Evaluierung der Energie-Effizienz von Sicherheitsmechanismen in drahtlosen Sensornetzen

zur Erlangung des akademischen Grades eines

DOKTORS DER INGENIEURWISSENSCHAFTEN

der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Dipl.-Inform. Christian Haas

aus Achern

Tag der mündlichen Prüfung: 7. Februar 2014

Erste Gutachterin:	Prof. Dr. habil. Martina Zitterbart Karlsruher Institut für Technologie (KIT)
Zweiter Gutachter:	Prof. Dr.-Ing. Jochen Schiller Freie Universität Berlin

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Der Trend zur Integration von immer mehr Alltagsgegenständen in das Internet hat sich in den vergangenen Jahren stetig verstärkt. Eine Vision ist das sogenannte *Internet der Dinge*, in dem herkömmliche Computersysteme durch intelligente Gegenstände ersetzt werden [2] [3]. Ziel hierbei ist es, den Menschen im Alltag mit Hilfe von intelligenten Gegenständen zu unterstützten ohne dabei abzulenken. Aktuelle Studien sagen voraus, dass bis zum Jahr 2020 bis zu 30 Milliarden Geräte im Internet der Dinge eingebunden sein könnten [4].

Typische Anwendungsbeispiele für das Internet der Dinge sind neben der intelligenten Umgebung, beispielsweise im intelligenten Haus (*Smart Home*), auch andere Bereiche wie der intelligenten Fahrzeugen oder in zukünftigen, intelligenten Energienetzen (*Smart Grid*). Gemeinsame Vision all dieser intelligenten Umgebungen ist es, dass neben Menschen auch physikalische Gegenstände über das Internet kommunizieren und interagieren.

Drahtlose Sensornetze (DSN) sind ein integraler Bestandteil des Internet der Dinge. Sie bestehen aus einer Vielzahl von drahtlosen Sensorknoten. Sensorknoten sind eingebettete Systeme, welche physikalische Größen in ihrer Umwelt mittels Sensorik messen (Temperatur, Bewegung, Helligkeit,...), verarbeiten und diese drahtlos an andere Sensorknoten oder Teilnehmer des Internet der Dinge übertragen können. Sensorknoten unterliegen einer starken Beschränkung der vorhanden Ressourcen, sowohl was die vorhandene Rechenkapazität als auch den vorhandenen Energievorrat angeht. Die Energieversorgung erfolgt dabei typischerweise über Batterien. Die auf den Sensor-

knoten eingesetzten Mikroprozessoren verfügen im Allgemeinen über einen Prozessortakt von wenigen Megahertz und einen Speicher von wenigen Kilobyte. Viele klassische Kommunikationsprotokolle können daher aufgrund der Ressourcenbeschränkungen nicht eingesetzt werden.

1.1 Zielsetzung und Beiträge der Arbeit

In den meisten Einsatzszenarien ist neben dem robusten Betrieb des Netzes auch die Notwendigkeit einer sicheren Kommunikation gegeben.

Ein Beispiel für eine Anwendung für drahtlose Sensornetze mit Sicherheitsanforderungen ist das Projekt *FleGSens* (Sichere und flexible Grenz- und Liegenschaftsüberwachung durch drahtlose Sensornetze). Das Projekt beschäftigte sich mit dem Entwurf, der prototypischen Implementierung und Erprobung eines Sensornetzes zur Überwachung einer *grünen Grenze* [40][41]. Ziel des Projekts war es, mit Hilfe eines Sensornetzes und einfacher Sensorik, ein unberechtigtes Überqueren der vorab definierten grünen Grenzen zu detektieren und den Ort des Übertritts an einer zentralen Basisstation hinreichend genau und zeitnah anzuzeigen. Besonderer Fokus lag dabei auf der Informationssicherheit und der Robustheit gegenüber Ausfällen einzelner Sensorknoten. Insbesondere sollte das Sensornetz nicht durch Angriffe in seiner Funktionalität gestört werden können.

Um Sicherheitsanforderungen wie diese umzusetzen, wurde innerhalb der Forschung eine Vielzahl von kryptografischen Mechanismen und Protokollen entworfen. Der Einsatz dieser Mechanismen und Protokolle ist jedoch mit einem erheblichen Ressourcenaufwand verbunden, kryptografische Algorithmen und Protokolle erfordern häufig relativ lange Berechnung bzw. einen erhöhten Kommunikationsaufwand. So steigen in vielen Fällen die Anzahl der Pakete bzw. die Größe der zu übertragenen Datenmenge. Ein realitätsnaher Vergleich der Energieeffizienz von verschiedenen Protokollen und Algorithmen stand bisher häufig nicht im besonderen Fokus.

In vielen Arbeiten wurden zum Vergleich unterschiedlicher Protokolle theoretische Modelle zur Berechnung des Energiebedarfs entwickelt und eingesetzt. Häufig werden wichtige Einflussfaktoren, wie beispielsweise das eingesetzte Medienzugriffsverfahren, nicht in das Modell einbezogen. Ein Vergleich des theoretischen Modells mit den Ergebnissen einer realen Implementierung und Vermessung der Protokolle auf echten Sensorknoten ist selten vorzufinden.

Die realitätsnahe und genaue Messung und Evaluierung des Energiebedarfs ist eine noch nicht vollständig gelöste Herausforderung, da echte Messungen des Energiebedarfs zusätzliche Messhardware benötigen und einen erhöhten

Aufwand während der Evaluation darstellen. Die beste Genauigkeit bei der Messung des Energiebedarfs von drahtlosen Sensornetzen lässt sich durch reale Messungen an echten Sensorknoten erreichen, wie es beispielsweise in dem am Institut entwickelten Sensornetz-Testbed SANDBed [5] möglich ist.

Während die Genauigkeit und Realitätsnähe von Messungen auf realer Hardware ein entscheidender Vorteil gegenüber beispielsweise Simulationen ist, existieren auch einige Nachteile. Die Anzahl der im Testbed verfügbaren Sensorknoten ist üblicherweise begrenzt, eine Evaluation von großen Sensornetzen lässt sich nur mit erheblichem Aufwand durchführen. Weiterhin sind Experimente im Testbed nur in Echtzeit möglich, die Anzahl der durchführbaren Messungen ist durch die Laufzeit der Experimente beschränkt, eine Parallelisierung von Experimenten ist kaum möglich.

Simulationswerkzeuge bieten für die oben genannten Skalierungsprobleme geeignete Lösungsansätze, die Anzahl der simulierbaren Sensorknoten ist weniger stark beschränkt, zudem lassen sich Experimente häufig schneller als in Echtzeit durchführen und sind einfach parallelisierbar. Die Messung des Energiebedarfs ist jedoch nur mit wenigen Simulationswerkzeugen möglich. Ein solches Simulationswerkzeug ist der Sensornetzsimulator AVRORA. Zur Simulation des Energiebedarfs emuliert AVRORA zyklengenau die Sensorknotenhardware und kann somit sowohl die einzelnen Zustände der Hardware als auch Zustandswechsel identifizieren und einem Energiebedarf zuordnen. Die Genauigkeit und Realitätsnähe des so implementierten Energiemodells wurde bisher nicht im Vergleich zu echter Hardware analysiert oder evaluiert.

Die Zielsetzung dieser Dissertation ist es, die Energie-Effizienz von Sicherheitsmechanismen in drahtlosen Sensornetzen systematisch und möglichst realitätsnah zu evaluieren. Zur Durchführung einer realitätsnahen Evaluierung werden das am Institut entwickelte Sensornetz-Testbed SANDBed [5] und AVRORA+ [6] als Evaluationswerkzeuge genutzt.

Diese Dissertation umfasst folgende Beiträge:

- *Evaluierung des Energiebedarfs von drahtlosen Sensornetzen mittels AVRORA+:* Im Rahmen dieser Dissertation wird die Realitätsnähe von AVRORA im Vergleich zu SANDBed untersucht und das Energiemodell erweitert bzw. angepasst. Es wird gezeigt, dass AVRORA sowohl den Energiebedarf der einzelnen Hardwarezustände als auch die Kosten für Zustandsübergänge nicht realitätsnah abbildet. Weiterhin berücksichtigt das Energiemodell in AVRORA nicht die in der Realität vorhanden Schwankungen des Energiebedarfs von unterschiedlichen Sensorknoten des gleichen Typs aufgrund von Ungenauigkeiten im Fertigungs-

prozess. Mit AVRORA+ wird eine Erweiterung des Energiemodells in AVRORA entwickelt, mit dessen Hilfe die Abweichung des gemessenen Energiebedarfs im Vergleich zu realen Messungen von ursprünglich bis zu 15% auf maximal 2% deutlich reduziert wird. Die Erweiterungen in AVRORA+ umfassen neben der Anpassung der Kosten der einzelnen Hardwarezustände auch die Einführung von Kosten für Zustandsübergänge und von Schwankungen des Energiebedarfs, wie sie auf realen Sensorknoten vorzufinden sind.

- *Evaluierung der Energieeffizienz von Medienzugriffsverfahren in drahtlosen Sensornetzen:* Die Kommunikation zwischen einzelnen Sensorknoten ist ein elementarer Bestandteil fast jeder Anwendung in drahtlosen Sensornetzen. Ziel von Medienzugriffsverfahren in DSNs ist es, die Funkschnittstelle so häufig wie möglich abzuschalten und somit nicht einen unnötig hohen Energiebedarf zu verursachen. In dieser Arbeit werden unterschiedliche Medienzugriffsverfahren für drahtlose Sensornetze implementiert und der Energiebedarf mit Hilfe von unterschiedlichen Testszenarien sowohl in AVRORA+ als auch in SANDBed evaluiert und analysiert. Es wird gezeigt, dass gängige Annahmen über den Energiebedarf der Kommunikation häufig nicht verallgemeinerbar sind. So wurde bisher beispielsweise häufig argumentiert, dass eine Verringerung der zu übertragenden Datenmenge automatisch zu einem geringeren Energiebedarf führt. Die Ergebnisse der Messungen mit Hilfe von AVRORA+ und SANDBed können diese Annahme nicht allgemein bestätigen, in einigen Testszenarien kann sogar der umgekehrte Fall beobachtet werden, eine Erhöhung der Datenmenge verringert den Energiebedarf der Sensorknoten.

- *Evaluierung der Energieeffizienz von Schlüsselaustauschprotokollen:* Im Rahmen dieser Dissertation wird die Evaluierung der Energieeffizienz von Sicherheitsmechanismen am Beispiel von Schlüsselaustauschprotokollen durchgeführt. Hierfür werden Kerberos und ECDH-ECDSA auf echten Sensorknoten implementiert und der Energiebedarf unter Verwendung von verschiedenen Medienzugriffsverfahren in AVRORA+ evaluiert. Es wird untersucht, welche Evaluationsmetriken, wie beispielsweise absoluter Energiebedarf im Vergleich zu relativem Energiebedarf, zu aussagekräftigen Ergebnissen führen. Es kann gezeigt werden, dass obwohl der absolute Energiebedarf je nach Medienzugriffsverfahren für ECDH-ECDSA bis zu 23-mal höher als für Kerberos ist, der relative Energiebedarf über ein festes Messintervall nur um den Faktor 5 variiert.

- *Evaluierung der Energieeffizienz von hardwarebasierten Sicherheitsmechanismen:* Eine häufig genannte Möglichkeit zur Steigerung der Energieeffizienz von Sicherheitsprotokollen in drahtlosen Sensornetzen ist der Einsatz von Hardware-Sicherheitsmodulen wie beispielsweise TPM-Chips. Diese Bausteine versprechen im Vergleich zur einer Implementierung der kryptografischen Algorithmen in Software eine deutliche Beschleunigung der Berechnung. Hierdurch verspricht man sich neben der schnelleren Berechnungszeit auch eine Senkung des Energiebedarfs. Allerdings wurde die mögliche Senkung des Energiebedarfs bisher nicht in einem realen Anwendungsszenario unter Verwendung einer echten Implementierung sowohl in Hardware als auch in Software untersucht. Im Rahmen dieser Dissertation wird die Energieeffizienz von hardwarebasierten Sicherheitsmodulen im Vergleich zu reinen Softwareimplementierungen durchgeführt. Die Messungen des Energiebedarfs werden hierfür im Testbed SANDBed durchgeführt. Es kann gezeigt werden, dass der Einsatz von hardwarebasierten Sicherheitsmodulen den Energiebedarf von kryptografischen Algorithmen um bis zu 76% gesenkt werden kann, allerdings nicht für alle implementierten Algorithmen und nur unter Verwendung eines geeigneten Duty-Cyclings.

1.2 Zugrundeliegende Veröffentlichungen

Diese Dissertation basiert auf verschiedenen Veröffentlichungen. Die dort präsentierten Ergebnisse wurden im Rahmen dieser Dissertation umfangreicher evaluiert als es im Rahmen der Veröffentlichung möglich war. Teilweise bauen die Beiträge dieser Arbeit auf gemeinsamen Veröffentlichungen mit Joachim Wilke auf, dessen Dissertation die Energieeffizienz von Concast Kommunikation untersucht. Die gemeinsamen Beiträge umfassen dabei die *Evaluierung des Energiebedarfs mittels AVRORA+* [7] und die *Evaluierung der Energieeffizienz von Medienzugriffsverfahren* [8]. Die Beiträge wurden im Rahmen dieser Dissertation erweitert und zum Teil umfangreicher evaluiert.

- Der in Kapitel 3 vorgestellte Simulator AVRORA+ ist im Rahmen einer Veröffentlichung auf der *European Conference on Wireless Sensor Networks (EWSN 2012)* [7] entstanden.

- Eine erste Evaluation des Energiebedarfs von Medienzugriffsverfahren wurde auf der Konferenz *International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2011)* [8] vorgestellt.

- Der Energiebedarf von Schlüsselaustauschprotokollen wurde auf den Konferenzen *International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2011)* [9] und der *38th IEEE Conference on Local Computer Networks (LCN) 2013* [10] vorgestellt.
- Die Evaluierung von hardwarebasierten Sicherheitsmechanismen wurde erstmals in einem Demonstrator auf der *European Conference on Wireless Sensor Networks (EWSN 2012)* [11] gezeigt. Eine ausführlichere Evaluation des Energiebedarfs wurde im Rahmen des *9th IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing (2013)* präsentiert [12].

2. Grundlagen

Ziel dieses Kapitels ist es, die für das Verständnis der Arbeit notwendigen Grundlagen hinsichtlich drahtlosen Sensornetzen, der eingesetzten Hardware und der Evaluierung des Energiebedarfs zu vermitteln.

Das Kapitel gliedert sich wie folgt: Im ersten Teil des Kapitels werden physikalische Messgrößen und deren Zusammenhänge vorgestellt. Anschließend werden die im Rahmen der Arbeit verwendeten Sensorknotenplattformen sowie das eingesetzte Betriebssystem erläutert. Im zweiten Teil des Kapitels werden die in dieser Arbeit verwendeten Medienzugriffsverfahren und die darin eingesetzten Duty-Cycling Mechanismen anhand einfacher Kommunikationsbeispiele vorgestellt. Zum Abschluß des Kapitels werden die verwendeten Evaluierungswerkzeuge für die Messung des Energiebedarfs der Sensorknoten vorgestellt.

2.1 Physikalische Messgrößen und Begriffsdefinitionen

- **Spannung U** [V]: Sensorknoten benötigen zur fehlerfreien Funktion eine bestimmte Spannung U.Die notwendige Spannung ist dabei abhängig von der angesetzten Sensorknotenplattform bzw. den dort verwendeten Hardwarekomponenten. Die Spannung wird typischerweise in Volt [V] gemessen. Die in dieser Arbeit verwendeten Sensorknoten MICAz und IRIS benötigen eine Spannung zwischen $2,7V$ und $3,3V$.

- **Stromstärke I** [A]: Die Stromstärke eines Sensorknotens ist abhängig von den Hardwarezuständen, in denen sich der Sensorknoten bzw. dessen Hardwarekomponenten zum Messzeitpunkt befinden. Die Strom-

stärke I wird dabei in Ampere [A] gemessen. Die in dieser Arbeit eingesetzten Sensorknoten weisen typischerweise eine Stromstärke zwischen 0 und $60mA$ auf.

- **Leistung P** [W]: Die Leistung, oder auch Leistungsaufnahme, P eines Sensorknotens zu einem bestimmten Messzeitpunkt lässt sich mit

$$P = U \cdot I$$

berechnen. In dieser Arbeit weisen die Sensorknoten eine Leistung zwischen 0 und $198mW$ auf.

- **Energiebedarf W** [mWs oder mJ]: Der Energiebedarf W eines Sensorknotens über einen bestimmten Messzeitraum t lässt sich berechnen mit

$$W = U \cdot I \cdot t$$

Die für den Betrieb der Sensorknoten MICAz oder IRIS eingesetzten Batterien weisen typischerweise eine Kapazität von $3600 - 5200mAh$ auf. Bei einer konstanten Leistung von $198mW$ würde dies einen Betrieb von maximal 87 Stunden bedeuten[1].

2.2 Drahtlose Sensornetze

Drahtlose Sensornetze (DSN) sind Netze bestehend aus einer Menge von drahtlosen Sensorknoten. Es existieren verschiedene Sensorknoten Plattformen, typischerweise besteht ein Sensorknoten dabei mindestens aus Prozessor (Mikrocontroller mit Speicher), Funkschnittstelle und verschiedenen Sensoren zur Erfassung von (Umwelt-)Daten. Beispiele für Sensoren sind dabei Temperatur-, Infrarot- oder Feuchtigkeitssensoren. Die Sensorknoten kommunizieren und interagieren je nach Anwendungsanforderung untereinander mit Hilfe der Funkschnittstelle. Im Vergleich zu handelsüblichen Personal Computern verfügen Sensorknoten über sehr begrenzte Ressourcen und zeichnen sich durch eine kompakte Baugröße aus. Die Energieversorgung wird typischerweise über Batterien durchgeführt, eine Versorgung über externe Energiespeicher oder Energieumwandler, wie beispielsweise Solarzellen, ist möglich.

[1] Diese Rechnung ist idealisiert und gilt nur bei konstanter Temperatur und unter der Annahme, dass die Spannung über den Entladezeitraum konstant bleibt. In der Realität sind voraussichtlich deutlich kürzere Laufzeiten zu erwarten.

Abbildung 2.1 MICAz-Sensorknoten der Firma Memsic [1].

2.2.1 Hardware

Es existiert eine Vielfalt an Sensorknoten Plattformen, eine Übersicht über häufig verwendete Hardware findet sich in [13]. In der Forschung werden häufig Sensorknoten der Mica Baureihe [14] verwendet, wie beispielsweise die in dieser Arbeit verwendeten MICAz [1] oder IRIS-Sensorknoten [15].

Die MICAz-Sensorknotenplattform

Abbildung 2.1 zeigt die in dieser Arbeit häufig eingesetzte Sensorknotenplattform MICAz der Firma MEMSIC. Der MICAz-Sensorknoten verfügt über einen ATmega128L Prozessor der Firma Atmel mit 4 kByte RAM zur Speicherung von Konfigurationsdaten und 128 kByte ROM für die Anwendungsdaten. Als Funkschnittstelle wird der CC2420 Funktransceiver der Firma Texas Instruments eingesetzt, welcher laut Datenblatt eine maximale Übertragungsgeschwindigkeit von 250 kbit/s aufweist. Der CC2420 arbeitet dabei nach der IEEE 802.15.4 [16] Spezifikation. Der Sensorknoten wird über zwei handelsübliche AA-Batterien betrieben. Zur Anbindung externen Komponenten oder Sensorik stehen verschiedene Schnittstellen, wie beispielsweise der I^2C oder der SPI-Bus, zur Verfügung.

Die IRIS-Sensorknotenplattform

Im Vergleich zur MICAz-Sensorknotenplattform verfügt ein IRIS-Sensorknoten über 8 kByte statt 4 kByte RAM, hierfür wird der ATmega1281-Prozessor der Firma Atmel eingesetzt. Weiterhin wird statt des CC2420 der RF230

Funktransceiver genutzt, welcher ebenfalls zu der IEEE 802.15.4 Spezifikation kompatibel ist. Die weiteren Austattungsmerkmale der IRIS-Plattform sind identisch mit denen des MICAz-Sensorknotens.

2.2.2 Betriebssysteme für drahtlose Sensornetze

Es existieren eine Vielzahl von Betriebssystemen speziell für drahtlose Sensornetze. Eine Übersicht über die am weitesten verbreiteten Betriebssysteme findet sich beispielsweise in [17].

Eines der am weitesten verbreiteten Betriebssysteme ist dabei TinyOS [18], welches sich laut Farooq et. al. [17] durch die Unterstützung vieler Hardwareplattformen und der flexiblen Einsatzmöglichkeiten auszeichnet.

TinyOS ist ein quelloffenes Betriebssystem. Sowohl das Betriebssystem selbst, als auch Sensornetzanwendungen welche mit TinyOS entwickelt werden, sind in der Programmiersprache nesC [19] geschrieben. NesC ist einem Derivat der Programmiersprache C, welche Erweiterungen für das ereignisbasierte Ausführungsmodell und die komponentenbasierte Architektur der Anwendungen in TinyOS einführt.

Das ereignisbasierte Ausführungsmodell in TinyOS entspricht den Anforderungen an die möglichst energie-effiziente Ausführung von Anwendungen in drahtlosen Sensornetzen [20]. Anstehende Ereignisse sollen demnach möglichst zeitnah abgearbeitet werden, so dass die Hardware der Sensorknoten anschließend in einen energie-effizienteren (Schlaf-)Modus versetzt werden kann.

Die Architektur von Anwendungen in TinyOS ist komponentenbasiert. Eine Anwendung besteht typischerweise aus mehreren eigenständigen Komponenten, welche über frei definierbare Schnittstellen zu einer Anwendung zusammengefasst werden können. Eine Wiederverwendung von bereits implementierten Komponenten ist somit einfach möglich. TinyOS bietet von Haus aus eine Vielzahl von bereits implementierten und getesteten Komponenten, wie beispielsweise Module zum Empfang oder dem Versand von Nachrichten.

TinyOS unterstützt eine Vielzahl von Sensorknotenplattformen, eine Integration von zukünftigen Plattformen wird insbesondere durch die komponentenbasierte Architektur stark vereinfacht. Die in dieser Arbeit verwendeten Plattformen MICAz und IRIS werden von TinyOS unterstützt. TinyOS wird im Rahmen dieser Arbeit in der Version 2.1 verwendet.

2.3 Medienzugriff in drahtlosen Sensornetzen

Mit Hilfe von Medienzugriffsverfahren wird der Zugriff auf das geteilte Funkmedium geregelt. Im Rahmen dieses Abschnitts sollen die Grundlagen der

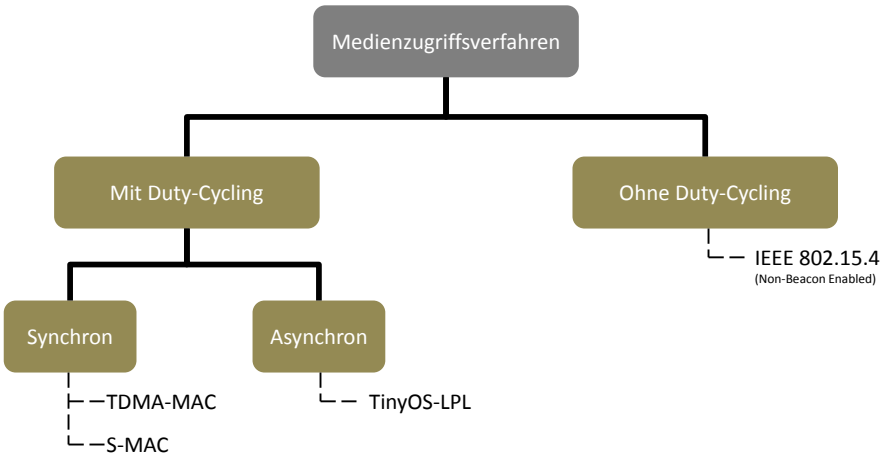


Abbildung 2.2 Klassifikation der in dieser Arbeit verwendeten Medienzugriffsverfahren.

Medienzugriffsverfahren erläutert, die im weiteren Verlauf der Arbeit genutzt werden. Da die Funkschnittstelle eines Sensorknotens typischerweise die Hardwarekomponente mit dem größten Energiebedarf ist, wurden verschiedene *Duty-Cycling* Strategien für die Funkschnittstelle entwickelt um den Energiebedarf zu senken. Das Duty-Cycling wird häufig zusammen mit dem eigentlichen Medienzugriffsverfahren implementiert. Abbildung 2.2 klassifiziert die im Rahmen dieser Arbeit genutzten Medienzugriffsverfahren anhand der verwendeten Duty-Cycling Strategie: Neben einem Medienzugriffsverfahren *ohne* Duty-Cycling, werden Medienzugriffsverfahren mit *synchronem* und *asynchronem* Duty-Cycling genutzt. Um die Funktionsweise der verschiedenen Medienzugriffsverfahren zu verdeutlichen, sollen im Folgenden Ablaufdiagramme genutzt werden, die sowohl den Zustand der Funkschnittstelle (an/aus) als auch die Übertragung von Paketen zwischen zwei Sensorknoten A und B veranschaulichen. Neben Datenpaketen werden dabei je nach Medienzugriffsverfahren auch Bestätigungen (Acknowledgements - ACKs) eingesetzt.

IEEE 802.15.4

Der Standard IEEE 802.15.4 [16] definiert ein Medienzugriffsverfahren für Wireless Personal Are Networks (WPANs). Neben dem Medienzugriffs wird auch die Bitübertragungsschicht spezifiziert. IEEE 802.15.4 spezifiziert somit die Schichten 1 und 2a des OSI-Modells. Der im Rahmen dieser Arbeit häufig

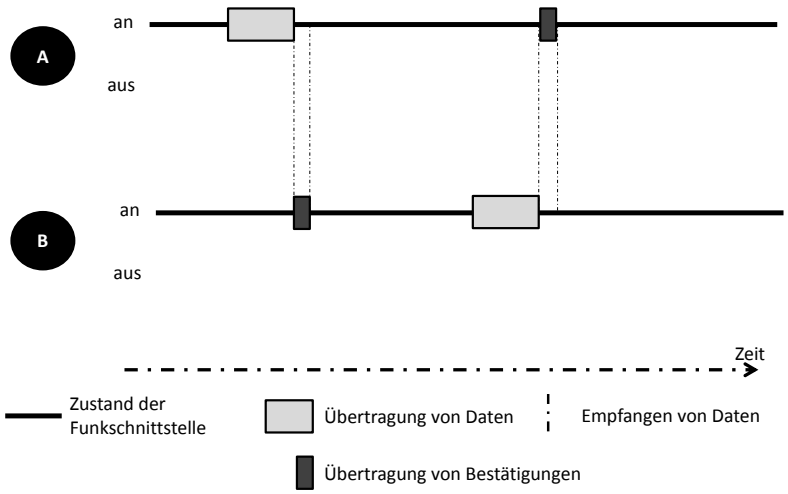


Abbildung 2.3 Datenübertragung zwischen zwei Sensorknoten A und B mit IEEE 802.15.4.

eingesetzte Funktransceiver CC2420 ist kompatibel zum IEEE 802.15.4 Standard. Für den Medienzugriff sieht der IEEE 802.15.4 Standard zwei mögliche Betriebsmodi vor: Im *Non-Beacon Enabled* Modus verwenden die Sensorknoten einen *unslotted CSMA/CA*[2] Mechanismus mit einem zufälligen Backoff. Im *Beacon Enabled* Modus werden die Daten mit *slotted CSMA/CA* übertragen, wobei die Zeitschlitze nach den von einem Koordinator versendeten *Beacon* ausgerichtet werden. Der von TinyOS standardmäßig verwendete Modus ist der Non-Beacon Enabled Modus, welcher daher im Rahmen dieser Arbeit verwendet wird.

Der Ablauf einer Datenübertragung zwischen zwei Sensorknoten A und B mit IEEE 802.15.4 im Non-Beacon Enabled Modus ist in Abbildung 2.3 zu sehen. Da der IEEE 802.15.4 Standard keinen Duty-Cycling Mechanismus spezifiziert, befindet sich die Funkschnittstelle dauerhaft im angeschalteten Zustand. Die gesendeten Datenpakete werden mit Hilfe von ACKs bestätigt. Ein im Bezug auf den Energiebedarf offensichtlicher Nachteil ist die dauerhaft eingeschaltete Funkschnittstelle, die einen Energiebedarf verursacht, obwohl unter Umständen keine Daten übertragen werden (auch *idle-listening* genannt). Ein Vorteil von IEEE802.15.4 im Non-Beacon Enabled Modus ist es jedoch, dass keine zusätzliche Latenz durch das Medienzugriffsverfahren hinzugefügt wird. Eine Datenübertragung kann im Normalfall sofort durchgeführt werden.

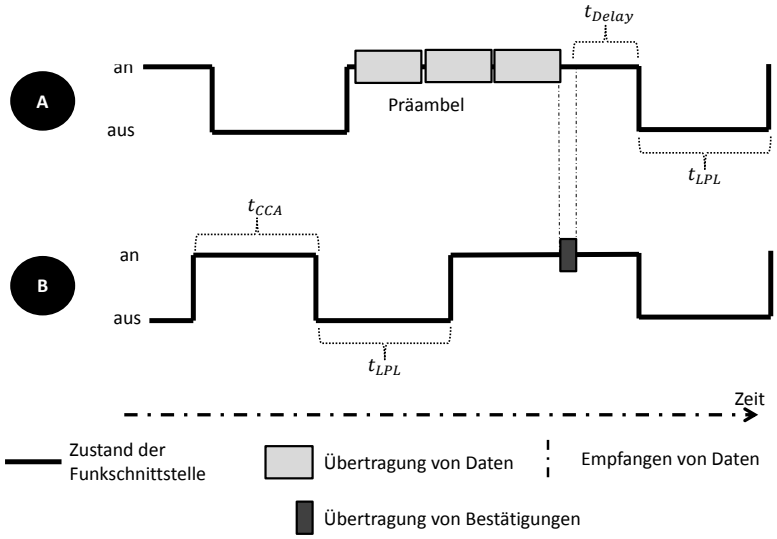


Abbildung 2.4 Datenübertragung zwischen zwei Sensorknoten A und B mit TinyOS-LPL.

TinyOS-LPL

TinyOS-LPL [21] ist eine Implementierung des BoX-MAC-2 Protokolls [22] für TinyOS. TinyOS-LPL ist ein Medienzugriffsverfahren mit einem asynchronem Duty-Cycling Mechanismus. Eine Datenübertragung zwischen zwei Sensorknoten A und B mit TinyOS-LPL ist in Abbildung 2.4 dargestellt. Das grundlegende Funktionsprinzip von TinyOS-LPL ist es, dass sich die Funkschnittstelle grundsätzlich im ausgeschalteten Zustand befindet. Die Sensorknoten schalten die Funkschnittstelle jedoch in einem periodischen, voneinander unabhängigen Zeitintervall an und überprüfen das Medium auf potenzielle Datenübertragungen. Dieser Vorgang wird *Clear Channel Assessment* (CCA) genannt. Die Dauer dieser Überprüfung (t_{CCA}) ist typischerweise für alle Sensorknoten gleich lang und beträgt standardmäßig für MICAz-Sensorknoten etwa $4ms$. Die Dauer zwischen zwei CCAs ist ein konfigurierbarer Parameter in TinyOS-LPL und wird in dieser Arbeit als Aufwachintervall (t_{LPL}) bezeichnet.

Will ein Sensorknoten ein Datenpaket übertragen, nutzt er hierfür die sogenannte Präambel, welche in TinyOS-LPL aus dem wiederholt unmittelbar hintereinander gesendeten Datenpaket besteht. Die Datenpakete werden maximal für die Dauer von t_{LPL} übertragen. Hiermit wird sichergestellt, dass die Funkschnittstelle des Empfängerknotens im Rahmen eines CCAs angeschaltet wird und eine Datenübertragung stattfinden kann. Der Empfängerknoten bestätigt den Empfang des Datenpakets mit einem ACK. Nach der erfolg-

[2]Carrier Sense Multiple Access/Collision Avoidance

reichen Übertragung wird die Funkschnittstelle beider Knoten nicht sofort abgeschaltet, vielmehr bleiben beide Funkschnittstellen für t_{Delay} ms für potenzielle Anschlussübertragungen angeschaltet. Standardmäßig beträgt die Dauer für $t_{Delay} = 100ms$.

Die verwendete Parametrisierung für t_{CCA}, t_{Delay} und t_{LPL} haben somit offensichtlich sowohl einen Einfluss auf den Energiebedarf der Datenübertragung als auch auf die Dauer der Übertragung. Die Parameter t_{CCA} und t_{LPL} beeinflussen, wie lange sich die Funkschnittstelle im angeschalteten Zustand befindet und somit einen Energiebedarf aufweist. Generell gilt, je größer t_{CCA} bzw. je kleiner t_{LPL}, desto länger befindet sich die Funkschnittstelle im angeschalteten Zustand. Dies gilt für alle im Sensornetz beteiligten Sensorknoten, unabhängig davon ob Datenpakete übertragen werden. Bei der Datenübertragung gilt, dass ein größerer Wert für t_{LPL} unter Umständen eine längere Präambel verursacht. Maximal kann die Präambel $t_{LPL}ms$ lang sein. Dies verursacht beim Senderknoten einen Energiebedarf aufgrund der angeschalteten Funkschnittstelle. Da sich die Funkschnittstelle der Sensorknoten nach einer erfolgreichen Übertragung für $t_{Delay}ms$ im angeschalteten Zustand befindet, verursacht dies zusätzlich bei beiden Sensorknoten einen erhöhten Energiebedarf.

Die Dauer einer erfolgreichen Übertragung mit TinyOS-LPL hängt von der Dauer der notwendigen Präambel ab. Da die Präambel maximal t_{LPL} ms übertragen werden muss, führen größere Werte für t_{LPL} im Durchschnitt zu einer längeren Übertragungsdauer.

S-MAC

Sensor-MAC [23] (S-MAC) ist ein Medienzugriffsverfahren mit einem synchronen Duty-Cycle. Dies bedeutet, dass sich die Funkschnittstellen benachbarter Sensorknoten möglichst gleichzeitig im angeschalteten Zustand (Wachphase) befinden. Bei asynchronen Medienzugriffsverfahren, wie beispielsweise TinyOS-LPL, ist die Funkschnittstelle verschiedener Sensorknoten unabhängig voneinander und nicht notwendigerweise gleichzeitig im angeschalteten Zustand.

Zur Synchronisation der Zustände der Funkschnittstelle verwendet S-MAC sogenannte Zeitpläne. Jeder Sensorknoten unterhält dabei einen eigenen Zeitplan, in dem die Wachphasen der benachbarten Sensorknoten verzeichnet sind. Die Erkennung von Nachbarknoten wird dabei innerhalb einer *Initialisierungsphase* durchgeführt, in der alle beteiligten Sensorknoten die Funkschnittstelle dauerhaft im angeschalteten Zustand behalten. Innerhalb der Initialisierungsphase werden die Zeitpläne der einzelnen Sensorknoten ausgetauscht. Jeder Sensorknoten versucht dabei, dem Zeitplan eines benachbarten

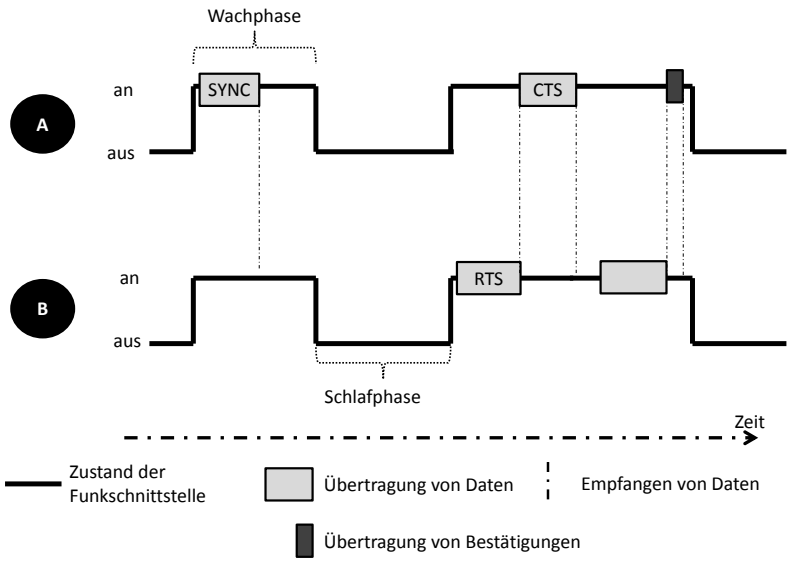


Abbildung 2.5 Datenübertragung zwischen zwei Sensorknoten A und B mit S-MAC.

Sensorknotens zu folgen und somit die Wachphase zum gleichen Zeitpunkt zu beginnen. Empfängt ein Sensorknoten innerhalb der Initialisierungsphase keinen Zeitplan so nimmt er an, der einzige Sensorknoten in Funkreichweite zu sein und erzeugt einen eigenen Zeitplan. Da Sensorknoten auch im laufenden Betrieb des Netzes ausfallen können, kann die Initialisierungsphase falls notwendig periodisch wiederholt werden. Mit dem in S-MAC verwendeten Synchronisierungsmechanismus der Zeitpläne kann es vorkommen, dass ein einzelner Sensorknoten mehreren Zeitplänen folgen muss. Dies ist insbesondere in Szenarien notwendig, in denen sich die Sensorknoten nicht alle innerhalb einer Funkreichweite befinden sondern vielmehr ein *Multi-Hop* Netzwerk bilden müssen.

Nach der initialen Synchronisierung der Zeitpläne beginnen die Sensorknoten mit dem Duty-Cycling. Jeder Sensorknoten führt demnach periodisch eine Wachphase gefolgt von einer Schlafphase aus. Sensorknoten, die dem gleichen Zeitplan folgen, schalten die Funkschnittstelle gleichzeitig an bzw. aus. Zur Aufrechterhaltung der Synchronisierung wird zu Beginn der Wachphase mit Hilfe sogenannter Synchronisationspakete (SYNC) eine Nachsynchronisierung der Sensorknoten erfolgen[3].

Innerhalb der Wachphase können, sofern notwendig, Daten zwischen den Sensorknoten ausgetauscht werden. Zur Vermeidung von Kollisionen bei der

[3]Bei den internen Uhren der Sensorknoten kann es zum sogenannten clock-drift kommen, d.h. das Zeitverständniss der Sensorknoten weicht von Sensorknoten zu Sensorknoten mit der Laufzeit voneinander ab.

Datenübertragung, sowie zur Verhinderung des Problems der versteckten Endgeräte, verwendet S-MAC das MACA-Verfahren [24]. Die Wachphase ist weiterhin in 2 Phasen geteilt: Innerhalb der ersten Phase wird die Synchronisierung sowie die Datenübertragung mit MACA ausgehandelt. Die zweite Phase wird durchgeführt, sofern Daten übertragen werden. Ist innerhalb der Wachphase keine Datenübetragung vereinbart, wird die Funkschnittstelle ausgeschaltet.

Der Ablauf einer Datenübertragung zwischen zwei Sensorknoten A und B mit S-MAC ist in Abbildung 2.5 zu sehen. Sensorknoten A und B haben sich innerhalb der Initialisierungsphase auf einen gemeinsamen Zeitplan verständigt, die Funkschnittstelle der Sensorknoten wird somit synchron an- bzw. ausgeschaltet. Sensorknoten A sendet periodisch innerhalb der Wachphase ein SYNC-Paket, um die Synchronisierung aufrecht zu erhalten. Will Sensorknoten B Daten an Sensorknoten A übertragen, so sendet er innerhalb der nächsten Wachphase eine RTS-Nachricht an Sensorknoten B. Sensorknoten A antwortet mit einer CTS-Nachricht. Im zweiten Teil der Wachphase findet daraufhin die Datenübertragung statt, welche von Sensorknoten A bestätigt wird. Im Anschluss wird die Funkschnittstelle synchron abgeschaltet.

Als Duty-Cycle Parameter kann in S-MAC der gewünschte Duty-Cycle angegeben werden. Der Duty-Cycle entspricht hierbei dem Verhältnis der Länge der Wachphase zu der anschließenden Schlafphase. Die in dieser Arbeit verwendete Implementierung von S-MAC berechnet aus dem angegebenen Duty-Cycle automatisch die Länge der Schlafphase, die Länge der Wachphase ist fix und so gewählt, dass sowohl die Synchronisierung als auch die Übertragung der Daten zuverlässig durchgeführt werden können.

Der tatsächlich durch S-MAC erreichte Duty-Cycle von S-MAC kann je nach Synchronisierung der Sensorknoten vom eingestellten Duty-Cycle abweichen: Folgt ein Sensorknoten mehreren Zeitplänen, so erhöht sich der Duty-Cycle der Knoten. Der eingestellte Duty-Cycle ist somit eine Art untere Schranke für die minimale Wachzeit eines Sensorknotens.

Die Dauer einer erfolgreichen Übertragung mit S-MAC hängt vom Zeitpunkt des Sendewunsches ab. Da Daten nur innerhalb der Wachphase übertragen werden können, muss der Sensorknoten unter Umständen Datenpakete zwischenspeichern, um diese in der nächsten Wachphase zu versenden.

TDMA-MAC

Im Rahmen dieser Arbeit wird ein sehr einfaches auf Zeitmultiplexing basierendes Medienzugriffsverfahren verwendet. Im folgenden wird es als TDMA-MAC Verfahren bezeichnet. Bei TDMA-MAC schalten die Sensorknoten periodisch und synchron die Funkschnittstelle für eine bestimmte Zeitspanne

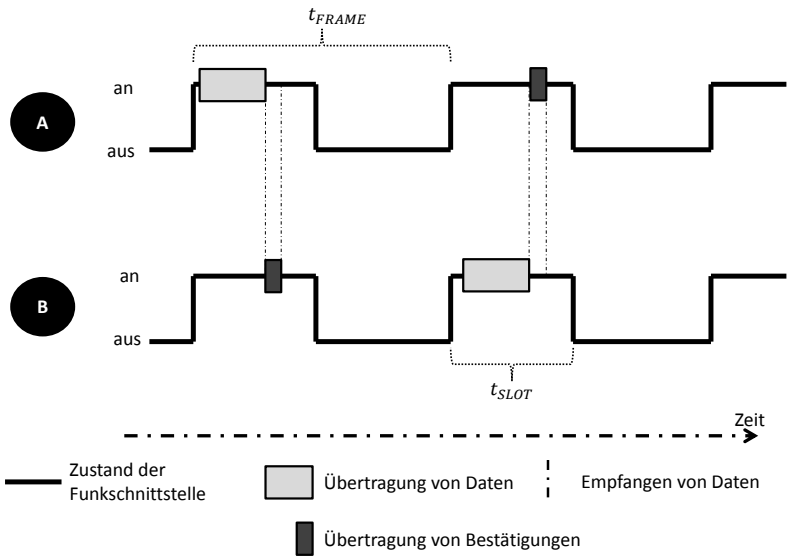


Abbildung 2.6 Datenübertragung zwischen zwei Sensorknoten A und B mit TDMA-MAC

(t_{SLOT}) an. Diese Zeitspanne ist für alle Sensorknoten gleich lang. Im Anschluss schalten alle Sensorknoten gleichzeitig die Funkschnittstelle aus. Die Periodenlänge wird als t_{FRAME} bezeichnet. Eine Synchronisierung der Wachphasen findet in TDMA-MAC nicht statt, in dieser Arbeit werden die Sensorknoten beim Betrieb mit TDMA-MAC gleichzeitig gestartet um eine initiale Synchronisierung der Wachphasen zu erhalten[4].

Der Ablauf einer Datenübertragung zwischen zwei Sensorknoten A und B mit TDMA-MAC ist in Abbildung 2.5 zu sehen. Die Datenübertragung innerhalb der Wachphase findet per CSMA-CA statt, der Empfang eines Datenpakets wird mit Hilfe eines ACKs bestätigt.

Wie bei S-MAC muss der Sensorknoten unter Umständen Datenpakete zwischenspeichern, um diese in der nächsten Wachphase zu versenden. Die Parametrisierung von TDMA-MAC hat somit wie S-MAC einen Einfluss auf die Übertragungsdauer der Datenpakete.

2.4 Werkzeuge zur Evaluierung des Energiebedarfs

Im Rahmen dieser Arbeit wird der Energiebedarf von Sensorknoten mit Hilfe von zwei Werkzeugen evaluiert. Für Experimente mit realer Hardware wie dem MICAz-Sensorknoten wird das Testbed SANDbed verwendet. Um si-

[4]In den Simulationen und den relativ kurzen Experimenten im SANDbed konnte der Einfluss des clock-drifts vernachlässigt werden, beim dauerhaften Betrieb über einen längeren Zeitraum ist eine externe Synchronisation der Wachphasen notwendig.

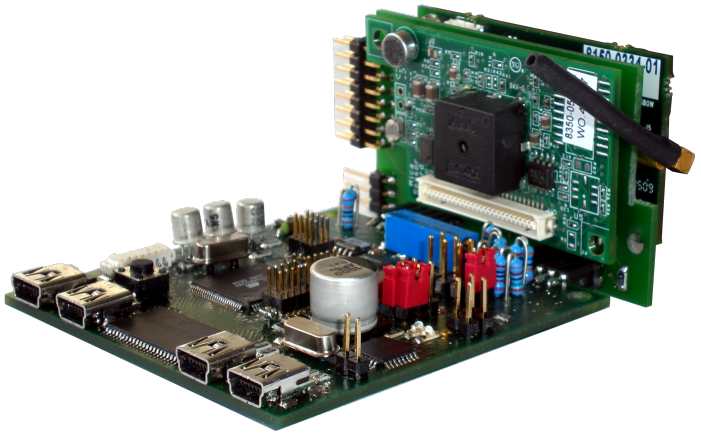


Abbildung 2.7 Ein SNMD mit einem angeschlossenen MICAz-Sensorknoten.

mulative Energiebedarfswerte zu ermitteln, wird AVRORA bzw. AVRORA+ eingesetzt.

2.4.1 Testbed SANDbed

Das Testbed SANDbed [5] [25] des Instituts für Telematik am KIT besteht zum Zeitpunkt der Erstellung dieser Arbeit aus 16 MICAz und 4 IRIS-Sensorknoten. Jeder dieser Sensorknoten ist an ein dediziertes Energiemessgerät, das *Sensor Node Management Devices* (SNMDs) [26] angeschlossen. Mit Hilfe der SNMDs ist eine Messung der Stromstärke I und der Spannung U der angeschlossenen Sensorknoten mit einer Samplingrate von bis zu 250 kHz möglich. Bei der Messung des Energiebedarfs wird jeweils der gesamte vom Sensorknoten benötigte Energiebedarf gemessen. Der Messfehler bei einer Energiebedarfsmessung mit den SNMDs liegt dabei im genutzten Messintervall von $0mA$ bis $120mA$ bei unter 1 %.

Abbildung 2.7 zeigt ein SNMD mit einem angeschlossenen MICAz-Sensorknoten. Jeweils bis zu sechs SNMDs sind über einen USB-Anschluss mit sogenannten Management Nodes (MNs) verbunden. Die Management Nodes dienen zur Steuerung der einzelnen SNMDs und zur Speicherung der Messergebnisse. Im SANDbed sind zum Zeitpunkt der Erstellung dieser Arbeit sechs MNs über eine TCP/IP-Anbindung erreichbar.

Für die Messung des Energiebedarfs stellen die SNMDs verschiedene Messmodi zur Verfügung: Neben einer dauerhaften Messung des Energiebedarfs über einen konfigurierbaren Zeitraum ist es mit den SNMDs möglich, bestimmte Anwendungsabschnitte isoliert zu vermessen. Hierfür stehen neben sogenannten *getriggerten Messungen* auch *Marker* zur Verfügung.

Mit getriggerten Messungen ist es möglich, den Start- und Endzeitpunkt der Energiemessung innerhalb der Sensornetzanwendung zu kennzeichnen. Die SNMDs messen daraufhin genau in diesem gekennzeichneten Zeitraum den Energiebedarf. Beispielsweise lässt sich somit der Energiebedarf einer Nachrichtenübertragung von Beginn der Übertragung bis zum Erhalt der Bestätigung vermessen. Die Nutzung von getriggerten Messungen verursacht dabei keinen zusätzlichen Overhead in Form von Wartezeit oder Energiebedarf.

Mit Markern ist es möglich, virtuelle Markierungen in die Messergebnisse der SNMDs einzufügen. Die Energiebedarfsmessung erfolgt somit über den definierten Messzeitraum, zusätzlich ist es jedoch möglich, innerhalb der Messergebnisse bestimmte Ereignisse oder Zeiträume zu kennzeichnen. Die Nutzung der Marker ist nicht komplett frei von Kosten. Für jeden Marker innerhalb des Anwendungscodes fällt ein Overhead von etwa 1,2 μJ an und verursacht eine Verzögerung der Anwendungsausführung von 30 Prozessorzyklen.

2.4.2 Simulationswerkzeug AVRORA

AVRORA [6] ist eine Mischung aus Simulations- und Emulationswerkzeug. Die Sensorknotenhardware, beispielsweise MICAz oder IRIS-Sensorknoten, werden dabei zyklengenau emuliert, das Funkmedium und somit das Kanalmodell werden simuliert. Ein Vorteil von AVRORA gegenüber anderen Simulationswerkzeugen ist die Möglichkeit, unmodifizierten, für die konkrete Sensorknotenplattform übersetzten Anwendungscode zu nutzen. Es ist somit möglich, die gleiche Binärdatei sowohl in SANDbed als auch in AVRORA zu nutzen.

Mit AEON [27] steht zudem ein Energiemodell für die emulierte Sensorknotenhardware zur Verfügung. In AvroraZ [28] wird beispielhaft ein Energiemodell für den MICAz Sensorknoten erstellt und evaluiert. Auf das Energiemodell wird im Rahmen des Kapitels 3 genauer eingegangen.

2.5 Sicherheit in Sensornetzen

Eine gute Übersicht über Sicherheitsmechanismen in drahtlosen Sensornetzen findet sich in [29]. Die Autoren stellen neben den klassischen Schutzzielen Vertraulichkeit, Integrität und Authentizität auch Herausforderungen vor, welche den Einsatz von etablierten Sicherheitsmechanismen und Protokollen erschwert bzw. verhindert. So ist die Ressourcenbeschränkung der Sensorknoten und hier insbesondere das limitierte Energiebudget die größte Herausforderung. Weiterhin argumentieren die Autoren, dass die Evaluation der Energie-Effizienz der Sicherheitsprotokolle eine noch nicht vollständig gelöste Herausforderung ist. Bei der Gegenüberstellung der unterschied-

lichen Mechanismen und Protokollen wird aufgezeigt, dass Protokolle basierend auf symmetrischer Kryptografie gegenüber asymmetrischer Kryptografie aufgrund der Ressourceneffizienz zu bevorzugen sind.

Symmetrische Kryptografie

Bei *symmetrischer Kryptografie* besitzen sowohl Sender- als auch Empfängerknoten den gleichen Schlüssel k um eine Nachricht zu ver- bzw. entschlüsseln. Die Verschlüsselung ist eine Funktion $E()$, deren Eingabeparameter der Schlüssel k und der Klartext der Nachricht p ist. Die Ausführung von $E_k(p)$ liefert den sogenannten Chiffretext c, in diesem Fall die verschlüsselte Nachricht. Es gilt somit:

$$E_k(p) = c$$

Für die Entschlüsselung existiert eine Funktion $D()$, für die gilt:

$$D_k(c) = p$$

Bei vielen symmetrischen Verfahren sind $E()$ und $D()$ die gleiche Funktion. In Sensornetzen häufig eingesetzte symmetrische Verfahren sind beispielsweise AES [30] und RC4/RC5 [31]. Ein Problem beim Einsatz von symmetrischer Kryptografie ist es, dass sowohl Sender- als auch Empfänger einen gemeinsamen, geheimen Schlüssel benötigen. Wird dieser Schlüssel offengelegt, ist die Vertraulichkeit der verschlüsselten Nachrichten nicht mehr gegeben. Um die symmetrischen Schlüssel auszutauschen, existieren verschiedene Schlüsselaustauschprotokolle. In einem Netz mit n Sensorknoten ist es notwendig, $n \cdot (n-1)$ Schlüssel auszutauschen, damit je zwischen zwei Sensorknoten ein symmetrischer Schlüssel existiert.

Asymmetrische Kryptografie

Bei *asymmetrischer Kryptografie*, auch *Public Key Kryptografie* genannt, besitzt jeder Sensorknoten einen öffentlichen Schlüssel (*pub*) und einen geheimen, privaten Schlüssel (*priv*).
Für alle möglichen Eingaben einer Nachricht p gilt:

$$D_{priv}(E_{pub}(p)) = p$$

Ein mit dem privaten Schlüssel verschlüsselte Nachricht muss somit mit dem dazugehörenden öffentlichen Schlüssel entschlüsselt werden. Um asymmetrische Kryptografie einzusetzen, verteilt ein Knoten A seinen öffentlichen

Schlüssel pub_A an die Kommunikationspartner, beispielsweise Sensorknoten B. Die Verschlüsselung einer Nachricht p erfolgt dann mit dem privaten Schlüssel $priv_A$:

$$E_{priv_A}(p) = c$$

Sensorknoten B kann mit dem vorher verteilten Schlüssel pub_A die Nachricht entschlüsseln:

$$D_{pub_A}(c) = p$$

Bei asymmetrischer Kryptografie ist es somit nicht notwendig, zwischen je zwei Kommunikationspartnern einen eigenen Schlüssel auszutauschen. Es genügt, lediglich den öffentlichen Schlüssel an die Kommunikationspartner zu übermitteln. Somit müssen bei n Sensorknoten in einem Netz auf jedem Sensorknoten $n - 1$ öffentliche Schlüssel der anderen Sensorknoten gespeichert werden. Bekannte asymmetrische Algorithmen sind RSA [32] und ECIES [33].

Ein Problem bei asymmetrischer Kryptografie ist es, dass die Berechnung von E und D um ein vielfaches länger dauern als bei symmetrischen Algorithmen. Damit einhergehend ist auch der Energiebedarf um ein vielfaches höher. Weiterhin ist die zur gleichen Sicherheit notwendige Schlüssellänge bei asymmetrischen Algorithmen im allgemeinen ebenfalls deutlich größer. Beispielsweise schlägt das National Institute of Standards and Technology (NIST) für die symmetrischen Algorithmen eine Schlüssellänge von mindestens 112 Bit, für asymmetrische Algorithmen basierend auf elliptischen Kurven mindestens 224 Bit und für RSA mindestens 2048 Bit vor.

Sicherheitsprotokolle für drahtlose Sensornetze

Es existieren verschiedene Arbeiten zum Schlüsselaustausch in Sensornetzen. Bei den meisten Arbeiten wird davon ausgegangen, dass ein Schlüsselaustausch basierend auf asymmetrischer Kryptografie aufgrund der Ressourcenbeschränkung der Sensorknoten nicht praktikabel ist. Aus diesem Grund werden häufig Random-Key Pre-Distribution Verfahren, also Verfahren die auf der zufälligen Verteilung der Schlüssel basieren. Das erste Verfahren für drahtlose Sensornetze wurde in [34] vorgestellt. Ähnliche Verfahren wurden daraufhin beispielsweise in [35] veröffentlicht. Eine grundsätzliche Idee dieser Verfahren ist es, *probabilistische* Sicherheit aufgrund der zufälligen Verteilung von Schlüsselmaterial zu bieten. Hierfür werden auf jedem Sensorknoten eine Teilmenge an symmetrischen Schlüsseln aus einem Schlüsselpool vor der Ausbringung der Sensorknoten vorverteilt. Mit einer berechenba-

ren Wahrscheinlichkeit können nun zwei Sensorknoten einen gemeinsamen Schlüssel berechnen, in dem sie gemeinsame Schlüssel aus dem Schlüsselpool finden und diese zur Erzeugung eines gemeinsamen Sitzungsschlüssels nutzen. Innerhalb der Evaluierung des Protokolls wird lediglich die Resistenz des Protokolls gegenüber Angriffen untersucht, eine Evaluierung des Energiebedarfs findet nicht statt.

Im Rahmen der Standardisierung von Kommunikationsprotokollen der Internet Engineering Task Force (IETF) finden sich unter dem Begriff *Internet der Dinge* verschiedene Protokollvorschläge, die auch für drahtlose Sensornetze geeignet sind bzw. dafür entwickelt werden. Innerhalb der 6LoWPAN (IPv6 over Low Power WPAN) Arbeitsgruppe existiert eine Sicherheitsanalyse der 6LoWPAN Protokollfamilie [36]. Bei der Analyse der existierenden Schlüsselaustauschprotokolle wird argumentiert, dass der Einsatz von Verfahren basierend auf asymmetrischer Kryptografie, beispielsweise der Diffie-Hellman Austausch, aufgrund der besseren Sicherheit gegenüber Kryptoanalyse wünschenswert sei. Alternativ wird der Einsatz von Verfahren basierend auf symmetrischer Kryptografie, beispielsweise Kerberos, vorgeschlagen. Eine offene Frage für beide Verfahren ist es, in wie weit der Einsatz dieser Protokolle auf den ressourcenschwachen Sensorknoten und mit dem limitierten Energiebudget möglich ist.

Der Energiebedarf von Schlüsselaustauschverfahren wird häufig mit Hilfe von theoretischen Modellen abgeschätzt [37], [38], [39]. Ein Vorteil von theoretischen Modellen im Vergleich zu Simulationen oder Experimenten im Testbed ist der vergleichsweise geringere Aufwand. Häufig kann auf eine konkrete Implementierung oder langwierige Experimente verzichtet werden. Ein Problem bei der Nutzung von theoretischen Modellen ist jedoch die Genauigkeit des so ermittelten Energiebedarfs. Sowohl eine vereinfachte Modellierung oder eine falsche Parametrisierung der Modellparameter kann dazu führen, dass der Energiebedarf im Vergleich zu Experimenten oder Simulationen große Abweichungen aufweist. Weiterhin wird in vielen Modellen von einem konkreten Medienzugriffsverfahren abstrahiert, ein Duty-Cycling der Funkschnittstelle wird nicht angenommen [37].

Das theoretische Modell von de Meulenaer et. al [38] vergleicht den Energiebedarf von Kerberos und ECDH-ECDSA unter Verwendung von TinyOS-LPL. Ein Ziel von de Meulenaer et. al. war neben der Bestimmung des konkreten Energiebedarfs der beiden Schlüsselaustauschprotokolle auch der Entwurf einer Methode, mit der der relative Energiebedarf der Kommunikation im Vergleich zum Energiebedarf der kryptografischen Berechnungen bestimmt werden kann. Das grundsätzliche Vorgehen zur Berechnung des Energiebedarfs der beiden Schlüsselaustauschprotokolle ist dabei wie folgt: Für

das zu untersuchende Protokoll wird gemessen, wie viel Zeit für die kryptografischen Berechnungen benötigt wird. Mit Hilfe eines Energiemodells des MICAz-Sensorknotens wird daraufhin aus der gemessenen Zeit der Energiebedarf der Berechnungen errechnet. Weiterhin werden die für den Schlüsselaustausch notwendigen Nachrichtenübertragungen und die zusätzlich notwendige Wartezeit der beteiligten Sensorknoten ermittelt. Wartezeit kann beispielsweise auftreten, wenn ein Sensorknoten auf eine Nachricht des anderen Sensorknotens warten muss, dieser aber zum Versand der Nachricht vorher kryptografische Berechnungen (z.B Verifikation eines Zertifikats) durchführen muss. Für die Übertragung der Nachrichten und das Duty-Cycling der Funkschnittstelle wird TinyOS-LPL als Medienzugriffsverfahren angenommen. Neben den reine Nachrichtenübertragungen muss somit zusätzlich der Overhead für die Synchronisierung der Funkschnittstelle berücksichtigt werden. Aus der Anzahl der zu übertragenen Nachrichten und der Wartezeit der Sensorknoten wird mit Hilfe des Energiemodells der Energiebedarf der Kommunikation berechnet. Ein Problem bei der Erstellung von theoretischen Modellen für den Energiebedarf ist die Parametrisierung der Modellbestandteile. Insbesondere beim Medienzugriffsverfahren und dem Kommunikationsverhalten der Sensorknoten ist auf eine korrekte Parametrisierung zu achten. Im theoretischen Modell wurde genau eine Parametrisierung von TinyOS-LPL untersucht. Ein Vergleich mit realen Messungen und Simulation in [9] hat gezeigt dass je nach Parametrisierung Seiteneffekte auf den Energiebedarf auftreten können, die in das theoretische Modell einfließen müssen. Beispielsweise sei hier der Einfluss der Timing-Effekte bei günstiger Wahl des Aufwachintervalls genannt, was zu einem unerwarteten, geringeren Energiebedarf führt. Dies wird in Kapitel 5 genauer erläutert. Je nach eingesetztem Betriebssystem oder Sensorknoten können sich noch weitere Seiteneffekte ergeben.

