.g. in case of independent data aggregation. This scenario allows us to

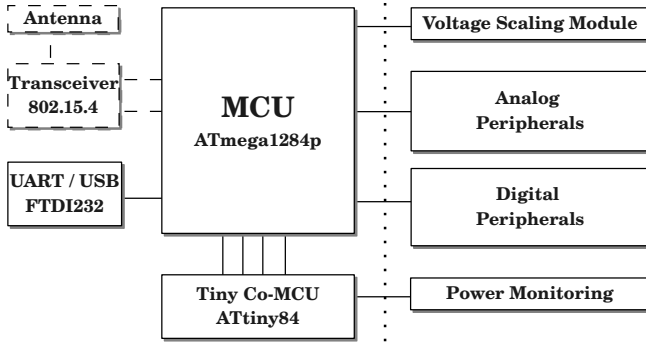


Figure 4. Block diagram of the prototype implementation.

reorder queries and the question arises, which ordering results in the cheapest voltage schedule. We claim that a cheapest optimal schedule is achieved when the queries $[1, \dots, n]$ are ordered such that, w.l.o.g., $v_{\min}(s_1) \geq \dots \geq v_{\min}(s_n)$ and outline a proof:

Suppose we know some ordering resulting in a cheapest schedule. Group its queries into segments $S_1, \dots, S_k$ of maximal length, such that all queries in segment S_i are served with the same voltage $V(S_i)$. For all $i \neq j$ we have $V(S_i) \neq V(S_j)$, because otherwise merging S_i and S_j would result in a cheaper schedule, contradicting optimality. Sorting queries within a segment doesn't change the energy consumption and neither does sorting the segments themselves. Note that every query j belongs to the segment S_i with minimum $V(S_i) \geq v_{\min}(s_j)$: If it didn't, the schedule would not be optimal due to the monotonicity of $e_{s_j}(v)$. Putting it all together proves that ordering the queries by descending minimum voltage yields a schedule with the same, i.e., optimal, costs.

3) *Voltage Limits*: There might be some lower bound $V_{\min}$ to the voltage of some inactive peripheral that is violated by $v_{\min}(s)$ for some other peripheral $s \in S$. Then Algorithm 1 is easily extended to only use voltages of $V_{\min}$ or higher: Recall that we only use $V_1 < \dots < V_m$ as voltage candidates and let $V_i = \max\{V_i \mid V_i < V_{\min}\}$. Now Algorithm 1 can just use the voltage candidates $V_{\min} < V_{i+1} < \dots < V_m$ instead. This even speeds up the process, especially the self-parameterization phase, because less voltages need to be considered, see Section V-A.

V. IMPLEMENTATION

The demand of energy-efficiency leads to limited resources like computational performance. Hence, typical wireless sensor nodes are equipped with ultra low-power (ULP) micro-controllers of small complexity. We use an 8-bit Atmel ATmega1284p micro-controller as processing unit, because it is wide-spread [13], [14], i.e., has practical relevance, and due to its scarce (CPU-) resources, a common limitation in WSNs. So it demonstrates that our approach is sufficiently lightweight for such requirements.

Related to the DVS capable node of [5], we borrow the design of its voltage-scaling module to achieve the ability of a software-defined voltage level for the peripherals: The voltage scaling module is connected to the processing unit via I2C-bus and provides a voltage level of $1.8 \text{ V} \leq v \leq 3.3 \text{ V}$ with

Table I. PROTOTYPE PERIPHERALS.

Peripheral s	Device	Description	$v_{\min}(s)$ [V]
A	ADXL345	Accelerometer	2.000
E	AT24C08C	EEPROM	1.800
P	BMP085	Pressure Sensor	1.800
G	L3G4200D	Gyroscope	2.400
M	MAG3110	Magnetometer	1.950
M_b	MAG3110	Magnetometer (high accuracy)	1.950

 Table II. ENERGY CONSUMPTION $e_s(v_{\min}(s'))$ PER QUERY FOR EACH SENSOR-VOLTAGE PAIR $(s, v_{\min}(s'))$.

Reference Voltage s'	$v_{\min}(s')$ [V]	Energy Consumption $e_s(v_{\min}(s'))$ [μJ] per Query				
		$s = E$	$s = P$	$s = M$	$s = A$	$s = G$
E, P	$V_1 = 1.800$	4.943	13.140	—	—	—
M	$V_2 = 1.950$	6.106	15.072	282.827	—	—
A	$V_3 = 2.000$	6.363	15.466	292.033	18.962	—
G	$V_4 = 2.400$	8.019	18.915	365.285	25.585	14.391

an 8-bit resolution. In this case, the overhead of switching to an arbitrary voltage level is $C \approx 7.76 \mu\text{J}$. This includes the increased active time of the MCU and the voltage scaling module's static power dissipation, refer to [5] for details.

Our prototype's sensing unit is divided into an analog and a digital section. The analog section offers the ability of connecting fully analog sensors to the ADC channels of the ATmega1284p, while the digital section includes the devices of Table I. All of them are connected via I2C bus.

The basic sensor set is inspired by motion-sensing applications like gait analysis [15] or flight control of quadcopters [6]. These applications observe various sensors, so our scheduling approach promises to increase their energy efficiency. Nevertheless, in order to expand the set of peripherals for further experiments, the I2C-bus is accessible through a pin header. Figure 4 shows a block diagram of the prototype.

A. Self-Parameterization of $e_s(v)$

As described above, in order to calculate an optimal schedule, we need information describing the overall peripheral energy consumption, which depends on the active peripheral device s and the peripheral voltage v . We denote that by the function $e_s(v)$, compare Equation (1). For this reason, we added a tiny co-micro-controller (co-MCU) to the prototype, which is able to concurrently sample the current consumption of the peripherals (a shunt is used in connection with current sense amplifiers) and to measure the time (the co-MCU can be triggered by the ATmega1284p via digital GPIOs). That design allows us to measure $e_s(v)$ for any given values of s and v .

Together with the predefined minimum voltage level $v_{\min}(s)$ of each peripheral s , the co-MCU is able to determine the energy consumption of all components as depicted in Equation (1): For each $(s, s') \in S \times S$, we measure $e_s(v_{\min}(s'))$, see Table II.

In connection with the subsequent calculation of the schedule, it is sufficient to use raw values instead of the physical units for time, current, and voltage. With regard to limited processing power of wireless sensor nodes, this simplification replaces expensive floating-point arithmetic with cheap integer operations.

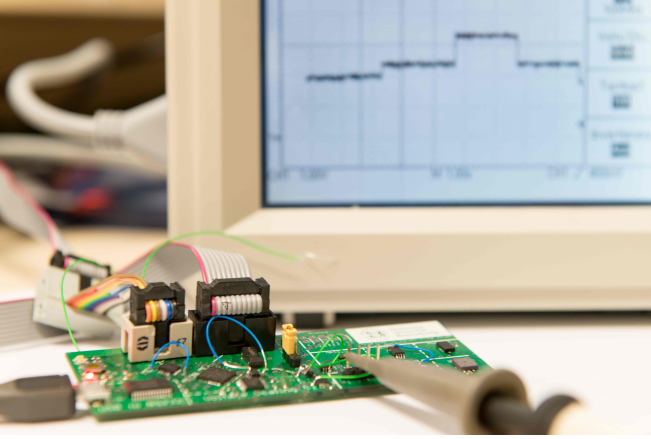


Figure 5. Test-setup of the evaluation.

VI. EVALUATION

Figure 5 shows the prototype implementation and the test-setup for evaluating our voltage scheduling concept.

We compare four peripheral voltage handling strategies, three heuristics and Algorithm 1:

- 1) **CONSTDEFAULT** is what happens when a sensor node has no mechanism to adapt the peripheral voltage. A constant peripheral voltage of 3.3 V is kept.
- 2) **CONSTMAXMIN** is the trivial strategy that uses the maximum minimum voltage, i.e., $\max_{s \in S} v_{\min}(s)$, for all queries. It never changes the peripheral voltage. Note that this strategy is, to our best knowledge, the current best practice. It is motivated by the scenario of a statically configured voltage that cannot be adjusted dynamically.
- 3) **ALWAYSSWITCH**, another trivial strategy, which *always*, i.e., for every query with a different voltage requirement than its predecessor, switches the voltage to its minimum requirement. It ignores the switching overhead.
- 4) **SCHEDULED** refers to a schedule computed by Algorithm 1, carefully weighing off the benefits of a lower peripheral voltage against the switching overhead.

In order to test the strategies, we devise various test cases, i.e., query sequences. Each of the above strategies is applied to every test case. We run the resulting voltage schedules 10 times each and measure their average energy consumption. Consult Section IV-A1 on how to use Algorithm 1 to efficiently deal with periodically repeating query sequences.

All test sequences, the resulting peripheral energy consumption of each strategy, as well as the percentage of energy saved by the **SCHEDULED** strategy as compared to the other three are presented in Table III. Which character refers to which sensor in a test sequence is specified in Table I, e.g., sequence 9 alternates between the gyroscope and the pressure sensor.

The average peripheral energy consumption for a query sequence is depicted in the center block of Table III. Clearly, the **SCHEDULED** strategy, i.e., Algorithm 1, never consumes more energy than any other strategy.

We present the percentage of energy saved by the **SCHEDULED** strategy compared to **CONSTDEFAULT**, **CONSTMAXMIN**, and **ALWAYSSWITCH** in the right block of Table III.

The improvement of **SCHEDULED** over **CONSTDEFAULT** is between 31.54 % (sequence 9) and 47.90 % (sequence 10), clearly a significant improvement.

Compared to **CONSTMAXMIN**, **SCHEDULED** saves up to 20.41 % (sequence 10), the majority of the improvements is comfortably above 17.00 %. As **CONSTMAXMIN** is a much better approach than **CONSTDEFAULT**, these still are outstanding improvements. On the other hand, the improvement regarding sequences 5 and 9 is zero within measuring noise. This is explained as follows: For sequences 5 and 9, the optimal schedule is to constantly keep the minimum maximum voltage. The **CONSTMAXMIN** strategy does that by chance, **SCHEDULED** does it, because it is optimal.

ALWAYSSWITCH performs better than **CONSTDEFAULT** and, in most cases, better than **CONSTMAXMIN**, but **SCHEDULED** still saves more energy. Despite the median improvement of **SCHEDULED** compared to **ALWAYSSWITCH** being between 1 % and 2 %, there are some cases where it is significant: For sequences 8, 5, and 9, **SCHEDULED** saves 8.45 %, 10.29 %, and 20.29 %, respectively. There are cases where there is little to no improvement of **SCHEDULED** compared to **ALWAYSSWITCH**. As above, this is owed to the fact that the **ALWAYSSWITCH** strategy happens to provide an optimal or near-optimal schedule.

Concluding, there are three important observations:

- 1) Much energy is saved by **SCHEDULED** compared to the three other strategies.
- 2) **SCHEDULED** never uses *more* energy than any of the other strategies. I.e., there is no situation where using it wastes energy. This is owed to the fact that Algorithm 1 finds an *optimal* schedule within our model, while any other strategy is bound to fail for some cases.
- 3) Occasionally, **SCHEDULED** is just as good as **CONSTMAXMIN** or **ALWAYSSWITCH**. These are the scenarios where the heuristics **CONSTMAXMIN** or **ALWAYSSWITCH** happen to find an optimal solution. This does not impair the quality of **SCHEDULED**, it rather demonstrates that **CONSTMAXMIN** and **ALWAYSSWITCH** are worthy competitors.

A final insight of our experiments is the following. In our prototype, the switching overhead is 7.76 μ J, which is quite high compared to peripheral queries requiring between 4.94 μ J and 365.29 μ J. It could be improved using more sophisticated hardware, compare Section V. But even under those adverse conditions, dynamically switching the voltage saves energy, even with a simple strategy as **ALWAYSSWITCH**, even more so with the optimal **SCHEDULED** strategy.

Our concluding observation is that the good experimental results stem from the fact that dynamic voltage scheduling can save much energy, that we use a good model, and that **SCHEDULED**, i.e., Algorithm 1, is not a heuristic but provides provably optimal voltage schedules.

Table III. IMPACT OF VOLTAGE SCHEDULING ON PERIPHERAL ENERGY-EFFICIENCY COMPARED TO CLASSICAL APPROACHES.

#	Query Sequence	Average Peripheral Energy Consumption [μ J]				Energy saved by SCHEDULED compared to		
		CONSTDEFAULT	CONSTMAXMIN	ALWAYS SWITCH	SCHEDULED	CONSTDEFAULT	CONSTMAXMIN	ALWAYS SWITCH
1	AEPGMAEPM	1321	864	723	716	45.80 %	17.13 %	0.97 %
2	GAMGAMGAM	1872	1215	1013	1008	46.15 %	17.04 %	0.49 %
3	GAMPE	663	432	357	352	46.91 %	18.52 %	1.40 %
4	GAMPEGAMPE	1325	863	725	712	46.26 %	17.50 %	1.79 %
5	GEPGEPGEP	183	123	136	122	33.33 %	0.81 %	10.29 %
6	PEMAG	663	432	358	352	46.91 %	18.52 %	1.68 %
7	PEMAGPEMAG	1328	864	725	712	46.39 %	17.59 %	1.79 %
8	GAPEGAPE	209	133	142	130	37.80 %	2.26 %	8.45 %
9	GPGPGPGPGP	241	165	207	165	31.54 %	0.00 %	20.29 %
10	PAMPE	666	436	356	347	47.90 %	20.41 %	2.53 %
11	APEGAME	720	465	398	385	46.53 %	17.20 %	3.27 %

A. Limitations

Our method guarantees that active peripherals never are undervolted, but relies on undervolting the inactive ones – otherwise CONSTMAXMIN would be optimal. So a potential drawback of our method is that the measurements of an active peripheral might be impaired by the fact that it has been undervolted in the recent past. We tried to find evidence for such an impairment, but none of our peripherals, see Table I, was in any way compromised by the undervolting.

Should, however, such an effect be observed in an application, e.g., that one sensor must not be undervolted below a certain limit, Section IV-A3 shows how to easily integrate such a limit into Algorithm 1.

VII. CONCLUSION

In this paper, we address sensor nodes capable of dynamically adjusting the peripheral voltage level. We optimize peripheral energy-efficiency by (1) carefully modeling and (2) optimally solving the following optimization problem: Peripheral hardware consumes more energy when exposed to a higher voltage level, each peripheral hardware device requires some minimum voltage to be properly operated, and adjusting the voltage level requires energy as well, the *switching overhead*. Our approach weighs off the benefits of a downscaled voltage level against the switching overhead.

We formulate a mathematical model of the above optimization problem and provide an algorithm finding optimal voltage schedules. The optimality of our schedules is verified with an implementation and a dedicated prototype sensor node.

In experiments we observe energy savings of up to 47 % as compared to a fixed voltage level. The experimental evaluation also confirms the practical performance of our approach, thus verifying both our model and our algorithm. Additionally, due to its optimality, our approach never requires more energy than its alternatives.

Further work includes the generalization of our approach to related scheduling problems, as well as the distribution of the problem: Neighboring sensor nodes may collaborate when observing the environment and thus avoid the switching overhead by sharing their sensor set.

REFERENCES

[1] U. Tietze and C. Schenk, *Electronic Circuits: Handbook for Design and Application*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[2] T. Hamachiyo, Y. Yokota, and E. Okubo, “A cooperative power-saving technique using dvs and dms based on load prediction in sensor networks,” in *Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on*, July 2010, pp. 7–12.

[3] L. B. Hoermann, P. M. Glatz, C. Steger, and R. Weiss, “Energy efficient supply of wsn nodes using component-aware dynamic voltage scaling,” *Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless), 11th European*, pp. 1–8, April 2011.

[4] W. Tuming, Y. Sijia, and W. Hailong, “A dynamic voltage scaling algorithm for wireless sensor networks,” in *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, vol. 1, Aug. 2010, pp. V1–554–V1–557.

[5] U. Kulau, F. Büsching, and L. C. Wolf, “A node’s life: Increasing WSN lifetime by dynamic voltage scaling,” in *The 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013 (IEEE DCoSS 2013)*, Cambridge, USA, May 2013. [Online]. Available: <http://www.ibr.cs.tu-bs.de/papers/kulau-dcross2013.pdf>

[6] M. Muller, S. Lupashin, and R. D’Andrea, “Quadcopter ball juggling,” in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011, pp. 5113–5120.

[7] ST Microelectronic, “L3g4200d mems motion sensor: ultra-stable three-axis digital output gyroscope,” 2010. [Online]. Available: <http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/CD00265057.pdf>

[8] Texas Instruments, “Msp430f15x, msp430f16x, msp430f161x mixed signal microcontroller,” 2002. [Online]. Available: <http://www.ti.com/lit/ds/symlink/msp430f1611.pdf>

[9] W. Dargie, “Dynamic power management in wireless sensor networks: State-of-the-art,” *Sensors Journal, IEEE*, vol. 12, no. 5, pp. 1518–1528, May 2012.

[10] F. Khoshnoud and C. de Silva, “Recent advances in mems sensor technology x2013; biomedical applications,” *Instrumentation Measurement Magazine, IEEE*, vol. 15, no. 1, pp. 8–14, 2012.

[11] T. Ishihara and H. Yasuura, “Voltage scheduling problem for dynamically variable voltage processors,” in *Low Power Electronics and Design, 1998. Proceedings. 1998 International Symposium on*, 1998, pp. 197–202.

[12] Y. Cho, Y. Kim, Y. Joo, K. Lee, and N. Chang, “Simultaneous optimization of battery-aware voltage regulator scheduling with dynamic voltage and frequency scaling,” in *Low Power Electronics and Design (ISLPED), 2008 ACM/IEEE International Symposium on*, 2008, pp. 309–314.

[13] Crossbow, “Micaz,” 2012. [Online]. Available: http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf

[14] Atmel, “Avr raven,” 2012. [Online]. Available: <http://www.atmel.com/Images/doc8117.pdf>

[15] J. Chen, K. Kwong, D. Chang, J. Luk, and R. Bajcsy, “Wearable sensors for reliable fall detection,” in *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, 2005, pp. 3551–3554.

