On the Lifetime and Usability of Environmental Monitoring Wireless Sensor Networks

Sebastian Bader

Supervisor: Professor Bengt Oelmann

Department of Electronics Design
Mid Sweden University, SE-851 70 Sundsvall, Sweden

ISSN 1652-893X
Mid Sweden University Doctoral Thesis 161

ISBN 978-91-87557-04-0
On the Lifetime and Usability of Environmental Monitoring Wireless Sensor Networks

Sebastian Bader

©Sebastian Bader, 2013

Department of Electronics Design
Mid Sweden University, SE-851 70 Sundsvall
Sweden

Telephone: +46 (0)60 148495

Printed by Kopieringen Mittuniversitetet, Sundsvall, Sweden, 2013
Abstract

Wireless sensor networks have been demonstrated, at an early stage in their development, to be a useful measurement technology for environmental monitoring applications. Based on their independence from existing infrastructures, wireless sensor networks can be deployed in virtually any location and provide sensor samples in a spatial and temporal resolution, which otherwise would only be achievable at high cost or involve significant work by humans. The feasibility of the usage of wireless sensor networks in real-world applications, however, is only maintained if certain technological challenges are overcome. Amongst these challenges, are the limited lifetime of the distributed sensor nodes, and user interfaces, which allow for the technology to be utilized in an efficient manner. Contributions to the solution of these challenges have been the objective of this thesis.

After an analysis of the contributions wireless sensor networks can provide to the application domain of environmental monitoring, and the introduction to the restrictions, which are posed by a limited operational lifetime and low system usability, these issues are addressed at the system level of sensor node devices.

The lifetime of sensor nodes, which is closely linked to the lifetime of the complete wireless sensor network, is addressed with regards to the energy efficiency of nodes, as well as the utilization of solar energy harvesting in order to increase the available energy resources. With respect to energy efficiency, an analysis has been performed of the contributions to the energy consumption of environmental monitoring sensor nodes, which leads to the desire to minimize the nodes’ duty cycles and quiescent currents. A sensor node design is presented, which features energy efficiency as a key attribute by utilizing modern semiconductor architectures. Moreover, an argument for the usage of synchronization-based, contention-free communication is made in order to reduce active communication periods and, thus, the duty cycle of a sensor node. A synchronization method with its focus on low protocol
Abstract

overhead is introduced as a basis for such communication forms.

After an initial feasibility study in relation to using battery-less solar energy harvesting architectures in locations with limited solar irradiation, multiple architectural implementations are analyzed in a comparative manner. Among these comparisons is an analysis of short-term energy storage devices in the form of double-layer capacitors and thin-film batteries, which provide prolonged component lifetimes than those for conventional secondary batteries, but which can only buffer for short periods of time due to their limited energy capacity. In order to be able to dimension such energy harvesting systems with respect to the individual application constraints at hand, state of charge simulations are proposed. A method for such simulations is presented and demonstrated for the implementation of an energy harvester model on a component basis. While the modeling in this manner is time consuming, the model can predict the state of charge of the energy buffer in the architecture with a high level of accuracy. Finally, a method for the systematic evaluation of solar energy harvesting architectures is presented. The presented method can be summarized as a solar energy harvesting testbed, which utilizes configurable energy harvesting circuits in order to create a deploy-once-test-many type of system. The output results of this testbed can significantly improve the efficiency of architecture comparisons and system modeling.

Contributions to the improvement of the usability of wireless sensor nodes are made on two separate levels, namely, developer usability and end user usability. A method for the programming of sensor nodes based on hierarchical finite state machines is presented, which improves the usability of software development by creating familiarity for technically experienced users. Moreover, the utilization of finite state machine principles allows for the software to be developed in a systematic and modular manner. As implemented applications typically require to be verified, which, in the environmental monitoring domain, usually results in outdoor deployments, usability considerations for sensor nodes are presented, which can simplify this process. Special attention has been paid in order for these improvements to be achieved with low overheads.

While software development is a familiar concept for most system developers, this is not the case for the end users of these systems, who are typically domain experts. In order to allow for wireless sensor nodes to be operated by domain experts, a method for the configuration of sensor nodes has been
proposed. The method uses a combination of graphical specification of the node behavior and a configurable sensor node. The evaluation of this method, which has been based on a proof-of-concept implementation, demonstrated that the performance can remain high, while end users, without technical experience, are enabled to configure sensor nodes without prior training.

In summary, the contributions, presented in this thesis, address system lifetime and usability with regards to the sensor node level. The results have led to the implementation of an energy efficient sensor node, which allows for the operation from battery-less solar energy harvesting sources. Furthermore, support tools for the implementation of these nodes, both on the hardware and software level, have been proposed.
Acknowledgements

During the last years, there have been a number of people, who were, both
directly and indirectly, involved in the process that has led to the work pre-
presented in this thesis. I would like to take the opportunity, at this point, to
show my gratitude.

First and foremost, I want to thank my main supervisor, Professor Bengt
Oelmann, who has given me the opportunity to conduct this work and who
has supported me during the years. I am very grateful for your guidance, but
also the freedom and trust, which made my work develop the way it did.

I thank my colleagues at the Electronics Design Department for creating a
great work environment. In particular, I want to thank Peng Cheng, Stefan
Haller, David Krapohl and Najeem Lawal, who I can always count on to be
available for interesting discussions. I also thank Johan Sidén, who has taken
the time to review this work.

I thank Matthias Krämer, Torsten Schölzel, Matthias Anneken and Manuel
Goldbeck, who have been involved in individual projects that have con-
tributed to the work that is presented in this document. Particular apprecia-
tion goes to Matthias Krämer, who has been a great support in the technical
implementation of a number of systems and who always shared my fascina-
tion for the technical development in wireless embedded systems.

Furthermore, I am grateful to Michael Brünig and Wen Hu, who have
given me the chance to broaden my horizon and who have been supervising
me during my stay at CSIRO. I also like to thank everyone at the Autonomous
Systems Laboratory for turning my stay in Australia into a great experience.

Finally, I thank my family and friends for bringing balance to my life. I
am very grateful to know that you are there and that I can count on your
support.
## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSMA</td>
<td>Carrier Sense Multiple Access</td>
</tr>
<tr>
<td>CTS</td>
<td>Clear-to-Send</td>
</tr>
<tr>
<td>DLC</td>
<td>Double Layer Capacitor</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processor</td>
</tr>
<tr>
<td>EPR</td>
<td>Equivalent Parallel Resistance</td>
</tr>
<tr>
<td>FDMA</td>
<td>Frequency Division Multiple Access</td>
</tr>
<tr>
<td>FPGA</td>
<td>Field Programmable Gate Array</td>
</tr>
<tr>
<td>FSM</td>
<td>Finite State Machine</td>
</tr>
<tr>
<td>FTSP</td>
<td>Flooding Time Synchronization Protocol</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communications</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HAL</td>
<td>Hardware Abstraction Layer</td>
</tr>
<tr>
<td>HFSM</td>
<td>Hierarchical Finite State Machine</td>
</tr>
<tr>
<td>IC</td>
<td>Integrated Circuit</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>ISM</td>
<td>Industrial, Scientific and Medical</td>
</tr>
</tbody>
</table>
**Acronyms**

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISR</td>
<td>Interrupt Service Routine</td>
</tr>
<tr>
<td>MCU</td>
<td>Micro Controller Unit</td>
</tr>
<tr>
<td>MPP</td>
<td>Maximum Power Point</td>
</tr>
<tr>
<td>MPPT</td>
<td>Maximum Power Point Tracking</td>
</tr>
<tr>
<td>NiCd</td>
<td>Nickel-Cadmium</td>
</tr>
<tr>
<td>NiMH</td>
<td>Nickel-Metal Hydride</td>
</tr>
<tr>
<td>LDO</td>
<td>Low Dropout</td>
</tr>
<tr>
<td>LOS</td>
<td>Line of Sight</td>
</tr>
<tr>
<td>LTS</td>
<td>Lightweight Time Synchronization</td>
</tr>
<tr>
<td>Li-Ion</td>
<td>Lithium-Ion</td>
</tr>
<tr>
<td>PBS</td>
<td>Pairwise Broadcast Synchronization</td>
</tr>
<tr>
<td>PCB</td>
<td>Printed Circuit Board</td>
</tr>
<tr>
<td>PnO</td>
<td>Perturb-and-Observe</td>
</tr>
<tr>
<td>RBS</td>
<td>Reference Broadcast Synchronization</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RTC</td>
<td>Real-Time Clock</td>
</tr>
<tr>
<td>RTS</td>
<td>Request-to-Send</td>
</tr>
<tr>
<td>SFD</td>
<td>Start Frame Delimiter</td>
</tr>
<tr>
<td>SMPS</td>
<td>Switch Mode Power Supply</td>
</tr>
<tr>
<td>SOC</td>
<td>State-of-Charge</td>
</tr>
<tr>
<td>TDMA</td>
<td>Time Division Multiple Access</td>
</tr>
<tr>
<td>TDP</td>
<td>Time Diffusion Protocol</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>TFB</td>
<td>Thin-Film Battery</td>
</tr>
<tr>
<td>TPSN</td>
<td>Timing-Sync Protocol for Sensor Networks</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WSN</td>
<td>Wireless Sensor Network</td>
</tr>
</tbody>
</table>
## Contents

Abstract i

Acknowledgements v

Acronyms vii

List of Figures xiii

List of Tables xvii

List of Papers xix

1 Introduction 1

1.1 Environmental Monitoring 2

1.2 Wireless Sensor Networks 3

1.2.1 Environmental Monitoring with Wireless Sensor Networks 6

1.2.2 Application Scenario under Scope 8

1.3 Case Study - SAQnet 10

1.3.1 Background, Architecture and Implementation 11

1.3.2 Project Analysis 15

1.4 Application Analysis 18

1.5 Problem Formulation 20

1.6 Thesis Contributions 21

1.7 Thesis Outline 22

2 Energy Efficiency 25

2.1 Wireless Sensor Network Lifetime 25

2.2 Wireless Sensor Node Architecture 26
Contents

2.3 Duty-Cycling ........................................ 30
2.4 Energy-efficient Sensor Node Design ......................... 35
  2.4.1 SENTIO-em Architecture and Implementation ........ 36
  2.4.2 SENTIO-em Evaluation .......................... 41
2.5 Energy-efficient Communication ............................ 45
  2.5.1 Time Synchronization for Wireless Sensor Networks 47
  2.5.2 Synchronized Duty-Cycling ....................... 54

3 Solar Energy Harvesting 67
  3.1 Solar Energy Harvesting ................................ 67
    3.1.1 Energy Neutral Operation .......................... 69
  3.2 Solar Energy Harvesting Architectures .................... 71
    3.2.1 Low Irradiance Conditions ....................... 77
    3.2.2 Short-Term Energy Buffers ....................... 83
  3.3 Solar Energy Harvesting Dimensioning ..................... 90
    3.3.1 Modeling of the Solar Energy Harvesting Architecture 91
    3.3.2 Simulating Available Energy Levels ............. 95
  3.4 Solar Energy Harvesting Testbed ........................ 98
    3.4.1 Architecture and Implementation ................ 101
    3.4.2 Setup and Results .............................. 105

4 Usability ............................................. 111
  4.1 Usability in Wireless Sensor Networks .................... 111
  4.2 Usability for Developers ................................ 112
    4.2.1 Programming with Hierarchical Finite State Machines 113
    4.2.2 Usability Considerations for Sensor Nodes ........ 119
  4.3 Usability for End Users ................................ 123
    4.3.1 Concealing Node Programming Complexity .......... 124

5 Discussion and Conclusions ................................ 131

Bibliography .......................................... 139

Papers .................................................. 149
# List of Figures

1.1 Comparison of environmental monitoring sampling techniques .................................. 4
1.2 Network architecture of applications under scope ............................................. 9
1.3 SAQnet system architecture ........................................................................ 12
1.4 SAQnet node architecture ........................................................................... 12
1.5 Deployment overview of the SAQnet system ................................................... 14
1.6 Energy profile of a SAQnet node .................................................................. 15
1.7 Packaging of a SAQnet node ....................................................................... 17

2.1 General sensor node architecture ................................................................. 27
2.2 Graphical representation of the duty-cycling principle ..................................... 31
2.3 Duty-cycling effect on the overall current consumption .................................. 35
2.4 Architectural structure of the SENTIO-em platform .......................................... 37
2.5 Temperature influence on the SENTIO-em quiescent currents ...................... 43
2.6 Temperature influence on the SENTIO-em communication currents ............. 43
2.7 Evaluation of the processing performance of SENTIO-em .............................. 44
2.8 Energy efficiency comparison of sensor node platforms .................................. 45
2.9 Structure of a typical TDMA protocol ............................................................ 47
2.10 Occurrence of delay components during a packet transmission ................... 52
2.11 Unilateral and bidirectional synchronization methods ..................................... 53
2.12 Interval-measurement based synchronization .................................................. 56
2.13 Master-Slave synchronization procedure implementation .............................. 59
2.14 Accuracy results of the synchronization method ............................................. 60
2.15 Temperature dependency of wake-up accuracy (oven) ................................... 61
2.16 Temperature dependency of wake-up accuracy (freezer) ............................... 62
2.17 In-box temperature of a plastic enclosure in direct sunlight ........................... 63
2.18 Comparison of temperature drift behavior .................................................... 63
2.19 Power consumption of temperature drift compensation .............................. 66
List of Figures

3.1 Simplified energy supply time from an AA-type battery . . . 68
3.2 Energy harvesting system overview ............................. 71
3.3 Energy density and power density of energy storage devices . 72
3.4 Characteristics of a small-scale solar panel ...................... 74
3.5 Accuracy of MPPT based on fractional open-circuit voltage and fractional short-circuit current ......................... 76
3.6 Annual solar irradiance distribution in Sundsvall, Sweden . . 78
3.7 Directly coupled energy harvesting circuit architecture . . . 79
3.8 Ideal and real zener-diode characteristics ....................... 80
3.9 System behavior of the solar energy harvesting architecture during one week of deployment ......................... 82
3.10 Architecture of the configurable energy harvesting module . . 85
3.11 Experimental setup for short-term energy storage architecture evaluation ................................................. 86
3.12 Comparison of the charge behavior of DLCs and TFBs ....... 86
3.13 Peak discharge behavior of double layer capacitors and thin-film batteries ................................................. 87
3.14 Configurable solar energy harvester results (summer) ....... 88
3.15 Configurable solar energy harvester results (winter) ......... 88
3.16 MPPT utilization during periods of low irradiation .......... 90
3.17 Design constraints of solar energy harvesting systems ...... 91
3.18 System model overview for solar energy harvesting simulations ......................... 92
3.19 Equivalent circuit of a solar cell ................................ 93
3.20 Equivalent circuit of a DLC ...................................... 94
3.21 Overview of the solar energy harvesting system model on high abstraction level ................................. 95
3.22 Solar energy harvesting system model evaluation .......... 95
3.23 Exemplar output of solar energy harvesting simulation sweeps 97
3.24 Conceptual overview of the solar energy harvesting testbed 100
3.25 Layered architecture of the solar energy harvesting testbed 101
3.26 Architecture of the configurable energy harvesting module for the testbed nodes ................................. 102
3.27 Timeline of the testbed communication protocol .......... 104
3.28 Implementation of the configurable solar energy harvesting testbed nodes ................................. 105
List of Figures

3.29 Energy profile of a sensor node with sampling and communication tasks ........................................... 106
3.30 Comparison of ideal and measured load emulation for the configurable load profiles on testbed nodes .......... 107
3.31 Energy profile of a testbed node in operation ................................................................. 108
3.32 Output results of the solar energy harvesting testbed ................................................................. 109
4.1 State-graph representation of a sample-and-send application ...................................................... 114
4.2 Layered architecture of the HFSM framework implementation ...................................................... 116
4.3 Program flow of the execution of state transitions ........................................................................ 117
4.4 Picture of the SENTIO-em platform implementation ................................................................. 121
4.5 Conventional process of sensor system implementations ............................................................ 125
4.6 Adapted sensor system implementation process ............................................................................ 125
4.7 State-graph of the configurable sensor node firmware ................................................................... 127
4.8 Implementation of the configurable sensor node design ............................................................... 128
**List of Tables**

1.1 Component implementation of SAQnet nodes ........ 13
1.2 Environmental monitoring application analysis results .... 19

2.1 Operation states and consumption levels of typical sensor node modules ............... 32
2.2 Consumption parameters of SENTIO-em and SAQnet nodes 36
2.3 Microcontroller comparison for SENTIO-em ............... 38
2.4 Radio transceiver comparison for SENTIO-em .............. 40
2.5 SENTIO-em current consumption measurements ............ 42
2.6 Transition times between operation states of SENTIO-em .... 44
2.7 Energy consumption sources in wireless communication .... 48
2.8 Wireless sensor network transmission delay factors ........ 51
2.9 Main classification factors of time synchronization algorithms in Wireless Sensor Networks ............... 55
2.10 Summary of statistical results obtained from measurements shown in Figure 2.14 ............... 60
2.11 Parameters for the energy consumption estimation of the temperature compensated synchronization ............... 65

3.1 Typical energy harvesting sources and their power levels available in outdoor environments ............... 69
3.2 Characteristics of typically used battery types ............. 73
3.3 Overview of measured characterization parameters of a 450 mW silicon solar panel ............... 76
3.4 Implementation of the DLC-based harvesting architecture .... 81
3.5 Property comparison of DLC and TFB energy storage devices 83
3.6 Dimensioning parameters for solar energy harvesting systems 96
3.7 Comparison of properties of conventional and SEH testbeds 99
List of Tables

4.1 Memory allocation of exemplar HFSM application implementations ............................. 118
4.2 State transition and interrupt handling latencies ........................................ 119
4.3 Page overview of the specification wizard .................................................. 126
List of Papers

This thesis is based on the following publications:

Paper I  **SAQnet: Experiences from the Design of an Air Pollution Monitoring System based on Off-the-Shelf Equipment**  
Sebastian Bader, Matthias Anneken, Manuel Goldbeck, Bengt Oelmann  
*2011 Seventh International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*

Paper II  **A Domain-Specific Platform for Research in Environmental Wireless Sensor Networks**  
Sebastian Bader, Matthias Krämer, Bengt Oelmann  
Accepted in *2013 Seventh International Conference on Sensor Technologies and Applications (SENSORCOMM)*

Paper III  **Adaptive Synchronization for Duty Cycling in Environmental Wireless Sensor Networks**  
Sebastian Bader, Bengt Oelmann  
*2009 Fifth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*

Paper IV  **Durable Solar Energy Harvesting from Limited Ambient Energy Income**  
Sebastian Bader, Bengt Oelmann  
*International Journal On Advances in Networks and Services, 2011*

Sebastian Bader, Bengt Oelmann  
*2013 IEEE Tenth International Conference on Networking, Sensing and Control*
List of Papers

Sebastian Bader, Torsten Schölzel, Bengt Oelmann
2010 IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)

Sebastian Bader, Matthias Krämer, Bengt Oelmann
In manuscript for IEEE Sensors Journal

Paper VIII  Implementing Wireless Sensor Network Applications using Hierarchical Finite State Machines
Matthias Krämer, Sebastian Bader, Bengt Oelmann
2013 IEEE Tenth International Conference on Networking, Sensing and Control

Paper IX  Concealing the Complexity of Node Programming in Wireless Sensor Networks
Sebastian Bader, Bengt Oelmann
2013 IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)

Related papers not included in this thesis:

Sebastian Bader, Bengt Oelmann
2010 Fourth International Conference on Sensor Technologies and Applications (SENSORCOMM)

  Challenges for RF Two-Way Time-of-Flight Ranging in Wireless Sensor Networks
Sebastian Bader, Bengt Oelmann, Michael Brünig
2012 IEEE 37th Conference on Local Computer Networks Workshops (LCN Workshops)
Sensor technologies now play an ever increasing role in modern society. Providing devices with an interface to their physical environment, allows for process optimization, new manners of human-device-interaction, as well as new device functionalities. Mobile phones, for example, contain a large number of sensors, which provide, amongst other functionalities, capacitive input, accelerometer-based screen orientation, multiple audio and video detectors, global navigation, and adaptive screen brightness. Combining these physical interfaces with considerable computational performance and both, short and long range communication, has transformed, what was previously a mobile telephone, into a personal digital companion, typically referred to as a smartphone. This new device class interconnects about one seventh\(^1\) of the world population and this is, additionally, nearly independent of location or time.

While smartphones are primarily devices, which provide networking for their owners, other concepts envision the actual interconnection of devices themselves. One of such concepts, often titled the Internet of Things (IoT), is currently finding its way into reality. The underlying idea of an IoT is to provide communication interfaces to everyday devices, enabling them to transmit and receive information. A typically advertised scenario includes home automation, in which a network of devices can increase comfort in relation to daily living situations. An alarm clock, for example, could cause the heating and light in the bathroom to power on, and shortly after, initiate the brewing of the morning coffee. Thus, increasing both comfort and energy efficiency. Many of such examples are based on a detect-and-react scheme, in which one event causes a reaction. Because of the network interfaces, however, event and reaction are not restricted to occur within the same device.

\(^1\)According to market analyses in 2012
1 Introduction

In both of these technology examples, providing existing devices with sensing, communication and processing capabilities, leads to extended functionalities and ways of device interaction. Similarly, creating standalone systems with the same capabilities allows for the achievement of equivalent features, even if there is no existing device for these interfaces to be integrated into. These standalone systems are called Wireless Sensor Networks (WSNs) and have been utilized in a multitude of different application domains. One of such is environmental monitoring, the monitoring of natural environments.

1.1 Environmental Monitoring

Although environmental monitoring can mean the monitoring of any kind of environment, it is most often used to describe the observation and study of natural environments. The foundation of environmental monitoring is the collection of data, which enables a better understanding of our natural surrounding to be gained by means of observation. Scientifically, environmental monitoring includes the fields of Physics, Chemistry and Biology. However, as more technologies, especially for data acquisition, become involved, so do the number of technical fields of study.

Originating from an ever increasing world population, environmental monitoring is not limited to the pure understanding of environments, but also includes monitoring for preservation reasons. Environmental monitoring plays a key-role for showing the effects of human behavior on the environment and in disclosing its limits. Typical applications, in addition to purely environmental science purposes, include the protection of water supplies, radioactive waste treatment, air pollution monitoring, natural resource protection, weather forecasting and the enumeration and monitoring of species [1].

Environmental monitoring strives to determine the status of a changing environment by analyzing a representative sample of the environment. As such, data acquisition forms a major part of environmental monitoring. The data acquisition system in use has to allow for the collection of representative samples, which includes concerns such as the intrusiveness of the measurement system itself, sampling accuracy or sample storage. The impact of these concerns depends, on the one hand, on the application (e.g., the sensitivity
of the observed physical value to be externally influenced) and, on the other hand, on the type of data acquisition system used.

Typical sampling techniques are grab samples, (remote) sampling stations and remote sensing. Grab sampling is the manual removal of a sample from the environment for further analysis. While this is not performed as frequently as was the case previously due to technological progress, in some cases, grab sampling is still used to allow for random sampling or more complex analysis than is possible in the field. One immense drawback of grab sampling is the involvement of human beings, which leads to time and cost issues in addition to significant invasiveness. Sampling stations refer to sensor systems deployed in the environment of interest, monitoring the surrounding continuously or in defined intervals. These systems store the measurement samples taken locally or transfer them by means of GSM or satellite communication, thus significantly reducing the amount of human labor. Remote sensing usually means the sensing of environments from a distance by means of a satellite or aircraft, involving imagery or radiation detection. Because of its distant operation, remote sensing allows for the coverage of large areas and the monitoring of inaccessible or dangerous environments, but it does, however, commonly lack local resolution and does not allow for the use of contact-based sampling.

A comparison between the typical properties of these sampling techniques can be found in Figure 1.1. The graphic illustrates the analysis of each of the previously named techniques with respect to the parameters, which are typically the basis for the selection of an appropriate sampling method. The analysis result is depicted on a five-value scale from very low to very high, and thus, visually summarizes the strengths and weaknesses of each technique respectively.

### 1.2 Wireless Sensor Networks

Although distributed sensor networks have been used previously [2], Wireless Sensor Networks, as we know them today, originate from research projects in the mid 1990s. Since that time, WSNs have developed through efforts in both, academic and industrial environments, and they have been applied in a tremendous number of different application domains. Its widespread
Figure 1.1: Visual comparison of the properties of traditional measurement techniques in environmental monitoring. Eight different properties influencing the manner of operation are compared on a five-step scale (i.e., very low to very high).
application is a result of the general necessity for sensor technology in measurement, monitoring and control tasks, as well as the flexibility gained by the combination of this sensor technology with wireless communication and software-based processing.

While the implementation of a Wireless Sensor Network is greatly dependent on the application requirements of the task to be performed, a fundamentally similar architecture among different WSNs remains. A Wireless Sensor Network is composed of multiple wireless sensor nodes\(^2\), which typically combine sensing, processing and wireless communication capabilities. The hardware implementation of these sensor nodes is, in the majority of cases, based on specific wireless sensor node platforms, which implement the desired functionalities. Whereas other hardware platforms can provide the required functionalities (e.g., mobile phones), wireless sensor node platforms are more specifically designed with the restrictions and requirements of Wireless Sensor Networks in mind.

Depending on the applicationscenario and thenetwork topology that has been selected, the individual sensor nodes will communicate with the other nodes (or a specific subset), in order to form the Wireless Sensor Network. Furthermore, the nodes within the WSN can have identical functionalities and tasks (i.e., a homogeneous WSN), or their tasks and functionalities can be complementary (i.e., a heterogeneous WSN).

The main property of a Wireless Sensor Network is its capability of performing distributed sensing tasks within the same system. This enables a selection of features, which are not possible with a single device system. Typically applied examples of these features include

- Collection of sensor data at different times and locations.

- Transfer of data throughout the network to a remote location, which is different to the location where the data was collected.

- Data fusion to deduct information, which cannot be obtained from data taken at a single location.

---

\(^2\)In the remainder of this document the terms wireless sensor node, sensor node, wireless node and node will be treated synonymously, and will always refer to wireless sensor nodes.
In order to utilize these features fully, however, several generally desired system constraints are imposed on Wireless Sensor Networks. Sensor nodes, for example, should be of a small physical size, which maintains their placement flexibility. The implementation of each node should require a low cost, so as to result in an economically feasible system cost even when large number of nodes are required by the application. Moreover, sensor nodes should, at least to a large extent, operate in a maintenance-free manner.

These constraints have an influence on the implementation of Wireless Sensor Networks. As a result, the typical node platforms are resource-limited devices, which allows for their cost and power consumption to be maintained at a low level. This, in turn, enables networks with a high number of sensor nodes, but which require an optimization of the necessary functions to be performed, to match the available resources.