2.5.1 Sicherheit in Sensornetzen am Beispiel FleGSens

Das Projekt *FleGSens* (Sichere und flexible Grenz- und Liegenschaftsüberwachung durch drahtlose Sensornetze) beschäftigte sich mit dem Entwurf, der prototypischen Implementierung und Erprobung eines Sensornetzes zur Überwachung einer *grünen Grenze* [40][41]. Das Projekt wurde gemeinsam vom Institut für Telematik der Universität Lübeck, dem Institut für Telematik des Karlsruher Instituts für Technologie und der coalesenses GmbH durchgeführt. Auftraggeber des Projekts war das Bundesamt für Sicherheit in der Informationstechnologie (BSI).

Ziel des Projekts war es, mit Hilfe eines Sensornetzes und einfacher Sensorik, ein unberechtigtes Überqueren der vorab definierten grünen Grenzen zu detektieren und den Ort des Übertritts an einer zentralen Basisstation hinreichend genau und zeitnah anzuzeigen. Besonderer Fokus lag dabei auf der In-

formationssicherheit und der Robustheit gegenüber Ausfällen einzelner Sensorknoten. Insbesondere sollte das Sensornetz nicht durch Angriffe in seiner Funktionalität gestört werden können. Weiterhin war es notwendig, die Integrität und die Authentizität der übertragenen Daten, beispielsweise den Ort des Übertritts, zu schützen. Hierfür sollte mit Hilfe eines mehrstufigen Ansatzes ein sicheres Gesamtsystem entwickelt werden, welches am Projektende mit Hilfe eines aus 200 Sensorknoten bestehenden Demonstrators gezeigt werden sollte.

Innerhalb des Lastenhefts waren folgende funktionale Anforderungen an das Gesamtsystem gestellt worden:

- **Anzeige von Übertritten an einer Basisstation:** Die detektierten Übertritte sollten an einer zentralen Stelle, der Basisstation, angezeigt werden.
- **Anzeige des Orts des Übertritts auf** $10m$ **genau:** Der Ort des Übertritts sollte mit einer Genauigkeit von $10m$ an der Basisstation angezeigt werden.
- **Anzeige innerhalb von 5 Sekunden nach dem Übertritt:** Die Anzeige eines Übertritts sollte innerhalb von 5 Sekunden nach Beendigung des Übertritts angezeigt werden.
- **Lebenszeit des Netzes von mehr als** 7 **Tagen:** Das ausgebrachte Gesamtsystem sollte eine Lebenszeit von mehr als 7 Tagen haben.
- **Robustheit gegenüber** 10% **Knotenausfällen:** Die Detektion von Übertritten sollte auch mit bis zu 10% ausgefallenen Sensorknoten noch fehlerfrei funktionieren.
- **Absicherung des Netzes gegenüber Angriffen:** Das Gesamtsystem sollte gegen böswillige Angriffe abgesichert werden. Insbesondere standen hierbei Angriffe auf die Kommunikation im Vordergrund. Bei der Kommunikation innerhalb des Sensornetzes sollte die Integrität und die Authentizität der übertragenen Daten und der Kommunikationspartner gewährleistet sein.

Neben diesen funktionalen Anforderungen gab es noch weitere Anforderungen. So sollte das Gesamtsystem mit möglichst einfachen und kommerziell verfügbaren Sensorknoten und Komponenten umgesetzt werden.

Als Sensorknoten wurden deshalb die iSense-Sensorknoten der Firma coalesenses GmbH [42] eingesetzt. Der Sensorknoten umfasst einen Mikrocontroller und eine zu IEEE 802.15.4 kompatible Funkschnittstelle basierend auf dem

Hardwaremodul JN5139 der Firma Jennic [43]. Als Stromversorgung dient ein Lithium Ionen Akku. Insgesamt verfügt der Sensorknoten über 96 kByte RAM zur Speicherung der Anwendung und des Betriebssystems. Die Funkschnittstelle umfasst zudem einen kryptografischen Co-Prozessor zur Berechnung von AES. Als Sensorik wurde ein Passiv Infrarot Sensor (PIR) mit einer Reichweite von bis zu 10 Metern eingesetzt. Die Programmierung der Sensorknoten wurde mit Hilfe des iSense-Betriebssystems [44] durchgeführt.

Der Entwurf und die Implementierung des FleGSens-Gesamtsystems sollten in einem mehrstufigen Ansatz durchgeführt werden. In einem ersten Schritt sollte eine Sicherheitsanalyse der Anwendung durchgeführt werden. Ausgehend von einer Angreiferdefinition wurden unterschiedliche Angriffe und Sicherheitsmechanismen zur Abwehr dieser Gefahren untersucht. Anschließend wurde das FleGSens-Gesamtsystem entworfen. Neben dem eigentlichen Tracking Protokoll zur Erkennung von Übertritten wurden eine Weiterleitungsstrategie, ein Lokalisierungsprotokoll, ein Schlüsselaustauschprotokoll, ein Knotenausfallerkennungsprotokoll, ein Denial-of-Service Erkennungsprotokoll sowie ein Medienzugriffsverfahren mit Duty-Cycling entworfen und implementiert.

Das implementierte Gesamtsystem wurde anschließend innerhalb des Simulators Shawn [45] evaluiert. Ziel hierbei war es, die Funktionalität des Systems sowie die Auswirkungen unterschiedlicher Angriffe zu untersuchen. Das evaluierte System wurde anschließend innerhalb des Prototyps mit 200 Sensorknoten ausgebracht. Es wurde gezeigt, dass der Prototyp alle Anforderungen des Lastenheftes erfüllen konnte.

3. Evaluierung des Energiebedarfs von drahtlosen Sensornetzen mittels AVRORA+

Im Rahmen des FleGSens-Projekts wurde ein mehrstufiger Ansatz zum Entwurf des FleGSens-Gesamtsystems gewählt. Hierbei sollte insbesondere die Funktionalität des Gesamtsystems anhand von Simulation gezeigt werden, bevor das System auf echten Sensorknoten ausgebracht wurde. Eine der Herausforderung bei dem Entwurf des FleGSens-Gesamtsystem war die geforderte Mindestlaufzeit von bis zu 7 Tagen, da diese mit dem vorhandenen Energievorrat der eingesetzten Akkumulatoren nur durch Energiesparmaßnahmen, wie beispielsweise Duty-Cycling der Hardwarekomponenten, erreichbar war. Der in FleGSens genutzte Simulator Shawn [45] bot dabei nicht die Möglichkeit, den Energiebedarf des simulierten Gesamtsystems zu evaluieren. Dies hatte zur Folge, dass das Gesamtsystem bei ersten Testläufen auf echten Sensorknoten die geforderte Mindestlaufzeit aufgrund des Einsatzes eines unpassenden Medienzugriffsverfahrens nicht einhalten konnte. Es war somit notwendig, Anpassungen am Gesamtsystem und insbesondere beim Medienzugriffsverfahren durchzuführen. Der Energiebedarf des FleGSens-Gesamtsystems musste daraufhin aufwändig mit Hilfe von Experimenten evaluiert werden.

Es existieren verschiedene Werkzeuge zur Evaluierung des Energiebedarfs in drahtlosen Sensornetzen. Die höchste Genauigkeit lässt sich hierbei durch reale Messung aller beteiligten Sensorknoten, beispielsweise im Testbed

SANDbed erreichen. Jedoch zeigen sich bei diesem Ansatz auch einige Nachteile. Die Anzahl der im Testbed SANDbed verfügbaren Sensorknoten ist stark begrenzt, eine Evaluation von großen Sensornetzen mit mehreren hundert Sensorknoten lässt sich kaum durchführen bzw. ist sehr teuer. Experimente im Testbed sind zudem nur in Echtzeit möglich, die Anzahl der durchführbaren Messungen ist durch die Laufzeit der Experimente beschränkt. Eine Parallelisierung von Experimenten ist kaum möglich. Weiterhin sind Experimente im Testbed auf die dort verwendete Hard- und Software beschränkt. Simulationswerkzeuge bieten für die oben genannten Skalierungsprobleme geeignete Lösungsansätze. Zur Evaluierung des Energiebedarfs besitzen einige der eingesetzten Simulatoren, wie beispielsweise AVRORA [6], ein Energiemodell, mit dessen Hilfe der Energiebedarf der simulierten Knoten bestimmt werden kann. Der simulativ bestimmte Energiebedarf sollte dabei nach Möglichkeit *vergleichbar* mit dem Energiebedarf auf realen Sensorknoten sein, da sonst wie im Beispiel FleGSens im nachhinein Änderungen am simulierten Gesamtsystem notwendig sein könnten, da die Mindestlaufzeit der Anwendung nicht eingehalten werden kann.

Ziel dieses Kapitel ist es, die Genauigkeit der Evaluierung des Energiebedarfs mittels *Simulationen* in AVRORA im Vergleich zu Experimenten in SANDbed zu untersuchen. Mit Hilfe der Experimente werden Verbesserungen am Energiemodell in AVRORA zur Steigerung der Genauigkeit des simulierten Energiebedarfs identifiziert und in das Energiemodell von AVRORA+ integriert.

Das Kapitel gliedert sich wie folgt: Im folgenden Abschnitt werden die in diesem Kapitel verwendeten Begriffe definiert. Daraufhin werden Anforderungen an eine möglichst realitätsnahe und genaue Evaluierung des Energiebedarfs von Sensornetzen ermittelt. Hierbei wird aufgezeigt, warum die bisherigen Ansätze keine befriedigenden Lösungen für eine realitätsnahe Evaluierung darstellen und warum AVRORA als Simulationswerkzeug als Ausgangspunkt für die folgende Untersuchung ausgewählt wurde. Anschließend werden die im Energiemodell von AVRORA vorhandenen Hardwarekomponenten untersucht und die so ermittelten Energiebedarfswerte mit Ergebnissen aus SANDbed verglichen. Aufbauend auf diesen Ergebnissen werden verschiedene Verbesserungsmöglichkeiten aufgezeigt und die Umsetzung dieser in AVRORA+ beschrieben. Abschließend folgt ein Vergleich von AVRORA+ sowohl mit AVRORA als auch SANDbed.

AVRORA+ ist im Rahmen einer Diplomarbeit [46] entstanden, welche gemeinsam mit Joachim Wilke betreut wurde. Die Ergebnisse dieser Diplomarbeit wurden erweitert und in [7] veröffentlicht.

3.1 Begriffsdefinitionen

In diesem Kapitel werden folgende Begriffe verwendet:

- **Energiemodell von AVRORA**: Mit Hilfe von AVRORA kann die Hardware eines Sensorknotens emuliert werden. Hierbei wird für jede Hardwarekomponente taktgenau der Zustand der Komponente und Zustandsübergänge zwischen den einzelnen Zuständen emuliert. Zur Evaluierung des Energiebedarfs wird im Energiemodell von AVRORA jedem Zustand der Hardwarekomponenten eine zugehörige Stromstärke I zugeordnet. Es wird angenommen, dass die Spannung U des Sensorknotens über die Laufzeit konstant ist. Für jeden Taktzyklus wird die resultierende Stromstärke in einer Ausgabedatei protokolliert. Der Energiebedarf P über eine bestimmte Laufzeit t kann somit nach der Formel $P = U \cdot I \cdot t$ berechnet werden.

- **Testanwendung *TestAvrora***: Um einen Vergleich des Energiebedarfs bei Ausführung der gleichen Anwendung in AVRORA und SANDbed durchzuführen, wurde im Rahmen dieses Kapitels die Anwendung *TestAvrora* genutzt. Innerhalb der Testanwendung werden die Hardwarekomponenten Prozessor, Funkschnittstelle und die LEDs genutzt. Der Pseudocode von TestAvrora und eine beispielhafte Messung des Energiebedarfs finden sich im Abschnitt 3.4.

- **Mikrobenchmark**: Zur detaillierten Analyse des Energiebedarfs der einzelnen Hardwarekomponenten werden im Rahmen dieses Kapitels sogenannte Mikrobenchmarks eingesetzt. Ein Mikrobenchmark versetzt die zu untersuchende Hardwarekomponente gezielt in alle möglichen Zustände. Somit ist es möglich, den Energiebedarf jedes Zustands und der Zustandsübergänge zu bestimmen. Mikrobenchmarks werden in Abschnitt 3.5 genutzt.

- **Blink**: Als Mikrobenchmark für die LEDs wurde die TinyOS-Anwendung *Blink* verwendet, welche als Beispielanwendung mit dem Betriebssystem TinyOS mitgeliefert wird. Blink realisiert dabei einen Binärzähler für den Zahlenbereich $0 - 7$, wobei der binäre Zählerstand über die drei auf dem Sensorknoten verfügbaren LEDs angezeigt wird.

- **Lebenszeit:** Zur Bewertung der in AVRORA+ umgesetzten Verbesserungen wird die *Lebenszeit* eines Sensorknotens simuliert. Hierfür wird dem Sensorknoten ein festes Energiebudget zugewiesen. Die Anwendung auf dem Sensorknoten wird danach so lange ausgeführt, bis dieses Energiebudget aufgebraucht ist.

3.2 Anforderungen an eine simulative Evaluierung des Energiebedarfs

An einen Simulator zur Untersuchung des Energiebedarfs ergeben sich verschiedene Anforderungen, um eine möglichst gute Vergleichbarkeit zu realen Messungen zu erhalten:

- **A1 - Unterstützte Plattformen:** Der Simulator sollte eine breite Palette an realen Sensorknoten unterstützen, die auch in Testbeds zur Verfügung stehen.
- **A2 - Ausführung von Maschinencode:** Der gleiche Maschinencode sollte im Simulator und auf einem echten Sensorknoten ausgeführt werden. Dies schließt insbesondere ein, dass der Simulator unabhängig von dem für die Implementierung der Anwendung genutzten Sensornetz-Betriebssystem ist.
- **A3 - Energiemodell:** Der Simulator sollte ein eigenes Energiemodell besitzen, mit dessen Hilfe der Energiebedarf der Sensorknoten bestimmt werden kann.
- **A4 - Vergleichbarkeit des Energiebedarfs:** Der im Simulator ermittelte Energiebedarf sollte vergleichbar mit dem in der Realität gemessenen Energiebedarf sein. Hierbei ist gegebenenfalls zu untersuchen, welchen Einfluss die Umgebung (beispielsweise Temperatur oder Funkinterferenzen) auf den gemessenen Energiebedarf hat.

Der verwendete Simulator sollte möglichst weitverbreitete und in der Literatur häufig eingesetzte Sensorknotenplattformen unterstützen. Nur so ist gewährleistet, dass die simulierten Energiebedarfswerte vergleichbar mit Testbed Ergebnissen oder Messungen mit realer Hardware sind. Im Testbed SANDbed sind MICAz-Sensorknoten ausgebracht. Eine Vergleichbarkeit der Simulationsergebnisse zu den Testbedergebnissen ist somit gegeben, falls mit dem Simulator Energiebedarfsdaten des MICAz-Sensorknoten gewonnen werden können. Die am weitesten verbreiteten Sensorknotenplattformen, die als Quasi-Standard in der Literatur Verwendung finden, sind der MICAz [1] und der TelosB [47] Sensorknoten der Firma Memsic. Somit ist davon auszugehen, dass die Erkenntnisse aus der Analyse des Energiemodells der MICAz-Sensorknoten möglichst allgemeingültig und auch für andere Arbeiten relevant sind.

Weiterhin ist es notwendig, dass sowohl im Simulator als auch auf den Sensorknoten die gleiche Protokollimplementierung evaluiert werden kann. Ins-

Anforderung	Power-TOSSIM	OMNeT++	Castalia	Cooja	Shawn	Avrora
A1	Mica2, MICAz	⊖	⊖	TelosB, MICAz	iSense	Mica2, MICAz, TelosB
A2	⊖	⊖	⊖	⊕	⊖	⊕
A3	⊕	⊖	⊕	⊖	⊖	⊕
A4	⊖	⊖	⊖	⊖	⊖	⊖

Tabelle 3.1 Gegenüberstellung der in dieser Arbeit betrachteten Simulatoren. ⊕ und ⊖ stehen hierbei für eine Erfüllung bzw. Nicht-Erfüllung einer Anforderung.

besondere sollte demnach keine (Neu-)Implementierung der zu untersuchenden Protokolle in einer simulatorspezifischen Programmiersprache notwendig sein. Optimalerweise sollte die gleiche, übersetzte Protokollimplementierung sowohl im Simulator als auch auf echter Hardware evaluierbar sein. Jede Veränderung der simulierten Protokollimplementierung birgt die Gefahr einer Beeinflussung der Energiebedarfswerte durch Seiteneffekte. In Kapitel 5 werden beispielsweise Seiteneffekte durch das Duty-Cycling des Betriebssystems auf den Energiebedarf von Schlüsselaustauschprotokollen aufgezeigt, die unter Umständen in einer simulatorspezifischen Programmierung nicht aufgefallen wären.

Eine weitere Anforderung ist ein integriertes Energiemodell für die simulierte Sensorknotenplattform. Mit Hilfe des Energiemodells sollte es möglich sein, Aussagen sowohl über den Gesamtenergiebedarf der simulierten Protokolle, als auch über den Energiebedarf der verwendeten Hardwarekomponenten zu formulieren. Eine weitere wichtige Anforderung ist die Genauigkeit des ermittelten Energiebedarfs. Nach Möglichkeit sollte der Energiebedarf der Simulation mit realen Energiebedarfswerten vergleichbar sein. Hierzu ist eine Evaluation bzw. ein Vergleich des Energiemodells mit dem Energiebedarf echter Sensorknoten notwendig.

3.2.1 Stand der Forschung und verwandte Arbeiten

In diesem Abschnitt werden verschiedene, in verwandten Arbeiten häufig genutzte Simulationswerkzeuge für drahtlose Sensornetze vorgestellt. Ein besonderer Gegenstand der Untersuchung ist dabei die Vergleichbarkeit der simulativ gewonnen Energiebedarfsdaten zu Messergebnissen von realen Sen-

sorknoten. Tabelle 3.1 zeigt eine Übersicht über die hier diskutierten Simulationswerkzeuge sowie eine Bewertung hinsichtlich der im vorigen Abschnitt definierten Anforderungen A1 bis A4 an einen Simulator.

TOSSIM [48] ist ein Simulator für das Sensornetz Betriebssystem TinyOS. TOSSIM unterstützt verschiedene Sensorknotenplattformen wie Mica2 und MICAz-Sensorknoten. PowerTOSSIM [49] bzw. PowerTOSSIMz [50] sind Erweiterungen von TOSSIM, welche TOSSIM um ein Energiemodell für Mica2 und MICAz-Sensorknoten erweitern. Somit ist Anforderung A1 erfüllt. Bei der Verwendung von TOSSIM und PowerTOSSIM als Simulationsumgebung ist es jedoch notwendig, den Anwendungscode in TOSSIM-spezifischen Maschinencode zu übersetzen. Bei der Übersetzung werden automatisch die Medienzugriffsschicht und der physikalische Zugriff auf Hardwarekomponenten durch TOSSIM-spezifische Implementierungen ersetzt. Somit muss für jedes Medienzugriffsverfahren eine eigene, TOSSIM-spezifische Implementierung angefertigt werden. Eine direkte Vergleichbarkeit der Ergebnisse von Simulation und realen Messergebnissen ist daher schwierig, da nicht ausgeschlossen werden kann, dass Einflussfaktoren auf den Energiebedarf bei der Ausführung auf echter Hardware existieren, die nicht in TOSSIM abgebildet werden. Weiterhin können mit TOSSIM und PowerTOSSIM nur Anwendungen evaluiert werden, die im Betriebssystem TinyOS geschrieben wurden. Somit wird Anforderung A2 nicht von TOSSIM oder PowerTOSSIM erfüllt.

Das Energiemodell in PowerTOSSIM wurde mit Messwerten der Mica2 Sensorknotenplattform parametrisiert, eine Evaluierung der Genauigkeit [49] des simulierten Energiebedarfs im Vergleich zum Gemessenen ergab einen Unterschied von bis zu 10%. Für das Energiemodell der MICAz-Sensorknotenplattform existieren keine Angaben über die Genauigkeit des Energiebedarfs. Die Anforderung A3 wird daher als erfüllt, Eigenschaft A4 jedoch als nicht erfüllt bewertet.

OMNeT++ [51] ist ein diskreter, allgemein einsetzbarer Netzwerksimulator, der mit dem MiXiM Framework [52] um eine Unterstützung von drahtlosen und mobilen Systemen erweitert wurde. Weiterhin existieren für OMNeT++ Erweiterungen für ein realistisches Kanalmodell für drahtlose Sensornetze [53], dessen Vergleichbarkeit zu realen Messungen belegt wurde. Um Anwendungen für drahtlose Sensornetze mit OMNeT++ zu evaluieren ist es notwendig, eine OMNeT-spezifische Implementierung der Anwendung zu erstellen. Es ist somit nicht möglich, die gleiche Anwendungsimplementierung und den gleichen Maschinencode in OMNeT++ und auf einem realen Sensorknoten zu evaluieren. Weiterhin abstrahiert OMNeT++ von konkreten Sensorknotenplattformen, eine Unterstützung von verschiedenen Platt-

formen ist somit nicht gegeben. Anforderungen A1 und A2 können somit von OMNeT++ nicht erfüllt werden.

Mit [54] existiert ein Energiemodell für OMNeT++ auf Basis des Mica2 Sensorknotens. Die Genauigkeit des Energiemodells ist jedoch nicht mit realen Messungen belegt. Es ist somit fraglich, ob die mit dem Energiemodell gewonnen Energiebedarfswerte quantitativ und qualitativ mit Messergebnissen vergleichbar sind. Die Anforderungen A3 und A4 werden somit als nicht erfüllt bewertet.

Castalia [55] ist ein Open-Source Simulator für drahtlose Sensornetze, welcher aufbauend auf OMNeT++ entwickelt wurde. Somit ist wie bei OMNeT++ keine Möglichkeit gegeben, den gleichen Maschinencode in Castalia und auf realen Sensorknoten auszuführen. Dementsprechenden werden die Anforderungen A1 und A2 als nicht erfüllt bewertet. Castalia bietet jedoch die Möglichkeit, den Energiebedarf der simulierten Anwendung zu bestimmen. Die Parametrisierung des Energiemodells basiert dabei auf unterschiedlichen Messergebnissen von typischen Hardwarekomponenten, so sind beispielsweise Energiebedarfswerte des CC2420-Funktransceivers enthalten. Eine Evaluierung der Genauigkeit des Energiemodells ist nicht gegeben. Somit wird Anforderung A3 als erfüllt, Anforderung A4 jedoch als nicht erfüllt bewertet.

Cooja [56] ist ein Java-basiertes Simulationswerkzeug für das Sensornetzbetriebssystem ContikiOS [57]. Cooja unterstützt verschiedene Hardwareplattformen, unter anderem auch MICAz und TelosB Sensorknoten. In Verbindung mit MSPSim [58] ist es möglich, übersetzten Maschinencode für MSP-basierte Sensorknoten wie dem TelosB Sensorknoten zu simulieren. Die Anforderungen A1 und A2 sind somit als erfüllt anzusehen. Für MSPSim und Cooja existiert jedoch kein eigenes Energiemodell, eine Bestimmung des Energiebedarfs der Sensorknoten ist nicht möglich. Somit sind die Anforderungen A3 und A4 als nicht erfüllt bewertet.

Shawn [45] ist ein Simulator zur Evaluierung von Anwendungsprotokollen für drahtlose Sensornetze. Ziel bei der Entwicklung von Shawn war es, Algorithmen auf einer möglichst hohen Abstraktionsebene und in möglichst großen simulierten Netzen in möglichst geringer Zeit evaluieren zu können. Shawn erreicht dies, in dem von vielen realen Phänomenen, beispielsweise Kollisionen auf Medienzugriffsschicht, abstrahiert wird und nur noch der Effekt (Nachrichteneinheit geht verloren) simuliert wird. Hierdurch soll im Vergleich zu beispielsweise OMNeT++ eine geringe Simulationszeit und größere simulierte Netze erreicht werden. Shawn abstrahiert somit von der eigentlichen Sensorknotenplattform. In Shawn ist es aber möglich Anwendungscode, der für iSense Sensorknoten geschrieben wurde, zu simulieren. Allerdings muss der Anwendungscode ähnlich wie bei TOSSIM speziell für den

Simulator übersetzt werden. Shawn erfüllt somit die Anforderung A2 nicht. In Shawn existiert kein Energiemodell für die Sensorknoten, somit sind die Anforderungen A3 und A4 ebenfalls nicht erfüllt.

Mit Hilfe von **AVRORA** [6] ist es möglich, Sensorknoten auf Hardwareebene zu emulieren. Somit ist es nicht notwendig, Anwendungsimplementierungen erneut zu übersetzen, der gleiche Maschinencode kann sowohl in AVRORA als auch auf realen Sensorknoten genutzt werden. AVRORA unterstützt dabei mehrere typische Sensorknotenplattformen wie beispielsweise die Mica2 und MICAz-Sensorknoten. Weiterhin wurde in [28] ein Energiemodell für MICAz-Sensorknoten in AVRORA integriert. Im Energiemodell von AVRORA werden die Hardwarekomponenten Mikrokontroller, Transceiver sowie die Light Emitting Diodes (LEDs) emuliert und jedem der vorkommenden Hardwarezuständen ein Energiebedarf gegenübergestellt. Der so ermittelte Energiebedarf ist damit grundsätzlich vergleichbar mit gemessenen Energiebedarfswerten, jedoch existiert bisher kein systematischer Vergleich des Energiebedarfs mit Messergebnissen mehrerer MICAz-Sensorknoten. AVRORA erfüllt somit die Anforderungen A1 bis A3. Die Ergebnisse im weiteren Verlauf dieses Kapitels zeigen, dass die mit AVRORA gewonnen Energiebedarfswerte im Vergleich zu Messergebnissen mit realen Sensorknoten quantitative Unterschiede aufweisen. Daher ist Anforderung A4 nicht erfüllt.

Aus dem Vergleich der verschiedenen Simulatoren wird deutlich, dass AVRORA ein geeignetes Simulationswerkzeug für die Ermittlung des Energiebedarfs für MICAz-Sensorknoten ist. AVRORA erfüllt alle Anforderungen bis auf die Vergleichbarkeit des Energiebedarfs mit realen Messungen im Testbed. Ausgehend von dieser Erkenntnis wird in den folgenden Kapiteln die Genauigkeit des Energiemodells von AVRORA untersucht.

3.3 Verbesserung des Energiemodells in AVRORA

Bei der Untersuchung des Energiemodells in AVRORA konnten einige Verbesserungsmöglichkeiten herausgearbeitet werden, die zum Entwurf von AVRORA+ führten. Im Einzelnen sind dies die *Kalibrierung des Energiemodells*, das *Hinzufügen von Hardwarevarianzen* und das *Hinzufügen der Kosten für Zustandsübergänge*:

1. **Kalibrierung des Energiemodells:** Es existiert ein systematischer Offset zwischen realen Energiebedarfsdaten und den in AVRORA ermittelten Werten. Die Werte in AVRORA sind grundsätzlich immer zu niedrig angesetzt. Daher wurde das Energiemodell in AVRORA+ neu kalibriert und an die in SANDbed gemessenen Werte angepasst.

2. **Hinzufügen von Hardwarevarianzen:** Es existieren signifikante Hardwarevarianzen bezogen auf den Energiebedarf. Unterschiedliche Sensorknoten des gleichen Typs MICAz weisen bei der Ausführung der gleichen Anwendung Varianzen bis zu 20% auf, welche nicht im Energiemodell in AVRORA abgebildet sind. Diese Hardwarevarianzen wurden in AVRORA+ hinzugefügt.

3. **Hinzufügen der Kosten für Zustandsübergänge:** Zustandsübergänge der Hardwarekomponenten in AVRORA sind nicht mit Energiekosten verbunden. Die Untersuchung auf realen Sensorknoten zeigt jedoch, dass Zustandsübergänge zusätzlichen Energiebedarf verursachen, der bisher nicht in AVRORA abgebildet wird. AVRORA+ bildet die Kosten der Zustandsübergänge im Energiemodell des MICAz-Sensorknoten ab.

Bewertung der Verbesserungen

Die Relevanz der einzelnen Verbesserungen im Energiemodell von AVRORA+ soll anhand eines einfachen Anwendungsbeispiels erläutert werden. Der Pseudocode der verwendeten TinyOS-Anwendung ist in Algorithmus 1 abgebildet.

Algorithmus 1 : Pseudocode der Anwendung zur Ermittlung der Lebenszeit eines Sensorknotens bei der Verwendung von TinyOS-LPL und einem festen Energiebudget von 50 Joule.

Eingabe : Periodischer Timer mit Intervall 10 s
1 **wenn** *Timer abgelaufen* **dann**
2 Broadcaste Nachrichteneinheit der Größe 1 Byte;
3 **Ende**

Im Anwendungsbeispiel sendet ein Sensorknoten periodisch alle 10 Sekunden eine Nachrichteneinheit der Größe 1 Byte per Broadcast. Als Medienzugriffsverfahren kommt TinyOS-LPL mit einer Aufwachzeit von $t_{LPL} = 100ms$ zum Einsatz. Hiermit soll gewährleistet werden, dass sowohl für die Funkschnittstelle als auch für den Mikrocontroller periodisch Zustandsübergänge ausgelöst werden. Zustandsübergänge werden durch das An- und Ausschalten der Funkschnittstelle, der Übertragung der Nachrichten und durch das Ablaufen des Timers verursacht. Insgesamt werden somit alle im Rahmen dieses Kapitels untersuchten Zustandsübergänge der Funkschnittstelle und des Mikrocontrollers genutzt.

Die Anwendung wird mit Hilfe von AVRORA und AVRORA+ simuliert. Die Simulation wird dabei so parametrisiert, dass der Sensorknoten ein Energiebudget von 50 Joule besitzt. Ist das Energiebudget aufgebraucht, unterbricht der Simulator die Simulation und gibt die Lebenszeit des Sensorknotens bis

Simulator	Durchschnittliche Lebenszeit	Minimale Lebenszeit	Maximale Lebenszeit
Avrora	20138,50 s	20138,50 s	20138,50 s
Avrora+	14937,80 s	12221,22 s	16734,69 s

Tabelle 3.2 Ergebnisse der Simulation der Lebenszeit bei der Verwendung eines festen Energiebudgets von 50 Joule.

Simulator	Energiebedarf der Zustandsübergänge
Avrora	0
Avrora+	1,399 J

Tabelle 3.3 Energiebedarf der Zustandsübergänge.

zur Erschöpfung des Energiebudgets aus. Die Simulationen wurden für beide Simulatoren 100-mal wiederholt.

Tabelle 3.2 fasst die mit AVRORA und AVRORA+ simulierten Lebenszeiten des Sensorknotens zusammen. Die durchschnittliche Lebenszeit in AVRORA beträgt $20138,50s$, in AVRORA+ wurde eine durchschnittliche Lebenszeit von $14937,80s$ gemessen. Neben der durchschnittlichen Lebenszeit wurde zusätzlich noch die minimal und maximal erreichbare Lebenszeit gemessen. In AVRORA+ betrug die minimale Lebenszeit des Sensorknotens $12221,22s$, die maximale Lebenszeit $16734,69s$. Da AVRORA keine Varianzen abbildet, ist die Lebenszeit in allen Simulationen gleich lang.

Neben der Lebenszeit des Sensorknotens wurde in den Simulationen mit AVRORA+ auch untersucht, wie viel Energie für die Zustandsübergänge, welche im Energiemodell von AVRORA nicht abgebildet sind, aufgebracht werden muss. Tabelle 3.3 fasst den ermittelten Energiebedarf für die Zustandsübergänge zusammen. Insgesamt wurden in den hier durchgeführten Simulationen mit AVRORA+ $1,399J$ der verfügbaren $50J$ für Zustandsübergänge aufgewendet. Da das Energiemodell in AVRORA keine Zustandsübergänge abbildet, wird auch keine Energie dafür aufgewendet.

Die *ungenaue Kalibrierung des Energiemodells* in AVRORA führt beim hier durchgeführten Vergleich der durchschnittlichen Lebenszeit der Sensorknoten dazu, dass ein Sensorknoten in AVRORA eine im Vergleich zu Simulationen in AVRORA+ um etwa 35% längere Lebenszeit aufweist. Wird beispielsweise im Rahmen einer Anwendungsentwicklung AVRORA zur Bestimmung des

Energiebedarfs verwendet ist davon auszugehen, dass eine Ausführung der Anwendung auf realen Sensorknoten, beispielsweise in SANDbed, zu deutlich kürzeren Lebenszeiten führt.

Bei den in SANDbed verwendeten MICAz-Sensorknoten konnten *Hardwarevarianzen* im Bezug zum Energiebedarf festgestellt werden. In den hier durchgeführten Evaluationen konnten diese Hardwarevarianzen anhand der unterschiedlichen minimalen und maximalen Lebenszeit des simulierten Sensorknotens festgestellt werden. So ist die maximale Lebenszeit etwa 37% länger als die minimale Lebenszeit. Wird eine Anwendung auf mehreren Sensorknoten ausgeführt ist somit davon auszugehen, dass einige Sensorknoten aufgrund der Hardwarevarianzen zu einem deutlich früheren Zeitpunkt ausfallen als andere.

Das Energiemodell in AVRORA enthält keine *Kosten für Zustandsübergänge* der Hardwarekomponenten. Reale Messungen zeigen jedoch, dass Zustandsübergänge mit zusätzlichen Energiekosten verbunden sind. Bei der Simulation der Lebenszeit, vergleiche Tabelle 3.2, wird etwa 3% des Energiebudgets nur für Zustandsübergänge verwendet. Die Anzahl der Zustandsübergänge innerhalb einer Anwendung ist anwendungsabhängig. Es ist daher schwierig zu verallgemeinern, welcher Anteil des Energiebudgets eines Sensorknotens für die Zustandsübergänge aufgebracht werden muss. Generell ist davon auszugehen, dass in sehr dynamischen Anwendungen mit vielen Schlafphasen, beispielsweise bedingt durch das Duty-Cycling der Funkschnittstelle, mehr Zustandsübergänge auftreten als in weniger dynamischen Anwendungen.

Da der Einfluss der einzelnen Verbesserungen stark von der Anwendung abhängt ist es schwierig, allgemeingültige Aussagen über die Auswirkungen auf den Energiebedarf zu formulieren. Insgesamt konnte mit der hier verwendeten Anwendung der Einfluss der unterschiedlichen Verbesserungen des Energiemodells exemplarisch aufgezeigt werden. Den größten Einfluss auf die Lebenszeit des Sensorknotens haben die ungenaue Kalibrierung des Energiemodells und die Hardwarevarianzen. Ein geringerer Einfluss konnte für das Fehlen der Zustandsübergänge im Energiemodell beziffert werden.

In den folgenden Abschnitten werden die Messungen vorgestellt, auf deren Basis die oben genannten Verbesserungen des Energiemodells in AVRORA+ gewonnen wurden.

Algorithmus 2 : *transmitPacket()*: Nach dem Start des Funktransceivers angestoßene Prozedur auf Sensorknoten B-D.

```
1  Sende Nachricht der Größe 100 Byte per Unicast an Sensorknoten A;
2  wenn Übertragung erfolgreich dann
3     Schalte rote LED an;
4  Ende
```

Algorithmus 3 : *receivePacket()*: Nach dem Empfang einer Nachricht angestoßene Prozedur auf Sensorknoten A-D.

```
1  wenn Empfang einer Unicast Nachricht dann
2     Versende Bestätigung;
3     Schalte gelbe LED an;
4  Ende
5  wenn Sensorknoten A dann
6     wenn Nachricht von Sensorknoten B-D empfangen dann
7        Sende Nachricht per Broadcast;
8        Schalte grüne LED an;
9     Ende
10 Ende
```

3.4 Analyse des Energiemodells von AVRORA mit *TestAvrora*

In diesem Abschnitt wird eine Analyse und ein Vergleich des Energiemodells von AVRORA und Messungen des Energiebedarfs in SANDbed anhand der Testanwendung *TestAvrora* durchgeführt.

Beschreibung der Testanwendung TestAvrora

Das Energiemodell in AVRORA beinhaltet Stromstärke und Spannung für die Komponenten Mikrokontroller, CC2420 Funktransceiver und die drei verbauten Light Emitting Diodes (LEDs). Innerhalb von TestAvrora werden daher diese Hardwarekomponenten genutzt, um simulative Energiebedarfswerte zu gewinnen und diese mit Messungen in SANDbed zu vergleichen.

Abbildung 3.1 zeigt den zeitlichen Ablauf von TestAvrora. Insgesamt werden vier Sensorknoten genutzt, diese werden im folgenden mit Sensorknoten A-D bezeichnet. In [8] wurde nachgewiesen, dass der Energiebedarf der MICAz-Sensorknoten nach dem Einschalten des Sensorknotens starken Schwankungen unterworfen ist. Um verlässliche und über den Messzeitraum nicht schwankende Messwerte zu erhalten ist es notwendig, Energiebedarfsmessungen erst 20 Sekunden nach dem Anschalten des MICAz durchzuführen, wie es deshalb auch im Rahmen dieser Arbeit bei allen Messungen in SAND-

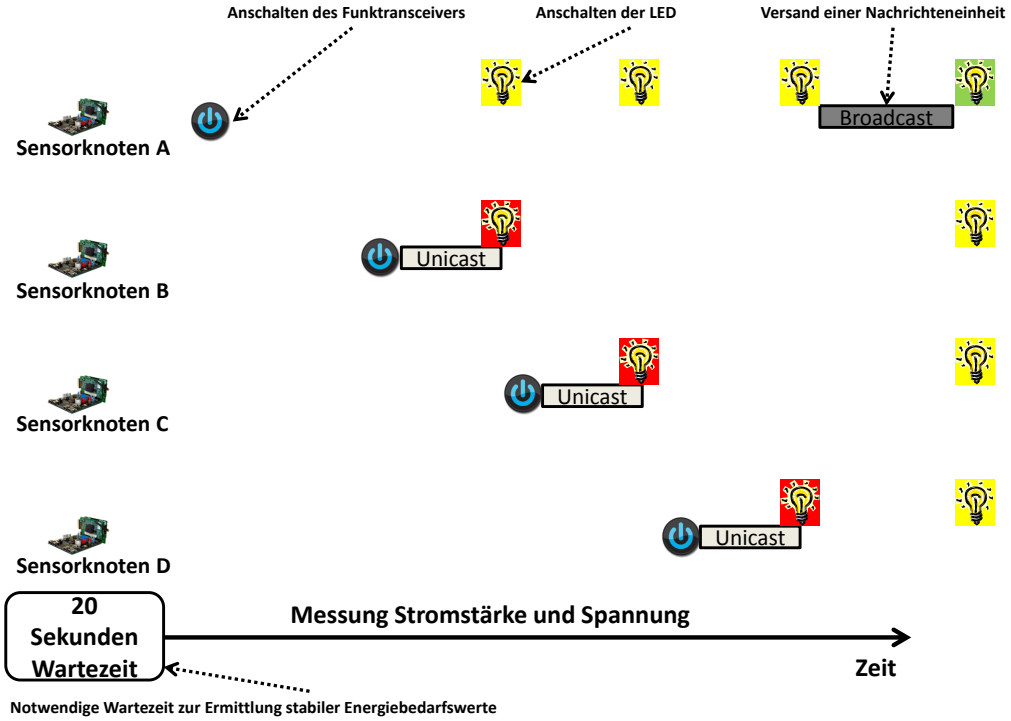


Abbildung 3.1 Zeitlicher Ablauf bei der Ausführung von TestAvrora.

bed durchgeführt wird. Die Wartezeit ist entsprechend in Abbildung 3.1 eingezeichnet. Zu Beginn sind die Transceiver der Sensorknoten noch ausgeschaltet. Nach 800*ms* wird der Transceiver auf Sensorknoten A angeschaltet und wechselt somit in einen Zustand, in dem er Nachrichten empfangen kann. Sensorknoten B bis D schalten jeweils zu den Zeitpunkten 900*ms*, 1000*ms* und 1100*ms* den Transceiver an. Auf den Sensorknoten B-D wird nach dem erfolgreichen Anschalten des Funktransceivers die Prozedur *transmitPacket()* angestoßen, deren Pseudocode in Algorithmus 2 dargestellt ist. Innerhalb von transmitPacket() wird genau eine Nachricht der Größe 100 Byte an Sensorknoten A übertragen. Ist die Übertragung erfolgreich und eine Bestätigung wurde empfangen, so wird die rote LED angeschaltet.

Empfängt ein Sensorknoten eine Nachricht, so wird die Prozedur *receivePacket()* angestoßen, deren Pseudocode in Algorithmus 3 dargestellt ist. Wurde eine Unicast Nachricht empfangen, so versendet der Sensorknoten eine Bestätigung an den Senderknoten. Daraufhin wird zur Signalisierung des Nachrichtenempfangs die gelbe LED angeschaltet. Ist der empfangende Sensorknoten der Sensorknoten A und hat Sensorknoten A bereits Nachrichten der Sensorknoten B-D empfangen, so versendet er eine Nachricht von 100 Byte per Broadcast und signalisiert dies mit dem Anschalten der grünen LED.

Ziel bei der Erstellung von TestAvrora war es nicht, eine Anwendung für drahtlose Sensornetze zu entwickeln, wie sie in einem konkreten Einsatzszenario benötigt werden könnte. Vielmehr war es das Ziel, eine möglichst einfache Anwendung zu entwerfen, die alle im Energiemodell von AVRORA enthaltenen Hardwarekomponenten in einem möglichst anschaulichen Beispiel verwendet.

Versuchsparameter

Parameter	Wert
Simulator	AVRORA
Knotenplattform	MICAz
Betriebssystem	TinyOS
Versuchswiederholungen	4 x 20
Versuchsdauer	2 Sekunden
Messrate des SNMD	100kHz
Gemessene Parameter	Stromstärke I und Spannung U
Genutzte Sensorknoten	2007, 2018, 2034, 2037
MAC Protokoll	IEEE 802.15.4 im Non-Beacon Mode
Kanal	22
Sendeleistung	31
Größe einer Nachricht	jeweils 100 Byte

Tabelle 3.4 Versuchsparameter in SANDbed.

Eine Übersicht über die relevanten Versuchsparameter ist in Tabelle 3.4 zu sehen. Die Versuche werden sowohl in AVRORA als auch in SANDbed mit der gleichen Implementierung und dem gleichen Maschinencode von TestAvrora durchgeführt. Der MICAz-Sensorknoten kommt als Sensorknotenplattform zum Einsatz. Als Betriebssystem der Sensorknoten wird TinyOS eingesetzt. Insgesamt werden für die Ausführung von TestAvrora vier MICAz-Sensorknoten im SANDbed am Mangament Node 2 (MN2) eingesetzt (Sensorknoten A bis D).

Der Versuch wird in 4 Messreihen jeweils 20 mal sowohl im Testbed als auch in Avrora wiederholt. Nach jeder Messreihe werden die Knotenrollen gewechselt, so dass jeder der vier Sensorknoten als Sensorknoten A fungiert.

Die Messung der Stromstärke und der Spannung der eingesetzten Sensorknoten mit Hilfe der SNMDs findet mit 100 kHz statt. In jedem Versuch werden sowohl Stromstärke und Spannung über einen Zeitraum von 2 Sekunden gemessen. Als Medienzugriffsprotokoll der MICAz-Sensorknoten wird IEEE 802.15.4 [16] im Non-Beacon Mode verwendet, wie es standardmäßig in TinyOS implementiert ist. Da der CC2420-Funktransceiver der genutzten MICAz-Sensorknoten Daten sowohl per Unicast als auch per Broadcast versenden kann, werden in der Testanwendung beide Arten der Adressierung verwendet. Der Messzeitraum für Stromstärke I und Spannung U beträgt 2 Sekunden, ein einzelner Versuch dauert somit zusammen mit der Wartezeit insgesamt 22 Sekunden.

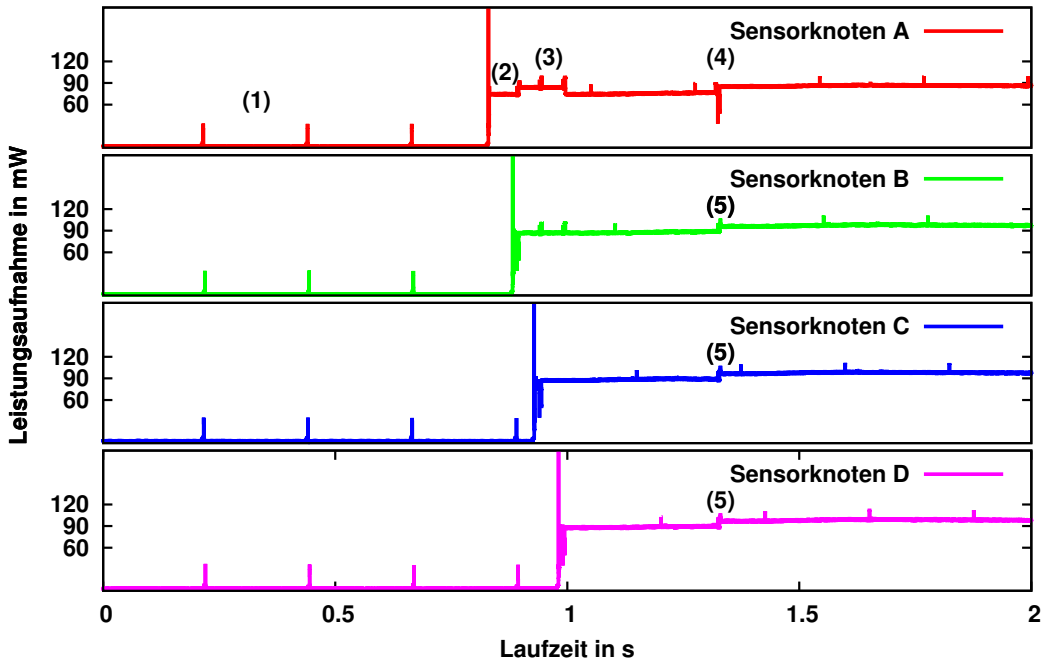
Beispielhafte Messung der Leistungsaufnahme in SANDbed

Eine beispielhafte Messung der Leistungsaufnahme der vier beteiligten MICAz-Sensorknoten ist in Abbildung 3.2a zu sehen. Dargestellt ist hierbei die Leistung in mW während der Laufzeit eines Versuchs für die Sensorknoten A-D. Zu Beginn des Experiments ist der Transceiver ausgeschaltet (Markierung (1)), was zu einer Leistung von etwa $4mW$ führt. Nach $800ms$ wird der Transceiver des Sensorknotens A angeschaltet, dieser wechselt daraufhin in einen empfangsbereiten Zustand (Markierung (2)). Zu erkennen ist hierbei der starke Anstieg der Leistung auf etwa $67mW$, verursacht lediglich durch das Einschalten des Transceivers.

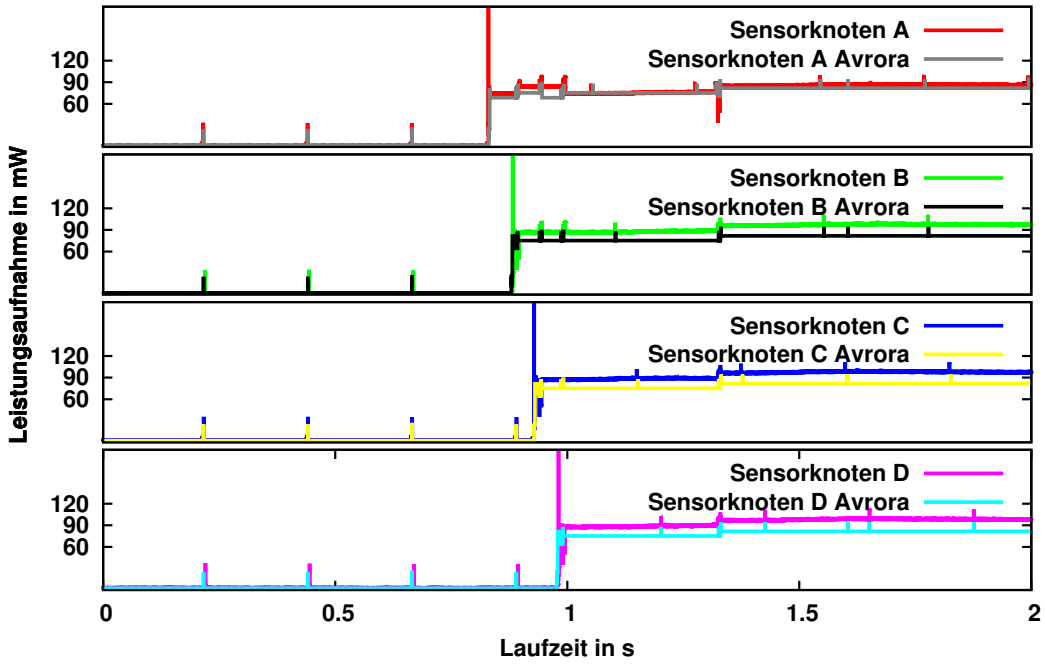
Sensorknoten B-D schalten je $100ms$ zeitversetzt den Transceiver an und versenden daraufhin eine 100 Byte Nachricht an Sensorknoten A per Unicast. Nach dem erfolgreichen Versand der Nachricht, wird jeweils die rote LED eingeschaltet. Der Empfang der Nachricht bei Sensorknoten A wird durch das Einschalten der gelben LED signalisiert, in der Abbildung ist dies durch die um etwa $3mW$ erhöhte Leistungsaufnahme und die drei kleineren Ausschläge bei Markierung 3 zu erkennen.

Nach dem erfolgreichen Empfang aller Nachrichten antwortet Sensorknoten A per Broadcast. Zu erkennen ist die Verwendung der grünen LED bei Sensorknoten A (Markierung (4)) und der roten LED bei Sensorknoten B-D (Markierung (5)) zur Signalisierung des Versands bzw. des Empfangs der Nachricht. Das Anschalten der LED bedingt eine um etwa $3mW$ erhöhte Leistungsaufnahme.

In der Abbildung der Leistungsaufnahme lassen sich zu verschiedenen Zeitpunkten Ausschläge der Leistungsaufnahme erkennen. Einige dieser Aus-



(a) Leistungsaufnahme der Testanwendung in SANDbed



(b) Vergleich der Leistungsaufnahme in SANDbedund AVRORA

Abbildung 3.2 Leistungsaufnahme von TestAvrora in SANDbed und AVRORA.

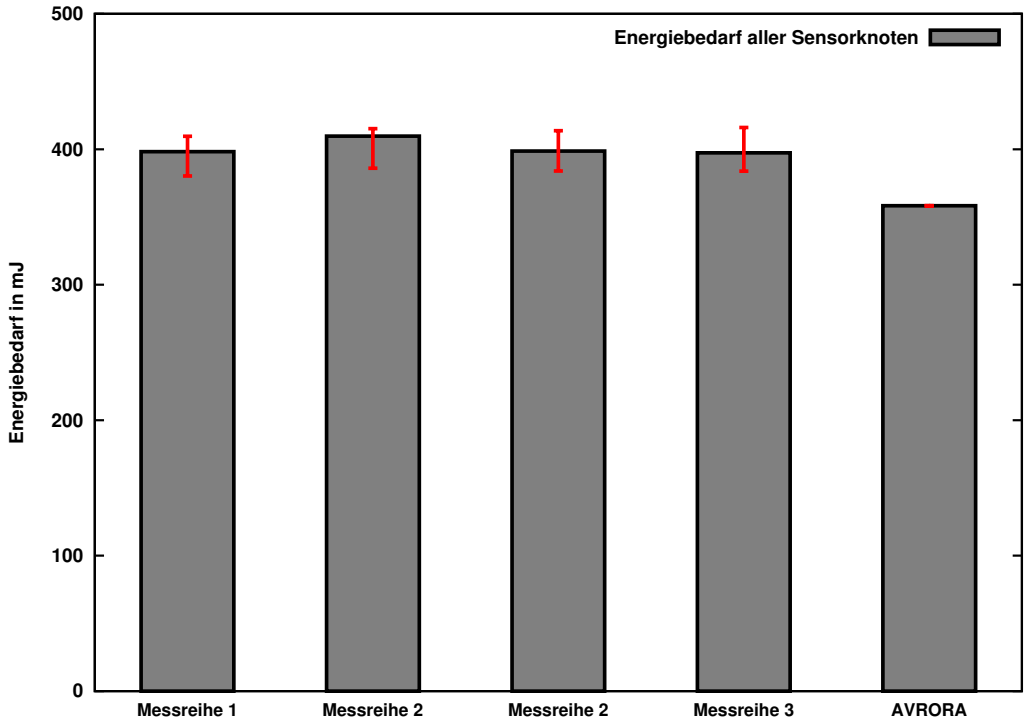


Abbildung 3.3 Kumulierter Energiebedarf der beteiligten Sensorknoten in SANDbed und AVRORA.

schläge sind bedingt durch interne Vorgänge der Sensorknotenhardware, wie beispielsweise interne Unterbrechungen oder dem Ablaufen eines Timers (beispielsweise bei Sekunde $0,45$). Andere Ausschläge wie der Ausschlag nach dem Einschalten des Funktransceivers sind durch kurzfristige Schwankungen der Stromstärke, vermutlich aufgrund der Entladung interner Kondensatoren, zu erklären.

Vergleich der Leistungsaufnahme in SANDbed und AVRORA

Abbildung 3.2b zeigt die Leistungsaufnahme der vier Sensorknoten während der Versuchsdurchführung sowohl in SANDbed als auch in AVRORA. Es ist zu erkennen, dass AVRORA die Ausführung der Anwendung korrekt abbildet. Das Anschalten des Transceivers, das Empfangen und Übertragen von Nachrichten, als auch das Anschalten der LEDs wird wie erwartet zeitlich korrekt initiiert und durchgeführt. Die absoluten Werte der Leistungsaufnahme sind jedoch für alle vier Knoten in diesem Experimentdurchlauf messbar zu gering. Dies ist insbesondere bei der Verwendung des Transceivers sichtbar, die Leistung zeigt hier eine Abweichung von bis zu $15mW$, was einer Abweichung von mehr als 15% entspricht.

Vergleich des Energiebedarfs in AVRORA und SANDbed

Abbildung 3.3 zeigt den durchschnittlichen, kumulierten Energiebedarf der vier an der Ausführung von TestAvrora beteiligten Sensorknoten. Insgesamt wurden 4 Messreihen mit jeweils 20 Wiederholungen durchgeführt, wobei in jeder Messreihe ein anderer Knoten als Sensorknoten A programmiert wurde.

Berechnet man aus den gemessenen Leistungsaufnahmen den Energiebedarf, so ergibt sich für die vier Messreihen ein durchschnittlicher Energiebedarf in SANDbed zwischen $398,25mJ$ und $409,65mJ$. In AVRORA konnte bei Ausführung der gleichen Testanwendung ein Energiebedarf von $358,32mJ$ gemessen werden. Dies entspricht einer Abweichung des durchschnittlichen Energiebedarfs von bis zu 15%.

Bei den hier durchgeführten Messungen konnte weiterhin festgestellt werden, dass es innerhalb der 20 Experimentwiederholungen zu leichten Schwankungen des Energiebedarfs der Sensorknoten kommt. In Abbildung 3.3 wurde deshalb für jede Messreihe der minimale und maximale gemessene Energiebedarf aufgezeichnet. Insgesamt konnte festgestellt werden, dass innerhalb einer Messreihe Schwankungen von bis zu $\pm 6\%$ möglich sind. Im Gegensatz hierzu kam es bei den Messungen in AVRORA zu keinerlei Schwankungen, der Energiebedarf der Testanwendung ist konstant $358,32mJ$.

Zusammengefasst lassen sich folgende Erkenntnisse aus dem Vergleich des Energiebedarfs und der Leistungsaufnahme ziehen:

- **Qualitativ korrekte Abbildung der Leistungsaufnahme:** Wie erwartet bildet AVRORA qualitativ die Leistungsaufnahme der Sensorknoten zeitlich korrekt ab. Hierdurch wird nochmals bestätigt, dass AVRORA prinzipiell geeignet ist den Energiebedarf im Vergleich zu realen Messungen abzubilden.

- **Quantitative Abweichung im Energiebedarf:** Der Vergleich des Energiebedarfs zeigt quantitative Unterschiede. Dies gilt für alle in TestAvrora eingesetzte Hardwarekomponenten und konnte in den Vergleichen mit bis zu 15% identifiziert werden.

- **Abhängigkeit des Energiebedarfs von der Knotenrolle:** Beim Vergleich des Energiebedarfs wurde deutlich, dass in den vier durchgeführten Messreihen in SANDbed ein unterschiedlicher kumulierter Energiebedarf gemessen wurde. Je nach Knotenrolle weisen die Sensorknoten somit einen unterschiedlichen Energiebedarf auf.

Mit Hilfe von TestAvrora konnte somit gezeigt werden, dass der simulierte Energiebedarf, im Vergleich zu SANDbed, Abweichungen und bisher nicht

abgebildete Schwankungen aufweist. Aufgrund dieser Tatsache werden im folgenden Abschnitt die im MICAz-Sensorknoten verbauten Komponenten Mikrocontroller, Transceiver und die LEDs detailliert auf ihren tatsächlichen Energiebedarf untersucht.

3.5 Detaillierte Analyse des Energiemodells in AVRORA

In diesem Abschnitt werden die im MICAz-Sensorknoten verbauten Komponenten Mikrocontroller, Transceiver und die LEDs mit Hilfe sogenannter *Mikrobenchmarks* detailliert auf ihren tatsächlichen Energiebedarf untersucht. Ein Mikrobenchmark versetzt die zu untersuchende Hardwarekomponente gezielt in alle möglichen Zustände. Somit ist es möglich, den Energiebedarf jedes Zustands und der Zustandsübergänge zu bestimmen.

3.5.1 Analyse des Mikrokontrollers

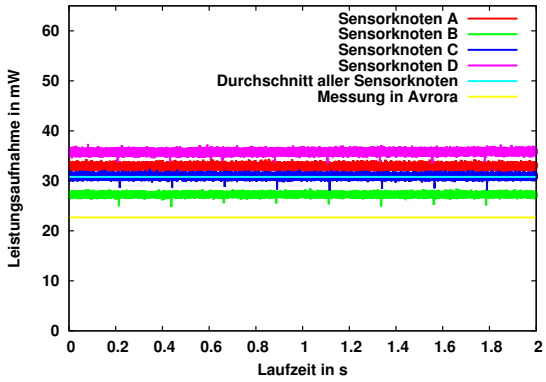
Ziel dieses Abschnitts ist es, die Leistungsaufnahme der einzelnen Zustände und der Zustandsübergänge des Mikrokontrollers mit Hilfe von SANDbed zu bestimmen und diese Werte zur Verbesserung des Energiemodells in AVRORA zu verwenden. Laut Datenblatt des Atmel ATmega128L [59] kann sich der Mikrokontroller in den Zuständen *Active*, *Idle*, *ADC Noise Reduction*, *Power-save* und *Power-down* befinden.

Mikrobenchmark des Mikrokontrollers

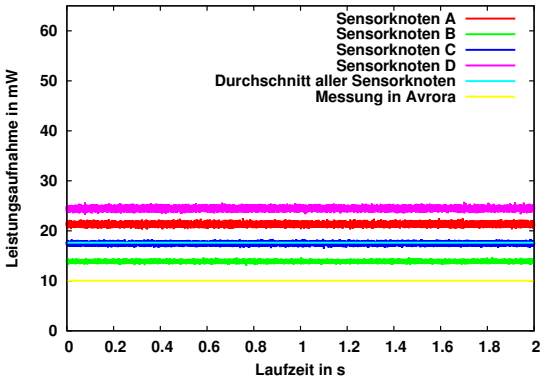
TinyOS bietet die Möglichkeit, mit Hilfe des *Mikrocontroller Power Management* gezielt zu steuern, in welche Zustände der Prozessor bei einem Zustandswechsel wechseln kann. Um die Leistungsaufnahme der einzelnen Zustände zu vermessen, werden daher im folgenden so genannte Mikrobenchmarks [46] eingesetzt, mit deren Hilfe der Mikrokontroller über einen fest definierten Zeitraum von 2 Sekunden in einem Zustand gehalten und gleichzeitig die Leistungsaufnahme mit Hilfe der SNMDs gemessen wird.

Versuchsparameter der Mikrobenchmarks

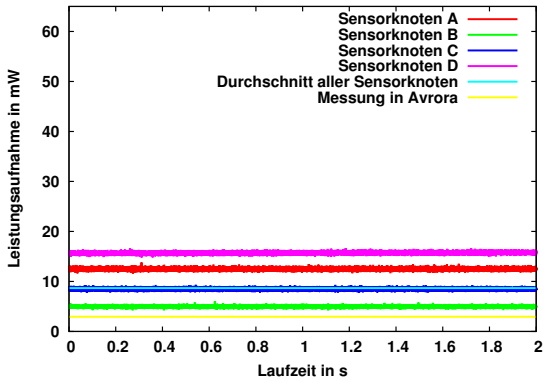
Tabelle 3.5 fasst die Versuchsparameter für die Mikrobenchmarks zusammen. Die Experimente werden sowohl in SANDbed als auch in AVRORA durchgeführt. Gemessen werden in SANDbed die Stromstärke und die Spannung der Sensorknoten über die Versuchsdauer von 2 Sekunden. Es wurden insgesamt alle Sensorknoten der SANDbed Management Nodes MN1, MN2, MN4 und MN5 für die Messungen verwendet, insgesamt 16 MICAz-Sensorknoten. Die ermittelten Werte sind die durchschnittliche Leistungsaufnahme (Mittelwert), sowie die minimale und maximale Leistungsaufnahme.