Demo Abstract: Voltage Scheduling of Peripheral Components on Wireless Sensor Nodes

Ulf Kulau*, Stephan Friedrichs[†] and Lars Wolf*

*Technische Universität Braunschweig
Institute of Operating Systems and Computer Networks (IBR)
Mühlenpfordtstr. 23, 38106 Braunschweig, Germany
Email: [kulau,wolf]@ibr.cs.tu-bs.de

[†]Max Planck Institute for Informatics
Saarbrücken, Germany
Email: sfriedri@mpi-inf.mpg.de

Abstract—While existing work focuses on the transceiver and the processing unit to increase the energy efficiency of wireless sensor nodes, it is missed that peripheral energy consumption may dominate that of the entire node. Related to Dynamic Voltage Scaling (DVS), even peripherals’ energy efficiency benefit from a downscaled voltage level, but different peripherals require different minimum voltage levels. With this demo we combine theory and practice to present the implementation of an algorithm weighing off the benefits of a downscaled voltage level against the switching overhead, e.g. for calculating an optimal peripheral voltage schedule.

I. INTRODUCTION

As the dynamic power consumption of CMOS gates shows a quadratic dependency on the voltage level, DVS helps to significantly improve the energy efficiency of microelectronic systems [1]. Hence, several existing DVS approaches [2], [3] lead to an increase of WSN lifetime. Nevertheless, not only MCUs but also peripherals like memory devices, sensors, or actuators benefit from a downscaled voltage level.

Each peripheral hardware device requires a minimum voltage to be properly operated. The common practice is to statically configure the lowest peripheral voltage conform to all peripheral devices’ voltage requirements. This can be very inefficient, because most hardware consumes more energy when exposed to higher voltage. Hence, we seek to exploit a sensor node’s mechanism to dynamically switch the peripheral voltage.

The crux is that switching the voltage does not come for free. If it would, one could simply operate every peripheral device with its minimum required voltage. But switching the voltage consumes energy as well: The additional time interfacing a scalable voltage supply prolongs the duty-cycle of a processing unit, leading to a higher energy consumption.

The concept of incorporating switching cost among power different radio modes was discussed in [4], but is focussed on the radio transceiver only. In the following we outline the algorithm introduced in [5] which will be showed in our demo. This algorithm allows for peripheral voltage scheduling that weighs off the energetic benefits of switching to a lower peripheral voltage against the switching overhead without violating the minimum voltage requirements of active hardware.

Consider a sensor node with a set S of peripheral hardware devices. In order to assess the benefits of switching to a lower peripheral voltage before using $s \in S$, we need to know how much energy is consumed when querying s using the

peripheral voltage v . $e_s(v)$ depends on the time t_s necessary to query s , the peripheral voltage v , and the accumulated current $I_s(v, t)$ flowing through s as well as through the inactive peripheral hardware $S \setminus \{s\}$:

$$e_s(v) = v \int_0^{t_s} I_s(v, t) dt \quad (1)$$

Each $s \in S$ has two attributes: 1. a minimum voltage $v_{\min}(s)$ required to properly operate s , and 2. the energy consumption $e_s(v)$ of all peripherals while only s is active, depending on the peripheral voltage v , see above. Throughout this work, we assume $e_s(v)$ to be a monotonically increasing function, i.e., that a reduction of the peripheral voltage never results in an increased energy consumption. For a constant amount C of energy, the *switching overhead*, the sensor node can adapt its peripheral voltage. The sensor node is presented a sequence of queries denoted by $[1, \dots, n]$, so that the energy consumption E of a voltage schedule is given by:

$$E = \sum_{i=1}^n e_{s_i}(v(i)) + \sum_{i=2}^n \begin{cases} C & \text{if } v(i-1) \neq v(i), \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Our goal is to minimize E , so we call a voltage schedule *optimal* if E is minimal. It follows from the monotonicity of $e_s(v)$ that an optimal schedule only uses $v(i) \in \{v_{\min}(s) \mid s \in S\} = \{V_1, \dots, V_m\}$ with $V_1 < \dots < V_m$.

II. ALGORITHM

Let us, for a pair of query and voltage (i, V_j) , determine the minimum amount of energy $E_{i,j}$ necessary to reach it using a feasible schedule $v(1), \dots, v(i)$ while assuming an infinite energy consumption for infeasible configurations. For the first query, we have:

$$E_{1,j} = \begin{cases} \infty & \text{if } V_j < v_{\min}(s_1), \\ e_{s_1}(V_j) & \text{otherwise.} \end{cases} \quad (3)$$

For $2 \leq i \leq n$, there is the mandatory energy consumption $e_{s_i}(V_j)$ to answer the query i itself, as well as the accumulated costs for traversing $i-1$ preceding configurations. There are two ways to reach the configuration (i, V_j) with an optimal energy consumption: Either the peripheral voltage from the previous query is kept, or it is changed. The former case yields an additional energy consumption of $E_{i-1,j}$. In the latter case we require the minimum amount of energy $\hat{E}_{i-1}$ to reach the

cheapest feasible predecessor configuration and the additional costs C for switching the voltage, where $\hat{E}_{i-1} = E_{i-1,\hat{j}}$ with $\hat{j} := \arg \min_j E_{i-1,j}$. This yields, for $2 \leq i \leq n$:

$$E_{i,j} = \begin{cases} \infty & (i, V_j) \text{ is infeasible,} \\ e_{s_i}(V_j) + E_{i-1,j} & E_{i-1,j} < \hat{E}_{i-1} + C, \\ e_{s_i}(V_j) + \hat{E}_{i-1} + C & \text{otherwise.} \end{cases} \quad (4)$$

We use dynamic programming to efficiently solve the recursion by determining $E_{i,\cdot}$ before $E_{i+1,\cdot}$; the optimal overall schedule is that ending in the configuration (n, V_j) , where $E_{n,j} = \hat{E}_n$ is minimal. For a detailed description of the algorithm and some extensions please refer to [5].

III. IMPLEMENTATION

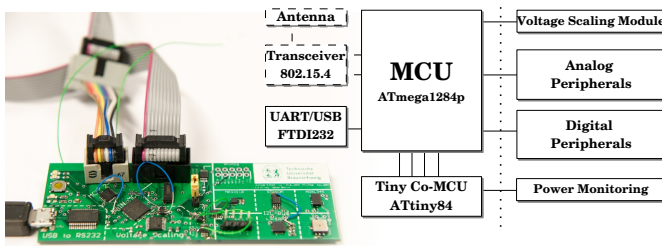


Figure 1. Block diagram and a picture of the actual prototype implementation.

Related to the INGA sensor node [6], we use an 8-bit Atmel ATmega1284p MCU as processing unit. Thus, the low computational capabilities of this MCU demonstrates that our approach is sufficiently lightweight for WSN requirements. Figure 1 shows a picture and a block diagram of the prototype. Compared to ordinary sensor nodes a voltage scaling module is connected to the processing unit via I2C-bus. This module provides a voltage level of $1.8\text{ V} \leq v \leq 3.3\text{ V}$ with an 8-bit resolution to the peripherals. In this case, the overhead of switching to an arbitrary voltage level is $C \approx 7.76\text{ }\mu\text{J}$. This includes the increased active time of the MCU and the voltage scaling module's static power dissipation, refer to [3], [5] for details.

Our prototype's sensing unit is divided into an analog and a digital section. The analog section offers the ability of connecting fully analog sensors to the ADC channels of the ATmega1284p, while the digital section includes the devices of Table I. All of them are connected via I2C bus. In order to

Table I. EQUIPPED PERIPHERALS FOR DEMONSTRATION.

Peripheral s	Device	Description	$v_{\min}(s)$ [V]
A	ADXL345	Accelerometer	2.000
E	AT24C08C	EEPROM	1.800
P	BMP085	Pressure Sensor	1.800
G	L3G4200D	Gyroscope	2.400
M	MAG3110	Magnetometer	1.950

calculate an optimal schedule, we need information describing the overall peripheral energy consumption. For this reason, we added a tiny co-MCU to the prototype, which is able to concurrently sample the current consumption of the peripherals (a shunt is used in connection with current sense amplifiers) and to measure the time (the co-MCU can be triggered by the ATmega1284p via digital GPIOs). Hence, $e_s(v)$ can be measured for any given values of s and v .

IV. EVALUATION

Although the demonstration will give the user already the chance to optimize custom schedules, the following table depicts some exemplary schedules to show the general benefit of our approach. The energy savings are compared against three classical strategies. CONSTDEFAULT is what happens when a sensor node has no mechanism to adapt the peripheral voltage. A constant peripheral voltage of 3.3 V is kept. CONSTMAXMIN is the trivial strategy that uses the maximum minimum voltage, i. e., $\max_{s \in S} v_{\min}(s)$, for all queries. ALWAYS SWITCH always switches the voltage to its minimum requirement. It ignores the switching overhead.

Table II. SAMPLE SCHEDULES TO SHOW THE BENEFIT OF PERIPHERALS' VOLTAGE SCHEDULING COMPARED TO NAIVE APPROACHES.

Query Sequence	Energy saved by SCHEDULED compared to		
	CONSTDEFAULT	CONSTMAXMIN	ALWAYS SWITCH
AEPGMAEPM	45.80 %	17.13 %	0.97 %
GAMGAMGAM	46.15 %	17.04 %	0.49 %
GAMPE	46.91 %	18.52 %	1.40 %
GPGPGPGPGP	31.54 %	0.00 %	20.29 %
PAMPE	47.90 %	20.41 %	2.53 %

V. DEMONSTRATION

With a GUI a custom query of peripherals (cf. Table I) can be created. This query is transferred to the prototype board via USB. As the implementation follows a fully self-optimizing approach, the prototype board firstly self-parametrizes the energy functions $e_s(v)$ of involved peripherals. Afterwards, the board executes the optimization algorithm to get the optimal voltage schedule for the given query. Finally the query is processed while the optimal schedule is compared against the trivial voltage strategies as described in the previous section. For this purpose, the second tiny MCU samples the current consumption of CONSTDEFAULT, CONSTMAXMIN, ALWAYS SWITCH and of course SCHEDULED. The results are send back to the PC, where the GUI displays an oscilloscope of the current consumptions as well as an analysis of the saved energy.

REFERENCES

- [1] U. Tietze and C. Schenk, *Electronic Circuits: Handbook for Design and Application*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.
- [2] L.B. Hoermann, P.M. Glatz, C. Steger and R. Weiss, "Energy Efficient Supply of WSN Nodes using Component-Aware Dynamic Voltage Scaling," *Wireless Conference 2011 - Sustainable Wireless Technologies (European Wireless)*, 11th European, pp. 1–8, april 2011.
- [3] U. Kulau, F. Büsching, and L. C. Wolf, "A node's life: Increasing WSN lifetime by dynamic voltage scaling," in *The 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013 (IEEE DCoSS 2013)*, Cambridge, USA, May 2013.
- [4] R. Jurdak, A. G. Ruzzelli, and G. M. P. O'Hare, "Radio sleep mode optimization in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, 2010.
- [5] S. Friedrichs, U. Kulau and L. Wolf, "Energy-efficient voltage scheduling of peripheral components on wireless sensor nodes," in *Communications Workshops (ICC), 2014 IEEE International Conference on*, June 2014, pp. 860–865.
- [6] F. Büsching, U. Kulau, and L. Wolf, "Architecture and Evaluation of INGA - An Inexpensive Node for General Applications," in *Sensors, 2012 IEEE*. Taipei, Taiwan: IEEE, oct. 2012, pp. 842–845.

Paint it Black – Increase WSN Energy Efficiency with the Right Housing

Ulf Kulau, Sebastian Schildt, Stephan Rottmann and Lars Wolf

Institute for Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
[kulau|schildt|rottman|wolf]@ibr.cs.tu-bs.de

ABSTRACT

It is a well-known fact that Wireless Sensor Networks (WSNs) that are exposed to real environmental conditions suffer from harsh temperatures. Yet, the temperature does not only have negative impact as the energy efficiency of processing units benefits from higher temperatures. The minimal voltage for correct operation of CMOS circuits is bounded by the temperature. Thus, temperature-dependent undervolting schemes for WSN nodes have been proposed in the past to extend the network lifetime. However, not much thought has been given into directly influencing the most relevant factor: Temperature. In this work we look at the influence of various WSN housings onto the temperature profile of WSN nodes and quantify the energy saving potential of choosing the right housing.

1. INTRODUCTION

In the past the WSN community gained a lot of experience with the challenges of using WSNs for outdoor applications e.g. [1, 7]. One aspect is that nodes often are exposed to rough environmental conditions, particularly temperatures [11, 2]. Unfortunately, the nodes suffer from such extreme temperatures as the efficiency of IEEE 802.15.4 transceivers decreases with growing temperatures and thus, the reliability of transmissions decreases [9, 3]. However, an increased temperature does not only have negative effects on the operation of a WSN as the temperature has a significant impact on the energy efficiency of the sensor node.

Safe operating voltage for Complementary Metal-Oxide Semiconductor (CMOS) circuits depends on the environment temperature: With higher temperatures, and thus higher electron mobility, the minimum threshold voltage of CMOS gates decreases [6]. This has led to approaches undervolting components below their specified minimum voltage in order to save energy for battery powered devices like wireless sensor nodes [5].

As a micro controller unit (MCU), which is commonly used on wireless sensor nodes, is specified to cover a wide

temperature range, even under normal conditions (room temperatures) the nominal voltage level can be undershot significantly without any negative impact on reliability. The voltage level itself has a quadratic impact on the dynamic power consumption of CMOS parts [10], thus for normal environmental conditions energy savings up to 42% at the processing unit are possible. Therefore, the occurrence of higher temperatures will lead to even lower acceptable minimum voltage levels where an MCU can be operated reliably [5]. Hence, temperature becomes an influencing factor for energy consumption and influencing a nodes temperature is one way to increase energy efficiency. Obviously, actively heating a node is not sustainable, however housings of different materials with different heat capacities and thermal conductivities can change the temperature a device experiences.

In this paper we will first present an energy model and the undervolting capacities of the INGA WSN node, which have been determined by controlled experiments. We will then present logged temperature data over three months using different WSN housings. Together with the energy model this allows us to calculate the energy needed to operate INGA in each of the housings and locations to see if any of them offers an advantage over others. Finally we wrap up by pointing out limits of the presented approach and discuss how the findings in this paper can help to optimize the energy efficiency of a WSN application.

2. TEMPERATURE VS. ENERGY

The current consumption of node components such as the processing unit and the transceiver unit are temperature dependent due to physical effects, which is well documented in the corresponding datasheets. Considering undervolting the connection between temperature and energy efficiency can be strengthened even more. As mentioned above, in our previous work [5] we have shown that MCUs can be operated below their recommended voltage levels, as the threshold voltage of CMOS gates is temperature dependent. Thus, the absolute minimum voltage level of an MCU is given as a function of the surrounding temperature ($V_{uv}(T)$). We took $i = [0..14]$ undervolting-capable INGA [5, 4] sensor nodes and measured their absolute minimum voltage levels for reliable operation at different temperatures to determine $V_{uv_i}(T)$. In this case the nominal minimum voltage for the MCU on INGA is 2.4V@8MHz. Due to imperfections and fluctuations in CMOS production the minimum acceptable voltage of individual chips differs, so that every sensor node has its own individual characteristic curve.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

RealWSN'15, November 1, 2015, Seoul, South Korea..

© 2015 ACM. ISBN 978-1-4503-3840-0/15/11 ...\$15.00.

DOI: <http://dx.doi.org/10.1145/2820990.2820992>.

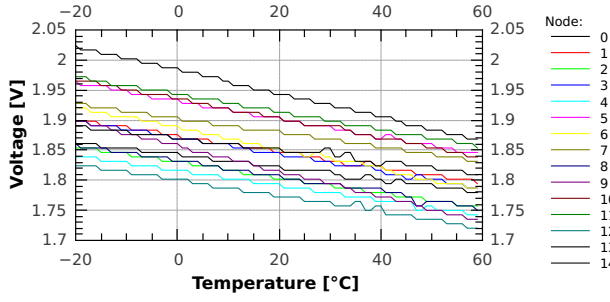


Figure 1: Characteristic curves of undervolted ATmega1284p MCUs on the INGA sensor node.

Figure 1 shows the result of this evaluation. All nodes are able to operate significantly below their recommended voltage level. Moreover, higher temperatures allow nodes to run at lower voltage levels. To get a comprehensive energy model for the INGA node we measured the current consumption of sensor nodes with changing voltage levels at different temperatures in a climatic chamber. For this evaluation the consumption of a whole sensor node with all components is considered while the transceiver unit was set to sleep state, so that the main current consumption depends on the MCU only. The considered temperature range reaches from $-15^{\circ}\text{C} \leq T \leq 60^{\circ}\text{C}$. Figure 2 shows the result of this measurement. When considering both, voltage and current consumption. Overall, higher temperatures leads to increased energy efficiency.

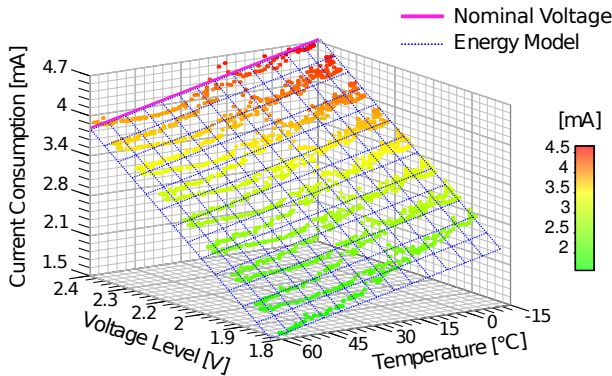


Figure 2: Measurement of the current consumption of the INGA wireless sensor node at different voltage levels and temperatures (transceiver unit in sleep mode)

Based on these results we derived an energy model by using regression calculation. With an average deviation of 1.97% from the measured values, the model of the current consumption as a function of the temperature T and the voltage level v can be given as follows:

$$\forall \begin{cases} 1.6\text{V} \leq v \leq 2.4\text{V} \\ -15^{\circ}\text{C} \leq T \leq 60^{\circ}\text{C} \end{cases} : I_{cc}(v, T) = p + s \cdot T + t \cdot v \quad (1)$$

with $p = -4.558872[\text{mA}]$, $s = -11.97692[\mu\text{AK}^{-1}]$ and $t = 3.770542[\text{mAV}^{-1}]$.

Combined with the individual characteristics of each node as shown in Figure 1, which can be represented by a linear



(a) Direct sunlight

(b) Shadowed location

Figure 3: Experimental Setup

approximation of the form $V_{uvi}(T) = a_i + b_i \cdot T$, the power consumption of an undervolted sensor node can be calculated as a function of the temperature only.