Although Wireless Sensor Networks can be used in many different application scenarios and, thus, can be perceived as different types of systems, this document will treat WSNs as a measurement instrument. Furthermore, the remainder of this document will be restricted to the utilization of this measurement instrument in the domain of environmental monitoring.

1.2.1 Environmental Monitoring with Wireless Sensor Networks

Environmental monitoring applications are based on the development from data to information to knowledge [1]. Hence, the more meaningful data that is obtained, the more knowledge that can be derived. Because data is gathered through measurement and observation, the measurement system capabilities of WSNs offer several advantages to the field of environmental monitoring. Probably the most fundamental is the autonomy of data aggregation. While traditional sampling methods demand increased labor input for larger amounts of samples (e.g., sampling at several locations in the same area), an ideal WSN observes the environment at multiple locations and automatically transmits the data to a gathering point via the networked infrastructure. Furthermore, the autonomous sampling allows for the unobtrusive observation of phenomena and for monitoring in harsh locations and under extreme conditions [3]. Because the sensing networks are usually directly connected to the operator via the Internet or some type of local connection,
1.2 Wireless Sensor Networks

data is gathered in real-time or near-real-time. This enables problems to be detected at an earlier stage than would be possible in systems with local storage and manual downloading at the end of an acquisition period. In addition, the remote connection to the sensor network means the removal of distance between the scientist and the monitored site [3], as the researcher can directly observe what is happening at a particular area of interest.

Nevertheless, environmental monitoring is an extensive area and different applications impose different requirements on the Wireless Sensor Network. A rough, but useful, classification of WSN-applications is made by Barrenetxea et al., who divide these systems into time-driven, event-driven and query-driven sensor networks [4]. However, as sensing in most applications is time-driven (e.g., by continuous or periodically sampling of the attached sensing devices), the classification mainly describes the communication activity in the system. Within these, time-driven applications usually transmit their sensor readings periodically, which is typically used in data gathering applications, such as [5, 6]. However, event-driven sensor networks attempt to minimize the actual ongoing traffic and the flooding of the gathering point with meaningless information. These types of systems observe the area of interest, transmitting sensor information only on those occasions when particular events occur, such as a fire [7] or volcanic eruption [8]. Query-driven systems store gathered information locally and communicate it on request. This type of sensor network can, for example, be useful in logistics or home applications, but is not very common in applications relating to environmental monitoring.

The system requirements of the different application classes differ tremendously. While it is the case that time-driven sensor networks are usually more predictable, especially in terms of network traffic, event-driven applications behave more randomly and in an unforeseeable manner. Because of these differences, it is not usual for system designs, particularly in communication protocols, for both classes to be interchangeable. Therefore, the presented work in this document is limited to time-driven applications in the environmental monitoring domain. A more detailed definition of the application scenario, which is targeted by this work, is described below.
1 Introduction

1.2.2 Application Scenario under Scope

Due to the broad applicability of Wireless Sensor Networks, as well as its highly application-specific implementation, this section is used to define the type of applications addressed by the work that is presented in the remainder of this document. While a large set of design parameters has been made based on the application scenario under scrutiny, the majority of presented results can, directly or with slight modifications, be utilized in other application scenarios within the WSN design space.

The application scenario primarily addressed in this work involves data gathering Wireless Sensor Networks. These applications are manifold in the field of environmental monitoring and are usually used to aggregate large sets of data, which originate at different locations over an extended period of time. According to the WSN taxonomy introduced by Mottola et al. in [9], these applications are referred to as sample-and-send applications. Examples include studies of water quality [10], microclimates of glaciers [4, 11] or rainforests [5], as well as the measurement of light intensity under shrubs [12]. The data collected in these applications is typically utilized as the input in order to provide a better understanding, modeling or site status observation.

An additional restriction is made in the manner that the focus lies on time-driven applications, where physical values are measured and gathered continuously (or semi-continuously). As opposed to the event-driven sensor networks, time-driven systems possess only limited data processing within the sensor node units. While event-driven applications communicate the occurrence of events, and therefore, in the majority of cases, require to translate data into information before any communication can occur, time-driven applications typically transfer pure sensor data. The drawback associated with time-driven applications is in relation to their increased amount of communication, which is, in most implementation forms, more costly than processing. Using WSNs in environmental monitoring as a scientific measurement tool, however, provides the low-level sensor data with increased value.

As a result of this, typical wireless sensor nodes in the network can be rather simple devices, thus reducing system complexity, size and cost. Additionally, large amounts of these simple devices are necessary in order to cover large-scale areas, while maintaining the required spatial resolution. The number
of nodes opens up several challenges which must be addressed in these applications, such as the handling and visualization of data, communication and configuration protocols, as well as the deployment and maintenance of the network. Further details with reference to these research challenges will be given in Section 1.5.

Figure 1.2 shows a possible network architecture for data gathering WSNs in environmental monitoring, and the type of network architecture we address in this work. As many environmental monitoring systems are deployed in remote locations (or at least at a distance from the operator), a distinction between the local and global domain is made. The local sensor field contains the sensor nodes, which are monitoring the deployment site, while being indirectly connected to the global domain via some kind of gateway (i.e., a local to global communication translator).

In the depicted architecture example, the sensor field is organized in a
1 Introduction

cluster-star topology, which requires two different node types, namely, cluster heads and end nodes. Although this is not the sole implementation option, several design considerations have led us to the belief that this is an appropriate choice for the application at hand. Details on the motivation for this architectural choice will be given in Section 2.5.

The global domain usually includes a data processing and storage unit, such as a data server. From here, data is converted into a representable form and made available to the operator of the network and environmental researchers, typically via the Internet. Based on the distance between the sensor field and the data storage that can be found in the majority of WSNs, the link between the local and global domain is, in many cases, implemented using long-distance communication technologies, such as GSM/GPRS. In other cases, a pure translation from the communication methods used within the Wireless Sensor Network to those used in computer networks may be sufficient.

1.3 Case Study - SAQnet

SAQnet (Sundsvall’s Air Quality network) has been an experimental application for the usage of Wireless Sensor Networks in environmental monitoring, which has been conducted at Mid Sweden University during 2011. The results that have been obtained from conducting this project were twofold. On the one hand, experiences on the general – and also application-specific – development and deployment of WSN systems has been gained. On the other hand, the application scenario has been used in order to evaluate the state of the art off-the-shelf sensor network equipment. These results have supported the research decisions that have been made prior to this study, as well as providing new directions, which have subsequently been addressed. Thus, this case study acts as a motivator towards the study of the research problem, which is formally stated in Section 1.5.

The case study is presented in the next two sections. Firstly, Section 1.3.1 will provide the technical background, which includes the motivation, architectural choices and the implementation of the application. Then, in Section 1.3.2, the experiences and analysis results of the development and final system will be presented. We will finally establish a link between these results and open research problems within the area of WSNs.
1.3.1 Background, Architecture and Implementation

As a result of urbanization and industrialization, the amount of air pollutants produced by human sources increases and the majority of the world population is exposed to these pollutants on a daily basis. The monitoring of the air quality becomes a crucial activity in order to establish and maintain regulatory thresholds, which will protect us from harmful consequences. Furthermore, monitoring can be used to analyze the production and distribution processes of the air pollutants, as well as to relate the concentrations to external parameters. Because the pollution levels are affected by internal parameters (e.g., the number of cars driving on a road), as well as external parameters (e.g., the wind speed and direction), they have high spatial and temporal variations. While currently applied monitoring stations provide high accuracy and a good temporal resolution, a sufficient spatial resolution is rarely accomplished, which is a result of their high monetary costs. Applying WSN technology allows there to be an increase in the spatial resolution of air pollution monitoring, however, typically at the cost of per-location sampling accuracy.

In order to establish a measurement of the outdoor air quality, the parameters to be monitored in the SAQnet system are primarily gas concentrations. Nevertheless, temperature and humidity levels have been collected as reference parameters alongside the gas level measurements. The system architecture follows a standard sample-and-send scenario as described in Section 1.2.2. The resulting system design can, therefore, be split into three system layers, which are the sensing layer, the storage layer, and the data access layer. This layered system organization, together with the components each layer consists of, is depicted in Figure 1.3.

The actual monitoring task is performed by a number of sensor nodes, which collect a set of environmental parameters in a time-driven manner. The design of these nodes follows a rather conventional architecture, which contains a low-power microcontroller, a Radio Frequency (RF) transceiver, sensors, and a battery-based power supply (see Figure 1.4). As was previously mentioned, the system implementation was intended to be largely based on off-the-shelf equipment. Thus, the sensor node has been implemented on the basis of a commercial node platform. Table 1.1 lists the individual implementation choices for the node’s components. As relatively long distances
1 Introduction

Figure 1.3: The abstracted architecture of the SAQnet system, divided into three functional layers.

Figure 1.4: The high-level architecture of an SAQnet node. Each node includes computing, communication, sensing and power modules.

between sensor nodes were expected, an RF module with increased output power was chosen. This enables there to be a coarser node deployment at the cost of an increased communication power consumption. The selection of the sensors has been based on parameters of interest in air pollution monitoring, as well as the sensor availability for the chosen node platform. Furthermore, the initial node design did use a high capacity primary battery as its energy source (i.e., power option A in Table 1.1). Based on the limitations in the resulting node lifetime, however, an alternative power option has been considered, which is based on a secondary battery in connection with a 2.5 W solar panel. A more detailed description of the underlying reason for the selection of an alternative power supply will be provided at a later stage.

In order to make use of the sensor data that is sampled by the sensor nodes,
1.3 Case Study - SAQnet

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computing/Control</td>
<td>Platform</td>
<td>Libelium Waspmote</td>
</tr>
<tr>
<td>Communication</td>
<td>Radio Module</td>
<td>Digi XBee 802.15.4 PRO</td>
</tr>
<tr>
<td>Sensing</td>
<td>Temperature</td>
<td>Microchip MCP9700A</td>
</tr>
<tr>
<td></td>
<td>Humidity</td>
<td>Sencera 808H5V5</td>
</tr>
<tr>
<td></td>
<td>CO</td>
<td>Figaro TGS2442</td>
</tr>
<tr>
<td></td>
<td>CO₂</td>
<td>Figaro TGS4161</td>
</tr>
<tr>
<td></td>
<td>NO₂</td>
<td>e2v MiCS-2710</td>
</tr>
<tr>
<td>Power (Option A)</td>
<td>Primary Battery</td>
<td>Saft LSH20 3.7 V – 13 Ah</td>
</tr>
<tr>
<td>Power (Option B)</td>
<td>Secondary Battery</td>
<td>Li-Ion 3.7 V – 6.6 Ah</td>
</tr>
<tr>
<td></td>
<td>Solar Panel</td>
<td>12V - 200mA</td>
</tr>
</tbody>
</table>

Table 1.1: Implementation details of the modules in a SAQnet node.

The data has to be collected, stored, and made available for user access. For this purpose, a gateway node has been implemented, which receives the data of each node and stores it in a MySQL database. The implementation of the gateway is based on the Meshlium platform\(^3\), which operates on a x86 processor. This allows for the usage of standard PC-class operating systems with their respective libraries. The platform, furthermore, provides Ethernet and WLAN, which enables the system to be connected to an TCP/IP network, and, thus, an easy interface to the Internet.

Using the TCP/IP connection, two different user interfaces have been designed. Firstly, data can be accessed from a web interface, which is mainly intended for the end user of such a system. The data obtained from the different sensors and locations is visualized and can be compared based on different parameters. The implementation of a web interface was selected in order to allow a simple and uniform interaction with the system, which does not require a special software installation and can be performed from any location. In order to perform system maintenance, which normal users should not have access to, a Graphical User Interface (GUI) program has been implemented. In addition to the data inspection that can also be performed with the web interface, the GUI allows for maintenance tasks, such as

\(^{3}\)http://www.libelium.com/products/meshlium/
adding, editing and removing of sensor nodes, as well as the access to the raw measurement data.

The system has been evaluated in a small network with five sensor nodes and a gateway in the surroundings of the university campus in Sundsvall. Figure 1.5 provides an overview of the deployment locations, in which each node had a Line of Sight (LOS) condition, at least, to its next communication partner. The nodes sampled their sensors in a fifteen minute interval, and then forwarded the data through the network to the gateway. In order to obtain a reasonable sensor value, the sensors are powered 30 s prior to their actual readout⁴. After the sensor samples have been acquired, the sensor nodes switch to a communication state for a period of 10 s. In this period, the nodes receive a synchronization message, transmit their own data, as well as forward data of other nodes if necessary. Figure 1.6 depicts the resulting energy profile of a SAQnet node. For visualization purposes the sampling interval during the measurement has been reduced to two minutes. Based

⁴According to sensor requirements (http://www.figarosensor.com/)
1.3 Case Study - SAQnet

Figure 1.6: Energy profile of a SAQnet node, which runs a 2 min periodic sampling configuration. Contributions to the consumption are marked with their respective tasks.

on a fifteen minute interval, the average current draw can be estimated to 5.35 mA, of which sensing clearly consumes the largest part.

As a result, the battery powered sensor nodes in the network showed a lifetime of between 9 days and 34 days, which was considerably shorter than a back-of-the-envelope calculation would suggest (i.e., approximately 100 days). The solar powered sensor node, on the other hand, has been providing sensor readings uninterruptedly since its initial deployment.

1.3.2 Project Analysis

In addition to the data that has been collected by the sensor nodes, the system development and implementation did provide insights and experiences with respect to the application of WSNs in the environmental monitoring domain. These qualitative results, which have mainly been gained from dealing with design and implementation challenges, have indicated unsolved problems in the technology to be applied. The main experiences made, which concern both application-specific and general design considerations, are listed below.

1. Sensing can significantly contribute to energy consumption in wireless sensor nodes. While in many applications, simple sensors might be sufficient, which results in the communication being the major energy consumer of the node, this is not always the case. Depending on the sensor types used, the energy consumption due to sensing can easily
1 Introduction

 exceed the communication. In the application scenario presented, the energy consumption due to sensing was more than ten times larger than that of communication.

2. Sensor calibration can be a hindering factor for the development and operation of low-cost autonomous monitoring systems based on Wireless Sensor Networks. As data of spatially distributed sensors should, in many applications, be compared or used collaboratively, the calibration of these sensors becomes of significant importance. Although, similarly to the energy consumption, the necessity for individual calibration is dependent on the sensor type used, the large number of sensors possibly used and the tendency towards low-cost sensor choices, may result in calibration becoming the reason which makes these applications economically infeasible.

3. Batteries are limiting the autonomy of Wireless Sensor Networks. Batteries are a commonly used energy source for wireless sensor nodes, as they deliver power from a reservoir that requires no infrastructure. Their limited capacity, however, means that the usage of batteries limits the lifetime of sensor nodes, and, thus, the complete system. The manual charging or replacement of batteries increases the maintenance to be performed. The number of sensor nodes in the network, as well as their respective energy consumptions, defines the required amount of maintenance. The sensor node lifetime, and the maintenance requirement connected to it, can greatly benefit from the utilization of alternative energy sources. While the battery powered sensor nodes, in the presented system, lasted for between 9 and 34 days on their respective energy reservoirs, the node that utilizes the solar energy harvesting has been sampling data, so far, for approximately two years without interruption.

4. A majority of current sensor node platforms are designed for laboratory/desk usage, but provide limited deployment support. While development functionalities are certainly important in order to simplify the implementation of applications, these functionalities should not restrict the ability to deploy the final system. Furthermore, platforms should consider their easy deployment already in their design. In the
1.3 Case Study - SAQnet

SAQnet deployment an example of this was the connection of sensors. The waspmote platform provides interfaces to sensors via the so called sensor boards. In order to connect sensors to these boards, they contain multiple sockets, which are mainly composed of standard 2.54 mm headers. While this allows for the easy connection of sensors in the laboratory (i.e., they are just placed inside the sockets), preparing the node for deployment becomes tremendously difficult, because sensors have to be wired to these connectors (see Figure 1.7).

5. Wireless Sensor Networks do not provide sufficient end user interfaces. The design, implementation and operation of Wireless Sensor Networks requires a considerable technical understanding by their end users. The typical end users of such a system, however, are domain experts (i.e., in the environmental monitoring domain biologists, chemists, farmers, etc.). It is not uncommon for these end users not to possess the required skills in order to independently work with WSNs. In the majority of existing deployments, this has led to a collaborative work between domain experts and system experts and it is due to their common interest in gaining experiences with these systems, that this collaboration has been possible. Over time, however, the interest in utilizing the system will shift towards the end users, which will lead to additional costs if a system support remains a requirement.
1.4 Application Analysis

Application oriented development has been a common methodology in WSN research and, thus, a plethora of different applications of Wireless Sensor Networks have been reported in the literature. Table 1.2 presents the analysis of a subset of these applications based on the experiences we have encountered in the SAQnet application (see Section 1.3.2).

The selection of the ten projects has been based on multiple criteria. Firstly, all of the presented applications are in the application domain of environmental monitoring. Secondly, the applications are, from an application scenario point of view, close to the application scenario, which is addressed in this work. In order to determine this application similarity, the WSN taxonomy introduced in [9] has been applied. According to this taxonomy, all presented applications are sense-only application with a many-to-one interaction. Furthermore, they are static and data collection occurs in a periodic manner. Finally, applications have been selected based on their level of detail with respect to the description of parameters of interest. These applications have been analyzed with respect to their system lifetime and their usability in relation to the intended end users, as these parameters represent the general issues of those that have previously been described.

Although usability, as a qualitative design parameter, is rarely evaluated in the presented works, its influence can be determined from the manner in which the applications have been conducted. All of the applications that have been presented, have been performed collaboratively by system experts and domain experts. In this collaboration, the system experts have been the provider and the domain experts the consumer of the measurement data. While the domain experts, thus, have been the end user of the WSN system, they have, in none of the presented scenarios, been the operator of the system. Martinez et al. state this as an issue to be addressed, in order to develop Wireless Sensor Networks to become a competitive technology as compared to the state-of-the-art measurement methods in environmental sciences [13, chap. 9].

Despite their differences with regards to their sampling intervals, which are ranging from continuous sampling to a few sensor readings per day, the majority of application examples list lifetime as a restricting system parameter. While the systems operating on solar energy have overcome this limitation
Table 1.2: Analysis results of existing applications in the environmental monitoring domain. The selection process has been based on the applications’ similarity to the application scenario addressed in this work.
by replenishing their energy reservoir, only one battery powered system reports node lifetimes in the order of years [16, 17]. This, however, is only achieved by greatly reducing the sampling rates as compared to the other applications. The discrepancy between the expected and the actual node lifetime, which has been experienced in the SAQnet deployment, has been confirmed multiple times, such as in [15, 18].

As a result, the end user interface and the system lifetime prove to be general constraints in the field of Wireless Sensor Networks for environmental monitoring applications, as opposed to being an application-specific concern. Furthermore, solar energy harvesting has been demonstrated repeatedly as a complementary solution in order to extend the lifetime of battery-powered, outdoor Wireless Sensor Networks.

1.5 Problem Formulation

Wireless Sensor Networks have the possibility to become a new measurement standard in environmental monitoring applications. Enabling autonomous measurements on a large scale, but possibly with a high spatial resolution, makes this technology an attractive solution for manifold problems.

A distributed measurement system, which is composed of a large number of individual but connected devices, however, does not only increase the application possibilities in the environmental monitoring domain, but poses a number of system challenges. These challenges need to be addressed and solved in order for WSNs to fulfill their promised vision. During two decades of system research, WSN technology has been developed considerably. Today, low-power platforms can be bought at low cost, and standardized communication protocols can be obtained from multiple sources. While this allows for an overall low equipment cost, the operational costs of a sensor network remain high, due to the reappearing challenges of end user interfaces and the system lifetime (see Section 1.4).

Although lifetime demands can vary from application to application, a long system lifetime is desirable in most application scenarios in order to reduce maintenance requirements. The majority of implemented applications, however, have demonstrated system lifetimes of the order of weeks (see Table 1.2). In combination with the possibly large number of nodes, this lifetime
restriction leads to high maintenance requirements and, thus, makes the application of battery-based Wireless Sensor Networks in the majority of scenarios, economically infeasible.

The utilization of solar energy harvesting can increase the lifetime of sensor nodes within the network and, thus, increases the lifetime of the network itself. However, this solution will lead to an increase in size and cost of the sensor nodes. The implementation of a solar energy harvesting system should therefore be well dimensioned in order to limit the additional size and cost to an appropriate level.

The second influencing factor with regards to the operational cost is system support. In environmental monitoring applications, the provider and end user of the measurement system are typically not the same. In existing applications, this has been only a limited problem, as the applications have been conducted collaboratively, which means that the system provider was responsible for the system operation. This, however, has only been an option because system experts have been interested in the development and advertisement of WSNs, and, thus, have been funded from their own sources. As Wireless Sensor Networks mature, this interest will decrease and this, in turn will mean that the end users will have to pay for any system support, or, in order to avoid the additional costs, be capable, themselves, of operating the system. End user interfaces are therefore an important part of system design, in order to develop WSNs into a economically competitive measurement instrument.

1.6 Thesis Contributions

In this thesis, both the lifetime restriction and the system usability, have been addressed in order to reduce the operational cost of Wireless Sensor Networks as a measurement instrument. As parts of this research problem have been previously targeted, and is too broad to be solved as a whole, the following list provides the main contributions from this work:

1. Analysis and design of a low-power hardware platform for the implementation of wireless sensor nodes with respect to their lifetime and development usability. (Paper II)
1 Introduction

2. Analysis and experimental verification of synchronized communication as a method to reduce overall energy consumption by allowing efficient resource use, such as low duty cycles. (Paper III)

3. Investigation of micro-scale energy harvesting architectures with a focus on solar energy harvesting in locations with challenging light conditions. (Paper IV)

4. Experimental evaluation of short-term energy storage devices in solar energy harvesting architectures for low-power wireless sensor nodes. (Paper V)

5. Proposition, implementation and analysis of energy level simulations as a tool for dimensioning solar energy harvesting systems. (Paper VI)

6. Design and evaluation of a solar energy harvesting testbed, in order to simplify the comparison of harvesting architectures and the generation of input data for the development of respective simulation models. (Paper VII)

7. Design and implementation of a framework for the programming of wireless sensor nodes based on finite state machines in order to improve development usability for experienced users. (Paper VIII)

8. Implementation and analysis of a method for the programming of wireless sensor nodes by non-technical end users. (Paper IX)

1.7 Thesis Outline

The remainder of this thesis is divided into four chapters. Chapter 2 addresses the energy efficiency of wireless sensor nodes. After an analysis of the lifetime constraints in WSNs and the contributions of node modules and tasks to the energy consumption of the system, a sensor node design with the focus on energy efficiency is presented. Furthermore, energy efficient communication based on synchronous task scheduling is proposed. An appropriate synchronization protocol with low communication overhead is presented and experimentally evaluated.
Chapter 3 deals with the utilization of solar energy harvesting for the improvement of sensor node lifetime. While, particularly, battery-based systems have been used before, the focus of this chapter lies on short-term energy storage based architectures to be analyzed. A number of architectures is experimentally compared and analyzed. Moreover, the modeling and simulation of respective architectures is addressed in order to allow for the systems to be dimensioned with respect to individual application constraints. Finally, a method, which uses an adapted testbed approach, is introduced in order to allow for systematic experimentation and data generation to be performed.

The usability of sensor node devices is addressed in Chapter 4. An argument is made for the division of usability development with respect to developers and system end users. The presented contributions contain a programming method for technically-experienced developers, which utilizes the concept of Finite State Machines (FSMs) in order to improve the systematic and modular development of sensor node software. In addition, sensor node considerations for the improvement of usability are presented and demonstrated on the implementation of a sensor node platform. Finally, end user usability is addressed by the development and implementation of a specification-based configuration approach.

In Chapter 5, the presented contributions are summarized and discussed. Furthermore, conclusions based on the presented results are made.
2 Energy Efficiency

In this chapter, the lifetime of a Wireless Sensor Network will be addressed in terms of the energy efficiency of sensor node operation. We will present an analysis of the lifetime constraints, as well as influencing factors in relation to the lifetime on a sensor node level. In order to improve the lifetime, our contributions address the energy efficiency of task execution on both the hardware and software level.

2.1 Wireless Sensor Network Lifetime

As described in the previous chapter, WSNs enable autonomous sensing on a large scale. Ideally, a large number of sensor nodes are spread over a wide area (possibly at a remote or harsh location), organizing themselves and communicating their sensor data back to the researcher without further attendance. A key parameter for allowing the nodes to fulfill this task is that there must always be a sufficient energy supply. If the system demands the user to re-visit the deployment site too frequently, in order to exchange or replenish the systems’ energy reservoirs, this defeats the initial advantages offered by the system in the first place.

Because of the importance of the system lifetime in the final application, lifetime has been used extensively as an evaluation parameter in the designs at all levels in sensor networks. However, providing a definition for the WSN-lifetime at a general level is very difficult. Several definitions have been proposed in the literature and a comprehensive coverage of these definitions (or definition-classes) is given in [22].

A first classification is usually made by separating Wireless Sensor Network lifetime from sensor node lifetime. While these two parameters are typically connected to each other, they have to be targeted on different levels.

Sensor node liveliness can usually be determined directly from the operational status of a sensor node at a given time. Typically, the lifetime of a
sensor node is defined by its energy reservoir and its energy consumption over time. Limited energy resources are, thus, the only limiting factor for the nodes lifetime (ignoring hardware defects at this point). Nevertheless, operational interruptions can certainly occur during the lifetime of a sensor node, which might require additional consideration.

In comparison, WSN lifetime is not as easy to determine without taking more application constraints into account. Typical definitions are based on the availability of sensor nodes, the coverage of the monitored terrain or the connectivity of the network. In addition, combinations and extensions of these factors might also be placed into the definition of a sensor network lifetime [22]. Since sensor nodes are the building blocks of Wireless Sensor Networks, their lifetimes are connected, but the extent of this connection can vary from application to application.

Different scenarios regarding the WSN lifetime definition based on operating sensor nodes include the n-of-n, k-of-n or m-in-k-of-n scenarios. The first two cases are straightforward solutions, where n-of-n means all nodes have to be alive, while in k-of-n at least k-nodes have to be alive. If the case is not met because more nodes are dead than are allowed, then the lifetime of the entire sensor network is considered as being ended. Different sensor nodes, however, can have tasks with different significance to the system they are operating in. A node that regularly forwards data of other nodes, shows a larger significance to the network than an end node. These cases are included in an m-in-k-of-n scenario, in which at least m critical nodes (e.g., routers) out of k remaining nodes are required for the task execution.

Although WSN lifetime is typically the final parameter of interest, due to its application dependency, the remainder of this document will address the sensor node lifetime. However, based on the link between sensor node lifetime and system lifetime, WSN lifetime is addressed indirectly. Furthermore, we will limit the focus of the sensor node lifetime to the availability of energy.