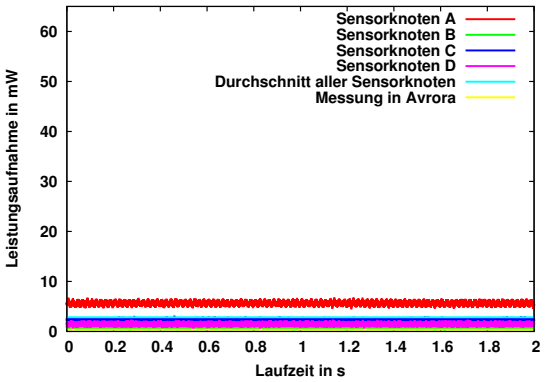
(a) Mikrokontrollerzustand *Active*.



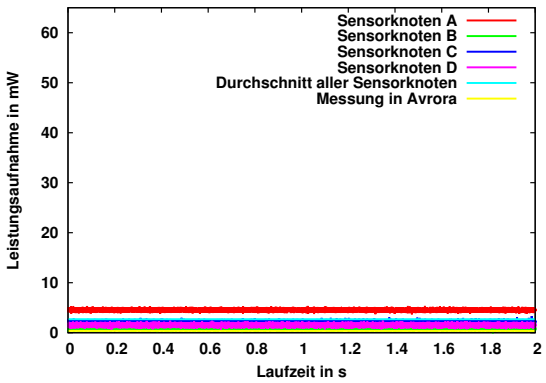
(b) Mikrokontrollerzustand *Idle*.



(c) Mikrokontrollerzustand *ADC Noise Reduction*.



(d) Mikrokontrollerzustand *Power-Save*.



(e) Mikrokontrollerzustand *Power-Down*.

Abbildung 3.4 Minimale, maximale und durchschnittliche Leistungsaufnahme des Mikrokontrollers in den Zuständen *Active*, *Idle*, *ADC Noise Reduction*, *Power-Save* und *Power-Down*.

Parameter	Wert
Simulator	AVRORA
Testbed	SANDbed
Knotenplattform	MICAz
Betriebssystem	TinyOS
Versuchswiederholungen	20
Versuchsdauer	2 Sekunden
Messrate des SNMD	100kHz
Messgrößen	Stromstärke I und Spannung U
Genutzte Sensorknoten	alle 16 Sensorknoten Management Nodes MN1, MN2, MN4, MN5
Vermessene Zustände	Active, Idle, ADC Noise Reduction, Power-save und Power-down
Ermittelte Werte	Durchschnittliche, minimale und maximale Leistungsaufnahme

Tabelle 3.5 Versuchsparameter der Mikrobenchmarks für den Mikrokontroller ATMega128L.

Leistungsaufnahme der einzelnen Zustände

Abbildung 3.4a zeigt die Leistungsaufnahme von vier exemplarisch ausgewählten Sensorknoten über den Messzeitraum von 2 Sekunden für den Mikrokontrollerzustand *Active*. In den 20 Testreihen konnte die maximale Leistungsaufnahme im Zustand *Active* bei dem Sensorknoten D mit durchschnittlich $36,37mW$ gemessen werden. Die geringste Leistungsaufnahme konnte bei dem Sensorknoten an B mit $27,26mW$ beobachtet werden. Im Durchschnitt ergaben die Messungen in SANDbed eine Stromstärke von $9,33mA$ bei einer Spannung von $3,3V$, was einer durchschnittlichen Leistungsaufnahme von $30,79mW$ entspricht. Die Messungen in AVRORA ergaben für alle Simulationsdurchläufe eine Leistungsaufnahme von $22,68mW$.

Abbildung 3.4b zeigt die Leistungsaufnahme für den Mikrokontrollerzustand *Idle*. Auch bei diesen Messungen konnte die maximale Leistungsaufnahme bei dem Sensorknoten D mit $21,32mW$ ermittelt werden, die geringste Leis-

tungsaufnahme von $13,86mW$ bei Sensorknoten B. Die durchschnittliche Leistungsaufnahme gemittelt über alle Messungen und Sensorknoten ergab einen Wert von $17,62mW$, Simulationen in Avrora ergaben einen konstanten Wert von $10,02mW$.

Aus den Messungen der Zustände *Idle* und *Active* lassen sich einige Rückschlüsse auf die Ursache der unterschiedlichen Energiebedarfswerte der Testanwendung aus dem vorigen Abschnitt ziehen. Zum Einen ist zu erkennen, dass die Energiebedarfswerte in SANDbed eine Varianz aufweisen. So zeigen die Messungen deutlich, dass unterschiedliche Sensorknoten des gleichen Typs unterschiedliche Energiebedarfswerte aufweisen. Zwischen der minimal bestimmten Leistungsaufnahme der Zustände *Active* und *Idle* und dem maximalen Wert zeigt sich in der Realität eine Varianz von bis zu 20%. Weiterhin ist zu erkennen, dass die Ermittlung der Energiebedarfswerte in AVRORA zu deutlich geringeren Werten führt und die in der Realität beobachtete Varianz nicht abgebildet wird.

Die Messergebnisse der Zustände *ADC Noise Reduction*, *Power-save* und *Power-down* sind in Abbildung 3.4c, 3.4d bzw. 3.4e dargestellt. Auch hier lässt sich erkennen, dass in AVRORA geringere Werte für die Leistungsaufnahme ermittelt wurden, als dies mit Hilfe von SANDbed messbar ist. Ebenfalls sind in den einzelnen Messungen die Varianzen zwischen den einzelnen gemessenen Leistungsaufnahmen zu erkennen.

Zustand	P_{avg}	P_{min}	P_{max}	P_{Avrora}
Idle	17,62	13,86	21,31	10,2
Active	30,79	27,26	36,37	22,68
ADC Noise Reduction Mode	8,68	4,95	12,44	2,97
Power-save	2,84	0,79	5,84	0,36
Power-down	2,71	0,76	4,65	0,36

Tabelle 3.6 Durchschnittliche, minimale und maximale Leistungsaufnahme der einzelnen Mikrokontrollerzustände in mW.

Tabelle 3.6 fasst die Ergebnisse der Vermessung des Mikrokontrollers zusammen. In der Tabelle sind sowohl die durchschnittliche (P_{avg}) als auch die minimal (P_{min}) und maximal (P_{max}) gemessene Leistungsaufnahme den mit AVRORA simulierten Werten (P_{Avrora}) gegenübergestellt.

Beim Vergleich der hier ermittelten Werte sind die zu geringen Werte des in AVRORA verwendeten Energiemodells und auch die fehlende Varianz im Vergleich zu den in SANDbed ermittelten Leistungswerten zu sehen.

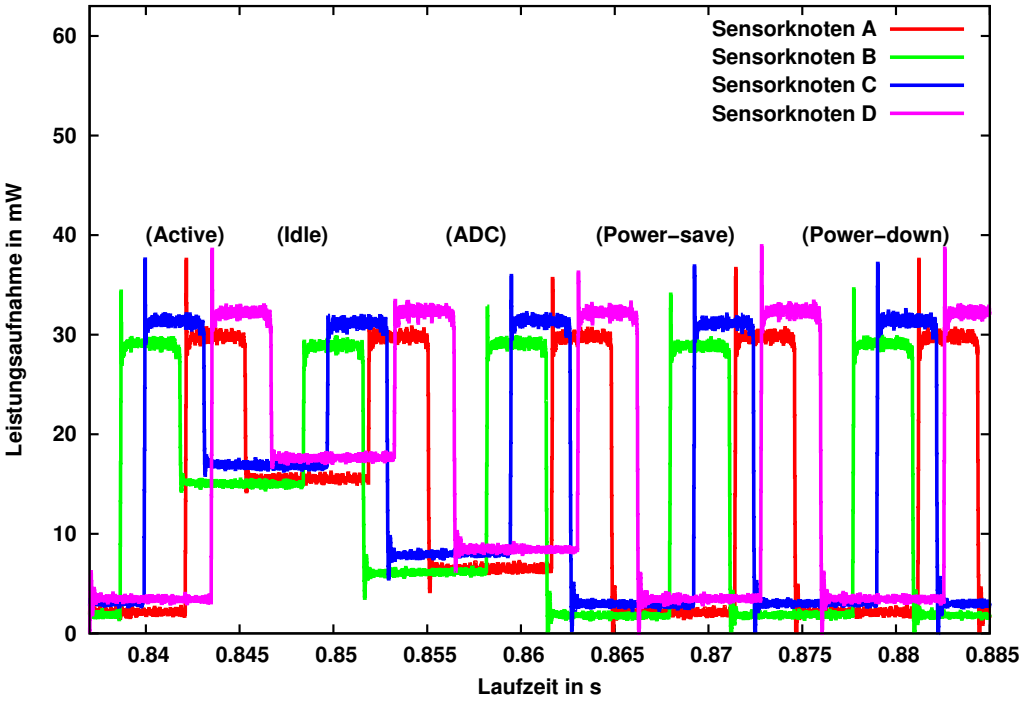


Abbildung 3.5 Leistungsaufnahme der Zustände des Mikrocontrollers und der Zustandsübergänge.

Leistungsaufnahme der Zustandsübergänge

Neben den absoluten Werten der einzelnen Zustände wurde in einem weiteren Versuch auch das Verhalten der Sensorknoten bei einem Zustandsübergang betrachtet. Hierfür wurde eine TinyOS Anwendung entworfen, die alternierend zwischen den Zuständen *Active*, *Idle*, *ADC Noise Reduction* und *Power-save* bzw. *Power-down* wechselt. Der Zustand Active wurde hierbei $10ms$ gehalten, die anderen Zustände $12ms$ [5].

Abbildung 3.5 zeigt die Leistungsaufnahme der einzelnen Zustände sowie die Zustandsübergänge für vier Sensorknoten in SANDbed. Wie in den vorigen Experimenten ist auch in diesem Versuch die Varianz des gemessenen Energiebedarfs zu erkennen. Weiterhin wird deutlich, dass die Sensorknoten für alle Zustände ein ähnliches Verhältnis des Energiebedarfs zueinander aufweisen. Der Sensorknoten D ist beispielsweise in allen Zuständen des Mi-

[5] In Voruntersuchung erwiesen sich diese Werte als minimal notwendige Zeiten, mit deren Hilfe sich ein stabile Stromstärke für jeden Zustand messen ließ. Größere Werte haben keinen Einfluss auf die Untersuchung der Zustandsübergänge.

krokontrollers der Sensorknoten mit der höchsten Leistung, im Schnitt 11% höher als der Sensorknoten B.

Abbildung 3.6 zeigt einen Vergleich der Zustandsübergänge zwischen AVRORA und SANDbed. Hierzu ist in Abbildung 3.6a ein direkter Vergleich der Messungen der Leistungsaufnahme von Sensorknoten A mit einer Simulation der gleichen Anwendung in AVRORA zu sehen. Ebenso wie in den vorigen Messungen ist erkennbar, dass AVRORA zwar die Übergänge zeitlich korrekt initiiert und auch den Verlauf des Energiebedarfs qualitativ korrekt abbildet, der absolute Energiebedarf der Zustände *Active*, *Idle* und *ADC Noise Reduction* in AVRORA aber deutlich zu gering ist.

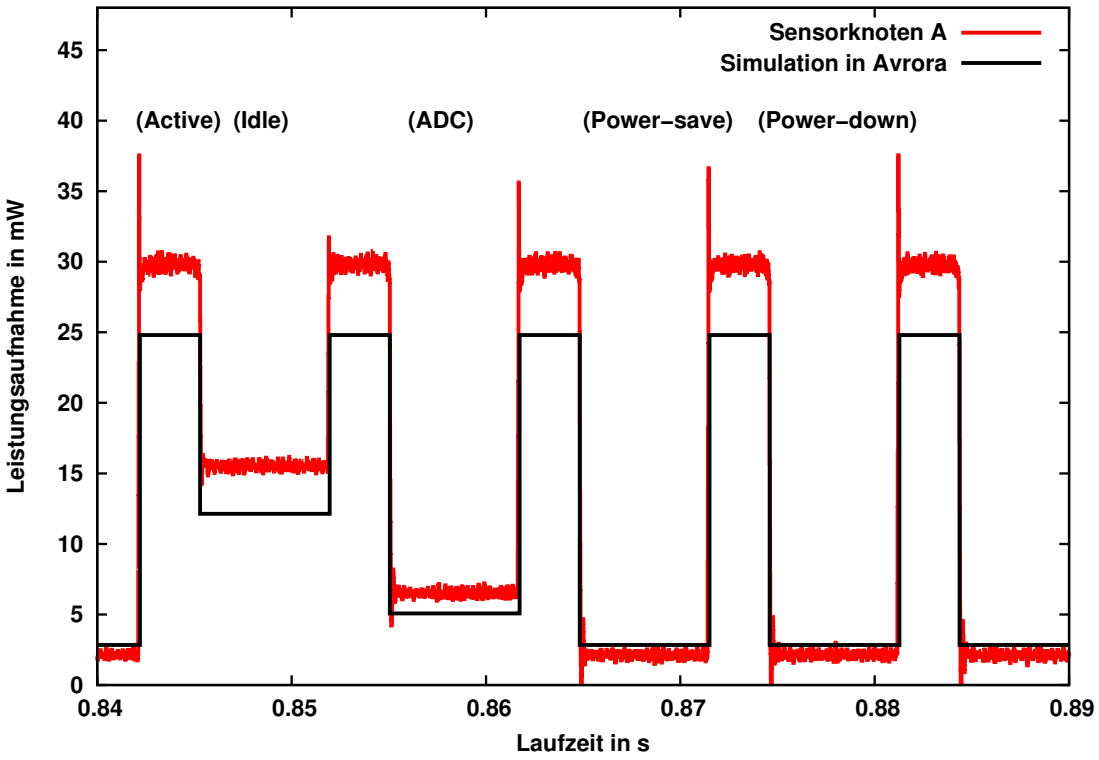
Abbildung 3.6b zeigt einen detaillierten Ausschnitt der Zustandsübergänge des Mikrokontrollers vom Zustand *Idle* zu *Active* (1) und danach in den Zustand *ADC Noise Reduction* (2). In AVRORA erfolgen Zustandsübergänge sofort, die Leistungsaufnahme ändert sich sofort nach der Initiierung des Zustandübergangs. Die realen Messungen in SANDbed zeigen jedoch, dass ein Zustandsübergang mit zusätzlichen Kosten verbunden ist. Der Zustandsübergang dauert eine kurze Zeit und verursacht hierdurch zusätzliche Energiekosten. Die Kosten des Zustandsübergangs sind in der Abbildung eingekreist. Diese Kosten werden in AVRORA nicht ermittelt und sind nicht Teil des Energiemodells. Während die Kosten für einen einzelnen Zustandsübergang gering erscheinen, war es zu diesem Zeitpunkt nicht möglich abzuschätzen, wie sich die Gesamtkosten aller nicht abgebildeten Zustandsübergänge auf den Energiebedarf einer Anwendung auswirken. Daher wurden die Kosten der Zustandsübergänge in AVRORA+ aufgenommen und deren Auswirkungen anschließend evaluiert.

3.5.2 Analyse des CC2420-Transceivers

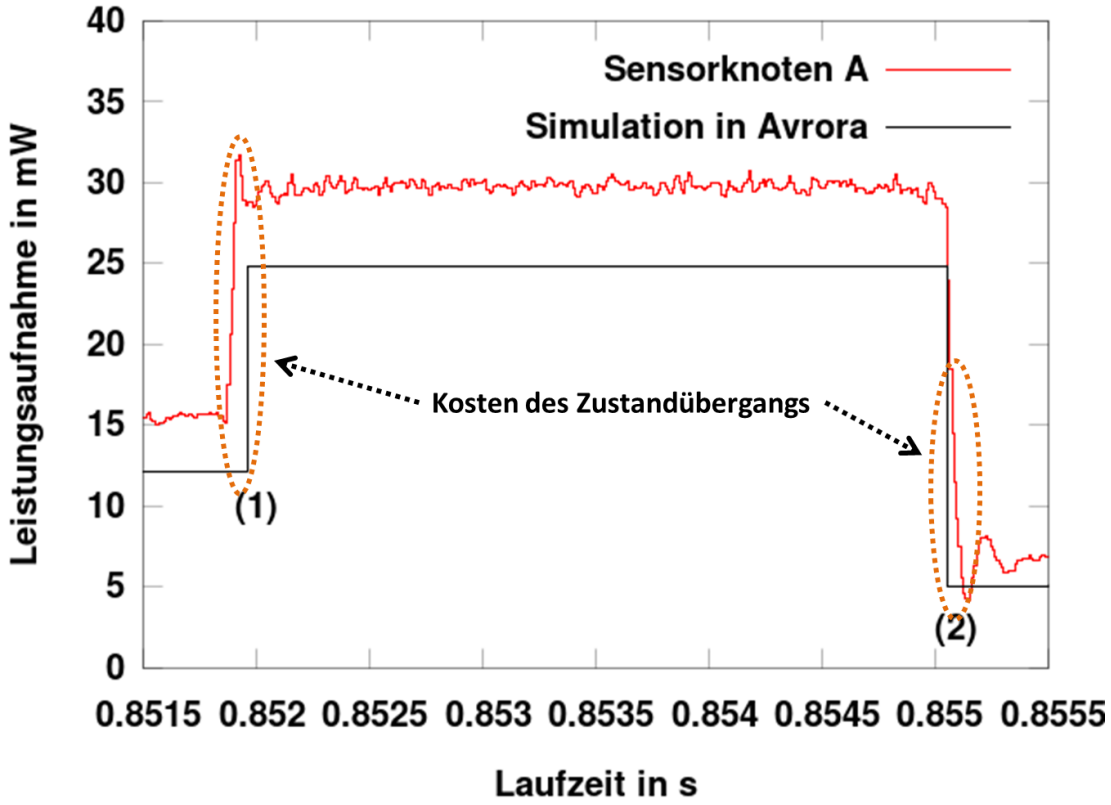
Neben dem Mikrokontroller ist der CC2420-Transceiver die Komponente des MICAz-Sensorknotens mit dem größten Energiebedarf. Wie im vorigen Abschnitt wird mit Hilfe von SANDbed untersucht, wie sich der Energiebedarf in AVRORA von einer realen Nutzung in SANDbed unterscheidet.

Mikrobenchmark des CC2420-Transceivers

Wie bei der Untersuchung des Mikrokontrollers, werden die verschiedenen Systemzustände *Off*, *Listen*, *Receive* und *Transmit* des CC2420 vermessen. Hierfür wurden die Messungen von TestAvrora genutzt, um die entsprechenden Aktionen des CC2420 zu analysieren und die Leistungsaufnahme zu bestimmen. TestAvrora wurde hierfür auf allen Sensorknoten in MN1, MN2, MN4 und MN5 ausgeführt, insgesamt somit wieder auf 16 Sensorknoten.

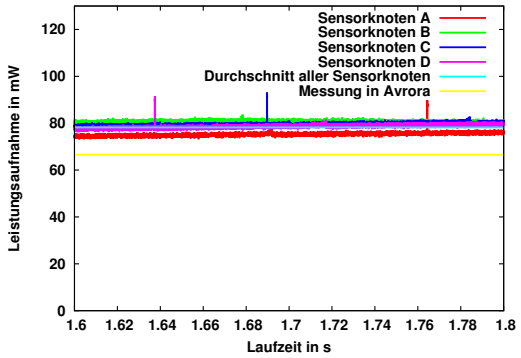


(a) Leistungsaufnahme des Sensorknotens A und AVRORA

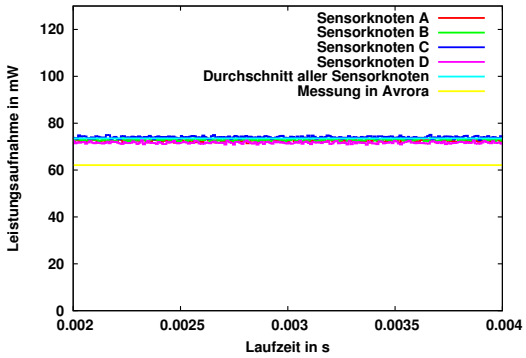


(b) Detailierte Ansicht eines Zustandsübergangs

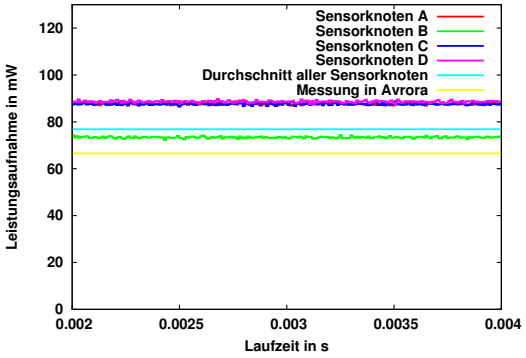
Abbildung 3.6 Leistungsaufnahme der Zustände des Mikrocontrollers und der Zustandsübergänge in Avrora und auf Sensorknoten A im Vergleich.



(a) CC2420 Zustand Listen



(b) CC2420 Zustand Transmit



(c) CC2420 Zustand Receive

Abbildung 3.7 Durchschnittliche, minimale und maximale Leistungsaufnahme des CC2420-Transceiver in den Zuständen *Listen*, *Transmit* und *Receive*

Versuchsparamter der Mikrobenchmarks

Im Gegensatz zu der Untersuchung des Mikrokontrollers ist es bei der Analyse des Transceivers nicht möglich, jeden Zustand des Transceivers über einen Zeitraum von 2 Sekunden konstant zu halten. Beispielsweise dauert die Übertragung einer Nachrichteneinheit zwischen 2 und 5ms, der Transceiver befindet sich also nur genau über diesen Zeitraum im Zustand *Transmit*. Dementsprechend wurde bei den Messungen der Leistungsaufnahme für die Zustände *Transmit* und *Receive* jeweils die Dauer der Übertragung einer Nachricht vermessen, die Leistungsaufnahme der Zustände *Off*, *Power-Down* und *Idle* über einen Zeitraum von 2 Sekunden. Tabelle 3.7 fasst die Versuchsparameter zusammen.

Leistungsaufnahme der einzelnen Zustände

Abbildung 3.7a zeigt die Leistungsaufnahme des CC2420-Transceivers im Zustand *Listen*. Die größte Leistungsaufnahme im Zustand *Listen* konnte am Sensorknoten B mit $81,4mW$, die kleinste Leistungsaufnahme von $74,6mW$

Parameter	Wert
Simulator	AVRORA
Knotenplattform	MICAz
Betriebssystem	TinyOS
Versuchswiederholungen	20
Versuchsdauer	2 Sekunden für die Zustände *Listen* und *Off*
Versuchsdauer	Dauer einer Übertragung für Zustände *Transmit* und *Receive*
Messrate des SNMD	100kHz
Messgrößen	Stromstärke I und Spannung U
Genutzte Sensorknoten	alle Sensorknoten in MN1, MN2, MN4 und MN5
Vermessene Zustände	*Transmit*, *Receive*, *Listen* und *Off*
Ermittelte Werte	Durchschnittliche, minimale und maximale Leistungsaufnahme

Tabelle 3.7 Versuchsparameter der Mikrobenchmarks für den Transceiver CC2420.

an Sensorknoten A ermittelt werden. Die durchschnittliche Leistungsaufnahme über alle vermessenen Sensorknoten lag mit $78,21mW$ deutlich über der in AVRORA simulierten Leistungsaufnahme von $66,54mW$.

In Abbildung 3.7b ist die Leistungsaufnahme des CC2420 beim Versenden und beim Empfang von Nachrichten zu sehen. Die Übertragung der Nachrichten wurde hierbei mit der höchsten einstellbaren Sendestärke des MICAz-Sensorknotens durchgeführt. Die maximal gemessene Leistungsaufnahme lag mit $58,37mW$ 11% über der simulierten Leistungsaufnahme von $52,20mW$ in AVRORA. Es ist zu erkennen, dass der Empfang einer Nachricht eine Leistungsaufnahme von $56,40mW$ verursacht. Die maximale Leistungsaufnahme lag mit $59,89mW$ 6% über den simulierten Werten. Analog zu den Messungen für den Mikrokontroller konnte in den Messungen eine große Varianz zwischen verschiedenen MICAz-Sensorknoten und im Schnitt eine Abweichung von 8% gegenüber der Simulation beobachtet werden.

Zustand	P_{avg}	P_{min}	P_{max}	P_{Avrora}
off	0,000066	-	-	0,00006
Listen	60,62	57,58	62,80	-
Receive	57,35	55,01	59,89	56,40
Transmit 0/31	56,03	51,31	58,37	52,20
Transmit 30/29/28	54,55	52,57	55,77	51,73
Transmit 27	52,31	50,63	53,30	49,50
Transmit 26/25/24	50,39	48,74	51,35	48,57
Transmit 23	48,35	46,54	49,37	45,60
Transmit 22/21/20	46,37	44,65	47,26	44,62
Transmit 19	44,19	42,64	44,98	41,70
Transmit 18/17/16	42,24	40,66	43,00	40,66
Transmit 15	40,00	38,61	40,63	37,50
Transmit 14/13/12	37,95	36,80	38,54	36,49
Transmit 11	35,71	34,72	36,27	33,60
Transmit 10/9/8	33,63	32,64	34,22	32,64
Transmit 7	31,55	30,43	32,38	29,70
Transmit 6/5/4	29,60	28,61	30,33	28,69
Transmit 3	27,32	26,40	28,15	25,50
Transmit 2/1	25,34	24,45	26,07	24,38

Tabelle 3.8 Durchschnittliche, minimale und maximale Leistungsaufnahme der einzelnen Zustände des CC2420 in mW.

Tabelle 3.8 fasst die gemessene Leistungsaufnahme für den CC2420 zusammen. Neben der durchschnittlichen Leistungsaufnahme (P_{avg}), sind auch die minimalen (P_{min}) und maximalen Leistungsaufnahmen (P_{max}), sowie die in AVRORA verwendete Leistungsaufnahme (P_{Avrora}) dargestellt.

Wie in [46] erläutert, unterscheidet AVRORA nicht zwischen den Zuständen *Listen* und *Receive*. Für beide Zustände wird ein identischer Energiebedarf angenommen. Messungen in SANDbed zeigen jedoch, dass der Empfang einer Nachricht eine im Vergleich zum reinen Lauschen auf den Kanal $3,27mW$ geringere Leistungsaufnahme aufweist (siehe Abbildung 3.7a und 3.7c). Deshalb wurden diese beiden Zustände mit der unterschiedlichen Leistungsaufnahme in AVRORA+ integriert.

Der Funktransceiver CC2420 bietet verschiedene Sendeleistungen an, die sich in der resultierend Übertragungsreichweite und der notwendigen Leistungsaufnahme unterscheiden. Die Sendeleistung mit der größten Reichweite ist die Sendeleistung 31, welche in SANDbed zu einer Übertragungsreichweite von etwa $25m$ führt. Die geringste Sendeleistung 1 resultiert in einer Reichweite von unter $1m$. Die Sendeleistung kann mit Hilfe des Betriebssystems eingestellt werden.

In Abhängigkeit der eingestellten Sendeleistung weist der Zustand *Transmit* unterschiedliche Leistungsaufnahmen auf. Bei voller Sendeleistung (*Transmit 0/31*) ist die gemessene Leistungsaufnahme mit $56,03mW$ im Schnitt $3,83mW$ höher als die in AVRORA verwendete Leistungsaufnahme. Ähnliche Unterschiede lassen sich auch für die anderen Sendeleistungen (*Transmit 1-30*) beobachten. Generell sind die Leistungswerte in AVRORA zu gering. Wie auch beim Mikrokontroller, lässt sich bei allen Zuständen eine Varianz in den gemessenen Werten erkennen, welche nicht in AVRORA abgebildet wird.

3.5.3 LEDs

Der MICAz-Sensorknoten ist mit drei LEDs ausgestattet. Jede dieser LEDs kann einzeln über das Betriebssystem an- oder ausgeschaltet werden. Abbildung 3.8 zeigt die Ausführung der TinyOS Anwendung *Blink* auf vier SNMDs in SANDbed. Blink setzt dabei einen Binärzähler mit Hilfe der LEDs um, alle möglichen Kombinationen an LED Zuständen werden somit genutzt.

Wie bei den anderen Hardwarekomponenten, ist auch bei den LEDs eine Varianz im Vergleich unterschiedlicher MICAz-Sensorknoten zu erkennen. Weiterhin weisen die unterschiedlichen LEDs unterschiedliche Leistungsaufnahmen auf. Beispielsweise weist die gelbe LED (Markierung (1)) einen leicht höhere Leistungsaufnahme auf als die grüne LED (Markierung (2)).

Tabelle 3.9 fasst die ermittelten durchschnittlichen Leistungsaufnahmen und die dazugehörige Varianz zusammen. Zusätzlich ist die im Energiemodell von AVRORA verwendete Leistungsaufnahme aufgetragen. Die Leistungsaufnahme in AVRORA liegt deutlich unter den in SANDbed ermittelten Werten und weist auch nicht die in der Realität beobachtete Varianz auf.

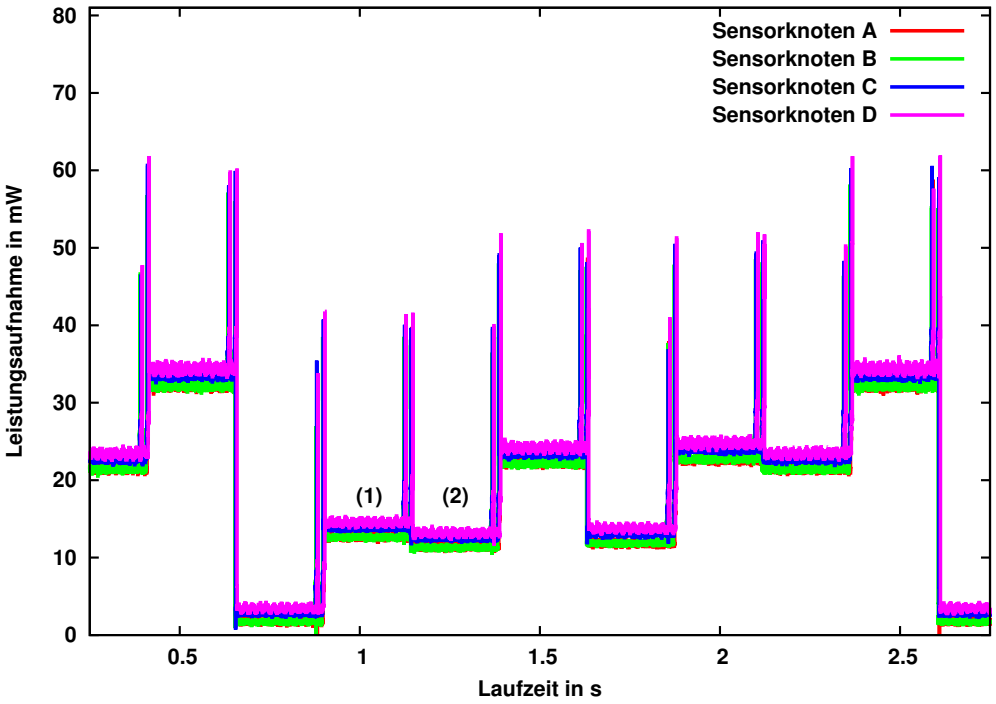


Abbildung 3.8 Leistungsaufnahme während der Ausführung von *Blink*

Zustand	P_{avg}	P_{min}	P_{max}	P_{Avrora}
off	0,0	0,0	0,0	0
on_{RED}	10,56	10,23	10,89	6,6
on_{GREEN}	9,44	9,21	9,67	6,6
on_{YELLOW}	9,97	9,74	10,20	6,6

Tabelle 3.9 Durchschnittliche, minimale und maximale Leistungsaufnahme der LEDs in mW.

3.6 Entwurf und Implementierung von Avrora+

In den vorigen Abschnitten wurde der Energiebedarf von Messungen in SANDbed der Hardwarekomponenten Mikrokontroller, CC2420-Transceiver und LEDs mit den Ergebnissen von Simulationen in AVRORA verglichen. Hierbei wurde deutlich, dass es Abweichungen im Energiebedarf von bis zu 15% bei der Ausführung der Testanwendung gab. Mit Hilfe der detaillierteren Untersuchung der Hardwarekomponenten des MICAz-Sensorknotens konnten drei mögliche Verbesserungen des Energiemodells in AVRORA identifiziert werden:

- **Kalibrierung des Energiemodells:** Wie mit Hilfe der Messungen deutlich wurde, existiert ein systematischer Offset zwischen realen Energiebedarfsdaten und den in AVRORA ermittelten Werten. Die Werte in AVRORA sind grundsätzlich immer zu niedrig angesetzt.
- **Hinzufügen der Hardwarevarianzen:** Wie durch die Messung auf verschiedenen Sensorknoten deutlich wurde, existiert eine signifikante Hardwarevarianz im Bezug auf den Energiebedarf. Unterschiedliche Sensorknoten des gleichen Typs MICAz weisen bei der Ausführung der gleichen Anwendung Varianzen bis zu 20% auf.
- **Hinzufügen der Kosten für Zustandsübergänge:** Zustandsübergänge in AVRORA sind nicht mit Energiekosten verbunden. Die Untersuchung auf realen Sensorknoten zeigte jedoch, dass Zustandsübergänge zusätzlichen Energiebedarf verursachen, welcher bisher nicht in AVRORA abgebildet wird.

Die Ergebnisse aus den vorigen Abschnitten werden im Folgenden genutzt, eine neue Version von AVRORA zu entwerfen, die die drei oben genannten Verbesserungsmöglichkeiten adressiert. Die neue Version wird in den folgenden Abschnitten als AVRORA+ bezeichnet.

Kalibrierung des Energiemodells in AVRORA+

Das Energiemodell in AVRORA wurde mit den Werten aus den Datenblättern der einzelnen Hardwarekomponenten und mit Hilfe von Einzelmessungen kalibriert, vergleiche [28]. Die Messungen in SANDbed zeigen jedoch, dass die mit AVRORA gewonnen Energiebedarfswerte deutlich unter den in der Realität bestimmten Werten liegen. Da ein Ziel dieser Arbeit eine möglichst hohe Vergleichbarkeit zwischen Simulation und realen Messungen in SANDbed ist, wurden die in AVRORA verwendeten Energiebedarfswerte an die Messergebnisse angepasst. Hierzu wurden die Energiebedarfswerte der Zustände des Mikrokontrollers, des CC2420-Transceivers und der LEDs an die Werten aus Tabelle 3.6, 3.8 und 3.9 angepasst.

Simulation von Hardwarevarianzen in AVRORA+

Die Vermessung der einzelnen Zustände der Hardwarekomponenten zeigen, dass zwischen verschiedenen Sensorknoten und Hardwarekomponenten des gleichen Typs eine starke Varianz des Energiebedarfs festgestellt werden kann. So konnte in Abschnitt 3.4 anhand der Anwendung TestAvrora und einer Messung des Energiebedarfs über einen Zeitraum von 2 Sekunden auf vier SNMDs bereits eine Abweichung von maximal 15% gemessen wurde. Die

genaue Analyse der beteiligten Hardwarekomponenten führte zu dem gleichen Ergebnis. In AVRORA+ wird neben des neu kalibrierten Energiemodel auch die Hardwarevarianz mit berücksichtigt. Für jede Hardwarekomponente und jeden Zustand werden die in Tabellen 3.6, 3.8 und 3.9 dargestellten minimalen und maximalen Energiebedarfswerte genutzt, um ein variables Energiemodell umzusetzen.

In jedem Simulationsdurchlauf wird in AVRORA+ für jede Hardwarekomponente eine Leistungsaufnahme zufällig mit Hilfe eines Zufallszahlengenerators bestimmt. Der Zufallszahlengenerator liefert hierbei eine normalverteilte Zufallszahl innerhalb des Intervalls [Minimale Leistungsaufnahme, maximale Leistungsaufnahme].

Hierbei ist zu beachten, dass die Messungen in SANDbed nur mit 16 Sensorknoten durchgeführt werden konnten. Aus diesen (wenigen) Messungen ließ sich keine allgemeingültige Aussage über die Verteilung der Energiebedarfswerte gewinnen. Deswegen wurde die relativ einfache Modellierung mit Hilfe einer normalverteilten Zufallszahl genutzt. Die Evaluierungsergebnisse im weiteren Verlauf des Kapitels zeigen, dass die so gewonnenen Ergebnisse für die Sensorknoten des SANDbeds vergleichbar sind. In wie weit diese Modellierung für alle Sensorknoten des Typs MICAz gültig ist müsste mit einer größeren Anzahl an Sensorknoten verifiziert werden.

Einfügen von Kosten für Zustandsübergänge in AVRORA+

Jeder Zustandsübergang verursacht Zustandsübergangskosten, die im bisherigen Energiemodell in AVRORA nicht berücksichtigen werden. In AVRORA+ werden daher jedem Zustandsübergang eine Zustandsübergangszeit und die durch den Zustand verursachten Kosten zugewiesen. Abbildung 3.9 zeigt beispielhaft den Zustandsübergangsgraph inklusive durchschnittlicher Energiebedarfskosten in AVRORA und AVRORA+, sowie die jeweiligen Zustandsübergangszeiten und Zustandsübergangskosten des Mikrokontrollers. Ein Zustandsübergang von *Active* nach *Idle* in AVRORA+ verursacht neben den Zustandskosten von $30,79mW$ bzw. $17,62mW$ noch Zustandsübergangskosten von $20,82mW$ über einen Zeitraum von $39\mu s$. Diese Zustandsübergangskosten werden separat in der von AVRORA bzw. AVRORA+ erzeugten Energie-Log Datei ausgegeben. Die Zustandsübergänge des Transceivers sowie der LEDs wurden analog zu den Zustandsübergängen des Mikrokontrollers modelliert und in das Energiemodell eingefügt.

3.7 Evaluation

Die Evaluation von AVRORA+ wird mit Hilfe der Mikrobenchmarks und TestAvrora durchgeführt. Innerhalb der Evaluation soll gezeigt werden, in

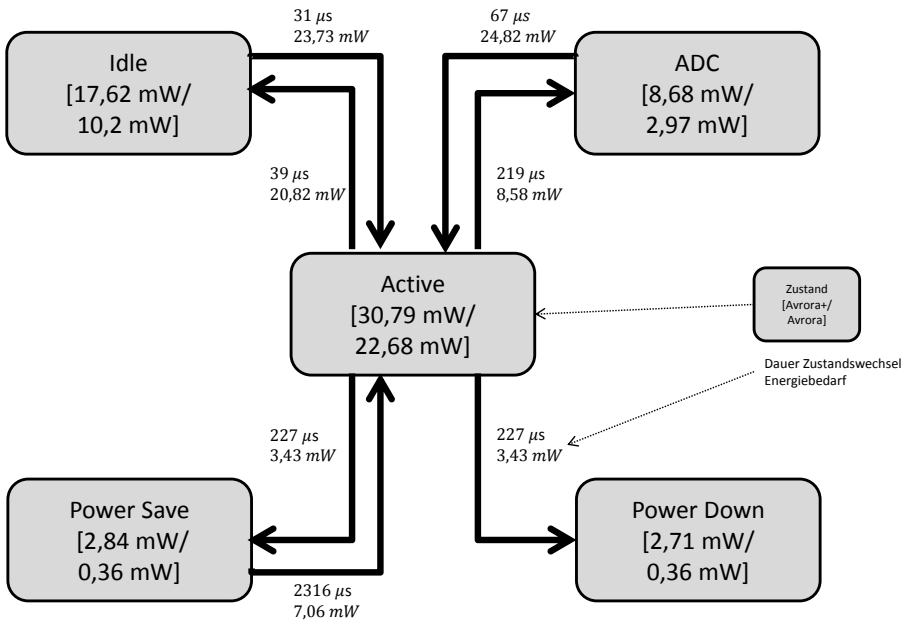
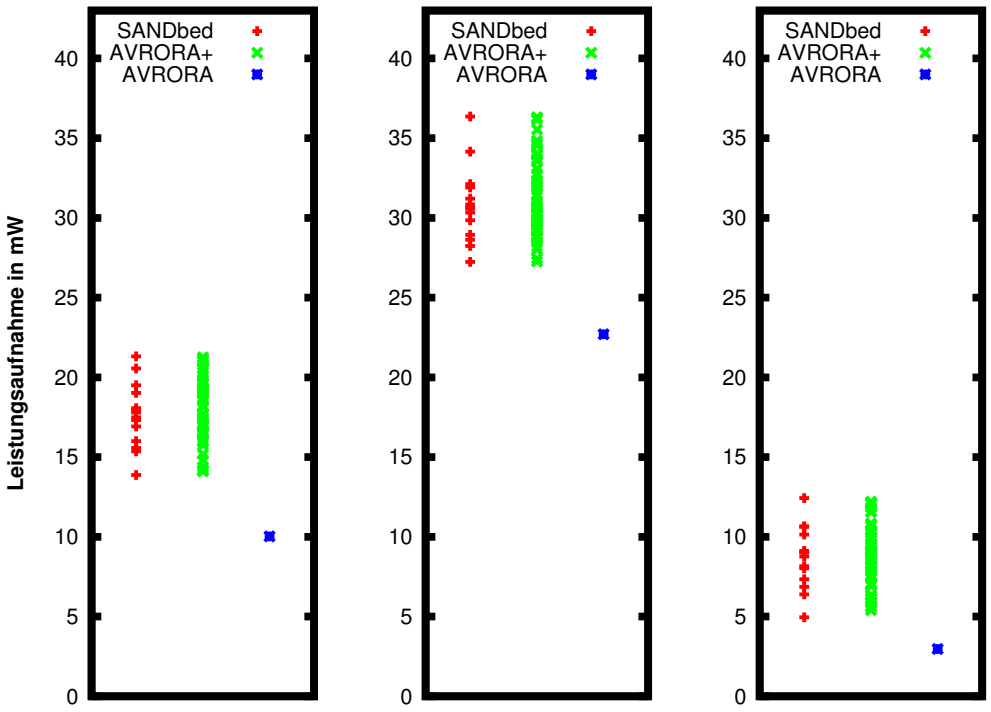


Abbildung 3.9 Zustandsübergänge des ATMega128L.

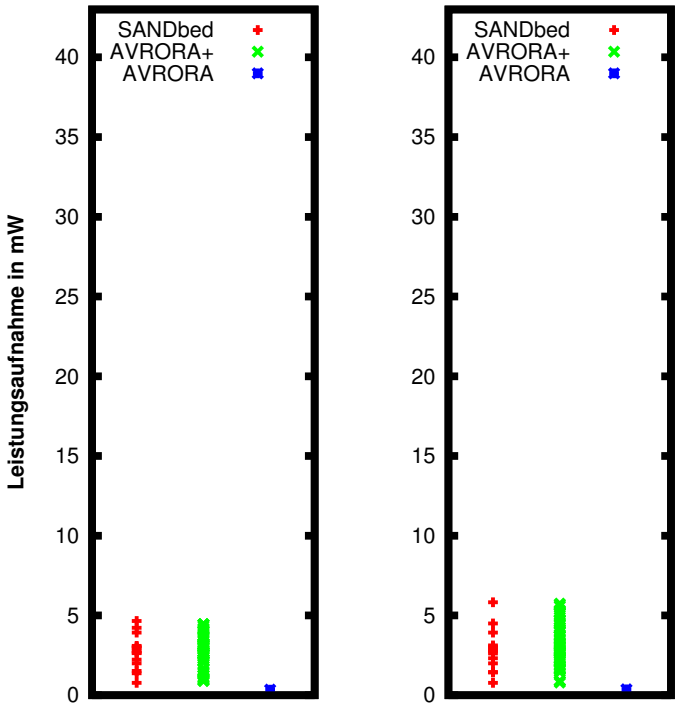
wie weit die im Abschnitt 3.6 aufgezeigten Verbesserungen die Vergleichbarkeit des Energiebedarfs aus Simulationen mit den Messergebnissen aus SANDbed erhöht. Im ersten Teil der Evaluation wird anhand der Ergebnisse aus Simulationen der Mikrobenchmarks in AVRORA+ auf die *Kalibrierung des Energiemodells in AVRORA+* und der *Hardwarevarianz* des Mikrokontrollers eingegangen. Im zweiten Abschnitt wird anhand der Ergebnisse von Test-Avrora die Abweichung der Energiebedarfswerte zwischen SANDbed und AVRORA+ unter Nutzung aller Verbesserungen gezeigt.

3.7.1 Mikrobenchmarks

In Abschnitt 3.6 wurde erläutert, dass das Energiemodell in AVRORA ungenau kalibriert ist und zudem die in der Realität zu messenden Varianzen des Energiebedarfs nicht abbildet. Diese beiden Verbesserungsmöglichkeiten wurden in AVRORA+ umgesetzt. Abbildung 3.10a und 3.10b zeigen hierzu beispielhaft die Leistungsaufnahme des Mikrokontrollers in den Zuständen *Idle*, *Active*, *ADC Noise Reduction* bzw. *Power-Down* und *Power-Save* in SANDbed, AVRORA und AVRORA+. Hierfür wurden neben den Messungen in SANDbed auch 50 Simulationsdurchläufe der Mikrobenchmarks in AVRORA+ durchgeführt.



(a) Zustände *Idle*, *Active* und *ADC*.



(b) Zustände *Power-Down* und *Power-Save*.

Abbildung 3.10 Vergleich der Mikrokontrollerzustände *Idle*, *Active* und *ADC Noise Reduction* in SANDbed, AVRORA und AVRORA+.

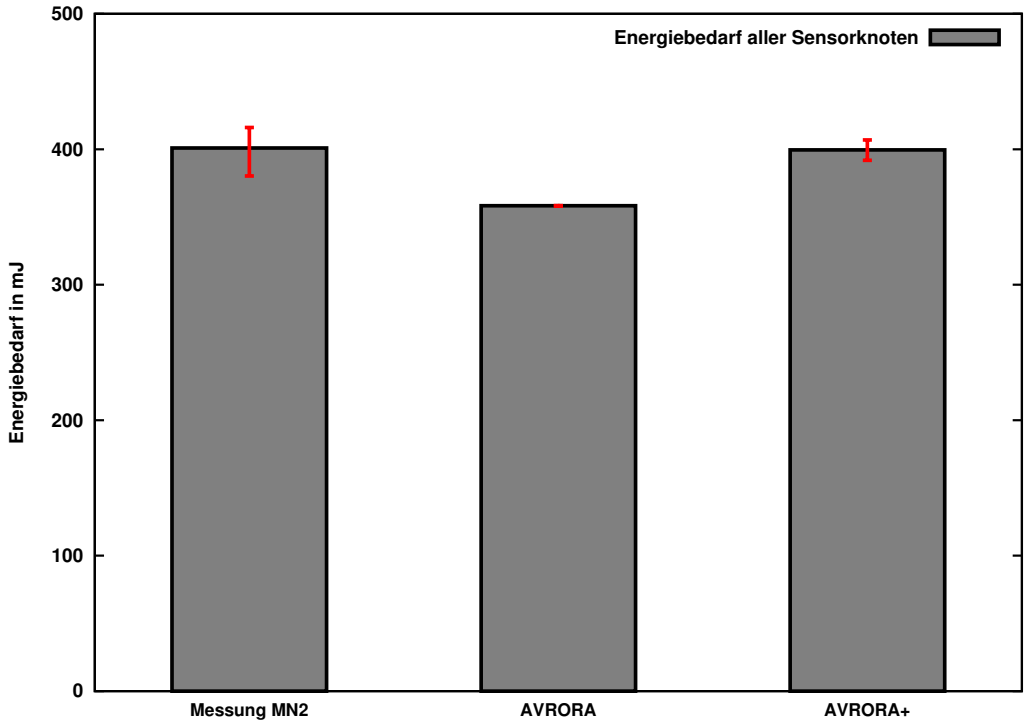


Abbildung 3.11 Kumulierter Energiebedarf der beteiligten Sensorknoten in SANDbed, AVRORA und AVRORA+.

Bei der so bestimmten Leistungsaufnahme ist zu erkennen, dass AVRORA+ sowohl die Varianz der Leistungsaufnahme als auch die absolute Leistungsaufnahme quantitativ und qualitativ genauer abbildet. Beispielsweise ist bei der Leistungsaufnahme des Zustands *Idle* zu erkennen, dass sowohl die minimal als auch die maximal in SANDbed gemessene Leistungsaufnahme ($21,9mW$ und $13,8mW$) mit den in AVRORA+ ermittelten minimalen und maximalen Werten übereinstimmt. Die Leistungsaufnahme in AVRORA ist mit konstant $9,9mW$ deutlich geringer.

Wie in den vorigen Abschnitten erläutert, sind die mit AVRORA bestimmten Leistungsaufnahmen in allen Versuchen deutlich unter den in SANDbed gemessenen Werten und weisen keine Varianz auf. Die Abweichung zwischen SANDbed und AVRORA+ an den jeweiligen Intervallgrenzen der Leistungsaufnahme ist mit unter 1% wesentlich geringer als die in Abschnitt 3.5.1 bestimmten Abweichungen.

3.7.2 Vergleich des Energiebedarfs von TestAvrora

Bei der Ausführung von TestAvrora wurde eine Abweichung des Energiebedarfs zwischen AVRORA und SANDbed von 15% gemessen. Im Rahmen dieser Evaluation wurden die Simulationen der Testanwendung auch in AVRORA+ durchgeführt. Es wurden insgesamt 50 Simulationsdurchläufe durch-

geführt und der Energiebedarf ermittelt. Abbildung 3.11 zeigt den kumulierten Energiebedarf der Testanwendung in AVRORA, AVRORA+ und die durchschnittlichen Ergebnisse bei der Messung in SANDbed. Die Ausführung der Testanwendung in AVRORA+ ergab einen Energiebedarf von durchschnittlich $399,56mJ$, Messungen in SANDbed ergaben einen durchschnittlichen Energiebedarf von $400,96mJ$ für alle beteiligten Sensorknoten. Die Messungen in AVRORA ergaben einen Energiebedarf von $358,32mJ$. Die Abweichung im Energiebedarf zwischen AVRORA+ und SANDbed beträgt somit weniger als 1%. Weiterhin ist zu erkennen, dass die ermittelten Varianzen der Energiebedarfswerte in AVRORA+ deutlich näher an den Ergebnissen aus SANDbed liegen, der maximale Energiebedarf betrug mit $416,07mJ$ nur etwa $2,5\%$ mehr als die mit AVRORA+ ermittelten $406,83mJ$.

3.8 Zusammenfassung

Ziel dieses Kapitels war es, die Genauigkeit der Energiebedarfsbestimmung des Simulators AVRORA im Vergleich zu Messungen in SANDbed zu evaluieren. Hierzu wurden ausgehend von Messungen einer Testanwendung und der detaillierten Analyse der einzelnen Hardwarekomponenten des Sensorknotens MICAz drei Verbesserungsmöglichkeiten des Energiemodells in AVRORA erarbeitet:

- Kalibrierung des Energiemodells
- Simulation von Hardwarevarianzen
- Einfügen der Kosten der Zustandsübergänge

Diese wurden in einer verbesserten Version des Simulators, AVRORA+, umgesetzt und die so erreichten Verbesserungen evaluiert.

Mit AVRORA+ steht eine wichtige Verbesserung der Energiebedarfsbestimmung von MICAz-Sensorknoten zur Verfügung. Während Simulationen in AVRORA bei der Ausführung der Testanwendung eine Abweichung im Energiebedarf von bis zu 15% aufweisen, konnte diese Abweichung mit AVRORA+ auf unter 1% verbessert werden. Zudem ist es mit AVRORA+ nun möglich, die Varianzen im Energiebedarf zwischen Sensorknoten des gleichen Typs abzubilden. Die fehlenden Kosten von Zustandsübergängen der Hardware wurden mit Hilfe von Messungen in SANDbed bestimmt und in das Energiemodell von AVRORA+ eingearbeitet.

Mit AVRORA+ steht somit ein Simulationswerkzeug bereit, mit dem es möglich ist, vergleichbare Evaluierungen des Energiebedarfs von MICAz-Sensorknoten durchzuführen, ohne explizit zeitaufwendige Experimente in SAND-

bed durchführen zu müssen. AVRORA+ ist daher eine sinnvolle Ergänzung zu SANDbed.

4. Evaluierung des Energiebedarfs von Medienzugriffsverfahren

Im Rahmen des FleGSens-Projekts war es notwendig, ein Medienzugriffsverfahren mit Duty-Cycling zu nutzen, um den Energiebedarf der Funkschnittstelle im Vergleich zu reinem IEEE 802.15.4 zu verringern. Ohne dieses Duty-Cycling konnte die Mindestlaufzeit von 7 Tagen nicht erreicht werden. Weiterhin war eine Anforderung, dass die Anzeige eines Übertritts spätestens 5 Sekunden nach der Beendigung des Übertritts an der Basisstation angezeigt werden musste. Dies bedeutet, dass neben dem Energiebedarf auch die Latenz bei der Multi-Hop Nachrichtenübertragung minimiert werden musste. Die Auswahl und die Evaluierung von Energiebedarf und Latenz von Medienzugriffsverfahren gestaltete sich schwierig, da eine Evaluierung des Energiebedarfs weder in den Simulationen noch im eigentlichen Prototyp möglich war.

Ziel dieses Kapitels ist die Evaluierung des Energiebedarfs von Medienzugriffsverfahren in drahtlosen Sensornetzen. Hierfür werden anhand von einem einfachen Szenario und Experimenten in SANDBed und Avrora+ der Energiebedarf und die Latenz verschiedener Medienzugriffsverfahren analysiert.

Das Kapitel gliedert sich wie folgt: Im ersten Teil des Kapitels werden die Anforderungen an die Evaluierung des Energiebedarfs von Medienzugriffsverfahren vorgestellt. Anschließend wird die am Institut entworfene Programmierschnittstelle EEMAC (**E**nergie-**E**ffiziente **MAC**), sowie die damit nutzba-

ren Medienzugriffsverfahren analysiert. EEMAC wird in dieser Arbeit als Abstraktionsschicht verwendet, um eine einheitliche Programmierschnittstelle auf verschiedene in TinyOS implementierte Medienzugriffsverfahren zu erhalten. Im zweiten Teil des Kapitels wird der Energiebedarf einer Testanwendungen unter Verwendung verschiedener Medienzugriffsverfahren untersucht. Die Ergebnisse der Evaluierung führen hierbei zu überraschenden, im Vergleich zu gängigen Annahmen und Voraussetzungen aus der aktuellen Forschung widersprüchlichen Ergebnissen.

Die Ergebnisse der Evaluierung des Energiebedarfs des TinyOS-LPL Medienzugriffsverfahrens sind im Rahmen einer Veröffentlichung [8] entstanden, welche gemeinsam mit Joachim Wilke erstellt wurde. Die Ergebnisse dieser Veröffentlichung wurden für diese Arbeit erweitert, indem weitere Medienzugriffsverfahren sowie ein Vergleich zu Evaluierungsergebnissen in AVRORA+ durchgeführt wurden.

4.1 Einflussfaktoren auf den Energiebedarf einer Anwendung

Im Rahmen dieses Kapitels wird der Energiebedarf verschiedener Medienzugriffsverfahren analysiert. Hierfür ist es notwendig, die Einflussfaktoren auf den Energiebedarf einer Anwendung oder eines Protokolls zu kennen und zu untersuchen.

Ein Sensorknoten, wie beispielsweise der MicaZ Sensorknoten [1], besteht mindestens aus den Hardwarekomponenten *Funkschnittstelle* und *Prozessor*. Mit dem Prozessor ist typischerweise auch direkt *Speicher* für die Programm- bzw. Datenspeicherung verbaut. Mit Hilfe dieser Hardwarekomponenten kann eine Anwendung für drahtlose Sensornetze Daten verarbeiten und Berechnungen durchführen (Prozessor/Speicher) und gegebenenfalls mit anderen Sensorknoten Nachrichten austauschen (Funkschnittstelle). Je nach Einsatzgebiet und Anwendung können zusätzlich noch weitere Hardwarekomponenten notwendig sein, wie beispielsweise Sensoren oder Aktoren. Da die eingesetzten Sensoren und Aktoren anwendungsabhängig sind, werden sie im Rahmen dieser Analyse und der weiteren Evaluierung nicht näher betrachtet.

Wie im vorigen Kapitel erläutert wurde, kann jede der Hardwarekomponenten Prozessor und Funkschnittstelle in verschiedene Zustände versetzt werden, welche wiederum eine unterschiedliche Leistungsaufnahme aufweisen. Ist die Funkschnittstelle ausgeschaltet (Off), können keine Nachrichten gesendet oder empfangen werden, gleichzeitig ist der Energiebedarf der Funkschnittstelle jedoch mit $0,000066mW$ vernachlässigbar. Ist die Funkschnitt-

stelle eingeschaltet (Listen/Receive/Transmit), können Nachrichten gesendet oder empfangen werden, der Energiebedarf ist jedoch mit $25,34 - 60,62mW$ deutlich höher.

Für die energie-effiziente Nutzung der Funkschnittstelle ist es notwendig, diese so häufig wie möglich auszuschalten. Da eine Kommunikation zwischen zwei Sensorknoten nur möglich ist, wenn beide Sensorknoten die Funkschnittstelle gleichzeitig angeschaltet haben (Wachphase), ist es im Allgemeinen notwendig, die Wachphasen benachbarter Sensorknoten zu koordinieren. Diese Koordination, im Folgenden Duty-Cycling Strategie genannt, wird in drahtlosen Sensornetzen typischerweise zusammen mit dem Medienzugriffsverfahren implementiert.

Der konkrete Energiebedarf der Hardware, kombiniert mit der Duty-Cycling Strategie und der Charakteristik der von der Anwendung zu sendenden bzw. empfangenden Nachrichten hat Einfluss auf den Energiebedarf der Anwendung. Beispielsweise ist es einsichtig, dass im Allgemeinen eine Erhöhung der zu versendenden Datenmenge mehr Energiebedarf verursacht, da die Funkschnittstelle länger oder häufiger in den Sendezustand versetzt werden muss.

Das Duty-Cycling des Prozessors ist üblicherweise ein Teil des eingesetzten Betriebssystems. Beispielsweise wird im Betriebssystem TinyOS der Prozessor immer automatisch in einen energie-effizienteren Schlafzustand versetzt, sobald der Prozessor nicht mehr für Berechnungen benötigt wird.

Eine vereinfachte Darstellung der Einflussfaktoren auf den Energiebedarf einer Anwendung ist in Abbildung 4.1 abgebildet: Eine *Anwendung* für drahtlose Sensornetze beeinflusst durch die zu versendenden oder die zu empfangenden Nachrichten den Energiebedarf des Medienzugriffs. Das Medienzugriffsverfahren steuert hierbei über die jeweilige implementierte Duty-Cycling Strategie des *Medienzugriffsverfahrens* die Hardwarezustände der *Funkschnittstelle*. Wie im weiteren Verlauf des Kapitel gezeigt wird, spielt neben der Sendehäufigkeit auch die Menge an Daten eine Rolle im Bezug auf den Energiebedarf. Führt die Anwendung zusätzlich noch *Berechnungen* durch, hat dies über das Duty-Cycling des Betriebssystems einen Einfluss auf den Energiebedarf des *Prozessors*.

4.2 EEMAC-Programmierschnittstelle

Um den Einfluss verschiedener Medienzugriffsverfahren auf den Energiebedarf einer Anwendung zu untersuchen ist es notwendig, unterschiedliche Duty-Cycling Strategien zu vergleichen. Ein Übersicht über grundlegende, in der Forschung häufig eingesetzte Medienzugriffsverfahren und deren Duty-Cycling Strategie wurde bereits im Kapitel 2 gegeben. Ein Problem

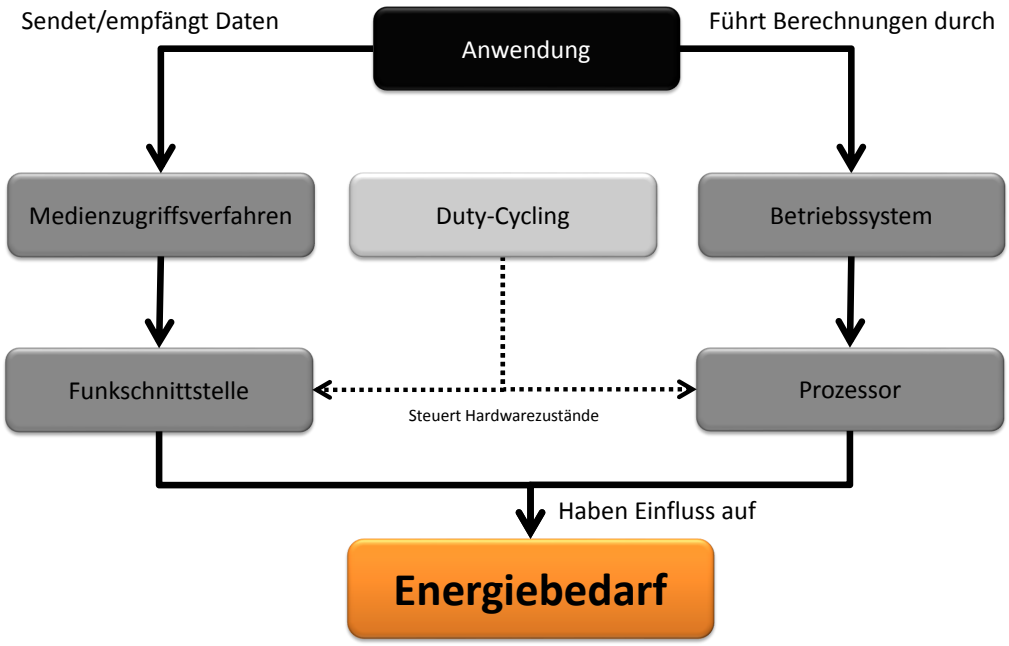


Abbildung 4.1 Vereinfachte Darstellung der Einflussfaktoren auf den Energiebedarf einer Anwendung.

hierbei ist häufig das Fehlen von Implementierungen für eine aktuelle Version des Betriebssystem TinyOS. Lediglich TinyOS Low-Power-Listening als Quasi-Standard Duty-Cycling Medienzugriffsverfahren ist in einer aktuellen Version verfügbar. Weiterhin existiert keine einheitliche Schnittstelle zur Nutzung verschiedener Medienzugriffsverfahren. Dies bedeutet, dass eine Anwendung je nach zu nutzendem Medienzugriffsverfahren unterschiedliche Schnittstellen zum Senden und Empfangen von Nachrichten nutzen und auch gleichzeitig die passenden Parameter für die Duty-Cycling Strategie setzen müsste.