3. EXPERIMENT

To quantify the potential energy savings from optimal temperature profiles we performed a long term measurement of the temperatures experienced by sensor nodes in 4 different housings and 2 different locations. As a baseline we measured the air temperature, which an unpackaged sensor node would experience during its probably short life. In addition to that we measured the temperature inside the normal INGA case, which is an ABS printed plastic enclosure. Further, a transparent glass enclosure has been used, to determine whether the expected green-house effect will have a positive impact. Finally, one housing was a stone normally used in storage heaters. The idea is that the large thermal capacity of the stone might be helpful by sustaining high temperatures for a longer time. Each housing as well as the air temperature has been measured under direct sunlight as well as in the shadow.

The measurement setup can be seen in Figure 3. We ran the measurements for approximately 3 months from May to July in Northern Germany. Every 10 seconds we took a temperature sample for each housing. Current measurement status and the opportunity of requesting temperature traces are available for the community¹.

3.1 Results

With the measured temperature charts for each day, the energy model from Equation 1 and the undervolting characteristic of a node as shown in Figure 1 it is possible to calculate how much energy a node consumes at a given location in a given housing. For the sake of simplicity for the following analysis we used a generic node characteristic which has been calculated as average from the 15 nodes examined in Figure 1.

Figure 4 shows the measured temperature. The detail plot on the right shows the temperature data from June 4th to June 7th. It is easy to see the daily pattern. During night, after everything cools down, there is virtually no difference between the different housing types. During day time the standard case and the glass reach the highest temperatures and thus seem to offer the best undervolting potential. It can be seen that the stone dampens temperature changes due to its higher heat capacity, but it does not seem to be able to hold the temperature significantly longer than the other housings. Table 1 concludes the measurement and

¹www.ibr.cs.tu-bs.de/users/kulau/PotatoNet-heat.html

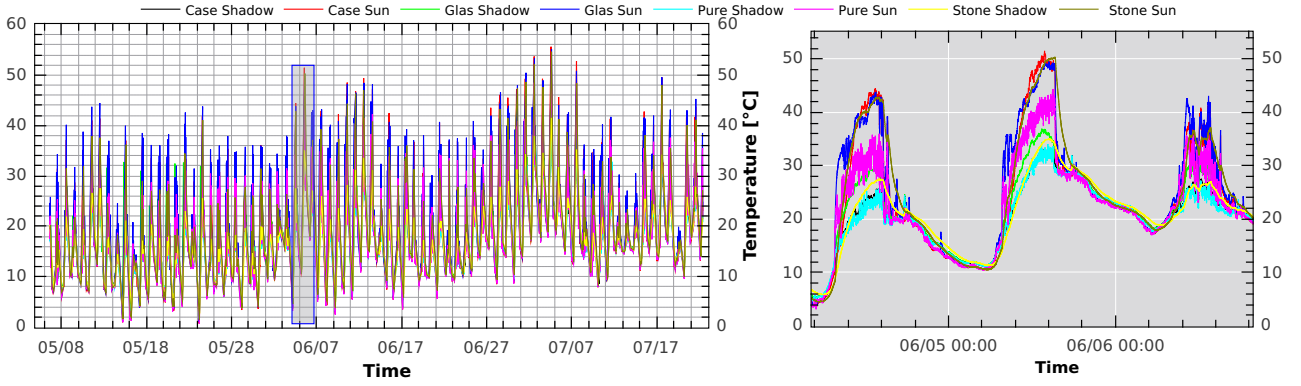


Figure 4: Longterm temperature measurement of different housings and locations.

			Case ●	Case ☆	Glas ●	Glas ☆	Pure ●	Pure ☆	Stone ●	Stone ☆	
Absolute	Day	$\bar{T}[^{\circ}C]$	17.663	20.719	18.949	21.803	17.231	19.116	17.915	21.280	☆ - Sun
		$\sigma_T[^{\circ}C]$	4.388	5.629	4.436	5.480	4.460	5.072	4.518	5.733	● - Shadow
	Night	$\bar{T}[^{\circ}C]$	12.397	11.533	12.512	11.683	12.277	11.650	13.085	12.447	σ_x - Std. Deviation
		$\sigma_T[^{\circ}C]$	3.871	3.917	3.871	3.925	3.871	3.908	3.852	4.056	$\bar{x}$ - Mean
Percent	Day	$\Delta\bar{T}[\%]$	2.745	20.077	10.696	27.083	0.000	10.828	4.131	23.464	■ - Baseline
		$\sigma_{\Delta T}[\%]$	1.359	8.428	5.888	10.648	0.000	4.943	2.211	10.093	
	Night	$\Delta\bar{T}[\%]$	1.097	-6.889	2.115	-5.557	0.000	-5.799	7.514	1.051	
		$\sigma_D[^{\circ}C]$	1.037	3.868	1.137	3.208	0.000	3.274	5.588	3.727	

Table 1: Summary of the average temperature and the percent deviation from the baseline (Pure node deployed in shadow). During daytime the average temperatures of the housings differ a lot with the node in the glass showing the highest average temperature. During nighttime the differences between the housings is minor.

shows the averaged temperature with corresponding standard deviation of each housing for day- and night-time. For this evaluation the day-time is defined from dawn to dusk and recalculated for each day. Nevertheless, it should be mentioned, that the day-time does not equal the time of direct sunlight, as surrounding circumstances like trees and houses shadowed the experimental area well ahead of dusk. In addition Table 1 depicts the deviation of the temperature compared to the baseline, which is defined by the node that is deployed unpacked in the shadow (Pure Shadow).

We have chosen a very warm day (July 3rd) and a rather cold day (May 16th) within the measurement period and calculated how much energy a node would consume in each housing during day-time. The results can be seen in Table 2. In general it is clear that undervolting can achieve huge efficiency gains reducing energy consumption by a factor of up to 2.7. The data also clearly shows that higher temperatures are preferred, as they enable more aggressive undervolting, therefore energy efficiency is generally better in hot days. Due to the dependence between current consumption and temperature this is even true in the case when no undervolting is used. In any case a housing always improves things. On the hot day the stone housing with direct exposure to the sun gains another 4.0% energy efficiency compared to just putting the node in the shadow. For the cold day the glass exposed to the sun beats a shadowy place by 4.4%

To show the change of energy efficiency over a day Figure 5 outlines the percent gain of power dissipation compared to the baseline housing (Pure Shadow) for an exemplary day (July 17th). It can be seen that the direct sunlight deployment leads to up to $\approx 8\%$ more efficient processing

just because of the right housing. All in all between the maximum reached temperature of $55.62^{\circ}C$ (Case Sun) and the minimum temperature of $0.69^{\circ}C$ (Pure Sun) the savings of power dissipation amounts to 40.5% when undervolting is applied.

4. DISCUSSION AND LIMITATIONS

We have seen that higher temperatures offer more undervolting potential and thus can improve energy efficiency. The evaluation focuses on processing unit energy consumption. When communication is considered the effects of temperature and potentially undervolting on the transceiver or used sensors need to be considered. As a rule of thumb undervolting a transceiver does not gain much, because it will always need to put a certain amount of energy in the air. However, the already mentioned negative impact on the transceiver's reliability at increased temperatures could lead to energy consuming retransmissions, which have to be taken into account.

There is also the question, whether higher temperatures will affect the aging of the electronic components significantly [8]. We assume with the tested housings, considering the process structures and tolerances of WSN node electronics this is not a problem, after all we did not reach higher temperatures than can be expected for a WSN node deployed outdoors, we just try to influence circumstances so those temperatures can be reached faster, more often and for a longer time. Of course, when trying to build special housings acting as a solar oven to further increase energy efficiency, this might become an issue.

It can be argued, that the differences between housings are

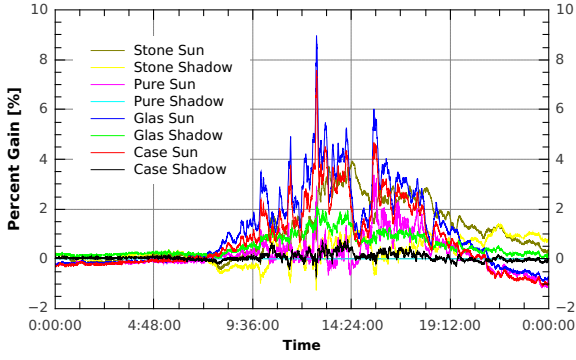


Figure 5: Gain in energy efficiency due to undervolting and housing effects compared to baseline (Pure Shadow) over the course of an exemplary day.

	Undervolted		Nominal Voltage	
	hot day	cold day	hot day	cold day
Case ●	323.76	376.83	852.46	902.86
Case ☆	313.07	374.58	843.63	901.05
Glas ●	320.99	374.26	850.17	900.90
Glas ☆	312.73	372.04	843.27	899.10
Pure ●	325.06	377.99	853.54	903.80
Pure ☆	318.94	376.02	848.44	902.32
Stone ●	322.68	377.95	851.67	903.87
Stone ☆	312.05	374.87	842.79	901.38

Table 2: Energy consumption of the nodes from dawn to dusk in [J]. For hot days the black stone case enables the highest energy savings.

not extraordinarily large, but keep in mind that you can get these optimizations for free: Just choose your sensor housing accordingly. At the very least just paint it black. Also, making sure your nodes are exposed to the sun, if possible, helps. This will also help in other ways: Your battery will be warmer too, which enables most battery technologies to deliver more capacity, thus the lifetime of a node should grow even more than suggested by the results presented in this paper. Vice versa, a damping housing, e.g. the Stone, could help to mitigate the adverse effects on the transceiver unit.

5. CONCLUSION

In this paper we firstly showed, that the temperature has a significant impact on the energy efficiency of wireless sensor nodes' processing units. When using undervolting this effect can be enhanced so that up to 40.5% more efficient processing was reached just because of the environmental temperature.

Hence, we evaluated different types of housing and deployment locations (direct sunlight and shadowed) to analyse the influence of their thermal characteristics. Together with a proposed energy model of a sensor node's processing unit, that also takes the temperature into account, we were able to determine the power dissipation under various environmental conditions.

A long term study was performed to show the benefit when choosing the right housing for an outdoor WSN deployment. We showed that the total amount of energy can be reduced

and that there is a substantial variation over a single day. Nevertheless, we are also aware of the negative impact of high temperatures on the transceiver unit but at least we have a crux: High temperatures have a negative impact on the transceiver units but lead to a more energy efficient processing on the wireless sensor node.

6. REFERENCES

- [1] J. Beutel, B. Buchli, F. Ferrari, M. Keller, M. Zimmerling, and L. Thiele. X-sense: Sensing in extreme environments. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2011, pages 1–6. IEEE, 2011.
- [2] C. A. Boano, H. Wennerström, M. A. Zúñiga, J. Brown, C. Keppitiyagama, F. J. Oppermann, U. Roedig, L.-Å. Nordén, T. Voigt, and K. Römer. Hot Packets: A Systematic Evaluation of the Effect of Temperature on Low Power Wireless Transceivers. In *Proceedings of the 5th Extreme Conference on Communication (ExtremeCom)*, pages 7–12. ACM, August 2013.
- [3] C. A. Boano, M. A. Zuniga, J. Brown, U. Roedig, C. Keppitiyagama, and K. Roemer. TempLab: A Testbed Infrastructure to Study the Impact of Temperature on Wireless Sensor Networks. In *Proceedings of the 13th International Conference on Information Processing in Sensor Networks (IPSN '14)*, pages 95–106. ACM, April 2014.
- [4] F. Büsching, U. Kulau, and L. Wolf. Architecture and Evaluation of INGA - An Inexpensive Node for General Applications. In *IEEE Sensors 2012*, pages 842–845. IEEE, October 2012.
- [5] U. Kulau, F. Busching, and L. Wolf. Undervolting in wsns: Theory and practice. *Internet of Things Journal, IEEE*, 2(3):190–198, June 2015.
- [6] R. Kumar and V. Kursun. Temperature-adaptive energy reduction for ultra-low power-supply-voltage subthreshold logic circuits. In *Electronics, Circuits and Systems, 2007. ICECS 2007. 14th IEEE International Conference on*, pages 1280–1283, Dec 2007.
- [7] M. Navarro, T. W. Davis, Y. Liang, and X. Liang. A study of long-term wsn deployment for environmental monitoring. In *24th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, Sept 2013.
- [8] F. Reynolds. Thermally accelerated aging of semiconductor components. *Proceedings of the IEEE*, 62(2):212–222, Feb.1974.
- [9] F. Schmidt, M. Ceriotti, N. Hauser, and K. Wehrle. If You Can't Take The Heat: Temperature Effects On Low-Power Wireless Networks And How To Mitigate Them. In *12th European Conference on Wireless Sensor Networks (EWSN 2015)*, February 2015.
- [10] U. Tietze and C. Schenk. *Electronic Circuits: Handbook for Design and Application*. Springer-Verlag, 2007.
- [11] H. Wennerstrom, F. Hermans, O. Rensfelt, C. Rohner, and L.-A. Norden. A long-term study of correlations between meteorological conditions and 802.15. 4 link performance. In *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2013 10th Annual IEEE Communications Society Conference on*, pages 221–229. IEEE, 2013.

Demo: PotatoNet – Robust Outdoor Testbed for WSNs

Experiment like on your desk. Outside.

Ulf Kulau, Sebastian Schildt, Stephan Rottmann, Björn Gernert and Lars Wolf
Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
[kulau|schildt|rottmann|gernert|wolf]@ibr.cs.tu-bs.de

ABSTRACT

We present PotatoNet, an outdoor testbed for Wireless Sensor Networks (WSNs). Its primary focus is robustness, reliability and flexibility. PotatoNet is designed to operate without on-site maintenance for extended periods of time. It can withstand heat, dust and rain and has already been tested running outside for several months.

1. MOTIVATION

Many testbeds for WSNs have been proposed with the goal of enabling users to test new methods and algorithms in a realistic environment. However, many of these testbeds have only been developed for indoor use [8, 4, 7, 3]. Nodes in an outdoor testbed need to be weather proof, and withstand the environmental conditions specific to their application. However, because outdoor testbeds might be installed in remote areas, software and architecture are equally important: When it is not possible to perform regular maintenance, and any hard- or software failure can lead to a prolonged downtime, robustness and reliability become the main focus of hard- and software design. Smart Farming provides an especially challenging outdoor environment. Nodes are deployed in the open field and are not protected from direct sunlight. Additionally, they need to be permanently water-proof as they are exposed to rain and irrigation (see Figure 1).

Nevertheless, there are also some outdoor testbeds deployed previously. Indeed, [5] reports the experience of running a testbed on a potato field and highlighted many challenges encountered when operating a testbed outdoors. Due to the complexities involved, many outdoor testbeds are rather special purposes WSN deployments [1, 6] and do not offer the same flexibility and generic usability as indoor testbeds. With the PotatoNet we combine the robustness needed for an outdoor testbed with maximum flexibility.

2. IMPLEMENTATION

The overall architecture of PotatoNet is shown in Figure 2a. The basic architecture consists of a central box providing management functionalities and distributing power to

all nodes. For cost-efficiency reasons the nodes are designed to be cheap, while complexity and redundancy features are moved to the central component. We are using Ethernet with passive PoE to distribute power and managing field nodes. Several features in the architecture lead to PotatoNet's high robustness and resilience against failures:

Every WSN node is paired with a small embedded Linux board running OpenWRT Linux. The separate programming platform is able to flash the sensor node using ISP. PotatoNet offers two separate Internet uplinks. The main uplink is using Ethernet and can be connected to a standard SOHO router. The management server in the central box includes a cellular data card. Power to individual field nodes can be switched on and off via software. In case a short circuit triggers a reboot, all field nodes are powered off when the system restarts. This allows switching them on one by one identifying the faulty one.

We use a INGA [2] WSN node (DVS-capable Version 1.6.1) in PotatoNet's field nodes. The programming platform inside each field node is based on the WRTNode¹. This USD 25 platform is based on a 680 MHz MIPS CPU (Mediatek MT7620N) with 64 MiB RAM running OpenWRT².

We developed a special companion board that provides an efficient DC-DC converter to supply the WRTnode and INGA from the PoE supply voltage (see Figure 2d).

All components of the central box are mounted inside an 73 liter (600 x 400 x 410 mm³) aluminum box. We use rugged industrial connectors for Ethernet and Power to protect against dust and humidity (see Figure 2b). For the enclosure of the field nodes we use standard PVC-U tubing with 160 mm diameter. The enclosure consists of a drainage double socket closed with two socket plugs (see Figure 2c). As only one (or two for cascading) Ethernet cable needs to go into enclosure we used simple cable glands to connect the Ethernet cable and seal the enclosure.

3. CONCLUSIONS

We presented the PotatoNet outdoor WSN testbed. PotatoNet is a generic testbed for outdoor applications of WSN nodes with a strong focus on robustness and availability. It is currently deployed in a Smart Farming scenario and will be successively extended with more field nodes in the future.

Software and custom-designed hardware schematics will be made available at the PotatoNet project page³.

¹<http://wrtnode.com>

²<http://openwrt.org/>

³<https://www.ibr.cs.tu-bs.de/projects/potatonet/>

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author(s). Copyright is held by the owner/author(s).

CHANTS'15, September 11, 2015, Paris, France.

ACM ISBN 978-1-4503-3543-0/15/09.

DOI: <http://dx.doi.org/10.1145/2799371.2799374>.