### 2.2 Wireless Sensor Node Architecture

The basic architecture of wireless sensor nodes has not changed significantly since the initial proposal of WSNs. It usually contains modules for computation, communication, sensing and power management. Application-specific
tasks can require some additional resources, but in the majority of cases these functionalities can be classified as belonging to one of the basic modules, mentioned previously. An abstract overview of the hardware architecture of general sensor nodes is provided in Figure 2.1.

The computation module of a sensor node usually has several tasks to fulfill. It controls the other components on the platform, processes and stores data, and provides an interface to the user/programmer. While the implementation of the computational module will truly depend on the specific application, the majority of nodes implement the computation module based on a low-power microcontroller. Microcontrollers integrate a variety of different hardware modules, which are useful for the accomplishment of the sensor node tasks. Furthermore, they are easily programmed and have low power demands. Popular choices in existing sensor nodes include Atmel’s ATmega series [23, 24], Texas Instruments MSP430 [25–27], as well as PIC controllers from Microchip [28]. For processing intensive applications, however, Field Programmable Gate Arrays (FPGAs) and Digital Signal Processors (DSPs) might be used as co-processors [29].

Because an active operation consumes a considerable amount of power (i.e., normally of the order of hundreds of μA MHz⁻¹), most microcontrollers

![Figure 2.1: Abstracted architecture of typical sensor nodes, used in Wireless Sensor Networks. These nodes typically combine processing, communication and sensing capabilities, and are powered by an energy source (e.g., a battery)](image-url)
offer a series of operating modes, allowing the system to conserve energy when activity is not required. Nonetheless, to allow this operation, the microcontroller must facilitate the means to be woken up again in order to continue the normal operation when necessary.

In a similar manner to the computational module, the communication module implementation also depends to some degree on the application. Nevertheless, it is also the case that the majority of systems use similar communication devices, namely low-power RF transceiver, typically operating in the license-free ISM-band. Other communication methods, such as acoustical or optical techniques are only used in special cases (e.g., for underwater sensor networks). While RF has been a popular communication method since the early stages of WSN research, over the last few years transceivers that are compatible with the IEEE 802.15.4 protocol\(^1\) have become the de-facto communication standard. This is partly due to its world-wide available frequency band, as well as the increased interest in commercialization, which is positively affected by applying a standardized communication protocol.

The main task for the communication module is the establishment of a link between individual nodes. This link, in turn, is used in order to exchange data between sensor nodes, and, particularly in environmental monitoring applications, in order to propagate data from the individual sensor nodes to a central data collection unit (i.e., the network sink). In addition to this local communication link, a global communication module might be implemented. This module has the purpose of connecting the local sensor network to the outside world and, thus, provides a universal access method to the monitoring system. Typical implementations of this global communication module are WiFi [14], long-range radio communication [8, 11] and GSM/GPRS [5, 20]. However, the number of nodes that are equipped with such a module is strictly limited, as both the power consumption and the price for these devices are excessive.

Although the majority of sensor nodes within the system only contain local communication, communicating remains costly for these resource limited devices. Even when operating in idle mode, and only listening to surrounding noise, energy consumption of the communication module is tremendous, easily reaching similar levels to those involved in transmitting data. Therefore,

\(^1\)http://www.ieee802.org/15/pub/TG4.html
sensor nodes should disable their communication modules whenever possible, reducing power consumption by several orders of magnitude. However, waking up in time to receive a packet destined for this node can become a considerable problem.

While the implementations of computation and communication modules vary depending on the resource requirements of the application, it is definitely the case that the most application-specific part of a sensor node is its sensing module. A given application will be required to monitor certain physical parameters or detect specific events. This, in turn, requires specific sensors that have the ability to fulfill these application demands.

Although any type of sensor is imaginable for WSN operation, the choice of sensors found in the literature is rather limited. The majority of work is limited to low-power sensors, with a typical example being temperature sensing. The possible reasons for this are, limited application-oriented research, difficulties in performance comparison when there are different underlying assumptions, as well as the simplicity and availability of these sensing devices. Nonetheless, this easily leads to assumptions within the community that do not generally hold true. One typical example being the negligible energy consumption of the sensing module. While this might hold true for the above mentioned temperature sensors, there are many sensors that have a higher power consumption than the communication module or require tremendous warm-up times in order to provide accurate readings. The presentation of the SAQnet deployment in Section 1.3 provides an example in this regard.

While it is typically not considered to be a main function of a sensor node, underlying all other node modules is the power supply. Its main task is as simple as it is important, namely in providing a stable power supply to all active components of the sensor node system. This means that it converts the input from the energy source to acceptable levels in order to power the connected devices.

The manner in which this conversion actually appears will generally depend on the type of energy source used for the sensor nodes. In some cases the sensor node might be able to extract power from a power grid, requiring a module for AC-DC conversion. However, especially in environmental monitoring WSN applications, this is seldom the case. The use of battery supplies is more popular, and, recently, the use of harvesting energy from ambient energy sources, such as wind, temperature difference or the sun has gained
increased interest [30, 31]. For these sources, the power supply is based on DC-DC converters.

Additionally, the power module might include monitoring and control functions. The type of monitoring is mainly dependent on the energy source in use. For example, in the case of a battery-based source, a typically desired monitoring function to be implemented is the determination of the battery’s State-of-Charge (SOC) in order to predict, for example, its remaining lifetime. On the other hand, for energy harvesting systems, the observation of the incoming energy level might be more important, due to its intermittent behavior. Control functions are implemented, for example, to react to monitored events or periodic tasks. A typical example is the shutdown of certain modules on the node in order to conserve energy, either because they are unnecessary or because reduced energy income was detected. In case these modules do not have own low-power modes, or their quiescent current is too large, the power supply module can contain features in order to disable the modules.

2.3 Duty-Cycling

As mentioned previously, most active components (e.g., microcontroller and RF transceiver) provide multiple power states, with tremendous differences in their energy consumption. In reality, the choice of operation states can be quite broad. For example, a typical MSP430 microcontroller\(^2\) from Texas Instruments offers seven different states (i.e., one active mode, five low power modes, as well as shutdown), thus allowing the optimization of energy conservation by picking the most appropriate operation mode at any given time. However, for simplification reasons, in the following, a division will only be made between *active state* and *inactive state*. This will provide the basic underlying principles, that can be extended by means of further operation state layering when required.

Duty-cycling is a common approach for the reduction of the average energy consumption of wireless sensor nodes. Its main principle is to achieve lower energy consumption by reducing the power levels whenever possible (see Figure 2.2 for a graphical representation). Defining the portion of time in

\(^2\)http://www.ti.com/product/msp430fg4618
2.3 Duty-Cycling

![Graphical representation of the duty-cycling principle](image)

Figure 2.2: Graphical representation of the duty-cycling principle (simplified for bi-modal operation states)

The active state as $T_{active}$ and the period in the inactive state as $T_{inactive}$, the duty cycle $\delta$ can be formally described as

$$\delta = \frac{T_{active}}{T_{active} + T_{inactive}}, \quad 0 \leq \delta \leq 1. \quad (2.1)$$

In order for the method to be effective, the duty cycle $\delta$ should be considerably smaller than one. Furthermore, the respective active and inactive states have to differ in their power consumption levels. In typical WSN applications, both requirements are fulfilled. Particularly in environmental monitoring, sampling rates are rather low (i.e., often in the order of minutes [3, 6, 32]). This leads to extremely low duty cycles in the network, which are often much smaller than one percent. The consumption level differences are accomplished by toggling the individual sensor node modules between active and inactive states. An overview of the current consumption levels usually achieved in different module states is given in Table 2.1. As a result, power requirement reductions of multiple orders of magnitude are not uncommon when disabling modules that do not perform a task.

Taking these factors into account, duty-cycling can be applied to the different consumers of the sensor node. In general, the average power consumption of a duty-cycled sensor node can be described as
2 Energy Efficiency

\[ P_\delta = \frac{P_{active} \cdot T_{active} + P_{inactive} \cdot T_{inactive}}{T} \]  
(2.2)

\[ = \delta \cdot P_{active} + (1 - \delta) \cdot P_{inactive}. \]  
(2.3)

Herein, the interval time \( T \) is the sum of \( T_{active} \) and \( T_{inactive} \) and \( \delta \) as described in Equation 2.1. However, for each sensor node module individual challenges have to be addressed. The simplest of these is usually the computational unit, as it is the the central control unit in the system. The computational unit, thus, can switch to a low-power mode whenever idle (i.e., waiting for the next event to occur). The only limitation being, that it is possible to awaken the module from its low-power state as soon as an event has to be processed, which is usually implemented by utilizing interrupts. The resulting power consumption can be estimated by substituting computational unit values into equations 2.2 and 2.3

<table>
<thead>
<tr>
<th>Module</th>
<th>State</th>
<th>Operation</th>
<th>Classifier</th>
<th>Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication</td>
<td>RX (RF)</td>
<td>Active</td>
<td>Tens of mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TX (RF)</td>
<td>Active</td>
<td>Tens of mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sleep</td>
<td>Inactive</td>
<td>μA</td>
<td></td>
</tr>
<tr>
<td>Computation</td>
<td>Processing</td>
<td>Active</td>
<td>Hundreds of μA MHz⁻¹</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Memory access</td>
<td>Active</td>
<td>mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sleep</td>
<td>Inactive</td>
<td>μA</td>
<td></td>
</tr>
<tr>
<td>Sensing</td>
<td>Sampling</td>
<td>Active</td>
<td>μA – hundreds of mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Warm-up</td>
<td>Active</td>
<td>μA – hundreds of mA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sleep*</td>
<td>Inactive</td>
<td>μA</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Operation states and consumption levels of typical sensor node modules for environmental monitoring Wireless Sensor Networks. (*Many sensors do not integrate low-power modes and, thus, have to be disconnected from their power supply.)
2.3 Duty-Cycling

\[
P_{\text{comp}, \delta} = \frac{P_{\text{comp,active}} \cdot T_{\text{comp,active}} + P_{\text{comp,inactive}} \cdot T_{\text{comp,inactive}}}{T_{\text{comp}}} \]  \hspace{1cm} (2.4)
\]

\[
= \delta_{\text{comp}} \cdot P_{\text{comp,active}} + (1 - \delta_{\text{comp}}) \cdot P_{\text{comp,inactive}}. \hspace{1cm} (2.5)
\]

However, \(T_{\text{comp,active}}, T_{\text{comp,inactive}},\) and as a result of this \(\delta_{\text{comp}},\) typically remain unknown to a great extent. As the active and inactive periods are usually unscheduled and the computation occurs only when required, it is difficult to predict the exact active and inactive time intervals. Nevertheless, quite accurate estimations can be made, when events that trigger computation are known and occur periodically.

Similarly, it is common for the sensing module’s activities to be scheduled by the computational unit. This is particularly true in periodic sampling applications, where the ratio of active to inactive time is, to a great extent, predefined. However, many sensors do not have low-power states, which means that they have to be disabled externally at times when they are not required to be operational. This, in turn, leads to longer power-up intervals, as sensors might require some time to warm-up, before producing accurate measurement results. Integrating this into the equations leads to an average sensing power consumption of

\[
P_{\text{sens}, \delta} = \frac{P_{\text{sens,on}} \cdot (T_{\text{sens,warm}} + T_{\text{sens,samp}}) + P_{\text{sens,off}} \cdot T_{\text{sens,off}}}{T_{\text{sens}}} \]  \hspace{1cm} (2.6)
\]

\[
= \delta_{\text{sens}} \cdot P_{\text{sens,on}} + (1 - \delta_{\text{sens}}) \cdot P_{\text{sens,off}}, \hspace{1cm} (2.7)
\]

with \(T_{\text{sens,warm}}\) being the warm-up time and \(T_{\text{sens,samp}}\) the sampling time, which can be considerably different for individual sensor implementations. The duty cycle \(\delta_{\text{sens}},\) for this module, results in

\[
\delta_{\text{sens}} = \frac{T_{\text{sens,warm}} + T_{\text{sens,samp}}}{T_{\text{sens,warm}} + T_{\text{sens,samp}} + T_{\text{sens,off}}}, \hspace{1cm} 0 < \delta_{\text{sens}} < 1. \hspace{1cm} (2.8)
\]

A large variation can be observed with respect to the warm-up time of sensors. While many sensors can be considered operational instantly, gas sensors (e.g.,
2 Energy Efficiency

those implemented in the SAQnet nodes), for example, can easily reach tens of seconds. In these cases, the sensor warm-up time has a large impact on the efficiency of the duty-cycling. Moreover, the consumption levels in the active state can be immensely different from sensor to sensor. Based on the minimal difference in consumption during active and inactive module states, the duty-cycling of low power sensors might not be effective.

On the contrary, typically used communication transceivers in the RF band show a considerable consumption difference between their active and inactive states. The duty-cycling of these modules, thus, becomes essential for energy conservation in the majority of systems. However, transmission and reception of data can only occur when the transceiver is in the respective states. While this is not a considerable problem for the transmission of data, data reception can be difficult to predict. At the same time, however, one does want to predict this event as accurately as possible in order to reduce the active period and, thus, minimize the duty cycle. A mechanism that addresses this issue, particularly suited for periodic data gathering applications, is presented in Section 2.5.

Completing the description set, for the communication module we can approximate duty-cycled power consumption as

\[
P_{\text{com, } \delta} = \frac{P_{\text{com, active}} \cdot (T_{\text{com, RX}} + T_{\text{com, TX}}) + P_{\text{com, inactive}} \cdot T_{\text{com, sleep}}}{T_{\text{com}}} = \delta_{\text{com}} \cdot P_{\text{com, active}} + (1 - \delta_{\text{com}}) \cdot P_{\text{com, inactive}},
\]

with

\[
\delta_{\text{com}} = \frac{T_{\text{com, RX}} + T_{\text{com, TX}}}{T_{\text{com, RX}} + T_{\text{com, TX}} + T_{\text{com, sleep}}}, \quad 0 < \delta_{\text{com}} < 1.
\]

In this case, \( T_{\text{com, RX}} \) is the reception time, \( T_{\text{com, TX}} \) the transmission time and \( T_{\text{com, sleep}} \) the period in sleep state. Furthermore, transmission and reception power consumption are simplified to be the same, defined as \( P_{\text{com, active}} \).

The effectiveness of duty-cycling is illustrated in Figure 2.3. It shows the influence of a changing duty cycle for the computational module and the communication module based on parameters obtained from two different platforms, respectively. In order to obtain the illustrated results, platform
2.4 Energy-efficient Sensor Node Design

Figure 2.3: Impact of the duty cycle of different modules on the overall current consumption of a sensor node.

specific consumption values for active and inactive states have been inserted into the equations that have previously been presented. An overview of these parameters is given in Table 2.2.

The graphs in the figure visualize the significance of two parameters on the duty cycling influence. Firstly, the optimal result (i.e., \( \lim_{\delta \to 0} \)) is defined by the consumption during inactivity. Secondly, a greater difference between inactive and active consumption results in a faster reduction of average consumption with decreasing duty cycle. As a result, the reduction of the communication duty cycle is typically producing a larger impact than that of the computation module. The sensing module has been ignored in this evaluation, because of its highly application dependent consumption and the small user influence on the sensor’s active period.

2.4 Energy-efficient Sensor Node Design

A fundamental influence on the power consumption, and thus the lifetime of a sensor node, is the hardware platform it is based on. In order to achieve the desired low duty cycles, presented in the previous section, the underlying node platform must operate in an energy efficient manner. This means, that
2 Energy Efficiency

<table>
<thead>
<tr>
<th>Platform</th>
<th>Module</th>
<th>State</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>SENTIO-em</td>
<td>Computation</td>
<td>Active</td>
<td>8 mA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inactive</td>
<td>1.8 μA</td>
</tr>
<tr>
<td></td>
<td>Radio</td>
<td>Active</td>
<td>25 mA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inactive</td>
<td>1 μA</td>
</tr>
<tr>
<td>SAQnet</td>
<td>Computation</td>
<td>Active</td>
<td>9 mA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inactive</td>
<td>62 μA</td>
</tr>
<tr>
<td></td>
<td>Radio</td>
<td>Active</td>
<td>63 mA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inactive</td>
<td>10 μA</td>
</tr>
</tbody>
</table>

Table 2.2: Platform consumption parameters of SENTIO-em and SAQnet nodes. These parameters have been utilized for the computations that have led to Figure 2.3.

for the accomplishment of a preordained task, as little energy as possible should be spent. For low duty cycle applications, such as those typically found in the environmental monitoring domain, this is achieved by (i) reducing the power consumption in inactive states, and by (ii) reducing the duty cycle. Considering a fixed sample rate, which has been defined by the application, in order to reduce the duty cycle, the time periods of active operation have to be shortened.

Energy efficiency has been one of the key features in the design of the SENTIO-em hardware platform. Its design and implementation has been performed, particularly, with the constraints of environmental monitoring applications in mind. The resulting node platform, thus, is an application-domain-specific design, which is optimized for the usage in environmental monitoring applications, while providing sufficient flexibility for its utilization in different applications within the domain.

2.4.1 SENTIO-em Architecture and Implementation

The fundamental architecture of SENTIO-em is similar to most existing sensor node platforms. As described in Section 2.2, most wireless sensor nodes combine three capabilities, namely, processing, communication and sensing. These three functionalities are reflected in hardware designs, which
result in a modular implementation that contains the respective resources. In addition, a power module is required in order to supply active components with the necessary energy.

While SENTIO-em includes several architectural differences from existing platforms, these differences are not directly related to the energy efficiency of the sensor node. Thus, a closer description of these architectural choices will be provided at a later stage in this document. A general overview of the platform’s architecture is depicted in Figure 2.4. In this figure, a differentiation is made between those modules that are situated on the core platform, and those, which are connected via defined interfaces. The external placement of certain modules increases the flexibility of the platform and allows for the component overhead to be kept to a minimum. In addition to other reasons, this affects the final quiescent current of the platform, which defines its power consumption during inactive periods and, thus, the overall energy consumption in low duty cycle applications.

Nevertheless, the energy efficiency of the platform is mainly affected by its module implementation (i.e., the component selection), rather than its
2 Energy Efficiency

<table>
<thead>
<tr>
<th></th>
<th>ATMega1281</th>
<th>MSP430F1611</th>
<th>AT32UC3L</th>
<th>EFM32G280</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>Atmel</td>
<td>Texas Instruments</td>
<td>Atmel</td>
<td>Energy Micro</td>
</tr>
<tr>
<td>Architecture</td>
<td>8-bit</td>
<td>16-bit</td>
<td>32-bit</td>
<td>32-bit</td>
</tr>
<tr>
<td>Clock Frequency [MHz]</td>
<td>0–16</td>
<td>0–8</td>
<td>0–50</td>
<td>0–32</td>
</tr>
<tr>
<td>Flash [kB]</td>
<td>64–256</td>
<td>48</td>
<td>16–64</td>
<td>32–128</td>
</tr>
<tr>
<td>SRAM [kB]</td>
<td>8</td>
<td>10</td>
<td>8/16</td>
<td>8/16</td>
</tr>
<tr>
<td>Operating Voltage [volt]</td>
<td>2.7–5.5 (&lt;8 MHz)</td>
<td>1.8–3.6 (&lt;4 MHz)</td>
<td>1.62–3.6</td>
<td>1.8–3.8</td>
</tr>
<tr>
<td>Current Draw</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Active [mA MHz⁻¹]</td>
<td>1</td>
<td>0.5</td>
<td>0.2</td>
<td>0.18</td>
</tr>
<tr>
<td>Sleep [μA]</td>
<td>5–100</td>
<td>1.1–75</td>
<td>5–45</td>
<td>0.6–0.9</td>
</tr>
<tr>
<td>Off [nA]</td>
<td>250</td>
<td>100</td>
<td>9</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 2.3: Comparison of typical microcontroller choices for the implementation of hardware platforms in sensor networks

underlying architecture. Due to the application-dependency of external modules, the presentation of implementation choices will be limited to the core module, which means the computation, communication and power regulation module.

A major selection to be made involves the implementation of the computational unit, as this module performs the central control of the entire sensor node. As the desire is for long operation periods and, additionally, because of the rather low demands on the computational performance, most existing sensor node platforms have been designed with low-power microcontrollers. Typical examples of these microcontrollers include the Atmel AVR series and the MSP430 family from Texas Instruments, which run at clock speeds of a few megahertz to tens of megahertz. As a result of its energy efficiency, ARM processors have become the standard in other mobile computing areas, such as mobile phones. While many ARM processors are targeting applications with much higher processing demands as compared to Wireless Sensor Networks, low-power processors also exist, such as the ARM Cortex series. Several microcontroller manufacturers have started to incorporate these cores into their controller families.

Table 2.3 shows a comparison of different low-power microcontrollers. While the ATMega1281 and the MSP430F1611 are controller types that have been used extensively in existing sensor node platforms, the AT32UC3L and the EFM32G280 series represent more powerful processors, which maintain their
2.4 Energy-efficient Sensor Node Design

focus on low-power consumption. As can be extracted from the property listing in the table, both of the more powerful processors outperform the previously used controllers in most regards. The EFM32 controller, in particular, demonstrates the energy efficiency of its Cortex M3 core in both active and inactive states. As a result, this microcontroller allows for the desired tasks to be completed in a shorter period, leaving the controller running at lower power levels for a larger period of time.

Therefore, the EFM32G280 has been chosen as the core of the computation module on the SENTIO-em platform. It is driven by a 32 MHz Q-MEMS crystal, which enables the platform to process data with a considerable speed improvement as compared to traditional Micro Controller Unit (MCU) solutions, while maintaining similar power levels during processing due to its improved energy efficiency. In addition, the computational unit of SENTIO-em includes a Real-Time Clock (RTC) for accurate time-keeping, as well as a microSD card slot for the storage of large amounts of sensor data. In both cases, the energy efficiency of the final platform has driven the selection process of the implementation. As SD cards have a considerable quiescent current, the SD card interface contains an analog switch in its supply line, which allows for the interface to be completely cut from power, leaving only the leakage of the switch as the inactive consumption of this module. Such a creation of individual power domains, which can be controlled by the microcontroller, is used more extensively with external platform modules and, thus, allows for the power consumption during inactive states to be kept at a minimum.

The implementation choice of the communication module, on the other hand, is rather more difficult. While the decision for the usage of an RF transceiver is rather straightforward, different transceivers provide their individual properties, which makes it difficult to determine a clearly advantageous transceiver option. Table 2.4 contains several alternatives that have been considered for the implementation on SENTIO-em. The de-facto standard in existing sensor nodes are transceivers with an integrated protocol in compliance with the IEEE 802.15.4 standard. Typical choices include the Texas Instruments CC2420 (and its successor, the CC2520), as well as Atmel’s RF23x series. Although these transceivers allow for a large set of existing communication protocols to be used, at times when these solutions are not required, the implemented protocol creates an undesired overhead. Further-
2 Energy Efficiency

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>CC1101</th>
<th>SX1233</th>
<th>CC2520</th>
<th>XBEE 802.15.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency [MHz]</td>
<td>Texas Instruments</td>
<td>Semtech</td>
<td>Texas Instruments</td>
<td>Digi International</td>
</tr>
<tr>
<td>Max. Data Rate [kbps]</td>
<td>315/433/868/915</td>
<td>433/868/915</td>
<td>2400</td>
<td>2400</td>
</tr>
<tr>
<td>Sensitivity [dBm]</td>
<td>-116</td>
<td>-120</td>
<td>-98</td>
<td>-100</td>
</tr>
<tr>
<td>Output Power [dBm]</td>
<td>-30 to +12</td>
<td>-18 to +17</td>
<td>-18 to +5</td>
<td>-10 to +18</td>
</tr>
<tr>
<td>TX [mA]</td>
<td>12–34</td>
<td>16–95</td>
<td>25–37</td>
<td>45–250</td>
</tr>
<tr>
<td>Sleep [μA]</td>
<td>0,2</td>
<td>0,1</td>
<td>&lt;1</td>
<td>&lt;10</td>
</tr>
<tr>
<td>Implemented Protocols</td>
<td>none</td>
<td>none</td>
<td>low-level</td>
<td>low / high-level</td>
</tr>
</tbody>
</table>

Table 2.4: Comparison of considered radio transceiver choices for the implementation of SENTIO-em

more, the majority of these Integrated Circuits (ICs) operate in the 2.4 GHz band, which limits their communication range considerably as compared to those transceivers, which operate at lower frequencies [33].

As a result of their individual advantages and the application-dependency of the desired transceiver properties, a flexible solution for the implementation of the radio transceiver has been chosen. Physically, this implementation is oriented in accordance with the XBee communication module\(^3\), which has gained tremendous attention, particularly, in the field of hobby electronics. Following the XBee’s footprint enables us to reuse existing transceiver module solutions, which range from simple IEEE 802.15.4 modules to high-power 802.11 radios. At the same time, it becomes possible for personal implementations of transceivers to be conducted in an easy manner. This allows for a transceiver with the best performance for a given application to be chosen, while limiting the overhead to a minimum.

Finally, while the interface for an external power supply circuit is provided, the core platform of SENTIO-em contains basic power regulation in order to guarantee the required levels for the components on board. This regulation has been implemented based on a low-power Low Dropout (LDO) regulator, which only consumes quiescent currents of the order of 500 nA. Although the usage of a Switch Mode Power Supply (SMPS) leads to less losses as compared to LDOs, these converters typically have higher quiescent currents, which

\(^3\)http://www.digi.com/xbee/
results in an overall reduction in operational efficiency in low duty cycle applications. As has been mentioned previously, additional power domains are created for modules that have a considerable power consumption, even during inactivity. Utilizing analog switches or LDO regulators allows for these modules to be disabled with a minimal overhead in the system. Based on the module that is switched off, however, increased start up times might be the consequence, which has to be considered in each case.

2.4.2 SENTIO-em Evaluation

This section describes the evaluation of the SENTIO-em platform with respect to its energy efficiency and both methods for evaluating the energy efficiency and the obtained results will be presented.

In order to evaluate the energy efficiency of the platform, both its power consumption and computational performance have been measured. Furthermore, as the platform is intended for usage in applications within the environmental monitoring domain and, thus, will be deployed under varying conditions, measurements have been conducted over a wide temperature range. Finally, the obtained measurement results are compared to the results obtained from existing sensor node implementations.