Im Rahmen dieser Arbeit wird daher die (**E**nergie-**E**ffiziente **MAC**) Programmierschnittstelle als einheitliche Abstraktionsschicht für Medienzugriffsverfahren genutzt, welche gemeinsam mit Joachim Wilke entwickelt [60] wurde.

4.2.1 Aufbau und Nutzung der EEMAC-Schnittstelle

Die grundlegende Schichtenarchitektur bei Verwendung der EEMAC (**E**nergie-**E**ffiziente **MAC**) Programmierschnittstelle ist in Abbildung 4.2 abgebildet.

Sie bietet einer Anwendung eine einheitliche Schnittstelle zum *Versenden* und *Empfangen* von Nachrichten an.

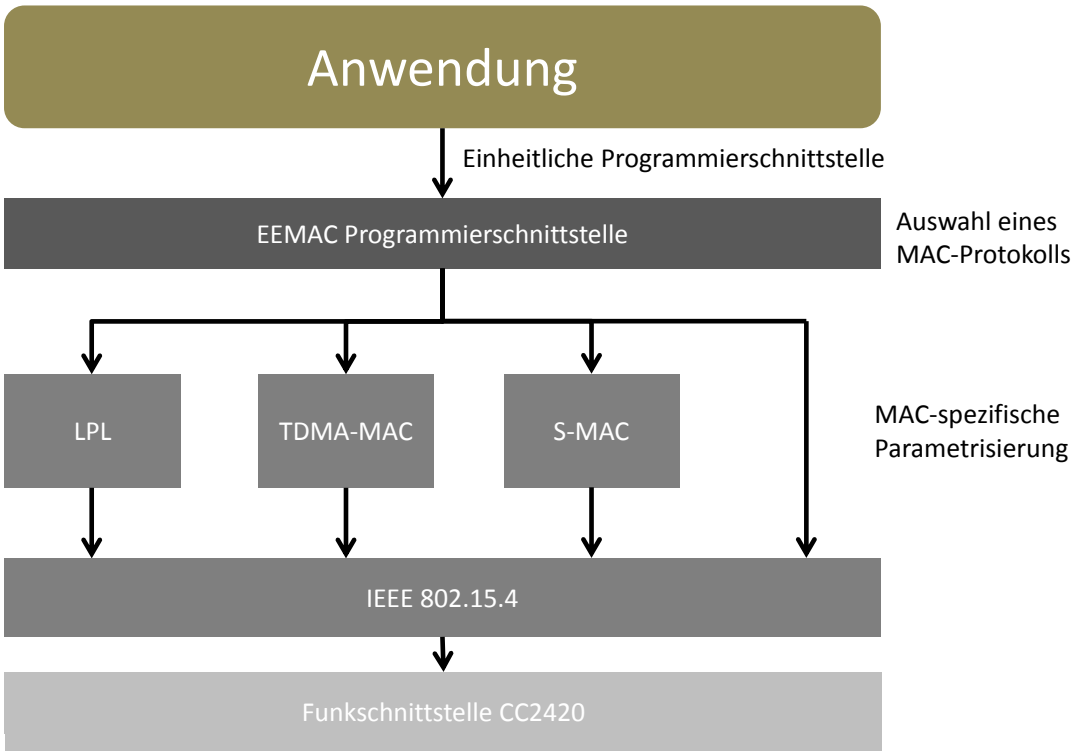


Abbildung 4.2 EEMAC-Programmierschnittstelle und die implementierten und genutzten Medienzugriffsverfahren.

Bei der Nutzung verschiedener Medienzugriffsverfahren hat es sich als Problem herausgestellt, dass die unterschiedlichen Verfahren jeweils andere Parameter und Einstellungsmöglichkeiten für das Duty-Cycling anbieten. Beispielsweise wird der Duty-Cycle in S-MAC und TDMA-MAC jeweils in Prozent angegeben, bei TinyOS Low-Power-Listening wird das konkrete Aufwachintervall eines Knotens eingestellt. Um die Nutzung verschiedener Medienzugriffsverfahren zu vereinfachen, wurden die Parameter und deren Einstellungsmöglichkeiten an einer zentralen Stelle in der EEMAC-Programmierschnittstelle gebündelt und dort zusammengefasst.

Die Auswahl von Medienzugriffsverfahren und deren Parametrisierung erfolgt in EEMAC über das Setzen von Umgebungsvariablen, welche beim Übersetzen der Anwendung automatisch von der EEMAC- Programmierschnittstelle ausgewertet und umgesetzt werden. Die Auswahl und Parametrisierung der Medienzugriffsverfahren ist somit unabhängig von der Anwendung. Somit kann eine Anwendung mit unterschiedlichen Medienzugriffsverfahren und Parametrisierungen genutzt werden, ohne die Anwendungsimplementierung zu ändern.

In der aktuellen Version der EEMAC-Programmierschnittstelle sind vier Medienzugriffsverfahren verwendbar, im Einzelnen sind dies IEEE 802.15.4, TinyOS-LPL, S-MAC und TDMA-MAC.

IEEE 802.15.4

Das Medienzugriffsverfahren IEEE 802.15.4 ist bei vielen drahtlosen Sensorknoten auf Schicht 1 und 2a des ISO/OSI-Layers bereits in der Hardware der Funkschnittstelle fest implementiert. Dies bedeutet, dass bei der Verwendung des Sensorknotens der Medienzugriff immer mit IEEE 802.15.4 durchgeführt werden muss. In TinyOS wird standardmäßig der *Non-Beacon Enabled* Modus von IEEE 802.15.4 verwendet, dies bedeutet, es gibt keinen zentralen Koordinator, der Sende- oder Empfangszeiten verteilt. Die Funkschnittstelle befindet sich daher immer in eingeschaltetem Zustand.

TinyOS Low-Power-Listening

TinyOS Low-Power-Listening ist das Quasi-Standard Medienzugriffsverfahren mit Duty-Cycling in TinyOS. Die Funktionsweise und die Parametrisierungen des TinyOS LPL-Protokolls wurden in einer Studienarbeit in SANDbed getestet und analysiert [61]. Die im folgenden verwendeten Parameter für die Dauer des periodischen Abhörens des Mediums (t_{CCA}) basieren auf den Ergebnissen dieser Studienarbeit.

S-MAC

Die in dieser Arbeit verwendete Implementierung von S-MAC ist im Rahmen einer Diplomarbeit entstanden [62]. Eine Implementierung des S-MAC Proto-

kolls existierte für die Mica- Sensorknotenplattform unter Verwendung von TinyOS 1. Da beim Entwurf von TinyOS 2.1 die Architektur und die Schnittstellen des Betriebssystems zahlreichen Änderungen unterworfen wurden und die Mica Plattform keine zeitgemäße Sensorknotenplattform darstellt, musste S-MAC auf die aktuelle Version TinyOS 2.1.2 unter Verwendung des MicaZ-Sensorknotens portiert werden.

TDMA-MAC

Die in dieser Arbeit verwendete Implementierung des TDMA-MAC Medienzugriffsverfahrens ist im Rahmen einer Diplomarbeit entstanden [63]. TDMA-MAC führt einen periodischen, synchronen Schlaf-Wachzyklus der Funkschnittstelle durch. Nachrichteneinheiten werden grundsätzlich nur in den Wachphasen gesendet oder empfangen. Will eine Anwendung Nachrichten während der Schlafphasen versenden, werden diese zwischengespeichert und in der nächsten Wachphase versendet. Bei dieser TDMA-MAC Implementierung wird davon ausgegangen, dass die Knoten synchron eingeschaltet werden. Es wird keine Synchronisierung der Schlafzyklen wie beispielsweise bei S-MAC durchgeführt. TDMA-MAC dient daher dazu, die maximal mit einem synchronen Medienzugriffsverfahren zu erreichende Energie-Effizienz zu untersuchen.

Je nach Medienzugriffsverfahren existieren in EEMAC verschiedene Parametrisierungsmöglichkeiten. Es ergeben sich folgende Parameter mit Einfluss auf den Energiebedarf:

- **IEEE 802.15.4:** Bei dem in TinyOS genutzten Non-Beacon Modus von IEEE 802.15.4 gibt es keine Parameter, die einen Einfluss auf den Energiebedarf haben. Die Funkschnittstelle ist dauerhaft eingeschaltet, es existiert kein Duty-Cycling.

- **TinyOS-LPL:** In TinyOS-LPL ergeben sich nach [61] verschiedene Parameter, die einen Einfluss auf den Energiebedarf haben. Zum Einen kann das Aufwachintervall der Funkschnittstelle eingestellt werden (t_{LPL}). Weiterhin kann die Dauer der Überprüfung des Mediums auf Aktivität (t_{CCA}) verändert werden. In der Praxis hat sich ein Wert von 2400 Prozessorzyklen für t_{CCA} herausgestellt, da nur so Nachrichtenverluste aufgrund einer zu kurzen Überprüfung des Mediums ausgeschlossen werden[6]. Dies entspricht einer Überprüfungsdauer von ungefähr $12ms$.

- **S-MAC:** Bei S-MAC kann über den Parameter t_{DC} der S-MAC Duty-Cycle in Prozent angegeben werden. Dies entspricht dem Verhältnis der

[6] Verschiedene Werte für t_{CCA} wurden in der Studienarbeit [61] untersucht. Ein Wert von 2400 lieferte in den Untersuchungen den geringsten Energiebedarf bei keinen Nachrichtenverlusten.

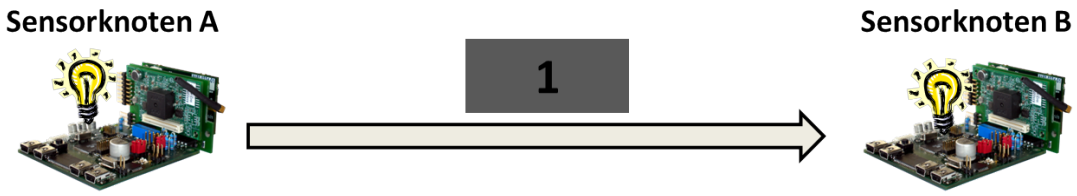


Abbildung 4.3 Ablauf der PING-Anwendung.

Wachphase zur Schlafphase. Aufgrund der in S-MAC integrierten Synchronisierung mit Hilfe von Zeitplänen kann nicht garantiert werden, dass der eingestellte Duty-Cycle dem tatsächlichen Duty-Cycle entspricht. Speziell in Multi-Hop Umgebungen kann es vorkommen, dass einige Knoten mehreren unterschiedlichen Zeitplänen folgen und somit häufiger als eigentlich durch t_{DC} eingestellt aufwachen müssen. t_{DC} ist somit lediglich eine untere Schranke für den Duty-Cycle.

- **TDMA-MAC:** In der hier verwendeten TDMA-MAC Implementierung existieren zwei Parametrisierungsmöglichkeiten. Es kann die sogenannte Slot-Länge, die Wachphase des Sensorknotens, eingestellt werden. Weiterhin kann auch die Dauer der Schlafphase, die Frame-Länge, verändert werden. Aus dem Verhältnis von Wach- zu Schlafphase lässt sich der Duty-Cycle t_{DC} berechnen. Da alle Sensorknoten durch das gleichzeitige Anschalten synchronisiert dem gleichen Schlaf-Wachzyklus folgen, entspricht der eingestellte Duty-Cycle dem tatsächlichen Duty-Cycle.

4.3 Entwurf und Implementierung der PING-Anwendung

In Rahmen dieses Kapitels wird der Energiebedarf der verschiedenen Medienzugriffsverfahren anhand einer Testanwendung evaluiert werden. Die Testanwendung wird im folgenden *PING-Anwendung* genannt. Mit Hilfe der sehr einfachen PING-Anwendung kann der Zusammenhang zwischen Duty-Cycling, Latenz und dem Energiebedarf einer Anwendung verdeutlicht werden.

4.3.1 Implementierung und Ablauf der PING-Anwendung

Der grundlegende Ablauf der PING-Anwendung ist in Abbildung 4.3, die genutzten Versuchsparameter in Tabelle 4.1 zu sehen.

Die PING-Anwendung versendet eine Nachrichteneinheit (1) zwischen zwei Sensorknoten per Unicast. Die PING-Anwendung ist für das Betriebssystem TinyOS 2.1.2 erstellt worden. Das Duty-Cycling des Betriebssystems wurde dabei in den Standardeinstellungen verwendet. Dies bedeutet, das Betriebs-

system versucht, sofern es die Anwendungsausführung zulässt, den Prozessor so häufig wie möglich in den Schlafzustand zu versetzen. Wie im vorigen Kapitel wird als Sensorknoten der MICAz-Sensorknoten verwendet.

Die Evaluierung wird sowohl in SANDbed als auch in AVRORA+ durchgeführt. Dabei werden die Stromstärke I und die Spannung U während der Durchführung der Experimente gemessen. In SANDbed wird hierfür eine Samplingrate von 10 kHz verwendet, die sich in Voruntersuchungen [8] als guter Kompromiss zwischen Messgenauigkeit und der Menge an zu speichernden Evaluationsdaten herausgestellt hat[7].

In SANDbed werden die Sensorknoten an den SNMDs 2030 und 2033 verwendet. Alle Experimente werden mit 2 x 50 Wiederholungen durchgeführt, wobei die Rolle des sendenden Sensorknotens nach den ersten 50 Wiederholungen getauscht wird.

Durch die insgesamt 100 Experimentwiederholungen soll gewährleistet werden, dass statistisch aussagekräftige Ergebnisse erzielt werden. Das Tauschen der Rollen ergibt sich aus den Messungen des vorigen Kapitels, in denen gezeigt werden konnte, dass die Sensorknoten des Testbeds starke Hardwareschwankungen im Bezug auf den Energiebedarf aufweisen. Weiterhin wird vermutet, dass die Rolle des Sensorknotens (Sender oder Empfänger) zu einem unterschiedlichen Energiebedarf je nach eingesetztem Medienzugriffsprotokoll führt. Um weitere Einflüsse der Medienzugriffsverfahren zu untersuchen, werden die Ergebnisse je nach Rolle getrennt ausgewertet.

4.3.2 Parametrisierung der Medienzugriffsverfahren

Zur Untersuchung des Energiebedarfs verschiedener Medienzugriffsverfahren wird die EEMAC-Programmierschnittstelle genutzt. Somit werden die Medienzugriffsverfahren IEEE 802.15.4, TinyOS-LPL, TDMA-MAC und S-MAC evaluiert. Die Experimente werden auf Kanal 25 des IEEE 802.15.4 Standards durchgeführt, welcher sich in Voruntersuchungen als nahezu störungsfrei herausgestellt hat. Ein Einfluss durch beispielsweise WLAN während der Durchführung der Experimente konnte so möglichst klein gehalten werden. Die eingesetzten Sensorknoten senden mit maximaler Sendeleistung 31. Der IEEE 802.15.4 Standard bietet einen Versand von Nachrichteneinheiten mit bis zu 128 Byte Payload an. Nach Abzug der notwendigen Headerfelder für den Medienzugriff per 802.15.4 ist es mit der EEMAC-Programmierschnittstelle möglich, bis zu 100 Byte an Daten pro Nachrichteneinheit zu verschicken. Im Rahmen der Evaluierung des Energiebedarfs werden verschiedene Payload-Größen verwendet, um den Einfluss der Größe der Nach-

[7] Die Evaluationsdaten der Experimente in diesem Kapitel haben zusammen einen Umfang von etwa 800 GB

Parameter	Wert
Knotenplattform	MICAz
Betriebssystem	TinyOS 2.1.2
Simulator	AVRORA+
Testbed	SANDbed
Anzahl gesendeter Nachrichteneinheiten	1
Messdauer	10 Sekunden
Gemessene Parameter	Stromstärke I und Spannung U
Messrate des SNMD	10kHz
Genutzte Sensorknoten	2030 und 2033 im SANDbed MN5
Experimentwiederholungen	2 x 50
MAC Protokolle	802.15.4 im Non-Beacon Mode, TinyOS-LPL, S-MAC und TDMA-MAC
Kanal	25
Sendeleistung	31
Größe einer Nachrichteneinheit	1 Byte und 100 Byte

Tabelle 4.1 Parameter der Evaluierung der PING-Anwendung.

richteneinheiten auf den Energiebedarf zu untersuchen. Deswegen werden sowohl Nachrichteneinheiten der minimalen Größe (1 Byte) als auch Nachrichten mit maximaler Größe (100 Byte) versendet.

Die Parametrisierung der EEMAC-Programmierschnittstelle ist in Tabelle 4.2 aufgelistet. Für TinyOS-LPL ist die Dauer einer Überprüfung des Mediums (t_{CCA}) nach [61] auf 2400 Prozessorzyklen eingestellt, was einer Dauer von etwa 12ms entspricht. Weiterhin werden Werte für das Aufwachintervall zwischen 50ms und 4000ms in den Experimenten verwendet. Für TDMA-MAC und S-MAC werden Duty-Cycles zwischen 1% und 20% untersucht[8].

[8]Durch Voruntersuchungen wurden die Wertebereiche für t_{LPL} und t_{DC} auf die interessanten Bereiche eingeschränkt. Eine Erhöhung des Duty-Cycles über 20% führt in Multi-Hop Szenarien zu

Parameter	Wert
CCA Länge t_{CCA}	2400
Aufwachintervall t_{LPL}	50, 100, 250, 500, 750, 1000, 1250, 1500, 2000, 2500, 3000, 3500, 4000 ms
S-MAC Duty-Cycle t_{DC}	1, 2, 4, 8, 12, 16, 20 %
TDMA-MAC Duty-Cycle t_{DC}	1, 2, 4, 8, 12, 16, 20 %

Tabelle 4.2 Experimentparameter der EEMAC-Programmierschnittstelle.

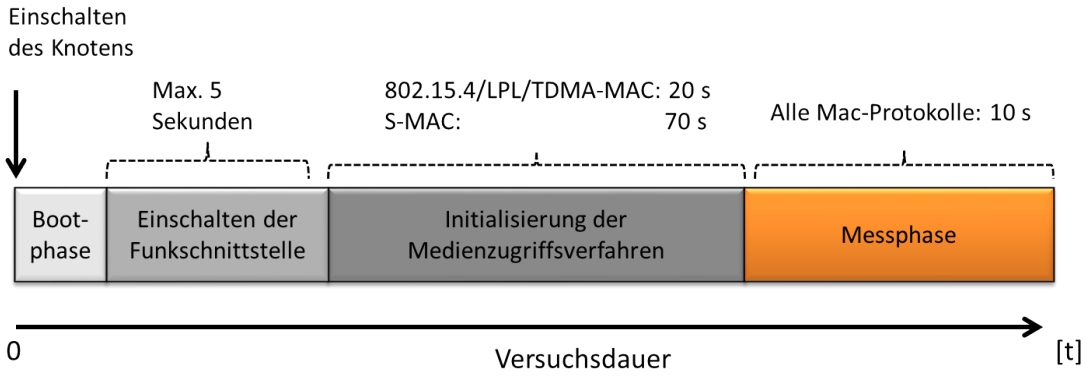


Abbildung 4.4 Ablauf eines Experiments mit der PING-Anwendung.

4.3.3 Ablauf eines Experiments

Der Ablauf eines Experiments mit der PING-Anwendung ist in Abbildung 4.4 dargestellt. Die Sensorknoten werden gleichzeitig eingeschaltet. Die Funkschnittstelle wird bei der Verwendung von IEEE 802.15.4, TinyOS-LPL und S-MAC danach zu einem zufälligen Zeitpunkt, nicht später als 5 Sekunden nach Start des Knotens, eingeschaltet. Dies vermeidet Synchronisationseffekte, die durch das gleichzeitige Einschalten der Sensorknoten auftreten könnten. Der Zeitraum von maximal 5 Sekunden entspricht dem maximalen Abstand zweier Schlafphasen der untersuchten Medienzugriffsverfahren. Da TDMA-MAC über keinen Synchronisierungsmechanismus verfügt, wird die Funkschnittstelle bei TDMA-MAC synchron 5 Sekunden nach Start des Knotens eingeschaltet.

Nach Einschalten der Funkschnittstelle erfolgt eine Wartezeit von 20 bzw. 70 Sekunden. Diese Wartezeit dient zum Einen dazu, Messungenauigkeiten nach dem direkten Einschalten eines Knotens zu vermeiden als auch dazu,

einer nahezu dauerhaft eingeschalteten Funkschnittstelle und ist somit in der Praxis oft nicht mehr sinnvoll.

bei S-MAC die Zeitpläne zu etablieren. Das Erstellen und Austauschen dieser Zeitpläne dauert bei einem Duty-Cycle von $t_{DC} = 1\%$ im schlechtesten Fall etwa 65 Sekunden. Die Ausführung der PING-Anwendung startet erst nach der Initialisierung der Medienzugriffsverfahren. Trotz der unterschiedlichen Initialisierungszeiten der Medienzugriffsverfahren lassen sich die Ergebnisse für das Versenden der Nachricht somit direkt vergleichen.

Hierauf folgt die eigentliche Messphase, in der die PING-Anwendung eine Nachrichteneinheit von 1 Byte bzw. 100 Byte versendet. Innerhalb dieser Messphase versendet die PING-Anwendung zu einem zufälligen Zeitpunkt genau eine Nachrichteneinheit. Der Sendezeitpunkt wird dabei je nach Parametrisierung des Medienzugriffsverfahrens so gewählt, dass eine Nachrichtenübertragung garantiert durchgeführt werden kann. Beispielsweise beträgt die maximal mit S-MAC gemessene Latenz bei einem Duty-Cycle von 1% etwa 7 Sekunden. Um eine erfolgreiche Nachrichtenübertragung innerhalb des Messzeitraums durchführen zu können, muss die Messphase ausreichen lang gewählt werden.

4.4 Evaluierung des Energiebedarfs

In diesem Abschnitt wird die Detailanalyse des Energiebedarfs der einzelnen Medienzugriffsverfahren durchgeführt. Zu Beginn wird die Funktionsweise der verschiedenen Medienzugriffsverfahren anhand von Messungen in SANDbed gegenübergestellt. Anschließend wird der Energiebedarf der einzelnen Medienzugriffsverfahren evaluiert.

Gegenüberstellung der Medienzugriffsverfahren

Abbildung 4.5 stellt die verschiedenen Medienzugriffsverfahren der EEMAC-Programmierschnittstelle bei der Ausführung der PING-Anwendung gegenüber. Auf der x-Achse ist die Messphase von 10 Sekunden aufgezeichnet, auf den y-Achsen ist die Leistungsaufnahme der einzelnen Medienzugriffsverfahren abgebildet. Abgebildet ist neben IEEE 802.15.4 TinyOS-LPL mit einem Aufwachintervall von $2000ms$, S-MAC mit einem Duty-Cycle von 4% und TDMA-MAC mit einem Duty-Cycle von 2%. In den einzelnen Abbildungen ist die Leistungsaufnahme jeweils für Sender- und Empfängerknoten getrennt dargestellt. Der Sendezeitpunkt der Nachrichteneinheit ist bei allen Medienzugriffsverfahren etwa $0,6$ Sekunden nach Beginn der Messung. Die Nachrichteneinheit hat eine Größe von 1 Byte.

In der Abbildung ist die Duty-Cycling Strategie und die daraus resultierende Leistungsaufnahme im direkten Vergleich ersichtlich:

Für IEEE 802.15.4 ist erkennbar, dass die Leistungsaufnahme sowohl für Sender als auch Empfänger relativ konstant bei etwa $80mW$ ist. Lediglich klei-

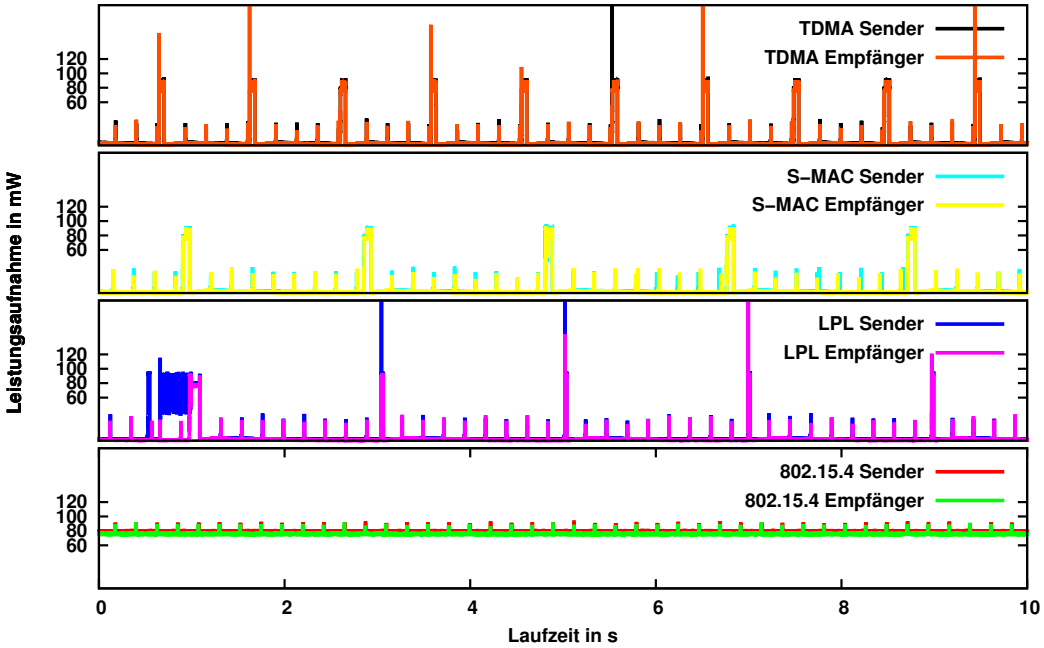


Abbildung 4.5 Leistungsaufnahme der PING-Anwendung mit den Medienzugriffsverfahren der EEMAC-Schnittstelle in SANDbed.

nere, periodische Ausschläge auf etwa $90mW$ aufgrund von Betriebssystem internen Timern sind sichtbar. Die Übertragung der Nachrichteneinheit ist aufgrund der geringen Änderungen der Leistungsaufnahme der Funkschnittstellenzustände *Listen*, *Transmit* und *Receive* nicht direkt ersichtlich.

Für TinyOS-LPL ist sowohl das periodische Aufwachintervall als auch der Versand der Nachrichteneinheit anhand der Präambel ersichtlich. Das periodische Aufwachen und der CCA-Check ist sowohl für Sender als auch Empfänger im Abstand von 2 Sekunden sichtbar, beispielsweise bei etwa Sekunde 3. Der Versand der Nachricht ist mit Hilfe der Leistungsaufnahme des Senderknotens sichtbar. Die Präambelübertragung startet bei Sekunde $0,6$ und endet, sobald der Empfängerknoten aufwacht und die Nachrichteneinheit quittiert. Ebenfalls erkennbar ist, dass nach der Übertragung der Nachricht die Aufwachzeitpunkte der beiden Sensorknoten synchronisiert sind, obwohl diese vor dem Versand der Nachricht nicht synchron waren.

Für S-MAC sind die synchronen Wachphasen mit einer Leistungsaufnahme von etwa $80mW$ im Abstand von 3 Sekunden sichtbar. Bei TDMA-MAC sind ebenfalls die synchronen Schlaf- und Wachphasen im Abstand von etwa 1 Sekunde sichtbar. Die Nachrichtenübertragung findet für beide Medienzugriffsverfahren in der ersten Wachphase statt, was aufgrund der geringen Änderungen der Leistungsaufnahme der Funkschnittstellenzustände *Listen*,

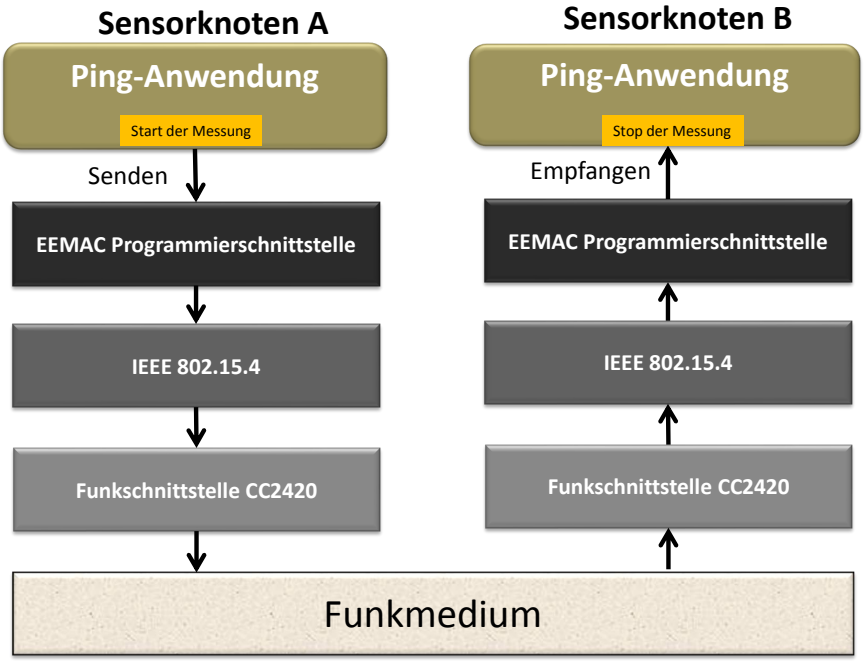


Abbildung 4.6 Latenzmessung innerhalb der PING-Anwendung.

Transmit und *Receive* nicht direkt ersichtlich ist. Auch die Synchronisation der Sensorknoten bei S-MAC anhand des Zeitplans während der Wachphasen ist nicht ersichtlich.

Aus dieser einfachen Gegenüberstellung lassen sich schon einige Vermutungen im Bezug auf den Energiebedarf ableiten. Zum Einen ist deutlich zu erkennen, dass die Leistungsaufnahme von IEEE 802.15.4 nahezu dauerhaft etwa $80mW$ beträgt, während bei den anderen Medienzugriffsverfahren unterschiedliche Leistungsaufnahmen bedingt durch das Duty-Cycling erkennbar sind.

Latenzmessung

Neben dem Energiebedarf der verschiedenen Medienzugriffsverfahren soll im folgenden auch die durch verschiedene Duty-Cycling Strategien hinzugefügte Latenz analysiert werden. Hierfür wird in den Experimenten gemessen, wann die PING-Anwendung den Sendevorgang initiiert und wann die Nachrichteneinheit beim Empfänger angekommen ist. Es soll untersucht werden, in wie weit Latenz und Energiebedarf der Medienzugriffsverfahren in Bezug zueinander stehen.

Abbildung 4.6 zeigt, zwischen welchen Zeitpunkten die Latenzmessung in den folgenden Abschnitten durchgeführt wird. Die Latenzmessung startet,

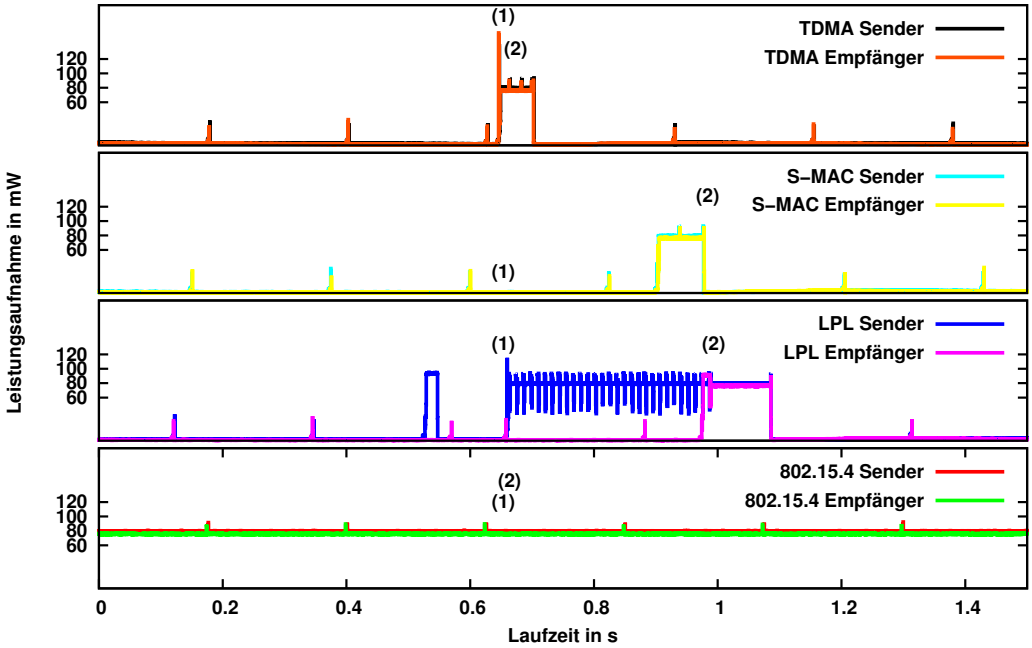


Abbildung 4.7 Detaillierte Ansicht der Leistungsaufnahme während der Übertragung der PING-Nachricht in SANDbed.

sobald die PING-Anwendung auf Sensorknoten A die Nachricht versenden will und diese über das Sende-Interface an die EEMAC-Programmierschnittstelle übergibt. Die Latenzmessung wird gestoppt, sobald die Nachricht von der PING-Anwendung auf Sensorknoten B empfangen wird. Die gemessenen Latenzen beinhalten somit sowohl die Verarbeitungszeiten der Medienzugriffsverfahren und der Funkschnittstelle sowie die Sende- bzw. Empfangszeit und die Ausbreitungsverzögerung.

Abbildung 4.7 zeigt einen detaillierteren Ausschnitt der Medienzugriffsverfahren der EEMAC- Programmierschnittstelle bei Ausführung der PING-Anwendung. Besonderer Fokus wird hierbei auf den zeitlichen Abstand vom Sendezeitpunkt der Nachrichteneinheit bis zum Empfangszeitpunkt gelegt.

Zum Zeitpunkt (1) initiiert die PING-Anwendung den Versand der Nachrichteneinheit und übergibt diese an die EEMAC-Programmierschnittstelle. Danach wird die Nachrichteneinheit in Abhängigkeit vom eingesetzten Medienzugriffsverfahren und dessen Parametrisierung übertragen. Zum Zeitpunkt (2) hat der Empfänger die Nachrichteneinheit erfolgreich empfangen. Je nach Medienzugriffsverfahren und Parametrisierung liegen der Sende- und der Empfangszeitpunkt, die Latenz, unterschiedlich weit auseinander.

Da bei IEEE 802.15.4 die Funkschnittstelle dauerhaft eingeschaltet ist, kann die Nachrichteneinheit sofort übertragen werden, die Latenz beträgt in etwa 3-4 ms. Der Unterschied im Energiebedarf beim Versenden einer Nachricht im Vergleich zu der eingeschalteten Funkschnittstelle bzw. zum Empfang einer Nachricht beträgt zwischen 3 und 4 mW.

Bei der Verwendung von TinyOS-LPL startet der Empfänger bei (1) das Versenden der Präambel. Wacht der Empfängerknoten auf, (2), erfolgt die Nachrichtenübertragung mit anschließender Bestätigung. Je nach Aufwachzeitpunkt des Empfängerknotens kann die Latenz bis zu einem Aufwachintervall betragen, in der abgebildeten Konfiguration maximal 4 Sekunden. In dieser Messung ist ebenfalls zu erkennen, dass der Sendeknoten durch die Präambel (zwischen (1) und (2)) eine deutlich höhere Leistungsaufnahme hat als der Empfängerknoten.

In S-MAC und TDMA-MAC wird die Nachrichteneinheit falls nötig so lange zwischengespeichert, bis der Duty-Cycle dazu führt, dass sowohl Sender als auch Empfängerknoten gleichzeitig wach sind. Diese Wachphasen sind bei beiden Medienzugriffsverfahren durch die nahezu gleiche Leistungsaufnahme zu erkennen. In der hier abgebildeten Messung ist dies bei S-MAC zu erkennen, Sendezeitpunkt (1) und eigentlicher Nachrichtenübertragung (2) liegen deutlich auseinander. Bei TDMA-MAC fällt der Sendezeitpunkt in dieser Messung auf eine Wachphase beider Knoten, die Nachricht muss nicht zwischengespeichert werden und kann sofort versendet werden.

Vergleich SANDbed und AVRORA+

In den folgenden Abschnitten werden jeweils der Energiebedarf der PING-Anwendung unter Verwendung verschiedener Medienzugriffsverfahren verglichen. Als Werkzeuge zur Bestimmung des Energiebedarfs kommt neben SANDbed auch AVRORA+ zum Einsatz. In AVRORA+ werden in jedem Simulationslauf Sensorknoten mit einem Energiebedarf erzeugt, der den in SANDbed gemessenen Hardwarevarianzen entspricht. Um nun die Messergebnisse zweier Sensorknoten in SANDbed mit unterschiedlichen Simulationsläufen in AVRORA+ zu vergleichen gibt es in AVRORA+ die Möglichkeit, Sensorknoten mit minimalem und maximalem Energiebedarf zu erzeugen. Hierfür wird das Energiemodell in AVRORA+ mit den entsprechenden Werten parametrisiert. Die minimalen und maximalen Energiebedarfswerte entsprechen dabei den im gesamten SANDbed gemessenen Minimal- und Maximalwerten. Im folgenden wird dies als *AVRORA+ MIN* und *AVRORA+ MAX* bezeichnet.

Abbildung 4.8 zeigt beispielhaft Messergebnisse für einen Vergleich von Messungen des Energiebedarfs in SANDbed und AVRORA+. Abgebildet sind ne-

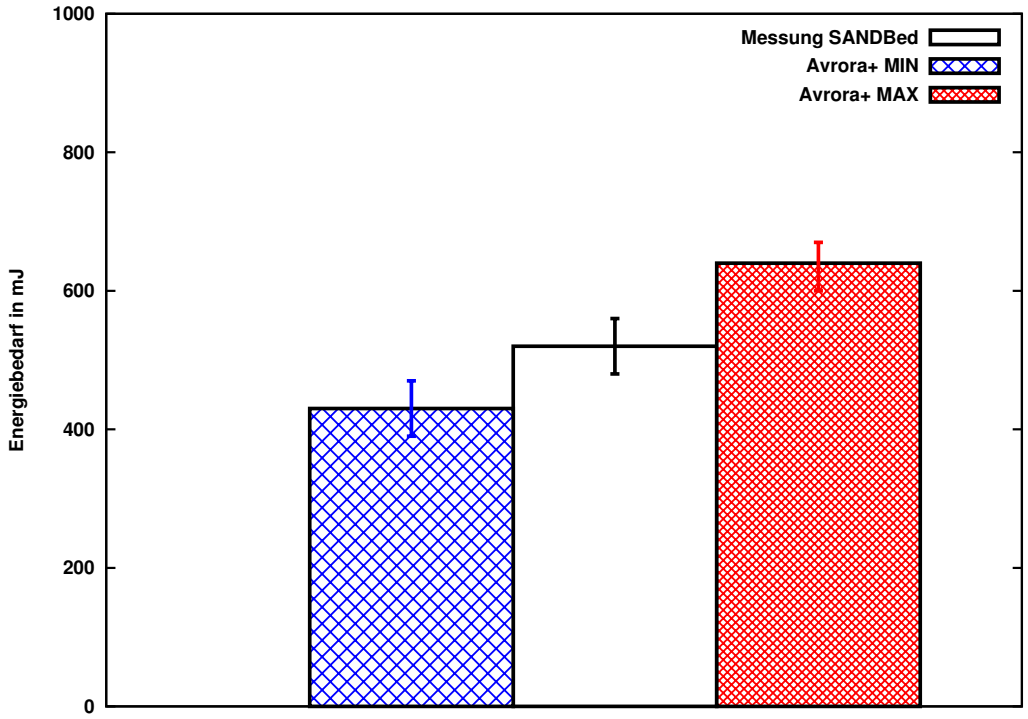


Abbildung 4.8 Beispielhafter Vergleich des Energiebedarfs SANDbed und AVRORA.

ben den durchschnittlichen Energiebedarfswerten der Sensorknoten auch die minimal und maximal gemessenen Energiebedarfswerte in den 100 durchgeführten Experimenten.

Eine wichtige Erkenntnis des Vergleichs ist es, dass die in SANDbed gemessenen Energiebedarfswerte immer größer als die Energiebedarfswerte von AVRORA+ MIN und kleiner als AVRORA+ MAX sind. In den folgenden Energiebedarfsmessungen is ersichtlich, dass AVRORA+ den Energiebedarf im Vergleich zu SANDbed in allen Experimenten korrekt abbildet.

Zusammenfassung der Evaluationsmetriken

Es werden folgende Evaluationsmetriken untersucht:

- *Energiebedarf der Medienzugriffsverfahren:* Für jedes Medienzugriffsverfahren wird der kumulierte Energiebedarf der PING-Anwendung analysiert. Hierbei wird insbesondere der Einfluss von unterschiedlichen Parametrisierungen des Duty-Cyclings der Medienzugriffsverfahren untersucht.
- *Evaluierung des Energiebedarfs verschiedener Knotenrollen:* Der Energiebedarf der PING-Anwendung bei Verwendung der unterschiedlichen Medienzugriffsverfahren wird getrennt für Sender- und Empfängerknoten analysiert.

- *Evaluierung des Energiebedarfs unterschiedlicher Nachrichtengrößen:* Für jedes Medienzugriffsverfahren wird der Einfluss der Nachrichtengröße auf den Energiebedarf der PING-Anwendung untersucht. Hierfür wird das Verhältnis des Energiebedarfs bei der Übertragung einer 100 Byte Nachrichteneinheit im Vergleich zu einer 1 Byte Nachrichteneinheit untersucht.

- *Evaluierung der Latenz:* Die Latenz der Übertragung der Nachrichteneinheit bei der Ausführung der PING-Anwendung wird für die unterschiedlichen Medienzugriffsverfahren analysiert.

Erwartungen an den Energiebedarf der PING-Anwendung

Vor der eigentlichen Evaluierung des Energiebedarfs wird in diesem Abschnitt eine Erwartungshaltung formuliert. Die Erwartungen sind hierbei Schlussfolgerungen aus der bisherigen Analyse der Medienzugriffsverfahren als auch aus den Messungen im vorigen Kapitel und der Beschreibungen der Medienzugriffsverfahren in Kapitel 2.3. Im Anschluss an die Evaluierung ist zu überprüfen, in wie weit sich die Erwartungshaltung bestätigen konnte.

- **TinyOS LPL:** Je größer der Wert für t_{LPL}, desto kleiner der Energiebedarf.

- **S-MAC:** Je kleiner der Wert für t_{DC}, desto kleiner der Energiebedarf.

- **TDMA-MAC:** Je kleiner der Wert für t_{DC}, desto kleiner der Energiebedarf.

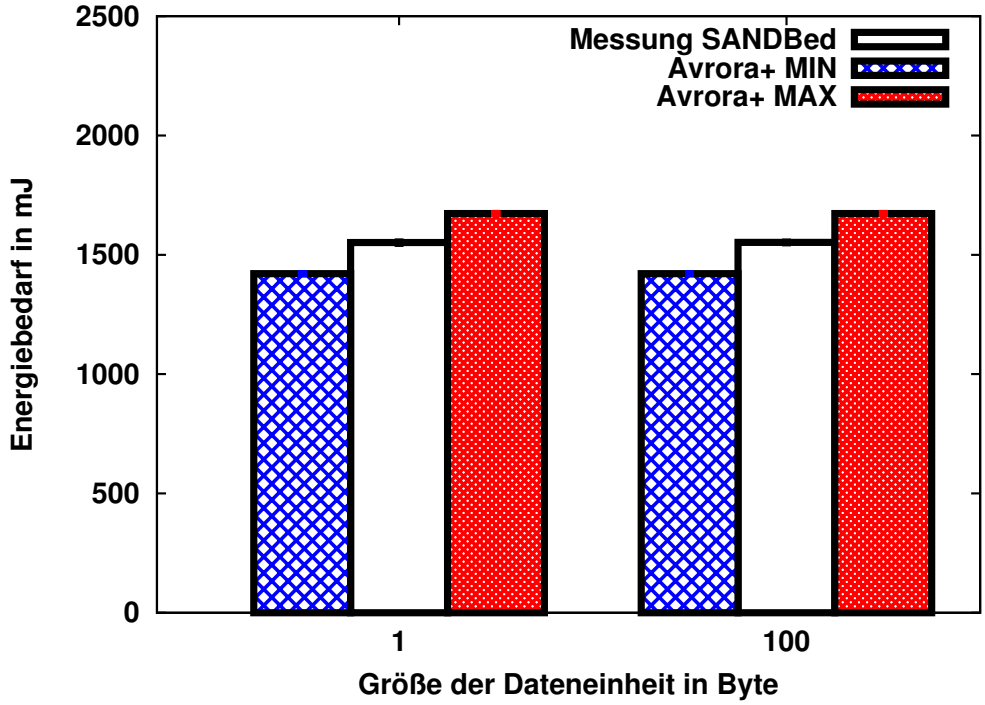
- **Größe der Nachrichteneinheiten:** Je größer die Nachrichteneinheit, desto größer der Energiebedarf.

- **Gesamtenergiebedarf:** Die Nachrichtenübertragung ist maßgeblich für den Energiebedarf in der Messphase verantwortlich.

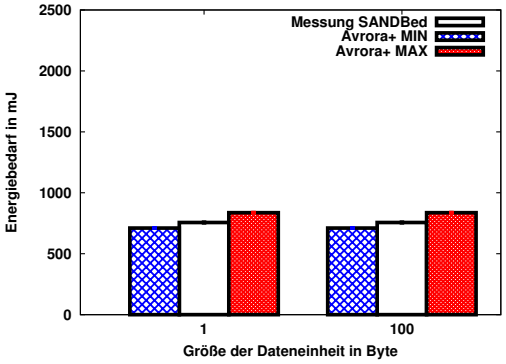
- **Rolle der Sensorknoten:** Die Rolle der Sensorknoten hat kaum Einfluss auf den Energiebedarf.

4.4.1 IEEE 802.15.4

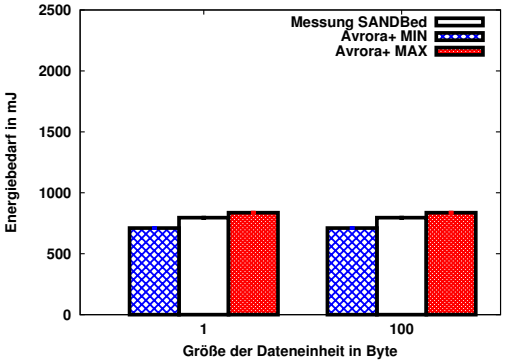
Im Rahmen dieses Abschnitts wird das Medienzugriffsverfahren IEEE 802.15.4 untersucht.



(a) Energiebedarf aller Knoten



(b) Energiebedarf des Empfängerknotens



(c) Energiebedarf des Senderknotens

Abbildung 4.9 Kumulierter Energiebedarf und Energiebedarf aufgeteilt nach Knotenrolle der PING-Anwendung mit IEEE 802.15.4 in SANDbed und AVRORA+.

Evaluierung des Energiebedarfs

Abbildung 4.9a zeigt den kumulierten Energiebedarf der PING-Anwendung unter Verwendung von IEEE 802.15.4. Der Energiebedarf ist für Nachrichteneinheiten von 1 Byte und 100 Byte jeweils für Messungen in SANDbed und AVRORA+ MIN bzw. AVRORA+ MAX auf der y-Achse dargestellt.

Für IEEE 802.15.4 wurde ein durchschnittlicher Energiebedarf von $1551,63mJ$ bei einer Nachrichteneinheit von 1 Byte und $1551,99mJ$ bei einer Nachrichteneinheit von 100 Byte in SANDbed gemessen. In AVRORA+ ergibt sich für 1 Byte Nachrichteneinheiten ein minimaler Energiebedarf von $1421,07mJ$ und ein maximaler Energiebedarf von $1673,96mJ$. Bei Nachrichteneinheiten von 100 Byte ergibt sich in AVRORA+ ein minimaler Energiebedarf von $1422,00mJ$ und ein maximaler Energiebedarf von $1673,98mJ$.

Wie bereits bei der Gegenüberstellung der Medienzugriffsverfahren festgestellt wurde, nutzt IEEE 802.15.4 im Non-Beacon Enabled Modus keine Duty-Cycle Mechanismen. Hierdurch gibt es keine Möglichkeit, den Energiebedarf über unterschiedliche Parametrisierungen des Duty-Cycles zu beeinflussen.

Evaluierung des Energiebedarfs verschiedener Knotenrollen

Abbildungen 4.9b und 4.9c stellen den Energiebedarf aufgeteilt nach Knotenrolle dar. Es ist zu erkennen, dass der Energiebedarf unabhängig von der Knotenrolle und der Größe der Nachrichteneinheit nahezu gleich ist.

Für den Senderknoten wurden bei einer Nachrichtengröße von 1 Byte durchschnittlich $775,66mJ$ in SANDbed gemessen. Für den Empfängerknoten ergibt sich ein Energiebedarf von $775,97mJ$. Bei 100 Byte Nachrichteneinheiten wurde ein Energiebedarf von $776,00mJ$ bzw. $775,99mJ$ gemessen. Für AVRORA+ ergibt sich für den Senderknoten ein minimaler Energiebedarf von $710,53mJ$ bzw. $710,54mJ$ bei einer Nachrichtengröße von 100 Byte. Für den Empfängerknoten ergeben sich Energiebedarfswerte von $710,54mJ$ bzw. $710,56mJ$. Der maximale Energiebedarf in AVRORA+ für den Senderknoten konnte für beide Nachrichtengrößen mit $836,48mJ$ gemessen werden. Für den Empfängerknoten ergeben sich maximale Energiebedarfswerte von $836,49mJ$ und $836,50mJ$.

Insgesamt lässt sich aus diesen Messergebnissen ableiten, dass es kaum einen Einfluss der Knotenrolle auf den Energiebedarf der PING-Anwendung bei IEEE 802.15.4 gibt. Sowohl in SANDbed und AVRORA+ ist der Energiebedarf in den hier durchgeführten Messungen nahezu identisch.

Theoretisch müsste der Senderknoten einen geringeren Energiebedarf aufweisen als der Empfängerknoten. Im vorigen Kapitel wurde gezeigt, dass die Funkschnittstelle im Zustand *Transmit* (Senden) eine durchschnittlich $1,32mW$ geringere Leistungsaufnahme aufweist als im Zustand *Receive* (Empfangen).

Insgesamt ergibt sich bei einer Nachrichtengröße von 1 Byte ein theoretisch verringerter Energiebedarf der PING-Anwendung von $0,00000264mJ$, bei einer Nachrichtengröße von 100 Byte $0,00000528mJ$.

Es ist anzumerken, dass der Messfehler der SNMDs zwar unter 1% liegt, der hier identifizierte theoretisch geringere Energiebedarf auf die 10 Sekunden Messzeitraum betrachtet lediglich einen Unterschied von etwa $0,0000007\%$ ausmacht [9]. Dies lässt sich mit den zur Verfügung stehenden Messinstrumenten somit kaum zuverlässig messen.Diese geringen Unterschiede im Energiebedarf konnten in den Messungen nicht immer nachgewiesen werden. Hierbei ist zu bedenken, dass äußere Einflüsse wie beispielsweise Schwankungen des Energiebedarfs aufgrund der Hardwaretoleranzen einen wesentlich größeren Einfluss auf den Energiebedarf aufweisen.

Evaluierung des Energiebedarfs unterschiedlicher Nachrichtengrößen

Für IEEE 802.15.4 wurde ein durchschnittlicher Energiebedarf von $1551,63mJ$ bei einer Nachrichteneinheit von 1 Byte und $1551,99mJ$ bei einer Nachrichteneinheit von 100 Byte in SANDbed gemessen, vergleiche Abbildung 4.9a. Sowohl in SANDbed als auch AVRORA ließ sich kaum ein messbarer Unterschied zwischen dem Versand einer 1 Byte und einer 100 Byte Nachricht feststellen.

Betrachtet man die Leistungsaufnahme der Funkschnittstelle aus Kapitel 3, so sollte ein geringerer Energiebedarf beim Versand einer größeren Nachrichteneinheit feststellbar sein. Dies ergibt sich daraus, dass die Funkschnittstelle im Zustand *Transmit* (Senden) einen geringeren Energiebedarf als im Zustand *Receive* oder *Listen* aufweist. Da sich die Funkschnittstelle bei dem Versand einer 100 Byte großen Nachricht etwa $2ms$ länger im Zustand Transmit befindet, ergibt sich rechnerisch ein geringerer Energiebedarf von $0,000009mJ$ [10].

Insgesamt kann festgestellt werden, dass sich der Energiebedarf der PING-Anwendung für unterschiedliche Nachrichtengrößen kaum unterscheidet.

Evaluierung der Latenz

In Tabelle 4.3 sind die durchschnittlichen Latenzen der Nachrichtenübertragung mit IEEE 802.15.4 in AVRORA+ aufgelistet. Das Versenden von 1 Byte führt zu einer Latenz von $4,91ms$. Diese Latenz setzt sich zusammen aus der eigentlichen Zeit für die Übertragung (etwa $2ms$) und der Verarbeitungszeit innerhalb des CC2420, TinyOS und der EEMAC-Programmierschnittstelle sowohl beim Sender als auch Empfänger. Das Versenden von 100 Byte führt zu

[9] Annahme: Durchschnittlicher Energiebedarf des Senders im Messzeitraum von 10 Sekunden $775mJ$, theoretisch geringerer Energiebedarf $0,00000528mJ$

[10] Der Unterschied zwischen Zustand Listen und Transmit beträgt $4,59mW$, der der Zeitunterschied 2 ms, somit ergibt sich ein geringerer Energiebedarf von $2ms * 4,59mW$.

Größe der Nachrichteneinheit	Durchschnittliche Latenz in ms
1 Byte	4,91
100 Byte	10,44

Tabelle 4.3 Durchschnittliche Latenz der PING-Anwendung unter Verwendung von IEEE 802.15.4 in AVRORA+.

einer Latenz von $10,44ms$. Hierbei entfallen etwa $4ms$ auf die reine Übertragung.

Die hier gemessene Latenz für die Nachrichtenübertragung mit IEEE 802.15.4 dient im weiteren Verlauf des Kapitels dazu, den Zusammenhang zwischen Duty-Cycling und Latenz zu analysieren. Da die untersuchten Medienzugriffsverfahren alle auf IEEE 802.15.4 aufbauen und dieses zum physikalischen Medienzugriff nutzen müssen, lässt sich somit genau die zusätzliche Latenz der einzelnen Medienzugriffsverfahren bzw. des dort eingesetzt Duty-Cyclings bestimmen.

4.4.2 TinyOS-LPL

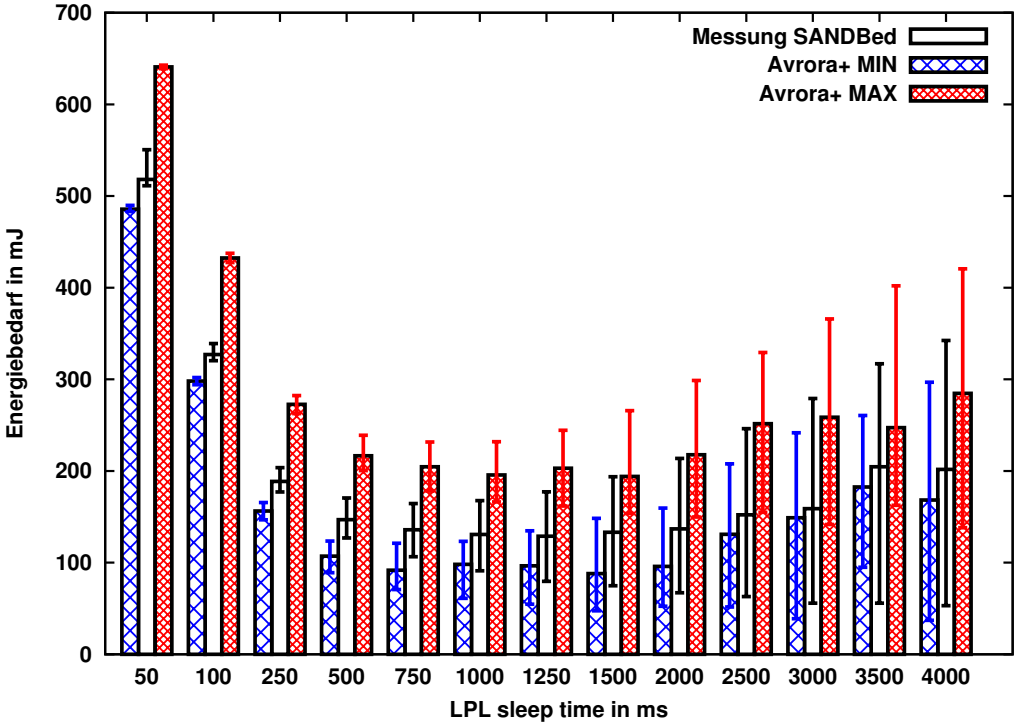
Im Rahmen dieses Abschnitts wird das Medienzugriffsverfahren TinyOS-LPL untersucht.

Evaluierung des Energiebedarfs

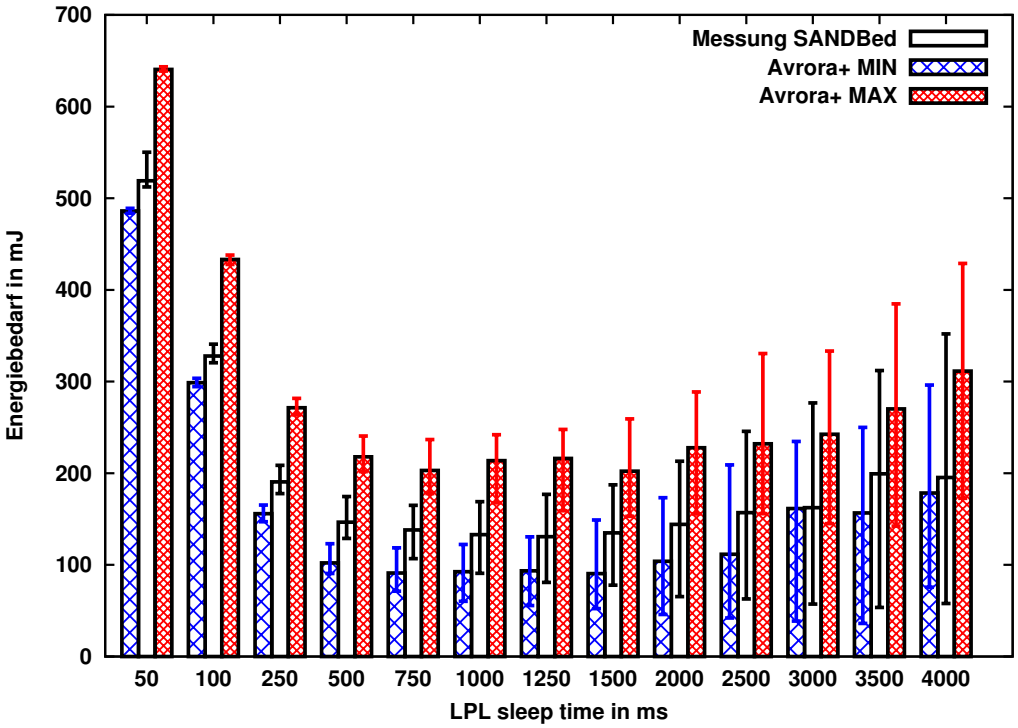
Die Abbildungen 4.10a und 4.10b zeigen den kumulierten Energiebedarf beider Sensorknoten der PING-Anwendung unter Verwendung von TinyOS-LPL für unterschiedliche Größen der gesendeten Nachrichteneinheit und für unterschiedliche Parametrisierungen von t_{LPL}.