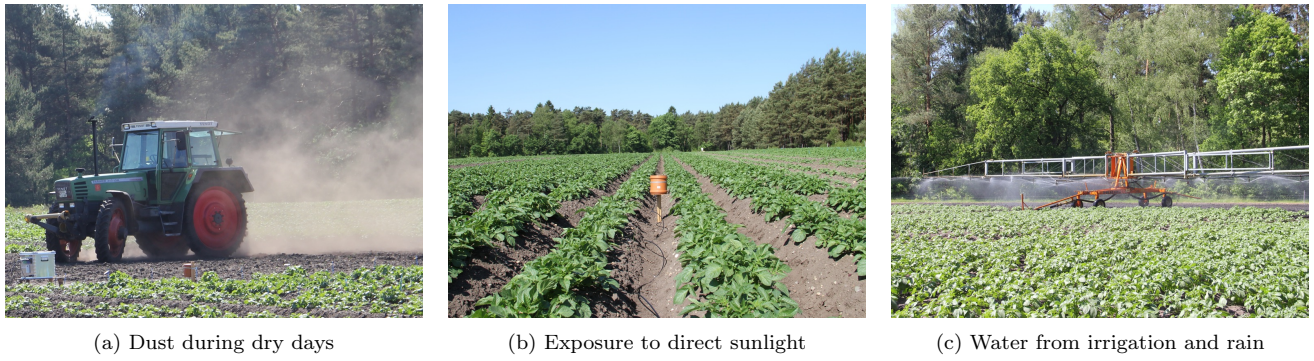


Figure 1: PotatoNet Environment

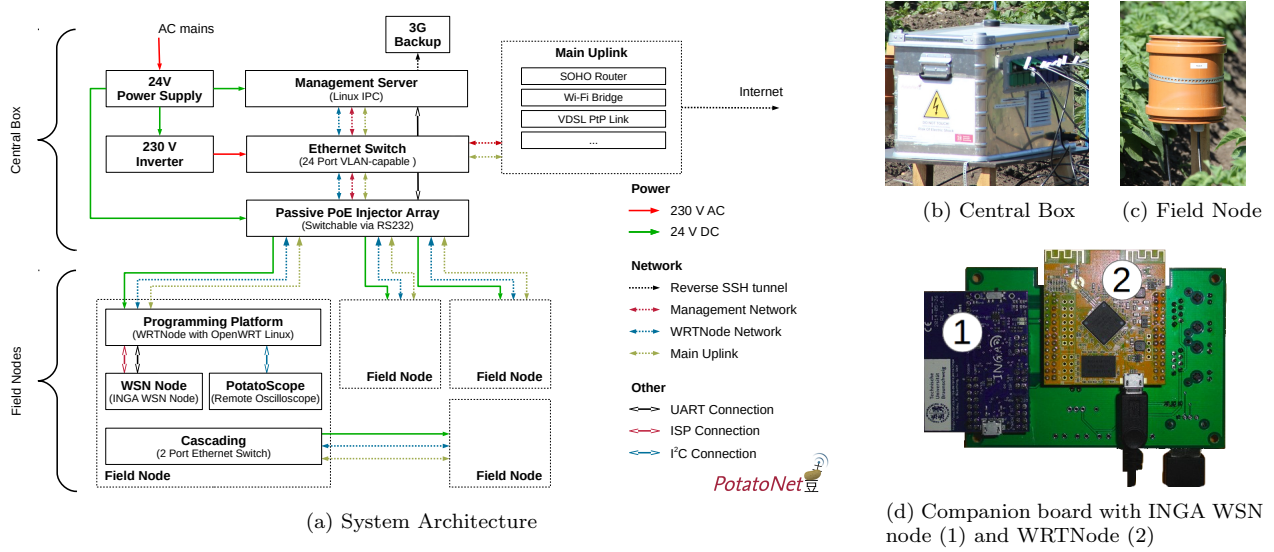


Figure 2: PotatoNet Architecture and Components

Acknowledgments

We would like to thank the VSD Dethlingen potato research station⁴ for providing the field and technical support.

4. REFERENCES

- [1] J. Beutel, B. Buchli, F. Ferrari, M. Keller, M. Zimmerling, and L. Thiele. X-sense: Sensing in extreme environments. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2011*, March 2011.
- [2] F. Büsching, U. Kulau, and L. Wolf. Architecture and Evaluation of INGA - An Inexpensive Node for General Applications. In *IEEE Sensors 2012*, October 2012.
- [3] H. Hellbruck, M. Pagel, A. Kroller, D. Bimschas, D. Pfisterer, and S. Fischer. Using and operating wireless sensor network testbeds with wisebed. In *The 10th IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, June 2011.
- [4] P. Hurni, M. Anwender, G. Wagenknecht, T. Staub, and T. Braun. Tarwis - a testbed management architecture for wireless sensor network testbeds. In *7th International Conference on Network and Service Management (CNSM)*, Oct 2011.
- [5] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In *20th International Parallel and Distributed Processing Symposium*, April 2006.
- [6] M. Navarro, T. W. Davis, Y. Liang, and X. Liang. A study of long-term wsn deployment for environmental monitoring. In *24th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, Sept 2013.
- [7] J.-P. Sheu, C.-J. Chang, C.-Y. Sun, and W.-K. Hu. Wsntb: A testbed for heterogeneous wireless sensor networks. In *First IEEE International Conference on Ubi-Media Computing*, July 2008.
- [8] G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: a wireless sensor network testbed. In *Fourth International Symposium on Information Processing in Sensor Networks (IPSN)*, April 2005.

⁴<http://www.vsd-dethlingen.de>

Undervolting in Real World WSN Applications: A Long-Term Study

Ulf Kulau, Stephan Rottmann, Sebastian Schildt, Johannes van Balen and Lars Wolf

Institute of Operating Systems and Computer Networks

TU Braunschweig

Email: [kulau|rottman|schildt|vanbalen|wolf]@ibr.cs.tu-bs.de

Abstract—The fault-tolerant character of WSN protocols and applications that do not assume completely reliable systems legitimize undervolting – a highly efficient energy management technique where the supply voltage is set below the minimum specifications. As has been shown in earlier work, by using the reliable IdealVolting undervolting scheme the lifetime of WSN applications can be increased significantly while keeping the node in a safe state even under rough environmental conditions. To show the usability of undervolting in a real world WSN deployment, we performed a long-term study of IdealVolting in a Smart Farming application. All measurements were performed on a generic outdoor testbed for WSNs (PotatoNet) which is also presented within this paper. We collected a long-term dataset of a WSN running for one farming season on a potato field to compare the reliability and performance characteristics of IdealVolting against a regular powered WSN.

I. INTRODUCTION

The limping evolution of batteries on the one hand and the growing field of challenging Wireless Sensor Network (WSN) applications on the other necessitate advanced power-management techniques on several layers. Existing solutions for energy management act conservatively, including reserves and safety margins, neglecting the fact that a WSN on the lower layers is not a reliable system: For example, connectivity can be fluctuating and unpredictable, and nodes may fail any time due to energy problems or hardware failures due to rough environmental conditions. Instead reliability is realized by upper layers such as specific network protocols and careful application design.

Hence, it is time to rethink the constraint of absolute reliability when aiming to increase the energy efficiency significantly. Operating WSN nodes at voltage levels below their minimum recommendations, offers such a potential of significant energy savings but at an increased risk of failures [1]. In [2] we proposed IdealVolting, which is an adaptive and reliable undervolting scheme for WSN nodes. IdealVolting tries to combine aggressive energy savings with reliability by finding safe operating areas below the minimum specification for each individual node under different temperatures. IdealVolting uses a supervised learning strategy which is able to keep the nodes in safe states even if the environmental conditions are constantly and rapidly changing.

In this paper we focus on a real world application and perform a long-term study of about 4 months to examine the performance of IdealVolting in WSNs. While [2] has demonstrated the energy saving potential of IdealVolting, in

this paper we wanted to measure by how much, if any, IdealVolting decreases baseline reliability of a real WSN.

For this purpose we firstly present PotatoNet – a robust outdoor WSN testbed for Smart Farming applications. The PotatoNet testbed was used to evaluate the WSN reliability by continuously alternating between IdealVolting and operating at a nominal voltage level.

In consequence we are able to compare an undervolted WSN using IdealVolting against a normally powered WSN. As far as we know this is the first work that evaluates undervolting within WSN in a real world deployment. We will show that there is a difference between individual nodes regarding their undervolting capability and that IdealVolting can adapt to these individual differences. The measurements show that IdealVolting fulfills the promise of increasing energy efficiency significantly without having a measurable negative effect on WSN reliability.

II. IDEALVOLTING

An important aspect to be considered when undervolting the micro controller units (MCUs) of a sensor node is that the voltage ratings given in the datasheet cover a large temperature range. Yet, the threshold voltage (V_{th}) of a Complementary Metal-Oxide Semiconductor (CMOS) gate is temperature dependent. For the threshold voltage V_{th} at temperature T the following equation holds

$$V_{th}(T) = V_{th0} + \alpha \cdot (T - T_0) \quad (1)$$

where V_{th0} is the threshold voltage at the temperature T_0 and α is a temperature coefficient [3]. While MCU manufacturers need to specify their minimum voltage for worst-case conditions at the extreme of the allowed temperature range and taking production variances into account, Equation (1) implies that under most conditions an MCU has the ability to run well below its specified minimum voltage.

The actual voltage level at which a node or an MCU still works properly depends on the clock speed, the temperature, and its individual properties due to variations in the manufacturing process. To adapt each node to its “ideal” voltage level, IdealVolting implements a control loop between an undervolted main MCU and a tiny but reliable co-MCU. While the main MCU executes a spot-test periodically, the co-MCU validates the test-results to decide whether the voltage level can be lowered or not. By collecting a set of absolute minimum voltage levels where malfunctions are detected, the co-MCU

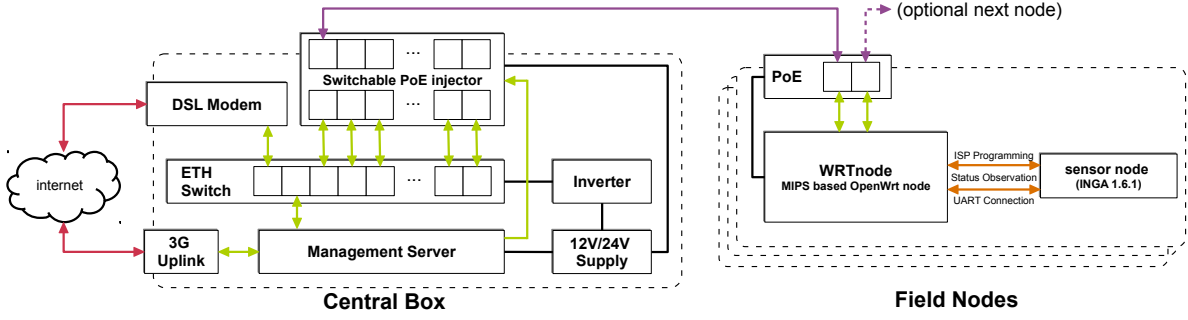


Figure 1: PotatoNet system architecture.

facilitates a supervised learning process. It is able to learn and predict a safe voltage level for a given temperature value $V(T)$ on a node-individual basis.

Of course, the additional hardware incurs some overhead that needs to be offset by the gains of undervolting. In our earlier work we have already shown, that this is possible and the lifetime of WSN applications can be increased significantly [2]. In this paper the scope is to determine whether adaptive undervolting has an impact on the reliability of WSN applications.

IdealVolting mainly affects the MCU of the sensor node (ATmega1284p). In this context it is important to mention that although the transceiver is connected to the same voltage path as the undervolted MCU, during our evaluation the specified minimum voltage level of the transceiver ($V_{r,f} = 1.8\text{ V}$) was never undershot. Hence, the transceiver unit of the nodes has never been undervolted during this experiment.

III. POTATONET – OUTDOOR WSN TESTBED

In this work the goal is to determine the stability of adaptively undervolted WSN nodes under realistically changing environmental conditions. These conditions can be encountered in Smart Farming applications. Smart Farming is an upcoming application area for WSNs, which are used to analyze and monitor the conditions of the soil and crops in order to optimize yield and minimize the use of fertilizer and pesticides. Hence, to evaluate IdealVolting against the backdrop of a real world application, we entered a cooperation with a potato research station¹ in Northern Germany, that allowed us to use one of their trial fields for our outdoor WSN testbed – PotatoNet.

Although the basic intention of PotatoNet was to evaluate undervolting in WSNs, the testbed was designed to be generic for further research. As the testbed is around 100 km away from the university and is intended for long-term experimentation, the overall system design strongly focuses on robustness.

This section gives an overview on the PotatoNet’s architecture and the challenges that are typically encountered when setting up a WSN testbed outdoors.

The PotatoNet architecture depicted in Figure 1 consists of a “Central Box” that manages the communication uplink,

data collection and power distribution and a number of “Field Nodes” that contain a WSN node and supporting hardware.

A. Power Supply and Distribution

Due to the need for a management server, the programming hardware and a potentially large amount of nodes, it is clear that the whole PotatoNet setup cannot be run off a couple of batteries. However, since for certain remote scenarios operating from a battery-buffered off-grid solar array might be the only option, the whole electronics of the central box is designed to be able to run off a 24 V DC supply. The central switch is the only component that requires 230 V AC, therefore an AC inverter is used to power it. At our test field mains power is available, therefore the central box includes a 300 W 24 V DC power supply for itself and all field nodes.

Power is distributed to the field nodes using Passive Power over Ethernet (PoE), which uses some lines in an Ethernet cable for DC power without any signaling. This reduces the maximum Ethernet speed to 100 Mbps, which is not a problem for PotatoNet’s use case. The PoE injectors can be switched on and off individually by the management server. This allows a hard reset of nodes that are stuck and it can be used to power off nodes that are not part of a running experiment.

Upon reboot all PoE injector ports are disabled and will not be enabled automatically. This is a safety feature in case one of the nodes develops a short circuit, triggering a crash of the rest of the system. By restarting without any active node, such problems can be debugged remotely by activating the nodes one by one.

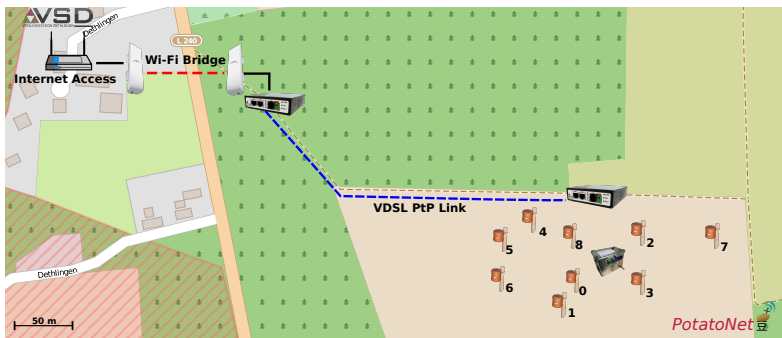
The power usage of the whole system is around 45 W for the central box with 2 W added for each field node.

B. Central Box

The central box of PotatoNet is mainly based on an embedded PC – called “Management Server”.

Management Server: The management server is an IPC (Advantech ARK-1360F-S6A1E) running Debian Linux. It has no moving parts as it is fan-less and boots from a Compact Flash card. We integrated a cellular data card so the management server has a network connection, even if the rest of the PotatoNet network breaks down. We choose fairly robust hardware for the management server, as it is a crucial component. While it is not a problem if some remote

¹<http://www.vsd-dethlingen.de>



(a) Node locations and uplink components



(b) Field view – Central box and field nodes.

Figure 2: PotatoNet deployment at trial field of the VSD research station in Dethlingen.

nodes fail, without the management server it is not possible to interact with the testbed remotely. As long as electricity supply is available, the management server should boot up and build a network connection.

Central Box Mechanics: All components of the central box are mounted inside a 73 liter aluminum box (60 x 40 x 41 cm³, see Figure 2b). We use rugged connectors for Ethernet and power to protect against dust and humidity. For thermal reasons a filter fan with 15 m³ h⁻¹ throughput and an additional filtered air intake on the opposite side provide some airflow through the case. This kept the air temperature under 40 °C even under direct sunlight during the summer season.

C. Field Node

The PotatoNet design includes an independent system that communicates with the management server and is able to program and collect data from a WSN node.

Programming Platform: We decided to go for an embedded Linux platform for cost and flexibility reasons. Every WSN node is paired with a WRTnode². The WRTnode includes a 680 MHz MIPS (MT7620 [4]) processor and 64 MiB of RAM. It runs the OpenWRT³ embedded Linux distribution. The WRTnode supports a number of GPIOs, which can be used for In System Programmer (ISP) programming of the WSN node. Two Ethernet MACs are supported which allows communication with the management server as well as chaining WRTnodes. As an added bonus the WRTnode includes a complete Wi-Fi implementation, which means the PotatoNet can also be used for Wi-Fi experiments. A single WRTnode costs USD 25, which is cheaper than having a low production run of less powerful, custom designed hardware. By supporting OpenWRT a large amount of standard Linux software is available for installation directly from the OpenWRT repositories. This makes programming or processing data collected from the WSN node easy.

To interface the WRTnode with the WSN node and connect it to the management server we designed a simple companion board which includes some passive components for the ISP programmer and two Ethernet jacks and a DC-DC voltage

regulator that can generate the required 5 V DC supply from the PoE supply. The WRTnode and the INGA WSN node can be plugged into the companion board directly. Including components, the board adds about USD 10 to the price of a node.

WSN Node: We use the INGA WSN node [5] in version 1.6.1. This version of the node is a derivation of our first undervolting capable sensor node presented in [2]. The design includes an IEEE 802.15.4 transceiver (AT86RF233) as well as two microcontrollers (ATmega1284p, ATtiny84) to enable the IdealVolting implementation. However, for more details about the node's architecture we would like to refer to [2].

The WRTnode is able to program both INGA processors independently by using standard programming software⁴.

Node enclosure: We needed an enclosure to hold the INGA node and WRTnode. In contrast to the central box, we went for a cheap design without costly rugged connectors. Of course, weather ruggedness should not be compromised. We ended up with a design based on standard PVC-U tubing with 160 mm diameter. We used a drainage double socket closed with two socket plugs. The electronic components have been mounted to one socket plug. As only one (or two for cascading) Ethernet cable needs to go into enclosure we used simple cable glands. The total cost for one enclosure is about USD 10. Figure 2b shows two deployed enclosures (in orange) on the potato field.

D. Network Architecture

In the PotatoNet system there are several physical and logical networks running in parallel. The Ethernet networks are separated by VLANs and managed by the switch in the central box. Only the management server has access to all VLANs. The following networks exist in PotatoNet:

IEEE 802.15.4 network: Network between the INGA WSN nodes (cf. Section III-C). This network is used for the actual long-term evaluation described in this paper. It is neither required for logging nor for reprogramming. Hence, the evaluation is not influenced by any overhead due to maintenance messages.

²<http://wrtnode.com/>

³<https://openwrt.org>

⁴<http://www.nongnu.org/avrdude/>

WRTnode Ethernet network: One Ethernet port of each WRTnode is connected to the central box. The management server can connect to the WRTnodes and execute actions such as reprogramming the attached INGA node or collecting logged data. If the main uplink is available, the WRTnodes can access the Internet which simplifies updating or installing new software. If the PotatoNet falls back to its (metered) backup uplink, Internet access for the WRTnodes is cut, but they can still connect with the management server. Because we equipped each WRTnode with two Ethernet ports in a switch configuration, WRTnodes can be daisy-chained. Due to the low power draw and the comparatively high PoE voltage of 24 V this is efficient enough to daisy-chain several field nodes.