As SENTIO-em operates with a constant voltage, its power consumption can be measured by measuring the current consumption. For the measurement of platform currents, an Agilent 34410A\footnote{http://www.agilent.com} digital multimeter has been used. Furthermore, all results are based on a platform equipped with a Semtech SX1233 RF transceiver module. In order to create different environmental conditions, a measurement series has been conducted in an environmental test chamber\footnote{http://www.testequity.com/products/1104/}. While the platform will most likely not be exposed to all of these conditions, for the sake of completeness, the measurements have been conducted over the entire temperature range specified for the platform. Finally, the transition times between different operating states and timings for the evaluation of the processing performance have been measured with a digiview logic analyzer\footnote{http://www.tech-tools.com/logic-analyzer.htm}.
### 2 Energy Efficiency

<table>
<thead>
<tr>
<th>State</th>
<th>Current</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>6.1 mA</td>
<td>14 MHz</td>
</tr>
<tr>
<td></td>
<td>8 mA</td>
<td>32 MHz</td>
</tr>
<tr>
<td>Low-Power</td>
<td>2.3 μA</td>
<td>EM1</td>
</tr>
<tr>
<td></td>
<td>1.6 μA</td>
<td>EM2</td>
</tr>
<tr>
<td></td>
<td>1.2 μA</td>
<td>EM3</td>
</tr>
<tr>
<td></td>
<td>1.8 μA</td>
<td>EM3 + RTC</td>
</tr>
<tr>
<td>Radio</td>
<td>11–50 mA</td>
<td>TX</td>
</tr>
<tr>
<td></td>
<td>16.5 mA</td>
<td>RX</td>
</tr>
</tbody>
</table>

Table 2.5: Current consumption of SENTIO-em in its different operation states. The measurements have been conducted under laboratory conditions.

The results of the current consumption measurements, which were performed under laboratory conditions are listed in Table 2.5. These values represent the complete platform consumption. However, during Active and Low-Power state, the radio module has been set to sleep mode, whereas the microcontroller has been in a low-power state (i.e., EM3) for the Radio measurements. The obtained values comply with the expectations based on datasheet values. In the low-power mode EM3, for example, a platform current consumption of 1.2 μA has been measured, which matches the expected values ideally (i.e., 600 nA MCU, 500 nA LDO, 100 nA radio). Figure 2.5 shows the platform’s sleep mode consumption over its entire temperature range. While the current consumption remains low for all temperature values tested, a considerable increase in consumption can be detected at temperatures above 40 °C. As opposed to the exponential leakage increase in low-power modes, the active consumption increases linearly with temperature. An example is shown in Figure 2.6 for the transmission and reception current in the radio transceiver. In comparison with the consumption increase during inactivity, which in the worst condition can add up to 300% of its initial consumption, the power consumption in active states only shows a small percentage increase.

In addition to the effect of low quiescent currents on the energy efficiency of a sensor node, the active periods must be short. Thus, a sensor node should
2.4 Energy-efficient Sensor Node Design

Figure 2.5: Temperature influence on the leakage currents of the SENTIO-em platform. Measurements have been conducted over the entire specified temperature range for SENTIO-em. Humidity levels have been constant at 40%.

Figure 2.6: Temperature influence on the current consumption of SENTIO-em during communication tasks. Humidity levels have been constant at 40%.

be able to switch quickly between its different operation states, and perform any given task in a rapid manner. Table 2.6 lists the transition times that have been measured for the SENTIO-em implementation. The obtained values show that SENTIO-em is capable of fast transitions, taking less than 6 μs for the processor to be operational and a few hundreds of μs until it becomes ready to communicate\(^7\).

Figure 2.7 depicts the processing performance of the EFM32 controller on the SENTIO-em platform in comparison to that for an Atmel ATMega1281, \(^7\)Using the Semtech SX1233 radio transceiver module
2 Energy Efficiency

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Transition time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep</td>
<td>Active</td>
<td>1.8 – 5.8 μs</td>
</tr>
<tr>
<td>Radio sleep</td>
<td>TX</td>
<td>150 μs</td>
</tr>
<tr>
<td>Radio sleep</td>
<td>RX</td>
<td>380 μs</td>
</tr>
<tr>
<td>TX</td>
<td>RX</td>
<td>390 μs</td>
</tr>
<tr>
<td>RX</td>
<td>TX</td>
<td>70 μs</td>
</tr>
</tbody>
</table>

Table 2.6: Transition times between different operation states of the SENTIO-em platform. Transition times have been measured on a platform equipped with a Semtech SX1233 transceiver.

Figure 2.7: Evaluation of the processing performance of the SENTIO-em platform in comparison with a typical microcontroller choice. Processing performance is measured by the processing time of a set of benchmarking functions.

which has previously been a highly popular choice for the implementation of sensor nodes. The computations performed are based on a benchmarking suite for microcontrollers\(^8\). This set of functions ranges from simple math operations, via matrix manipulations, to more complex operations, such as the Whetstone benchmark. The figure contains the time periods taken by the respective platform to perform the given task. While SENTIO-em has been operating at a clock frequency of 32 MHz, and thus four times faster than that of the ATMega1281 under test, its processing results show a time improvement of 6 to 46 times.

\(^8\)http://www.ti.com/lit/an/slaa205c/slaa205c.pdf
Finally, as energy efficiency combines both the processing performance and the power consumption, Figure 2.8 attempts to quantify the energy efficiency of a set of node platforms based on its clock frequency and quiescent power consumption. While the previously presented benchmarking results indicate that processing performance is not purely related to the clock frequency, the maximum available clock frequency is used as an estimator, because of the difficulty in obtaining the exact performance measurements for each individual platform. The result demonstrates the high energy efficiency of SENTIO-em with reference to typically utilized sensor node platforms.

2.5 Energy-efficient Communication

In addition to the sensor node hardware, which determines the physical power consumption in each operation state and the period of time a processing task requires, the manner in which communication is performed has a tremendous impact on the energy efficiency of the sensor node. As has been presented previously, in order to increase energy efficiency, the amount of time that is spent in the active operating states has to be reduced. Because communication represents an active state with a considerable contribution to the power consumption of the platform, limiting the time period that is required for a certain amount of data to be transferred is a desirable goal.
A typical classification of communication methods in wireless communications is the distinction between contention-based and contention-free protocols. Herein, contention-free protocols allow for certain communication resources, such as time periods (Time Division Multiple Access (TDMA)) or frequency channels (Frequency Division Multiple Access (FDMA)), to be allocated to individual nodes, whereas contention-based protocols create a competitive environment for sensor nodes to perform their communication task. Based on its increased flexibility with respect to communication times, contention-based protocols, such as Carrier Sense Multiple Access (CSMA), have become the typical choice in wireless communication standards, such as in IEEE 802.15.4 and IEEE 802.11. A common reason for this is that the unscheduled manner of communication in these protocols allows for nodes to compete for the channel at any point in time. As a result, the communication latencies are typically reduced and networks operating with a contention-based protocol support a higher degree of system dynamics, such as nodes joining and leaving the network. Contention-free protocols, on the other hand, provide a high reliability within static communication networks and operate in an energy efficient manner.

The energy efficiency of a communication protocol is defined by its capability to handle certain wasteful activities, such as those listed in Table 2.7. As most of these parameters originate from unknown communication times, they cannot be fully eliminated in contention-based communication protocols. Furthermore, functionalities provided in order to reduce such activities typically lead to an increase in protocol overhead. An example for this is the implementation of Request-to-Send (RTS) and Clear-to-Send (CTS) control packets for the reduction of packet collisions, which, in turn, increase the amount of data to be transmitted. Time scheduled protocols (i.e., TDMA), on the other hand, can significantly reduce (if not eliminate) the majority of the named activities by the mere knowledge of the point in time during which the communication will be conducted.

As the sensor nodes in the targeted application scenarios typically perform a periodically occurring sampling task, TDMA-like communication protocols become an ideal candidate for the energy efficient transport of measured data from the sensor node to the data collection point in the network. The typical structure of an TDMA protocol is depicted in Figure 2.9. In order to minimize the effects of the parameters listed in Table 2.7, however, any uncertainties
2.5 Energy-efficient Communication

Figure 2.9: Overview on the typical structure of a TDMA communication protocol. Communication occurs within individual time slots.

regarding time must be retained at a minimum. Because of the resource limitations on typical sensor nodes, which prohibits the implementation of timing circuits of high accuracy, there is a requirement for time synchronization between individual nodes to be performed. In the following sections, the difficulties of time synchronization in Wireless Sensor Networks will be presented, and an energy efficient method for the implementation of time synchronization for the purpose of communication will be presented.

2.5.1 Time Synchronization for Wireless Sensor Networks

There are several tasks in Wireless Sensor Networks, which demand or profit from a common notion of time between nodes. These include the timestamping of samples, collaborative sampling of an event, and the measurement of time related properties (e.g., velocity). In order to achieve this common notion of time, synchronization is mandatory. While synchronization requires communication and, thus, demands an increase in the spent energy, synchronized communication can minimize unnecessary active periods, which can lead to an overall more energy efficient communication method.

The need for time synchronization between sensor nodes is based on non-ideal clock sources, implemented on these devices. Ideally, the time retained by a sensor node $C(t)$, should represent the reference time $t$, thus

$$C(t) = t \, .$$

(2.12)
Collision

A collision is a superposition of two or more data transfers at the same time. At the receiver this leads to a non-decomposable reception and therefore to data loss. In non-scheduled communication collision is avoided by \textit{listen-before-send} mechanisms or additional control sequences, such as RTS and CTS packets. In scheduled protocols, this problem is solved by allocating unique time periods for communication to nodes respectively.

Overhead

Overhead is additional information that has to be sent with the data, e.g. to determine source and destination. Overhead increases in on-demand protocols when control sequencing is used. Depending on synchronization techniques, scheduled communication might also have increased overhead. In scheduled communication, however, source identification might be omitted by using time allocation instead.

Over-Hearing

Over-Hearing describes the problem of one node listening to communication, or part of the communication, dedicated to another node. While this problem is still applicable to on-demand protocols, in scheduled communication each node is aware of when data is meant to be targeted to itself and when not.

Over-Emitting

Over-Emitting is the effect of one node transmitting data to another node, while the receiver is not listening. The result is similar to collisions, and once again it is solved by defining communication times in scheduled protocols.

Idle-Listening

Idle-Listening occurs when the receiver is waiting for a transmission, listening to the channel without traffic occurring. Consumption in this state is similar to active reception. On-demand protocols usually target this problem by listening periodically to the channel. Thus reducing, but not eliminating the impact. Scheduled communication ideally knows exactly when the transmission will start. However, in reality they are unable to eliminate idle-listening completely, because of the inaccuracy of the synchronization.

Table 2.7: Parasitic communication factors, that contribute to the energy consumption of WSN nodes
2.5 Energy-efficient Communication

However, due to the non-ideal properties of the clock source, a relation of node $i$’s clock to the reference time of

$$C_i(t) = \omega \cdot t + \phi$$  \hspace{1cm} (2.13)

becomes a more appropriate description model. In this case, $\omega$ describes the clock skew or frequency offset of node $i$’s clock source and $\phi$ is a phase offset. Using one node as the reference, we can further describe the node clock relations as

$$C_j(t) = \omega_{ij} \cdot C_i(t) + \phi_{ij}, \quad i, j = 1, 2, ..., N,$$  \hspace{1cm} (2.14)

where $\omega_{ij}$ and $\phi_{ij}$ are the clock skew and phase offset between nodes $i$ and $j$ respectively and $N$ is the number of nodes in the network. Time synchronization between nodes is reached when

$$C_j(t) = C_i(t), \quad i, j = 1, 2, ..., N.$$  \hspace{1cm} (2.15)

It should be noted that this does not necessarily mean synchronization to real time, but synchronization of one node’s clock to the clock of its reference node. Comparing equations 2.14 and 2.15, depicting the initial and the desired case respectively, it is obvious that the goal of synchronization is to compensate for clock skew and phase offset (i.e., $\phi_{ij} = 0$ and $\omega_{ij} = 1$).

While the phase offset $\phi$ typically results from the different start configuration or start-up times of nodes, a single compensation run can usually eliminate the impact. However, clock skew is not a static factor, but will vary over time. Typically this variation is divided into short-term and long-term stability [34]. Short-term stability is usually influenced by environmental factors, such as temperature, pressure or supply voltage, showing an immediate effect on the stability of the clock source. On the other hand, long-term effects will influence the clock skew rather slowly over time. A common example of the long-term effects is oscillator aging.

The time-dependency of clock skew has the consequence that a single compensation of the frequency offset is not sufficient. Instead, regular resynchronization is mandatory in order to maintain synchronized node clocks. The resynchronization interval, herein, depends on the degree of environmental changes, the stability of the used clock source and the necessary
synchronization accuracy to be achieved in the respective applications.

In general, ideal synchronization cannot be guaranteed, as the synchronization method itself will introduce a certain amount of inaccuracy. Typical ways of establishing synchronization in WSNs, sources of inaccuracy, design consideration and trade-off factors will be introduced in the following section.

**Design Considerations and Sources of Inaccuracy**

When implementing a synchronization method for the creation of a common notion of time within the network, there are several design factors that must be considered. The majority of these factors are related and therefore lead to trade-off situations. These trade-offs have to be optimized for each application or system case individually. Typical design considerations include synchronization accuracy, the introduced energy cost of the synchronization method, as well as its scalability and behavior in dynamic networks.

Synchronization accuracy is the most obvious design factor, as synchronization is implemented for a purpose, namely to create a common notion of time. However, a reduced synchronization accuracy might be acceptable when the remaining accuracy fulfills the requirements of the target application. Synchronization accuracy is related to allocated resources, thus also to energy cost. Additionally scalability might be limited when very high accuracies are required to be reached. A major origin of inaccuracies is in relation to delay components during wireless communication. An overview of these delay components is given in Table 2.8 and their occurrence during a packet transmission is presented in Figure 2.10.

In energy constraint systems, such as Wireless Sensor Networks, the energy spent for synchronization can be a matter to be given consideration. Depending on the desired accuracy of the implementation, considerable amounts of energy can be spent in the communication or processing stages. In particular, when synchronization is used to reduce energy consumption, such as its utilization for accurate TDMA schedules, limiting the energy overhead spent by the synchronization approach is of major importance. Usually energy cost considerations conflict with synchronization accuracy and might influence the reliability of the network in dynamic situations.

Scalability and dynamic network behavior are factors that should be con-
sidered in any sensor network, which is planned to be deployed. Scalability herein describes how the synchronization accuracy is influenced with the increasing size of the network. A typical problem is in relation to additional delays due to multi-hop synchronization. On the other hand, dynamic network behavior deals with situations, such as the packet loss during the (re-)synchronization process.

**Time Synchronization Methods and Classification**

In essence, time synchronization between sensor nodes is established by reporting one node's time to the other node(s). As introduced in Table 2.8, the

<table>
<thead>
<tr>
<th>Delay Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send Time</td>
<td>Time period of handling the send request and preparing the packet transmission. Influenced by system load and overhead. Can be reduced by limiting system load by predicting time of resynchronization.</td>
</tr>
<tr>
<td>Access Time</td>
<td>Time of accessing the transmission medium. Influenced by network load. Usually high in contention-based protocols, while low for scheduled communication.</td>
</tr>
<tr>
<td>Transmission Time</td>
<td>Time to actually transmit the packet to the medium. Depends on the packet size, but can be estimated rather accurately.</td>
</tr>
<tr>
<td>Propagation Time</td>
<td>Time the message needs to propagate from sender to receiver. Depends on the distance between the two nodes. Comparably small in typical RF-based and short-range WSNs.</td>
</tr>
<tr>
<td>Reception Time</td>
<td>Time it takes to receive the packet at the receiver’s side. Complementary to Transmission Time.</td>
</tr>
<tr>
<td>Receive Time</td>
<td>Time for processing the received packet until the application is informed. Complementary to Send Time.</td>
</tr>
</tbody>
</table>

Table 2.8: Delay factors in Wireless Sensor Network transmissions, influencing synchronization accuracy when not addressed
transmission of packets occurs with a mainly unpredictable delay, reporting a timestamp of the past at the receiver when used for synchronization. This principle is illustrated in Figure 2.11a and can be formally described by

\[ C_B(t_2) = C_A(t_1) + \phi_{AB} + \delta , \]  

(2.16)

wherein \( \phi_{AB} \) is the clock offset between node A and B, and \( \delta \) describes the sum of the delay components. Being interested in determining \( \phi_{AB} \), \( \delta \) acts as a parasitic effect on the synchronization. In order to reduce its influence, the delay components in the system can be minimized, or they can be estimated/measured for compensation purposes.

A typical way of dealing with the latter case, is by using bidirectional communication in order to implement synchronization. This enables the estimation of the delay components from the measurement of round-trip delays. This method can be graphically demonstrated as shown in Figure 2.11b. Formally, it is possible to add the backward transmission

\[ C_A(t_4) = C_B(t_3) + \phi_{BA} + \delta , \quad \phi_{BA} = -\phi_{AB} \]  

(2.17)

to the description, and thus be able to estimate both, \( \delta \) and \( \phi_{AB} \) as

\[ \delta = \frac{(C_B(t_2) - C_A(t_1)) + (C_A(t_4) - C_B(t_3))}{2} \]  

(2.18)
Figure 2.11: Underlying timeline of packet transmission for (a) unilateral synchronization method and (b) bidirectional synchronization method

and

\[ \phi_{AB} = \frac{(C_B(t_2) - C_A(t_1)) - (C_A(t_4) - C_B(t_3))}{2}. \]  

However, this introduces some estimation error, because delays are simplified to be of the same length for both communication directions and an additional delay between \( t_2 \) and \( t_3 \) is added to \( \delta \). Furthermore, we increase the number of transmissions and now the receiver has to start the synchronization process (or an even higher number of packets is required). This might lead to energy overhead problems in systems with large number of nodes.

As the intention of using a common notion of time in the network varies from system to system, so does the design of synchronization algorithms. Due to the large amount and variance of solutions, common classification factors have been introduced, such as in [34–36]. A list of these classification factors is presented in Table 2.9, also summarizing the meaning of each factor.

Many time synchronization algorithms have been presented for use in Wireless Sensor Networks. The most popular ones within the community...
2 Energy Efficiency

of WSNs, are Timing-Sync Protocol for Sensor Networks (TPSN) [37], Reference Broadcast Synchronization (RBS) [38], Flooding Time Synchronization Protocol (FTSP) [39], Lightweight Time Synchronization (LTS) [40], Pairwise Broadcast Synchronization (PBS) [41] and Time Diffusion Protocol (TDP) [42]. Because of their intended use in WSN applications, energy efficiency is addressed in all of them. Nonetheless, as most of these synchronization mechanisms are based on the bidirectional synchronization method, the number of packet transmissions in order to establish synchronization can be rather high. However, FTSP is based on the unilateral approach and therefore limits the number of packet transmissions. To address delay components, however, low level time-stamping is used and several packets are sent to allow for the estimation of clock-skew via linear regression.

When time synchronization has its main purpose in energy savings, such as in the case targeted here, the number of communication packets used for establishing a common notion of time, which essentially describe the energy overhead for the synchronization algorithm, becomes one of the major concerns. However, at the same time the protocol has to maintain a certain degree of accuracy, allowing for energy-efficient, but stable communication.

2.5.2 Synchronized Duty-Cycling

As mentioned previously, typical sampling intervals in environmental monitoring WSNs are of the order of minutes. This is due to the measurement of typically slowly changing parameters, such as temperatures, humidities or gas concentrations. In networks with a large number of sensor nodes, this means that new data is sampled at a rather slow rate, but once sampling is due, a large number of nodes want to communicate their data to a central collection point. The lack of synchronization in this situations has a twofold impact. On the one hand, sampling should usually occur at the same time at all nodes in order to obtain comparable sensor samples. Lacking time synchronization in this case means that data correlation cannot be guaranteed. On the other hand, the transmission of the sampled data has to be coordinated. In a common on-demand protocol the nodes would attempt to transmit their data directly after sampling, which might lead to packet collisions. Using a scheduled transmission method without a common timebase only partially solves this issue, as different node clock skews can result in
overlapping transmission periods.

A synchronization algorithm, targeting these issues in the present application, should have certain properties. Synchronization should be network-wide, as all end-nodes are involved in the communication process. The synchronization should allow communication at any time, thus an algorithm that maintains synchronization of the nodes is preferable over on-demand synchronizing solutions. Resulting accuracy is not the main target of the protocol, but the best achievable accuracy, in order to minimize the energy requirements, is desired. As energy overhead is the main concern, the optimization goal for the solution is minimum energy, thus minimum communication overhead.

A major consequence of the limited communication to be performed, is that all pairwise synchronization schemes are inappropriate. Considering the goal of minimizing communication, a single synchronization message is the desired outcome. Using the broadcast capability of the RF-channel, the same message can be used to synchronize all nodes in the same broadcast

<table>
<thead>
<tr>
<th>Classification Factor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal/External</td>
<td>Describes whether the algorithm synchronizes network nodes to some sort of external clock source or only amongst each other</td>
</tr>
<tr>
<td>Always/On-demand</td>
<td>Describes whether nodes stay synchronized all the time or whether they are synchronized on-demand</td>
</tr>
<tr>
<td>All/Subset</td>
<td>Describes whether all nodes will be synchronized or only a subset of nodes</td>
</tr>
<tr>
<td>Rate/Offset</td>
<td>Describes whether nodes synchronize to the same clock rate or whether a clock offset is compensated for</td>
</tr>
<tr>
<td>Correction/Translation</td>
<td>Describes whether node clocks are corrected to the common time or whether the timescale is translated</td>
</tr>
</tbody>
</table>

Table 2.9: Main classification factors of time synchronization algorithms in Wireless Sensor Networks
domain. A single (unidirectional) synchronization message, does, to some extent, enable for the compensation of the phase offset between nodes, but it does not address the clock skew. However, as periodic resynchronization is required to address temporary changes, frequency information can be extracted from the time interval between two synchronization broadcasts.

As depicted in Figure 2.12, in an interval-based synchronization method, message transfer occurs periodically and unidirectionally. The node acting as the clock reference periodically broadcasts a synchronization packet, which can be interpreted as a typical Start Frame Delimiter (SFD) for the synchronization frame. Theoretically, there is no content necessary within this message, but, as the packet has to be transferred anyway, some global information might as well be included. If indeterministic delay components, such as access time or system load, are handled to be minimal, the transmission delay of the two following packets can be estimated as a constant value. Therefore, as indicated in Figure 2.12, we can define

$$\delta = t'_i - t_i = t'_2 - t_2 = t'_i - t_i . \quad (2.20)$$

This means, a synchronization message sent by node A at $t_i$ will arrive an unknown time later at node B at the time $t'_i$, with

$$t'_i = t_i + \delta . \quad (2.21)$$

Furthermore, when defining a constant interval $T$ between synchronization messages, node A will periodically send out synchronization messages with
its own understanding of periodicity $T$, namely
\[ C_A(T) = C_A(t_{i+1}) - C_A(t_i). \] (2.22)

Allowing node B to measure the interval between two consecutive synchronization messages will lead to a measured inter-packet length of
\[ C_B(C_A(T)) = C_B(t'_{i+1}) - C_B(t'_i). \] (2.23)

As the delay components of both transmissions are estimated as being constant, the real-time length of both intervals will be the same, thus
\[ C(C_A(T)) = C(C_B(C_A(T))), \] (2.24)

with
\[ C(T) = T. \] (2.25)

At this point, all receiver nodes only know their interpretation of the reference interval $C_A(T)$ and the ideal synchronization interval $T$. They do not know the real synchronization interval itself. However, as for communication purposes, internal synchronization is usually sufficient and the nodes can treat the measured interval as the ideal interval. This leads to the fact that the nodes do not synchronize to real time $C(t)$, but to the reference's interpretation of the real time $C_A(t)$. For the previous example, we can therefore redefine node B’s point of view as
\[ C_B(C_A(T)) := C_B(T). \] (2.26)

This will be false in the majority of cases and only holds true if
\[ C_A(T) = C(T). \] (2.27)

However, for the purpose of internal synchronization, the redefinition does not introduce any error, as long as node A is defined as holding the reference clock.

Knowing both, $T$ and $C_B(T)$, node B can simply calculate its difference
from the reference clock

\[ D(T) = T - C_B(T), \]  \hspace{1cm} (2.28)

wherein a positive \( D(T) \) denotes a slow clock compared to the reference and a negative \( D(T) \) a fast clock. Furthermore, by knowing the time interval \( T \) over which this clock difference occurred, node B has the possibility to estimate its drift rate relative to the reference node and thereby translate any point in time to its own timebase, using

\[ C_B(t) = t - \frac{D(T) \cdot t}{T}. \] \hspace{1cm} (2.29)

This translation is useful, when it comes to communication schedules, as the synchronization reference, which should be the clusterhead of a communication cluster, can schedule communication in terms of its own timebase. All nodes willing to communicate will then translate their allocated communication times to their local timebase. Hence, all nodes will communicate in a schedule, based on the reference timebase, avoiding overlapping communications (i.e., collisions). Additionally, local measurement of the reference synchronization period allows for unused time being spent in low-power mode, while waking up in a precise manner for the reception of the next synchronization message (i.e., the start of a new communication frame) or any other activity scheduled to be performed. A general time-line of the synchronization procedure is given in Figure 2.13. This provides a closer look at what happens each time a synchronization message is sent in Figure 2.12.

With ideal synchronization, waiting for the synchronization packet reception (colored in Figure 2.13) would not be necessary. However, because there are variations in local clock frequencies due to environmental parameters, quantization errors in reading and setting timer events, as well as variations in component wakeup times, a certain guard-time is desired. This guard-time should guarantee that the node is in a state allowing the reception of the synchronization message, once it occurs. Nonetheless, at the same time the guard-time should be as short as possible, while fulfilling the previously mentioned task.

An important measure to define the length of the guard-time is the accuracy of the wake-up time. This value is based on, but is not limited to, the
### 2.5 Energy-efficient Communication

**Figure 2.13:** General sequence of tasks performed by synchronization master and slave at the beginning of each communication frame

<table>
<thead>
<tr>
<th>Master</th>
<th>MCU wakeup</th>
<th>Radio wakeup</th>
<th>Sync transmission</th>
<th>Radio sleep</th>
<th>Local processing</th>
<th>MCU sleep</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slave</td>
<td>MCU wakeup</td>
<td>Wait for Sync</td>
<td>Sync verification</td>
<td>Radio sleep</td>
<td>Local processing</td>
<td>MCU sleep</td>
</tr>
</tbody>
</table>

synchronization accuracy. Figure 2.14 provides the measurement results of the wake-up accuracy for consecutive synchronization messages. Synchronization packets are sent periodically from the reference and received at the slave according to Figure 2.12. Radio wake-up at the slave is scheduled to be completed at a defined time period $T_{guard}$ before the synchronization packet is received. Wake-up accuracy is measured by measuring the deviations from the actual wake-up time to $T_{guard}$. In order to conduct the experiment, protocols for both master and slave have been implemented in Embedded-C code on the Sentio-e² platform. Sentio-e² is a sensor node platform, equipped with an MSP430 microcontroller and a CC1101 radio transceiver. Measurements were performed using a digiview logic analyzer with a time resolution of 10 ns and have been repeated for different synchronization interval lengths. Statistical results of these measurements are summarized in Table 2.10.