Bei einer Größe der Nachrichteneinheit von 1 Byte lässt sich in SANDbed der größte Energiebedarf bei $t_{LPL} = 50ms$ erkennen, die PING-Anwendung weist mit dieser Parametrisierung einen Energiebedarf von $518,14mJ$ auf. Der geringste Energiebedarf von $129mJ$ konnte bei $t_{LPL} = 1250ms$ gemessen werden. Für die Werte zwischen $t_{LPL} = 50ms$ und $t_{LPL} = 1250ms$ gilt jeweils, dass eine Erhöhung von t_{LPL} zu einer Verringerung des Energiebedarfs führt. Für Werte größer als $t_{LPL} = 1250ms$ konnte bei jeder Erhöhung von t_{LPL} eine Erhöhung des Energiebedarfs festgestellt werden.

Die Energiebedarfswerte für eine Nachrichtengröße von 100 Byte unterscheiden sich quantitativ jedoch nicht qualitativ von den Energiebedarfswerten bei einer Nachrichtengröße von 1 Byte. Der größte Energiebedarf in SANDbed konnte ebenfalls bei $t_{LPL} = 50ms$ mit $518,99mJ$ gemessen werden, der geringste Energiebedarf bei $t_{LPL} = 1250ms$ mit $130,72mJ$.

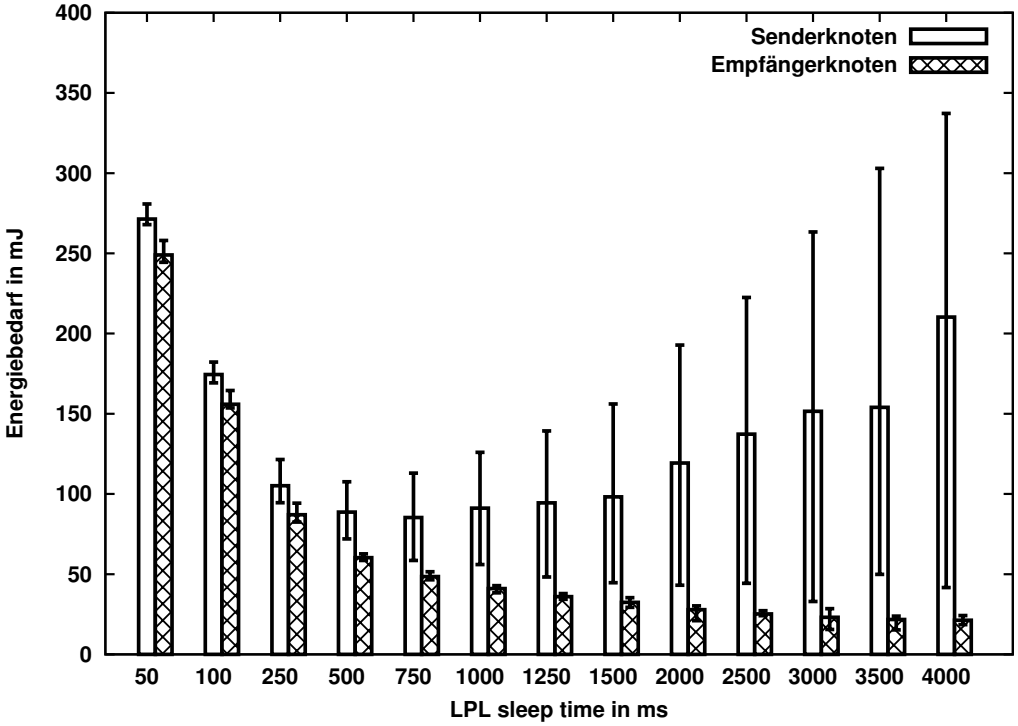


(a) Größe der Nachrichteneinheit 1 Byte

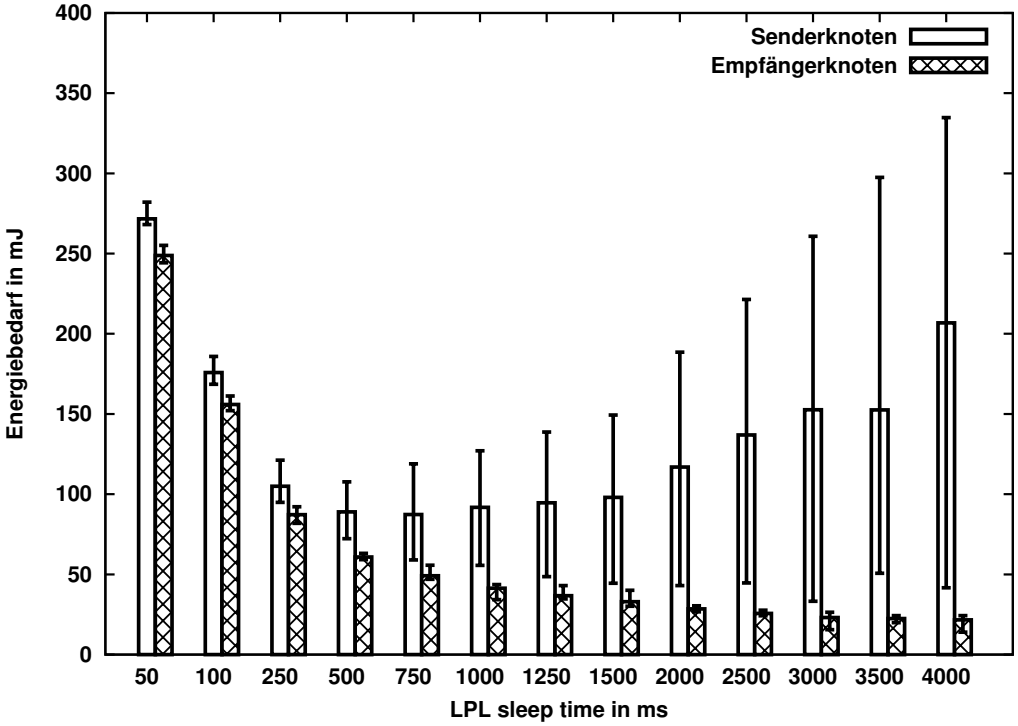


(b) Größe der Nachrichteneinheit 100 Byte

Abbildung 4.10 Kumulierter Energiebedarf der PING-Anwendung mit LPL in SANDbed und AVRORA.



(a) Größe der Nachrichteneinheit 1 Byte



(b) Größe der Nachrichteneinheit 100 Byte

Abbildung 4.11 Kumulierter Energiebedarf der PING-Anwendung mit TinyOS-LPL in SANDbed getrennt nach Sender und Empfänger.

Aus diesen Messungen lässt sich vermuten, dass für den Energiebedarf der PING-Anwendung ein optimaler Wert für t_{LPL} zwischen 1000 und 1500ms existiert. Wird ein anderer Wert für t_{LPL} verwendet, steigt der Energiebedarf. Dieser Anstieg hat zwei unterschiedliche Einflussfaktoren: Die Länge der gesendeten Präambel und die Aufwachhäufigkeit.

Für kleine Werte von t_{LPL} müssen die Sensorknoten häufiger aufwachen und das Medium überprüfen. Dies führt zu einem erhöhten Energiebedarf im Vergleich zur optimalen Einstellung. Bei großen Werten für t_{LPL} muss der Sendeknoten unter Umständen eine relativ lange Präambel senden bevor der Empfängerknoten aufwacht und die Nachrichteneinheit empfangen kann. In diesem Fall befindet sich der Senderknoten im Vergleich zur optimalen Einstellung länger im Sendezustand, was einen erhöhten Energiebedarf nach sich zieht.

Im Vergleich zu den vorigen Messungen mit IEEE 802.15.4 ist deutlich erkennbar, dass für alle Werte von t_{LPL} ein geringerer Energiebedarf gemessen wurde. Bei einem t_{LPL} Wert von 50ms ist der Energiebedarf im Vergleich zu IEEE 802.15.4 im Schnitt um Faktor 3 geringer. Der Einsatz von TinyOS-LPL führt somit in allen untersuchten Parametrisierungen zu einem geringeren Energiebedarf.

Vergleicht man den Energiebedarf der Messungen in SANDbed und in AVRORA+, so lässt sich feststellen, dass der durchschnittliche Energiebedarf wie erwartet zwischen den Energiebedarfswerten von AVRORA+ MIN und MAX liegt. Der gemessene Energiebedarf ist dabei jedoch nicht genau der Mittelwert von AVRORA+ MIN und MAX. Dies liegt daran, dass die Messungen nur auf zwei Sensorknoten in SANDbed durchgeführt wurden. Die Energiebedarfswerte in AVRORA+ wurden jedoch aus Durchschnittswerten aller in SANDbed verfügbarer Sensorknoten gewonnen. Dies führt dazu, dass die in SANDbed gemessenen Energiebedarfswerte zwar zwischen AVRORA+ MIN und MAX, nicht jedoch genau in der Mitte liegen. Der qualitative Verlauf der Energiebedarfswerte in Abhängigkeit von t_{LPL} ist jedoch sowohl in SANDbed als auch in AVRORA+ gleich.

In den Abbildungen 4.10a und 4.10b ist neben dem durchschnittlichen Energiebedarf auch der minimal und maximal in SANDbed gemessene Energiebedarf aufgezeichnet. Es ist zu erkennen, dass der minimal und maximal gemessene Energiebedarf je weiter auseinander liegt, desto größer der Wert für t_{LPL} ist. Beispielsweise beträgt der minimal in SANDbed gemessene Energiebedarf bei $t_{LPL} = 50ms$ $511,24mJ$, der maximal gemessene Energiebedarf $550,50mJ$. Bei $t_{LPL} = 4000ms$ beträgt der minimale Energiebedarf $53,13mJ$, der Maximale $342,35mJ$.

Der Grund für den wachsenden Unterschied ist der Einfluss der Präambellänge auf den Gesamtenergiebedarf für Sender- und Empfängerknoten. Dies wird im folgenden Abschnitt begründet.

Evaluierung des Energiebedarfs verschiedener Knotenrollen

Abbildung 4.11 zeigt den Energiebedarf getrennt nach Sender und Empfängerknoten für die unterschiedlichen t_{LPL} Werte. Zu erkennen ist hierbei, dass der Energiebedarf für kleine t_{LPL} Werte für Sender- und Empfängerknoten sehr nahe beieinander liegt. Beispielsweise konnte für eine Nachrichteneinheit von 1 Byte und bei $t_{LPL} = 50ms$ ein Energiebedarf des Senderknoten von $271,43mJ$, für den Empfängerknoten ein Energiebedarf von $249,04mJ$ gemessen werden. Für größere Werte von t_{LPL} ist erkennbar, dass der Unterschied zwischen dem Energiebedarf des Senders und des Empfängerknotens immer größer wird. Für $t_{LPL} = 4000ms$ ergibt sich ein Energiebedarf des Senders von $210,29mJ$, für den Empfänger ein Energiebedarf von $21,31mJ$.

Dieser Unterschied im Energiebedarf zwischen Sender- und Empfängerknoten lässt sich mit der für die Nachrichtenübertragung notwendigen Präambel erklären: Sowohl Sender- als auch Empfängerknoten befinden sich in Abhängigkeit von dem Wert von t_{LPL} gleich häufig in der Wachphase. Der Senderknoten muss jedoch zusätzlich noch die Übertragung der Nachrichteneinheit durchführen. Wird der Wert für t_{LPL} erhöht, steigt die durchschnittliche Länge der Präambel. Dies bedeutet für den Senderknoten, die Funkschnittstelle befindet sich länger im Zustand *Transmit* (Senden) und kann erst zu einem späteren Zeitpunkt in den Schlafmodus versetzt werden. Beispielsweise beträgt die durchschnittliche Präambellänge bei $t_{LPL} = 4000ms$ etwa 2 Sekunden. Dies bedeutet, dass der Empfängerknoten im Schnitt die Funkschnittstelle 2 Sekunden länger angeschaltet hat als der Empfängerknoten. Dies führt zu dem deutlich erhöhten Energiebedarf beim Senderknoten.

Neben dem höheren durchschnittlichen Energiebedarf des Senderknotens ist zudem eine große Schwankungsbreite der minimal und maximal gemessenen Energiebedarfswerte beim Sender zu erkennen. Dieser Effekt lässt sich auf Timing-Effekte aufgrund der Präambel zurückführen. Der Nachrichtenversand der PING-Anwendung erfolgt zu einem zufälligen Zeitpunkt innerhalb der Messphase. Dies führt zu unterschiedlichen Aufwachzeiten des Empfängerknotens während der Präambelübertragung und somit zu unterschiedlich langen Präambeln. Die Dauer der Präambel kann zwischen 0 und t_{LPL} ms betragen und somit zu stark unterschiedlichen Energiebedarfswerten auf Seiten des Senders führen. Der Energiebedarf des Empfängerknotens ist hiervon nicht betroffen, daher lässt sich kaum ein Unterschied zwischen minimal und maximal gemessenem Energiebedarf feststellen.

LPL sleep time t_{LPL}	Latenz in *ms*, Nachrichteneinheit 1 Byte	Latenz in *ms*, Nachrichteneinheit 100 Byte
50	28,20	34,00
100	58,73	65,19
250	124,52	130,085
500	234,81	240,92
750	369,85	375,54
1000	492,26	498,90
1250	677,92	684,041
1500	794,64	799,92
2000	1034,02	1039,53
2500	1242,73	1249,31
3000	1492,10	1497,98
3500	1643,08	1648,94
4000	1906,56	1912,50

Tabelle 4.4 Durchschnittliche Latenz der PING-Anwendung unter Verwendung von TinyOS-LPL in AVRORA+.

Zusammenfassend kann festgestellt werden, dass beim Einsatz von TinyOS-LPL die Knotenrolle einen erheblichen Einfluss auf den Energiebedarf des Sensorknotens aufweist.

Evaluierung der Latenz

Tabelle 4.4 zeigt die durchschnittlich gemessene Latenz der PING-Anwendung unter Verwendung von TinyOS-LPL in Abhängigkeit der Größe der Nachrichteneinheit. Die Latenz der Nachrichtenübertragung setzt sich dabei zusammen aus der Latenz des Duty-Cyclings von TinyOS-LPL und der Latenz von IEEE 802.15.4.

Die geringste Latenz ließ sich bei $t_{LPL} = 50ms$ mit $28,20ms$ bzw. $34,00ms$ messen. Eine Erhöhung von t_{LPL} führte jeweils zu einer Erhöhung der gemessenen Latenz. Die größte Latenz von $1906,56ms$ bzw. $1912,50ms$ wurde bei einer $t_{LPL} = 4000ms$ gemessen.

Aus den Messungen wird deutlich, dass ein Zusammenhang zwischen Latenz der Nachrichtenübertragung und dem Wert für t_{LPL} besteht. Dieser Zusammenhang ist auf die durchschnittliche Präambellänge zurückzuführen.

Der Senderknoten beginnt zu einem zufälligen Zeitpunkt die Nachrichten- und somit auch die Präambelübertragung. Der Empfängerknoten wacht periodisch alle $t_{LPL}ms$ auf, hört auf das Medium und empfängt somit die Nachricht. Aufgrund des zufälligen Sendebeginns ergibt sich eine durchschnittliche Präambellänge von $\frac{t_{LPL}}{2}ms$. Dies bedeutet, die Nachrichtenübertragung dauert im Schnitt ebenfalls etwa $\frac{t_{LPL}}{2}ms$. Dies lässt sich auch in den Messergebnissen erkennen. Die durchschnittliche Latenz liegt für alle Werte von t_{LPL} bei etwa $\frac{t_{LPL}}{2}ms$.

Die Latenz bei der Übertragung von 100 Byte Nachrichten lag im Schnitt um etwa $6ms$ über der Latenz bei einer Nachrichtengröße von 1 Byte. Dies entspricht dem Unterschied, der auch bei der Latenz von IEEE 802.15.4 gemessen wurde. Im Vergleich zu IEEE 802.15.4 ist weiterhin zu erkennen, dass die Latenz für alle gemessenen Werte von t_{LPL} größer ist. Somit entsprechen die gemessenen Latenzen den theoretisch erwarteten bzw. berechenbaren Latenzen.

Insgesamt kann festgestellt werden, dass das Duty-Cycling in TinyOS-LPL einen großen Einfluss auf die Latenz der Nachrichtenübertragung hat. Beim Einsatz von TinyOS-LPL ist somit ein Trade-off zwischen Minimierung des Energiebedarfs und der dadurch einhergehenden Erhöhung der Latenz einzugehen.

Evaluierung des Energiebedarfs unterschiedlicher Nachrichtengrößen

Tabelle 4.5 zeigt das Verhältnis des durchschnittlichen Energiebedarfs einer Nachrichtengröße von 100 Byte und einer Nachrichtengröße von 1 Byte in Abhängigkeit von t_{LPL}. Es ist zu erkennen, dass sich kaum ein messbarer Unterschied beim Versand unterschiedlicher Nachrichtengrößen messen lässt. Insgesamt ist für alle Werte von t_{LPL} und beide Nachrichtengrößen eine Abweichung des Energiebedarfs von unter 1% zu erkennen. Im Durchschnitt ergibt sich ein Verhältnis von $1,00043$. Wie auch bei IEEE 802.15.4 ergibt sich rechnerisch ein geringerer Energiebedarf beim Versand größerer Nachrichteneinheiten aufgrund der Tatsache, dass sich die Funkschnittstelle länger im Zustand *Transmit* befindet. Allerdings ist auch bei TinyOS-LPL der Einfluss auf den Gesamtenergiebedarf im Messzeitraum so gering, dass er sich nicht deutlich im Verhältnis des Energiebedarfs beider Nachrichtengrößen widerspiegelt.

LPL sleep time t_{LPL}	Verhältnis Energiebedarf $\frac{100Byte}{1Byte}$
50	1,00050
100	1,00367
250	1,00187
500	1,00407
750	1,00862
1000	1,00387
1250	1,00279
1500	0,99410
2000	0,99556
2500	0,99990
3000	1,00037
3500	0,99727
4000	0,99298
Durchschnitt	1,00043

Tabelle 4.5 Vergleich des Energiebedarfs bei Verschiedenen Nachrichtengrößen in SANDbed mit TinyOS-LPL.

Ein signifikanter Unterschied im Energiebedarf der PING-Anwendung für unterschiedliche Nachrichtengrößen kann mit den hier durchgeführten Experimenten somit auch für TinyOS-LPL nicht festgestellt werden.

4.4.3 S-MAC

Im Rahmen dieses Abschnitts wird das Medienzugriffsverfahren S-MAC untersucht.

Evaluierung des Energiebedarfs

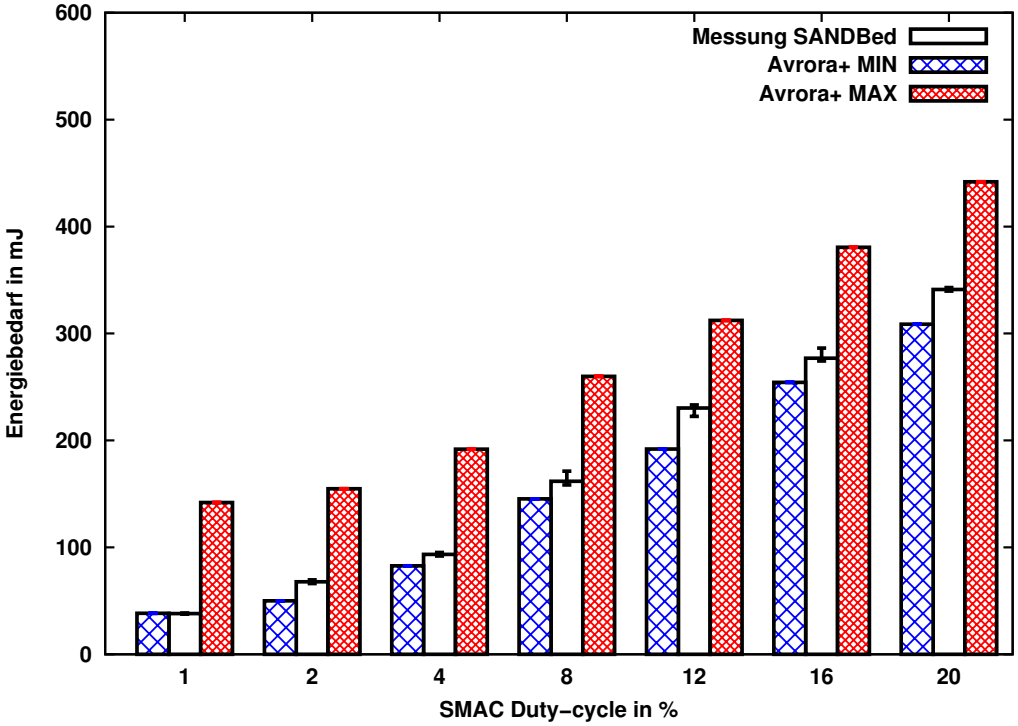
Die Abbildungen 4.12a und 4.12b zeigen den durchschnittlichen Energiebedarf der PING-Anwendung bei Verwendung von S-MAC und unterschiedlichen Nachrichtengrößen. Aufgetragen sind der Energiebedarf von Messungen in SANDbed und AVRORA+ MIN bzw. MAX für unterschiedliche Duty-Cycles. Deutlich zu erkennen ist der direkte Zusammenhang zwischen Duty-Cycle und Energiebedarf. Ein Vergrößern des Duty-Cycles verursacht einen höheren Energiebedarf der PING-Anwendung. Somit kann die Erwartungshaltung, dass ein kleinerer Duty-Cycle zu einem verringerten Energiebedarf führt in diesem Experiment bestätigt werden. Ein Duty-Cycle von 1% hat sich in den hier durchgeführten Experimenten als beste Parametrisierung von S-MAC hinsichtlich des Energiebedarfs der PING-Anwendung herausgestellt. Für eine Nachrichtengröße von 1 Byte konnte bei einem Duty-Cycle von 1% ein durchschnittlicher Energiebedarf von $77,17mJ$ in SANDbed gemessen werden. Für eine Nachrichtengröße von 100 Byte ergibt sich ein Energiebedarf von $78,46mJ$. Bei einem Duty-Cycle von 20% erhöht sich der Energiebedarf auf $660,14mJ$ bzw. $655,15mJ$.

Weiterhin ist zu erkennen, dass die minimal und maximal in SANDbed gemessenen Energiebedarfswerte kaum vom durchschnittlichen Energiebedarf abweichen. Da die Schlaf- und Wachphasen in S-MAC eine feste Dauer haben und synchronisiert stattfinden, ergeben sich innerhalb der Messung kaum Schwankungen im Energiebedarf.

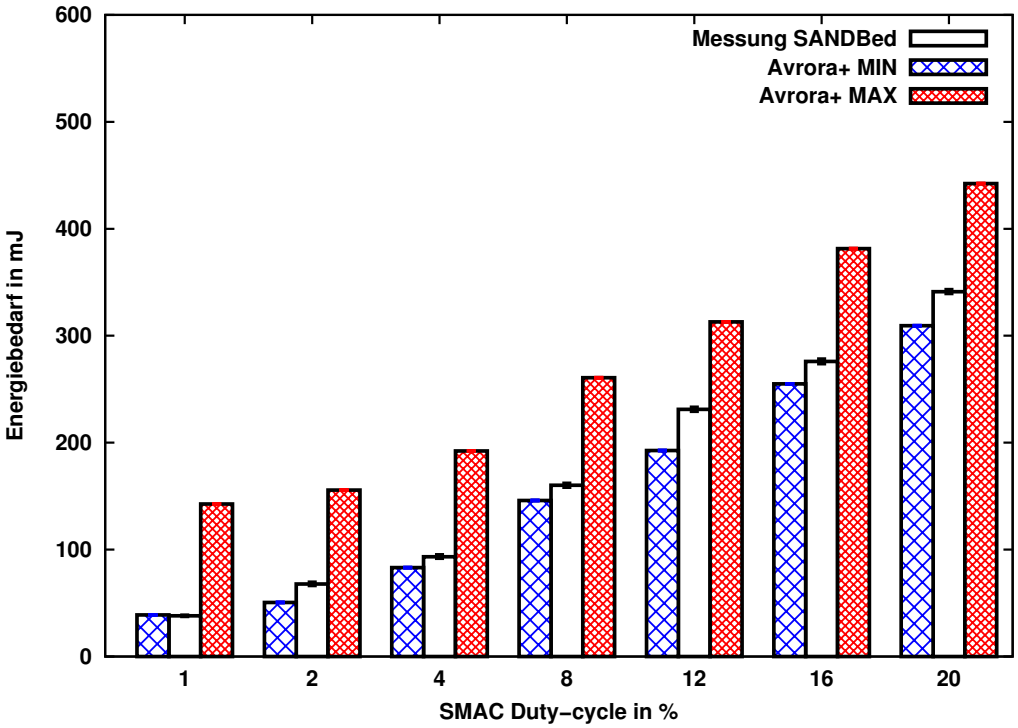
Wie auch bei den Energiebedarfsmessungen von TinyOS-LPL, lässt sich feststellen, dass der durchschnittliche in SANDbed gemessene Energiebedarf wie erwartet zwischen den Energiebedarfswerten von AVRORA+ MIN und MAX liegt. Auch hier gilt, dass der gemessene Energiebedarf aufgrund der Messung an nur zwei Sensorknoten nicht genau der Mittelwert von AVRORA+ MIN und MAX ist.

Evaluierung des Energiebedarfs unterschiedlicher Knotenrollen

Da S-MAC als synchrones Medienzugriffsverfahren für beide Sensorknoten den gleichen Zeitplan aushandelt und nutzt, ist kein unterschiedlicher Energiebedarf für Sender und Empfängerknoten zu erwarten. Abbildung 4.13 zeigt

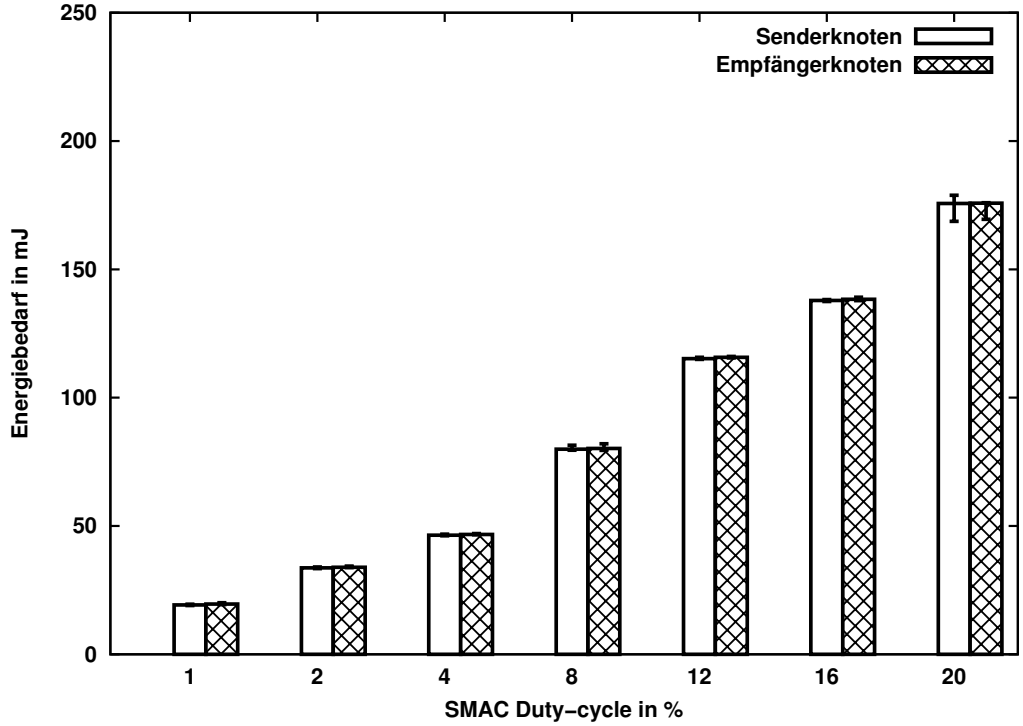


(a) Größe der Nachrichteneinheit 1 Byte

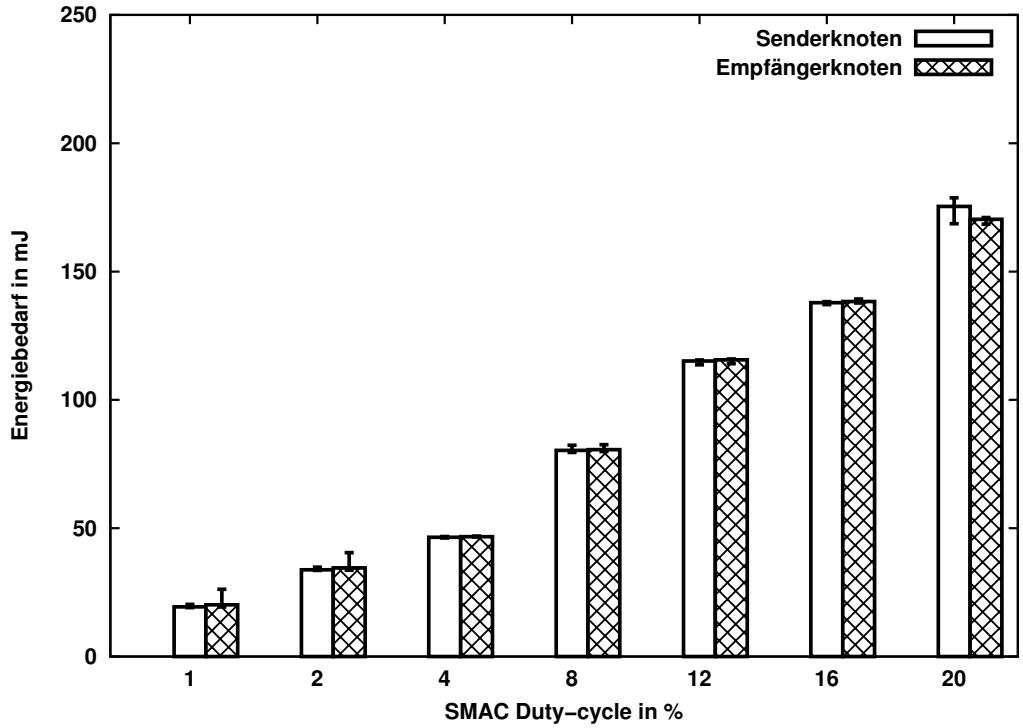


(b) Größe der Nachrichteneinheit 100 Byte

Abbildung 4.12 Kumulierter Energiebedarf der PING-Anwendung mit S-MAC in SANDbed und AVRORA+.



(a) Größe der Nachrichteneinheit 1 Byte



(b) Größe der Nachrichteneinheit 100 Byte

Abbildung 4.13 Kumulierter Energiebedarf der PING-Anwendung mit S-MAC in SANDbed getrennt nach Sender und Empfänger.

Duty-Cycle t_{DC} in %	Latenz in *ms*, Nachrichteneinheit 1 Byte	Latenz in *ms*, Nachrichteneinheit 100 Byte
1	6,93	6,93
2	3,01	3,02
4	1,97	1,98
8	1,07	1,08
12	0,99	1,00
16	0,51	0,52
20	0,29	0,29

Tabelle 4.6 Durchschnittliche Latenz der PING-Anwendung unter Verwendung von S-MAC in AVRORA+.

den Energiebedarf bei der Verwendung von S-MAC aufgeteilt nach Sender und Empfänger. Für beide Größen der Nachrichteneinheit und unabhängig vom eingestellten Duty-Cycle ist erkennbar, dass Sender und Empfängerknoten nahezu den gleichen Energiebedarf aufweisen. Dies ist durch die gleichzeitige, synchronisierte Wachphase im Einklang mit den Erwartungen.

Evaluierung der Latenz

Tabelle 4.6 zeigt die durchschnittliche Latenz der PING-Anwendung für verschiedene Duty-Cycle Einstellungen bei S-MAC. Je kleiner der Duty-Cycle in S-MAC, desto länger die Schlafphase der Sensorknoten zwischen zwei Wachzeiten. Da Nachrichten unabhängig vom eigentlichen Sendewunsch der Anwendung nur in den Wachzeiten der Sensorknoten versendet werden, bedeutet ein kleinerer Duty-Cycle automatisch eine höhere Latenz. Die größte Latenz von $6,93ms$ konnte in den hier durchgeführten Experimenten für einen Duty-Cycle von 1% gemessen werden. Eine Erhöhung des Duty-Cycles führte immer zu einer Verringerung der Latenz. Auch bei S-MAC ist somit ein Trade-off zwischen Minimierung des Energiebedarfs und der dadurch einhergehenden Erhöhung der Latenz einzugehen.

Evaluierung des Energiebedarfs unterschiedlicher Nachrichtengrößen

Tabelle 4.7 zeigt das Verhältnis des Energiebedarfs beim Versand unterschiedlicher Nachrichtengrößen bei der Verwendung von S-MAC. Auch bei S-MAC konnte ein signifikanter Einfluss der Nachrichtengröße auf den Energiebedarf der Anwendung nicht festgestellt werden. Im Durchschnitt ist das Versenden

Duty-Cycle t_{DC}	Verhältnis Energiebedarf $\frac{100Byte}{1Byte}$
1	1,01843
2	1,01057
4	1,00021
8	1,00481
12	0,99926
16	0,99996
20	0,98394
Durchschnitt	1,00246

Tabelle 4.7 Vergleich des Energiebedarfs bei Verschiedenen Nachrichtengrößen in SANDbed mit S-MAC.

von einer 100 Byte Nachrichteneinheit 0,246% teurer als der Versand einer 1 Byte Nachrichteneinheit. Insgesamt ist für alle Duty-Cycles eine Abweichung des Energiebedarfs für unterschiedliche Nachrichtengrößen von unter 2% zu erkennen. Ein signifikanter Unterschied im Energiebedarf für unterschiedliche Nachrichtengrößen kann mit den hier durchgeführten Experimenten nicht festgestellt werden.

4.4.4 TDMA-MAC

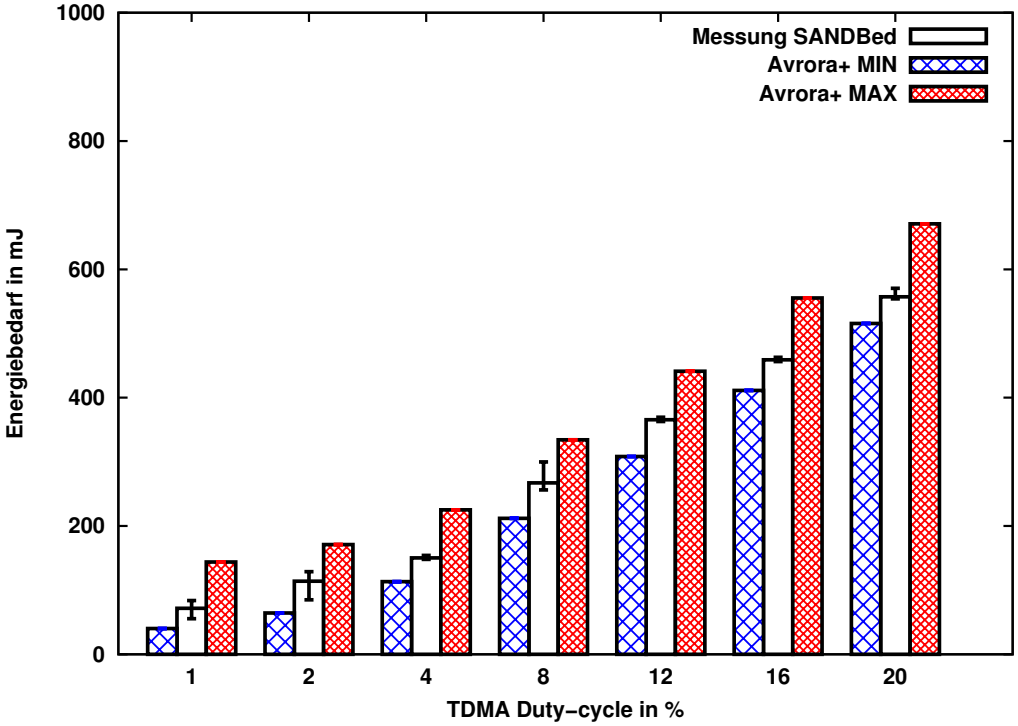
Im Rahmen dieses Abschnitts wird das Medienzugriffsverfahren TDMA-MAC untersucht.

Evaluierung des Energiebedarfs

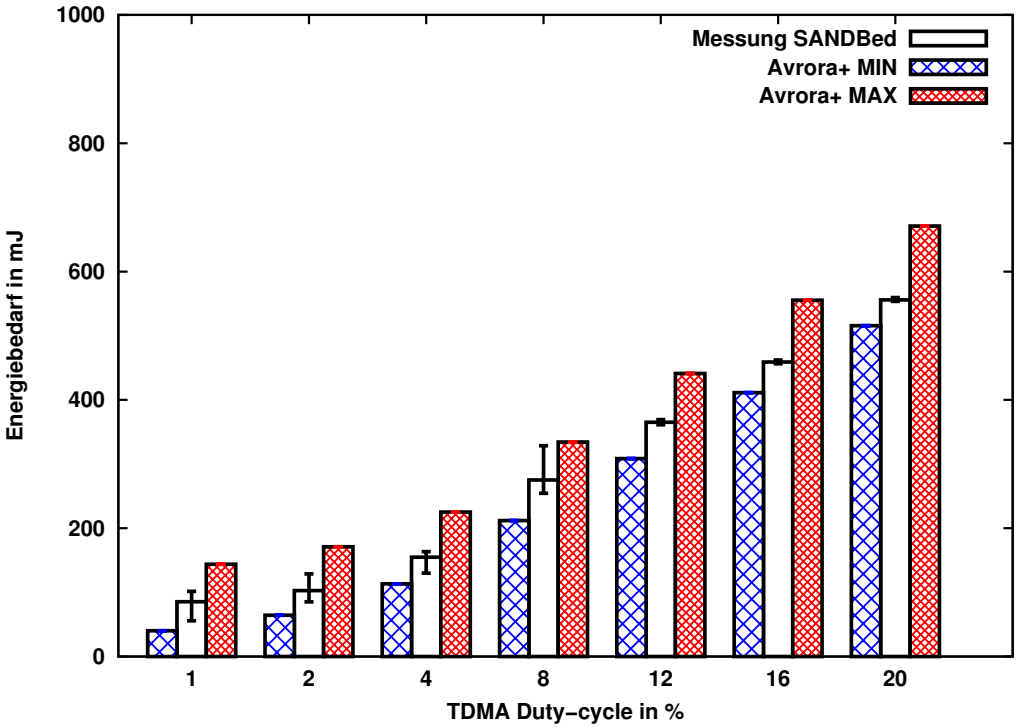
Abbildung 4.14a und 4.14b zeigen den Energiebedarf der PING-Anwendung unter Verwendung von TDMA sowohl in AVRORA+ als auch in SANDbed für Nachrichtengrößen von 1 Byte bzw. 100 Byte.

Deutlich zu erkennen ist der Anstieg des Energiebedarfs bei einer Vergrößerung des Duty-Cycles, wie es von TDMA-MAC als synchrones Medienzugriffsverfahren erwartet wurde. Der geringste Energiebedarf konnte somit für beide Nachrichtengrößen für einen Duty-Cycle von 1% gemessenen werden. Der maximale Energiebedarf ist bei einem Duty-Cycle von 20% zu sehen.

Im Vergleich zu S-MAC kann festgestellt werden, dass für den gleichen eingestellten Duty-Cycle TDMA-MAC einen etwas höheren Energiebedarf auf-

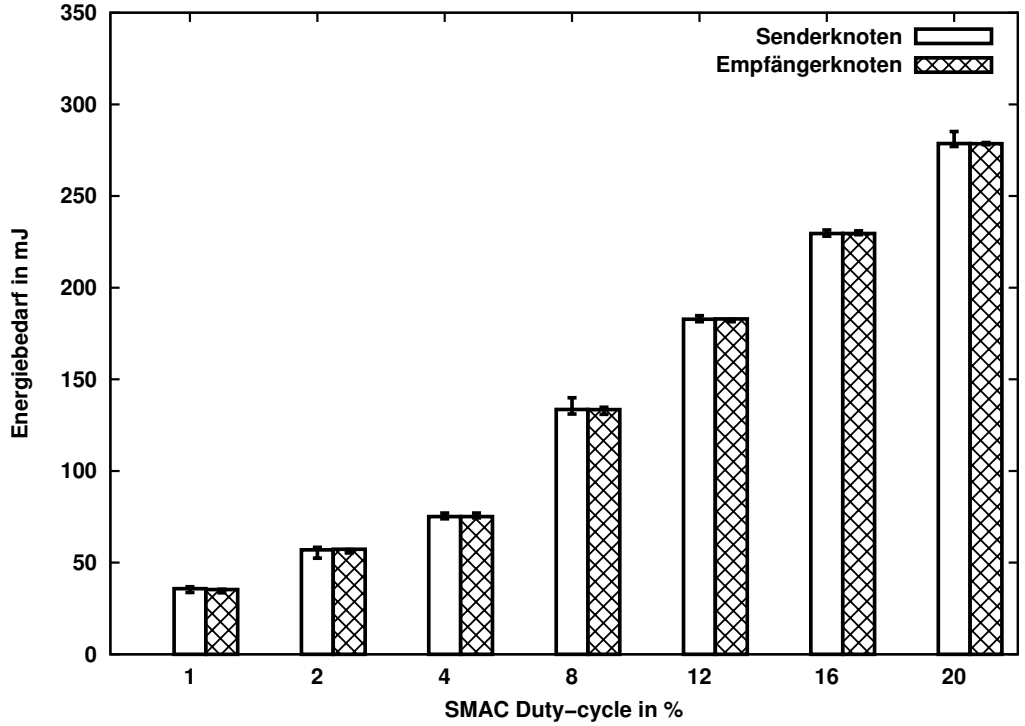


(a) Größe der Nachrichteneinheit 1 Byte

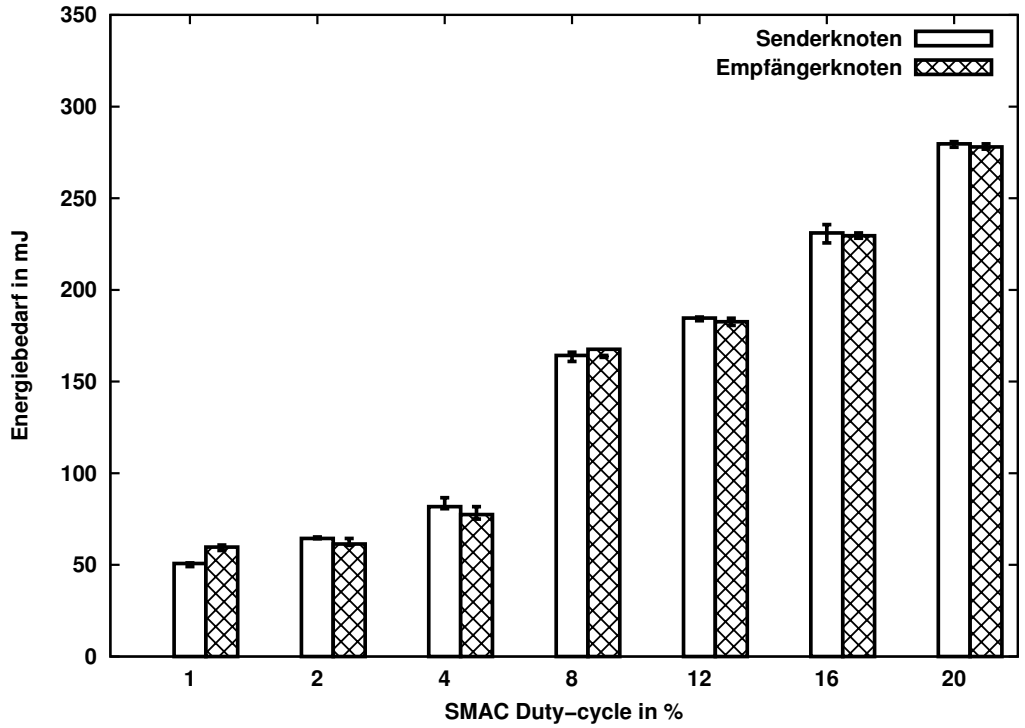


(b) Größe der Nachrichteneinheit 100 Byte

Abbildung 4.14 Kumulierter Energiebedarf der PING-Anwendung mit TDMA in SANDbed und AVRORA.



(a) Größe der Nachrichteneinheit 1 Byte



(b) Größe der Nachrichteneinheit 100 Byte

Abbildung 4.15 Energiebedarf der PING-Anwendung mit TDMA-MAC in SANDbed getrennt nach Sender und Empfänger.

Duty-Cycle t_{DC} in %	Latenz in *ms*, Nachrichteneinheit 1 Byte	Latenz in *ms*, Nachrichteneinheit 100 Byte
1	384,01	385,74
2	240,10	241,01
4	118,44	119,56
8	48,45	49,83
12	35,22	35,63
16	21,20	22,19
20	18,90	19,81

Tabelle 4.8 Durchschnittliche Latenz der PING-Anwendung unter Verwendung von TDMA-MAC in AVRORA+.

weist. Dies ist damit zu erklären, dass die hier eingestellte Wachphase in TDMA-MAC mit 50*ms* etwas länger als die Wachphase von S-MAC mit 40*ms* ist. Somit weist TDMA-MAC mit diesen Einstellungen für alle Duty-Cycle Werte einen höheren Energiebedarf auf.

Evaluierung des Energiebedarfs unterschiedlicher Knotenrollen

Wie bei S-MAC, weist auch TDMA-MAC als synchrones Medienzugriffsverfahren für Sender- und Empfängerknoten einen nahezu identischen Energiebedarf auf. Abbildung 4.15 zeigt den Energiebedarf bei der Verwendung von TDMA-MAC aufgeteilt nach Sender und Empfänger.

Evaluierung der Latenz

Tabelle 4.8 zeigt die durchschnittliche Latenz der PING-Anwendung bei Verwendung von TDMA-MAC. Wie bei S-MAC schlägt sich ein geringerer Duty-Cycle direkt in einer erhöhten Latenz nieder. Die geringste Latenz konnte bei einem Duty-Cycle von 20% mit 18*ms* bzw. 19*ms* gemessen werden. Die höchste Latenz von 385*ms* wurde bei einem Duty-Cycle von 1% und 385*ms* gemessen. Der Versand von 100 Byte Nachrichten wurde im Schnitt immer mit einer geringfügig höheren Latenz gemessen.

Evaluierung des Energiebedarfs unterschiedlicher Nachrichtengrößen

Tabelle 4.9 zeigt das Verhältnis des Energiebedarfs beim Versand unterschiedlicher Nachrichtengrößen in SANDbed. Wie bei den anderen untersuchten

Duty-Cycle t_{DC}	Verhältnis Energiebedarf $\frac{100Byte}{1Byte}$
1	0,99952
2	1,0000
4	1,0001
8	1,0070
12	0,9711
16	1,0431
20	1,0464
Durchschnitt	1,0039

Tabelle 4.9 Vergleich des Energiebedarfs bei Verschiedenen Nachrichtengrößen in SANDbed mit TDMA-MAC

Medienzugriffsverfahren konnte kein signifikanter Unterschied im Energiebedarf beim Versand unterschiedlicher Nachrichtengrößen festgestellt werden. Im Durchschnitt ist das Versenden von 100 Byte $0,39\%$ teurer als der Versand von 1 Byte. Insgesamt ist für alle Duty-Cycles eine Abweichung des Energiebedarfs von unter 4% zu erkennen.

4.5 Diskussion der Ergebnisse und Zusammenfassung

Im Rahmen dieses Abschnitts sollen die Ergebnisse der Evaluierung sowohl mit der zu Beginn des Kapitels formulierten Erwartungshaltung verglichen, als auch eine Bewertung der Ergebnisse durchgeführt werden.

4.5.1 Vergleich mit der Erwartungshaltung

Die Evaluierungsergebnisse zeigen, dass die Erwartungshaltung im Bezug zu dem Energiebedarf der PING-Anwendung nicht immer erfüllt wurden.

- **Je größer der Wert für t_{LPL}, desto kleiner der Energiebedarf:**

  Bei TinyOS-LPL führt ein größerer Wert für t_{LPL} nicht zwangsläufig zu einem geringeren Energiebedarf. Für die PING-Anwendung konnte festgestellt werden, dass der geringste Energiebedarf mit $t_{LPL} = 1250ms$ erreicht wurde. Ein kleinerer oder größerer Wert für t_{LPL} führte zu einem erhöhten Energiebedarf. Es wird vermutet, dass je Kommunikationsverhalten für jede Anwendung ein anderer optimaler Wert für t_{LPL} existiert.

Beispielsweise existiert für die untersuchten Schlüsselaustauschverfahren in Kapitel 5 jeweils ein anderer optimaler Wert für t_{LPL}. Eine ähnliche Erkenntnis wurde ebenfalls in der Dissertation *Energie-effiziente Concast-Kommunikation in drahtlosen Sensornetzen* [60] am Beispiel Concast Kommunikation gewonnen. Um eine allgemeingültige Aussage formulieren zu können, müsste diese Vermutung allerdings anhand weiterer Anwendungsbeispiele untersucht werden.

- **Je kleiner der Wert für t_{DC}, desto kleiner der Energiebedarf:**

  Bei der Verwendung von S-MAC und TDMA-MAC führte ein kleinerer Wert für t_{DC} zu einem geringeren Energiebedarf der PING-Anwendung, eine Erhöhung von t_{DC} immer zu einer Erhöhung des Energiebedarfs. Dies liegt daran, dass die Dauer der Schlaf- und Wachphase der Funkschnittstelle direkt von der Parametrisierung von t_{DC} abhängt und keine zusätzliche Synchronisation der Wachphase, beispielsweise durch eine Präambel, notwendig ist. Ein kleinerer Wert von t_{DC} führt somit automatisch zu einer kürzeren Wachphase der Sensorknoten und somit zu einem geringeren Energiebedarf.

- **Je größer die Nachrichteneinheit, desto größer der Energiebedarf:**

  Die Größe der Nachrichteneineinheit hat bei keinem der untersuchten Medienzugriffsverfahren einen signifikanten Einfluss auf den Energiebedarf. Im Durchschnitt über alle untersuchten Medienzugriffsverfahren führte der Versand des 100 Byte Datenpakets zu einer Erhöhung des Energiebedarf von unter 2%. Für alle Medienzugriffsverfahren sollte rechnerisch eine größere Nachrichteneinheit zu einem verringerten Energiebedarf führen. Allerdings ist die Verringerung des Energiebedarfs so gering, dass dieser anhand der PING-Anwendung kaum messbar war.

- **Die Nachrichtenübertragung ist maßgeblich für den Energiebedarf verantwortlich:**

  Die Ergebnisse der Evaluierung zeigen, dass die Nachrichtenübertragung nicht für alle Medienzugriffsverfahren maßgeblich für den Energiebedarf der PING-Anwendung ist. Vielmehr ist die genutzte Duty-Cycling Strategie so wie die daraus resultierenden Wachphasen der Funkschnittstelle entscheidend. Bei IEEE 802.15.4, S-MAC und TDMA-MAC ist kein Mehrbedarf durch den Versand einer Nachricht festzustellen, lediglich bei TinyOS-LPL kann aufgrund der Nutzung Präambel ein Mehrbedarf an Energie durch den Versand von Nachrichten festgestellt werden.

- **Die Rolle der Sensorknoten hat kaum Einfluss auf den Energiebedarf:**

  Die Rolle des Sensorknotens ist lediglich bei TinyOS-LPL entscheidend für den Energiebedarf. Aufgrund der Präambel kommt es dort beim Sendeknoten zu einer deutlichen Steigerung des Energiebedarfs im Vergleich zum empfangenden Sensorknoten. Bei den anderen untersuchten Medienzugriffsverfahren hat die Rolle des Sensorknotens kaum Einfluss auf den Energiebedarf.

- **Einfluss der Duty-Cycling Strategie auf die Latenz**

  Neben dem Energiebedarf der PING-Anwendung wurde auch die Latenz der Nachrichtenübertragung untersucht. Bei allen Medienzugriffsverfahren konnte kein signifikanter Unterschied der Nachrichtenlatenz bei verschiedenen Nachrichtengrößen festgestellt werden. Weiterhin konnte bei allen Medienzugriffsverfahren ein Zusammenhang zwischen Duty-Cycle Parametrisierung und Latenz gezeigt werden. Bei S-MAC und TDMA-MAC führt ein höherer Energiebedarf gleichzeitig auch zu einer geringeren Nachrichtenlatenz. Eine Minimierung des Energiebedarfs bedeutet zwangsläufig eine Erhöhung der Nachrichtenlatenz. Bei TinyOS-LPL konnte dieser Zusammenhang nicht beobachtet werden. Ein höherer Wert für t_{LPL} erhöht gleichzeitig die Nachrichtenlatenz, verringert den Energiebedarf aber nicht zwangsläufig.

4.5.2 Bewertung der Ergebnisse

Die hier gewonnen Ergebnisse sind spezifisch für die eingesetzte Hardwareplattform und die Implementierung der Medienzugriffsverfahren in TinyOS. Trotzdem lassen sich einige der Ergebnisse allgemeiner formulieren:

Eine Kernaussage der Evaluierung ist der starke Einfluss der Duty-Cycling Strategie auf den Energiebedarf der Anwendung. Dies lässt sich auf der im Vergleich zu anderen Hardwarekomponenten hohen Leistungsaufnahme der Funkschnittstelle erklären. Diese Eigenschaft lässt sich auf fast alle Funktransceiver und Sensorknotenplattformen übertragen. Somit ist davon auszugehen, dass sich die hier gewonnenen qualitativen Zusammenhänge zwischen Duty-Cycling Strategie und Energiebedarf auch auf andere Plattformen übertragen lassen. Dies müsste allerdings anhand von beispielhaften Messungen für jede Sensorknotenplattform überprüft werden.

Weiterhin sind die Erkenntnisse zum Einfluss der Nachrichtengröße, der Energiebedarf in Abhängigkeit von der Knotenrolle und der Latenz unabhängig von der konkreten Implementierung der Medienzugriffsverfahren. Es lassen sich vielmehr Aussagen über synchrone und asynchrone Duty-Cycling

Strategien treffen. Asynchrone Duty-Cycling Strategien benötigen einen Synchronisationsmechanismus der Wachzeiten wie beispielsweise einer Präambel. Hierdurch kann es zu unterschiedlichen Wachphasen und somit Energiebedarfswerten für Sender und Empfänger kommen. Auch die Latenz des Nachrichtenversands hängt direkt vom eingesetzten Synchronisierungsmechanismus ab. Bei synchronen Duty-Cycling Strategien sind die Wachphasen schon vor dem Nachrichtenversand synchronisiert, der Nachrichtenversand hat kaum Einfluss auf den Energiebedarf[11].

Die hier gewonnenen Ergebnisse wurden in einem einfachen Szenario mit nur zwei Sensorknoten gewonnen. Die Aussagen lassen sich zum Teil auch auf größere Netze und Multi-Hop Umgebungen übertragen, insbesondere was den signifikanten Einfluss der Medienzugriffsverfahren auf die Latenz angeht. Allerdings müssen in Multi-Hop Szenarien noch viele weitere Parameter betrachtet werden, beispielsweise der Einfluss einer Empfangs- oder Sendewarteschlange auf die Latenz, Nachrichtenkollisionen oder die zeitliche Synchronisation der unterschiedlichen Sender- und Empfängerknoten. Es kann weiterhin davon ausgegangen werden, dass der Einfluss des Medienzugriffsverfahren weiterhin einen signifikanten Einfluss auf den Energiebedarf der untersuchten Anwendung hat. Beispielsweise konnte dies bei der Untersuchung des Energiebedarfs von Concast-Kommunikation unter Verwendung der Medienzugriffsverfahren der EEMAC-Programmierschnittstelle nachgewiesen werden [64].

Ein bisher kaum untersuchter Aspekt der Duty-Cycling Strategie ist der Zusammenhang zwischen Latenz und Energiebedarf. Im Rahmen dieses Kapitels konnte für alle Medienzugriffsverfahren ein Zusammenhang zwischen Duty-Cycle Parametrisierung und Latenz gezeigt werden. Dies wurde bisher bei der Evaluierung des Energiebedarfs von Anwendungen in Sensornetzen selten betrachtet. Insbesondere ist hierbei zu bedenken, dass die Latenz einen erheblichen Einfluss auf Protokolle hat, die verteilte Berechnungen durchführen und hierfür kommunizieren. Beispielsweise wird im folgenden Kapitel anhand des Schlüsselaustauschs gezeigt, dass die Latenz einen maßgeblichen Einfluss auf die Dauer und somit auf den Energiebedarf eines Schlüsselaustauschs hat.

Der Vergleich der Ergebnisse zwischen AVRORA+ und SANDbed zeigt deutlich, dass AVRORA+ wie erwartet sowohl qualitativ gleichwertige (kein Einfluss der Größe einer Nachrichteneinheit oder der Knotenrolle) als auch quantitativ vergleichbare (Ergebnisse in SANDbed liegen zwischen AVRORA+ MIN und MAX) Ergebnisse liefert. Mit AVRORA+ steht somit ein Simulati-

[11] Dies gilt für die im Rahmen dieses Kapitels betrachteten synchronen Duty-Cycling Strategien. Für andere synchrone Duty-Cycling Strategien können die Ergebnisse unterschiedlich ausfallen

onswerkzeug bereit, mit dem es möglich ist, vergleichbare Evaluierungen des Energiebedarfs von MICAz-Sensorknoten durchzuführen, ohne explizit zeitaufwendige Experimente in SANDbed durchführen zu müssen. AVRORA+ ist daher eine sinnvolle Ergänzung zu SANDbed.

4.6 Zusammenfassung

Ziel dieses Kapitels war es, den Energiebedarf und die Latenz unterschiedlicher Medienzugriffsverfahren in drahtlosen Sensornetzen zu evaluieren. Hierfür wurden die PING-Anwendung und Experimente in AVRORA+ und SANDbed genutzt. Es wurde untersucht, welchen Einfluss die Nachrichtengröße, die Knotenrolle, die Parametrisierung des Medienzugriffsverfahren und die eingesetzte Duty-Cycling Strategie auf den Energiebedarf und die Latenz der Anwendung haben. Zusammengefasst gilt, dass die Nachrichtengröße in den hier durchgeführten Experimenten und für alle hier untersuchten Medienzugriffsverfahren keinen Einfluss auf den Energiebedarf der PING-Anwendung hatte. Die Knotenrolle führte lediglich bei TinyOS-LPL zu einem unterschiedlichen Energiebedarf von Sender- und Empfängerknoten. Der Senderknoten hat mit TinyOS-LPL aufgrund der notwendig Präambel einen erhöhten Energiebedarf. Weiterhin konnte gezeigt werden, dass die Parametrisierung des Medienzugriffsverfahren den Energiebedarf und die Latenz der PING-Anwendung maßgeblich beeinflusst.

5. Evaluierung des Energiebedarfs von Schlüsselaustauschverfahren

Zur kryptografischen Absicherung der Kommunikation im FleGSens-Prototypen war es notwendig, ein Schlüsselaustauschprotokoll zu implementieren. Eine der Annahmen bei der Auswahl des Schlüsselaustauschprotokolls war es, dass asymmetrische Kryptografie aufgrund der im Vergleich zu symmetrischer Kryptografie langen Berechnungsdauer und des Kommunikationsoverheads nicht einsetzbar ist. Eine Evaluierung des Energiebedarfs des Schlüsselaustauschprotokolls wurde innerhalb von FleGSens nicht durchgeführt. Aktuelle Arbeiten und Protokollvorschläge im Kontext des Internets der Dinge legen nahe, dass asymmetrische Verfahren basierend auf elliptischen Kurven einsetzbar sein könnten. Beispielsweise werden für den Einsatz in 6LoWPAN Netzen sowohl Verfahren wie Kerberos oder ein Diffie-Hellman Schlüsselaustausch vorgeschlagen [36]. Allerdings wird die Frage aufgeworfen, in wie fern die Ressourcenbeschränktheit der Sensorknoten den Einsatz erlauben.

Wie im vorigen Kapitel deutlich wurde, beeinflusst die Wahl und Parametrisierung des Medienzugriffsverfahrens den Energiebedarf und die Latenz der Kommunikation innerhalb einer Anwendung maßgeblich. Ziel dieses Kapitels ist daher die Evaluierung des Energiebedarfs von Schlüsselaustauschverfahren in drahtlosen Sensornetzen unter Verwendung unterschiedlicher Medienzugriffsverfahren. Das Kapitel gliedert sich wie folgt: Im ersten Teil des Kapitels werden die Anforderungen an die Evaluierung des Energiebedarfs von Schlüsselaustauschverfahren vorgestellt. Im zweiten Teil des Kapitels werden die Einflussfaktoren auf den Energiebedarf von Schlüsselaus-

tauschverfahren ECDH-ECDSA und Kerberos unter Verwendung der EEMAC-Programmierschnittstelle untersucht.

Die Ergebnisse der Evaluierung des Energiebedarfs der Schlüsselaustauschverfahren sind im Rahmen zweier Diplomarbeiten [63], [65] und zweier Veröffentlichung [9], [10] entstanden, welche gemeinsam mit Joachim Wilke veröffentlicht wurden. Die Ergebnisse der Veröffentlichungen wurden für diese Arbeit erweitert.