Main uplink: One Ethernet port of the central box is reserved for the Internet uplink. It is just expected that an IP address with Internet access can be obtained via DHCP auto-configuration. Therefore, it is enough to connect the uplink port to a typical SOHO router. If needed, the uplink port can be permanently powered, for connecting PoE equipment such as a Wi-Fi bridge.

Cellular backup uplink: When the main uplink fails, a cellular backup, driven by a UMTS card in the management server will be used. Even when most components fail, as long as the management server gets electricity and boots up, a researcher can login through the cellular backup and analyze any problems.

Device management network: The manageable switch as well as components such as Wi-Fi bridges usually contain some web-interface for configuration. We put the management interfaces of these devices into a separate VLAN that can be reached via the management server. This allows monitoring and reconfiguration of these devices even after PotatoNet is deployed.

E. Deployment

Before deploying the testbed at its final location, the central box and field nodes have been tested by running on an exposed spot on the roof of a university building for 4 weeks. This enabled us to test resistance to various weather conditions and assess the general reliability of the system, while still being able to access it easily. We could verify that the enclosures can withstand heavy rain and that direct exposure to the sun is not a problem from a thermal point of view. Afterwards, we deployed 9 field nodes and the central box on a farmed testing field from the VSD research station for potato crops to perform our long-term evaluation.

As the installation is 100 km away from the university and cellular connection is too unreliable to act as the sole uplink, we needed a secondary (stable) internet connection. The next low-speed DSL landline is around 450 m away inside the research station. Due to a street and a small forest in between the station and the field, a direct Wi-Fi link is infeasible. We opted to cross the street with a Wi-Fi Bridge and use a VDSL point-to-point link to bridge the distance to the field. Figure 2a depicts the deployment, the location of field nodes and the main uplink topology.

IV. LONG-TERM EVALUATION

The intention of the long-term evaluation is to show the usability and reliability of an undervolted WSN compared to a WSN where all nodes run at specified voltage levels.

Therefore, we let all field nodes change periodically between IdealVolting and using nominal voltage according to the datasheet. By switching modes every 144 seconds, we can ensure that nodes experience similar environmental conditions whether using IdealVolting mode or running with nominal voltage.

Every INGA WSN node executes a process which is able to handle commands via Universal Asynchronous Receiver/Transmitter (UART). These commands are sent by the central box via the WRTnode connected to an INGA. There are commands to force the node to (i) enable/disable the IdealVolting process and (ii) to send data to a given *destination* at a given *tx-level* with a given *payload*.

As the WRTnode is connected to the node's UART, a virtual connection to the management server can be used to delegate the entire evaluation by the central box. For the evaluation we used 9 field nodes. Every node is able to send the data at 16 different tx-levels ($[0 = +4 \text{ dBm}, \dots, 15 = -17 \text{ dBm}]$) [6]. The payload message includes a unique and consecutive sequence number n and the current temperature T at the sender itself, which is a very typical task for a WSN application.

All debug outputs of the nodes were collected by the central box via the virtual UART connection. Algorithm 1 depicts the pseudo-code running on the management server orchestrating the experiment.

Algorithm 1 Centralized scheduler

```

1: iv ← on
2: while true do
3:   for each node  $i \in N$  do
4:     tell node  $i$  set IdealVolting=iv
5:   end for
6:   for each node  $i \in N$  do
7:     for each tx-level  $l \in S$  do
8:       tell node  $i$  to broadcasts  $(i, l, T, n)$ 
9:        $n \leftarrow n+1$ 
10:      wait 1s
11:    end for
12:  end for
13:  if iv == on then
14:    iv ← off
15:  else
16:    iv ← on
17:  end if
18: end while

```

The central server will ask each WSN node in turn to send a packet at a given transmit power while executing either IdealVolting or not. The sent data includes the node's address i , the tx power level l , the sensed temperature T and a sequence number n uniquely identifying the transmission. The destination of each message was the broadcast address of the WSN. Every node within the communication range of the respective sender logs the received information (i, l, T, n) and adds information about the received signal strength indicator (RSSI) and local temperature T_l . This data is sent via UART and the WRTnode network to the management server.

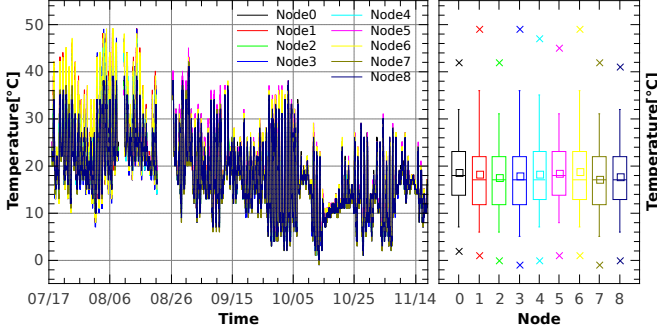
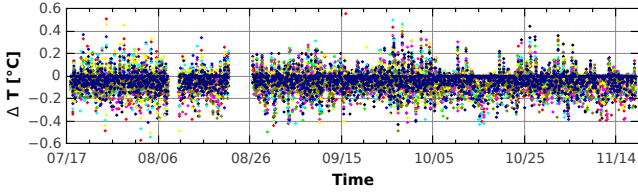


Figure 3: Temperature values of all nodes.

Figure 4: Differences of temperature measurements between undervolted and normal powered nodes ΔT .

Before the start of this study, each node has already learned its individual undervolting characteristic $V(T)$ [2]. This model will only be adapted by IdealVolting, if errors occur.

In total we collected 7.5 GB of data over a time span of almost 4 months. Three different scenarios can be considered, as the potato crops had been farmed and harvested during our study.

- At the beginning the potato crops grew and the links between nodes were influenced by the water-rich plants (07/17 – 08/21).
- In the second phase the plants were chopped so that no plants shadowed the communication between nodes (08/25 – 09/11).
- For the last stage all nodes were moved and deployed at the edge of the field, to avoid the destruction of the testbed as the potatoes were harvested (09/11 – 11/18).

A. Temperature Measurement

The simplistic Smart Farming task performed by each WSN node was to sample the surrounding temperature with 1 Hz sample rate, and to store the data on the WRTNode. This ensures that nodes are busy processing data even while they are not currently asked by the central scheduler (cf. Algorithm 1) to transmit a temperature update. Thus, we have a higher chance of provoking errors during the IdealVolting phase.

Figure 3 shows the measured temperature values of all nodes for the entire duration of this study. Beside the maximum values of $-1^\circ\text{C} \leq T \leq 49^\circ\text{C}$ the high fluctuation of T indicate that WSNs deployed on a field are exposed to a challenging environment. Although all nodes sampled almost the same environment, there are slight variations between the nodes which is also depicted in Figure 3. This is due to the

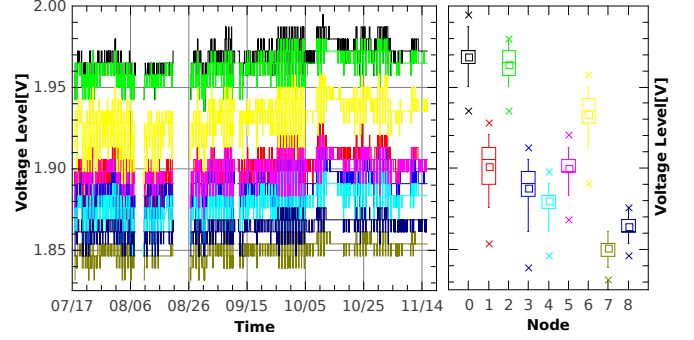


Figure 5: Voltage levels of all nodes during IdealVolting. Significant heterogeneity in terms of minimum voltage capability of individual nodes can be observed.

differently sized potato crops. While some nodes were covered by leaves, others experienced more exposure to direct sunlight.

B. IdealVolting Characteristics

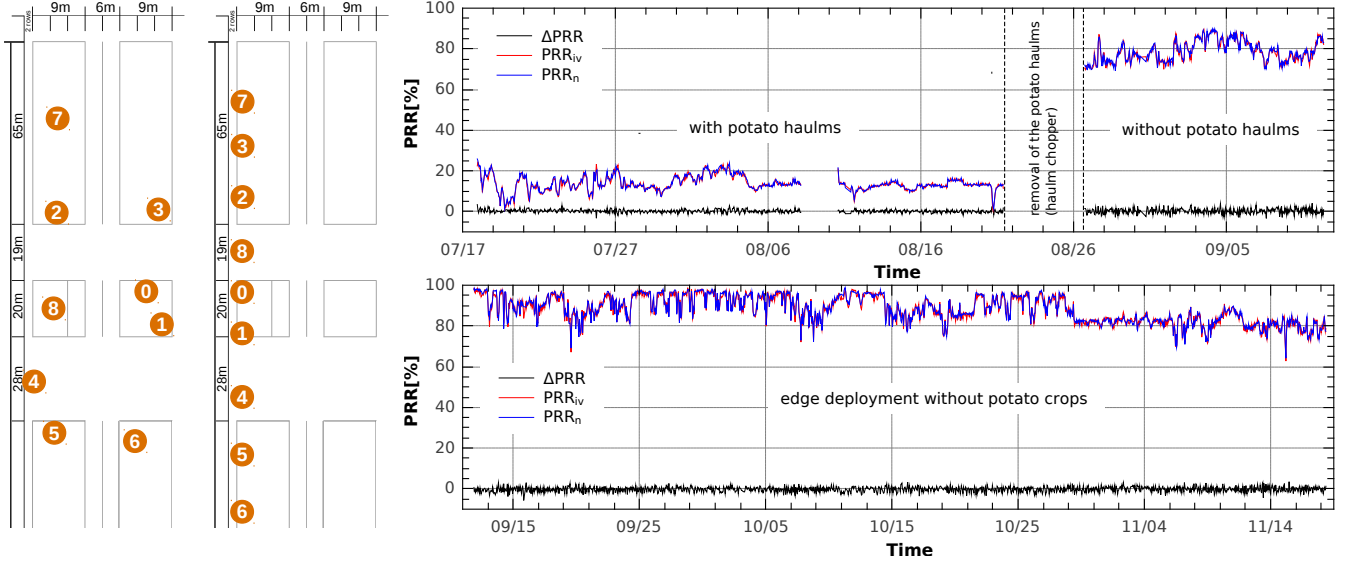
During this study each node operated at a fixed clock rate $f_{cpu} = 8 \text{ MHz}$. For this frequency the manufacturer specifies a safe operating voltage of at least 2.4 V [7].

The ideal voltage level (absolute minimum voltage level without malfunction) chosen by IdealVolting depends on (i) the individual characteristics of the MCU and (ii) on the temperature experienced by the node. The minimum voltage levels are inversely proportional to temperatures so that the IdealVolting process continuously adapts the voltage to operate the MCU with the lowest voltage that is still within the learned safe operating area. In Figure 5 the continuous adaptation of the nodes as well as the individual characteristics caused by different intensities of adaptation can be seen. As a consequence the strong temperature dependency of the absolute minimum voltage level and the individual characteristics of nodes add a strong heterogeneity to the network. This node-individual information might be used in the future to further optimize routing and load balancing approaches in WSNs to increase the lifespan of the whole network.

To check the correctness of results from an undervolted MCU, IdealVolting does a regular spot-test that is evaluated by the safe co-MCU (cf. Section II). This spot-test is based on the examinations of [8] which shows that a simple matrix-multiplication is sufficient to detect arithmetic logical unit (ALU) errors [9], [2]. For this evaluation the spot-test was also executed when the nodes were running with nominal voltage. All test-scores have been logged at the central box. In total each node executed $\approx 10^7$ spot-tests. During the whole study neither normal powered nodes, nor nodes running IdealVolting ever failed a spot-test.

C. Impact on Sensing Quality

As the temperature measurement was performed by both the WSN running in IdealVolting mode and the WSN running with normal voltage, we can answer the question whether undervolting influences the quality of sensed data. The correct operation of the MCU is only one aspect of a correctly



(a) Field-topology (b) Edge-topology (c) Node 0: PRR – field deployment (top figure) and PRR – edge deployment (bottom figure)

Figure 6: Topology of the field and edge deployment with exemplary results of the PRR for node 0.

 Table II: Summary of the PRR evaluation and comparison. Table includes the mean values of the PRR for undervolted and normal state ($\overline{PRR}_{iv}$, $\overline{PRR}_n$), the mean value of the difference with standard deviation (ΔPRR) and the correlation $r(\Delta PRR, v)$.

Node	Field deployment (with potato haulms)				Field deployment (without potato haulms)				Edge deployment (without potato crops)			
	$\overline{PRR}_n$	$\overline{PRR}_{iv}$	ΔPRR	$r(\Delta PRR, v)$	$\overline{PRR}_n$	$\overline{PRR}_{iv}$	ΔPRR	$r(\Delta PRR, v)$	$\overline{PRR}_n$	$\overline{PRR}_{iv}$	ΔPRR	$r(\Delta PRR, v)$
0	13.64	13.84	0.199 ± 0.756	0.0220	78.59	78.91	0.316 ± 1.990	-0.0241	88.15	87.88	-0.271 ± 1.183	-0.0941
1	9.61	10.00	0.387 ± 0.797	-0.0501	78.72	79.33	0.609 ± 1.548	-0.3945	85.24	85.43	0.184 ± 1.251	-0.0413
2	5.05	5.18	0.129 ± 0.726	0.0083	54.29	54.67	0.384 ± 1.380	-0.1904	74.41	74.66	0.241 ± 1.257	-0.2247
3	17.64	17.69	0.051 ± 0.939	0.0560	82.39	82.17	-0.218 ± 1.052	-0.0949	78.58	78.29	-0.287 ± 1.162	-0.0387
4	20.47	21.50	1.025 ± 1.462	-0.3214	83.00	83.18	0.184 ± 1.264	-0.2864	88.11	88.31	0.201 ± 1.269	-0.0183
5	6.84	7.07	0.229 ± 0.605	-0.1595	68.08	68.55	0.463 ± 1.431	-0.3150	76.36	76.91	0.544 ± 1.383	0.0383
6	11.96	12.07	0.114 ± 0.775	-0.1161	59.56	60.53	0.971 ± 1.630	-0.3444	55.14	55.93	0.786 ± 1.366	-0.2974

 Table I: Difference in measured temperature between Ideal-Volting and normal operation. No correlation $r(\Delta T, v)$ between temperature difference and voltage $\rightarrow$ IdealVolting is not influencing the sensed data.

Node	$\overline{\Delta T} [^\circ C]$	$max(\Delta T) [^\circ C]$	$min(\Delta T) [^\circ C]$	$r(\Delta T, v)$
0	-0.0327 ± 0.0705	0.4433	-0.4317	-0.0324
1	-0.0450 ± 0.0849	0.5490	-0.5711	0.0134
2	-0.0346 ± 0.0697	0.2962	-0.4938	0.0045
3	-0.0339 ± 0.0757	0.3658	-0.4165	0.0029
4	-0.0418 ± 0.0832	0.4948	-0.5472	0.0340
5	-0.0401 ± 0.0741	0.3605	-0.4115	-0.0134
6	-0.0364 ± 0.0808	0.4604	-0.5026	-0.0073
7	-0.0420 ± 0.0753	0.3920	-0.4873	-0.0376
8	-0.0429 ± 0.0755	0.3499	-0.4045	-0.0056

operating node. If IdealVolting reduces the accuracy of sensed data, we need to know as this might not be acceptable for some applications.

For each node, Figure 4 depicts the difference between the temperature measurements in IdealVolting and normal powered state for all nodes ΔT . It can be seen that the absolute deviation is always $< \pm 0.6^\circ C$. To prove that the quality of

sensing is not influenced by undervolting, we correlated the voltage level v of the nodes with the respective differences ΔT between the measurements performed in IdealVolting and normal mode. If there was a connection between measurement deviations and voltage levels, the correlation coefficient $r(\Delta T, v)$ would reveal this. The results are shown in Table I. For each node we calculated the average temperature difference $\overline{\Delta T}$ and standard deviation between IdealVolting and normal operation. The minimal and maximal measurement differences show that we do not have any outliers, where we get a temperature difference significantly above the average difference. Finally, the correlation coefficient $r(\Delta T, v)$ proves, that there is no correlation between voltage level v and the difference in temperature measurement ΔT .

Knowing this we can also infer that in both states, Ideal-Volting and normal operation, the WSN experienced the same environmental conditions. By switching between IdealVolting and normal mode every 144 s we really are comparing the two operating modes under the same environmental conditions.

D. Characteristics of PRR

Another major question is whether undervolting has an impact on the performance of the WSN communication. The Packet Reception Ratio (PRR) is the most important metric here. Any energy savings by the MCU would immediately be offset if a decreased PRR in IdealVolting mode demands costly packet retransmissions.

To analyze the connection between undervolting and the PRR of a WSN we first define the PRR as the percentage of packets that have been received out of all sent packets. Hence, we calculated the PRR of each node for all broadcasted messages from every *other* node at every tx-level. Due to the centralized logging we know

- 1) When a node transmitted a message.
- 2) Whether the WSN was operating in IdealVolting mode or not at this time.
- 3) What was the content of the message (i, l, T, n) (cf. Section IV).
- 4) Which other nodes received this message.

Therefore, we are able to consider the PRR in the IdealVolting mode (PRR_{iv}) and nominal voltage mode (PRR_n) separately.

Figure 6 shows the PRR of node 0 for the different scenarios described in Section III-E. Figure 6a shows the topology of the field deployment while the top figure of Figure 6c shows the corresponding PRR_n and PRR_{iv} over time. The difference between the PRRs is depicted as $\Delta PRR = PRR_n - PRR_{iv}$. The short interruption (08/08 - 08/10) results from a temporary usage of the PotatoNet for a different short-term experiment.

It is obvious that the potato crops have a significant impact on the PRR. After a haulm chopper removed the potato crop haulms on August 28th, the PRR immediately increased by ≈ 4 times. On September 11th we rearranged the nodes at the edge of the field due to the final harvesting of the potato crops. Figure 6b shows the changed topology. The bottom figure of Figure 6c plots the corresponding PRR values.

In all scenarios the PRR of node 0 stays almost equal for both WSN states. The difference ΔPRR seems negligible. This is a good indication that IdealVolting is not influencing the PRR. To prove this, similar to the argumentation in Section IV-C, we correlated the voltage level during IdealVolting v with the difference in PRR, ΔPRR . If undervolting impacts the PRR of the WSN the correlation coefficient $r(v, \Delta PRR)$ would show a statistical relationship.