The major sources of inaccuracy in the synchronization result from the interval measurement process. Because the time measurements in low-power mode are taken with an energy conserving auxiliary oscillator, which is sourced from a 32 kHz tuning-fork crystal, the maximum time resolution achievable is about 30.5 μs. This means, that a time period might be measured up to one clock cycle (30.5 μs) shorter than it actually is. Additionally, this leads to a shift in the wake-up time, leading to a wake-up of up to one clock cycle earlier than expected. This is the main reason for the offset from the predefined wake-up period, shown in Figure 2.14 and Table 2.10.
Moreover, adjustments can merely be taken in the number of clock cycles and thus allow no higher resolution. The results further show a maximum variation of about 100 μs (3 clock cycles) between two consecutive wake-ups. In addition, deviation does not appear to depend on the interval length, which is an expected result due to frequency offset compensation. However, these measurements were obtained under laboratory conditions (i.e., rather constant environmental parameters). In real deployments, increasing interval length also increases the probability of a change in environmental conditions. This can have a negative impact on the accuracy due to frequency drift.

One of the main influences on crystal frequency drift, as mentioned at an earlier stage, is crystal temperature, which is directly influenced by environmental temperature. Typically used crystals for low-power time keeping are 32 kHz tuning-fork crystals, which are cut in such a way as to provide a parabolic temperature dependence. This leads to a temperature dependent
clock frequency of

\[ f_{32kHz}(T_{env}) = \left[ 1 - B \cdot (T_{env} - T_0)^2 \right] \cdot f_{32kHz}(T_0), \]  

(2.30)

where \( T_{env} \) is the current environmental temperature, \( T_0 \) the turnover temperature and \( B \) the parabolic coefficient of the used crystal. Commonly mentioned parabolic coefficients are of the order of \(-0.04 \cdot 10^{-6} \text{°C}^{-2}\) with a typical turnover temperature of 25 °C.

Figures 2.15a, 2.15c, 2.16a and 2.16c show the effect on wake-up deviation under changing temperature conditions for both, heating and cooling conditions. While environmental temperature usually changes slowly, there are situations in which sudden temperature changes can occur. One example is depicted by the measurement results in Figure 2.17. In this case, a sensor node was enclosed in a typical housing unit made of light-colored plastic,

![Figure 2.15](image-url)

Figure 2.15: Measurement of wake-up accuracy under changing temperature conditions in the oven – (a) 2 min synch. interval, uncompensated; (b) 2 min synch. interval, compensated; (c) 5 min synch. interval, uncompensated; (d) 5 min synch. interval, compensated
then placed in direct sunlight. Similar situations might occur when a node was situated in the shade for a while, before being exposed to direct sunlight. The temperature inside the box was logged using an on-board temperature sensor with a rate of 1 Hz over a period of 40 min. It can be seen, that a rapid temperature change in the first minutes after exposure can be expected.

Knowledge of the temperature of the clock crystal enables compensation to be made for the drift occurring due to deviation from turnover temperature. Implementing a low-power temperature sensor on the sensor node is inexpensive and is often required for calibration or application purposes. When used for temperature drift compensation, however, close spatial location to the crystal should be ensured in order to achieve as accurate readings of crystal temperature as possible.

An initial measurement of clock drift over increasing temperature for
2.5 Energy-efficient Communication

Figure 2.17: Measurement of in-box temperature of a gray plastic enclosure under direct sunlight in summer conditions; location Sundsvall/Sweden; measurement with on-board temperature sensor of Sentio-e², 1 Hz

Figure 2.18: Comparison of measured, estimated and datasheet temperature drift behavior; measured with on-board temperature sensor of Sentio-e²; determined from drift over 1 minute while heated in an oven

The Sentio-e² platform has been conducted, using its on-board temperature sensor (see Figure 2.18). The graph shows a comparison of measured drift-rate and theoretical drift-rate according to datasheet parameters, provided by the crystal manufacturer⁹. As the match of these two showed some accuracy limitations, an estimation of clock drift based on a second order polynomial

---

⁹The crystal is a Epson-Toyocom FC-135
2 Energy Efficiency

has been added. The estimation is obtained in the form of

$$
\Delta_{est}(T_{env}) = a \cdot (T_{env} - T_0)^2 + b \cdot (T_{env} - T_0) + c,
$$

(2.31)

where $\Delta_{est}$ is the estimated drift and $a, b, c$ are the calibration parameters.

The results show, that if high accuracy is required, an initial calibration run can provide accuracy within 1 ppm. However, in the case where the calibration is too time and cost expensive, datasheet values can be used to predict occurring clock drift with restricted accuracies. For measurements shown in Figures 2.15 and 2.16, a calibrated temperature compensation method has been implemented. Comparing these results with those for uncompensated results, illustrates that temperature dependent clock drift can be almost completely eliminated by using a temperature compensation technique based on crystal temperature measurement.

Nonetheless, measuring the crystal temperature adds additional costs to the synchronization algorithm. Sampling the temperature sensor periodically during the inactive period requires the processor to wake-up and read the current temperature value. Furthermore, additional active time is mandatory to process the taken values and calculate the expected drift according to equation 2.31. This has to be accomplished once the last temperature reading has been sampled and before the radio wake-up should occur, allowing for an adjustment of the scheduled wake-up time. These necessary activities will increase the microcontroller duty cycle and therefore lower the energy-efficiency. On the other hand, temperature drift can also be handled by applying longer guard-times, while not compensating for changing temperature. While this consumes no sampling related energy, the radio module has to be in the reception state for a longer time, thus spending more energy.

As temperature drift has to be dealt with, the question remains which of the methods is the more energy-efficient.

Estimating additional energy costs by adding expected power consumption levels and periods for each of the methods can provide the answer. This leads to

$$
P_{tmp\text{Comp}} = \frac{T_{meas} \cdot N_{meas} \cdot P_{meas}}{T_{frame}} + T_{process} \cdot P_{process}
$$

(2.32)

for a temperature compensating approach. In this case, $T_{meas}$ is the time period for taking a sample from the temperature sensor, $N_{meas}$ are the number
2.5 Energy-efficient Communication

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{meas}$</td>
<td>950 $\mu$s</td>
<td>Time for measurement of a temperature sample</td>
</tr>
<tr>
<td>$T_{frame}$</td>
<td>60 s to 600 s</td>
<td>Time of the synchronization interval</td>
</tr>
<tr>
<td>$T_{process}$</td>
<td>150 $\mu$s</td>
<td>Time for processing the temperature drift</td>
</tr>
<tr>
<td>$T_{sync}$</td>
<td>2 ms</td>
<td>Time for receiving the sync packet</td>
</tr>
<tr>
<td>$P_{meas}$</td>
<td>2 mA</td>
<td>Consumption during temperature sampling</td>
</tr>
<tr>
<td>$P_{process}$</td>
<td>2 mA</td>
<td>Consumption during processing</td>
</tr>
<tr>
<td>$P_{RX}$</td>
<td>20 mA</td>
<td>Consumption when radio in reception state</td>
</tr>
</tbody>
</table>

Table 2.11: Parameter overview for estimations in Figure 2.19, obtained from measurements on Sentio-e²

of temperature samples in one synchronization frame, $P_{meas}$ the consumption level during a sample, $T_{frame}$ the time of a synchronization frame, while $T_{process}$ and $P_{process}$ are the time and power consumption of processing the temperature drift respectively.

Likewise, the consumption for providing an additional guard-period can be summed up as

$$P_{guard} = \Delta(\theta_{dif,max}) \cdot T_{frame} \cdot P_{RX},$$

(2.33)

with $P_{RX}$ being the power consumption level in reception state, $T_{frame}$ the synchronization period as above, and $\Delta(\theta_{dif,max})$ the drift rate, determined by the maximum temperature difference from the turnover temperature.

An estimation was conducted with parameters, measured on Sentio-e², which are listed in Table 2.11. The respective results obtained are shown in Figure 2.19. In addition to the compensation cost, the energy demands for the reception of the synchronization message have been included, which results in the complete energy cost of the temperature-robust synchronization algorithm to be provided. The calculations show that temperature compensation is the more energy-efficient solution in almost all cases. Providing enlarged guard-periods is only recommended if, during the whole deployment time, no large temperature variations are expected.

Finally, integrated RTC circuits become an essential part in many products and offer ever increasing functionality. As has been presented in Section 2.4, our current node platform uses such an RTC, which operates with a
Figure 2.19: Comparison of power consumption needed to deal with temperature drift – (a) Temperature compensation with different temperature sampling rates; (b) Extending reception guard-time for different maximum deviations from the turnover temperature temperature-compensated internal quartz crystal. While the synchronization method, which has been presented here, was also implemented on SENTIO-em, a comparison of the cost-performance ratio with respect to software and hardware based temperature compensation is yet to be made.
3 Solar Energy Harvesting

While the focus of the previous chapter lay on the increase of energy efficiency in order to prolong the sensor node lifetime, in this chapter solar energy harvesting will be addressed as a method to increase the energy supply to wireless sensor nodes and, thus, prolong their lifetime.

After providing the motivation for the usage of energy harvesting in general, and solar energy harvesting in particular, as the power supply for wireless sensor nodes in the environmental monitoring domain, solar energy harvesting will be addressed threefold. Firstly, we will address solar energy harvesting architectures, wherein the focus lies on long system lifetime and low irradiance conditions. Secondly, we will address solar energy harvesting simulations with a focus on architecture dimensioning. Finally, we will present the method and implementation of a solar energy harvesting testbed in order to simplify architectural evaluations and provide systematic comparison possibilities.

3.1 Solar Energy Harvesting

Although sensor nodes in Wireless Sensor Networks can typically be considered as low-power devices and therefore consume rather small currents over time, any energy source with finite capacity (e.g., batteries) will provide energy for only a limited amount of time. As was mentioned previously, this contradicts the desire for autonomous measurement systems to operate in a perpetual manner.

Figure 3.1 shows a simplified analysis of battery lifetime for varying load currents. Simplification is made in several ways, namely, the nominal battery capacity (in this case 2000 mAh) is assumed to be the real capacity, that the environment will not degrade capacity, no self-discharge is included and no energy is spent for voltage conversion inefficiencies (here assumed from 1.5 V
3 Solar Energy Harvesting

Figure 3.1: Simplified energy supply time from an AA-type battery in ideal situations

to 3 V). Hence, the illustrated case thus presents a very optimistic lifetime and a lower lifetime should be expected in reality.

Nevertheless, this qualitative description illustrates, that the lifetime for sensor nodes, powered by batteries, can be very limited and decreases dramatically with increasing consumption. One possible solution is to increase the number of batteries and thus accumulate capacity. However, this only partially solves the problem, as the available capacity will remain limited. In addition, this solution introduces some undesirable drawbacks. These include the linearly increasing size and cost of the system caused by the increasing number of batteries. Additionally, there is an increasing strain placed on the environment, especially if systems are deployed in locations where it is impossible to recover the system. Alternatively, rechargeable batteries can be implemented, thus reducing the number of battery exchanges. However, recharging still demands a considerable amount of manpower and is not usually a feasible option in the field and the usable capacity is reduced due to the greatly increased self-discharge rates, which may be as high as 10% to 20% per year [43]. Moreover, rechargeable batteries display a more rapid effect of degradation, due to the chemical process, which occurs during the charge and discharge at the electrodes. This leads to a reduced component lifetime for these batteries, typically of the order of a few years for commonly used technologies.

A desirable solution, in these scenarios, is to use an energy storage device with limited capacity, that can be autonomously recharged by using ambient energy sources. As ambient sources, such as wind or sun, can be classified
3.1 Solar Energy Harvesting

<table>
<thead>
<tr>
<th>Energy Source</th>
<th>Power Density</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solar</td>
<td>100 mW cm⁻²</td>
<td>Direct sunlight, outdoors</td>
</tr>
<tr>
<td>Wind</td>
<td>100 mW cm⁻²</td>
<td>9 m s⁻¹ wind speed, 10 m altitude</td>
</tr>
<tr>
<td>Ambient RF</td>
<td>&lt; 1 μW cm⁻²</td>
<td>Not close to emission source</td>
</tr>
<tr>
<td>Thermo-Electric</td>
<td>60 μW cm⁻²</td>
<td>5 °C temperature difference</td>
</tr>
</tbody>
</table>

Table 3.1: Typical energy harvesting sources and their power levels available in outdoor environments; based on [30, 44]

as inexhaustible energy sources – meaning that they have no capacity limit – it becomes possible to perpetually replenish a used storage device with limited capacity. From the load perspective, an unlimited energy supply has thus been attached. Nevertheless, for this to operate successfully, the conversion rate, consumption rate, storage capacity and storage lifetime have to be properties which are all matched. This means that, while the energy can be perpetually supplied from the ambient sources, this supply is limited to certain rates. Thus, as opposed to the capacity-limited energy source that batteries do represent, energy harvesting can typically be classified as rate-limited.

Table 3.1 provides an overview of ambient energy sources, commonly used in energy harvesting with their respective power densities. While conversion from solar and wind both provide sufficient power densities, solar energy conversion possesses some advantages over wind. Although both technologies can be considered as relatively mature conversion techniques, solar energy conversion is additionally readily scalable in size. Furthermore, solar energy involves no mechanical parts, which are exposed to limited component lifetime and/or require maintenance. Solar energy harvesting, thus, has been a popular choice in Wireless Sensor Networks that are deployed in outdoor environments, such as is typical in the environmental monitoring domain.

3.1.1 Energy Neutral Operation

Although solar energy harvesting can provide high power levels as compared to other ambient energy sources, it is not only rate-limited, but its rate is temporally and spatially dependent. Despite these variations in supply levels
that can be harvested, it is desirable to have an uninterrupted supply of the attached system (i.e., the sensor nodes). Therefore, the load power consumption has to match the power supply rate of the harvesting system. A metric, called the energy neutral operation [45], can be used in order to describe the relationship between consumption and supply, providing energy to the load at all times. Discussions concerning an energy neutral operation, however, only make sense in connection with energy sources of infinite energy supply capability, while being restricted by certain supply rates. This, for example, means that it is not possible for a primary battery to operate in an energy neutral manner, because its energy reservoir is finite.

The simplest form of an energy neutral operation is the direct connection of the energy harvesting source to the load. In this case, the supplied power has to exceed the consumed power at any time, such as

\[ P_s(t) \geq P_c(t), \quad (3.1) \]

with \( P_s(t) \) being the supplied power and \( P_c(t) \) the consumed power at time \( t \) respectively. However, in this scenario all the energy of times \( P_s(t) > P_c(t) \) and \( P_s(t) < P_c(t) \) is wasted. Moreover, in solar energy harvesting this solution does not provide an uninterrupted supply, since \( P_s(t) \) regularly becomes zero (e.g., at night), while \( P_c(t) \) in real systems is always greater than zero.

As an uninterrupted operation is a desired system property, for solar energy harvesting systems, an energy buffer has to be implemented, so that

\[ P_s(t) + P(W_b, t) \geq P_c(t). \quad (3.2) \]

In this case \( P(W_b, t) \) describes the power that can be supplied by the storage device at time \( t \), depending on the energy \( W_b \) which is simultaneously stored in the buffer. According to [45], over the whole system lifetime an energy neutral operation requires the following condition:

\[
B_o + \eta \int_0^T [P_s(t) - P_c(t)]^+ dt - \int_0^T [P_c(t) - P_s(t)]^+ dt - \int_0^T P_{\text{leak}}(t) \, dt \geq 0 \quad \forall T \in [0, \infty)
\quad (3.3)
\]
with $B_0$ being the initial energy stored in the buffer, $P_{\text{leak}}$ the losses of the buffer, $\eta$ the charging efficiencies, and $[x]^+$ the rectifying function

$$[x]^+ = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (3.4)

This does not include any capacity limitations of the used storage device, which in real systems is an important characteristic, leading to a waste of harvested energy. To account for this, we can further substitute $P_s(t)$ in equation 3.3 with

$$P'_s(t) = \begin{cases} P_s(t) & \text{if } B(t) < 100 \% \\ 0 & \text{otherwise} \end{cases}$$ \hspace{1cm} (3.5)

where $B(t)$ is the charge level of the buffer at time $t$.

### 3.2 Solar Energy Harvesting Architectures

The desire for an uninterrupted operation over the whole system lifetime, and therefore an energy neutral operation, means that a direct connection of energy converter and load becomes impossible. Instead, additional circuitry is required for an energy neutral operation to become a reality. Figure 3.2 depicts a generalized energy harvesting architecture with its typical modules. While input and output regulations are not required for an energy neutral operation to be fulfilled, these modules are typically required in order to allow for a reliable operation. Furthermore, they can increase the performance of the energy harvesting circuitry in use.

In particular for solar energy harvesting, the input regulation is necessary.
in order to convert the fluctuations of the solar panel output to appropriate levels for the charging of the energy storage. Additional intelligence can be provided, which allows for the power extraction to occur in the most efficient manner. The energy storage devices, which are typically utilized in energy harvesting systems, furthermore, require charge control, which protects the system from damage due to, for example, over- or undercharging. Finally, the output regulation adjusts the output of the energy buffer used to the respective input requirements of the attached sensor node, which acts as the load to the power supply circuitry.

While this basic structure is underlying all systems, the implementation of the individual modules can differ, depending on the focus and application of the individual systems, respectively. Utilized energy storage buffers, for example, range from rechargeable batteries to supercapacitors (also known as ultracapacitors or Double Layer Capacitors (DLCs)). Examples for these systems are found in [47, 48] for lithium-based rechargeable batteries, in [49–51] for nickel-based rechargeable batteries and in [52, 53] for supercapacitor-based systems. Additionally, some systems attempt to accommodate the advantages of both storage types by implementing hybrid storage solutions, such as in [54, 55]. Figure 3.3 depicts the relationship between these storage devices with regards to their energy and power density. The time values in

![Figure 3.3: Classification of energy density and power density of energy storage devices, obtained from [46]](image-url)
3.2 Solar Energy Harvesting Architectures

<table>
<thead>
<tr>
<th>Type</th>
<th>Voltage [V]</th>
<th>Energy Density [Wh kg(^{-1})]</th>
<th>Self-discharge [%/month]</th>
<th>Cycles [#]</th>
<th>Toxicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead-Acid</td>
<td>2</td>
<td>30-50</td>
<td>5</td>
<td>200-300</td>
<td>high</td>
</tr>
<tr>
<td>NiCd</td>
<td>1.2</td>
<td>40-80</td>
<td>20</td>
<td>1500</td>
<td>high</td>
</tr>
<tr>
<td>NiMH</td>
<td>1.2</td>
<td>60-120</td>
<td>30</td>
<td>500</td>
<td>low</td>
</tr>
<tr>
<td>Li-Ion</td>
<td>3.6</td>
<td>100-150</td>
<td>&lt; 10</td>
<td>1000</td>
<td>low</td>
</tr>
</tbody>
</table>

Table 3.2: Overview of main characteristics for different, typically used battery types; based on [43, 56]

the graph give a rough idea of the respective charge/discharge periods.

Although rechargeable batteries provide larger energy densities and have smaller self-discharge rates, which result in an energy storage for longer periods of time in comparison to supercapacitors, they have a very limited amount of charge/discharge cycles and, thus, have a limited lifetime. Supercapacitors, on the other hand, provide long component lifetimes due to charge/discharge mechanisms based on charge separation instead of chemical reactions. Furthermore, supercapacitors have higher power densities than batteries, which means that they can be charged and discharged at high rates, which allows for the effective utilization of short irradiation periods in solar energy harvesting. Nevertheless, supercapacitors suffer from relatively high discharge rates and low energy densities, which makes them usable only as a short-term energy buffer. This can introduce problems during longer periods of limited solar irradiation. While Lithium-Ion (Li-Ion) batteries provide some beneficial advantages, such as very low self-discharge, high energy density and nominal voltage levels that typically do not require adaptation to sensor node levels, one major problem associated with this type is its rather complex charging behavior. It is usual to have a dedicated charging circuit when implementing Li-Ion batteries to allow for their proper use, which introduces extra cost and complexity to the system. Table 3.2 provides an overview of the characteristics of the different rechargeable battery types, which are typically used in combination with solar energy harvesting architectures.

Despite the individual charging mechanisms and protection circuits, input regulators are often used to increase harvesting efficiencies by forcing the solar panel to operate at the Maximum Power Point (MPP). Solar panels behave
3 Solar Energy Harvesting

like a voltage-dependent current source, where the current is a function of the irradiance level. This leads to a typical current-voltage characteristic (often referred to as a solar panel’s IV-curve) as depicted in Figure 3.4a. For good quality solar cells (i.e., small serial resistance and large shunt resistance) the current remains almost constant up to the achievement of a particular voltage, then breaks down rapidly until zero is reached. When charting the power levels for each of the respective voltage levels, as shown in Figure 3.4b, a clear operating point of maximum power extraction is obvious (the MPP). Therefore, solar panels are often described according to three distinct parameters, the open-circuit voltage $V_{oc}$, the short-circuit current $I_{sc}$ and the Maximum Power Point ($V_{mpp}, I_{mpp}$). As the MPP is dependent on the environmental condition, however, an estimation of the optimal working point and a control circuitry that forces the solar panel to the respective voltage is required for power extraction to reach a maximum. This is typically referred to as Maximum Power Point Tracking (MPPT).

Nevertheless, performing MPPT also involves some system costs. On the one hand additional circuitry is required, which will increase the size and monetary costs, while on the other hand these required functionalities will consume energy themselves. The latter cost, in particular, depends on the accuracy, and therefore the method, of the MPPT. Accurate methods, such as the hill-climbing method [57, 58], continuously measure the power output of

![Figure 3.4: Solar panel characteristics of a 450 mW panel with 9 cells in series at different irradiance levels – (a) I-V characteristic and (b) P-V characteristic](image-url)
the solar panel, while slowly adjusting the terminal voltage. While this allows for the system to operate at, or very close to, the Maximum Power Point of the solar panel, this method is very costly, because measurements and adjustments are required to be performed at a high rate. Furthermore, depending on the step-size, the method can be rather slow to react to abrupt changes (e.g., due to abruptly changing irradiance conditions). Thus, a compromise has to be made for the perturbation step-size, as a small step-size leads to long reaction times, while a large step-size leads to a strong influence from the oscillation around the actual MPP.

Because the solar panels, which are typically used to supply sensor nodes, are of limited area and power output, the overhead costs of the Maximum Power Point Tracking solution can have a considerable impact on the system efficiency. Therefore hill-climbing methods (also similar methods, such as Perturb-and-Observe (PnO)) are too costly to implement. Simpler, less accurate methods are usually used, such as the fractional open-circuit voltage or fractional short-circuit current [59] methods. These two methods are based on the approximation of the relationship between the MPP and the open-circuit voltage $V_{oc}$ and the short-circuit current $I_{sc}$ to be linear. This leads to

$$V_{mpp} \approx k_V \cdot V_{oc}$$  \hspace{1cm} (3.6)

$$I_{mpp} \approx k_I \cdot I_{sc}$$  \hspace{1cm} (3.7)

where $V_{mpp}$ and $I_{mpp}$ are the voltage and current at the Maximum Power Point, while $k_V$ and $k_I$ are the proportionality factors, approximated as being constant for varying environmental conditions. However, it has to be borne in mind that this is only an approximation, as shown in Table 3.3. In this case it can be seen that the factors $k_V$ and $k_I$ change with the irradiance level. The achieved accuracy is still relatively good as can be seen through the effect on the MPP estimation, shown in Figure 3.5. Estimation did occur by averaging $k_V$ and $k_I$ respectively. It can be observed that accuracies for lower irradiance conditions are worse than those for stronger irradiance. In addition, these results show that for the particular solar panel under test, the fractional open-circuit voltage method provides better estimations in the majority of cases. Nonetheless, the mentioned factors are solar panel dependent, meaning that each individual panel has to be tested in order to achieve the optimal outcome. An analysis of panels from the same type can provide an idea
3 Solar Energy Harvesting

of variations within the same manufacturer. If the variation is found to be small, this could allow for some generalizations. Furthermore, although this method is usually less costly than the hill-climbing methods, there are still necessary costs associated with the performance of both measurements and adjustments. Moreover, in order to be able to react to changing irradiance conditions, sampling and adjustment rates are required to remain high.

As opposed to the input regulation, the output regulation in most systems is rather straightforward. While systems based on Li-Ion batteries, due to their high nominal voltage, might often suffice without output regulation, most

| $E$ [W m$^{-2}$] | $V_{mpp}$ [V] | $I_{mpp}$ [mA] | $V_{oc}$ [V] | $I_{sc}$ [mA] | $k_V$ | $k_I$
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>3.87</td>
<td>83.18</td>
<td>5.04</td>
<td>90.77</td>
<td>0.767</td>
<td>0.916</td>
</tr>
<tr>
<td>500</td>
<td>3.88</td>
<td>41.34</td>
<td>4.84</td>
<td>45.39</td>
<td>0.810</td>
<td>0.910</td>
</tr>
<tr>
<td>250</td>
<td>3.79</td>
<td>20.22</td>
<td>4.64</td>
<td>22.69</td>
<td>0.816</td>
<td>0.891</td>
</tr>
<tr>
<td>100</td>
<td>3.58</td>
<td>7.57</td>
<td>4.37</td>
<td>9.08</td>
<td>0.819</td>
<td>0.833</td>
</tr>
<tr>
<td>25</td>
<td>2.98</td>
<td>1.39</td>
<td>3.85</td>
<td>2.27</td>
<td>0.774</td>
<td>0.612</td>
</tr>
</tbody>
</table>

Table 3.3: Overview of measured characterization parameters of a 450 mW silicon solar panel

![Figure 3.5: Accuracy of MPPT based on fractional open-circuit voltage and fractional short-circuit current; real MPP marked by cross, estimated from $V_{oc}$ marked by squares, estimated from $I_{sc}$ marked by circles](image-url)
3.2 Solar Energy Harvesting Architectures

other storage types require a boost for their voltage levels in order to fulfill load demands. In the majority of cases, this involves the implementation of a standard step-up regulator which boosts the storage voltage according to

\[ V_{out} = V_{store} \cdot \frac{I_{store}}{I_{out}} \cdot \eta, \]  

(3.8)

where \( V_{out} \), \( V_{store} \), \( I_{out} \) and \( I_{store} \) are the voltages and currents of the energy buffer and the regulator output respectively, and \( \eta \) is the conversion efficiency. While consumption of these regulators can be relatively low, the converter efficiency is an important parameter for the selection process. Alternatively, multiple energy buffers can be operated in a series connection in order to increase the output voltage. While this is rather conventional for low-voltage batteries (e.g., NiMH, NiCd), Double Layer Capacitors can require external circuitry in order to allow for them to be operated in this manner.

3.2.1 Low Irradiance Conditions

While many solar energy harvesting systems for Wireless Sensor Networks have been proposed in the literature, the majority of these systems originate from locations where annual solar radiation is high. Typically, with reference to the design considerations of the systems, several hours of strong sunlight are expected each day [47, 55, 60]. This might be the case in the respective locations, but does not hold true in general. Figure 3.6 provides an overview of the solar radiation during 2008 in Sundsvall, Sweden (62° 24′ N, 17° 19′ E). In locations such as these, highly unequal irradiance distributions can be observed. While this might lead, in general, to acceptable solar radiation over the year, the solar irradiance during certain periods in the year can be immensely low. Furthermore, even at locations where solar radiation is usually high, limited irradiance conditions can occur due to shading by objects (e.g., trees and buildings). In situations involving frequently limited irradiance levels, different design considerations have to be applied rather than those in the previously mentioned systems.