5.1 Einflussfaktoren auf den Energiebedarf von Schlüsselaustauschverfahren

Im Rahmen dieses Kapitels wird der Energiebedarf zweier Schlüsselaustauschverfahren analysiert. Bereits im vorigen Kapitel wurde der Einfluss des Medienzugriffsverfahrens auf den Energiebedarf einer Anwendung aufgezeigt. Bei der sehr einfachen PING-Anwendung konnte gezeigt werden, dass der Energiebedarf und die Latenz der Anwendung maßgeblich von der Wahl und der Parametrisierung des Medienzugriffsverfahrens abhängig sind. Im Vergleich zu der relativ simplen PING-Anwendung enthalten Schlüsselaustauschverfahren typischerweise aufwändige kryptografische Berechnungen, deren Energiebedarf in der aktuellen Forschung häufig als alleiniges Auswahlkriterium für die Wahl eines Schlüsselaustauschverfahrens herangezogen wird. Insbesondere asymmetrische Schlüsselaustauschverfahren, wie das Diffie-Hellman Verfahren, werden aufgrund der Ressourcenbeschränktheit der Sensorknoten häufig als nicht geeignet für drahtlose Sensornetze angesehen [36], [34], [66], [67]. Eine Evaluierung des Energiebedarfs basierend auf einem möglichst realitätsnahen Energiemodell der Sensorknoten im Zusammenspiel mit Medienzugriffsverfahren und Duty-Cycling ist jedoch selten vorzufinden. Im Rahmen dieses Kapitels soll daher der Energiebedarf eines symmetrischen Schlüsselaustauschverfahrens (Kerberos) mit dem Energiebedarf eines asymmetrischen Verfahrens (ECDH-ECDSA) verglichen werden. Um Seiteneffekte durch die Wahl und die Parametrisierung des Medienzugriffsverfahrens zu evaluieren, werden die in der EEMAC-Programmierschnittstelle verfügbaren Medienzugriffsverfahren eingesetzt. Im Rahmen der Evaluierung wird gezeigt, dass die Wahl und Parametrisierung des Medienzugriffsverfahrens einen größeren Einfluss auf den Energiebedarf der Gesamtanwendung hat als die Wahl des Schlüsselaustauschverfahren. Weiterhin wird deutlich, dass der Energiebedarf für die Kommunikation einen größeren Anteil am Gesamtenergiebedarf hat als allgemein angenommen. Der Energiebedarf für die Kommunikation übersteigt dabei je nach Medi-

enzugriffsverfahren den Energiebedarf der kryptografischen Berechnungen deutlich.

Als Evaluationsmetrik für den Vergleich verschiedener Schlüsselaustauschverfahren kommt häufig der Direktvergleich des Energiebedarfs eines Schlüsselaustauschs zum Einsatz. Beim Vergleich verschiedener Verfahren ist dementsprechend häufig das Ergebnis der Evaluation, dass ein Schlüsselaustauschverfahren um einen bestimmten Faktor energie-effizienter als die anderen Verfahren ist. Ein solcher Vergleich wird im Rahmen dieser Evaluation mit ECDH-ECDSA und Kerberos und den unterschiedlichen Medienzugriffsverfahren durchgeführt.

Bei einer realen Anwendung, wie beispielsweise im Projekt FlegSens [41], [40] zur Überwachung einer grünen Grenze, ist der Schlüsselaustausch lediglich ein kleiner Teil der Gesamtanwendung. Dies bedeutet, dass insbesondere die Wahl des Medienzugriffsverfahrens und dessen Parametrisierung so durchgeführt wird, dass die Gesamtanwendung eine möglichst lange Lebensdauer bzw. einen möglichst geringen Energiebedarf aufweist. Weiterhin existieren innerhalb der Anwendungsausführung aus administrativen Gründen häufig Zeitspannen, in denen der Schlüsselaustausch auf allen Knoten gleichzeitig angestoßen und durchgeführt wird. In der FleGSens-Anwendung [41], [40] wurde beispielsweise der initiale Schlüsselaustausch innerhalb eines fest definierten Zeitfensters (Initialisierungsphase) durchgeführt. Hierbei ist es daher notwendig, den Energiebedarf sowohl des Schlüsselaustauschs als auch der möglichen Leerlaufzeit innerhalb der Initialisierungsphase zu betrachten. Der Energiebedarf wird in diesem Fall während einer festen Zeitspanne bestimmt, wobei innerhalb dieser Zeitspanne der Schlüsselaustausch durchgeführt wird. Im Rahmen der Evaluierung werden zwei unterschiedliche Evaluationsmetriken (Energiebedarf pro Schlüsselaustausch und Energiebedarf pro Zeitspanne) verwendet und verglichen. Es kann gezeigt werden, dass insbesondere der Energiebedarf durch das Medienzugriffsverfahren und des Duty-Cyclings einen erheblich Einfluss auf den Vergleich des Energiebedarfs bei unterschiedlichen Evaluationsmetriken hat. So weist der Schlüsselaustausch mit ECDH-ECDSA zwar im Schnitt immer noch einen höheren Energiebedarf auf, beim Vergleich der festen Zeitspanne für den Schlüsselaustausch ist der Unterschied jedoch deutlich geringer.

5.2 Beschreibung der Schlüsselaustauschverfahren

Im Rahmen dieses Kapitels wird der Energiebedarf von zwei Schlüsselaustauschverfahren evaluiert und verglichen. Mit beiden Schlüsselaustauschverfahren kann ein authentifizierter Schlüsselaustausch durchgeführt werden. Beide Verfahren beinhalten Mechanismen zur Verhinderung von Wiederein-

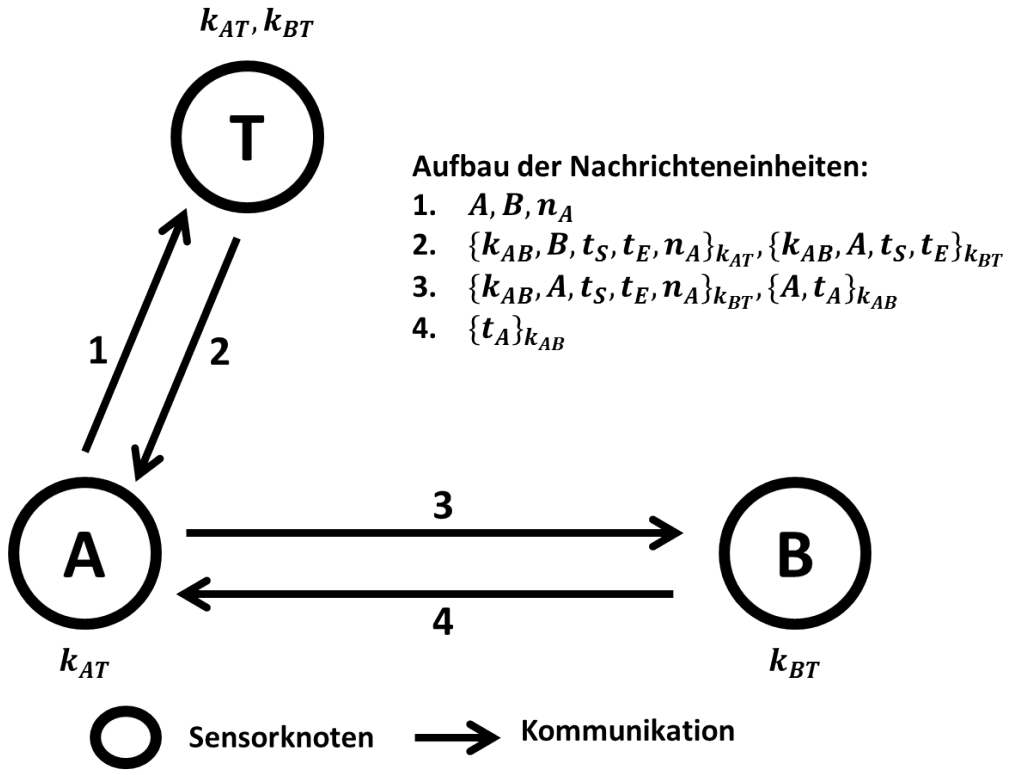


Abbildung 5.1 Ablauf eines Schlüsselaustauschs mit Kerberos.

spielungsangriffen oder Denial-of-Service Angriffen. Da in dieser Arbeit kein Vergleich der Schlüsselaustauschprotokolle im Bezug auf ihrer Resistenz gegenüber Angriffen durchgeführt wird, wird nicht weiter auf diese Angriffe eingegangen.

Im Folgenden sollen die verwendeten Implementierungen und Protokollvarianten analysiert werden. Die Implementierung der Schlüsselaustauschprotokolle wurde in TinyOS, Version 2.1.2, durchgeführt. Als Sensorknoten wurde der MicaZ Sensorknoten verwendet.

Als Schlüsselaustauschverfahren wurden eine Version des Kerberos Verfahrens und der Schlüsselaustausch nach dem Diffie-Hellman Verfahren basierend auf elliptischen Kurven genutzt. Kerberos wird hierbei als Vertreter einer Klasse von Schlüsselaustauschverfahren basierend auf symmetrischer Kryptografie, ECDH-ECDSA als Vertreter der asymmetrischen Verfahren genutzt, wie in [36] vorgeschlagen.

Kerberos

Der Ablauf eines Schlüsselaustauschs zweier Sensorknoten A und B mit Hilfe von Kerberos ist in Abbildung 5.1 dargestellt. Tabelle 5.1 fasst die verwendeten Abkürzungen und Parameter des Kerberos Schlüsselaustauschs zusammen. Eine Voraussetzung für den Schlüsselaustausch mit Kerberos ist die

Parameter	Beschreibung
A, B	Identifikatoren der Sensorknoten A und B
n_A	Zufallszahl, gewählt von Sensorknoten A
k_{AB}	Geheimer, symmetrischer Schlüssel zwischen Sensorknoten A und B
t_S, t_E, t_A	Zeitstempel zur Verhinderung von Wiedereinspielungsangriffen
Ticket Granting Ticket (TGT)	$\{k_{AB}, B, t_S, t_E, n_A\}_{k_{AT}}$
Ticket	$\{k_{AB}, A, t_S, t_E\}_{k_{BT}}$
Authenticator	$\{t_A\}_{k_{AB}}$

Tabelle 5.1 Abkürzungen und Parameter des Schlüsselaustauschverfahrens Kerberos.

Existenz und Verfügbarkeit einer sogenannten Vertrauensinstanz. Die Vertrauensinstanz muss dabei während der Durchführung des Schlüsselaustauschs für die beteiligten Sensorknoten jederzeit erreichbar und verfügbar sein. Sinnvollerweise sollte die Vertrauensinstanz über eine dauerhaften Stromversorgung verfügen und somit beispielsweise zusammen mit der Basisstation des Sensornetzes implementiert sein. In der folgenden Evaluierung wird jeweils angegeben, welche Annahmen über die Platzierung und der Stromversorgung der Vertrauensinstanz getroffen wurden.

Bevor ein Schlüsselaustausch mit Kerberos durchgeführt werden kann ist es notwendig, paarweise symmetrische Schlüssel zwischen jedem Sensorknoten (A und B) und der Vertrauensinstanz (T) einzurichten. In Abbildung 5.1 sind dementsprechend die symmetrischen Schlüssel k_{AT} und k_{BT} jeweils auf Sensorknoten A und T bzw. Sensorknoten B und T vor der eigentlichen Aushandlung des Schlüssels zwischen Sensorknoten A und B vorverteilt. Eine solche Vorverteilung wird üblicherweise Offline, beispielsweise bei der Ausbringung der Anwendung bzw. des Sensornetzes oder bei der Produktion der Sensorknoten durchgeführt.

Der Schlüsselaustausch der Sensorknoten A und B mit Kerberos benötigt insgesamt vier Nachrichten. Um einen Schlüssel mit Sensorknoten B auszutauschen, sendet Sensorknoten A in der ersten Nachrichten den eigenen Identifikator (A), den Identifikator des Sensorknotens B (B) und eine Zufallszahl (n_A) an die Vertrauensinstanz T.

Die Antwort der Vertrauensinstanz (Nachricht 2) ist in zwei Nachrichtenteile getrennt. Im ersten Teil der Nachricht sendet die Vertrauensinstanz ein sogenanntes *Ticket Granting Ticket*, *TGT*. Dieses ist mit dem geheimen Schlüssel k_{AT} verschlüsselt und somit nur für Sensorknoten A lesbar. Im TGT sind neben dem eigentlichen Schlüssel zwischen Sensorknoten A und B (k_{AB}) auch der Identifikator des Sensorknotens B, zwei Zeitstempel (t_S und t_E) und die Zufallszahl n_A enthalten. Die Zeitstempel t_S und t_E definieren den Start bzw. Endzeitpunkt ab bzw. bis zu dem der Schlüssel k_{AB} verwendet werden kann. Nachdem Sensorknoten A das TGT entschlüsselt hat, ist er somit insbesondere im Besitz des Schlüssels mit Sensorknoten B und kann diesen zur weiteren Verwendung abspeichern.

Der zweite Teil von Nachricht 2 wird bei Kerberos auch *Ticket* genannt. Das Ticket enthält ebenfalls die beiden Zeitstempel t_S und t_E, den Knotenidentifikator A und den Schlüssel k_{AB}. Da das Ticket jedoch mit dem Schlüssel k_{BT} verschlüsselt ist, kann Sensorknoten A den Inhalt nicht lesen. Das Ticket wird im nächsten Protokollschritt benötigt, um den Schlüssel k_{AB} an Sensorknoten B zu verteilen. Hierfür wird das Ticket in Nachricht 3 zusammen mit dem sogenannten *Authenticator* übertragen.

Sensorknoten B kann nun den Schlüssel k_{AB} aus dem Ticket extrahieren, die Gültigkeit überprüfen und den Schlüssel zur weiteren Verwendung abspeichern. Der Authenticator dient zur Authentifikation des Sensorknotens A gegenüber Sensorknoten B, beinhaltet einen Zeitstempel t_A, den Knotenidentifikator A und ist mit dem Schlüssel k_{AB} verschlüsselt. Sensorknoten B kann die Gültigkeit überprüfen, da er im Besitz von k_{AB} ist. Im letzten Protokollschritt (Nachricht 4) übermittelt Sensorknoten B den verschlüsselten Zeitstempel t_A an Sensorknoten B und kann somit den erfolgreichen Schlüsselaustausch gegenüber Sensorknoten A beweisen.

Für die symmetrische Verschlüsselung wird innerhalb der hier verwendeten Implementierung von Kerberos AES im CBC-Modus eingesetzt [30]. Die verwendete Implementierung von AES ist im TinyOS-contrib Repository [68] zu finden. Insgesamt werden bei einem Schlüsselaustausch mit Kerberos 196 Byte Nutzdaten in insgesamt 4 Nachrichten übertragen. Hierbei wird analog zu [38] von 64 Bit Zeitstempeln, 32 Bit für jede Zufallszahl und 8 Bit Knotenidentifikatoren ausgegangen. Dies führt aufgrund der Nutzung von AES mit einer Blocklänge von 128 Bit und insgesamt 11 zu verschlüsselnden Blöcken zu insgesamt 196 Byte Nutzdaten.

ECDH-ECDSA

Der Ablauf eines Schlüsselaustauschs zweier Sensorknoten A und B mit Hilfe von ECDH-ECDSA ist in Abbildung 5.2 dargestellt. Tabelle 5.2 fasst die ver-

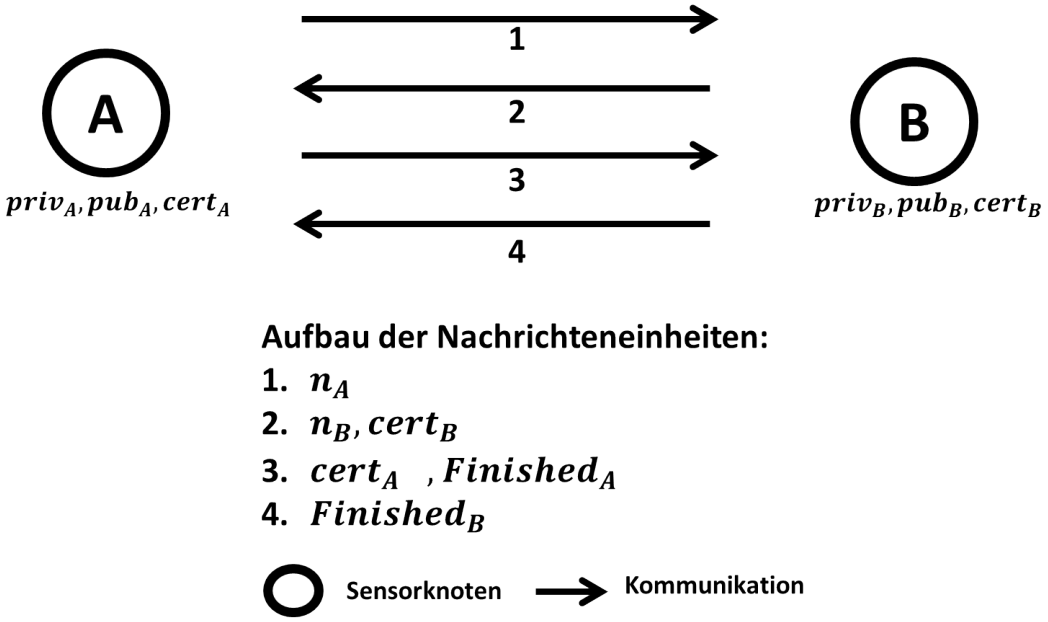


Abbildung 5.2 Ablauf eines Schlüsselaustauschs mit ECDH-ECDSA.

Parameter	Beschreibung
$priv_A$	Privater Schlüssel des Sensorknotens A
pub_A	Öffentlicher Schlüssel des Sensorknotens A
$cert_A$	Identitätszertifikat des Sensorknotens A. Beinhaltet signierten öffentlichen Schlüssel pub_A
$Finished_A$	Finished Nachricht zur Erkennung von Replay Angriffen. Verschlüsselter Hash der ausgetauschten Zufallszahlen n_A und n_B
n_A	Zufallszahl, gewählt von Sensorknoten A

Tabelle 5.2 Abkürzungen und Parameter des Schlüsselaustauschverfahrens ECDH-ECDSA.

wendeten Abkürzungen und Parameter des Schlüsselaustausches mit ECDH-ECDSA zusammen.

Das Schlüsselaustauschverfahren ECDH-ECDSA basiert auf asymmetrischer Kryptografie. Dementsprechend ist es notwendig, dass jeder Sensorknoten sowohl über einen privaten Schlüssel (*priv*) als auch einen dazugehörigen öffentlichen Schlüssel (*pub*) verfügt. Die Authentizität des öffentlichen Schlüssels und damit des Sensorknotens wird zudem über ein Identitätszertifikat (*cert*) verifiziert. Hierfür besitzt jeder Sensorknoten den öffentlichen Schlüssel der Zertifikatsinstanz, die die Zertifikate ausgestellt hat. Anzumerken ist hierbei, dass die Einrichtung und Verwaltung einer Public-Key Infrastruktur nicht Teil dieser Arbeit sind, insbesondere wird im Folgenden nicht auf den Widerruf oder die Verteilung der Zertifikate eingegangen.

Vor der Verwendung von ECDH-ECDSA muss jeder beteiligte Sensorknoten über einen privaten und einen öffentlichen Schlüssel als auch über das zugehörige Zertifikat verfügen. Eine solche Vorverteilung kann wie bei Kerberos Offline, beispielsweise bei der Ausbringung der Anwendung bzw. des Sensornetzes oder bei der Produktion der Sensorknoten durchgeführt werden.

Insgesamt werden für den Schlüsselaustausch mit ECDH-ECDSA 4 Nachrichten benötigt. Die erste Nachricht enthält lediglich die Zufallszahl n_A. Sensorknoten B antwortet mit einer eigenen Zufallszahl n_B und dem Zertifikat $cert_B$ in Nachricht 2. Sensorknoten A überprüft nun die in $cert_B$ enthaltene Signatur mit Hilfe des öffentlichen Schlüssels der Zertifikatsinstanz. Nach der erfolgreichen Überprüfung der in $cert_A$ enthaltenen Signatur, besitzt Sensorknoten A alle notwendigen Eingabewerte um einen gemeinsamen Schlüssel zu berechnen. Eingabewerte sind neben den beiden Zufallszahlen n_A und n_B auch die beiden öffentlichen Schlüssel der Sensorknoten A und B. Nach der Berechnung des gemeinsamen Schlüssels sendet Sensorknoten A $cert_A$ und $Finished_A$ an Sensorknoten B. Sensorknoten B kann nach dem Empfang von Nachricht 3 den gemeinsamen Schlüssel k_{AB} berechnen. Zum Abschluss des Schlüsselaustauschs sendet Sensorknoten B $Finsished_B$ in Nachricht 4.

$Finished_A$ und $Finished_B$ enthalten dabei jeweils einen mit AES verschlüsselten Hash der ausgetauschten Zufallszahlen n_A und n_B und der jeweiligen Knotenidentifikatoren. Für Sensorknoten A beispielsweise

$$\{Hash(n_A, n_B, A)\}_{k_{AB}}$$

Für die symmetrische Verschlüsselung wird innerhalb der hier verwendeten Implementierung von Kerberos AES im CBC-Modus eingesetzt [30]. Die verwendete Implementierung von AES ist im TinyOS-contrib Repository [68] zu finden. Die asymmetrischen kryptografischen Operationen und der Diffie-

Hellman Schlüsselaustausch basieren auf der TinyECC-Bibliothek für asymmetrische Kryptografie mit elliptischen Kurven [69]. Insgesamt werden für einen Schlüsselaustausch 4 Nachrichten und 276 Byte Nutzdaten übertragen. Hierbei kommen Zertifikate mit einer Größe von 86 Byte, 32 Byte Zufallszahlen und *Finished* Nachrichten der Länge 20 Byte zum Einsatz. Zusätzlich zur Kommunikation muss jeder Sensorknoten eine Überprüfung einer Signatur, eine Schlüsselberechnung sowie eine Verschlüsselung und Entschlüsselung der *Finished* Nachricht durchführen.

5.3 Beschreibung der Evaluationsmetriken und des Versuchsaufbaus

In diesem Abschnitt werden die Evaluationsmetriken und der Versuchsaufbau zur Evaluierung des Energiebedarfs der Schlüsselaustauschprotokolle vorgestellt.

5.3.1 Evaluationsmetriken

Bei der Evaluierung des Energiebedarfs der Schlüsselaustauschprotokolle werden verschiedene Aspekte und Einflussfaktoren auf den Energiebedarf untersucht. Zusammengefasst sind dies im Einzelnen:

- **Energiebedarf und Dauer der kryptografischen Operationen:** ECDH-ECDSA und Kerberos nutzen jeweils unterschiedliche kryptografische Verfahren. ECDH-ECDSA nutzt sowohl asymmetrische als auch symmetrische Kryptografie, Kerberos setzt lediglich symmetrische Kryptografie ein. Im ersten Teil der Evaluation wird untersucht, welchen Energiebedarf und welche Berechnungsdauer die einzelnen kryptografischen Algorithmen aufweisen. Die Messungen des Energiebedarfs umfassen lediglich die Kosten für die Berechnung auf dem Mikrocontroller.

- **Energiebedarf eines Schlüsselaustauschs:** Der Energiebedarf genau eines Schlüsselaustausches unter Verwendung unterschiedlicher Medienzugriffsverfahren wird analysiert. Der gemessene Energiebedarf umfasst somit sowohl die Berechnungskosten der kryptografischen Algorithmen, als auch den Energiebedarf der Kommunikation. Die Energiemessung startet, sobald der den Schlüsselaustausch initiierende Knoten den Schlüsselaustausch beginnt. Die Messung stoppt, sobald der Schlüsselaustausch bei allen beteiligten Sensorknoten beendet ist. Dies bedeutet bei Kerberos, dass sowohl der Energiebedarf der beiden austauschenden Sensorknoten, als auch der Energiebedarf der Vertrauensinstanz gemessen

wird. Im Vergleich zu den Energiebedarfsmessungen der PING-Anwendung ist die Dauer der Messung je nach eingesetztem Medienzugriffsverfahren und der Parametrisierung des Duty-Cycles unterschiedlich lang. Die Messung beinhaltet somit insbesondere keine Zeiträume, in denen der Schlüsselaustausch bereits erfolgreich durchgeführt wurde. Im Vergleich hierzu wurde bei der Evaluierung der PING-Anwendung ein fester Zeitraum vermessen, unabhängig davon ob die PING-Nachricht bereits übertragen wurde oder nicht. In den entsprechenden Abbildungen wird auf der y-Achse sowohl der durchschnittliche, als auch der minimal und maximal gemessene Energiebedarf in mJ dargestellt.

- **Dauer eines Schlüsselaustauschs:** Die Dauer eines Schlüsselaustauschs wird evaluiert. Hierbei wird untersucht, welchen Einfluss die Latenz der Medienzugriffsverfahren und die Dauer der kryptografischen Berechnungen auf den Energiebedarf des Schlüsselaustauschs haben. In den entsprechenden Abbildungen wird auf der y-Achse sowohl die durchschnittliche, als auch die minimal und maximal gemessene Dauer in Sekunden dargestellt.

- **Energiebedarf über einen festen Messzeitraum:** Der Energiebedarf der Schlüsselaustauschverfahren wird hierbei über einen festen Messzeitraum gemessen[12]. Ziel hierbei ist es zu untersuchen, welcher Energiebedarf aufgrund der Duty-Cycling Strategie des Medienzugriffsverfahrens verursacht wird und wie dieser im Verhältnis zu den Berechnungs- und Kommunikationskosten steht. In den entsprechenden Abbildungen wird auf der y-Achse sowohl der durchschnittliche, als auch der minimal und maximal gemessene Energiebedarf in mJ dargestellt. Neben dem Energiebedarf für ECDH-ECDSA und Kerberos (Energiebedarf der Knoten A,B und T) ist außerdem der Energiebedarf der Sensorknoten A und B, ohne Sensorknoten T, bei der Nutzung von Kerberos dargestellt.

Der Ablauf der Experimente ist in Abbildung 5.3 zu sehen. Nach der Bootphase der Sensorknoten wird die Funkschnittstelle für IEEE 802.15.4, TinyOS-LPL und S-MAC zu einem zufälligen Zeitpunkt, jedoch maximal 5 Sekunden nach Ende der Bootphase, eingeschaltet. Diese zufällige Anschaltzeit ist zur Verhinderung von Timing-Effekten notwendig. Bei TDMA-MAC ist es erforderlich, die Funkschnittstelle gleichzeitig einzuschalten, da in TDMA-MAC kein Verfahren zur Synchronisierung der Wachzeiten integriert ist. Nach dem Ein-

[12]Der Messzeitraum wurde so groß gewählt, dass mit allen Parametrisierungen der Medienzugriffsverfahren ein erfolgreicher Schlüsselaustausch durchgeführt werden kann. So soll verhindert werden, dass aufgrund hoher Latenzen durch das Duty-Cycling ein Schlüsselaustausch nicht innerhalb des Messzeitraums durchführbar ist.

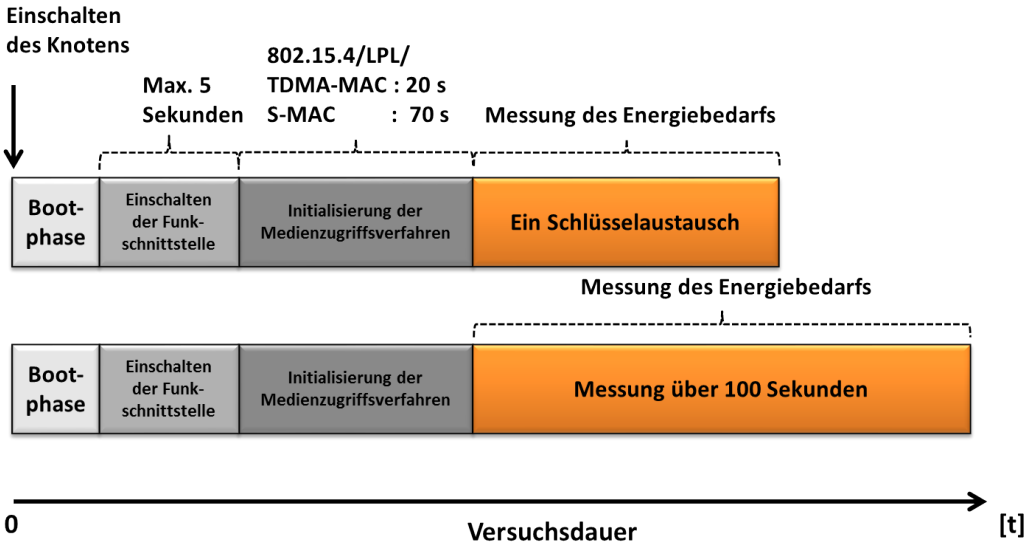


Abbildung 5.3 Ablauf der Experimente zum Schlüsselaustausch.

schalten der Funkschnittstelle wird das entsprechende Medienzugriffsverfahren initialisiert. Je nach Evaluationsmetrik wird daraufhin der Energiebedarf und die Dauer genau eines Schlüsselaustauschs oder der Energiebedarf des festen Messintervalls gemessen.

5.3.2 Simulationsparameter

Tabelle 5.3 fasst die bei der Evaluierung verwendeten Simulationsparameter zusammen. Als Sensorknotenplattform kommt der MICAz-Sensorknoten zum Einsatz. Die Schlüsselaustauschverfahren ECDH-ECDSA und Kerberos sind in TinyOS 2.1.2 implementiert, zur Evaluierung wird der Simulator AVRORA+ mit dem Energiemodell der MICAz-Sensorknoten eingesetzt. Insgesamt werden für beide Schlüsselaustauschverfahren pro Parametrisierung und Medienzugriffsverfahren 100 Wiederholungen des Experiments durchgeführt. Für die Nutzung verschiedener Medienzugriffsverfahren wird die EEMAC-Programmierschnittstelle eingesetzt. Somit sind die Medienzugriffsverfahren IEEE 802.15.4, TinyOS-LPL, S-MAC und TDMA-MAC nutzbar. Die einstellbaren Parameter der Medienzugriffsverfahren wurden im vorigen Kapitel beschrieben. Für die Evaluierung des Energiebedarfs des Schlüsselaustauschs werden unterschiedliche Aufwachintervalle für TinyOS-LPL und unterschiedliche Duty-Cycles für S-MAC und TDMA-MAC verwendet. Als Kanalmodell wird das verlustfreie Unit Disc Modell des Simulators AVRORA verwendet. Während dies nicht den Gegebenheiten in einer realen Anwendung entspricht, so kann das Kanalmodell als *best case* Beispiel dienen, um die grundsätzlichen Einflussfaktoren auf den Energiebedarf von Schlüsselaustauschverfahren in einem idealen Szenario zu untersuchen. Der Einfluss von Nach-

Parameter	Wert
Knotenplattform	MICAz
Betriebssystem	TinyOS 2.1.2
Simulator	AVRORA+
Schlüsselaustauschverfahren	ECDH-ECDSA und Kerberos
Experimentwiederholungen	100
Medienzugriffsverfahren	802.15.4 im Non-Beacon Mode, TinyOS-LPL, S-MAC und TDMA-MAC
Kanal	25
Sendeleistung	31
Sendereichweite	25 m
Kanalmodell	Unit Disc, verlustfrei
Aufwachintervall t_{LPL}	50, 100, 200, 500, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000 und 10000 ms
CCA Länge t_{CCA}	2400
S-MAC Duty-Cycle t_{DC}	1, 2, 4, 8, 12, 16, 20 %
TDMA-MAC Duty-Cycle t_{DC}	1, 2, 4, 8, 12, 16, 20 %

Tabelle 5.3 Simulationsparameter der Evaluierung des Energiebedarfs der Schlüsselaustauschverfahren.

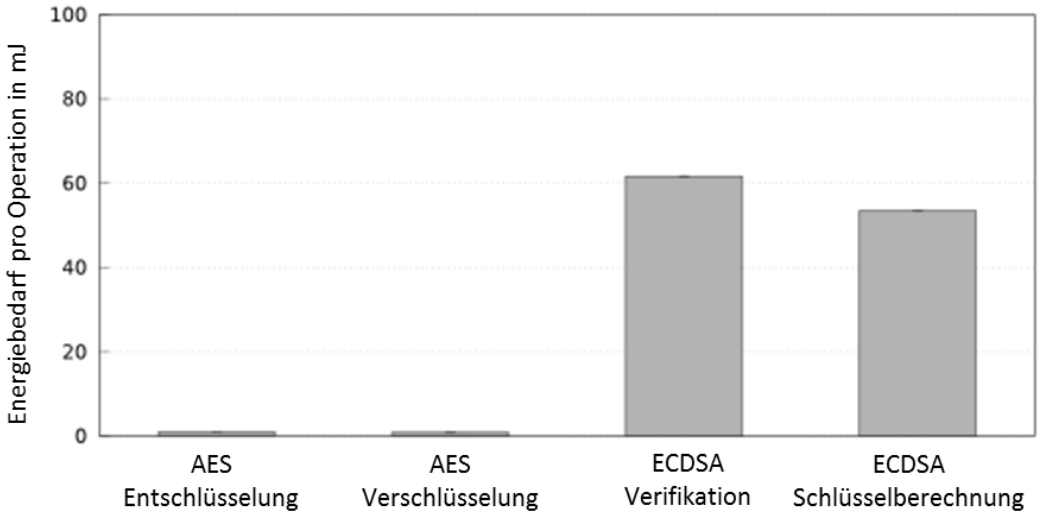


Abbildung 5.4 Energiebedarf der kryptografischen Operationen bei ECDH-ECDSA und Kerberos.

richtenverlusten auf den Energiebedarf müsste somit nochmals zusätzlich untersucht werden.

5.4 Evaluierung des Energiebedarfs

In diesem Abschnitt wird der Energiebedarf der Schlüsselaustauschverfahren Kerberos und ECDH-ECDSA evaluiert. Bei der Evaluierung werden verschiedene Medienzugriffsverfahren eingesetzt.

5.4.1 Energiebedarf der kryptografischen Operationen

Bereits bei der Beschreibung der Schlüsselaustauschprotokolle ECDH-ECDSA und Kerberos wurden die unterschiedlichen verwendeten kryptografischen Operationen angesprochen. Kerberos setzt lediglich symmetrische Verschlüsselung mit AES-CBC und einer Schlüssellänge von 128 Bit ein. Insgesamt werden bei einem Schlüsselaustausch mit Kerberos 196 Byte Nutzdaten ver- bzw entschlüsselt.

ECDH-ECDSA basiert dagegen auf asymmetrischen kryptografischen Operationen. Zum Einen wird auf den beteiligten Sensorknoten ein gemeinsamer Schlüssel mit Hilfe des Diffie-Hellman Verfahrens basierend auf elliptischen Kurven berechnet. Weiterhin wird die Authentizität der beteiligten Sensorknoten mit Hilfe von digitalen Signaturen und Zertifikaten nach dem ECDSA Standard sichergestellt. Jeder Sensorknoten muss somit die Verifikation einer Signatur, eine Schlüsselberechnung sowie die Verschlüsselung und Entschlüsselung der *Finished* Nachricht durchführen.

Protokoll	Operation	Dauer
ECDH-ECDSA	Berechnung des Schlüssels	1,96 Sekunden
	Überprüfung eines Zertifikats	2,29 Sekunden
	Berechnung / Überprüfung der Finished Nachricht jeweils	0,02 Sekunden
Kerberos	3-mal Ver- und Entschlüsselung	0,028 Sekunden

Tabelle 5.4 Dauer der kryptografischen Operationen in ECDH-ECDSA und Kerberos.

Der Energiebedarf der kryptografischen Operationen ist in Abbildung 5.4 dargestellt. Der durchschnittliche Energiebedarf in mJ ist jeweils für die symmetrischen Operationen in Kerberos und die asymmetrischen Operationen in ECDH-ECDSA auf der y-Achse dargestellt.

Deutlich zu erkennen ist der größere Energiebedarf der asymmetrischen Operationen in ECDH-ECDSA. Die Verifikation eines Zertifikats hat den größten Energiebedarf mit 62 mJ. Die Schlüsselberechnung mit ECDH weist einen Energiebedarf von 53 mJ auf. Die Ver- und Entschlüsselung mit AES in Kerberos hat einen Energiebedarf von 0,97 mJ bzw. 0,84 mJ. Basierend auf einem reinen Vergleich des Energiebedarfs der kryptografischen Operationen und somit der eingesetzten Rechenleistung der Sensorknoten sollte der Schlüsselaustausch mit Kerberos einen geringeren Energiebedarf aufweisen.

Tabelle 5.4 fasst die gemessene Dauer der einzelnen kryptografischen Operationen innerhalb von ECDH-ECDSA und Kerberos zusammen. Bei ECDH-ECDSA dauern die kryptografischen Operationen pro Knoten knapp über 4 Sekunden, wobei 1,96 Sekunden auf die Berechnung des Schlüssels, 2,29 Sekunden auf die Überprüfung eines Zertifikats und insgesamt 0,02 Sekunden auf den Vergleich und die Berechnung der Finished Nachricht entfallen. Bei Kerberos werden insgesamt drei Verschlüsselungen und drei Entschlüsselungen vorgenommen. Insgesamt dauern diese Operationen $0,028$ Sekunden.

Innerhalb von ECDH-ECDSA fallen somit insgesamt $8,54$ Sekunden an kryptografischen Berechnungen an, bei Kerberos nur $0,028$ Sekunden. Ausgehend von diesem Vergleich und den Energiebedarfswerten kann vermutet werden, dass ein Schlüsselaustausch mit ECDH-ECDSA im Vergleich zu Kerberos eine wesentlich größere Dauer und einen wesentlich höheren Energiebedarf haben müsste.

5.4.2 IEEE 802.15.4

Im Rahmen dieses Abschnitts wird der Schlüsselaustausch mittels ECDH-ECDSA und Kerberos unter Verwendung von IEEE 802.15.4 evaluiert.

Energiebedarf pro Schlüsselaustausch

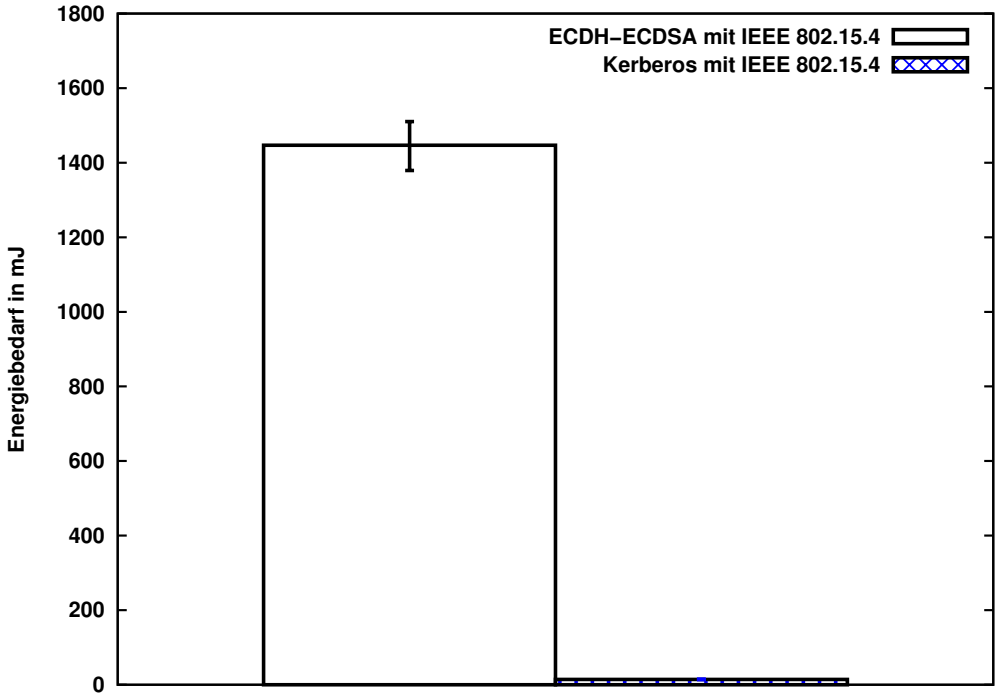
Abbildung 5.5a zeigt den Energiebedarf eines Schlüsselaustauschs mit ECDH-ECDSA und Kerberos bei Verwendung von IEEE 802.15.4. Ein Schlüsselaustausch mit ECDH-ECDSA hat einen durchschnittlichen Energiebedarf von $1446mJ$, wohingegen ein Schlüsselaustausch mit Kerberos lediglich einen durchschnittlichen Energiebedarf von $14,2mJ$ aufweist. Bei Kerberos wurde hierbei der Energiebedarf sowohl der Sensorknoten A und B als auch des Sensorknotens T über die Dauer des Schlüsselaustauschs gemessen.

Neben dem durchschnittlichen Energiebedarf ist zusätzlich der minimal und maximal in AVRORA+ gemessene Energiebedarf aufgezeichnet. Der minimale Energiebedarf für ECDH-ECDSA konnte mit $1378mJ$, der maximale Energiebedarf mit $1510mJ$ bestimmt werden. Kerberos wies einen minimalen Energiebedarf von $13,8mJ$ bzw. einen maximalen Energiebedarf von $14,7mJ$ auf. Die Unterschiede im minimalen und maximalen Energiebedarf sind hierbei auf die Verwendung von AVRORA+ und mit den im dortigen Energiemodell abgebildeten Hardwaretoleranzen begründet. In jedem Simulationsdurchlauf wird ein Sensorknoten mit unterschiedlich parametrisiertem Energiemodell verwendet.

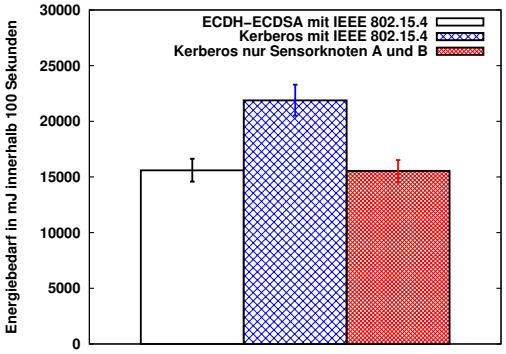
ECDH-ECDSA hat bei der Verwendung von IEEE 802.15.4 einen etwa 100-mal größeren Energiebedarf pro Schlüsselaustausch als der Schlüsselaustausch mit Kerberos. Der Grund hierfür ist die Dauer des Schlüsselaustauschs in Verbindung mit der dauerhaft angeschalteten Funkschnittstelle. Da bei IEEE 802.15.4 kein Duty-Cycling der Funkschnittstelle durchgeführt wird, ist diese während der gesamten Dauer des Schlüsselaustausch eingeschaltet.

Dauer eines Schlüsselaustauschs

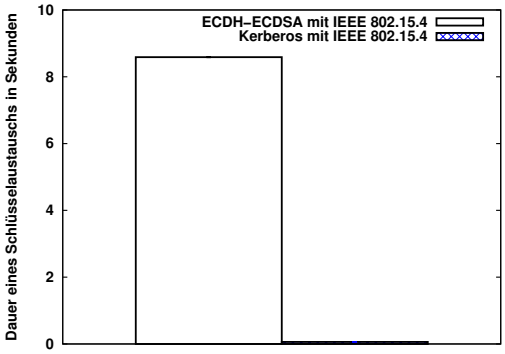
Abbildung 5.5c zeigt die Dauer eines Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von IEEE 802.15.4. Ein Schlüsselaustausch mit ECDH-ECDSA hat eine durchschnittliche Dauer von 8,6 Sekunden, wobei wie in einem vorigen Abschnitt gezeigt etwa 8,5 Sekunden auf die kryptografischen Berechnungen zurückzuführen sind. Der Schlüsselaustausch mit Kerberos hat eine Dauer von 0,063 Sekunden, hiervon entfallen etwa 0,044 Sekunden auf die kryptografischen Berechnungen. Es zeigt sich, dass die längere Dauer des Schlüsselaustauschs mit ECDH-ECDSA in einem direkten Zusammenhang zum Energiebedarf steht, da die Funkschnittstellen während der gesamten Dauer angeschaltet sind. Der Energiebedarf steigt somit mit der längeren Dauer des Schlüsselaustauschs.



(a) Energiebedarf eines Schlüsselaustauschs mit IEEE 802.15.4



(b) Fester Messzeitraum



(c) Dauer eines Schlüsselaustauschs

Abbildung 5.5 Energiebedarf und Dauer eines Schlüsselaustauschs mit Kerberos und ECDH-ECDSA unter Verwendung von IEEE 802.15.4.

Energiebedarf über einen festen Messzeitraum

Abbildung 5.5b zeigt den Energiebedarf der Schlüsselaustauschverfahren über einen Messzeitraum von 100 Sekunden. Der Vergleich von ECDH-ECDSA und Kerberos zeigt, dass Kerberos mit durchschnittlich 21877 mJ Energiebedarf einen deutlich größeren Energiebedarf aufweist auf als ECDH-ECDSA mit 15593 mJ. Dieser Unterschied im Energiebedarf ist auf den zusätzlichen Energiebedarf des Sensorknotens T zurückzuführen. Ohne Sensorknoten T haben die Sensorknoten A und B einen Energiebedarf von durchschnittlich 15544 mJ.

Aus den Messergebnissen wird deutlich, wie groß der Energiebedarf der Schlüsselaustauschverfahren im Vergleich zu den Kosten der dauerhaft eingeschalteten Funkschnittstelle in IEEE 802.15.4 ist. Da kein Duty-Cycle eingesetzt wird, ist die Funkschnittstelle dauerhaft an und hat somit eine Leistungsaufnahme von etwa 80 mW. Im Messzeitraum von 100 Sekunden fallen somit bei ECDH-ECDSA 1446 mJ auf den reinen Schlüsselaustausch (kryptografische Berechnungen und Kommunikation), die restlichen 14146 mJ Energiebedarf fallen allein durch die angeschaltete Funkschnittstelle an. Bei Kerberos entfallen $14,2mJ$ auf die Kommunikation und die kryptografischen Berechnungen, $15529mJ$ auf die im restlichen Messzeitraum angeschaltete Funkschnittstelle.

In dem hier genutzten Messintervall wird weiterhin deutlich, dass der Unterschied zwischen Kerberos und ECDH-ECDSA im Vergleich zu der Messung eines einzelnen Schlüsselaustauschs sehr gering ist. Beachtet man die zusätzlichen Kosten des Sensorknotens T, so weist Kerberos sogar einen höheren Energiebedarf auf.

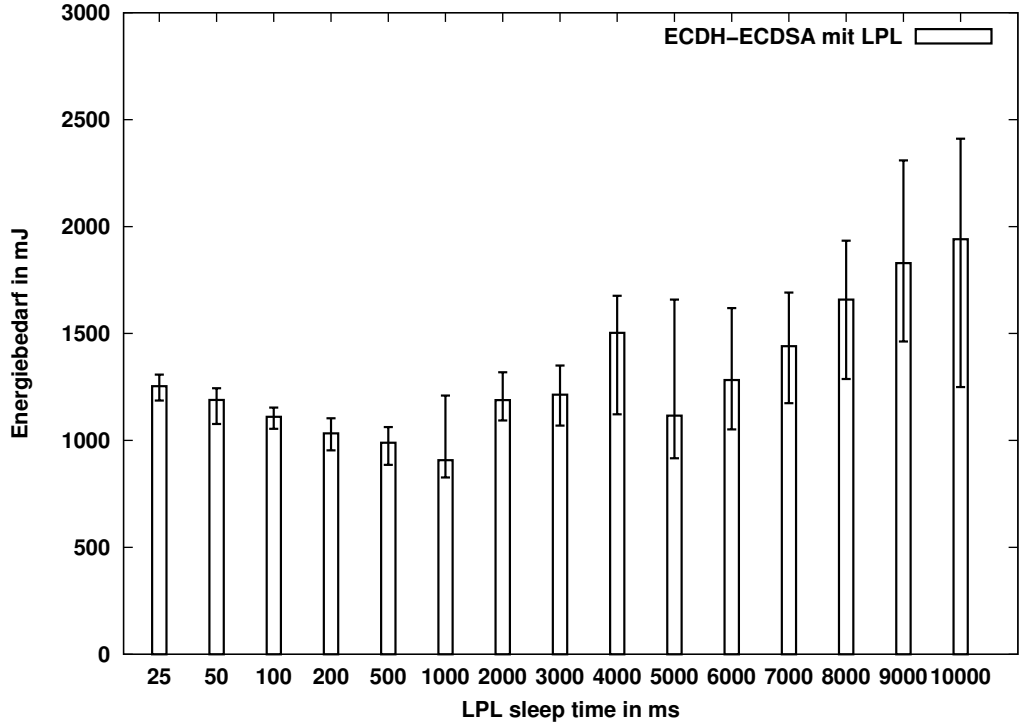
5.4.3 TinyOS-LPL

In diesem Abschnitt wird der Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von TinyOS-LPL evaluiert.

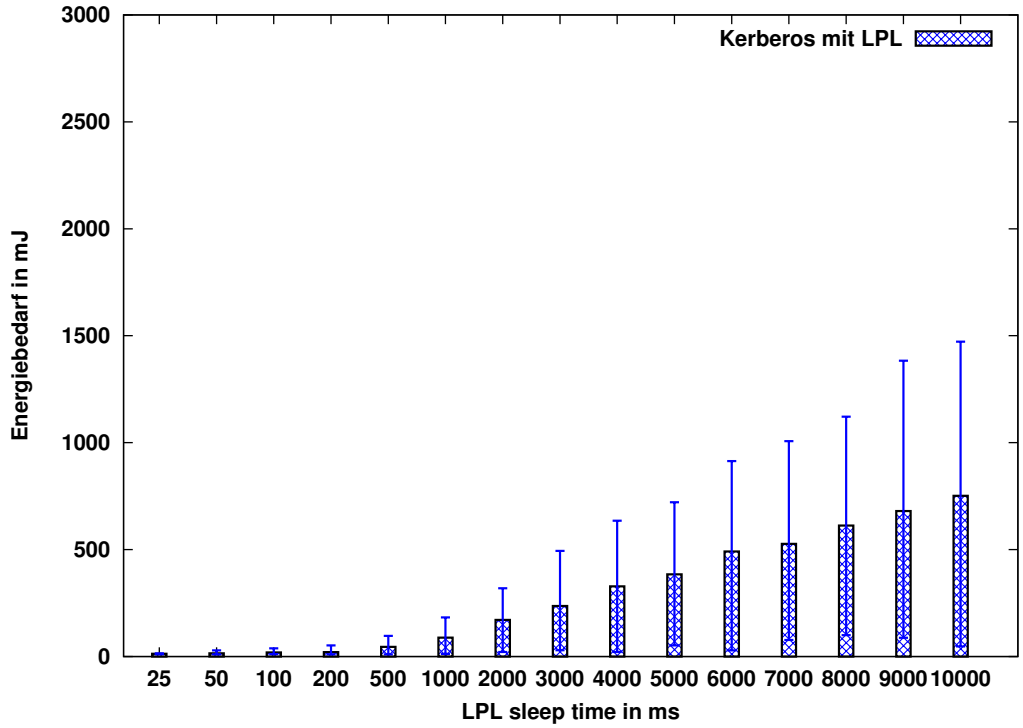
Energiebedarf eines Schlüsselaustauschs

Abbildung 5.6a zeigt den Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA, Abbildung 5.6b den Energiebedarf von Kerberos bei der Verwendung von TinyOS-LPL. Auf den x-Achsen sind die unterschiedlichen Werte für das TinyOS-LPL Aufwachintervall, t_{LPL}, zu sehen.

Der Energiebedarfs von ECDH-ECDSA ist stark abhängig von der gewählten Parametrisierung von t_{LPL}. Der geringste Energiebedarf wurde bei $t_{LPL} = 1000ms$ mit durchschnittlich $917,22mJ$ erreicht. Eine Verringerung oder Erhöhung von t_{LPL} führte zu einer Erhöhung des Energiebedarfs. Ein ähnlicher Zusammenhang zwischen Energiebedarf und t_{LPL} konnte bei der Analyse der PING-Anwendung in Kapitel 4 beobachtet werden. Ein Wert von



(a) Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA



(b) Energiebedarf des Schlüsselaustauschs mit Kerberos

Abbildung 5.6 Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von TinyOS-LPL.

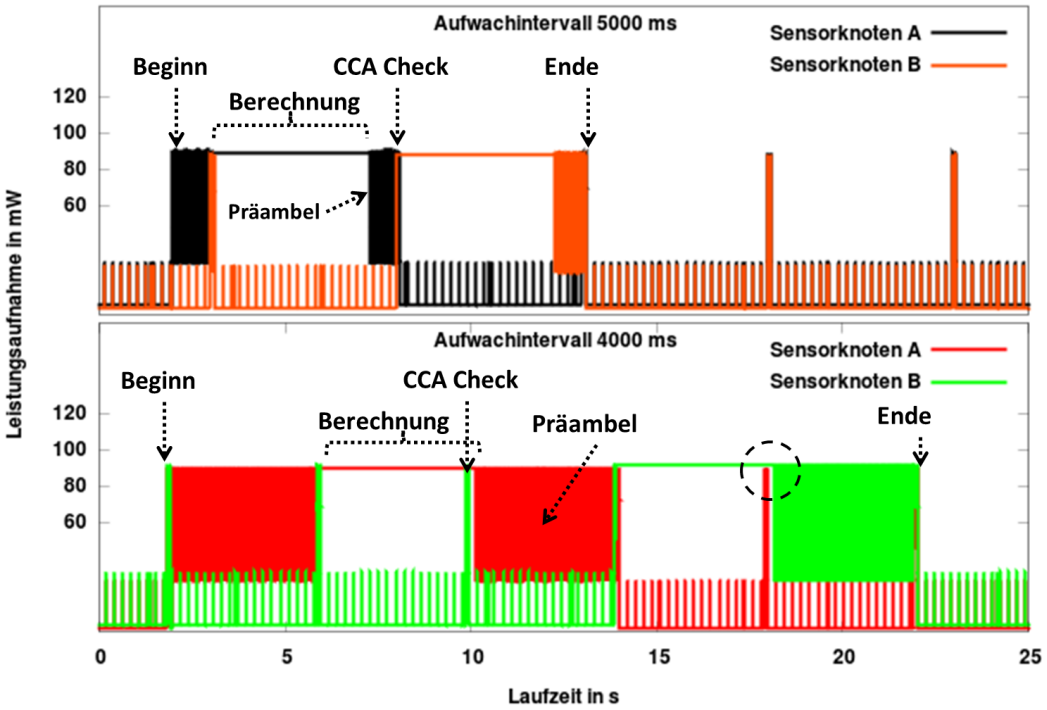


Abbildung 5.7 Leistungsaufnahme des Schlüsselaustauschs mit ECDH-ECDSA und TinyOS-LPL Aufwachintervallen von 4000 ms und 5000 ms.

$t_{LPL} = 1000ms$ ist somit von den hier untersuchten Parametrisierungen für ECDH-ECDSA bei Betrachtung des Energiebedarfs die optimale Parametrisierung.

Während bei der PING-Anwendung der Energiebedarf bei einer Verringerung oder Erhöhung im Vergleich zu der optimalen t_{LPL} Parametrisierung jeweils größer wurde, ist beim Energiebedarf von ECDH-ECDSA ein anderes Verhalten zu erkennen. Insbesondere ist der relativ große Unterschied im Energiebedarf zwischen $t_{LPL} = 4000ms$ und $t_{LPL} = 5000ms$ auffallend.

Für diesen unerwarteten Kurvenverlauf ist ein Zusammenspiel zwischen Berechnungszeit der kryptografischen Algorithmen, der Duty-Cycling Strategie von TinyOS-LPL und der Dauer des gesamten Schlüsselaustauschs verantwortlich. Dieser Zusammenhang soll im folgend erläutert werden.

Abbildung 5.7 zeigt die Leistungsaufnahme eines Schlüsselaustauschs mit ECDH-ECDSA und TinyOS-LPL für die Aufwachintervalle $4000ms$ und $5000ms$. Auf der y-Achse ist jeweils die Leistungsaufnahme in mW, auf der x-Achse die Laufzeit des Schlüsselaustauschs aufgezeichnet. Es ist jeweils der Beginn und das Ende des Schlüsselaustauschs eingetragen.

Zu erkennen ist die längere Dauer des Schlüsselaustauschs für $t_{LPL} = 4000ms$ im Vergleich zur Dauer des Schlüsselaustauschs mit $t_{LPL} = 5000ms$. Diese Tatsache ist überraschend, da im vorigen Kapitel gezeigt werden konnte, dass ein größerer Wert für t_{LPL} im allgemeinen zu einer größeren Latenz bei der Nachrichtenübertragung führt. Eine Erwartung wäre somit, dass die Dauer des Schlüsselaustauschs für $t_{LPL} = 5000ms$ länger als die Dauer für $t_{LPL} = 4000ms$ sein müsste, da in beiden Fällen die kryptografischen Berechnungen die gleiche Zeit dauern und somit die zusätzliche Latenz die Dauer des Schlüsselaustauschs verlängern sollte.

Neben Beginn und Ende des Schlüsselaustauschs ist in Abbildung 5.7 beispielhaft auch die Dauer der kryptografischen Berechnungen auf Sensorknoten A eingezeichnet. Innerhalb dieser Berechnung wird das Zertifikat von Sensorknoten B verifiziert und anschließend ein gemeinsamer Schlüssel berechnet. Nach der Schlüsselberechnung überträgt Sensorknoten A gemäß Protokollablauf sein eigenes Zertifikat an Sensorknoten B. In beiden Abbildungen ist die zu dieser Nachrichtenübertragung gehörende Präambel eingezeichnet.

Die unterschiedliche Dauer des Schlüsselaustauschs kommt hierdurch zustande, dass die Sensorknoten bei $t_{LPL} = 5000ms$ insgesamt kürzere Präambeln senden müssen als bei $t_{LPL} = 4000ms$. Der Grund für die längere Präambel ist die Aufwachzeit des Sensorknotens B, gekennzeichnet durch den CCA-Check. Während bei $t_{LPL} = 5000ms$ der CCA-Check während der Präambel von Sensorknoten A stattfindet und die Nachrichtenübertragung somit zu diesem Zeitpunkt durchgeführt werden kann, wacht Sensorknoten A bei $t_{LPL} = 4000ms$ zu einem Zeitpunkt auf, zu dem Sensorknoten A die Berechnungen durchführt. Hierdurch ist zu erkennen, dass die Präambel bei $t_{LPL} = 4000ms$ deutlich länger und bis zum nächsten CCA-Check gesendet werden muss. Es ergibt sich somit ein Timing- oder Seiteneffekt, da die Berechnungszeit von 4 Sekunden im Zusammenspiel mit dem TinyOS-LPL Aufwachintervall von 4 Sekunden zu einer längeren Dauer des Schlüsselaustauschs führt.

Der gleiche Seiteneffekt lässt sich nach der Berechnung von Sensorknoten B beobachten, gekennzeichnet durch einen Kreis bei Sekunde 18 für $t_{LPL} = 5000ms$. Auch hier wird der CCA-Check vor dem Ende der Berechnung durchgeführt und somit die folgende Präambel deutlich länger. Insgesamt gilt somit, dass der Energiebedarf für ECDH-ECDSA für $t_{LPL} = 5000ms$ deutlich geringer ist, da die Dauer des Schlüsselaustauschs geringer ist. Für $t_{LPL} = 4000ms$ wird deutlich mehr Energie für das Versenden der Präambeln aufgewendet.

Es ist zu erwarten, dass ähnliche Seiteneffekte auch für andere Berechnungszeiten und t_{LPL} Werte auftreten, beispielsweise beim Einsatz eines anderen asymmetrischen Algorithmus zur Signaturverifikation oder bei einer anderen Schlüssellänge. Je nach eingesetztem Algorithmus und den daraus resultierenden Berechnungszeiten ist somit zu erwarten, dass es zu großen Schwankungen im Energiebedarf kommen kann. Weiterhin ist davon auszugehen, dass bei einem anderen Schlüsselaustauschprotokoll bzw. Kommunikationsverhalten innerhalb des Protokolls ähnliche Seiteneffekte den Energiebedarf beeinflussen. Hierdurch ist es notwendig, konkrete Protokollimplementierungen mittels Experimenten in Testbeds oder in AVRORA+ zu evaluieren.

Bei der Betrachtung des Energiebedarfs von ECDH-ECDSA in Abbildung 5.6a fällt weiterhin auf, dass relativ große Unterschiede zwischen den minimal und maximal gemessenen Energiebedarfswerten existieren. Diese Unterschiede sind durch den zufälligen Anschaltezeitpunkt der Funkschnittstelle zu erklären. Hierdurch wird erreicht, dass die resultierende Präambellänge von Simulationslauf zu Simulationslauf unterschiedlich ist. Da auch in einer realen Anwendung nicht unbedingt davon ausgegangen werden kann, dass alle Sensorknoten gleichzeitig angeschaltet werden können kann es somit auch dort vorkommen, dass der Energiebedarf für den Schlüsselaustausch diese Unterschiede aufweist.