Table II summarizes the entire evaluation including the correlation $r(\Delta PRR, v)$ in every scenario for the nodes 0 to 6. Nodes 7 and 8 have not been considered for this evaluation. Although both nodes operated throughout the entire evaluation, they received far fewer messages (around 1% of the amount the other nodes received). There is just not enough data for any meaningful statistical analysis. The weak reception rate of these nodes was independent of the voltage state. While there seems to be a bias toward negative correlation values, the absolute values are very low so that a statistical connection between undervolting and the PRR of the WSN can not be attested.

This analysis proves that IdealVolting has no negative effect on the performance of the wireless links.

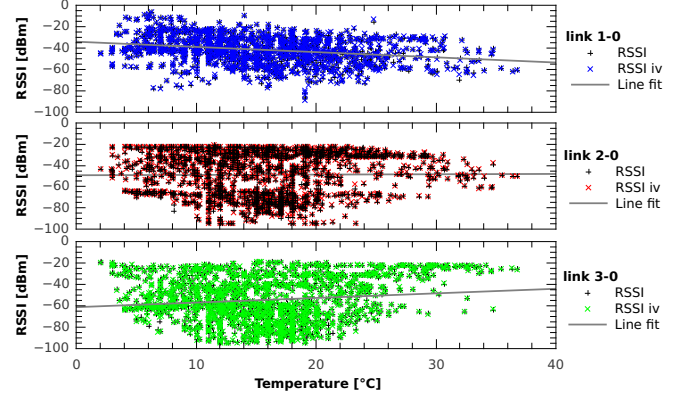


Figure 7: RSSI of undervolting and normal powered nodes against temperature (exemplary links).

E. Impact on RSSI

As already mentioned, the transceiver unit of the nodes is supplied by the same voltage path as the MCU. Hence, during IdealVolting the voltage level of the transceiver varies too. However, IdealVolting never falls below the minimum recommended voltage level of the transceiver (1.8 V).

To analyze whether the adaptation of the voltage level has an impact on the RSSI, we reused the method of correlating the voltage level v with the difference between RSSI of undervolting and normal powered nodes ($\Delta RSSI$). Table III shows the mean $\overline{\Delta RSSI}$ and standard deviation for links to node 0. As to be expected from the PRR results, the correlation coefficient $r(\Delta RSSI, v)$ indicates no connection between undervolting and the RSSI.

Recent publications stated that temperatures can influence the performance of wireless communication. Commonly used IEEE 802.15.4 transceivers suffer from high temperatures, as their internal amplifiers efficiency decreases [10], [11]. In [12] this effect is also validated. However, so far only one specific transceiver unit (TI CC2420 [13]) has been examined. As we have shown that undervolting has no impact on RSSI, we can take a look if and how temperature influences the performance of our 802.15.4 receiver. Different from previous work, the INGA node uses an AT86RF233 [6] transceiver.

Figure 7 shows the RSSI as a function of the temperature for three exemplary links. For this evaluation we only considered the edge topology, as this deployment was not affected by external events (e.g. plants). We can neither prove nor refute that higher temperatures tend to lower RSSI but a detailed

Table III: IdealVolting has no impact on the RSSI.

Link	$\overline{\Delta RSSI}[\text{dBm}]$	$r(\Delta RSSI, v)$
1 $\rightarrow$ 0	0.11509 ± 1.01578	0.05137
2 $\rightarrow$ 0	0.06573 ± 0.77007	0.02772
3 $\rightarrow$ 0	0.05139 ± 0.91692	0.00508
4 $\rightarrow$ 0	0.05924 ± 1.01099	0.01831
5 $\rightarrow$ 0	0.00975 ± 0.80512	0.00694
6 $\rightarrow$ 0	0.01332 ± 0.40945	-0.02496

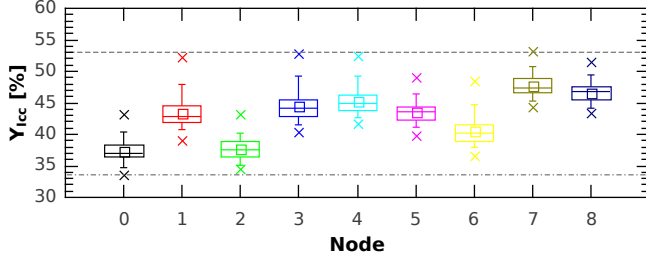


Figure 8: Savings $Y_{I_{cc}}$ of current consumption I_{cc} when using IdealVolting instead of normal supply voltage. High variations due to temperature and individual characteristics.

examination is beyond the scope of this paper.

F. Energy Efficiency of the MCU

Our results have shown that undervolting using IdealVolting neither impacts reliability nor sensing accuracy of the WSN application. However, IdealVolting has a significant impact on the energy efficiency of the nodes. In [14] we presented an energy model $I_{cc}(v, T)$ of the used MCU which was derived by measurements in a climatic chamber. Using this model, the current consumption of the MCU can be expressed by the following Equation:

$$\forall \begin{cases} 1.6 \text{ V} \leq v \leq 2.4 \text{ V} \\ -15^\circ \text{C} \leq T \leq 60^\circ \text{C} \end{cases} : I_{cc}(v, T) = p + s \cdot T + t \cdot v \quad (2)$$

with $p = -4.558 \text{ mA}$, $s = -11.976 \mu\text{A K}^{-1}$ and $t = 3.77 \text{ mA V}^{-1}$

With this model, it is possible to calculate the current consumption of the MCU for each node and every state and thus quantify the savings in current consumption $Y_{I_{cc}}$ due to undervolting.

For the sake of fairness we only compare the current consumption I_{cc} instead of the power dissipation. The power dissipation P would include the voltage level v which leads to even more savings due to IdealVolting as the voltage level is dynamically lowered depending on the temperature. However, when comparing the power dissipation the efficiency of voltage conversion plays a significant role. As this efficiency depends on the actual components (e.g., buck converter, linear dropout converter (LDO)) we limit our comparison to I_{cc} .

Figure 8 shows the distribution of current consumption savings across all nodes. Due to the experienced temperature and individual characteristics of each nodes the savings vary over time for each node. In total, the savings lay between 33.52% and 53.16%. This increase in the MCU's efficiency immediately translates into an increased lifetime for a WSN and knowledge about the individual characteristics of each node can be used to fine-tune, e.g., load balancing.

V. CONCLUSIONS

Violating the specified minimum voltage levels of components by undervolting increases the energy efficiency significantly. As operating MCUs outside their specification increases the risk of malfunctions, undervolting needs to be

managed carefully. In previous work we presented IdealVolting, an adaptive self-learning undervolting scheme for WSN nodes that aims to preserve reliability while saving energy by undervolting.

In this paper we performed a long-term study to examine whether undervolting can be used for real-world WSN applications. For this purpose we presented PotatoNet – a robust outdoor WSN testbed – to run evaluations in a realistic Smart Farming scenario.

Our evaluation has proven that IdealVolting has no impact on the reliability of the wireless links and no impact on the quality of sensed data. Thus, the service quality the application running on the network is experiencing does not decrease when using IdealVolting. At the same time, IdealVolting saved up to 53% of MCU supply current, which directly benefits the network by increasing the lifetime of its nodes. Finally we also release all collected data for the community⁵.

ACKNOWLEDGMENTS

We would like to thank the VSD Dethlingen research station for providing the field and technical support.

REFERENCES

- [1] U. Kulau *et al.*, “Undervolting in WSNs: Theory and Practice,” *Internet of Things Journal*, IEEE, vol. 2, no. 3, pp. 190–198, June 2015.
- [2] —, “IdealVolting – Reliable Undervolting on Wireless Sensor Nodes,” *ACM Transactions on Sensor Networks (TOSN)*, January 2016, accepted for publication.
- [3] R. J. Baker, *CMOS: Circuit Design, Layout, and Simulation*. John Wiley & Sons, 2011, vol. 18.
- [4] *MT7620 Programming Guide*, Ralink/Mediatek, 2012.
- [5] Büsching *et al.*, “Architecture and Evaluation of INGA - An Inexpensive Node for General Applications,” in *IEEE Sensors 2012*. IEEE, October 2012, pp. 842–845.
- [6] Atmel, “AT86RF233, Low Power, 2.4GHz Transceiver for ZigBee, RF4CE, IEEE 802.15.4, 6LoWPAN, and ISM Applications,” 2014. [Online]. Available: <http://www.atmel.com/devices/AT86RF233.aspx>
- [7] —, “Datasheet: Atmega 1284/1284p, rev. atmel-8272g-avr-01/2015,” 2015. [Online]. Available: <http://www.atmel.com/devices/atmega1284p.aspx>
- [8] A. Rohani *et al.*, “An analysis of fault effects and propagations in AVR microcontroller ATmega103 (L),” in *International Conference on Availability, Reliability and Security (ARES'09)*, 2009. IEEE, 2009, pp. 166–172.
- [9] U. Kulau *et al.*, “Undervolting in WSNs – A feasibility analysis,” in *IEEE World Forum on Internet of Things (WF-IoT)*, 2014, March 2014, pp. 553–558.
- [10] C. A. Boano *et al.*, “Hot Packets: A Systematic Evaluation of the Effect of Temperature on Low Power Wireless Transceivers,” in *Proceedings of the 5th Extreme Conference on Communication (ExtremeCom)*. ACM, August 2013, pp. 7–12.
- [11] —, “TempLab: A Testbed Infrastructure to Study the Impact of Temperature on Wireless Sensor Networks,” in *Proceedings of the 13th International Conference on Information Processing in Sensor Networks (IPSN '14)*. ACM, April 2014, pp. 95–106.
- [12] F. Schmidt *et al.*, “If You Can't Take The Heat: Temperature Effects On Low-Power Wireless Networks And How To Mitigate Them,” in *12th European Conference on Wireless Sensor Networks (EWSN 2015)*, February 2015.
- [13] Chipcon, “CC2420 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver,” 2012. [Online]. Available: <http://www.ti.com/lit/ds/symlink/cc2420.pdf>
- [14] U. Kulau *et al.*, “Paint it Black – Increase WSN Energy Efficiency with the Right Housing,” in *The 6th Workshop on Real World Wireless Sensor Networks*, ser. RealWSN15, Seoul, South Korea, Nov. 2015.

⁵<https://www.ibr.cs.tu-bs.de/projects/potatonet/longterm.html>

Distributed Energy Measurement in WSNs for Outdoor Applications

Robert Hartung, Ulf Kulau and Lars Wolf
Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
Email: [hartung | kulau | wolf]@ibr.cs.tu-bs.de

Abstract—Energy is one of the most critical resources in Wireless Sensor Networks (WSN). For any WSN deployment, being it for a productive use case or for a testbed, the ability to perform energy measurements is of major importance, e.g., to estimate the status of the WSN, the need for maintenance, or for remaining lifetime. This paper presents a distributed energy measurement system for outdoor applications. An existing testbed has been extended by the micro controller unit (MCU)-based oscilloscope PotatoScope. Measuring voltage and current of the sensor node, undervolting effects and energy consumption of realtime applications can be observed. The PotatoScope is capable of capturing the power dissipation precisely at a high temperature range of 100 K. Using high sample rates with up to 500 kHz, the system needs to be capable of storing and archiving the measured data. A final evaluation shows a 0.9% drift for voltage measurements and a 1.6% drift for current measurements over the range of 100K. Using an SD card, long-term measurements can be made with up to 500 kHz. Nevertheless, a live mode allows sample rates up to 2 MHz when using the PotatoScope with a computer.

I. INTRODUCTION

Although the academic research on WSNs has been going on for more than 15 years, real-world experiences show that it is still challenging to operate WSNs reliably. This is why testbeds have been established enabling users convenient examinations of their applications for such networks. Many testbeds and management solutions for WSNs exist like [1], [2], [3], [4], [5], [6], [7]. However, all of these testbeds have in common that they are only located indoors. However, especially when WSNs are deployed outdoors, the hardware of the sensor nodes suffers from the environmental conditions. Recent publications have shown that temperature is still an underestimated factor. The efficiency of transceivers, the energy efficiency of the components and batteries as well as the aging of electronic parts is affected by the temperature.

In Bannister et al. [8] the impact of temperatures on the reliability of radio transceivers has been evaluated. They measured the Receive Signal Strength Indicators (RSSIs) of a real world WSN deployment and observed a drop of the RSSI and the packet reception rate (PRR) at higher temperatures. Boano et al. [9] describe variations of up to 56°C within 24h and propose to raise the transmission power to mitigate the adverse effect on the transmission unit [10].

Beside this, Kulau et al. [11] take advantage of the temperature dependency of Complementary Metal-Oxide Semicon-

ductor (CMOS) gates and implement an active undervolting to extend the lifetime of wireless sensor nodes. The absolute minimum voltage level for MCUs gets therefore temperature dependent which allows lower voltage levels at higher temperatures. A basic study how this approach influences the energy efficiency of nodes deployed outdoors has been performed in [12].

In further consequence the energy efficiency and energy budget of a WSN is highly affected by its environmental condition.

To facilitate investigations on these effects and to evaluate power consumption of applications in realistic outdoor environments, we have build the PotatoNet testbed, consisting of 18 nodes, distributed over an area of almost 10000 square meters [13]. Measuring the energy profile of applications – including MAC and routing protocols – can help to increase the lifetime of WSNs significantly. Furthermore, knowing the current drawn from a node can help to optimize the discharge characteristics of batteries [14].

In this paper we propose a reliable, scalable and convenient distributed energy measurement of nodes by using a MCU-based oscilloscope – the so called PotatoScope. An essential requirement for the measurement is correctness within a widespread temperature range of 100 Kelvin. It has to be ruled out that changes in power dissipation due to environmental variances are not traced back on temperature dependencies of the PotatoScope itself.

The distribution of our nodes in our existing testbed requires an on-line measurement, this implies that we measure the energy at each node individually. A current mirror will not be used, as the current would be doubled and this would be a conflict to the goal to save energy in WSNs. Due to the low current consumption of nodes, we cannot use an inductive-based solution, as the magnetic field cannot be measured in all cases. Finally, our solution is using a shunt resistor as this is a common and widely used method.

Moreover, we have to ensure that the proposed approach is scalable. Hence, the PotatoScope should a) be inexpensive and b) implement a distributed storage of energy measurement results. The entire system has to be easy-to-use by the user and should provide transparent interfaces for further customizations.

The rest of this paper is structured as follows: The next

section will discuss related work and outlines differences to our solution. The existing testbed is presented in Section III. Then, Section IV will introduce the detailed architecture of the PotatoScope. A theoretical calculation of the measurement error will be performed in Section V. The evaluation of the PotatoScope is discussed in Section VI. We present our system for distributed energy measurements in Section VII before giving a final conclusion in Section VIII.

II. RELATED WORK

In most setups it is hard to get a realistic picture of energy usage. You can hook up a single node to expensive lab equipment such as an oscilloscope to study it in detail. This, however, does not give an overview how a whole network is performing. Furthermore, it is impractical to apply in outdoor testbeds (cf. previous Section I). Often researcher rely on energy accounting mechanisms built into the operating system. For every transmission or other action they assume a certain fixed amount of energy to be used [15], [16]. The problem is that these mechanisms are not very accurate as the current consumption between different nodes varies significantly even under constant environmental conditions [17]. When individual and precise measurements are required, currents from a few micro-amperes up to several hundred milli-amperes have to be covered. This can usually not be realized using a single shunt resistor, but a series of different resistors is needed. Thus, the design of such energy meters is more complex and adds additional costs to the system as they require multiple stages to measure these currents [18]. In fact, due to the costs and the complex switch-over such systems are neither scalable nor suitable for outdoor applications.

Li et al. [17] present an approach that uses different nodes to derive energy models. Evaluations have been performed in a lab environment to control the temperature easily. However, the impact of different temperatures on the measurement arrangement was not considered, but current consumption at different temperatures where recorded without any counter-check. This means that temperature effects resulting from the measurement circuit itself are not taken into account. An important note to take from their paper is that using a simple modeling approach is not enough. As mentioned above, even across identically constructed nodes there is a difference in power dissipation of up to 15%. As a consequence the model has to be customized for each node and for temperature effects as well.

A further approach performs an on-line measurement on every single node, to directly measure voltage and current consumption. Some *energy meters* have been presented in [19], [20], [21], [22]. Hergenroeder et al. [19] built an USB-based solution – the Sensor Node Management Device (SNMD) – that measures energy of a connected sensor node. It can also be used for performing maintenance tasks like programming the node. Measurements can be taken at a sample rate of up to 500 kHz in buffered mode and is limited to 20 kHz without a buffer on a single channel. The sampling resolution is 16bit and they support a high range for the voltage from 0 V-10 V

and for currents of up to 500 mA. Using a sample rate of 500 kHz the buffer – that can hold up to 448000 samples – is filled up after only 1.1 seconds. As a consequence, long term measurements of multiple seconds or minutes can only be performed in the slow, unbuffered mode at 20 kHz. With costs of about \$300 [23], slow continuous sample rates of 20 kHz and only a single measurement channel, the SNMD would not be applicable for use in our testbed.

Several more energy meters have been presented, like Multichannel Energy Measurement Device (MEMD) by Zhu et al. [20]. Their idea is to use a multi layer architecture to provide power for MCU, transceiver and sensor differently to get better knowledge of these parts. While the general approach seems solid, they only use a buffer of 1900 samples per channel which is filled fairly fast at samples rates between 5.5 kHz (unbuffered) and 150 kHz (buffered). The MEMD uses a serial interface for communication which automatically leads to slow data rates when saving data or using the unbuffered mode.

Finally, Prayati et al. [24] present the *Power Consumption Measuring System* (PCMS). This is a current-mirror based solution for creating a power consumption model of the TelosB processor board. It has six independent A/D convertes, that can sample at a rate of 50 kHz. For measuring currents it uses a pretty expensive device that is placed between the measurement platform and a PC on the other side. Unfortunately this solution is pretty expensive as the device itself costs about \$460.

The following Table I summarizes the features of the presented energy meters. A comparison to our PotatoScope is also given within this table.

TABLE I: Comparison of energy meters

Name	Channels	f_s [kHz]*	Resolution	Range [mA]	Duration
SNMD	1	400 (20)	16bit	0-150/200/500	1s
MEMD	4	150 (5.5)	10bit	0-40	< 1s
PCMS	6	40	16bit	0-50	-
PS**	2	500	12bit	0-26.6	2.5h

*Buffered (Unbuffered)

**PotatoScope

Firstly, it can be seen that sampling for more than a few seconds is either not possible or limits the the user to very unacceptably low sampling rates. In addition, none of the previously presented energy meters is able to provide a reliable energy measurement for challenging environmental conditions – e.g. outdoor deployments. Up to now temperature effects have not been taken into account and to the best of our knowledge we present the first examination of temperature effects on energy meters in Wireless Sensor Network (WSN) testbeds.