Although batteries can buffer considerable amounts of energy and, thus, supply the sensor node during long periods of weak irradiance conditions, their chemical composition and restricted lifetime makes them an undesired technology for the implementation of energy buffers in solar energy harvest-
As a result, DLC based solar energy harvesting has been analyzed for its capabilities to be applied in low-irradiance conditions. Due to their limited capacity, DLCs can only be used as short-term energy buffers. As in the targeted environments, however, relatively long periods of low-irradiance levels are to be expected, the architecture in use has to allow for energy to be harvested even from these weak conditions. The energy overhead of the harvesting architecture, thus, becomes one of the critical design parameters.

In a first deployment, the feasibility of using DLC based harvesting architectures under low-irradiance conditions has been tested. For this evaluation, the load has been restricted to low power sensor nodes, and the main design goal was to implement a solar energy harvesting architecture that operates in as simple a manner as possible. The resulting architecture is depicted in Figure 3.7. The central component in the system is the Double Layer Capacitor, which operates as the short-term energy buffer. Charging of the DLC takes place in a rather direct manner, by having a direct coupling to the solar panel. In order to protect the solar panel from reverse currents, however, a Schottky-diode provides the necessary directionality. Furthermore, the DLC is required to be protected from over-charge, as the solar panel can provide
3.2 Solar Energy Harvesting Architectures

![Abstracted circuit diagram of a directly coupled energy harvesting architecture with DLC and charge protection.](image)

Figure 3.7: Abstracted circuit diagram of a directly coupled energy harvesting architecture with DLC and charge protection.

A higher terminal voltage than the nominal voltage of the energy buffer. In the easiest manner, this over-voltage protection is implemented by using a Zener-diode, which redirects current away from the DLC after a certain voltage threshold has been crossed. Zener-diodes, however, have a non-ideal behavior, which is demonstrated in Figure 3.8. While in an ideal scenario, the complete current would be redirected when the threshold (i.e., $V_{nom}$) is reached, in reality a Zener-diode would perform this in a gradual fashion. As a result, the Zener-diode would either allow for charging to be continued after the threshold voltage has been reached, or introduce considerable charging losses. In the presented architecture, thus, the ideal Zener-diode behavior has been reproduced with the support of a MOSFET and a voltage comparator, as depicted in Figure 3.7. The analog comparator monitors the terminal voltage of the DLC and continuously compares it with a preprogrammed reference voltage (i.e., the nominal voltage of the DLC). As soon as the threshold is reached, the comparator will trigger the MOSFET to disconnect the solar panel from the energy buffer and, thus, stop the charge process by shortening the solar panel terminals. In order to avoid oscillations around the nominal voltage, hysteresis was added to the comparator. Finally, the architecture includes an output regulator in form of a DC-DC step-up converter. Because the voltage of the DLC follows its charge level in an approximately linear fashion, this regulator was selected to step the DLC output over a wide range, which allows for the energy extraction even at low states of charge.
In order to investigate the architecture’s behavior and performance, it has been implemented with the components listed in Table 3.4. It was then deployed in a real-world setup during the period from November 2009 to January 2010 in Sundsvall, Sweden (62° 24′ N, 17° 19′ E). For the deployment location, a building roof on the university campus was chosen in order to provide an obstacle-free operation. The intention of the deployment was to observe whether the system could supply the load during this period, which represents the darkest period of the year at the chosen deployment site. Thus, correct operation would represent that the system is capable of supplying the load during the entire year. As correct operation was not guaranteed prior to the deployment, and system parameters should be monitored even if the energy storage is depleted, the load to the solar energy harvesting system and its monitoring system have been separated. Thus, the monitoring has been performed with a sensor node platform, whereas the load was implemented in the form of a resistor bank, which is connected by MOSFET switches. The load profile has been implemented in a bi-modal manner and has resulted in an average load current of 20 μA. System parameters and environmental conditions have been digitalized and transmitted in an interval of 5 min to a mains powered receiver, which forwarded the data via an General Packet Radio Service (GPRS) connection to a database server.

The deployment demonstrated that the system could supply the load over the entire evaluation period. Both the tested DLC capacities remained above the threshold of approximately 0.8 V, below which the output regulation will fail to provide a stable output. Figure 3.9 shows the behavior of the
3.2 Solar Energy Harvesting Architectures

<table>
<thead>
<tr>
<th>Module</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solar Panel</td>
<td>450 mW, $94 \times 61 \text{mm}^2$</td>
</tr>
<tr>
<td>DLC</td>
<td>Cooper Bussmann, $C = 10 \text{ F}$, $V_{nom} = 2.5 \text{ V}$</td>
</tr>
<tr>
<td></td>
<td>Cooper Bussmann, $C = 22 \text{ F}$, $V_{nom} = 2.5 \text{ V}$</td>
</tr>
<tr>
<td>Reverse Current Protection</td>
<td>Shottky Diode (MBR0520L)</td>
</tr>
<tr>
<td>Over-Voltage Protection</td>
<td>Comparator (MAX9017)</td>
</tr>
<tr>
<td></td>
<td>MOSFET (MGSF2N02EL)</td>
</tr>
<tr>
<td>Output Regulator</td>
<td>Boost (TPS61070)</td>
</tr>
<tr>
<td>Load System</td>
<td>Resistor bank</td>
</tr>
<tr>
<td>Monitoring System</td>
<td>Sentio-e$^2$</td>
</tr>
</tbody>
</table>

Table 3.4: Module configuration for the implementation of the DLC-based harvesting architecture

architecture during one week of its deployment. The results demonstrate the typical daily cycle of solar energy harvesting (Figure 3.9a), wherein the energy storage charges during periods of sunlight and discharges when solar irradiation is absent. The charging allows for the DLC to be fully replenished even under low irradiance conditions, which in the presented scenario have remained below $100\ \text{Wm}^{-2}$ at all times.

A limitation in this architecture can be observed with respect to the over-voltage protection of the energy buffer. The trigger point configuration of the analog comparator involves some variation in threshold accuracy and while the circuitry was dimensioned in order to react to a threshold of $2.5 \text{ V}$, which is the nominal voltage of the DLC in the system, the actual implementations showed variations from this trigger point, which resulted in a $2.45 \text{ V}$ and $2.55 \text{ V}$ threshold for the 10 F and 22 F capacitors, respectively. Furthermore, the hysteresis of about $400 \text{ mV}$ was over-dimensioned, which resulted in a bi-daily charge cycle for the 22 F DLC. Finally, the implemented method of over-voltage protection leads to a waste of available ambient energy, as it stops harvesting energy as soon as the storage is full. Due to the additional hysteresis, the DLC starts discharging at this point and will, in the majority of cases, not harvest more energy during the same day. However, this behavior reduces the stress on the DLC and, thus, is likely to increase its component lifetime.
Figure 3.9: System behavior of the solar energy harvesting architecture during one week of deployment – (a) voltage level of the DLC; (b) irradiance conditions during the week; (c) temperature and relative humidity during the week.
3.2 Solar Energy Harvesting Architectures

<table>
<thead>
<tr>
<th></th>
<th>Double Layer Capacitor</th>
<th>Thin-Film Battery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typical capacity</td>
<td>tens of F</td>
<td>μAh to mAh</td>
</tr>
<tr>
<td>Voltage range [V]</td>
<td>0 to 2.5</td>
<td>3 to 4.1</td>
</tr>
<tr>
<td>Temperature range</td>
<td>-25 to 70</td>
<td>-40 to 85</td>
</tr>
<tr>
<td>Internal resistance</td>
<td>tens of mΩ</td>
<td>tens of Ω</td>
</tr>
<tr>
<td>Recharge cycles</td>
<td>$10^5$ to $10^6$</td>
<td>10000</td>
</tr>
</tbody>
</table>

Table 3.5: Property comparison of double layer capacitors and thin-film batteries

3.2.2 Short-Term Energy Buffers

After it was demonstrated that it is feasible to supply low-power Wireless Sensor Networks from solar energy harvesting architectures based on short-term energy buffers (e.g., DLCs as previously presented), and that these architectures can even be utilized in low-irradiance conditions, a deeper analysis of such architectures has been conducted. The energy storage technology in these architectures has a major influence on the operation and behavior of the system, as individual technologies come with their respective properties and architectural requirements. Double Layer Capacitors and Thin-Film Batteries have considerable lifetime advantages over conventional secondary batteries, which becomes clear by comparing the properties of these technologies, listed in Tables 3.2 and 3.5. Their limited capacity, on the other hand, allows for energy buffering to be performed only for short periods of time. Nonetheless, due to their advantageous lifetime properties, and the limited existing experience with these technologies in solar energy harvesting applications, the comparison that is presented in this section has been focused on these short-term energy buffers.

While Double Layer Capacitors can be charged in a simple manner, their low nominal voltage requires the harvesting architecture to contain a step-up regulator to match the DLC’s terminal voltage to the input requirement of the load. Furthermore, DLCs have a relatively high self-discharge rate, which reduces their capability of buffering energy for longer periods of low-irradiation. This self-discharge rate increases with increasing capacity, which makes the selection of a larger DLC only of limited benefit to this challenge.

A recently commercialized alternative to DLCs are Thin-Film Batteries
Solar Energy Harvesting

Table 3.5 contrasts the properties of both device types, which shows that Thin-Film Batteries have a higher nominal voltage and, thus, can avoid the usage of boost regulators. These typically have a relative high quiescent current as opposed to LDOs. In addition, the TFBs have a near-zero self-discharge, which means that harvested energy is stored more efficiently. On the contrary, available devices are highly restricted in their energy capacity. During the writing of this document, 2.2 mAh TFBs were the largest devices, which were commercially available. Furthermore, currently available products have an internal resistance of tens of Ohms, which makes them susceptible to current peaks, such as those typically occurring in WSN applications. Although the amount of recharge cycles that can be performed is at least one magnitude lower than for DLCs, this is not considered a great drawback of the technology, as the number of recharge cycles of Thin-Film Batteries will be sufficient for a long system lifetime, even in the case of daily recharge cycles, which is typical for solar energy harvesting sensor nodes.

In order to evaluate the impact of these properties on the actual system operation, an experimental comparison of DLCs and TFBs, as well as their respective harvesting architectures, has been performed. For this, a configurable solar energy harvesting platform has been designed, which can be selectively operated with DLCs or TFBs, and offers a choice of harvesting circuits. Integrating all implementation options on one board, which can be configured to the desired implementation at hand, allows for the design overhead and the production cost to be kept to a minimum. The reconfiguration of a fully equipped platform is possible by simply adjusting the placement of a group of jumpers. An architectural overview of the configurable harvesting circuit is depicted in Figure 3.10. Overall, four different paths from the solar panel to the load can be chosen, wherein two are using the Double Layer Capacitor and two are using the Thin-Film Battery, respectively. For each storage device, a minimal harvesting approach is implemented, which limits the circuit overhead to a minimum. In addition, a more complex energy harvester can be utilized, which increases the overhead (i.e., circuit complexity, cost, and energy overhead), but attempts to make the charging process more efficient by using a simplified MPPT technique.

In addition to the energy harvesting circuit, the platform also contains
3.2 Solar Energy Harvesting Architectures

Figure 3.10: Architecture overview of the configurable energy harvesting module. The charge path is altered by the placement of a set of jumpers.

components for parameter monitoring. Being intended for the connection to a typical sensor node, the node’s controller can utilize these components to obtain the parameters of interest, such as the incoming current from the solar panel or the voltage level of the energy buffer that is being used. Other reference sensors, such as for ambient temperature or solar irradiance measurements, are not included on the harvesting platform, but are intended to be connected to the sensor node separately. Thus, following the distinction of power and sensor interface, which has been presented for the SENTIO-em platform in section 2.4. The form-factor of the configurable harvesting platform follows the same physical shape as SENTIO-em in order to allow easy insertion in its standard enclosure.

The experimental evaluation was conducted both under laboratory conditions and in a system deployment. While for the laboratory measurements external equipment has been used in order to characterize the devices performance, the deployment formed a typical in-situ evaluation, that allowed for results on long-term behavior and under real-world conditions to be obtained. For the deployment, a small network of four nodes has been implemented, which are each equipped with a configurable harvesting platform. Each platform operates on a different harvesting configuration and is monitored by a SENTIO-em platform. The obtained samples are transmitted to a gateway, which logs the data in a simple database for further access. In addition to the system parameters, temperature and humidity are logged for future reference. As direct comparison between the individual configurations
Figure 3.11: Experimental setup for the short-term energy storage architecture evaluation. The sensor nodes are placed in close proximity in order to have similar irradiation patterns.

Figure 3.12: Comparison of the charge behavior of (a) double layer capacitors and (b) thin-film batteries

should be possible, all nodes are placed at minimal distance from each other (see Figure 3.11 for a picture of the experimental setup). Furthermore, all the configurations operate on the same type of solar panel (i.e., 450 mW) and have an identical load profile of a low duty cycle sampling application (i.e., 5 mA|500 ms sampling, 25 mA|50 ms communication, 2 μA|5 min sleep). For the DLC a Cooper Bussmann PowerStor HB has been selected, which offers a 10 F capacity with a nominal voltage of 2.5 V. The TFB was a Thinergy MEC201-10P with a 1 mAh capacity at 4.1 V.

A first measurement of interest is the charge and discharge behavior of each energy storage device. This can easily be evaluated in a laboratory environment using standard instruments, such as power supplies and multimeters in order to provide defined measurement conditions. Figure 3.12 shows the
3.2 Solar Energy Harvesting Architectures

![Graphs showing charge behavior of double layer capacitors and thin-film batteries](image)

Figure 3.13: Peak discharge behavior of (a) double layer capacitors and (b) thin-film batteries

The charge behavior of both devices under conditions that are similar to strong irradiance using the previously mentioned solar panel (i.e., current-limited at 100 mA). The DLC shows a nearly linear voltage increase over time, whereas the TFB has a non-linear behavior with only a small change in terminal voltage. The obtained measurements confirm the negative influence of the high internal resistance in the TFBs, which results in much smaller charge currents than are available from the power supply. While the DLC constantly draws the maximal available charge current from the source, the TFB’s charge current is limited to a few hundred microampere. Thus, these devices require relatively long periods of irradiation as compared to the DLCs in order to replenish their energy reservoir.

Another influence of the internal resistance can be observed in the discharge behavior. As was previously mentioned, Wireless Sensor Networks typically pose a bi-modal load profile on their power supplies, which contains long periods with near-zero consumption, but which are interrupted with periodically occurring peaks of relative high consumption. Figure 3.13 depicts the behavior of the energy buffers when being exposed to such a load profile. In a similar manner to that during the charging, the effect on the DLC is minimal due to its low internal resistance, whereas the TFB shows a considerable reaction to the applied load peaks. In order to prevent the load system from being reset, large capacitors have to be utilized in order to maintain an appropriate supply voltage. The high resistance of TFBs, thus, affects both charging and discharging characteristics in a negative manner.

Finally, the system has been deployed in an evaluation network for the
3 Solar Energy Harvesting

Figure 3.14: Sample results of the configurable energy harvester deployment during summer time (a) DLC architectures, (b) TFB architectures

Figure 3.15: Sample results of the configurable energy harvester deployment during winter time (a) DLC architectures, (b) TFB architectures

analysis of its operation under real-world conditions. Figures 3.14 and 3.15 depict the results of a daily cycle for all architectures in summer and winter periods, respectively. In order to obtain better scalability, the plots for DLCs and TFBs have been separated from each other. The presented results can be used in order to evaluate the system performance on different levels, such as comparing the different charging architectures, the different storage technologies, and different operating periods. Figures 3.14a and 3.15a show that the behavior of both DLC architectures is similar. A clear distinction, however, is visible in the influence of the over-voltage protection. While the incorporated energy harvesting IC applies a constant voltage to the DLC, the directly coupled architecture repeatedly connects and disconnects the energy storage and the solar panel. As a consequence, charging is stopped...
3.2 Solar Energy Harvesting Architectures

prematurely in this architecture, which leads to a faster voltage decrease rate during periods of no irradiation. Similarly, a difference can be observed between the charge methods of the TFBs (Figures 3.14b, 3.15b). As the Thin-Film Batteries have a relatively high nominal voltage of the order of 4 V, they require a relatively high terminal voltage in order to be charged. The utilized solar panel, however, will only deliver an appropriate voltage if irradiation is high. In situations when the irradiation is not sufficiently strong, the TFB can benefit from the harvesting IC, as it boosts the voltage from the solar panel before it is applied to the energy storage device. Comparing, on the other hand, the DLC based architectures with those using a TFB, the differences are more profound. While on first view, the general behavior is similar, thus there is a higher voltage during daylight and lower/reducing voltage during darkness, the previously presented laboratory results can be used in order to explain some of the differences. In the case of the DLC, terminal voltage changes accurately with the state of charge and, hence, a slower increase and decrease rate can be observed in the graphs, despite the much higher charge currents, which can be achieved with this energy buffer. For the TFB, in contrast, terminal voltage changes much less with regards to the state of charge, whereas the major influence on the voltage is determined by the solar panel’s voltage due to the high internal resistance of the TFB. In addition to the effect on charge and discharge behavior, this presents another limiting factor of TFBs, as it makes the estimation of the state of charge a difficult process. Finally, the seasonal influence on the system behavior follows the expectations. All figures clearly indicate the differences in daylight periods between summer (Figure 3.14) and winter (Figure 3.15) due to the resulting differences in terminal voltages.

Although the previously presented results suggest that using the energy harvester IC improves the system operation, this is not the case in each and every situation. Figure 3.16 depicts the results of the two DLC architectures during a period of extremely low irradiation during the winter. The voltage curves show that the directly coupled architecture has a benefit in relation to these conditions, because its overhead is low in comparison to the energy harvesting IC. Furthermore, the utilized IC uses a simple MPPT method, and these methods have been demonstrated to be inaccurate, particularly under low irradiance conditions.
3 Solar Energy Harvesting

![Graph of MPPT utilization during periods of low irradiation for DLC based energy harvesting systems.](image)

Figure 3.16: MPPT utilization during periods of low irradiation for DLC based energy harvesting systems

### 3.3 Solar Energy Harvesting Dimensioning

While, in general, because of their inexhaustible energy source, solar energy harvesting systems allow for a perpetual energy supply, energy harvesting systems are limited by the supply rate, as was previously mentioned. Moreover, this rate limitation is spatially and temporally dependent. Thus, an energy neutral operation is only possible when supply rate conditions and consumption conditions are met. The proper planning of the solar energy harvesting system becomes an essential part in the design process. Figure 3.17 provides an overview of the constraint factors involved in solar energy harvesting systems. The goal in relation to system planning is to determine a set of required constraints under otherwise given constraints. A typical exemplary case would be the determination of the necessary harvester dimensions (e.g., solar panel rating and energy storage capacity), while the environmental and application constraints are fixed.

This component dimensioning is necessary for harvesting systems, regardless of the particular architecture chosen. Taking the architectures presented in the previous section as an example, straightforward parameters include DLC capacity and solar panel size. The choice of these parameters depends mainly on the power profile of the system load and the environmental conditions at the intended location (i.e., especially their irradiance levels). While on the one hand, under-dimensioning leads to unreliable operation, over-dimensioning is also undesirable as it leads to an unnecessary increase in both the size and cost of the system. Even though a systematic method of dimensioning system components is desired, the complexity of the system...
3.3 Solar Energy Harvesting Dimensioning

and the challenge of simplifying location-based influences makes analytical approaches difficult to handle. Using a system model of the underlying architecture in order to simulate the State-of-Charge for given conditions, allows for the components of the architecture to be dimensioned. The possibility and performance of this type of dimensioning method will be demonstrated in the following section, using the directly-coupled architecture (as in Figure 3.7) as a case example.

3.3.1 Modeling of the Solar Energy Harvesting Architecture

Figure 3.18 provides an overview of the main interactions in the target architecture, which results in its system behavior. Because of the direct coupling between the solar panel and DLC, the voltage of the solar panel will follow the voltage of the energy buffer (with a small offset due to the reverse current protection diode). At the same time, however, the charge current depends on their irradiance level. Thus, the operating point of the solar panel, at any point in time, is given by the irradiance and the DLC charge level. In return, this operating point defines the charge-rate of the DLC, changing its voltage level over time, leading again to a change of the operating point of the solar panel. Additionally, the DLC is discharged continuously by the load system at a rate defined by the load’s energy profile. This leads, in combination with the incoming energy from the solar panel, to an overall charge or discharge condition.
In order to model this dynamic system behavior, accurate modeling of the involved modules is important. Nonetheless, any unnecessary complexity of the model should be avoided in order to increase the development time and make the model flexible to changes. Therefore, a hybrid modeling approach was chosen, in which the main modules with complex behavior are modeled electrically, while the other modules are simplified using their logical behavior. Typical examples for electrically modeled components are the Double Layer Capacitor and solar panel, while protection functions represent the logically implemented functions.

Implementation has been conducted in Matlab/Simulink, requiring analytical representation of the above mentioned functions. Therefore, electrical models have been implemented according to Kirchhoff’s circuit laws. Figure 3.19 illustrates a commonly used equivalent circuit of solar panels, based on a single-diode model. From Kirchhoff’s laws we can represent the output voltage and current as

\[
V_{solar} = V_D - I_{solar} \cdot R_s \quad (3.9)
\]

\[
I_{solar} = I_{ph} - I_D - I_{sh} \quad (3.10)
\]
Figure 3.19: Equivalent circuit of a solar panel according to the single-diode model

with the notations as indicated in Figure 3.19. Using (3.9) in (3.10) and replacing \( I_D \) with Shockley’s diode equation, leads to a more detailed description of the resulting output current

\[
I_{solar} = I_{ph} - I_0 \cdot \left( e^{\frac{q(V_{solar} + I_{solar} \cdot R_s)}{n k T}} - 1 \right) - \frac{V_{solar} + I_{solar} \cdot R_s}{R_{sh}},
\]

(3.11)

where \( I_{ph} \) can be deduced as being proportional to the irradiance level, such as

\[
I_{ph} = k_{irr} \cdot E.
\]

(3.12)

Substituting \( I_{ph} \) in (3.11) with (3.12), the solar panel model is implementable using standard operations. This leaves the key modeling parameters as being the serial resistance \( R_s \), the shunt resistance \( R_{sh} \), the irradiance level \( E \), as well as the proportionality factor \( k_{irr} \). With the exception of the irradiance, these parameters are solar panel properties and can be determined by various methods, e.g. by analyzing I-V curve characteristics, such as proposed in [61].

Similarly, the DLC can be implemented according to a voltage and current analysis of the equivalent circuit, shown in Figure 3.20. The circuit shows a two-branch model, which has been introduced in [62]. The two branches represent the short-term \((R_0, C_0, k_v \cdot V)\) and long-term behaviors \((R_2, C_2)\) of the DLC respectively, while leakage is modeled by means of an additional Equivalent Parallel Resistance (EPR). Following the currents and voltages as
3 Solar Energy Harvesting

![Diagram of a double layer capacitor equivalent circuit](image)

Figure 3.20: Equivalent circuit of a double layer capacitor according to the two-branch model

In this case, \( R_0, C_0, K_V, R_2, C_2 \) and \( EPR \) are constants and can be determined according to the description presented in [62]. The determination of these parameters, however, has to be repeated for every type and size of DLC to be used in simulations, as simple deduction is not feasible.

Implementing the two main modules and filling connections between them by adding logical representations of protection mechanisms, it is possible to produce a system model of the energy harvesting power supply. At a high abstraction level, this system model can be described as depicted in Figure 3.21. While architectural parameters are defined inside the system model, such as solar panel type or DLC capacity, external conditions are provided as inputs to the model. The final output of the model is the voltage level of the Double Layer Capacitor at any simulation point, which allows for there to be an estimate for the amount of time when there is insufficient energy under...
### 3.3 Solar Energy Harvesting Dimensioning

#### Figure 3.21: Overview of the solar energy harvesting system model on high abstraction level

![Diagram](image)

#### Figure 3.22: Evaluation of energy levels estimated by the system model compared to deployment results – configured with (a) a 10 F DLC and (b) a 22 F DLC

![Graphs](image)

3.3.2 Simulating Available Energy Levels

Before the model was used for simulations, an evaluation has been conducted against a real-world deployment of the same architecture, as was described in Section 3.2.1. The results of the measured energy level and estimated energy level from the model for one week of operation are shown in Figure 3.22. In the deployment two different DLC capacities have been implemented, leading to results from a 10 F version, shown in Figure 3.22a and results from a 22 F version in Figure 3.22b. While for both cases the evaluation shows that the simulations and measurements generally conform well, smaller error sources can be detected. This is visible especially in Figure 3.22a, where some time-shifts can be observed. It was found that, this is the result of slightly varying hysteresis bounds. While the model can be adjusted to average
3 Solar Energy Harvesting

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Optimization Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solar panel size</td>
<td>Minimization of cost and size</td>
</tr>
<tr>
<td>DLC capacity</td>
<td>Minimization of cost and size</td>
</tr>
<tr>
<td>Load consumption</td>
<td>Maximization of duty cycle (sampling rate)</td>
</tr>
<tr>
<td>Solar irradiance</td>
<td>Minimization of environmental dependency</td>
</tr>
<tr>
<td>Load availability</td>
<td>Maximization of system uptime</td>
</tr>
<tr>
<td>Hysteresis band</td>
<td>Minimization of component stress</td>
</tr>
</tbody>
</table>

Table 3.6: Dimensioning parameters of the example solar energy harvesting architecture with their respective optimization goals

Hysteresis bound levels, variations in the trigger level related to environmental conditions are very difficult to predict. In Figure 3.22b a difference at the sawtooth tip is obvious. However, this does not result from a modeling error of the energy harvesting system, but from a measurement error, due to a limited measurement range of up to 2.5 V. The evaluation confirmed that the model is capable of predicting the SOC in an accurate manner.

After the model was evaluated, it has been used in order to demonstrate its purpose to support the optimization process of parameter dimensioning. A number of parameters of interest to be optimized for the example architecture are listed in Table 3.6. While typically, not all the optimization goals can be met simultaneously, the simulations enable the discovery of trade-offs, which meet the given application requirements. An exemplar simulation outcome for the presented system architecture is shown in Figure 3.23. The depicted results have been obtained using two assumptions:

1. Solar panel size is perfectly scalable, meaning that an increase of solar panel size leads to an increase by the same factor in irradiance-to-current conversion at the same voltage.

2. At times of insufficient energy (i.e., the voltage level of the DLC is lower than the minimal boost converter input) the load will consume no energy.