Ein Vergleich des Energiebedarfs von ECDH-ECDSA unter Verwendung von TinyOS-LPL und IEEE 802.15.4 bringt ein weiteres überraschendes Ergebnis: Im Kontrast zu den Erwartungen an den Einsatz eines Medienzugriffsverfahrens mit Duty-Cycling, weist der Schlüsselaustausch mit ECDH-ECDSA und TinyOS-LPL nicht für alle Werte von t_{LPL} einen geringeren Energiebedarf im Vergleich zu IEEE 802.15.4 auf. Beispielsweise ist der Schlüsselaustausch mit einem Aufwachintervall von $4000ms$ mit durchschnittlich $1503mJ$ Energiebedarf teurer als der Schlüsselaustausch mit IEEE 802.15.4. Für andere Werte von t_{LPL}, beispielsweise alle Werte kleiner $4000ms$, ist der Schlüsselaustausch unter Verwendung von TinyOS-LPL mit unter $1253mJ$ Energiebedarf deutlich günstiger.

Die im Vergleich zu IEEE 802.15.4 höheren Kosten lassen sich auf die eingesetzte Duty-Cycling Strategie und somit indirekt auf die Dauer des Schlüsselaustauschs zurückführen. Mit steigenden Werten von t_{LPL} steigt sowohl die Dauer jeder Präambel als auch die Latenz bei jeder Nachrichtenübertragung. Dies führt im Vergleich zu IEEE 802.15.4 zu einem insgesamt länger dauernden Schlüsselaustausch und gleichzeitig zu einem deutlich höheren Energiebedarf beim Versenden jeder Nachricht.

Zur Verdeutlichung zeigt Abbildung 5.8 exemplarisch die Leistungsaufnahme des Schlüsselaustauschs mit ECDH-ECDSA und TinyOS-LPL für Aufwa-

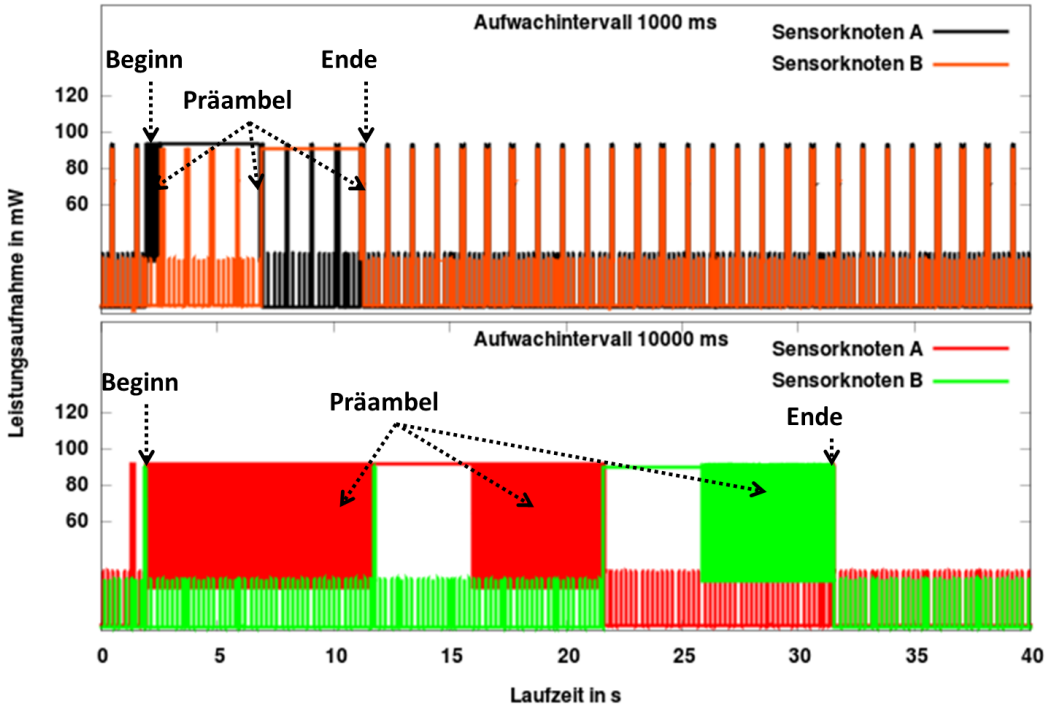


Abbildung 5.8 Leistungsaufnahme des Schlüsselaustauschs mit ECDH-ECDSA und TinyOS-LPL Aufwachintervallen von 1000 ms und 10000 ms.

chintervalle von $1000ms$ und $10000ms$. Auf der y-Achse ist jeweils die Leistungsaufnahme in mW, auf der x-Achse die Laufzeit des Schlüsselaustauschs aufgezeichnet. Zusätzlich ist jeweils der Beginn und das Ende des Schlüsselaustauschs für beide Werte von t_{LPL} eingetragen. Deutlich zu sehen ist neben den höheren Kosten für das Übertragen der Präambel auch die insgesamt längere Dauer eines Schlüsselaustausches. Bei einem Aufwachintervall von $t_{LPL} = 10000ms$ senden die Sensorknoten zwischen Sekunde 2 und 11, Sekunde 16-21 und Sekunde 26-31 Präambeln zur Nachrichtenübertragung. Für $t_{LPL} = 1000ms$ sind deutlich kürzere Präambeln notwendig, beispielsweise zwischen Sekunde 2 und 2,5.

Der Schlüsselaustausch mit $t_{LPL} = 10000ms$ dauert insgesamt etwa 30 Sekunden, für IEEE 802.15.4 ist die Dauer mit insgesamt etwa 8,5 Sekunden deutlich kürzer. Wird nun genau ein Schlüsselaustausch gemessen, überwiegen bei einer ungünstigen Wahl der Werte für t_{LPL} die zusätzlichen Kosten für die Übertragung und die längere Dauer eines Schlüsselaustausches die Einsparungen durch das Duty-Cycling der Funkschnittstelle. Hierdurch kann es somit trotz Duty-Cycling zu einem im Vergleich zu IEEE 802.15.4 höheren Energiebedarf kommen.

Der Vergleich des Energiebedarfs zwischen ECDH-ECDSA unter Verwendung von TinyOS-LPL und IEEE 802.15.4 zeigt, dass eine allgemeine Aussage über den Energiebedarf eines konkreten Schlüsselaustauschprotokolls schwierig ist. Je nach eingesetztem Medienzugriffsverfahren und je nach konkreter Parametrisierung kann es zu unterschiedlichen, zum Teil widersprüchlichen Energiebedarfswerten kommen. Somit ist es auch hier bei einer Evaluierung des Energiebedarfs notwendig, unterschiedliche Medienzugriffsverfahren mit unterschiedlichen Parametrisierungen gegeneinander zu evaluieren.

Abildung 5.6b zeigt den Energiebedarf des Schlüsselaustauschs mit Kerberos und TinyOS-LPL. Auf der x-Achse sind die unterschiedlichen Werte für das TinyOS Aufwachintervall, t_{LPL}, zu sehen. Dargestellt ist der kumulierte Energiebedarf aller beteiligten Sensorknoten (A,B und T).

Bei Kerberos ist erkennbar, dass ein größerer Wert für t_{LPL} automatisch zu einem höheren durchschnittlichen Energiebedarf führt. Der geringste Energiebedarf konnte für $t_{LPL} = 25ms$ mit $12,7mJ$ gemessen werden. Der größte Energiebedarf von $75,16mJ$ trat bei $t_{LPL} = 10000ms$ auf. Der Grund für diesen Verlauf des Energiebedarfs ist die steigende Dauer des Schlüsselaustauschs mit steigenden Werten von t_{LPL}. Die Dauer des Schlüsselaustauschs mit Kerberos ist in Abbildung 5.9 dargestellt und wird im folgenden Abschnitt erläutert. Für den Energiebedarf eines Schlüsselaustauschs mit Kerberos und TinyOS-LPL gilt, dass eine längere Dauer des Schlüsselaustausch automatisch zu einem erhöhten Energiebedarf führt. Seiteneffekte wie bei ECDH-ECDSA

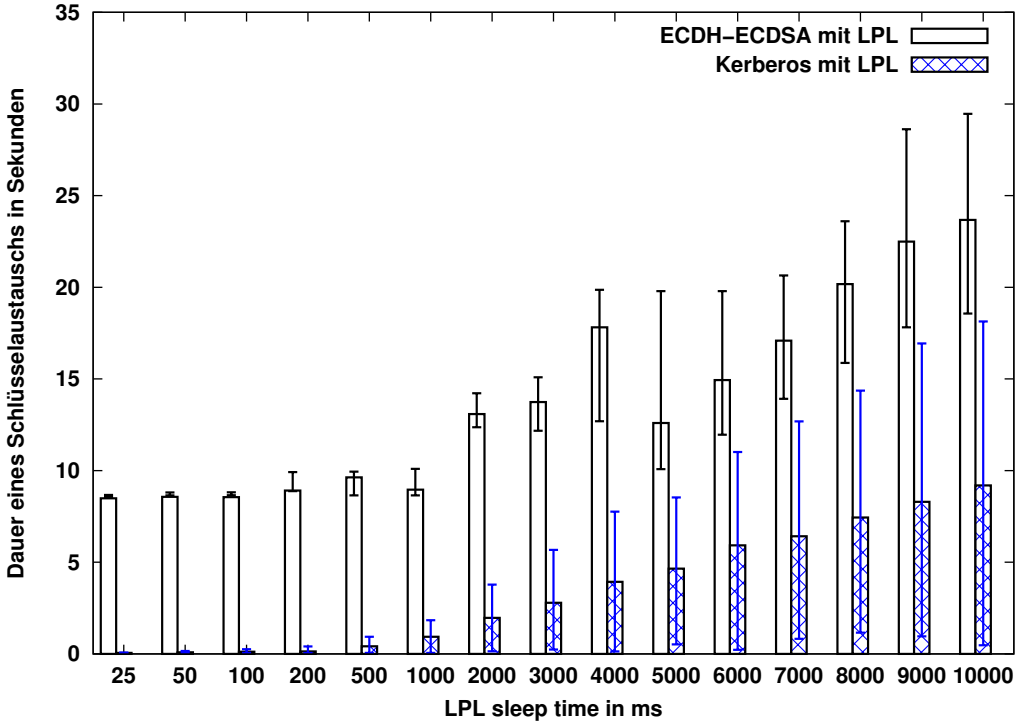


Abbildung 5.9 Dauer eines Schlüsselaustauschs mit TinyOS-LPL.

sind aufgrund der kurzen Berechnungszeiten der kryptografischen Algorithmen (etwa $0,02s$ je AES Operation) nicht aufgetreten.

Der Schlüsselaustausch mit Kerberos ist bis zu einem t_{LPL} Wert von $100ms$ geringer als der Energiebedarf des Schlüsselaustauschs mit IEEE 802.15.4. Für größere Werte steigen die Kosten für das Duty-Cycling aufgrund der Präambel so stark, dass keine Energieeinsparung im Vergleich zu IEEE 802.15.4 möglich ist.

Dauer eines Schlüsselaustauschs

Abbildung 5.9 zeigt die Dauer eines Schlüsselaustauschs mit Kerberos und ECDH-ECDSA unter Verwendung von TinyOS-LPL.

Für ECDH-ECDSA wird nochmals deutlich, dass ein Zusammenhang zwischen der Dauer eines Schlüsselaustauschs und dem Energiebedarf besteht. Insbesondere lassen sich die Schwankungen im Energiebedarf beim Schlüsselaustausch mit ECDH-ECDSA und TinyOS-LPL erklären.

Eine Erwartung aus dem vorigen Kapitel ist, dass die Erhöhung von t_{LPL} zu einer insgesamt längeren Dauer des Schlüsselaustauschs führt, da die Latenz pro Nachrichtenübertragung durch die längere Präambel steigt, siehe Tabelle 4.4. Im Gegensatz hierzu ist in Abbildung 5.9 zu erkennen, dass die Dauer eines Schlüsselaustauschs mit ECDH-ECDSA nicht zwangsläufig ansteigt, wenn sich der Wert für t_{LPL} erhöht. Beispielsweise führt eine Erhöhung von

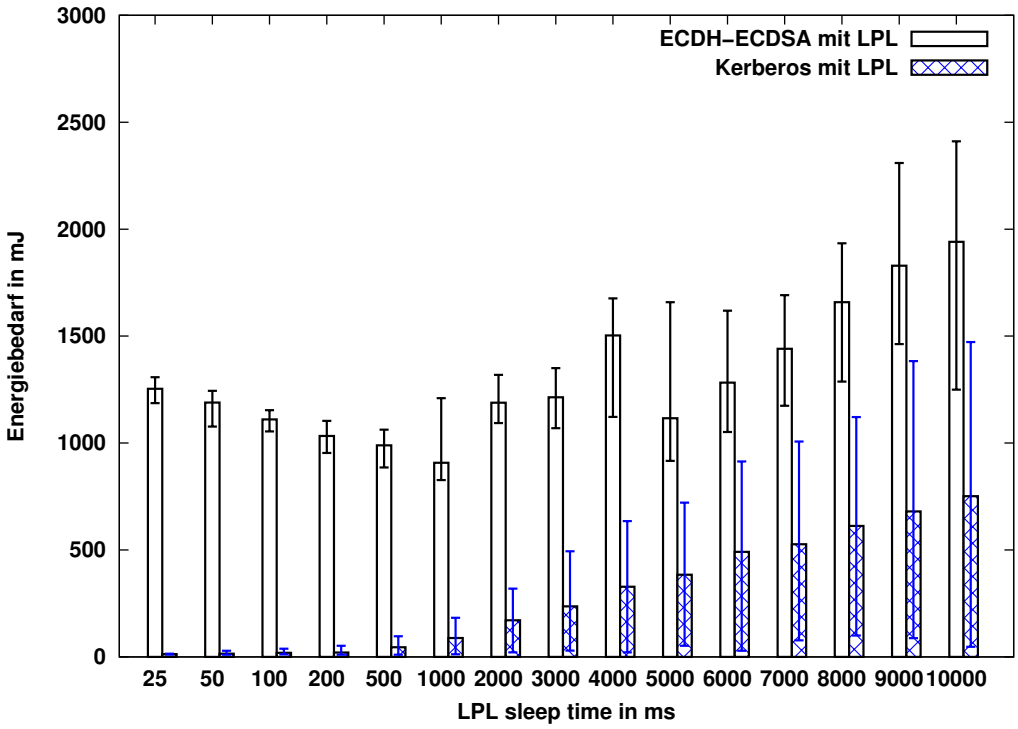


Abbildung 5.10 Energiebedarf des Schlüsselaustauschs mit TinyOS-LPL im Direktvergleich.

t_{LPL} von $4000ms$ auf $5000ms$ zu einer Verringerung der Dauer des Schlüsselaustauschs, von 19,97 Sekunden auf 11,06 Sekunden.

Für Kerberos gilt jedoch, dass eine Erhöhung von t_{LPL} sowohl die Dauer des Schlüsselaustauschs und somit auch den Energiebedarf erhöht.

Bei der Betrachtung der Dauer des Schlüsselaustauschs ist fraglich, welche Parametrisierung von t_{LPL} innerhalb einer realen Anwendung sinnvoll ist. Beispielsweise dauert ein Schlüsselaustausch mit ECDH-ECDSA bei $t_{LPL} = 10000ms$ durchschnittlich etwa 7 Sekunden mit Kerberos und 24 Sekunden mit ECDH-ECDSA. Zu beachten ist hierbei, dass dies für lediglich zwei Sensorknoten gilt. In einem größeren Netz wie beispielsweise bei der FleGSens-Anwendung mit bis zu 200 Sensorknoten ist zweifelhaft, ob eine Schlüsselaustauschdauer von 24 Sekunden zwischen jeweils zwei Sensorknoten praktikabel ist.

Vergleich von Kerberos und ECDH-ECDSA

Abbildung 5.10 zeigt den Energiebedarf eines Schlüsselaustausches mit ECDH-ECDSA und Kerberos im Direktvergleich. Für alle Werte von t_{LPL} ist ECDH-ECDSA teurer als der Schlüsselaustausch mit Kerberos. Während der Schlüsselaustausch mit ECDH-ECDSA unter Verwendung von IEEE 802.15.4 100-mal so teuer wie ein Schlüsselaustausch mit Kerberos ist, schwankt der Un-

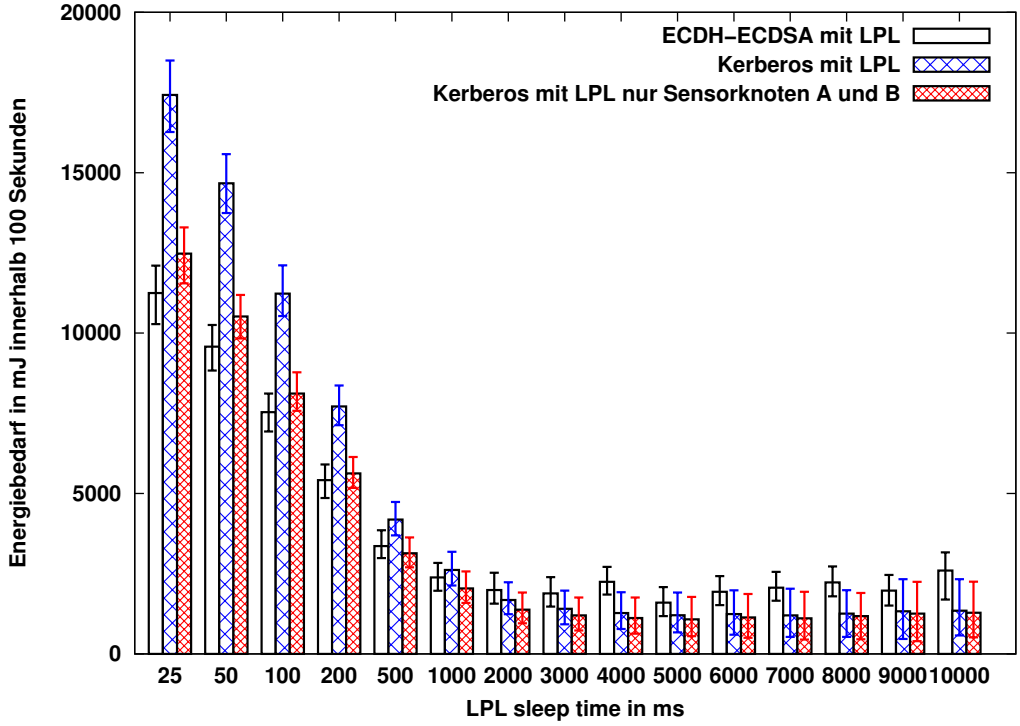


Abbildung 5.11 Energiebedarf des Schlüsselaustauschs mit TinyOS-LPL im Direktvergleich über einen Messzeitraum von 100 Sekunden.

terschied bei der Verwendung von TinyOS-LPL zwischen 2, 25 und 98. So ist der Schlüsselaustausch mit ECDH-ECDSA bei einem Aufwachintervall von $10000ms$ nur 2, 25-mal so teuer wie der Schlüsselaustausch mit Kerberos, bei einem Aufwachintervall von $25ms$ jedoch 98-mal.

Aus diesem Vergleich kann geschlossen werden, dass ein Vergleich dieser beiden Schlüsselaustauschprotokolle stark von der Parametrisierung des Medienzugriffs abhängt. Werden nur wenige Parametrisierungen verglichen, ist es unter Umständen möglich, dass bei einem Protokollvergleich das im Bezug auf Energiebedarf ungünstige Schlüsselaustauschprotokoll ausgewählt wird. Würde man beispielsweise nur den Energiebedarf bei $t_{LPL} = 10000ms$ vergleichen, könnte man zum Schluss kommen, dass der Unterschied des Energiebedarfs zwischen ECDH-ECDSA und Kerberos nur gering ausfällt.

Es gilt somit, dass bei einem Vergleich des Energiebedarfs verschiedener Protokolle jeweils umfassende Evaluierungen mit unterschiedlichen Parametrisierungen des Medienzugriffsverfahren notwendig sind. Ein Vergleich von nur einzelnen Parametrisierungen kann zu falschen Schlussfolgerungen führen.

Vergleich über einen festen Messzeitraum

Abbildung 5.11 zeigt den Energiebedarf für ECDH-ECDSA und Kerberos über einen Messzeitraum von 100 Sekunden. Neben dem Energiebedarf für ECDH-ECDSA und Kerberos ist außerdem der Energiebedarf der Sensorknoten A und B, ohne Sensorknoten T, bei der Nutzung von Kerberos dargestellt. Im Gegensatz zu den vorigen Ergebnissen ist ECDH-ECDSA nun nicht grundsätzlich teurer als ein Schlüsselaustausch mit Kerberos. Für Aufwachintervalle unter $500ms$ hat ECDH-ECDSA einen geringeren Energiebedarf, sogar wenn beim Austausch per Kerberos nur die Sensorknoten A und B in die Energiebedarfsermittlung einbezogen werden.

Für beide Schlüsselaustauschprotokolle konnte der größte Energiebedarf für $t_{LPL} = 25ms$ mit $11247mJ$ für ECDH-ECDSA und $12478mJ$ bzw. $17423mJ$ für Kerberos gemessen werden. Bei der Betrachtung des Energiebedarfs von Kerberos wird deutlich, warum es sinnvoll sein kann sowohl alle beteiligten Sensorknoten als auch nur Sensorknoten A und B zu betrachten. Die Vertrauensinstanz muss zu jedem Zeitpunkt im Netz verfügbar sein, da ohne sie kein Schlüsselaustausch durchgeführt werden kann. Der somit zusätzlich notwendige Sensorknoten bzw. dessen Energiebedarf sollte bei einem Vergleich des Energiebedarfs verschiedener Protokolle immer mitbetrachtet werden.

Wie schon bei IEEE 802.15.4 beobachtet, überwiegt auch in dieser Betrachtung über den Messzeitraum von 100 Sekunden der Grundbedarf für das Duty-Cycling. Der Einfluss der Schlüsselaustauschprotokolle ist relativ gering. So entfallen etwa $1446mJ$ bzw. $14,2mJ$ des Energiebedarfs bei ECDH-ECDSA und Kerberos auf den Schlüsselaustausch, der restliche Energiebedarf wird alleine durch das periodische Anschalten der Funkschnittstelle verursacht.

Im Vergleich zu IEEE 802.15.4 wird deutlich, dass TinyOS-LPL für alle Werte von t_{LPL} einen deutlich geringeren Energiebedarf aufweist. Der Einsatz von TinyOS-LPL verringert den Energiebedarf deutlich.

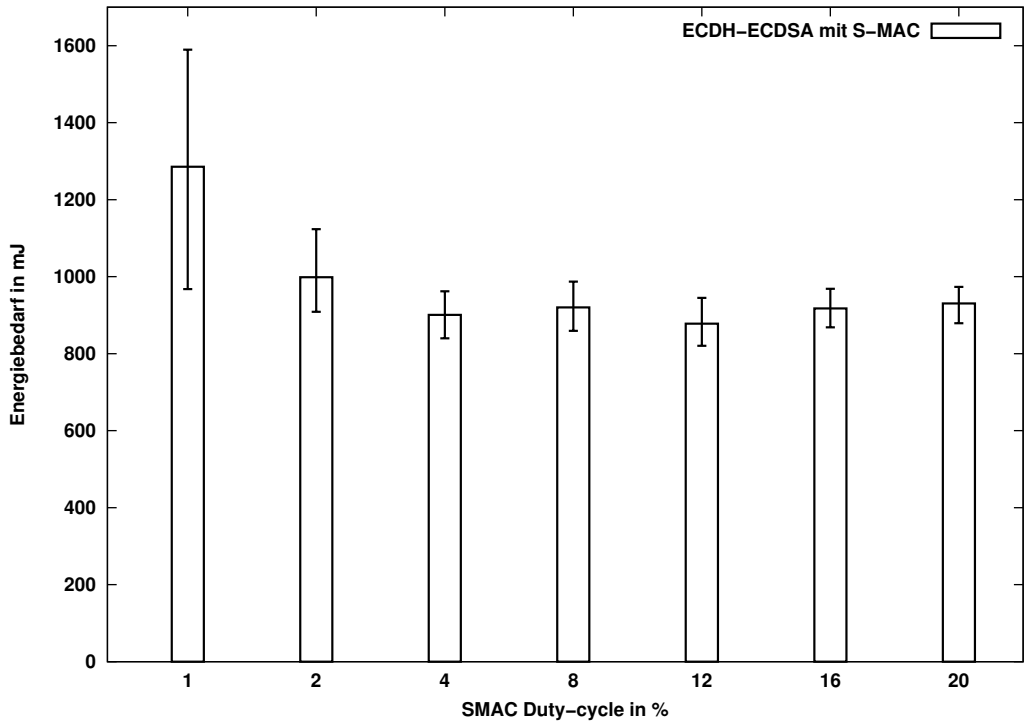
5.4.4 S-MAC

Im Rahmen dieses Abschnitts wird der Schlüsselaustausch mittels ECDH-ECDSA und Kerberos unter Verwendung von S-MAC evaluiert.

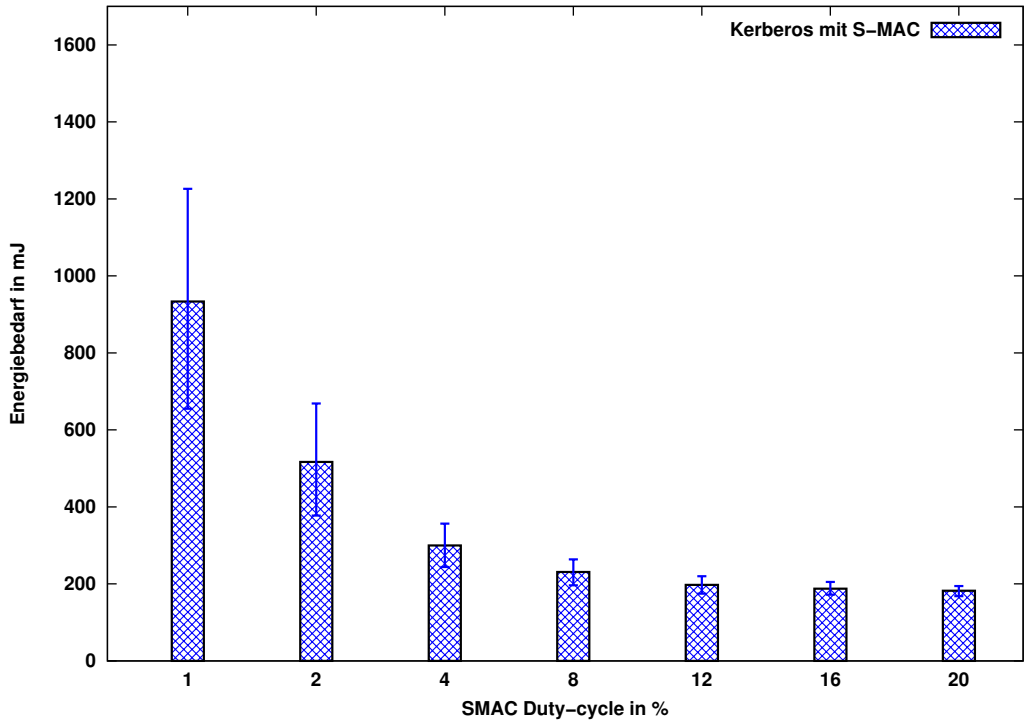
Energiebedarf pro Schlüsselaustausch

Abbildung 5.12a zeigt den Energiebedarf eines Schlüsselaustauschs mit ECDH-ECDSA, Abbildung 5.12b den Energiebedarf von Kerberos unter Verwendung von S-MAC. Auf den x-Achsen sind die unterschiedlichen Werte für den S-MAC Duty-Cycle, t_{DC}, zu sehen.

Für ECDH-ECDSA und S-MAC ist zu erkennen, dass kaum Unterschiede im Energiebedarf für die verschiedenen Werte des Duty-Cycles bestehen. Ledig-



(a) Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA



(b) Energiebedarf des Schlüsselaustauschs mit Kerberos

Abbildung 5.12 Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von S-MAC.

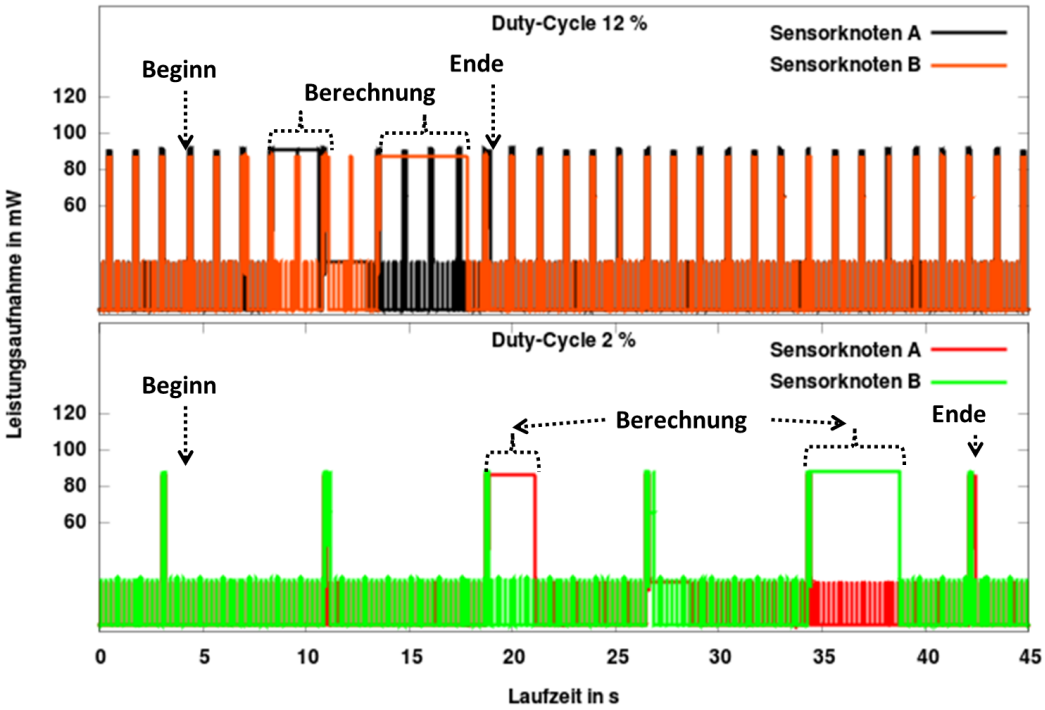


Abbildung 5.13 Leistungsaufnahme des Schlüsselaustauschs mit ECDH-ECDSA und S-MAC für einen Duty-Cycle von 2% und 12%.

lich bei einem Duty-Cycle von 1% ist mit $1298mJ$ ein leicht erhöhter Energiebedarf messbar. Für die anderen Duty-Cycle ergibt sich ein Energiebedarf zwischen $880mJ$ und $990mJ$. Der S-MAC Duty-Cycle von 1% ist so gering, dass insgesamt die Dauer des Schlüsselaustausch so stark ansteigt ($>$ 60 Sekunden), dass dies trotz Duty-Cycling der Funkschnittstelle zu einem höheren Energiebedarf führt.

Entgegen früherer Annahmen kann also nicht bestätigt werden, dass ein höherer Duty-Cycle bei S-MAC automatisch einen erhöhten Energiebedarf verursacht. Diese Erkenntnis ist besonders überraschend, da die Dauer des Schlüsselaustauschs, wie im folgenden Abschnitt gezeigt, mit einer Erhöhung des Duty-Cycles abnimmt. Bei S-MAC gilt somit, dass kein direkter Zusammenhang zwischen Dauer des Schlüsselaustauschs und dem Energiebedarf besteht. Vielmehr ist es so, dass der Energiebedarf für einen Schlüsselaustausch mit ECDH-ECDSA für $t_{DC} \geq 2\%$ nahezu gleich bleibt.

Abbildung 5.13 zeigt zur Erläuterung dieser Tatsache die Leistungsaufnahme des Schlüsselaustauschs mit ECDH-ECDSA und S-MAC für einen Duty-Cycle von 2% und 12%. Für beide Duty-Cycles sind sowohl der Beginn als auch das Ende des Schlüsselaustauschs eingezeichnet. Der Beginn ist hierbei der Zeitpunkt, an dem Sensorknoten A den Schlüsselaustausch startet, die

erste Nachrichtenübertragung findet in der ersten darauffolgenden Wachphase statt.

Anhand der Leistungsaufnahme ist zu erkennen, dass der Schlüsselaustausch bei einem Duty-Cycle von 2% fast 40 Sekunden dauert, während bei einem Duty-Cycle von 12% die Dauer etwa 12 Sekunden beträgt. Trotz der fast 3,5-mal längeren Dauer ist der Energiebedarf des Schlüsselaustauschs für beide Duty-Cycles mit $998mJ$ und $878mJ$ nur geringfügig unterschiedlich. Dies lässt sich damit erklären, dass die Sensorknoten bei einem Duty-Cycle von 2% die Funkschnittstelle während des Schlüsselaustauschs wesentlich häufiger anschalten und somit hierfür einen höheren Energiebedarf aufbringen müssen. Beispielsweise schalten die Sensorknoten für einen Duty-Cycle von 12% die Funkschnittstelle während des Schlüsselaustauschs nur insgesamt 5-mal an (Leistungsaufnahme etwa $90mW$, zweimal wird nach der Wachphase die kryptografische Berechnung durchgeführt). Bei einem Duty-Cycle von 2% ist die Funkschnittstelle hingegen für jeden Sensorknoten 10-mal angeschaltet.

Die Leistungsaufnahme der Wachphase (etwa $80mW$) ist im Vergleich zur Leistungsaufnahme der Schlafphase ($< 5mW$) sehr groß. Abbildung 5.14 zeigt hierfür einen detaillierten Ausschnitt der Leistungsaufnahme des Schlüsselaustausch mit ECDH-ECDSA und S-MAC. Für beide Duty-Cycles ist die Leistungsaufnahme für Wach- und Schlafphase kenntlich gemacht.

Insgesamt gilt somit, dass die längere Dauer in Verbindung mit einer geringeren Anzahl an Wachphasen dazu führt, dass sich der Energiebedarf für unterschiedliche Duty-Cycles nur geringfügig unterscheidet.

Abbildung 5.12b zeigt den Energiebedarf von Kerberos unter Verwendung von S-MAC. Deutlich ist zu erkennen, dass der Energiebedarf mit steigendem Duty-Cycle sinkt. Der geringste Energiebedarf von $182mJ$ zeigt sich bei einem Duty-Cycle von 20%. Der größte Energiebedarf von $933mJ$ und einem Duty-Cycle von 1% ist im Vergleich zu TinyOS-LPL und IEEE 802.15.4 deutlich höher. Aufgrund der im Vergleich zu anderen Medienzugriffsverfahren hohen Dauer des Schlüsselaustauschs erhöht sich der Energiebedarf für die Kommunikation stark. So entfallen $2,5mJ$ auf die kryptografischen Operationen, die restlichen $930mJ$ auf Kommunikation bzw. Duty-Cycling der Funkschnittstelle durch S-MAC.

Vergleich von Kerberos und ECDH-ECDSA

Im Direktvergleich, Abbildung 5.15, zeigt sich, dass ECDH-ECDSA zwischen 5 und 1,4-mal so teuer wie der Schlüsselaustausch mit Kerberos ist.

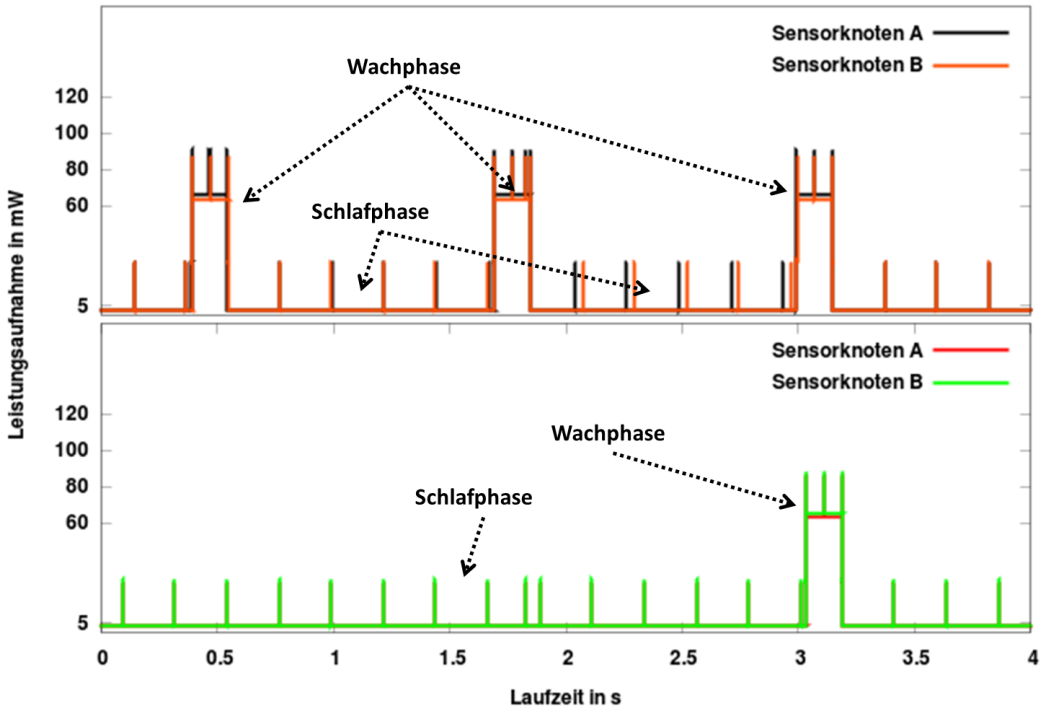


Abbildung 5.14 Leistungsaufnahme der Wach- und Schlafphase während eines Schlüsselaustauschs mit ECDH-ECDSA und S-MAC für einen Duty-Cycle von 2% und 12%.

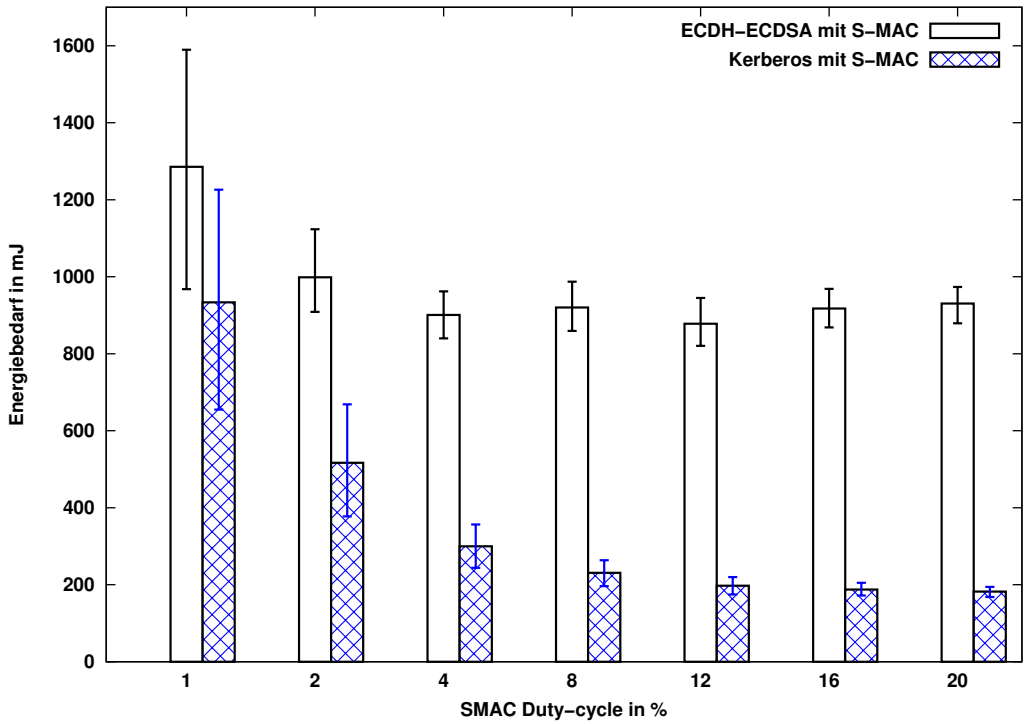


Abbildung 5.15 Energiebedarf des Schlüsselaustauschs mit S-MAC im Direktvergleich.

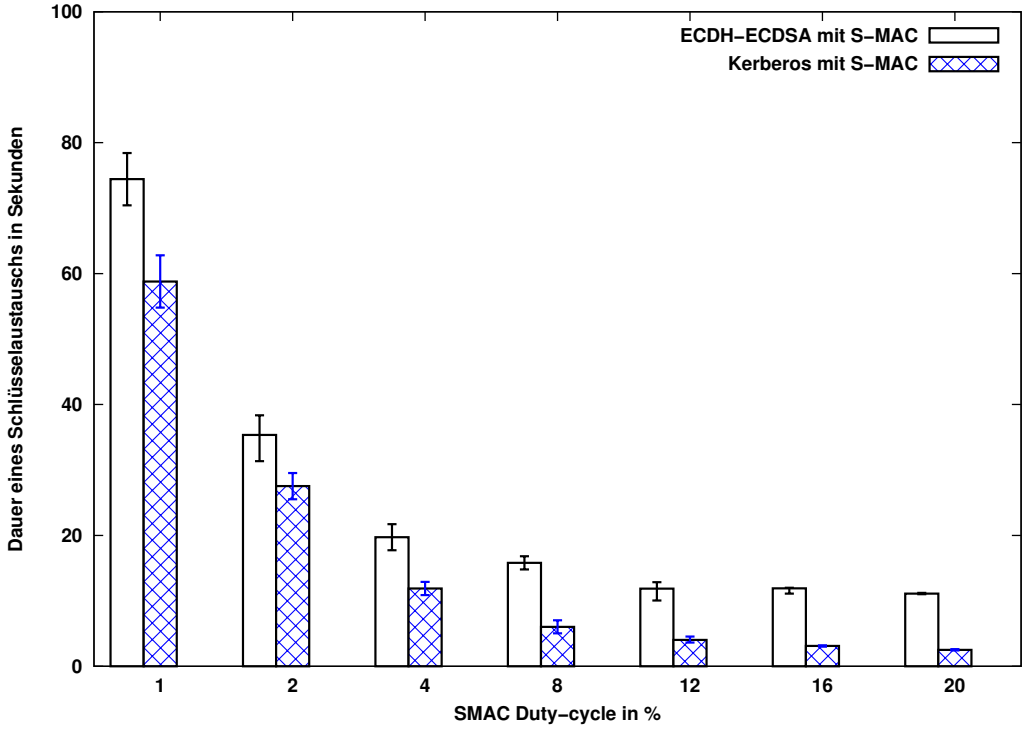


Abbildung 5.16 Dauer eines Schlüsselaustauschs mit S-MAC.

Dauer eines Schlüsselaustauschs

Abbildung 5.16 zeigt die Dauer eines Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von S-MAC.

Ein Schlüsselaustausch mit ECDH-ECDSA hat eine durchschnittliche Dauer zwischen 74 und 11 Sekunden, wobei wie im vorigen Abschnitt gezeigt etwa 8 Sekunden auf die kryptografischen Berechnungen zurückzuführen sind.

Der Schlüsselaustausch mit Kerberos hat eine Dauer zwischen 58 und 2,5 Sekunden, hiervon entfallen etwa 0,044 Sekunden auf die kryptografischen Berechnungen. Je kleiner der Duty-Cycle, desto länger ist die Dauer eines Schlüsselaustauschs, wobei der Unterschied ab einem Duty-Cycle von 8% unter 2 Sekunden bei ECDH-ECDSA und unter 1 Sekunde bei Kerberos sinkt.

Im Vergleich zum Energiebedarf in Abbildung 5.15 zeigt sich, dass ein Zusammenhang zwischen Dauer des Schlüsselaustauschs und Energiebedarf nur bei Kerberos besteht. Ein geringerer Duty-Cycle führt bei Kerberos zu einem erhöhten Energiebedarf und einer längeren Dauer des Schlüsselaustauschs.

Für ECDH-ECDSA gilt dieser Zusammenhang nicht, dies wurde im vorigen Abschnitt bereits dargelegt.

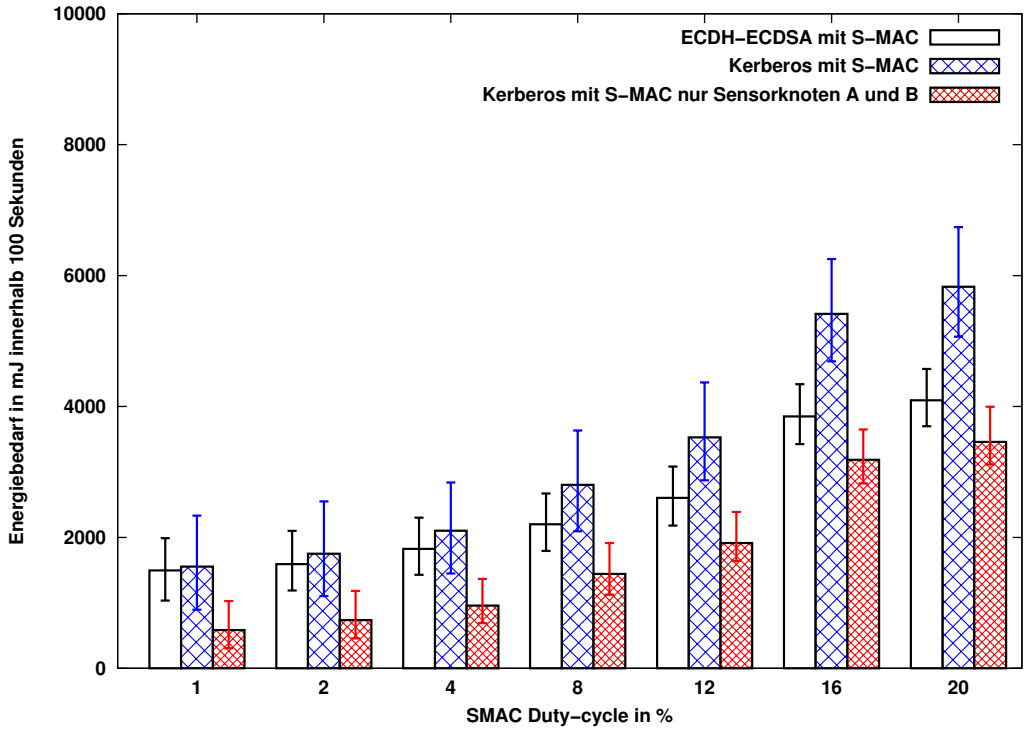


Abbildung 5.17 Energiebedarf des Schlüsselaustauschs mit S-MAC im Direktvergleich über einen Messzeitraum von 100 Sekunden.

Vergleich über einen festen Messzeitraum

Abbildung 5.17 zeigt den Energiebedarf für ECDH-ECDSA und Kerberos über einen Messzeitraum von 100 Sekunden.

Für beide Schlüsselaustauschverfahren ist zu erkennen, dass der Energiebedarf mit zunehmendem Duty-Cycle ebenfalls ansteigt. Der Energiebedarf von Kerberos ist dabei wie schon bei IEEE 802.15.4 und TinyOS-LPL beobachtet für alle Duty-Cycle größer als der Energiebedarf von ECDH-ECDSA.

Betrachtet man nur die Sensorknoten A und B, so zeigt Kerberos einen geringeren Energiebedarf. Wie schon bei IEEE 802.15.4 und TinyOS-LPL beobachtet, überwiegt auch in dieser Betrachtung über den Messzeitraum von 100 Sekunden der Grundbedarf für das Duty-Cycling, der Einfluss der Schlüsselaustauschprotokolle ist relativ gering.

Im Vergleich zu IEEE 802.15.4 wird deutlich, dass S-MAC für alle Werte von t_{DC} einen deutlich geringeren Energiebedarf aufweist. Beim Vergleich von TinyOS-LPL und S-MAC ist es abhängig von der konkreten Parametrisierung, welches Medienzugriffsverfahren den geringsten Energiebedarf verursacht. Insgesamt ist es mit TinyOS-LPL jedoch möglich, einen geringeren Energiebedarf im Messzeitraum zu erreichen (Vergleiche $t_{LPL} = 10000ms$ in Abbildung 5.11).

5.4.5 TDMA-MAC

Im Rahmen dieses Abschnitts wird der Schlüsselaustausch mittels ECDH-ECDSA und Kerberos unter Verwendung von TDMA-MAC evaluiert.

Energiebedarf eines Schlüsselaustauschs

Abbildung 5.18a zeigt den Energiebedarf eines Schlüsselaustauschs mit ECDH-ECDSA, Abbildung 5.18b den Energiebedarf von Kerberos unter Verwendung von TDMA-MAC. Auf den x-Achsen sind die unterschiedlichen Werte für den TDMA-MAC Duty-Cycle, t_{DC}, zu sehen.

Für ECDH-ECDSA ist zu erkennen, dass ein größerer Duty-Cycle zu einem erhöhten Energiebedarf führt. Der Energiebedarf liegt für die hier untersuchten Duty-Cycle Werte zwischen $809mJ$ und $1013mJ$. Die Energiebedarfswerte sind somit nahezu vergleichbar zu den bei S-MAC ermittelten Energiebedarfswerten.

Der Energiebedarf von Kerberos und TDMA-MAC ist in Abbildung 5.18b zu sehen. Wie bei S-MAC ist zu erkennen, dass ein größerer Duty-Cycle nicht zu einem erhöhten Energiebedarf führt. Auch für TDMA-MAC gilt somit entgegen früherer Annahmen, dass ein höherer Duty-Cycle bei TDMA-MAC nicht automatisch einen erhöhten Energiebedarf führt. Die Begründung ist die gleiche Begründung wie schon bei S-MAC, vergleiche die Argumentation zu Abbildungen 5.13 und 5.14.

Vergleich von ECDH-ECDSA und Kerberos

Im Direktvergleich, Abbildung 5.19, zeigt sich, dass ECDH-ECDSA zwischen 5 und 30-mal so teuer wie der Schlüsselaustausch mit Kerberos ist. Für einen Duty-Cycle von 20% ist ECDH-ECDSA 30-mal teurer als Kerberos, der geringste Unterschied konnte bei einem Duty-Cycle von 1% gemessen werden.

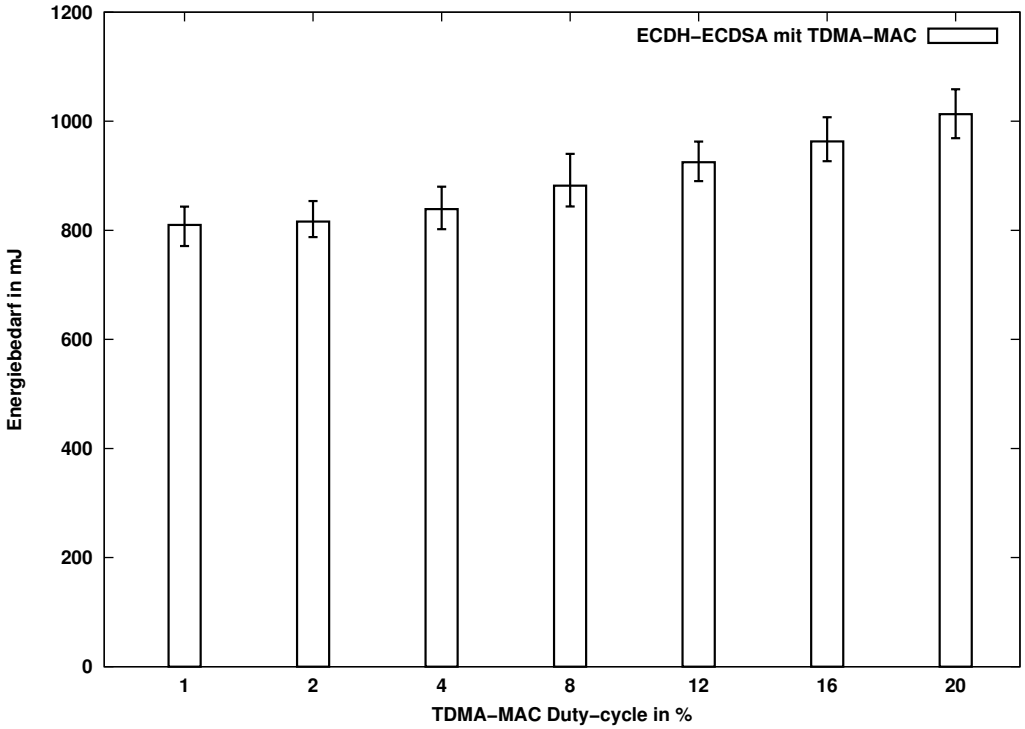
Dauer eines Schlüsselaustauschs

Abbildung 5.20 zeigt die Dauer eines Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von TDMA-MAC.

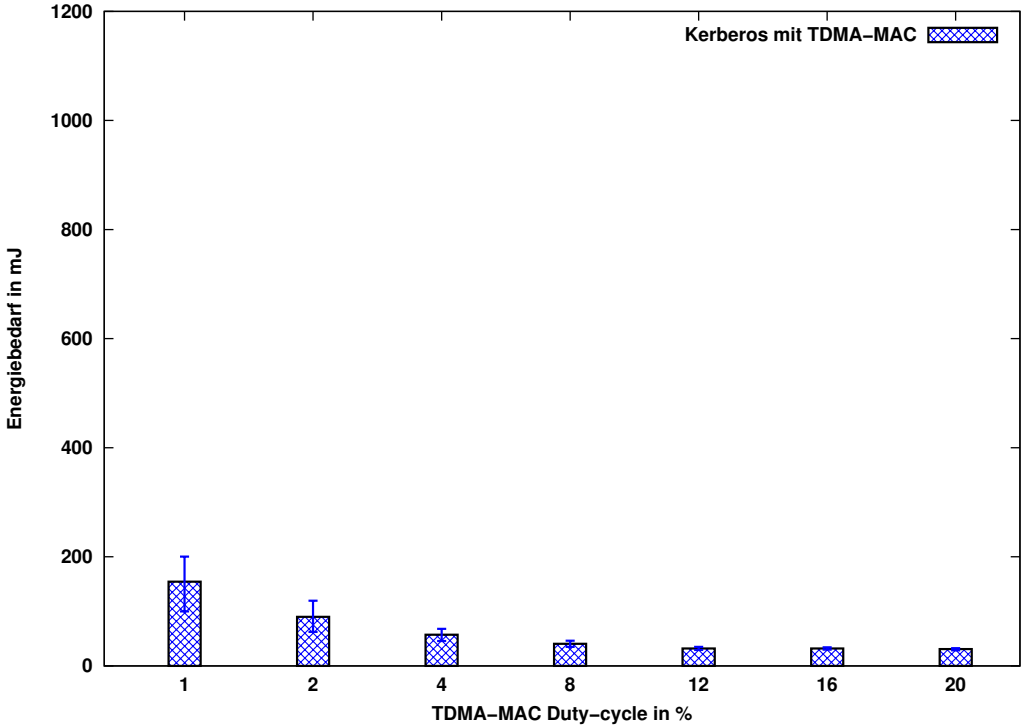
Der Schlüsselaustausch mit ECDH-ECDSA hat eine durchschnittliche Dauer zwischen $19,5$ und $9,8$ Sekunden. Der Schlüsselaustausch mit Kerberos hat eine Dauer zwischen $9,9$ und $0,057$ Sekunden. Je kleiner der Duty-Cycle, desto länger ist die Dauer eines Schlüsselaustauschs. Der Zusammenhang zwischen Dauer des Schlüsselaustauschs und Energiebedarf ist vergleichbar mit den Zusammenhängen bei S-MAC.

Energiebedarf über einen festen Messzeitraum

Abbildung 5.21 zeigt den Energiebedarf für ECDH-ECDSA und Kerberos mit TDMA-MAC über einen Messzeitraum von 100 Sekunden. Für beide Schlüs-



(a) Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA



(b) Energiebedarf des Schlüsselaustauschs mit Kerberos

Abbildung 5.18 Energiebedarf des Schlüsselaustauschs mit ECDH-ECDSA und Kerberos unter Verwendung von TDMA-MAC.

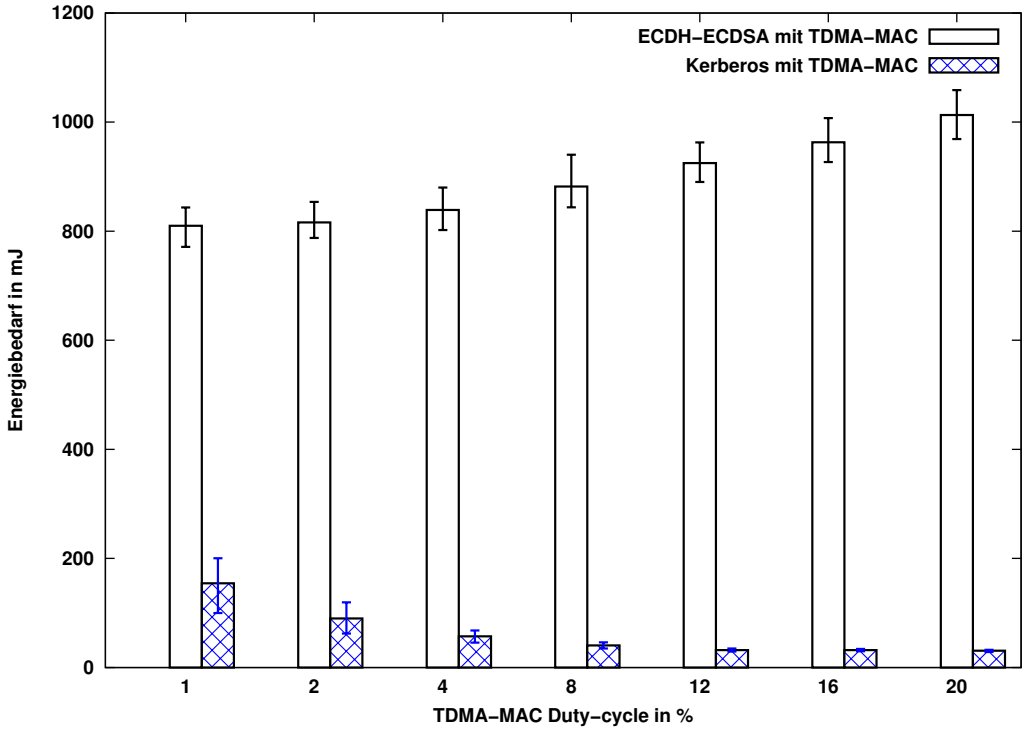


Abbildung 5.19 Energiebedarf des Schlüsselaustauschs mit TDMA-MAC im Direktvergleich.

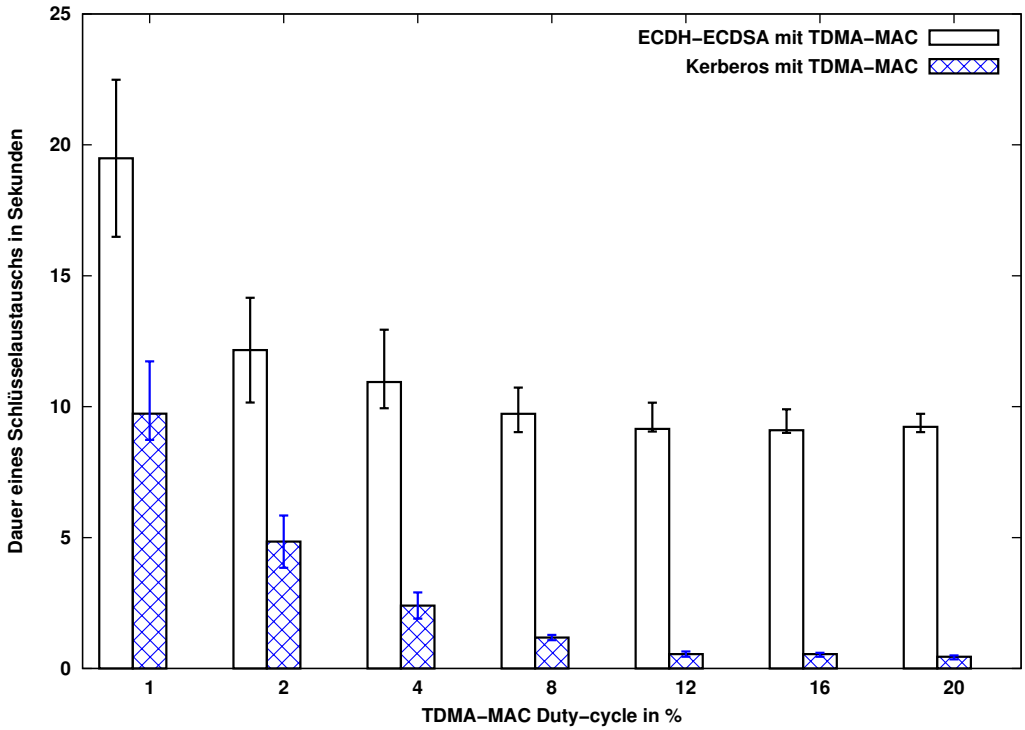


Abbildung 5.20 Dauer eines Schlüsselaustauschs mit TDMA-MAC.