III. POTATONET TESTBED

This section gives a brief overview about our existing outdoor testbed PotatoNet [13]. The testbed was built in the light of Smart Farming applications, so that it is located

in a rural area with poor network connectivity and limited maintainability. Thus, PotatoNet was designed to be robust and to ensure that the testbed can survive for extended periods of times without needing local maintenance from specialists.

A central management server (Central Box – embedded Linux PC) is used to manage and monitor all connected nodes. The nodes themselves are connected to the Central Box via a WRTnode¹ – a \$25 MIPS-based embedded Linux board with 680 MHz – that runs the embedded Linux distribution OpenWRT². The WRTnode supports both an Ethernet and a Wi-Fi interface, of which we use the first one in combination with power-over-ethernet (PoE) to provide permanent power supply. The actual sensor node [25] is connected to WRTnode GPIOs providing a full-fledged In System Programmer (ISP). All nodes are connected via Ethernet cables between 30 and 80 meters to the Central Box. If needed, nodes can be concatenated by using the internal Ethernet switch of the WRTnodes. Redundant uplinks at the Central Box allow remote access to the nodes for maintenance or reprogramming. A condensed block diagram of the testbed is shown in Figure 1.

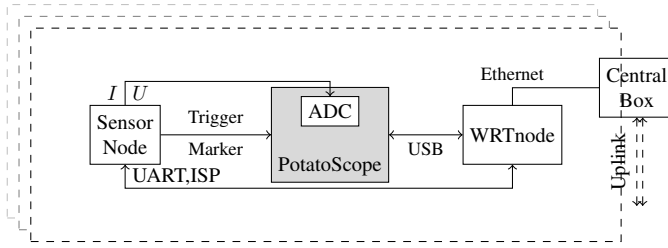


Fig. 1: PotatoNet architecture

In order to provide a distributed energy measurement the PotatoNet is extended by the PotatoScope – an MCU-based oscilloscope. After analog preprocessing the ADC of the PotatoScope’s MCU is used to capture the node’s voltage and current consumption. The sensor node itself can use dedicated GPIOs to start and stop (trigger) measurements. Moreover, GPIOs can be used as markers which allows tagging of the sensor node’s source code. Finally, the USB connection between WRTnode and the PotatoScope is used for changing the configuration, starting measurements and data exchange. The potentially large bulk of data is stored on the PotatoScope itself and can be transferred via ethernet to the central box on request. A more detailed description of the PotatoScope is given in the following section.

IV. POTATOSCOPE

On the one hand the PotatoScope should provide an accurate, fine-grained and temperature invariant sensing of the node’s energy consumption. On the other hand the costs for one PotatoScope must be relatively inexpensive to equip as many nodes as possible. Hence, a well balanced selection of the involved components is needed to fulfil these opposing requirements.

¹<http://wrtnode.com>

²<http://openwrt.org>

The used MCU is an ARM Cortex M3 implementation from ST’s STM32F205 series [26]. It keeps the costs low while allowing the attachment of an external, temperature-stable reference voltage $V_{ref} = 2.5V$. For communication with the WRTnode, we use the external USB 2.0 Phy *Microchip USB3300*³. Current and voltage measurements can be taken in parallel, using two different ADC channels. The detailed architecture of the PotatoScope is shown in Figure 2.

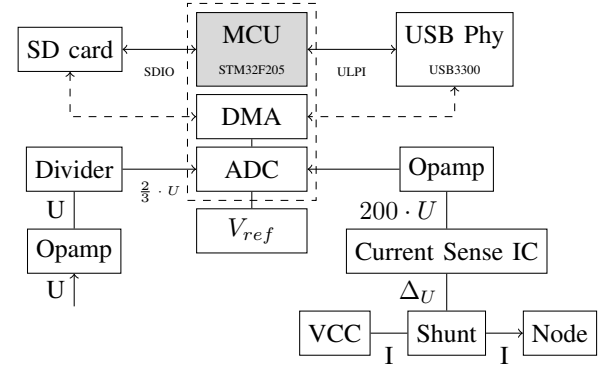


Fig. 2: Detailed architecture of the PotatoScope

The analog front-end of the PotatoScope is designed to exhibit a low temperature drift.

To prevent drawing current from the node, the voltage input channel is connected to an operational amplifier (impedance converter). A downstream voltage divider with a factor of $\frac{2}{3}$ is used to divide the input voltage of 3.3V to match the reference voltage.

For the current implementation we use a high precision shunt with 0.47Ω . Thus a current consumption from 0mA to 26.6mA can be measured. This range can be changed by replacing the shunt resistor. The V_{CC} is passed through the scope and the shunt so we can measure the voltage drop ΔU at the resistor. The potential difference is amplified by a current sense Integrated Circuit (IC) that outputs the voltage with a gain of 200. Due to the current drain of the ADC’s sample-and-hold stage, a small current will be drawn for every single conversion from the input channels. Hence, another impedance converter is used to avoid small but measurable drifts at different sample rates.

A. DMA Capabilities

For data transfers involving the SD card and the USB interface, we use the integrated Direct Memory Access (DMA) Controller. Thus, a deterministic sampling is guaranteed and the workload of the MCU is reduced. In parallel to ongoing measurements the MCU is able to receive and handle control messages from the USB interface.

B. Trigger and Markers

As shown in Figure 1, the nodes themselves are able to trigger measurements by using GPIOs. However, these signals

³<http://www.microchip.com/USB3300>

cannot only be used to start or stop a measurement, but can also be used as *markers*. The node pulls these signals *high* or *low* to tag a position within a running measurement. This can be used to identify states in the node's software, e.g. radio on/off and allows a fine-grained analysis of a nodes power consumption down to the level of specific lines of code. Marker positions are highly precise as they are saved subsequently after conversion whenever one of the signal has been changed.

C. Analog-Digital Converter

The ADC operates in the *dual regular* mode. Both ADC channels that are used for measuring voltage and current of the node are measured in parallel and temporally stored in a 4 bytes wide special register. The upper half-word is used for ADC channel 2 and the lower half-word is used for holding the data of channel 1. After the end of conversion (EOC), the DMA controller transfers the register's data to a circular SRAM buffer. When either half of the buffer is full (or the measurement has been stopped), a DMA transfer will be started, to store the data on the SD card. At the moment we use a circular buffer with a total size of 30720 bytes or 7680 samples. The seamless forwarding of data from the buffer to the SD card allows continuous sampling over long periods of time.

A dedicated timer is used to sample at a fixed rate. Overall there are 24 predefined sample rates (1 Hz - 500 kHz) that are preconfigured within the PotatoScope. Selecting a sample rate will automatically configure the timer and set the optimum hold (cycle) time for the ADC. As a result the noise is reduced and the most accurate measurement at a given sample rate is achieved.

The internal ADC has a resolution of 12 bit. This means that our reference voltage of 2.5V will be divided into 4096 levels with steps of $\frac{2.5V}{2^{12}-1} = 0.611mV$ between them. The resolution for the voltage is hereby given by $0.611mV \cdot \frac{3}{2} = 0.917mV$. The resolution for the current measurement is equal to $\frac{0.611mV}{200 \cdot 0.47\Omega} \approx 6.5\mu A$. Additionally, the ability to measure at high sample rates allows oversampling to achieve even higher resolutions by software.

D. Data Storage

Measuring the energy consumption at relatively high sample rates leads to a large amount of data. As the system is distributed and should scale with an increasing number of nodes, the data will be stored locally using an SD card. At the moment SD cards with a capacity of 8GB are used, which lasts for almost 2.5 hours of continuous measuring at a sample rate of 500 kHz. We do not use a file system like FATFS as it is not applicable for real-time applications [27].

In our implementation, we use one SD card block called *Data Description Block (DDB)* to describe subsequent measurement data. The data format is shown in Figure 3. This block is initially stored in block number 1 on the SD card and can describe 62 consecutive measurements. For each measurement, we store the number of samples and the sample

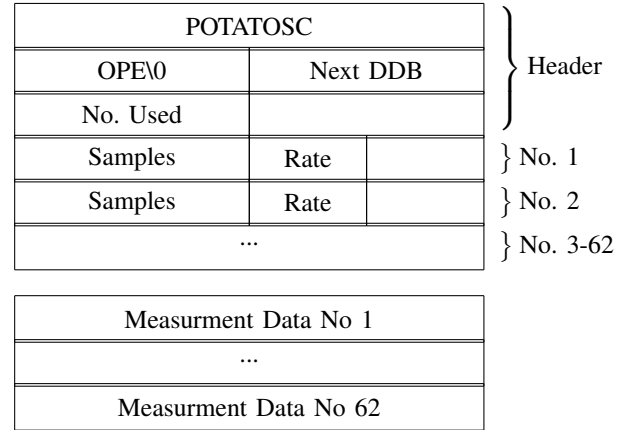


Fig. 3: Low Level Data Structure

rate. Finally, if 62 measurements have been taken, a new DDB will be stored after the data and the *Next DDB* field will be updated. Storing the absolute block number of the next DDB makes it easy to iterate over measurement information.

Besides storing measurement data on the SD card, we use the first block to store calibration results for reducing offset errors, as the processor does not have an integrated EEPROM.

E. Measurement Modes

The PotatoScope can be used in different operating modes. As described above, the *triggered* mode allows nodes to control energy measurements on their own. Triggers can be used to start right in time at an important event, e.g. package reception. The *single shot* mode can be used to arm a trigger, when waiting for an event. Triggering can be done from the node itself or manually by the WRTnode. Last but not least, a *live* mode allows even higher sample rates of up to the maximum ADC frequency of 2 MHz and the use of the PotatoScope with a regular computer outside the testbed.

F. Command Line Interface

Basically a simple libusb-based *command line interface* (CLI) is used to control the PotatoScope or read status information. The CLI is cross-compiled for the WRTnode, while its main purpose is to hide the complexity of the underlying USB interface from the user and to make it easy-to-use. In total three USB interfaces are implemented: requesting and reading data, changing settings and a third one that is responsible to send notification to the USB host to indicate the end of a measurement run. These interfaces are separated so that they can be used simultaneously, because USB requires an interface to be claimed (locked) before being used.

V. ERROR ESTIMATION

Due to requirements of an inexpensive oscilloscope and a temperature stable sensing, components have to be chosen wisely. Maintaining good quality of results while using low-cost components is a balance between accuracy and cost. Therefore, this section gives a short overview of how to model

the theoretical error of the PotatoScope. Based on the chosen components we give an estimate of the PotatoScope's error.

As shown in Section IV the external voltage reference and the ADC is used in both measurements. Almost all errors can be expressed as a general offset or tolerance ϵ_{offset} and the temperature drift per 100 Kelvin ϵ_{drift} . The error of the voltage reference V_{ref} is given in the following Equation 1.

$$\begin{aligned}\epsilon_{vref} &= \epsilon_{offset} + \epsilon_{drift} \\ &= 0.04\% + 0.06\% \\ &= 0.1\%\end{aligned}\quad (1)$$

According to the datasheet of the MCU, the ADC has an accuracy of 5 LSB.

With an ADC resolution of 12 bit this is equal to $\epsilon_{ADC} = \frac{5}{4095} = 0.122\%$. Subsequently the error for the analog front-end can be calculated. The error of the operational amplifier is given below:

$$\begin{aligned}\epsilon_{opamp} &= \epsilon_{offset} + \epsilon_{drift} \\ &= 3.2mV + 0 \\ &\approx 0.13\%\end{aligned}$$

The estimation for the voltage measurement path is given in Equation 2:

$$\begin{aligned}\epsilon_{voltage} &= \epsilon_{div} + (\epsilon_{opamp} + \epsilon_{ADC} + \epsilon_{vref}) \\ &=: \epsilon_{offset} + \epsilon_{common} \\ &\approx 2.36\%\end{aligned}\quad (2)$$

The voltage error $\epsilon_{voltage}$ consists of the common error of the operational amplifier ϵ_{opamp} , the ADC error ϵ_{ADC} and the voltage reference error ϵ_{vref} . Finally, the voltage divider error ϵ_{div} is added to the common error – that is simply equal to 2% as we use two resistors with a precision of 1%.

$$\begin{aligned}\epsilon_{current} &= \epsilon_{shunt} + \epsilon_{ic} + (\epsilon_{opamp} + \epsilon_{ADC} + \epsilon_{vref}) \\ &=: \epsilon_{offset} + \epsilon_{common} \\ &\approx 2.21\%\end{aligned}\quad (3)$$

The error of the current input channel uses the same common error and adds the tolerance of the shunt ϵ_{shunt} and the gain error of the current sense IC ϵ_{ic} to it. In conclusion the final error is shown in Table II. In total it is 2.36% for the voltage measurement and about 2.21% for the current measurement. It should be mentioned that the temperature drift $\frac{ppm}{K}$ for 100 K is already considered for each estimation.

To reduce the errors ϵ_{offset} and ϵ_{opamp} , a one-time calibration is performed. To get the digital value from an analog voltage, a linear relation between input and output value can

be formulated. The calibration uses this fact to calculate the two parameters of a line of the form $l(x) = m \cdot x + b$. As there might be noise on the measurement, three points are taken to determine the slope and intercept more precisely. Due to the calibration process, the total error could be reduced to $< 1\%$ for both channels.

VI. EVALUATION OF THE POTATOSCOPE

Before being deployed in PotatoNet, the PotatoScope has been evaluated in terms of accuracy and temperature invariance.

As introduced in Section V, the maximum theoretical error of about 2.5% can be reduced using a one-time calibration. Our calibration shows comparable results across the nodes and results in a measurement error $\epsilon_{voltage} \leq 0.87\%$ and $\epsilon_{current} \leq 1.56\%$.

The ADC accuracy is a lot worse than 5 LSB, as measured by ST Microelectronics (STM) itself [28]. Their measurements show dispersion of values of up to 21 LSB.

A. Noise

We applied a voltage to the circuit and observed a remarkable ADC dispersion. Results show that ADC values vary within a range of up to 60 LSB. This random noise is caused by several sources, e.g. lab conditions, power supply or other components of the PotatoScope itself. However, this means that values scatter ± 30 LSB around the expected value which corresponds to an error of 0.7% when considering a resolution of 12 bit. From this it follows that we cannot reduce errors below 0.7% due to noise in the signal chain.

B. USB 2.0 Interface

Evaluating the USB 2.0 interface is important for knowing limits of potential data transfer rates. Especially in the live mode, the data throughput might be a bottleneck. Figure 4 shows the results taken from several measurements.

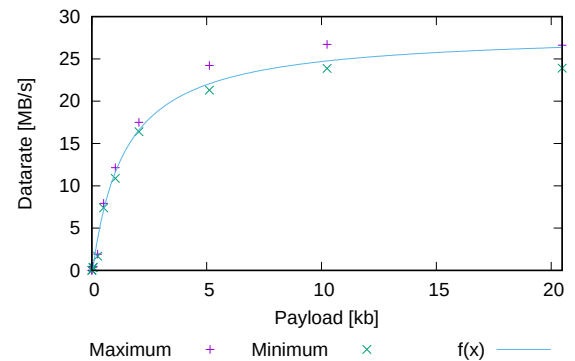


Fig. 4: Measurement of USB 2.0 throughput at different payload sizes

For this evaluation data of different payloads were transferred to the host. The total number of bytes received within a second were counted. Our evaluation shows a data rate of up to 26.5

TABLE II: Summary of the estimated errors (without calibration)

	ϵ_{vref}	ϵ_{ADC}	ϵ_{opamp}	ϵ_{offset}	Error
$\epsilon_{voltage}$	0.1%	0.12%	0.13%	2.00%	2.36%
$\epsilon_{current}$				1.84%	2.21%

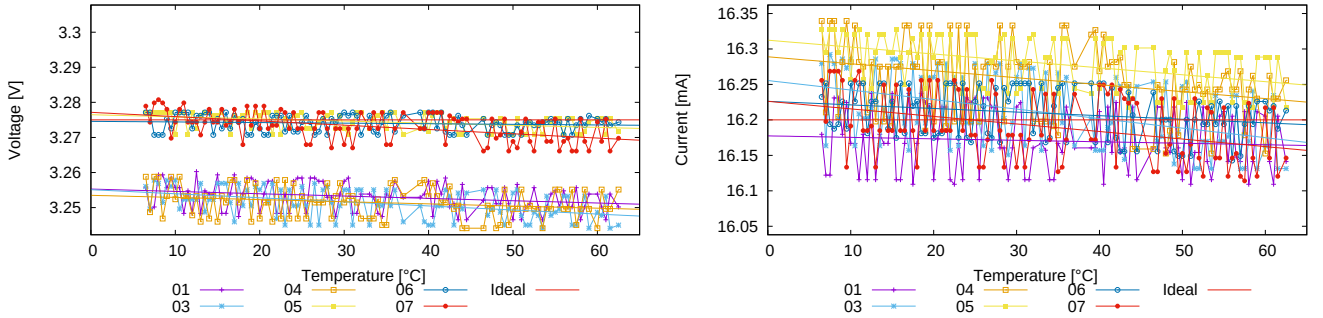


Fig. 5: Temperature drift of a set of nodes

MByte/s at a payload of 20kB. This is only about 45% of the theoretical maximum transfer rate of 480 MBit/s of USB 2.0, but still more than enough to transfer data at transfer rates of up to 2 MHz which requires a data rate of only 8MB/s.

C. Temperature Drift

The operation of WSNs in outdoor applications can lead to extreme temperature variations. Thus, a reliable measuring for a widespread temperature range has the highest priority. By using a climatic chamber, we also measured the impact of the temperature on the current and voltage measurement in practice. The results are depicted in Figure 5, where every node was calibrated beforehand. A small drift downwards within a range of 50 °C can be observed. The linear regression allows to approximate the total drift over a temperature range of 100K. The results for six nodes are shown in Table III.

TABLE III: Voltage and current drift over a range of 100K

Node	Δ_V	Δ_I
01	6.632mV 0.2%	0.020mA 0.13%
03	11.579mV 0.35%	0.134mA 0.83%
04	6.126mV 0.18%	0.098mA 0.61%
05	6.015mV 0.18%	0.097mA 0.60%
06	1.705mV 0.05%	0.050mA 0.31%
07	12.235mV 0.37%	0.106mA 0.66%

The maximum drift for voltage measurements is 0.37% or 12.235mV. For the current, the drift is bigger with 0.83% and 0.134mA.