Furthermore, all the simulations are based on the irradiance data of December 2009 in Sundsvall, Sweden, which is the darkest month of the year.
3.3 Solar Energy Harvesting Dimensioning

Figure 3.23: Simulation sweeps of different dimensioning parameters with their influence on the maximum average current consumption – (a) DLC sweep at 100 % availability, 350 mV hysteresis band; (b) Availability sweep at 10 F DLC capacity, 350 mV hysteresis band; (c) Hysteresis sweep at 100 % availability, 10 F DLC capacity
3 Solar Energy Harvesting

The results show the maximum allowable load current (i.e., the average current consumption) under varying system configurations. All sub-figures include variations of the solar panel size as a typical parameter which is to be changed. Additionally, Figure 3.23a shows the impact of a change in DLC capacity, Figure 3.23b the change of availability requirements and Figure 3.23c the variation of the hysteresis band setting of the over-voltage protection circuit. The information from these simulations can be used in order to select appropriate system parameters for a given application scenario.

In particular, these results can support decisions for system designers. Taking Figure 3.23b as an example, it can be extracted that it is not reasonable to increase the solar panel size by more than a factor of three, when lifetime is the constraint with an availability factor of 95%. At the same time, trade-off decisions can be made, such as determining, that a 45 μA average load current constraint can be reached with a 96% availability and double solar panel size, or with a 99% availability, combined with a triple-sized solar panel.

3.4 Solar Energy Harvesting Testbed

While the feasibility of using solar energy harvesting architectures as the power supply of wireless sensor nodes has been demonstrated, and energy level simulations have been presented as a suitable method of dimensioning these architectures, extensive experiments are required in order to evaluate architectures and create accurate system models. These experiments are time and cost consuming, due to the required network deployment, which has been accepted by the research community as an extremely difficult process [13]. The evaluations to be performed, such as those presented in the previous sections, are therefore limited to a subset of test cases, which results in a trade-off between time and cost expense and the desired evaluations to be carried out. In relation to solar energy harvesting architectures, test cases of interest include any variation that can be performed to the system as depicted in Figure 3.2. These include major variations, such as the comparison of different charging architectures and storage technologies, but at the same time minor changes, such as storage capacities, solar panel sizes and load profiles, may be of interest. In particular, minor variations will suffer from the considerable overhead required for their consideration in real-world experiments.
3.4 Solar Energy Harvesting Testbed

<table>
<thead>
<tr>
<th></th>
<th>Conventional Testbed</th>
<th>Solar Testbed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td>static</td>
<td>dynamic</td>
</tr>
<tr>
<td>Software</td>
<td>dynamic</td>
<td>static (^2)</td>
</tr>
<tr>
<td>Environment</td>
<td>controlled</td>
<td>uncontrolled</td>
</tr>
<tr>
<td>Monitoring</td>
<td>internal/external</td>
<td>external</td>
</tr>
<tr>
<td>Interface</td>
<td>wired</td>
<td>wireless</td>
</tr>
</tbody>
</table>

Table 3.7: Comparison of the typical properties in conventional WSN testbeds with those required for a solar energy harvesting testbed

Previously, other areas of research in Wireless Sensor Networks have faced similar situations. Protocol and algorithm development, for example, is often conducted based on analytics or simulations. Their implementations, however, require to be evaluated on physical systems in order to verify their performance and to allow for comparisons to be made with existing solutions. As these solutions can require large network sizes and specific evaluation environments, overhead also becomes an issue in these cases. A solution that gains popularity in these scenarios involves WSN testbeds, such as those presented in [63–66]. While the implementation of such testbeds can vary, the underlying principle remains the same. A testbed typically contains a pre-deployed number of nodes in an controlled environment. The nodes can be programmed with the software to be tested remotely, but run a separate firmware that monitors evaluation parameters, which are typically transmitted via a back-channel (e.g., USB) to a central server.

A similar approach can be utilized in order to address the overhead problem in solar energy harvesting evaluations and, thus, allow for a wider set of experiments to be conducted. In comparison to conventional testbeds, however, a solar energy harvesting testbed poses a different set of properties. The properties of the conventional and the proposed solar energy harvesting testbed are contrasted in Table 3.7. While conventional testbeds are typically used in order to test and verify software components, in a solar energy harvesting testbed the main point of interest is in relation to changes to the hardware, which requires the hardware of the testbed nodes to be configurable. Furthermore, the environment has a major impact on the harvesting performance. Because of the complexity regarding the implementation of an
artificial environment to be controlled for all scenarios of interest, a more realistic approach is for the system to be deployed in an outdoor environment, which is uncontrolled, but provides a wide range of real-world conditions. The monitoring of environmental parameters, thus, becomes of high importance in order to make sense out of the system data that has been collected. In addition, the outdoor deployment makes wired back-channeling infeasible, but favors the use of a wireless collection and control method.

The resulting system concept is depicted in Figure 3.24. A network of sensor nodes with configurable solar energy harvesting is deployed in an environment, which exposes the system to varying environmental conditions. The nodes are remotely configured in order to define the solar energy harvesting method and system conditions, which are to be evaluated. Moreover, the wireless link is utilized in order to obtain the parameters of the system behavior and its surrounding. As a result of this concept, the amount of architectures and configurations to be tested is much larger than the number of system components that have to be deployed. Hence, the testbed reduces overhead in the experimental setup as compared to that for a conventional evaluation network. The distributed architecture of the testbed, furthermore, allows for direct comparisons of architectures and locations to be performed.
3.4 Solar Energy Harvesting Testbed

Figure 3.25: Layered architecture of the solar energy harvesting testbed. Modules are distinct between supporting node-level or management-level functionality.

3.4.1 Architecture and Implementation

The architecture of the solar energy harvesting testbed can be separated into two levels, namely the network of nodes that contain the harvesting methods to be tested and the management functions which are required in order to operate the testbed in a remote fashion. Each of these levels, in turn, is composed of multiple modules. The function and implementation of these modules will be presented in the remainder of this section. An overview of the system architecture with its levels and modules is provided in Figure 3.25.

On the node level, a clear distinction is made between the configurable harvesting circuit (i.e., the device/circuit under test) and the sensor node, which supervises the harvester and collects evaluation parameters (i.e., the back-channel). Figure 3.26 illustrates the exemplar architecture of a configurable harvesting circuit. Its principle is similar to that for the configurable harvesting platform, which has been presented in Section 3.2.2, but the configurations, in this case, can be carried out by electrical switches and transistors as opposed to the mechanical configurations, which have been used previously. This allows for configurations to be performed in a remote manner, as the required electrical signals can be applied by the MCU of the sensor node and the respective commands can be distributed wirelessly. While the exact architecture of the harvesting circuit depends on the evaluations to be performed, the configuration typically takes place by device selection, device configuration and charge path adaptation. A typical example for device selection would be the choice of an energy storage device of a particular type.
3 Solar Energy Harvesting

Figure 3.26: Architecture of the configurable energy harvesting module of the solar energy harvesting testbed nodes. Configurations are performed by electrical signals.

and capacity, device configuration might be the adjustment of a setting at a harvesting IC, whereas the charge path adaptation defines which of the provided components are involved in the evaluation circuit and in what order the components are placed in order to construct a charge path. In the exemplar implementation (Figure 3.26), configurations can be performed to all modules within the solar energy harvesting circuit (see Figure 3.2 for general reference). The system enables there to be a variety of different solar panel and energy storage choices in order to test the influence of solar panel size/rating, as well as energy storage type and storage capacity. With regards to the input regulation, a commercial harvesting IC has been chosen, which can be configured to different output voltages and allows for MPPT to be enabled or disabled. Although different output regulation modules are integrated in the system, the choice is not a user configuration, but is dependent on the selection of the energy buffer type. Finally, the architecture allows for different load conditions to be tested. This has been accomplished by means of a resistive load emulation, which allows for the load profile to be emulated by creating defined power peaks for configurable periods of time. As was mentioned previously, this implementation does not attempt to allow for every evaluation that is of interest in a solar energy harvesting system to be performed, but rather shows the principle of how a remotely configurable architecture might be implemented.

The control and monitoring of the configurable harvesting circuit is performed by a typical sensor node platform (i.e., in this implementation a
3.4 Solar Energy Harvesting Testbed

SENTIO-em platform). Additional interfaces are, however, integrated in the harvesting circuitry, which enables the node to perform the sampling of evaluation parameters, such as current flows and voltage levels. These values can be utilized in order to correlate the influence of environmental conditions with the system behavior of the harvester under test, as well as to estimate the operation state of the circuit (e.g., the charge level of the energy buffer) at any given time. In addition to a number of system internal parameters, the sensor node is responsible for the measurement of reference values, such as for the classification of the environmental conditions during the evaluation period. In the exemplar implementation, reference values are obtained in the form of temperature, humidity and solar irradiance.

Using a conventional sensor node for the control and monitoring, it additionally provides an existing interface for communication and networking – including the respective protocols – to the testbed nodes. This interface is used, on the one hand, to distribute the configurations to the individual nodes and, on the other hand, to collect the sampled parameters in a central location, which is the basis for the management level.

Because available energy is one of the key parameters for the evaluation of an energy harvesting architecture, a clear separation is required between the energy source to be tested and the energy source that supplies the monitoring and control functionalities. Therefore, a two-domain architecture has been chosen for the testbed nodes. While the system to be tested is powered from its own resources, all modules, which are only for the observation and configuration of the system, are supplied from an additional source. These modules include all switches for the reconfiguration of the harvester, the reference sensors, as well as the sensor node supply (i.e., all intelligence and communication). The backbone power supply, in this system, should be dimensioned to guarantee continuous operation. In the implementation, this has been accomplished with an off-the-shelf combination of solar panel, charge regulator and high-capacity battery, which can supply a testbed node for approximately two months without solar energy income.

Management functions include the data storage, which is typically performed by a database system, and the user interface. In the implemented system, the database acts as the central buffer. The gateway uses the database to store all parameters that have been collected from the testbed and forwards configurations that are stored in the database to the testbed nodes. Similarly,
the user can access the database via an user interface in order to create or edit node configurations, assign configurations to the individual testbed nodes, as well as to access data that has been received from the nodes within the testbed. In a similar manner, other tools could access the database, for example, for the visualization of data on a website (i.e., without the requirement for a specific tool). This, however, has not been integrated to the proof-of-concept implementation.

In order to link the nodes in the network with the database, and hence the testbed’s management level, the network contains a gateway. As in any other Wireless Sensor Network, the gateway links the local network of sensor nodes to the outside world, which allows for remote access to be performed. Although this is not required, in the implementation of our testbed network the gateway also hosts the database. The gateway, thus, handles all the data flow between the nodes and the database and accesses the correct tables for data to be stored or configurations to be read. For this to be carried out in a reliable manner, a periodic sample-and-send protocol has been implemented. In order to reduce the risk of packet collisions, the testbed nodes follow a coarse synchronization schedule for their data to be transmitted. Figure 3.27 illustrates the typical communication during normal data collection (Fig. 3.27a), and node reconfiguration (Fig. 3.27b). During normal operation, the
3.4 Solar Energy Harvesting Testbed

Figure 3.28: Implementation of the configurable solar energy harvesting testbed nodes. Two nodes are mounted at the same location and share a common back-channel power supply.

database replies with an acknowledgment to incoming data from the master node, which leads to the periodic distribution of a synchronization broadcast in the network. Based on this synchronization and their respective node IDs, the nodes will consecutively transmit their data to the database. In the case of a node configuration, on the other hand, the database replies with a configuration frame instead of a bare acknowledgment. This frame will then be distributed instead of the synchronization packet, which allows the nodes to update their current configurations. Each node will acknowledge its new configuration within its following data transmission and continue normal operation until a new configuration broadcast is received.

3.4.2 Setup and Results

In order to evaluate the system concept, the previously described architecture has been implemented and deployed in a test network. Figure 3.28 shows a picture of the final system implementation of the testbed nodes. The evaluation of the implementation has been performed in two stages. Before the network deployment, the node and system functionality has been analyzed...
in a laboratory environment. For this, laboratory instruments have been utilized and the main goal of this stage was to verify the system implementation and test the functionality and performance of the nodes. Afterwards, a small testbed network has been deployed in its intended environment. The network consists of six testbed nodes, which are deployed pairwise on the university campus in order to allow architectures to be compared in a direct manner. The nodes communicate with the gateway, which is also deployed on campus, but has access to an Ethernet connection. As opposed to the laboratory evaluation, the goal of the deployment is the test of the overall system implementation in its target environment, and the demonstration of the functionality that can be gained from a reconfigurable solar energy harvesting testbed.

Initial verification measurements included the bare analysis of the hardware operation, such as the test of switches which are required for the reconfiguration of the harvesting unit. After these measurements demonstrated the hardware to be operational, functional tests have been conducted. One of these tests is the usage of the resistor bank as a configurable load emulation. Figure 3.29 depicts the current consumption profile of a typical sample-and-send application for a sensor node. In order to emulate this type of profile with a resistor bank, the individual consumption levels and time periods have to be configurable. In the figure, four separate consumption levels can be seen, which correspond to the low-power mode (2 μA), active processing (8 mA), as well as transmit and reception (30 mA and 25 mA respectively).

Figure 3.29: Energy profile of a sensor node with sampling and communication tasks
3.4 Solar Energy Harvesting Testbed

Figure 3.30: Comparison of ideal and measured load emulation for the configurable load profiles on testbed nodes

Figure 3.30 shows that these values can be emulated by a resistor network in a simple manner and with relatively high accuracy. For the implementation resistors with a tolerance of 1% have been used, which explains the increasing error with increasing current emulation. As this method of a load emulation only requires a combination of resistor and transistor for each current level to be emulated, increasing the load states is simple and inexpensive. Furthermore, time periods can be easily adjusted by utilizing standard timer modules, which allows for high resolution current profiles to be created. The time periods, in our implementation, can be defined with a resolution of 62.5 μs.

Additionally, the consumption profile of the monitoring and control function is of interest, as the system should operate reliably over a long period of time without having access to a grid-based power supply. The node's consumption profile, thus, defines how the back-channel power supply should be dimensioned. Figure 3.31 shows the profile of an exemplar scenario. The presented scenario is that of the master node, which can be deduced from the two communication periods (i.e., one with the database and one with the other testbed nodes). Communication, in this implementation, occurs in a five minute interval, which is a trade-off between data resolution and power consumption of the back-channel node. The load emulation period, however, does not have to follow the same interval, which is illustrated in this scenario. Instead, the controller awakes four times per period and performs its control and monitoring task. The number of these sampling activities is not fixed
and can be adjusted on a per node basis within the harvester configuration. In addition, it can be defined as to how many of these samples the load peaks should be applied to the architecture under test.

Finally, Figure 3.32 depicts exemplar output results of the deployed energy harvesting testbed. The results demonstrate how the testbed can support the analysis of location- and configuration-dependent performance of the harvesting architectures. Figure 3.32a, for example, shows a subset of the comparison results, which have been obtained from different locations. During the measurement, the same configurations have been applied to two separate nodes. One of these nodes has a southward orientation, while the other node is directed towards the north. The obtained measurements of the state of charge illustrate this difference by means of the length of the daily charge periods. Figures 3.32b and 3.32c, on the other hand, contain data for different testbed configurations at the same node location. The difference of the architectural behavior for varying DLC capacities is depicted in Figure 3.32b, whereas Figure 3.32c shows the influence of the load condition on the state of charge in the same harvesting architecture.
Figure 3.32: Exemplar output results of the solar energy harvesting testbed. The results show influence of (a) different node orientation, (b) different storage capacity, and (c) different load profile.
Usability has been an active field of research, particularly, in the field of website and user interface design. Its main target is to reach a desired goal in an easier manner, such as finding a certain piece of information on a web page. In order to quantify usability, five attributes have been defined [67], which are Learnability, Efficiency, Memorability, Errors and Satisfaction.

Learnability describes how easy it is for a new user to accomplish a task. Once the tool has been learned, efficiency defines how quickly a desired task can be performed. Memorability indicates how easy it is for a user to remember how to use the design, when it has been used previously. Errors include how likely it is for the user to make a mistake, how severe these mistakes are, and how simple it is to solve them. Finally, satisfaction describes how satisfied a user is with the usage of the tool and the method to reach the intended goal.

As usability is a non-functional requirement of a design, it typically cannot be measured directly. A typical evaluation method is the involvement of test users, who represent the target user group and, thus, can provide qualitative feedback. The integration of usability aspects should only have a minor impact on the performance of the system. Trade-offs, however, are often required to be made. Nonetheless, the final system will only be useful if a balance is found. This usually means that a system, which can perform a desired task, but which a user does not know how to use it, is as useless as a system that is easy to be used, but does not match the user’s performance requirements.

### 4.1 Usability in Wireless Sensor Networks

In Wireless Sensor Networks, usability can be addressed with regards to two distinct user groups, namely the system developer and the end user.
Particularly, in the environmental monitoring domain, but also in many other application areas of WSNs, these user groups pose vastly different requirements in relation to the interface to be provided and, thus, the usability requirements to be fulfilled. While for the development, parameters such as the ease of programmability and debugging methods are of major interest, for the end users the concealment of complexity becomes of particular importance. These factors can be addressed both on hardware and software level in the system design.

Existing work towards the improvement of usability in Wireless Sensor Networks has mainly focused on the software development aspect. This has led to solutions that attempt to simplify application development, ranging from node languages [68–70] and operating systems [71,72] to abstract language constructs that address the network as a whole [73–75]. Furthermore, the majority of these solutions are addressed towards system experts, that means users who have a considerable understanding of the underlying system to be used. While this requirement is typically met in system research applications, in which systems are operated by their developers in order to conduct a proof of concept, in the long run WSNs are required to be operated by their intended user group, who typically do not possess the currently required system skills.

As a result, usability must be addressed for both user groups separately. The remainder of this chapter will do so, and present usability considerations and resulting design decisions for both groups, respectively. Furthermore, it will be demonstrated that usability in sensor networks must be addressed for both the hardware and software.

4.2 Usability for Developers

As has been indicated previously, the main goal for usability with respect to development is the ease of programmability and debugging. This means, a developer should have interfaces in order to implement new applications or algorithms in a quick and effective manner. Although this interface should limit the chance of making a mistake during the implementation, additionally, debug interfaces should be available, which enable the user to efficiently test his/her implementation and locate mistakes that have been made.
While Wireless Sensor Networks can, to some extent, profit from previous efforts in conventional computer systems – particularly with regards to programming methods – the resource limitation of the sensor nodes that are typically used, restricts the methods that can be implemented in an effective manner. Furthermore, WSNs face challenges that complicate their operation, such as the deployment in remote and harsh environments. These conditions make proper debug interfaces a desired system attribute. Such interfaces, however, require time and energy and, thus, reduce the performance or the lifetime of the implemented system.

In the following, two approaches for the improvement of usability for system development will be presented. The major goal for both approaches has been the maintenance of high performance, despite an increase in usability. Section 4.2.1 contains the presentation of a systematic programming method based on Hierarchical Finite State Machines (HFSMs). The target group for this programming method are system experts, who typically possess the knowledge of describing systems and their behavior in the form of FSMs. Thus, a translation from application description and its implementation becomes easier, which reduces the risk of erroneous implementations. In Section 4.2.2, usability considerations for sensor node hardware platforms will be presented. These considerations are based on the typical constraints in the environmental monitoring domain. The implementation of such considerations is demonstrated on the SENTIO-em hardware platform design (see also Section 2.4).

### 4.2.1 Programming with Hierarchical Finite State Machines

Microcontrollers are the typical implementation choice of the computational module in wireless sensor nodes. Therefore, the development of the source code, which defines the node's behavior, typically occurs based on conventional embedded system development methods. This process is complicated by the number of tasks a sensor node should perform – only a subset of which are sensor sampling, processing, communication and network maintenance – and the desire of operating the node at low power consumption, which results in the usage of controllers with processing and memory limitations. As a consequence, programming methods for WSNs have been a topic of interest since the very beginning of sensor network research. This, in turn,
Figure 4.1: Description of a typical sample-and-send application in form of a FSM state graph. State transitions can occur based on task completion (TC) or events, such as a timer interrupt.

has led to the utilization of a broad variety of methods, which range from simple embedded C code for high performance optimization, via dedicated operating systems for resource constraint computing systems, to language models with a high level of abstraction for the simplified implementation of complex tasks (see [9] for a summary of programming methods). Many of these methods, however, accept reduced performance (e.g., due to additional latencies) as a trade-off for usability improvements.

As an alternative, Finite State Machine models can be used to improve the usability of software development in Wireless Sensor Networks. FSMs are a common method in order to describe systems, which are composed of multiple states and defined transitions between these states, such as control systems or digital circuits. As sensor nodes perform multiple tasks, their operation can easily be visualized by a state graph, in which states represent a distinction of tasks. Figure 4.1 illustrates such a state graph for a simplified sample-and-send application. After initialization, the system enters a periodic behavior of sensor sampling, data transmission and energy conservation, which are sequentially executed. While most transitions in this scenario are based in successful task execution, transitions can be triggered by events, such as a timer interrupt. This simple state graph could easily be extended, if the application requires additional events and states (e.g., for the handling of error signals). As the underlying construct of a state machine follows a simple mechanism, its implementation can be carried out with extremely
Programming methods can be evaluated based on a set of qualitative parameters, including *readability, modularity, reliability* and *efficiency*. The application of Finite State Machines as a programming method for sensor networks can improve the software development with regards to several of these aspects. Because of the separation of the overall application architecture and the individual state implementation, readability is improved. Moreover, the number of states and the code length and structure within each state can also increase readability. The same architectural properties make FSMs highly modular. Existing states, as well as complete state machines, can easily be reused in multiple applications. As FSMs are a highly systematic manner of describing a system, this method also has a positive influence on the reliability of the implemented code. Implementing a defined state transition for every possible event, for example, allows for the application to end up in invalid system states which are to be avoided. Finally, these improvements can be achieved with low overhead, which results in an efficient code to be generated from this programming method.

While FSMs allow for applications to be split into smaller modules (i.e., states) and, thus, improve a number of factors in software development, a frequently mentioned disadvantage is the readability of a complex state graph. As the number of states and transitions in a system increases, it can be difficult to maintain an overview of the complete system structure. This typically leads to a trade-off between the number of states and transitions in the system, and the code complexity of each state. Hierarchical Finite State Machines can provide a solution in these cases by allowing multiple state graphs to be composed, which are linked with each other. In Figure 4.1, for example, the *transmit-state* could contain a separate FSM, which handles a more complex communication protocol. In this manner, a simple state graph is maintained on the top level, but complex states can be broken down into submodules for improved readability and maintainability.

In order for the programming method to be tested, an HFSM framework has been implemented for the SENTIO-em sensor node platform. Portability of the implementation is maintained due to a layered architecture as depicted in Figure 4.2. The architecture is based on an Hardware Abstraction Layer (HAL), which allows for the code to be easily adjusted to other hardware platforms. This layer contains the abstraction of the individual
4 Usability

Figure 4.2: Layered architecture of the HFSM framework implementation for wireless sensor networks. Functions are separated by hardware abstraction, state machine support, and the application.

driver modules and their functionalities and feeds into an unified driver interface, which is part of the support layer. The driver interface allows for individual modules to be selected for their usage in any given application code. In addition, the support layer includes the actual FSM handling system, which evaluates the state graph and implements the defined state executions and transitions. In order to keep the overhead of the handling system at a minimum, it uses an array of pointers to functions, which contains the addresses of the individual state handlers, that means the functions in which the state contents are defined. The transitions between states in the application are, thus, only an index manipulation of an array, which results in fast execution times and low processing overhead. For multiple state machines being able to share event resources, which are typically the interrupt sources of the underlying hardware architecture, the Interrupt Service Routines (ISRs) have also to be shared. As common MCUs typically provide only a single ISR for a given interrupt source, the sharing of this resource has to be manually integrated. This has been achieved by a similar architecture to that of the state transitioning. In the case of a HFSM utilization, each state machine in the application is assigned a unique application ID. The ISRs are called from an array of pointers to functions, wherein each entry corresponds to one of the application IDs. In case an interrupt is triggered, the handling system
4.2 Usability for Developers

Figure 4.3: Program flow of the executions of state transitions in the HFSM framework implementation.

will forward the event by pointing to the ISR defined in the array field of the respective application ID. While the support layer and the HAL are framework internal, which means that they are typically not edited by the application developer, the application layer provides an interface to the developer that allows for the applications to be adjusted. This layer contains the code that defines the individual states, events and their relationships (i.e., the state graph). In order to maintain integrity, the complete framework is developed in C++, which allows for invalid function calls to be avoided. Moreover, access between different function layers can be easily established by using class inheritance.

Figure 4.3 provides an overview of the program flow in a typical HFSM application. As low energy consumption is a critical component in sensor network applications, particularly in the environmental monitoring domain, the usage of low power modes has been integrated directly in the FSM framework. The developer, thus, has the choice of executing the next state in the state graph directly, or entering an energy conserving low power mode while waiting for the event that will trigger the execution of the next state.

While the qualitative improvements of the implementation are difficult to measure, a quantitative evaluation of code efficiency has been performed.
based on the introduced overhead of latency and memory usage. Tables 4.1 and 4.2 list measurement results that have been obtained from an implementation on the SENTIO-em hardware platform. In order to evaluate memory overhead, two applications have been implemented using the previously described framework. The *Blink* application represents a minimal code implementation, which is based on a single FSM with three states. In this scenario, state transitions are triggered by a timer that periodically generates an interrupt. As opposed to this, the *Sample* application is an implementation of a typical data gathering application, which periodically samples sensors and transmits the obtained data using radio communication. This application uses multiple state machines in a hierarchical structure. In both cases, the measurement results show that the majority of the code size is occupied by driver implementations, which indicates the low memory overhead of the HFSTM framework. The majority of additional code, which is generated by the framework lies within the support layer, which has only a minor contribution of a few percent to the overall code size. Moreover, the timing measurements in Table 4.2 indicate the high code efficiency of the HFSTM implementation. The statistical results of average transition times and their standard deviations are based on a sampling set of ten measurements on the SENTIO-em platform, which is operating at a clock frequency of 32 MHz. The results demonstrate fast transition times for both, state transitions and the reaction to interrupt events. The low standard deviation of the individual measurements, which is of the order of the measurement accuracy of the utilized logic analyzer, furthermore, shows that the obtained values are highly deterministic.

Overall, the high efficiency of the HFSTM implementation and the familiarity,
4.2 Usability for Developers

<table>
<thead>
<tr>
<th>Transition</th>
<th>Condition</th>
<th>AVG [μs]</th>
<th>STD [μs]</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>direct</td>
<td>1.56</td>
<td>0.004</td>
</tr>
<tr>
<td></td>
<td>sleep (clock on)</td>
<td>3.8</td>
<td>0.007</td>
</tr>
<tr>
<td></td>
<td>sleep (clock off)</td>
<td>6.77</td>
<td>0.007</td>
</tr>
<tr>
<td>Interrupt</td>
<td>Enter ISR</td>
<td>0.77</td>
<td>0.007</td>
</tr>
<tr>
<td></td>
<td>Enter Handler</td>
<td>1.74</td>
<td>0.007</td>
</tr>
</tbody>
</table>

Table 4.2: Measurement results of the latencies for state-to-state transitions and interrupt handling

which has been generated by utilizing a programming method for developers, who are comfortable with system descriptions in the form of Finite State Machines, has the ability to improve the usability of software development for technically experienced users.

4.2.2 Usability Considerations for Sensor Nodes

Although software considerations can support the implementation of application development, such as the programming method presented in the previous section, an evaluation of the implemented software is required to verify its correct operation. Particularly in the development process, this can require evaluation measurements to be conducted and the localization of implementation mistakes. While software features, such as the utilization of log files and additional debug outputs, can support this process, these functionalities typically are associated with a cost in the form of software overhead and additional resource usage. This, in turn, can change the actual application behavior or lead to an undesired overhead once the verification process is completed. Hardware considerations on the sensor node platform can support the development process by providing features that simplify such evaluation measurements and, thus, improve the platform usability during the development period. At the same time, however, hardware overhead should be maintained at a minimum in order for the platform to agree with the desired cost and size constraints, which is of particular importance in applications in which the number of sensor nodes can be large (e.g., in the environmental monitoring domain).
Initially, sensor node platforms have been envisioned to be of minimal size in order to cover a large terrain at high density [76]. While the development of such miniaturized platforms continues [77], the majority of experimental research in WSNs is performed on sensor nodes of larger size, typically referred to as motes, which simplifies the development process due to the possibility to integrate additional resources and provide easier access in relation to those signals to be measured. Different concepts have been tested in order to combine the development support with the desired system properties, commonly agreed upon in the research community. The Telos mote [25], for example, can be connected to the development system (i.e., typically a personal computer) in a simple manner, as it provides an interface similar to that of a USB pen drive. While this has a positive impact on the platform's usability during laboratory operation, this feature does not support the deployment and, thus, experiments outside of the laboratory environment. On the contrary, this form-factor might actually hinder the physical enclosure and, thus, complicate deployment evaluations. Similar issues have been encountered on the waspmote platform, which has been used in the SAQnet case study (see Section 1.3). While the concept of sensor boards allows for the platform to be prepared for specific application scenarios and sensors can easily be plugged into these boards in the laboratory, preparing the nodes for outdoor deployment becomes difficult as sensor connections have to be extended. As a result, deployment becomes a tedious process, which has been acknowledged within the community [13].

In the design and implementation of the SENTIO-em sensor node platform, usability considerations with respect to both development and deployment have been taken into account. The deployment considerations are of particular importance for the development of environmental monitoring wireless sensor networks, because evaluation tests in the targeted application environment are common. A key factor in the design of SENTIO-em are the division of laboratory and deployment functionalities. While additional resources are desirable during the development phase of a system, these resources increase the size and cost of the node unnecessarily if they do not perform any contribution to the system operation after the deployment. Particularly in the environmental monitoring domain, in which outdoor operation might be performed for prolonged periods of time, these unused resources are undesirable. A typical example of such resources is the USB
Figure 4.4: Picture of the SENTIO-em platform implementation. The core platform is mounted on its dock solution for laboratory development.

interface. This interface is commonplace on many modern sensor node designs, as it allows for the node to be programmed via a serial bootloader, and for debug messages to be sent to a computer terminal. During deployment, however, a USB interface does not provide any useful functionality and, thus, only results in wasted circuit space and additional node cost. Similar arguments can be brought forward for other modules, especially in relation to user interfaces, which only provide useful functionality during development. On the SENTIO-em platform, these functionalities have been separated from those which are required for the actual node tasks. As a result, a laboratory docking solution has been created, which allows for extra development modules to be provided during laboratory work, whereas the actual sensor node only includes resources for its task execution in the field. Figure 4.4 depicts this separation in the SENTIO-em implementation.

While extension is a typically utilized concept in the design of wireless sensor nodes, on SENTIO-em, extension interfaces have been grouped according to the modules that are attached. Sensors are a typical module to be separated from the main sensor node, as the selection of sensors is highly application specific. Moreover, extended power supply units, such as for the utilization of ambient energy sources, require a power interface with increased complexity to be provided. As a result, signals and communication interfaces are available in order to control and monitor the power supply unit,
as opposed to the bare reception of energy. An additional debug interface is used, mainly, in order to enable the previously mentioned separation of development and deployment resources. For the implementation of these interfaces, standard 2 mm connectors have been used, which allow for board-to-board and wire-to-board connections to be established in a mechanically robust manner.

Although wireless communication is an essential part of a sensor node, in a later revision of SENTIO-em, this module has been removed, but is instead connected in a plug-in manner. This separation choice has multiple reasons, which affect the usability of the platform. Firstly, individual applications can have varying requirements on communication properties, such as carrier frequency and data rates. Moreover, a vast amount of communication protocols are available, which offer their respective advantages in specific application areas. Finally, the evaluation of a new radio transceiver is simplified, as the development is restricted to the communication module, as opposed to the implementation or adjustment of a complete sensor node. For the mechanical connection, the XBee form-factor has been chosen. The XBee series of radio transceivers are commercially available and have gained particular interest in hobby electronics, due to their simple usage. Selecting its form-factor for SENTIO-em allows for the rapid prototyping with a set of wireless communication standards and transceiver properties to be performed. Radio transceiver options, which have been used on the SENTIO-em platform, include low-frequency transceivers, IEEE 802.15.4 and IEEE 802.11 modules.

Considerations on the form-factor of a sensor node platform have typically been restricted to small physical size. In the environmental monitoring domain, however, the node size is, in the majority of cases, not a primary restriction. However, it is the case that easy deployability is typically of higher importance. As was previously mentioned, the deployment process in environmental monitoring applications can be rather tedious. A reason, amongst others, is the necessity of preparing the sensor node for outdoor deployment, which includes the enclosing of the electronic system. An option in order to improve this process is the design of a customized enclosure. This, however, is a costly solution, which is typically not feasible within the scope of most research projects. For SENTIO-em we have chosen the inverse procedure and designed the node with a customized form-factor for a specific physical enclosure. While this requires minor adjustments to the Printed
4.3 Usability for End Users

Circuit Board (PCB) size, this solution comes at virtually no cost. SENTIO-em has been designed with a form-factor in compliance with a Hammond 1591T enclosure\(^1\), which allows screwless mounting of PCBs in a number of molded card guides.

A great number of experimental projects in WSNs have documented the contradiction of debug possibilities after deployment and the resource limitation of typical sensor nodes. One reason is the uncertainty of the necessity for debugging at the time of system deployment. In the case for which debugging is required, but no method is implemented, errors are likely to remain undetected, which typically results in the return of the system to the laboratory. On the other hand, if debugging methods have been provided, but are not required, the system operation might be restricted, because resources are wasted for a function that is unnecessary. A desirable alternative is the option to enable debug options, if required, while disabling them otherwise. This, however, commonly requires opening up the enclosure of a node, which is highly undesirable in outdoor deployments, particularly after nodes have been weatherproofed. In order to prepare SENTIO-em for this scenario, two magnetic switches have been integrated, which allow interactions to be performed with an enclosed node. The switches have been placed with the largest possible distance between each other in order to avoid accidental triggering of the wrong switch. While one of the switches lies in the node's power line and, thus, allows for the node to be remotely disabled or reset, the second switch can trigger a user-defined interrupt, which might be used to enable or disable certain node functionalities without the requirement to open the node's enclosure.

4.3 Usability for End Users

As opposed to the system developer, who has a profound understanding of the underlying system architecture of a WSN, the typical end user of such a system has experience with the domain the system should be used in, rather than the system itself. Moreover, in the environmental monitoring domain, it is highly uncommon for end users to possess skills related to the programming of computer systems, let alone embedded systems, which are the primary

\(^1\)http://www.hammondmfg.com/dwg2a.htm
method of interacting with sensor nodes at the present time. Nonetheless, the requirement of hiring a system expert in order to operate a WSN as a measurement system is not feasible and limits the real-world applicability of such systems.

While there has been a great amount of research on the simplification of programming methods with respect to WSNs (see Section 4.2.1), the interaction methods that do not use any form of programming language are strictly limited. Solutions, that come closest to this requirement, are referenced in [73, 78]. While [78] presents a method of using spreadsheets in order to configure WSNs, [73] uses an interface similar to database abstractions. In both cases, however, an evaluation of the usability of the respective method with regards to the targeted end user is missing.

4.3.1 Concealing Node Programming Complexity

Concealing the technological complexity of a system is not limited to WSNs. Personal computers, for example, have very similar issues when being operated by an inexperienced user. Much effort, however, has been spent in making relatively complex tasks appear easy to the end user, by automating required tasks and by performing them in the background. Such tasks can be the installation of new software or the operation of hardware peripherals in a plug-and-play fashion. Similar methods can be utilized in order to make WSNs more accessible to non-technical users.

While different concepts can be utilized, in this study, a wizard-based configuration of sensor nodes has been selected. This choice has multiple reasons, including the user familiarity of the concept from software installations, the guided flow from the start to the end of the task to be executed, the step-wise execution of the task, as well as the possibility to maintain a similar process as used when collaborating with a system expert. Figure 4.5 depicts the conventional system design process, in which a domain scientist provides system specifications to a system expert, who, in turn, will implement the system to conform to the application requirements. Figure 4.6 illustrates how the conventional process can be adjusted in order to remove the system expert by utilizing a wizard-based specification method. Assuming that the domain expert has access to a sensor node, the specifications have to be translated into a firmware, which can be uploaded to the node,
4.3 Usability for End Users

Figure 4.5: Conventional process of sensor system implementations involving system, as well as domain experts

Figure 4.6: Adapted sensor system implementation process. The requirement for a system expert has been replaced by a specification tool and a configurable system.

in order to adjust the system’s behavior to the respective application. This can typically be accomplished in two ways. Firstly, the specification can be used to generate a sourcecode, which is compiled and loaded to the sensor node. Alternatively, a fixed firmware is already located on the sensor node, which allows for configurations to be made in order to adapt its behavior. As automatic code generation is prone to errors, and non-technical users are unlikely to be able to recover from these errors, the latter alternative has been selected for this study. The system implementation, thus consists of a GUI, which collects the users’ specifications and translates them into as set of configuration parameters and a configurable sensor node, which receives the parameter set and adjusts its application behavior accordingly.

As was previously mentioned, the GUI has been implemented in the form of a typical installation wizard. A list of the individual wizard pages, which leads to the configuration of the sensor node, is presented in Table 4.3. The table also describes which specifications are made by the user on each page. The GUI implementation has been performed with the QT software framework², ²http://qt-project.org/
4 Usability

<table>
<thead>
<tr>
<th>Wizard Page</th>
<th>Specification Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node/Network</td>
<td>Node name and network affiliation</td>
</tr>
<tr>
<td>Sensing</td>
<td>Sensor selection and sample rate</td>
</tr>
<tr>
<td>Communication</td>
<td>Time- or event-driven communication</td>
</tr>
<tr>
<td></td>
<td>Sensor threshold (only event-driven)</td>
</tr>
<tr>
<td>Processing/Storage</td>
<td>Processing type and sample set</td>
</tr>
<tr>
<td></td>
<td>Storage option (all or transmitted samples)</td>
</tr>
<tr>
<td>Verification</td>
<td>Clear text node behavior listing</td>
</tr>
<tr>
<td>Lifetime</td>
<td>Lifetime estimate and adjustment options</td>
</tr>
<tr>
<td>Configuration</td>
<td>Configuration save and upload option</td>
</tr>
</tbody>
</table>

Table 4.3: Page overview of the specification wizard with the respective specification parameters to be defined.

which provides access to a number of libraries for user interface development and a broad user community. In the implementation, special consideration has been paid to the usage of natural language, as opposed to technical terms. The intention was to allow the users to answer simple questions about their intended measurement application, in which the answers could be binary (i.e., yes or no), a selection from a provided list, or a numeric specification, such as required for sample periods. Combinations of answers, which are technically impossible or do not make sense, are avoided if possible. In general, special attention has been paid to avoid entries, which result in erroneous node behavior, as technically unexperienced users are unlikely to localize and revert such implementations.

In order to create a configurable sensor node, the firmware which runs on the node has been parametrized. This means that its behavior is altered based on the configuration parameters, which are received from the specification tool. As a result, unused functions are disabled and numeric configurations, such as sensor sample rates, are adjusted. In order to allow for the current configuration to be maintained, even after a reset condition (e.g., when the node has been disabled for a period of time), the received configuration parameters are stored in a non-volatile memory. The parametrized firmware
4.3 Usability for End Users

![State-graph of the configurable sensor node firmware. The configuration update state can be reached by an event from any state in the graph.]

has been implemented on the SENTIO-em platform, and is based on the HFSM programming model, which has been presented in Section 4.2.1. As opposed to if-else or switch conditions, which would typically be used in order to create a configurable firmware behavior, the utilization of an FSM based firmware allows for configurations to be performed by altering the state graph of the application. As the result is a single evaluation at the time of parameter reception, runtime overhead is significantly reduced in comparison to continuous condition evaluations. An overview of the implemented firmware is given in Figure 4.7. For the transmission of the configuration parameters from the specification wizard to the sensor node, a conventional USB interface has been utilized. The interface allows for the configurations to be performed on any PC-class workstation, without further requirement on specialized hardware resources. Moreover, the physical connection of the interface enables the specification software to automatically detect compatible sensor node devices. Based on the small number of configuration parameters to be transmitted, however, the interface can easily be switched in order to perform configurations wirelessly. As multiple devices might be
in communication range, however, additional precautions must be met for the correct device to be configured.

For the evaluation of the implemented system, the configurable firmware was loaded to an enclosed sensor node, which has been equipped with an USB extension and example sensors (see Figure 4.8). This sensor node, in combination with the implemented specification tool, has been supplied to a number of test users. The group of test users was selected on a voluntary basis and included users with limited programming skills, who represent the targeted end user type, but also technically experienced personal, who could provide detailed comments on the implemented configuration method. As the scope of the evaluation did not allow for actual measurement problems to be implemented, the users have been instructed with a set of measurement scenario descriptions. The scenarios have been developed in a role-play manner, which provided the user with his/her intentions and the measurement goal to be performed. Detailed instructions regarding how to perform the implementation of these measurement goals have been omitted purposefully. Afterwards, the users were asked to implement their measurement goal on the implemented system. During the implementation phase, no further
support was provided to the users. The major goal of the evaluation was to determine whether an unexperienced user would be able to independently implement his/her measurement problem on the provided system. Finally, success has been measured based on the sensor node being functional, and the implemented configuration conforming with the respective scenario description. In addition, the users have been asked to complete an evaluation form, which allowed them to provide additional feedback about the usage of the system.

The results of the performed evaluation have been highly successful. All the test users have completed their configuration process in time periods of about 10 to 15 min, which has resulted in operational sensor nodes. Moreover, nine of the ten test users have managed to configure the sensor node behavior to conform to their respective scenario description. While the remaining user managed to configure the provided sensor node, the implemented configuration mismatched the desired configuration, which was described in the respective scenario. Based on the comments in the evaluation form, this was, however, judged to be an issue with the terminology used in the scenario description rather than with the implemented system. Considering that in a real scenario the end user would have actual knowledge about the measurement to be performed, this is an issue unlikely to occur. Although, extended evaluation studies, particularly including actual domain scientists, are desirable, the preliminary study demonstrated the applicability and simplicity of the demonstrated method.
5 Discussion and Conclusions

Environmental monitoring can benefit greatly from the utilization of Wireless Sensor Networks, as its usage allows for measurements to be performed with adjustable spatial and temporal distributions. Although the network is composed of individual sensor nodes, which are distributed in the site to be monitored, the system typically can be operated from a single location by remotely accessing the network to be controlled. A number of application examples and proof-of-concept studies have demonstrated these benefits in-situ. However, they have also uncovered technical challenges to be addressed in order for WSNs to become a feasible measurement solution in the domain. Among these challenges are the operational lifetime of the network and the usability of the system setup, which have been addressed on the system level in this thesis.

The lifetime of a WSN, which is tightly correlated with the lifetime of the individual sensor nodes in the network, has been a concern since the beginning of research in this domain. As a result, a great amount of work has been conducted in order to prolong the lifetime of a node. Contributions to this research have been made by the work, presented in this thesis, in two separate ways, namely the energy efficiency improvement of the sensor node’s task execution and the extension of available energy resources by utilizing ambient energy sources.

With respect to the energy efficiency, a wireless sensor node implementation has been presented, which is specifically designed for its application in low sample rate data gathering scenarios, which are commonplace in the environmental monitoring domain. The utilization of a modern controller architecture and the separation of power domains for individual submodules, allowed for the node design to both increase the processing performance and decrease the power consumption as compared to common reference nodes. Special attention has been paid to the reduction of quiescent currents, which will define the power consumption in low-power modes and, thus,
define the overall energy consumption in low duty cycle applications. In order to achieve such low duty cycles, the active task periods are required to be minimized. This involves, particularly, wireless communication tasks, as its contribution to the energy consumption can be considerable, because of its relatively high power levels. A simple synchronization method has been presented, which allows for short communication periods to be achieved, due to the possibility of scheduling such tasks in a TDMA-like manner. The utilized method demonstrates synchronization accuracies of the order of tens to hundreds of μs, which allows for accurate time slots to be created, while maintaining a minimal overhead for the required synchronization. Although, a division exists within the research community between contention-free and contention-based communication protocols, with contention-based protocols being used in the majority of applications, contention-free protocols possess multiple advantages in relation to data gathering applications. The major advantages of contention-based protocols are its low latency and the possibility to adjust to a dynamic system behavior. These features, however, have only low value in typically static environmental measurement systems with low requirements on latency. A limitation of time-scheduled protocols is its restricted scalability, which, however, can be improved by utilizing hierarchical scheduling mechanisms. The optimized implementation and its respective performance evaluations of such a hierarchical method are currently being analyzed.

While the improvement of energy efficiency allows for the lifetime of sensor nodes to be extended, a limitation remains due to the restricted capacity of the energy source, which is commonly battery based. Particularly in outdoor applications, solar energy harvesting has been an appropriate method in order to virtually increase the energy capacity of a sensor node. The majority of such systems, however, have been demonstrated, based on secondary batteries, which are inherently limited in their device lifetime. While the large capacity of batteries are required in certain applications, low power applications have been demonstrated with DLCs as their storage medium. Double Layer Capacitors can store only relatively small amounts of energy, but they have a high power density, which allows for their charge level to be quickly restored. Moreover, as energy is stored in the form of charge separation, as opposed to the chemical reactions that are occurring in batteries, the lifetime of DLCs are much higher. For the usage of DLCs in solar energy
harvesting architectures, however, regular irradiation is required in order to supply the sensor node from a small capacity for prolonged periods of time. This has limited the application of these energy buffers to locations with high solar irradiation. Contributions from this thesis, have demonstrated that it is feasible to utilize short-term energy buffers, such as DLCs, even at locations with limited solar irradiation. A number of solar energy harvesting architectures have been implemented and analyzed in order to evaluate the system behavior. The major focus, in relation to this, has been the energy overhead of the harvester itself, meaning the amount of energy it takes to operate the harvester, which will define the irradiance level from which energy can be harvested. As an alternative to DLCs, Thin-Film Batteries have been analyzed for usage in solar energy harvesting architectures. While TFBs have been demonstrated to be a feasible alternative in low power applications, their currently high internal resistance, in combination with a limitation to extremely small capacities, restricts the scalable utilization of these devices. As a result, the DLCs present a number of advantages in applications, for which size is not the primary limitation. Further analysis of the scalability of DLC based systems for sensor nodes with higher power consumption are under consideration.

A common dispute exists between researchers in the community as to whether MPPT should be utilized in small-scale solar energy harvesting applications, such as those implemented as sensor node power supplies. While the performance improvement of MPPT, in general, is irrefutable, its operational overhead does not scale with the scaling harvesting system. As a result, the overhead created by common MPPT techniques becomes considerable, which results in the utilization of simplified, less accurate, tracking methods. While, initially, MPPT has been avoided in the harvesting architectures, architectures with a commercial harvesting circuit, which utilize MPPT, have been evaluated. In comparison with previous architectures, which were optimized for low energy overhead, these architectures have demonstrated a reduced performance in low irradiance situations. This result verified the initial expectations, as most simplified MPPT methods utilize constant factoring of the open-circuit voltage or short-circuit current, which is typically optimized for high irradiance levels.

As experimental evaluations have proven to be valuable for the comparison of harvesting architectures, however, the work load, which is required
for repetitive system deployments, is considerable, a novel approach for the evaluation of multiple architectures and configurations has been developed. The system is based on the ideas of remote configuration and monitoring, which is typically implemented in WSN testbeds. Such a solar energy harvesting testbed can, therefore, be used to conduct multiple experiments, while the system is deployed only once. The system design of the presented system uses a typical WSN architecture in order to send configuration packets and collect measurement parameters. Each node, however, is extended by means of a harvesting module, which allows configurations to be made to its hardware. The testbed is operated via a user interface that allows for configurations to be created and assignments of configurations to individual testbed nodes to be made. Moreover, the same interface can be used in order to access the collected data for visualization or further analysis. The system has been implemented and tested in a proof-of-concept deployment of the solar energy harvesting testbed. Evaluation results on the performance of the testbed nodes and exemplar output values from the testbed have been collected, and they demonstrate how the solar energy harvesting can be used in order to collect experimental results for the analysis and modeling of solar energy harvesting architectures in a more efficient manner. A limitation to the system is the requirement to have knowledge of the type of desirable configurations, as the configurable hardware has to be prepared accordingly. This limitation can be addressed by developing a unified interface, which allows for modular harvesting modules to be implemented in the system in a plug-and-play fashion. While the control and monitoring architecture could be maintained, the actual harvesting circuitry would be exchangeable, which allows for the experimental setup to be altered in order to integrate new harvesting methods to the testbed. An adjustment in this form is planned to be developed for further improvement in relation to system flexibility.

Although experimental evaluations are important in order to understand and compare individual harvesting methods, their results can additionally contribute to the development of accurate component and system models. These models, in turn, are important to accelerate the study of ambient energy harvesting, as well as their appropriate utilization in actual WSN applications. While ambient energy sources, such as solar irradiation, are not capacity limited, a rate limitation applies which is based on diurnal, seasonal and spatial variations. This means that systems with the same configuration will behave
differently if the system is deployed at a different time or location. Prior to network deployment, adjustments to the harvester configuration (e.g., storage or panel size) or the energy profile of the node (e.g., by adjustment to the sample rate) might be required in order to establish perpetual system operation. As an analogy, capacities in pure battery operated sensor nodes are selected dependent on load conditions and the desired system lifetime. While this typically occurs on an analytical basis for batteries, the complexity of irradiation patterns and the energy harvesting circuitry prohibits an analytical method if solar energy harvesting is utilized. We have proposed the modeling of the energy harvesting architecture in order to simulate the state of charge at any given point in time. Similarly, to the calculation of the battery lifetime, these estimations can be used in combination with load conditions and irradiation parameters in order to determine the optimal configuration of the system to be used. The implementation of an exemplar harvesting architecture based on individual component models demonstrated the accuracy of this modeling technique and their respective simulations. Difficulties arise, however, due to the time consuming development of accurate system models. Furthermore, in the case of the utilization of commercial ICs, the creation of an accurate component model becomes tremendously difficult. The study of alternative modeling methods, which promise faster model development, has been initiated. In this respect, particularly, the application of artificial neural networks appears to be promising, as a complete harvesting architecture can be modeled at one time, without the necessity of modeling each individual component of the architecture. A loss in accuracy, however, is expected as a trade-off for model development time.

In addition to the contributions to the improvement of sensor node lifetime, the usability of sensor nodes has been addressed in this thesis. This has been carried out for developers and end users separately, as each group presents their individual needs and skills. While developers typically require interfaces that allow for the development of new applications and algorithms to be performed in an effective manner, end users require simplified interfaces, which enable them to configure an existing system towards their application requirements. In order to increase the usability for the developers, who possess a technical understanding of the underlying system, a Hierarchical Finite State Machine programming framework has been designed and implemented. The utilization of FSMs for the software development in
sensor nodes allows for the development process to be carried out in a systematic manner. Moreover, software can be easily split into smaller modules (i.e., states), which improves the readability and modularity of the generated source code. Evaluation results show that the implemented applications contain only limited overhead, both in timing and memory usage. The usability improvements, which are gained from this programming method, thus, come at low cost. Although usability is only improved in cases of user familiarity with FSMs, this is to be expected in the majority of cases, as system developers are typically computer scientists or engineers, who are commonly exposed to FSM description models.

While improving the usability of programming methods can reduce the amount of mistakes to be made, it cannot completely eliminate the necessity of verification measurements. The resource limitation of sensor nodes, which is typically a result of size and cost restrictions, hinders these measurements to be conducted in an effective manner. We proposed a number of simple considerations to the design of wireless sensor nodes, which can improve the usability in evaluation scenarios. These considerations have been presented based on typical environmental monitoring requirements, and have been implemented in the design of the SENTIO-em node platform. The usage of the SENTIO-em platform in a number of outdoor experiments, has successfully demonstrated the gained usability improvements of the implemented modifications.

Finally, end user accessibility to WSNs has been the focus of the investigations performed in this thesis. Although domain experts have been part of the development process in a number of environmental monitoring applications, the systems, in these scenarios, have been operated by the system developers (i.e., system experts) rather than the designated end user (i.e., the domain experts). As a result, the real-world applicability of WSNs as a measurement instrument in the domain, is strictly limited. A method has been presented, which allows for end users to modify the node behavior according to application requirements, without the necessity of a technical understanding of the utilized system. The modifications are performed via a GUI, which reproduces the method of software installations. The created configuration parameters are used in a configurable firmware in order to alter the sensor node behavior. The feasibility of the configuration method has been evaluated by means of a number of test users, which demonstrated
the usability of the implemented system. Improvements are planned to be conducted in two ways. On the one hand, the hardware configuration has to be simplified in order for users to equip the sensor node according to their specific needs. This means, in the main, that users should be able to connect the desired sensor modules to the node in a simple manner. The development of a unified sensor interface, which allows for the sensor to be detected and configured by the sensor node in an automatic fashion, appears to be a feasible solution. Furthermore, extended usability tests are to be conducted, which involves actual domain experts and real-world application tasks. A more detailed evaluation, which could include the recording of the user and the activity in the GUI, might support the optimization of the layout and formulations in the design.

In overall conclusion, the work that has been presented in this thesis, contributed to node level improvements, which improve the real-world application of WSNs in measurement scenarios for the environmental monitoring domain. While improvement opportunities remain, the presented methodologies and implementations support the prolonged operational lifetime and efficient system utilization. These are both required steps for Wireless Sensor Networks in order to leave proof-of-concept evaluations and become real-world applicable measurement instruments.
Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