VII. DISTRIBUTED ENERGY MEASUREMENT

When measuring across a large number of nodes, data amount becomes larger with an increasing number of nodes. A theoretical consideration of the data rate R as the function of the sample rate f_s and the number of nodes n is given in Equation 4.

$$R(f_s, n) = f_s \cdot 4\text{byte} \cdot n \left[\frac{\text{byte}}{s} \right] \quad (4)$$

Considering the data rate of $R = 100\text{Mbit/s}$ and a sample rate of $f_s = 500\text{kHz}$. As a consequence, only $n = 6$ nodes

could be used simultaneously before the entire network would be congested.

Thus transferring data directly to the Central Box is not an option. The solution is to leave data on the SD card of each PotatoScope and keep an index of all measurements across the nodes on a central server. This requires a database – like SQLite – to be used for indexing. Actual measurements can be downloaded from the nodes on demand by using the central database. The advantage is, that amount of storage available increases with every node added to the system.

A. General Architecture

Figure 6 shows the general architecture of the distributed energy measurement. All PotatoScopes are connected to the central server via the WRTnode. The question might come up why we do not use a SSH tunnel to interface the PotatoScope by using the CLI. Unfortunately the delay of a SSH connection is relatively high (few seconds), as the WRTnode is less performant. This would not allow a convenient handling of the energy measurement e.g. configuring or starting measurements immediately.

Another fundamental requirements are reliability and fail-over solutions as e.g. cables might suffer from external damages. When there is no active connection between the WRTnode and the central server, a fall-back solution is to keep a local log file of finished measurements. The list of these measurements will be transferred once the connection is (re-)established again. With this technique, pre-configured measurements can be performed, even if the central server is not reachable for some reason.

The core architecture consists of a simple client-server-architecture. Instead of the SSH tunnel, a more lightweight TCP connection is used to reduce the transmission delays between PotatoScope and central server. A ping message is

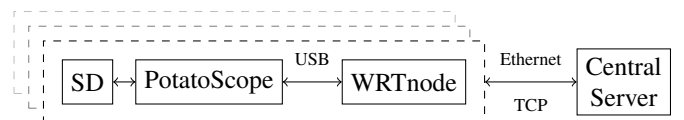


Fig. 6: Architecture of the distributed energy measurement system

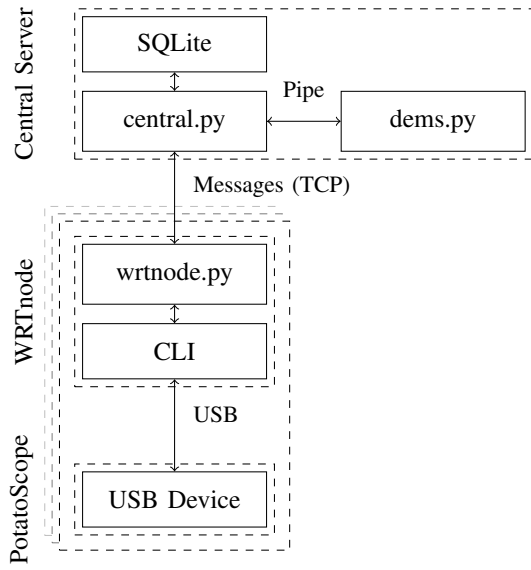


Fig. 7: Software architecture of our system

send to each node periodically to check whether a PotatoScope is still available.

B. Software implementation

The detailed software architecture of the entire system is shown in Figure 7.

On the bottom layer, the PotatoScope acts as a USB device to the WRTnode. The CLI, as presented in Section IV, is used to interface the PotatoScope by the WRTnode. On the one side a python script (wrtnode.py) is wrapped around the CLI to be responsible for calling the CLI on the WRTnode. On the other side the same script establishes, handles and maintains the TCP connection to the central server. As a measurement can end at every point in time, the notification interface is used, to wait for notifications permanently. Due to the fact that different USB interfaces are implemented (cf. Section IV), waiting for notifications and interaction with the PotatoScope e.g. reading data, can be performed concurrently.

The Central Box executes the central server where the nodes are connected to. This is where the user accesses the system. A further python script (central.py) is responsible for handling notifications, e.g. finished measurements. Moreover it provides a pipe for communication with the user interface. The communication uses serialized python classes, as the overhead is reasonable, but the gain of easy extendability is much higher. New types of messages can be implemented very easy, without knowledge of the underlying system.

The third script (dems.py) provides the distributed energy measurement system (DEMS) to the user. Using a pipe for communication, the dems can access the TCP connection indirectly and thus send data to the nodes, such as requesting the current configuration or starting measurements. Sampled data can be requested directly or a combination of command

line options can be used for starting a measurement and reading the data afterwards. A simple call to the system allows to query used space of the SD cards, or a list of measurements across all nodes. For graphical representation of measurements *gnuplot*⁴ into the system. This allows for plotting results directly on the central server. Downloading raw measurement data can take fairly long, as the capacity of the uplink might be limited.

As an extension the grouping of nodes is supported. This features makes it easy to start (different) measurements on a predefined subset of the nodes. Nodes are identified over the hostname. Using groups of nodes, different users can use the testbed at the same time, or nodes can be grouped by functionalities or applications running on them.

Two separate SQLite databases allow to parallel storage of measurement data and groups of nodes. This allows that groups can be exchanged, while keeping the list of data when different measurements are performed within the testbeds. A user can simply use its own group of nodes, upload applications and go ahead with measuring data.

The following listing demonstrates how a measurement can be taken on a group of the nodes. The first three parameters configure the scope to sample in single shot mode, at 10kHz sampling rate and to stop after 10000 samples (1 second). Finally, when the measurement has been completed, the data will be drawn into a figure.

```
dems -m S -r 10000 -l 10000 --plot -g group
```

Throughout the design of the system, we took special care to use clear interfaces. This enables us to exchange parts of our testbed. For example, management nodes or messages within the system can be easily extended or replaced.

C. Evaluation

To perform initial evaluations of our system, we emulate a smaller version of the testbed in our lab. A schematic view of the setup is depict in Figure 9, while a photo of this test arrangement is shown in Figure 8a.

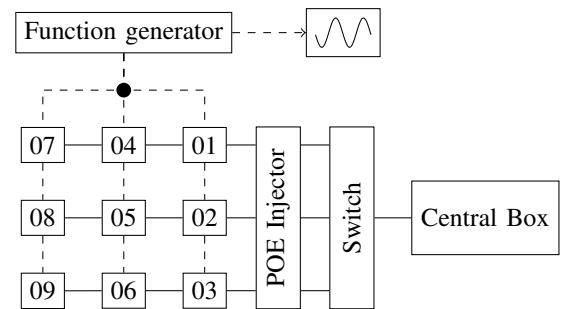


Fig. 9: Setup for the evaluation of our system

Instead measuring the energy characteristics of a sensor node, the PotatoScopes are connected to a signal generator. The advantage of an alternating test signal is, that we are

⁴<http://www.gnuplot.info/>

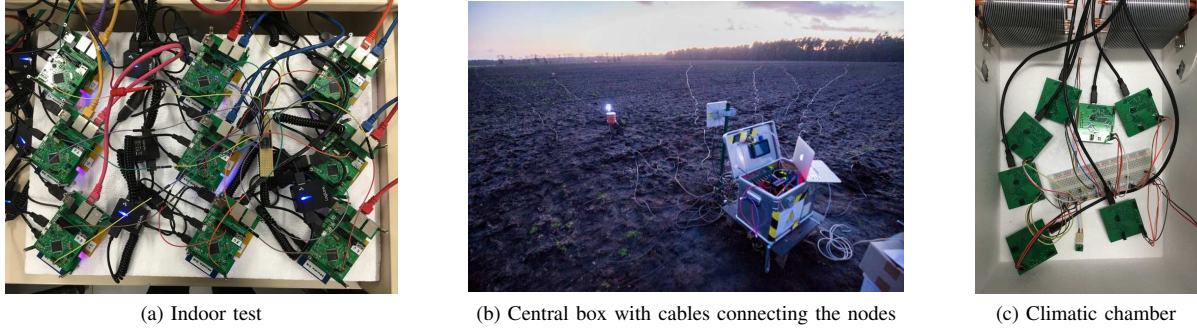


Fig. 8: Pictures before and after installation

therefore able to measure differences in the amplitude and in the phase shift between nodes. A total of nine nodes are used for this evaluation. Using a star connection provides the same test signal to all nodes. For power supply, we also use a POE injector here that provides a constant power supply and the Ethernet connection to the nodes. Three nodes are concatenated as groups of three. Finally the Central Box is connected to the nodes by using an ethernet switch.

For the evaluation we use a low frequency sine wave. Our evaluation result is shown in Figure 10. It shows a small offset in amplitude, that has also been seen in the temperature drift measurements. The phase difference is surprisingly small with only 30ms, considering the fact that we do not use a common triggering mechanism. The triggering happens by sending a custom message to all nodes using the TCP connection. Nevertheless, further synchronization between measurements can be achieved by using the markers (cf. Section IV-B).

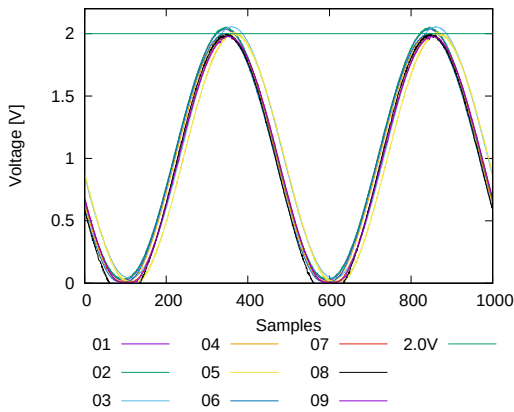


Fig. 10: Result of the evaluation of our distributed energy measurement system

VIII. CONCLUSION

In this paper, we presented a solution for distributed energy measurements in WSN testbeds. To the best of our knowledge, we are the first presenting an outdoor testbed, that supports

testing applications of many kinds in actual outdoor environments while reliably measuring energy for research purposes. Most important was a dependable method for measuring energy on a high number of nodes. For this purpose, we designed the PotatoScope, a low-cost, high precision and temperature invariant MCU-based oscilloscope. PotatoScope can measure voltages from 0 V to 3.75 V and currents in the range of 0 mA to 26.6 mA. Using sample rates of up to 500 kHz is absolute sufficient to characterize the energy consumption of WSN applications.

An theoretical estimation of measurement accuracy as well as practical evaluations within a climatic chamber showed that the PotatoScope is well suited to be used in outdoor applications. The total error plus the drift of energy measurements due to widespread temperatures within a range of 100 K is quite small ($< 1.56\%$).

A convenient distributed energy measurement system (dems) was implemented on top of the distributed PotatoScopes. The advantage, that the storage of measurements is also distributed among the nodes leads to a highly scalable system design.

Finally the entire system has been rated by measuring test signals across all nodes in our lab. Recently the distributed energy measurement system was applied to our testbed (cf. Figure 8b) to be used for ongoing research in WSN outdoor applications.

REFERENCES

- [1] K. Yi, R. Feng, N. Yu, and P. Chen, "Pared: A testbed with parallel reprogramming and multi-channel debugging for wsns," in *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*, April 2013, pp. 4630–4635.
- [2] P. Humi, M. Anwender, G. Wagenknecht, T. Staub, and T. Braun, "Tarwis – a testbed management architecture for wireless sensor network testbeds," in *Network Operations and Management Symposium (NOMS), 2012 IEEE*, April 2012, pp. 611–614.
- [3] G. Werner-Allen, P. Swieskowski, and M. Welsh, "Motelab: a wireless sensor network testbed," in *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, April 2005, pp. 483–488.
- [4] A. Dlodla, A. Abu-Mahfouz, C. Kruger, and J. Isaac, "Wireless sensor networks testbed: Asntbed," in *IST-Africa Conference and Exhibition (IST-Africa), 2013*, May 2013, pp. 1–10.

- [5] J. Albesa, R. Casas, M. Penella, and M. Gasulla, "Realnet: An environmental wsn testbed," in *Sensor Technologies and Applications, 2007. SensorComm 2007. International Conference on*, Oct 2007, pp. 502–507.
- [6] C. Boano, M. Zuniga, J. Brown, U. Roedig, C. Keppitiyagama, and K. Romer, "Templab: A testbed infrastructure to study the impact of temperature on wireless sensor networks," in *Information Processing in Sensor Networks, IPSN-14 Proceedings of the 13th International Symposium on*, April 2014, pp. 95–106.
- [7] J. Sheu, C. Chang, and W. Yang, "A distributed wireless sensor network testbed with energy consumption estimation," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 6, no. 2, pp. 63–74, Jul. 2010. [Online]. Available: <http://dx.doi.org/10.1504/IJAHUC.2010.034321>
- [8] K. Bannister, G. Giorgetti, and E. K. S. Gupta, "Wireless sensor networking for "hot" applications: Effects of temperature on signal strength, data collection and localization."
- [9] C. A. Boano, H. Wennerstrom, M. A. Zúñiga, J. Brown, C. Keppitiyagama, F. Oppermann, U. Roedig, L.-A. Norden, T. Voigt, and K. Roßmer, "Hot packets: A systematic evaluation of the effect of temperature on low power wireless transceivers," 2013.
- [10] C. A. Boano, N. Tsiftes, T. Voigt, J. Brown, and U. Roedig, "The impact of temperature on outdoor industrial sensor network applications," *Industrial Informatics, IEEE Transactions on*, vol. 6, no. 3, pp. 451–459, 2010.
- [11] U. Kulau, F. Busching, and L. Wolf, "Undervolting in wsns: Theory and practice," *Internet of Things Journal, IEEE*, vol. 2, no. 3, pp. 190–198, June 2015.
- [12] U. Kulau, S. Schildt, S. Rottmann, and L. Wolf, "Paint it black: Increase wsn energy efficiency with the right housing," in *Proceedings of the 6th ACM Workshop on Real World Wireless Sensor Networks*, ser. RealWSN '15. New York, NY, USA: ACM, 2015, pp. 3–6. [Online]. Available: <http://doi.acm.org/10.1145/2820990.2820992>
- [13] U. Kulau, S. Schildt, S. Rottmann, B. Gernert, and L. Wolf, "Demo: PotatoNet – Robust Outdoor Testbed for WSNs: Experiment Like on Your Desk. outside," in *Proceedings of the 10th ACM MobiCom Workshop on Challenged Networks*, ser. CHANTS '15. Paris, France: ACM, Sep. 2015, pp. 59 – 60. [Online]. Available: <http://doi.acm.org/10.1145/2799371.2799374>
- [14] L. Feeney, C. Rohner, P. Gunningberg, A. Lindgren, and L. Andersson, "How do the dynamics of battery discharge affect sensor lifetime?" in *Wireless On-demand Network Systems and Services (WONS), 2014 11th Annual Conference on*, April 2014, pp. 49–56.
- [15] J.-P. Sheu, C.-C. Chang, and W.-S. Yang, "A distributed wireless sensor network testbed with energy consumption estimation," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 6, no. 2, pp. 63–74, 2010.
- [16] A. Dunkels, J. Eriksson, N. Finne, and N. Tsiftes, "Powertrace: Network-level power profiling for low-power wireless networks," 2011.
- [17] Q. Li, M. Martins, O. Gnawali, and R. Fonseca, "On the effectiveness of energy metering on every node," in *Proceedings of the 2013 IEEE International Conference on Distributed Computing in Sensor Systems*, ser. DCOSS '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 231–240. [Online]. Available: <http://dx.doi.org/10.1109/DCOSS.2013.67>
- [18] R. Zhou and G. Xing, "Nemo: A high-fidelity noninvasive power meter system for wireless sensor networks," in *Information Processing in Sensor Networks (IPSN), 2013 ACM/IEEE International Conference on*. IEEE, 2013, pp. 141–152.
- [19] A. Hergenroeder, J. Wilke, and D. Meier, "Distributed energy measurements in wsn testbeds with a sensor node management device (snmd)," in *Architecture of Computing Systems (ARCS), 2010 23rd International Conference on*, Feb 2010, pp. 1–7.
- [20] N. Zhu and I. O'Connor, "Energy measurements and evaluations on high data rate and ultra low power wsn node," in *Networking, Sensing and Control (ICNSC), 2013 10th IEEE International Conference on*, April 2013, pp. 232–236.
- [21] A. Kopke and A. Wolisz, "Measuring the node energy consumption in usb based wsn testbeds," in *Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems Workshops*, ser. ICDCSW '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 333–338. [Online]. Available: <http://dx.doi.org/10.1109/ICDCSW.2008.53>
- [22] X. Jiang, P. Dutta, D. Culler, and I. Stoica, "Micro power meter for energy monitoring of wireless sensor networks at scale," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, April 2007, pp. 186–195.
- [23] P. Hurni, B. Nyffenegger, T. Braun, and A. Hergenroeder, "On the accuracy of software-based energy estimation techniques," in *Proceedings of the 8th European Conference on Wireless Sensor Networks*, ser. EWSN '11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 49–64. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1966251.1966257>
- [24] A. Prayati, C. Antonopoulos, T. Stoyanova, C. Koulamas, and G. Papadopoulos, "A modeling approach on the telosb wsn platform power consumption," *J. Syst. Softw.*, vol. 83, no. 8, pp. 1355–1363, Aug. 2010. [Online]. Available: <http://dx.doi.org/10.1016/j.jss.2010.01.015>
- [25] F. Büsching, U. Kulau, and L. Wolf, "Architecture and evaluation of inga - an inexpensive node for general applications," in *Sensors, 2012 IEEE*. Taipei, Taiwan: IEEE, oct. 2012, pp. 842–845. [Online]. Available: <http://www.ibr.cs.tu-bs.de/papers/buesching-sensors2012.pdf>
- [26] STMicroelectronics. (2015-11-20) Stm32f205xx and stm32f207xx datasheet. [Online]. Available: <http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/CD00237391.pdf>
- [27] S. Schildt, W.-B. Pottner, F. Busching, and L. Wolf, "Ratfat: Real-time fat for cooperative multitasking environments in wsns," in *Distributed Computing in Sensor Systems (DCOSS), 2013 IEEE International Conference on*, May 2013, pp. 388–393.
- [28] STMicroelectronics. (2015-11-20) How to improve adc accuracy when using stm32f2xx and stm32f4xx microcontrollers. [Online]. Available: http://www.st.com/web/en/resource/technical/document/application_note/DM00050879.pdf