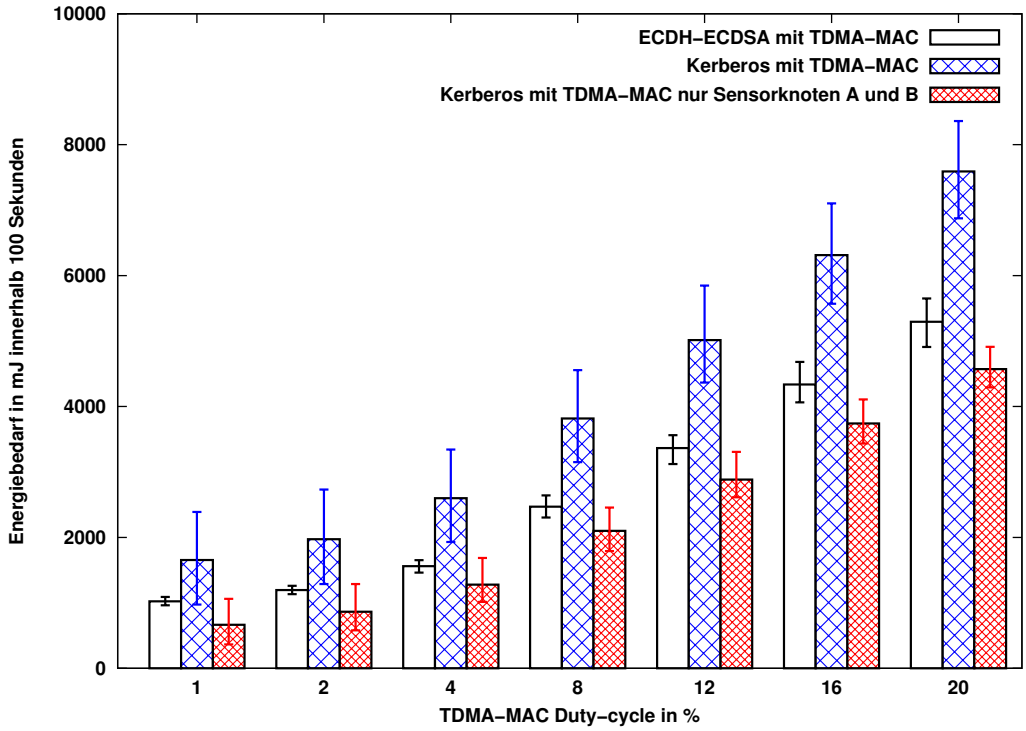


Abbildung 5.21 Energiebedarf des Schlüsselaustauschs mit TDMA-MAC im Direktvergleich über einen Messzeitraum von 100 Sekunden

selaustauschverfahren ist zu erkennen, dass der Energiebedarf mit zunehmendem Duty-Cycle ebenfalls ansteigt. Der Energiebedarf von Kerberos ist dabei für alle Duty-Cycle größer als der Energiebedarf von ECDH-ECDSA. Betrachtet man nur die Sensorknoten A und B, so zeigt Kerberos einen geringeren Energiebedarf. Auch hier sind die Erkenntnisse vergleichbar mit den Ergebnissen mit S-MAC.

5.4.6 Vergleich über alle Medienzugriffsverfahren

Tabelle 5.5 stellt den minimal und maximal erreichbaren Energiebedarf für genau einen Schlüsselaustausch mit ECDH-ECDSA und Kerberos für die verschiedenen Medienzugriffsverfahren gegenüber.

Für ECDH-ECDSA ist der minimale Energiebedarf mit TDMA-MAC erreichbar. Das Medienzugriffsverfahren mit dem höchsten minimalen Energiebedarf ist IEEE 802.5.4 mit $1446mJ$. Allerdings gilt auch, dass IEEE 802.15.4 nicht den höchsten maximalen Energiebedarf aufweist. Den höchsten maximalen Energiebedarf erreicht TinyOS-LPL mit $t_{LPL} = 10000ms$. Den geringsten maximalen Energiebedarf erreichte TDMA-MAC.

Für Kerberos konnte der minimale erreichbare Energiebedarf mit TinyOS-LPL erreicht werden. Der höchste minimale Energiebedarf konnte mit S-MAC

	ECDH-ECDSA		Kerberos	
Medienzugriffsverfahren	MIN	MAX	MIN	MAX
IEEE 802.15.4	1446 mJ	1446 mJ	14,2 mJ	14,2 mJ
TinyOS-LPL	917,22 mJ	1941 mJ	12,7 mJ	751,6 mJ
S-MAC	878,0 mJ	1285,6 mJ	182,2 mJ	933,5 mJ
TDMA-MAC	809,9 mJ	1013,0 mJ	30,8 mJ	154,5 mJ

Tabelle 5.5 Vergleich des minimal und maximal erreichbaren Energiebedarfs getrennt nach Medienzugriffsverfahren.

mit $182,2mJ$ gemessen werden. Vergleicht man den Energiebedarf von Kerberos und IEEE 802.15.4 mit dem maximalen Energiebedarf der anderen Medienzugriffsverfahren, so fällt auf, dass IEEE 802.15.4 in diesem Fall den geringsten Energiebedarf erreicht.

Beim Vergleich des Energiebedarfs von ECDH-ECDSA zu Kerberos kann somit gezeigt werden, dass ECDH-ECDSA nicht grundsätzlich einen höheren Energiebedarf aufweist als Kerberos. Je nach Parametrisierung und Wahl des Medienzugriffsverfahren kann ECDH-ECDSA sogar günstiger als Kerberos sein. Da die Parametrisierung und Wahl des Medienzugriffsverfahren typischerweise aufgrund von Anforderungen der eigentlich ausgeführten Anwendung gewählt wird, kann somit ECDH-ECDSA durchaus eingesetzt werden, ohne einen größeren Energiebedarf zu verursachen.

Der Vergleich der minimal und maximal gemessenen Energiebedarfswerte zeigt, dass die Auswahl und Parametrisierung des Medienzugriffsverfahren einen großen Einfluss auf den Energiebedarf des Schlüsselaustauschverfahrens hat. Es ist offensichtlich, dass nicht ein Medienzugriffsverfahren oder eine Parametrisierung optimal für unterschiedliche Schlüsselaustauschprotokolle oder allgemeiner für alle Anwendungen ist. Eine allgemeingültige Aussage der Form, Protokoll A hat einen geringeren Energiebedarf als Protokoll B, ist somit ohne Betrachtung des konkreten Medienzugriffsverfahren nicht sinnvoll.

5.4.7 Diskussion der Ergebnisse

Die Evaluationsergebnisse der Schlüsselaustauschprotokolle Kerberos und ECDH-ECDSA mit verschiedenen Medienzugriffsverfahren zeigen, dass im Gegensatz zu früheren Annahmen der Energiebedarf der kryptografischen Operationen als Vergleichskriterium für Schlüsselaustauschprotokolle unzu-

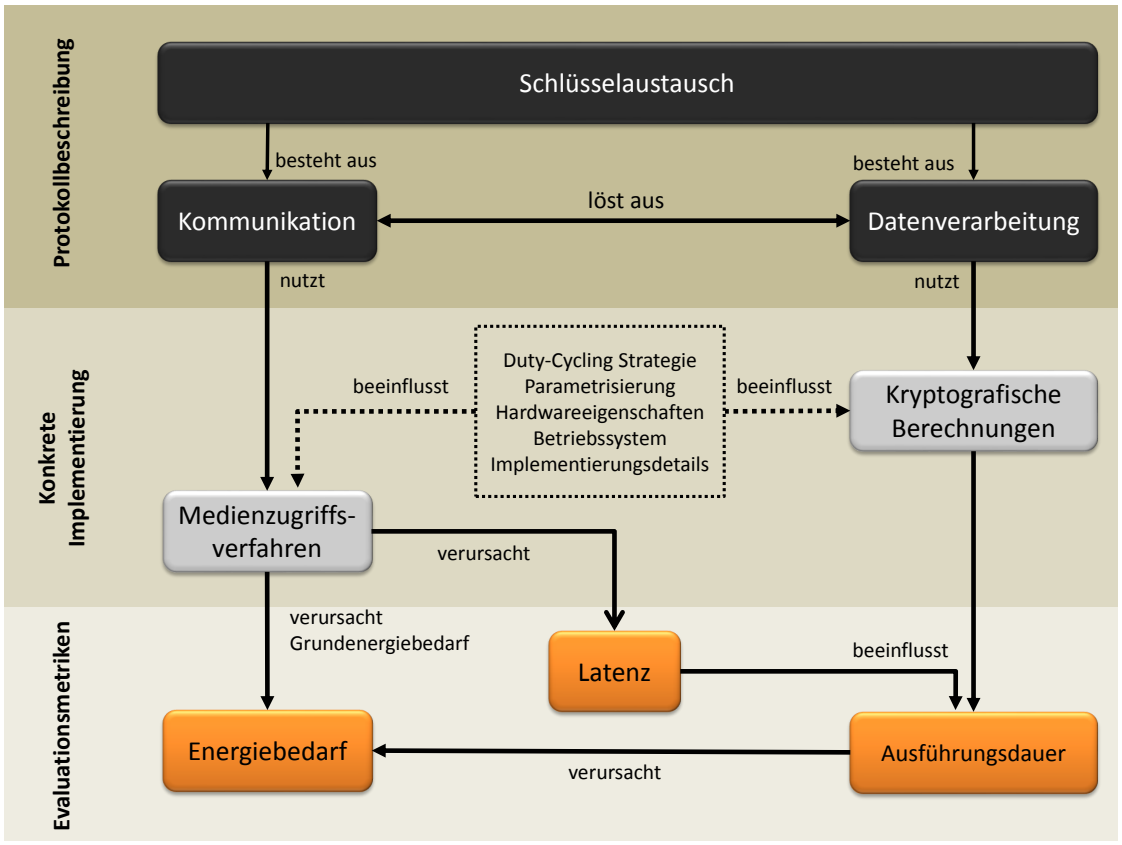


Abbildung 5.22 Zusammenfassung der Einflussfaktoren auf den Energiebedarf der Schlüsselaustauschprotokolle.

reichend ist. ECDH-ECDSA hat einen etwa 16-mal größeren Energiebedarf für die kryptografischen Berechnungen. Bei der Evaluation unter Verwendung verschiedener Medienzugriffsverfahren wurde jedoch je nach Medienzugriffsverfahren ein bis zu 100-mal größerer Energiebedarf (IEEE 802.15.4) oder auch ein lediglich 2,5-mal größerer Energiebedarf (TinyOS-LPL) von ECDH-ECDSA festgestellt. Das gewählte Medienzugriffsverfahren und die verwendete Parametrisierung haben somit insgesamt einen größeren Einfluss auf den Energiebedarf der beiden Protokolle als die durchgeführten kryptografischen Berechnungen.

Weiterhin ist fraglich, in wie weit der Vergleich des Energiebedarfs genau eines Schlüsselaustauschs sinnvoll ist. Bei der Evaluierung des Energiebedarfs über einen festen Messzeitraum wurde deutlich, dass die Unterschiede im Energiebedarf der beiden Schlüsselaustauschprotokolle für alle Medienzugriffsverfahren deutlich geringer wurden. Auch hier zeigt sich, dass der Grundenergiebedarf des gewählten Medienzugriffsverfahrens einen größeren Einfluss auf den Energiebedarf hat als das eigentliche Schlüsselaustauschverfahren. Je nach Anwendungsszenario und Ablauf der Anwendung kann es somit sinnvoll sein, verschiedene Evaluationsmetriken zu vergleichen.

Die Ergebnisse der Evaluierung zeigen auch, dass eine Verallgemeinerung der Aussagen zum Energiebedarf verschiedener Protokolle schwierig ist. So konnte beispielsweise bei der Evaluierung von ECDH-ECDSA mit TinyOS-LPL gezeigt werden, dass unter Umständen Timing-Effekte bei der Ausführung des Schlüsselaustauschs im Zusammenspiel mit dem Medienzugriffsverfahren den Energiebedarf deutlich beeinflussen können. So kann bei TinyOS-LPL nicht davon ausgegangen werden, dass ein größerer Wert für t_{LPL} immer zu einem höheren Energiebedarf führt. Weitere Einflussfaktoren durch beispielsweise eine komplexere Topologie, Routing oder die Anwendung sind nicht auszuschließen. Der Energiebedarf des Gesamtsystems sollte wenn möglich immer im Zusammenspiel evaluiert werden.

Abbildung 5.22 fasst die im Rahmen dieses Kapitels gewonnenen Erkenntnisse zusammen. Ausgehend von der Protokollbeschreibung der beiden untersuchten Schlüsselaustauschprotokolle wurde der Energiebedarf beider Protokolle untersucht. Einfluss auf den Energiebedarf haben dabei die Kommunikation und die Datenverarbeitung, im wesentlichen die kryptografischen Berechnungen. Bei der Kommunikation wurden unterschiedliche Medienzugriffsverfahren genutzt. Die Medienzugriffsverfahren und die kryptografischen Berechnungen wurden innerhalb der Evaluierung beeinflusst von den konkreten Implementierungen, dem Duty-Cycling des Betriebssystems, der Hardwareeigenschaften, der Parametrisierungen und der bei den Medienzugriffsverfahren genutzten Duty-Cycling Strategie.

Es konnte gezeigt werden, dass je nach eingesetztem Medienzugriffsverfahren und kryptografischem Algorithmus, ein Zusammenhang zwischen Latenz, Dauer eines Schlüsselaustauschs und dem Energiebedarf aufgrund des Duty-Cyclings der Funkschnittstelle besteht. Weiterhin konnte gezeigt werden, dass unter Umständen Seiteneffekte auftreten können, die in früheren Untersuchungen nicht beachtet wurden.

Mit Hilfe dieser Zusammenfassung soll deutlich gemacht werden, dass die Evaluierung aufgrund der komplexen Zusammenhänge zwischen den einzelnen Protokollbestandteilen und der Medienzugriffsverfahren anhand einer konkreten Implementierung notwendig ist. Verallgemeinerbare Aussagen sind aufgrund der vielen Einflussfaktoren kaum machbar.

5.5 Zusammenfassung

Ziel dieses Kapitels war die Evaluierung der Schlüsselaustauschverfahren Kerberos und ECDH-ECDSA unter Verwendung unterschiedlicher Medienzugriffsverfahren.

Es wurden folgende Evaluationsmetriken genutzt:

- Energiebedarf und die Dauer der kryptografischen Operationen
- Energiebedarf genau eines Schlüsselaustauschs
- Dauer eines Schlüsselaustauschs
- Energiebedarf über einen festen Messzeitraum

Anhand der Evaluationsergebnisse wurde gezeigt, dass der Energiebedarf der kryptografischen Operationen als Vergleichskriterium für Schlüsselaustauschprotokolle unzureichend ist. Die Parametrisierung der Medienzugriffsverfahren und die dort eingesetzte Duty-Cycling Strategie haben einen größeren Einfluss auf den Energiebedarf als die genutzten kryptografischen Operationen. Weiterhin konnte gezeigt werden, dass Seiteneffekte durch die Dauer der kryptografischen Operationen auf den Energiebedarf und die Dauer eines Schlüsselaustauschs existieren, die so bei einer rein analytischen Betrachtung nicht sofort einsichtig sind. Die Evaluierung komplexer Protokolle, wie beispielsweise Schlüsselaustauschprotokolle, sollte daher möglichst anhand einer konkreten Implementierung und mit möglichst realitätsnahen Werkzeugen wie AVRORA+ erfolgen.

6. Evaluierung des Energiebedarfs von hardwarebasierten Sicherheitsmechanismen

Im Rahmen des FleGSens-Projekts wurde AES als symmetrischer kryptografischer Algorithmus genutzt, um die Integrität und Authentizität der übertragenen Nachrichten mittels Message Authentication Codes (MACs) zu sichern. Es wurde untersucht, in wie weit der auf der Funkschnittstelle vorhandene kryptografische Co-Prozessor die kryptografischen Berechnungen beschleunigt. Es konnte gezeigt werden, dass durch den Einsatz des Co-Prozessors die Berechnungsdauer für AES deutlich geringer wurde. Eine Untersuchung von asymmetrischen Algorithmen war nicht möglich, da der Co-Prozessor lediglich AES unterstützte. Aus diesem Grund wurde in verschiedenen Arbeiten die Eignung von speziellen Hardwarebausteinen, wie beispielsweise Trusted Platform Modulen (TPM) [71] für drahtlose Sensornetze untersucht. Die Evaluierung des Energiebedarfs stand dabei häufig nicht im Fokus der Untersuchung. Wie im vorigen Kapitel deutlich wurde, weist die Berechnung von kryptografischen Algorithmen auf den relativ ressourcenschwachen Sensorknoten jedoch einen hohen Energiebedarf auf. Dies gilt insbesondere bei der Verwendung von asymmetrischer Kryptografie wie ECDSA oder RSA.

Ziel dieses Kapitels ist die Evaluierung des Energiebedarfs von hardwarebasierten Sicherheitsmechanismen anhand von Experimenten in SANDbed. Im Fokus der Untersuchung steht dabei die Frage, ob der Energiebedarf der

kryptografischen Berechnungen mit hardwarebasierten Sicherheitsmechanismen im Vergleich zu Softwareimplementierungen verringert werden kann. Das Kapitel gliedert sich wie folgt: Im ersten Teil des Kapitels werden die in verwandten Arbeiten genutzten hardwarebasierten Sicherheitsmechanismen vorgestellt. Im Anschluss wird der Mess- und Demoaufbau vorgestellt, mit deren Hilfe im weiteren Verlauf des Kapitels der Energiebedarf verschiedener kryptografischer Algorithmen und des Demoszenarios evaluiert wird.

Die Ergebnisse der Evaluierung des Energiebedarfs von hardwarebasierten Sicherheitsmechanismen sind im Rahmen zweier studentischer Arbeiten [72] [73] und zweier Veröffentlichung [12] [11] entstanden. Die Ergebnisse der Veröffentlichungen wurden für diese Arbeit erweitert.

6.1 Einsatz von hardwarebasierten Sicherheitsmechanismen in Sensornetzen

Ein grundlegendes Problem beim Einsatz von kryptografischen Algorithmen in drahtlosen Sensornetzen ist die Ressourcenbeschränktheit (Prozessor, Speicher) der Sensorknoten. Dies führt zu relativ langen Berechnungszeiten und somit einem erhöhten Energiebedarf. Dies gilt insbesondere für asymmetrische kryptografische Algorithmen, wie sie standardmäßig in vielen Internetprotokollen wie TLS oder IPSec verwendet werden. Im vorigen Kapitel wurde beispielsweise gezeigt, dass die Erzeugung und Verifikation einer Signatur mit ECDSA bis zu 2 Sekunden in Anspruch nimmt. Aus diesen Gründen wurde in verschiedenen Arbeiten die Eignung von speziellen Hardwarebausteinen, wie beispielsweise Trusted Platform Modulen (TPM) [71] für drahtlose Sensornetze untersucht.

In [74]und [75] beschreiben Hu et. al. die Sensorknotenplattform *secFleck*, die unter anderem ein TPM-Modul integriert hat. Ziel der Arbeit war es, RSA effizient und mit geringem Energie- und Zeitbedarf auf Sensorknoten einzusetzen. Hierfür wird das AT97SC3203S TPM-Modul der Firma Atmel in die Sensorknotenplattform Fleck [76] integriert. Ein TPM-Modul implementiert einen echten Zufallszahlengenerator, asymmetrische (RSA) sowie symmetrische kryptografische Algorithmen (XTEA). Somit ist es möglich, sowohl Ver- und Entschlüsselung sowie die Erzeugung und Verifikation von digitalen Signaturen durchzuführen.

Hu et. al. zeigen in ihrer Evaluierung des secFleck Sensorknotens, dass insbesondere der Einsatz von RSA mit dem secFleck einen deutlich geringeren Energiebedarf und Berechnungsaufwand im Vergleich zu einer Softwareimplementierung aufweist. Beispielsweise weist die Verschlüsselung von 1 Bit mit dem TPM-Modul einen Energiebedarf von $5,4\mu J$ auf, die im Vergleich

genutzte Softwareimplementierung $7030\mu J$. Ein offenes Problem ist die hohe Stromstärke des TPM-Moduls von $51mA$. Dies ist beispielsweise deutlich mehr als die Funkschnittstelle des MICAz Sensorknotens (etwa $20mA$). Dies bedeutet, dass ein dauerhaft eingeschaltetes TPM-Modul einen im Vergleich zu den anderen Hardwarekomponenten deutlich erhöhten Energiebedarf aufweist. Hu et. al. schlagen deshalb vor, das TPM-Modul zu deaktivieren, sofern es nicht benötigt wird. Innerhalb der Evaluierung wird nicht untersucht, in wie weit das Duty-Cycling des TPM-Moduls seiteneffektfrei und ohne zusätzlich Kosten ist. In [77] wird ein Ansatz zur Absicherung von DTLS mit Hilfe eines TPM-Chips vorgestellt. Neben der Ausführungszeit wird auch der Energiebedarf der einzelnen Operationen des TPM mit Hilfe eines Oszilloskops vermessen. Allerdings wird der Einfluss der Funkschnittstelle so wie ein Duty-Cycling des TPM nicht betrachtet.

Im Rahmen dieses Kapitels soll daher der Energiebedarf eines hardwarebasierten Sicherheitsmoduls bei Verwendung eines Duty-Cycling Mechanismus untersucht werden. Es wird gezeigt, dass der Overhead durch das Duty-Cycling in vielen Szenarien den Energiebedarf des Hardwaremoduls derart erhöht, dass eine vergleichbare Softwareimplementierung einen deutlich geringeren Energiebedarf aufweist.

Die Evaluierung des Energiebedarfs erfolgt bei Hu et. al. indirekt über die Messung der Ausführungszeiten der einzelnen kryptografischen Algorithmen anhand eines Prototypen. Eine Messung des Gesamtenergiebedarfs mit Hilfe eines Oszilloskops oder innerhalb eines Testbeds findet nicht statt. Wie in den vorigen Kapiteln deutlich wurde, können bei dieser Form der Evaluierung des Energiebedarfs Seiteneffekte durch das Betriebssystem oder durch Timing-Effekte nicht abgebildet werden.

Im Rahmen dieses Kapitels wird der Energiebedarf eines hardwarebasierten Sicherheitsmoduls innerhalb eines einfachen Anwendungsszenarios evaluiert. Zur Evaluierung werden Energiebedarfsmessungen in SANDbed und mit Hilfe der SNMDs durchgeführt. Somit werden Seiteneffekte durch das Betriebssystem oder sonstige Kosten (beispielsweise der Kommunikation des Sensorknotens mit dem Hardwaremodul über den I^2C-Bus) einbezogen.

Das im Rahmen dieses Kapitels eingesetzte Hardwaremodul (VaultIC420) bietet neben den auf dem TPM-Modul vorhandenen kryptografischen Algorithmen (RSA, XTEA, SHA-1) noch weitere Algorithmen. Insbesondere ist es möglich, AES als symmetrisches Verfahren und asymmetrische Verfahren basierend auf elliptischen Kurven (beispielsweise ECDSA) zu nutzen. Im Rahmen dieses Kapitels wird der Energiebedarf ausgesuchter kryptografischer Algorithmen sowohl mit dem Hardwaremodul als auch mit vergleichbaren Softwareimplementierungen evaluiert. Es zeigt sich, dass nicht für alle Algorith-

Algorithmus 4 : Pseudocode der Demonstrator Anwendung

```
1  wenn PIR-Event detektiert dann
2     wenn Sensorknoten A dann
3        Signiere PIR-Event mit VaultIC420;
4        Sende PIR-Event und Signatur an Sensorknoten C;
5     Ende
6     wenn Sensorknoten B dann
7        Signiere PIR-Event mit TinyECC;
8        Sende PIR-Event und Signatur an Sensorknoten C;
9     Ende
10 Ende
```

men der Energiebedarf des Hardwaremoduls geringer als der Energiebedarf der Softwareimplementierung ist. Insbesondere bei den symmetrischen Algorithmen ist es häufig energie-effizienter, die Softwareimplementierung zu nutzen. Neben dem Energiebedarf wird auch der Speicherbedarf des Hardwaremoduls analysiert. Es zeigt sich, dass der Einsatz des Hardwaremoduls wesentlich speicher-effizienter ist. So lässt sich beispielsweise der Bedarf an RAM von 5917 Byte mit den Softwareimplementierungen auf 2177 Byte mit dem Hardwaremodul verringern.

6.2 Beschreibung des FleGSens-Demonstrators

Da für das hier verwendete Sicherheitsmodul kein Energie- oder Ausführungsmodell für einen Simulator wie beispielsweise AVRORA+ existiert, war eine Implementierung auf echten Sensorknoten notwendig. Die im Rahmen dieses Kapitels genutzt Anwendung *TestVault* basiert im wesentlichen auf dem Demonstrator aus [11]. Der Demonstrator basiert auf dem FleGSens-Szenario [40] [41], bei dem drahtlose Sensorknoten zur Überwachung einer grünen Grenze eingesetzt werden. Zur Erkennung von unbefugten Übertritten sind die eingesetzten Sensorknoten mit Passiv-Infrarot (PIR) Sensoren ausgestattet, mit deren Hilfe ein Vorbeigehen am Sensorknoten in bis zu 10 m Entfernung detektiert werden kann. Ein so erkannter unbefugter Übertritt (PIR-Event) wird von den Sensorknoten anschließend an eine Basisstation übertragen und dort angezeigt.

Der Aufbau des Demonstrators ist in Abbildung 6.1 zu abgebildet. Innerhalb des Demonstrators werden zwei IRIS-Sensorknoten (A und B) zur Überwachung der grünen Grenze eingesetzt. Der Einsatz von IRIS-Sensorknoten war dabei notwendig, da der auf den MICAz vorhandene Speicher (4000 Byte RAM) nicht für die Implementierung von TestVault und den kryptografischen Algorithmen ausreichte.

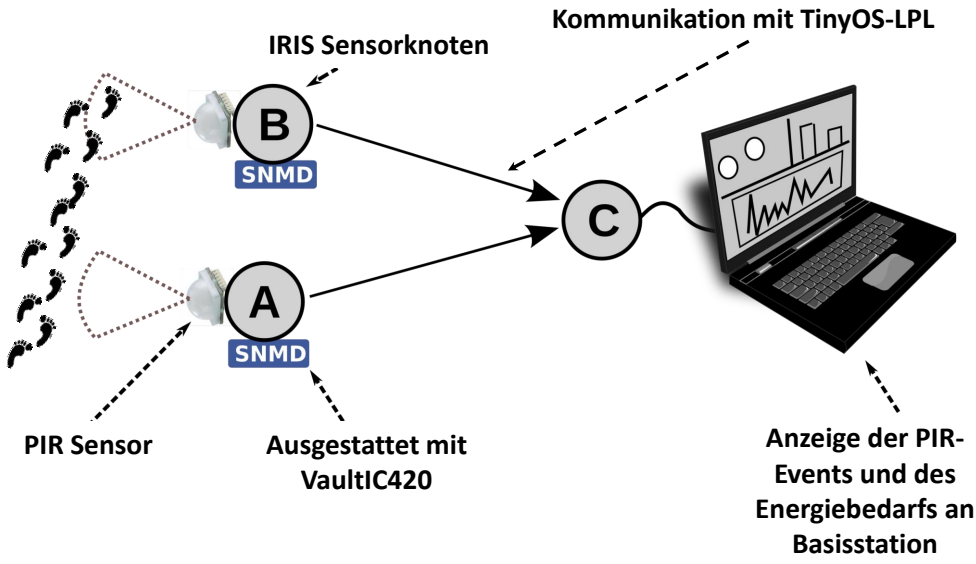


Abbildung 6.1 Aufbau des Demonstrators.

Der Pseudocode der Demonstrator-Anwendung ist in Algorithmus 4 abgebildet. Die Sensorknoten A und B sind mit PIR-Sensoren ausgestattet. Die detektierten PIR-Events werden an Sensorknoten C übertragen. Sensorknoten C leitet die PIR-Events nach dem Empfang an die Basisstation weiter. Zur Absicherung gegen unbefugte Manipulation der PIR-Events während der Übertragung wird jedes PIR-Event vor der Übertragung mit einer digitalen Signatur mit Hilfe des ECDSA Algorithmus versehen. Auf Sensorknoten A kommt zur Erzeugung der Signatur ein hardwarebasierter Sicherheitsbaustein zum Einsatz, Sensorknoten B nutzt eine Softwareimplementierung. Als hardwarebasierter Sicherheitsbaustein wird der VaultIC420 [78] der Firma Inside Secure eingesetzt, als Softwareimplementierung wird TinyECC [69] genutzt. Das notwendige Schlüsselmaterial ist vorab auf Sensorknoten A und B als auch auf der Basisstaion vorverteilt. Die Basisstation kann somit die Signaturen der PIR-Events beider Sensorknoten überprüfen. Die Anwendung ist in TinyOS geschrieben, als Medienzugriffsverfahren wird TinyOS-LPL eingesetzt.

Der VaultIC420 ist über das Two-Wire Serial Interface (auch I^2C genannt) über ein Erweiterungsboard an den Sensorknoten angeschlossen. Das Two-Wire Serial Interface bietet eine Datenübertragung mit bis zu 400 kbit/s [79]. Abbildung 6.2 zeigt das im Demonstrator eingesetzte Sensorboard sowie den VaultIC420 und den PIR-Sensor.

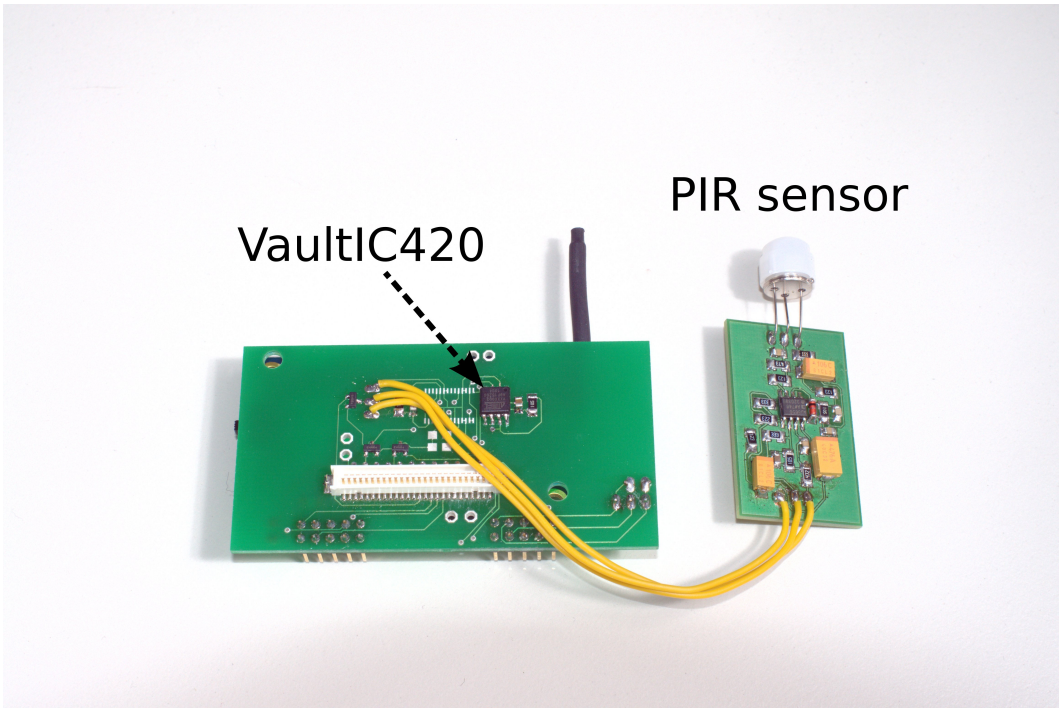


Abbildung 6.2 Sensorboard mit PIR-Sensor und VaultIC420.

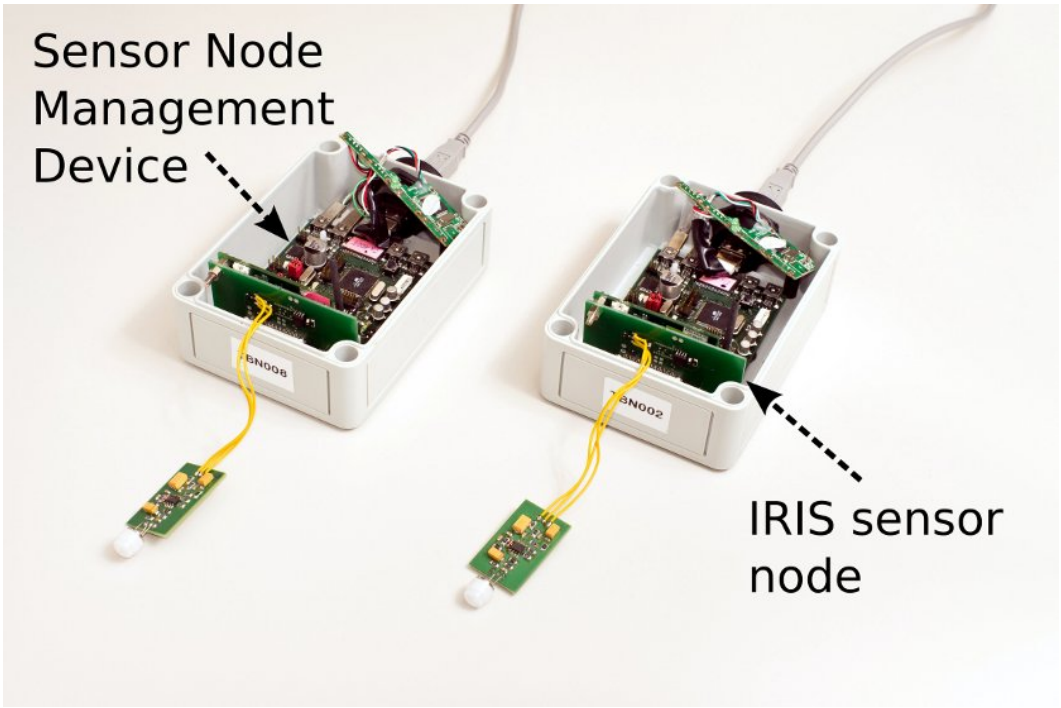


Abbildung 6.3 Die im Demonstrator genutzten SNMDs mit angeschlossenen IRIS-Sensorknoten und PIR-Sensoren.

Zur Messung des Energiebedarfs sind die IRIS-Sensorknoten an zwei SNMDs angeschlossen. Abbildung 6.3 zeigt die SNMDs, sowie die IRIS-Sensorknoten inklusive des angeschlossenen Erweiterungsboards mit VaultIC420 und PIR-Sensor. Die SNMDs sind per USB an die Basisstation angeschlossen. Somit kann der Energiebedarf bei Ausführung der kryptografischen Operationen oder der Kommunikation angezeigt werden. Zur Messung und Analyse des Energiebedarfs von einzelnen Anwendungsteilen bieten die SNMDs die Möglichkeit, sogenannte *Marker* in den Code einzufügen. Mit Makern kann beispielsweise der Beginn und das Ende einer kryptografischen Berechnung im Anwendungscode gekennzeichnet werden. Während der Energiebedarfsmessung kann so mit den SNMDs der Energiebedarf zwischen diesen Markierungen bestimmt und analysiert werden. Die Nutzung der Marker ist nicht komplett frei von Kosten. Für jeden Marker innerhalb des Anwendungscodes fällt ein Overhead von etwa 1,2 μJ an.

Innerhalb des Demonstrators werden die Marker eingesetzt, um zum Einen den Energiebedarf der Softwareimplementierung von ECDSA mit dem Energiebedarf des hardwarebasierten Sicherheitsbausteins zu vergleichen, als auch um den Overhead für die Kommunikation mit TinyOS-LPL zu analysieren.

Ein Beispiel für die Anzeige der PIR-Events und des Vergleichs des Energiebedarfs ist in Abbildung 6.4 zu sehen. Die Anzeige ist in 4 Bereiche aufgeteilt. Die Anzeige der von der Basisstation empfangenen PIR-Events ist oben links im Fenster *FleGSens View* zu sehen. In diesem Beispiel hat die Basisstation gerade einen PIR-Event des Sensorknoten B erhalten und zeigt diesen an.

Neben der Anzeige der beiden Sensorknoten und der empfangenen PIR-Events ist im Fenster *Last 10 Measurements* der Energiebedarf der letzten 10 Energiebedarfsmessungen mit Hilfe von Balkendiagrammen angezeigt. Der Energiebedarf des Sensorknotens B ist dabei in orange, der Energiebedarf des Sensorknotens A in grün dargestellt. Der angezeigte Energiebedarf enthält dabei sowohl die Kosten für die kryptografische Berechnung (unterer Teil des Balkens) als auch des Nachrichtenversands (oberer Teil des Balkens), dargestellt durch unterschiedliche Farbschattierungen.

Ebenfalls dargestellt ist der durchschnittliche Energiebedarf aller bisher gemessenen PIR-Events im Fenster *Average*. Unterhalb der Balkendiagramme sind zudem der Energiebedarf und die Dauer der Verarbeitung und des Versands des letzten PIR-Events und der Durchschnitt aller PIR-Events in Tabellenform dargestellt.

Zum direkten Vergleich der Sensorknoten ist die Stromstärke der beiden Sensorknoten innerhalb der Livemessung im unteren Bereich der Anzeige abgebildet. Auf der x-Achse ist die Laufzeit in Sekunden, auf der y-Achse die

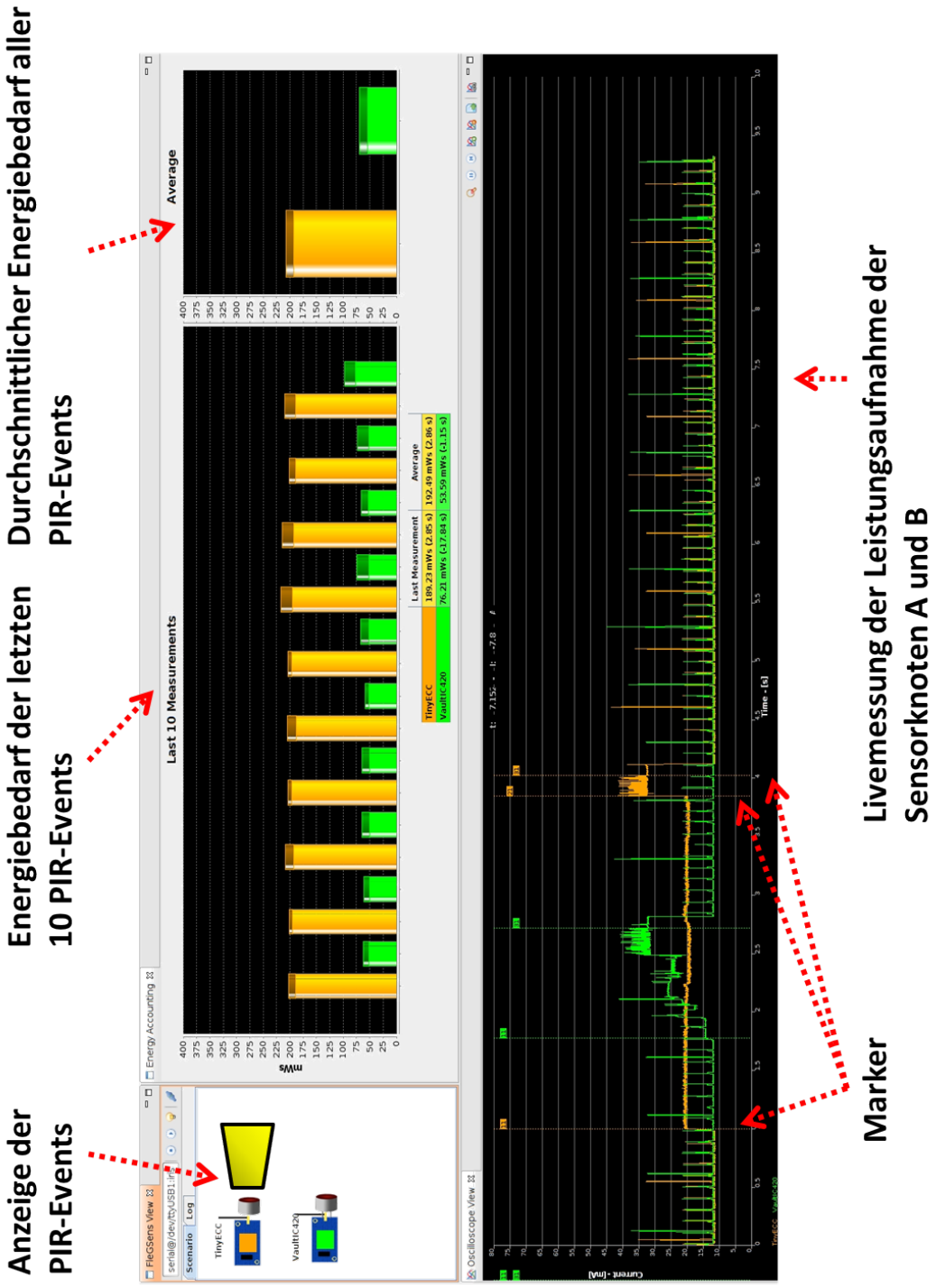


Abbildung 6.4 Anzeige der PIR-Events und des Vergleichs des Energiebedarfs wie innerhalb des Demonstrators an der Basisstation gezeigt.

Algorithmus 5 : Pseudocode der Anwendung *TestVault*

```
1  für Beide Testvektoren tue
2     für Hardware- und Softwareimplementierung tue
3        wenn Algorithmus == AES dann
4           Verschlüssle Testvektor ;
5           Entschlüssle Testvektor ;
6        Ende
7        wenn Algorithmus == HMAC-SHA1 dann
8           Berechne Message Digest ;
9           Verifiziere Message Digest ;
10       Ende
11       wenn Algorithmus == ECDSA dann
12          Berechne Signatur ;
13          Verifiziere Signatur ;
14       Ende
15    Ende
16 Ende
```

Stromstärke in mA abgebildet. Neben der Stromstärke sind zudem die Marker zu erkennen, die für den Vergleich des Energiebedarfs eingesetzt werden. Jeder Marker ist hierbei durch eine vertikale Linie in der jeweiligen Farbe des Sensorknotens eingezeichnet. Der in den Balkendiagrammen dargestellte Energiebedarf wird mit Hilfe der Marker berechnet. Hierfür wird der erste Marker gesetzt, sobald das PIR-Event ausgelöst wurde. Der zweite Marker wird nach der Berechnung der Signatur eingezeichnet. Der dritte Marker wird gesetzt, sobald die Nachricht erfolgreich an Sensorknoten C übertragen wurde und die dazugehörige Bestätigung empfangen wurde. Innerhalb der Anzeige sind beispielhaft die drei Marker des Sensorknotens B (orange) markiert.

Die Marker kennzeichnen somit die Dauer der jeweiligen Aktion. Mit Hilfe der Dauer t, der Stromstärke I und der Spannung U (nicht in der Anzeige dargestellt) der Sensorknoten lässt sich somit der Energiebedarf P der einzelnen Aktionen berechnen.

6.3 Evaluierung des Energiebedarfs

Der Energiebedarf des hardwarebasierten Sicherheitsbausteins soll mit Hilfe der Testanwendung *TestVault* evaluiert werden. Der Pseudocode der Anwendung TestVault ist in Algorithmus 5 abgebildet.

Tabelle 6.1 fasst die im Rahmen dieses Kapitels verwendeten Experimentparameter zusammen. TestVault wurde für das Betriebssystem TinyOS 2.1.2 und die Sensorknotenplattform IRIS erstellt. Für die Energiemessung wurden zwei SNMDs des SANDbed eingesetzt. Die Messrate betrug für alle Messun-

Parameter	Wert
Knotenplattform	IRIS
Betriebssystem	TinyOS 2.1.2
Messrate der SNMDs	10kHz
Gemessene Parameter	Stromstärke I und Spannung U
Untersuchte Algorithmen	AES, ECDSA, HMAC-SHA1
Untersuchte kryptografische Operationen	Ver- und Entschlüsselung, Signatur Erzeugung und Verifikation, Hashwertberechnung
Hardwarebaustein	VaultIC420
Softwareimplementierungen	ECDSA und SHA/HMAC-SHA aus TinyECC , AES aus TinyOS-contrib
Verwendete Testvektoren	16 Byte und 96 Byte
Experimentwiederholungen	20
Medienzugriffsverfahren	802.15.4 im Non-Beacon Mode, TinyOS-LPL
Kanal	25
Sendeleistung	31
Aufwachintervall t_{LPL}	1000 ms
CCA Länge t_{CCA}	2400

Tabelle 6.1 Experimentparamter der Evaluierung des Energiebedarfs der hardwarebasierten Sicherheitsbausteine.

gen 10 kHz, es wurde jeweils die Stromstärke I und die Spannung U gemessen. Die Experimente wurden für jede Parametrisierung 20-mal wiederholt. Nach jeweils 10 Experimentwiederholungen wurde die Rolle der SNMDs getauscht und der VaultIC420 umgesteckt. Hierdurch sollen Messabweichung aufgrund der Hardwarevarianzen ausgeglichen werden.

Als Medienzugriffsverfahren wurden IEEE802.15.4 im Non-Beacon-Modus und TinyOS-LPL eingesetzt. Hiermit sollte der Zusammenhang zwischen Duty-Cycling der Funkschnittstelle und Duty-Cycling des Hardwarebausteins gezeigt werden. Die Sendestärke der Funkschnittstelle wurde für beide Medienzugriffsverfahren auf die maximal mögliche Sendestärke in TinyOS (31) eingestellt. Als Funkkanal wurde Kanal 25 genutzt. Bei der Verwendung von TinyOS-LPL wurde, wie bereits in den vorigen Kapiteln begründet, eine CCA Länge von 12 ms genutzt. Das Aufwachintervall wurde mit $t_{LPL} = 1000ms$ festgelegt.

Für den Vergleich des Energiebedarfs wurden die Algorithmen AES, ECDSA und HMAC-SHA1 ausgewählt. Als kryptografische Operationen kamen dabei die Ver- und Entschlüsselung mit AES, die Signatur Erzeugung und Verifikation mit ECDSA und die Erzeugung und Verifikation von Message Digests mit HMAC-SHA1 zum Einsatz. Die restlichen vom VaultIC420 unterstützen Algorithmen (beispielsweise RSA oder 3DES)[13] wurden im Rahmen dieser Evaluierung nicht betrachtet, da die Algorithmen im allgemeinen als ungeeignet für drahtlose Sensornetze gelten [80] [81]. So sind für RSA deutlich größere Schlüssellängen notwendig um ein ähnliches Sicherheitsniveau wie bei ECDSA zu erreichen, was gleichzeitig einen erhöhten Speicherbedarf und Kommunikationsaufwand erfordert. Für die kryptografischen Algorithmen kommen die Softwareimplementierungen aus TinyECC (ECDSA und HMAC-SHA1), sowie die AES Implementierung aus dem TinyOS-contrib Verzeichnis zum Einsatz. Diese Implementierungen sind zwar im Bezug auf Effizienz und Berechnungsdauer nicht die optimiertesten Verfahren, werden jedoch in anderen Arbeiten häufig eingesetzt und sind im Gegensatz zu anderen Implementierungen öffentlich verfügbar.

Um den Einfluss von unterschiedlichen Eingabegrößen auf den Energiebedarf zu testen, wurden für alle untersuchten Algorithmen zwei *Testvektoren* mit unterschiedlicher Länge (16 Byte und 96 Byte) verwendet. Die Länge der Testvektoren wurde dabei so gewählt, dass sie zum Einen ein Vielfaches der

[13] Die in dieser Arbeit gemessenen Werte für ECDSA unterscheiden sich zwar quantitativ von den Messergebnissen für RSA und DSA, die generellen, qualitativen Aussagen über den Energiebedarf und die Berechnugszeiten sind jedoch vergleichbar.

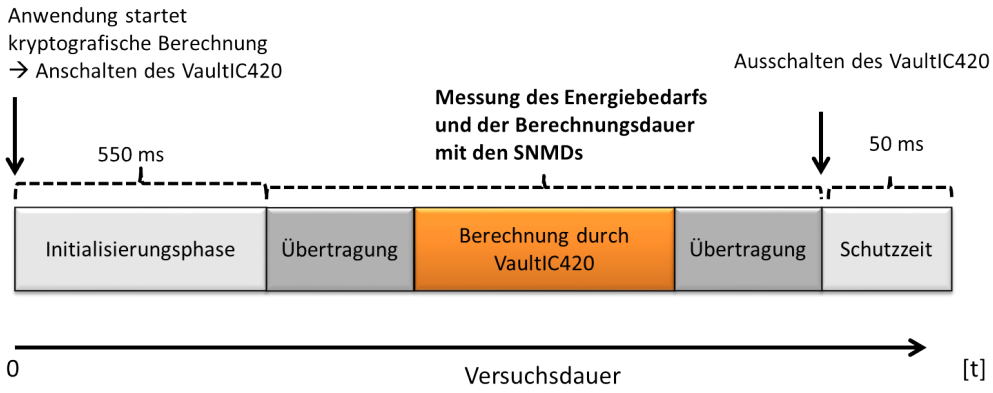


Abbildung 6.5 Ablauf einer Messung der Anwendung TestVault

in AES verwendeten Blockgröße (16 Byte) waren[14] und zum Anderen innerhalb einer Nachrichteneinheit mittels der EEMAC Programmierschnittstelle versendet werden konnten ($\leq$ 100 Byte).

6.3.1 Evaluationsmetriken

Der Ablauf einer Messung bei Verwendung des VaultIC420 ist in Abbildung 6.5 abgebildet. Sobald die Anwendung TestVault eine kryptografische Berechnung durchführen will, wird der VaultIC420 angeschaltet. Nachdem der VaultIC420 angeschaltet wird, benötigt der Hardwarebaustein eine *Initialisierungsphase*. Innerhalb der Initialisierungsphase lädt der VaultIC420 das benötigte Schlüsselmaterial und führt einen Selbsttest durch. Die Initialisierungsphase ist bei jedem Anschaltevorgang des VaultIC420 durchzuführen. Der Energiebedarf der Initialisierungsphase hat einen entscheidenden Einfluss auf den Energiebedarf des VaultIC420 im Vergleich zu einer reinen Softwareimplementierung ohne Initialisierungsphase.

Bei Voruntersuchungen hat sich gezeigt, dass für die korrekte und fehlerfreie Durchführung der Initialisierungsphase eine Dauer von 550 ms haben sollte, bevor der VaultIC420 genutzt werden kann [15].

Nach der Initialisierungsphase beginnt die eigentliche Berechnung. Hierfür sind in einem ersten Schritt die zu verarbeitenden Daten über den I^2C-Bus an den VaultIC420 zu übertragen. Anschließend berechnet der VaultIC420 die eigentliche kryptografische Operation, das Ergebnis wird anschließend wieder zurück an die Anwendung übertragen.

[14] Bei der Verwendung von AES muss die Eingabge bzw. die Ausgabe immer ein Vielfaches von 16 Byte sein. Ist dies nicht der Fall, müssten mittels *Padding* Bits hinzugefügt werden. Um dies zu vermeiden wurde die Länge der Testvektoren als Vielfaches der Blockgröße gewählt.

[15] Eine kürzere Dauer führte in einigen Experimenten zu einer fehlerhaften Ausführung der Testanwendung. Trotz des großen Einflusses der Initialisierungszeit auf den Energiebedarf des VaultIC420 wurde daher darauf verzichtet, den Wert zu verringern.

Falls die Anwendung den Hardwarebaustein nicht mehr für weitere Berechnungen benötigt, wird der VaultIC420 ausgeschaltet. Der VaultIC420 sichert hierbei den internen Zustand der Hardware. Um den Baustein danach wieder verwenden zu können ist hierbei eine Schutzzeit von 50 ms einzuhalten.

Die Dauer der Messung des Energiebedarfs und der Berechnungsdauer mit Hilfe der SNMDs ist ebenfalls in der Abbildung eingezeichnet. Neben der eigentlichen Berechnung des VaultIC420 wird weiterhin auch die Nachrichtenübertragung über den I^2C-Bus Übertragung mitgemessen, da die Übertragungskosten und die Übertragungsdauer Teil der Nutzung des VaultIC420 sind.

Im Vergleich zu der Nutzung des VaultIC420 wurde bei der Verwendung der Softwareimplementierung nur die eigentliche Berechnung (orange) vermessen, da die Software weder eine Initialisierungsphase, noch eine Schutzzeit benötigt. Auch eine Übertragung der Daten über einen zusätzlichen Bus ist nicht notwendig.

Bei der Evaluierung des Energiebedarfs der hardwarebasierten Sicherheitsmechanismen werden verschiedene Metriken betrachtet. Zusammengefasst sind dies:

- **Dauer der kryptografischen Berechnung**: Die Berechnungsdauer der unterschiedlichen kryptografischen Algorithmen wird für beide Testvektoren gemessen.
- **Energiebedarf der kryptografischen Berechnung**: Der Energiebedarf der unterschiedlichen kryptografischen Algorithmen wird für beide Testvektoren gemessen. Bei der Nutzung des VaultIC420 schließt dies die Übertragung der Daten auf dem I^2C mit ein.
- **Speicherbedarf:** Der Speicherbedarf für die Implementierung der kryptografischen Algorithmen bzw. der Treiber des VaultIC420 wird ermittelt.
- **Einfluss des Duty-Cyclings auf den Energiebedarf:** Da für die Nutzung des VaultIC420 eine Initialisierungsphase und eine Schutzzeit notwendig war, wurde der dadurch verursachte Overhead gemessen. Hierfür wurde die Energiebedarfsmessung beim Starten des VaultIC420 gestartet, die Messung endet nach der Schutzzeit. Als Anwendung hierfür wurde die Demonstrator-Anwendung genutzt.

6.3.2 Speicherbedarf

Vor der eigentlichen Evaluierung des Energiebedarfs wird in diesem Abschnitt der Speicherbedarf der Testanwendung analysiert. Tabelle 6.2 fasst den Speicher-

	VaultIC420	Software
ROM	16970 Byte	43426 Byte
RAM	2177 Byte	5917 Byte

Tabelle 6.2 Speicherbedarf der Testanwendung sowohl mit VaultIC420 als auch mit den Softwareimplementierungen.

bedarf der Testanwendung bei der Verwendung des VaultIC420 und der Softwareimplementierungen zusammen. Hierfür wurde die Testanwendung einmal unter Verwendung des VaultIC420 und einmal unter Verwendung der Softwareimplementierungen übersetzt.

Die hier dargestellten Werte für den Speicherbedarf stellen den maximalen (statischen) Speicherbedarf bei gleichzeitiger Einbindung aller Algorithmen (AES, HMAC-SHA1 und ECDSA) dar. Während der Ausführung muss zusätzlich der Stack und der Heap im RAM untergebracht werden. Bei Verwendung des VaultIC420 sind insgesamt 2177 Byte RAM und 16970 Byte ROM notwendig. Die Softwareimplementierungen benötigen hingegen deutlich mehr RAM und ROM, 5917 Byte bzw. 43426 Byte. Der IRIS-Sensorknoten verfügt insgesamt über 8000 Byte RAM und 128000 Byte ROM. Insbesondere beim Bedarf an RAM kann es, wenn die implementierte Anwendung komplexer als die Testanwendung ist, bei der reinen Softwareimplementierung je nach Heap oder Stack Bedarf zu Problemen mit zu wenig verfügbarem Speicher kommen. Insgesamt benötigt die Softwareimplementierung etwa 3-mal mehr Speicher als die Implementierung mit dem VaultIC420.

6.3.3 Evaluierung von AES

Der Energiebedarf bei der Verwendung von AES zur Ver- und Entschlüsselung ist in Abbildung 6.6 und 6.7 abgebildet. Auf der y-Achse ist der Energiebedarf in mJ bei Durchführung einer Ver- bzw. Entschlüsselung abgebildet. Bei den Messungen wurde sowohl beim VaultIC420 als auch bei der Softwareimplementierung ein 128 Bit Schlüssel verwendet.

Die Verschlüsselung von 96 Byte weist einen Energiebedarf von 2,8 mJ mit dem VaultIC420 und 0,50 mJ mit der Softwareimplementierung auf. Die Verschlüsselung von 16 Byte hat einen Energiebedarf von 1,27 mJ mit dem VaultIC420, 0,12 mJ in Software.

Im Vergleich zu der Softwareimplementierung ist der VaultIC420 folglich 10-mal so teuer bei der Verschlüsselung von 16 Byte, etwa 5-mal so teuer bei der

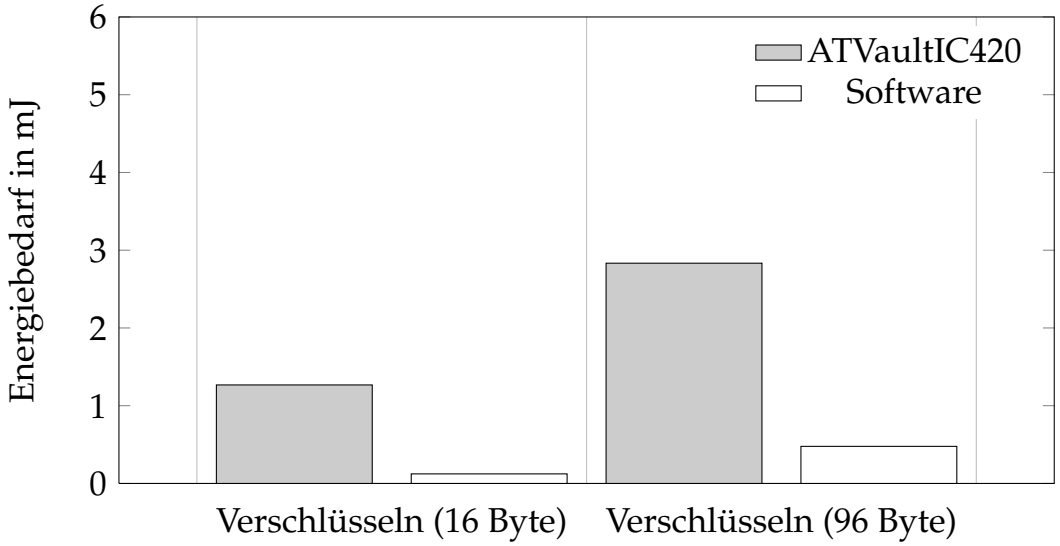


Abbildung 6.6 Energiebedarf der Verschlüsselung mit AES.

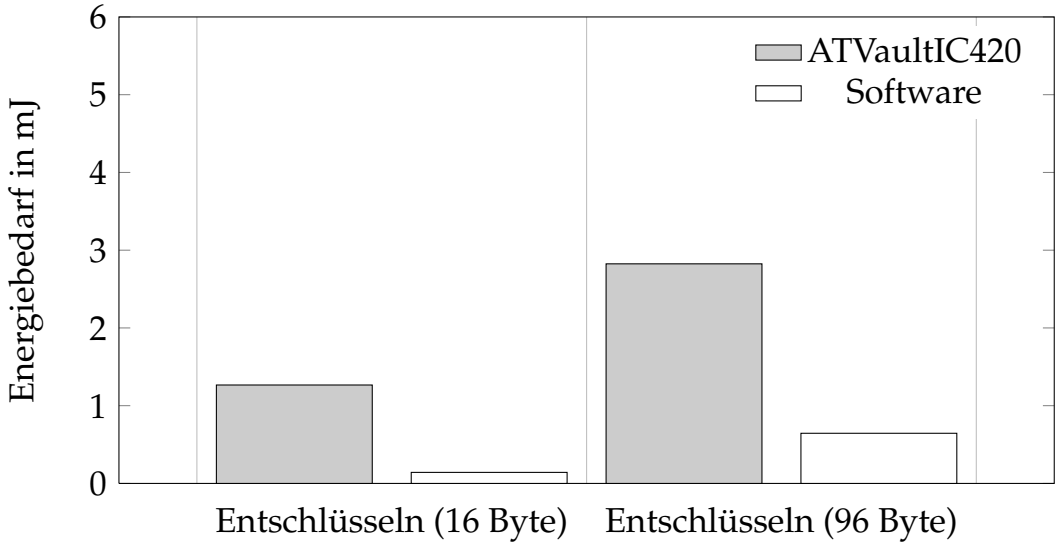


Abbildung 6.7 Energiebedarf der Entschlüsselung mit AES.

Funktion	Durchschnitt in s
Verschlüsseln (16 Byte)	0,01406
Verschlüsseln (96 Byte)	0,03181
Entschlüsseln (16 Byte)	0,01406
Entschlüsseln (96 Byte)	0,03196

Tabelle 6.3 Durchschnittliche Berechnungszeit von AES mit dem ATVaultIC420

Verschlüsselung des 96 Byte Testvektors. Ein ähnliches Verhältnis ergibt sich für die Entschlüsselung in Abbildung 6.7.

Dieser Unterschied lässt sich auf die deutlich längere Berechnungszeit auf dem VaultIC420 zurückzuführen. Die Berechnungszeiten für den VaultIC420 und die Softwareimplementierung sind in Tabelle 6.3 bzw. 6.4 abgebildet. Die Berechnungszeit des VaultIC420 beinhaltet hierbei auch die Dauer der Datenübertragung.

Beispielsweise benötigt AES zur Verschlüsselung des 96 Byte Testvektor in Software $0,00630$ Sekunden, der VaultIC $0,03181$ Sekunden. Hierbei entfallen beim VaultIC420 etwa 8 ms auf die eigentliche Kommunikation über den I^2C-Bus, zusätzlich hierzu kommen nochmals etwa 10 ms für die Verarbeitung der Daten im I^2C-Treiber von TinyOS[16].

Während der Übertragung und der Berechnung hat der VaultIC420 eine Leistungsausfnahme von $60mW$. Insgesamt ergibt sich somit, dass durch die längere Dauer bei der Ausführung auf dem VaultIC420 der Energiebedarf gegenüber der Softwareimplementierung deutlich höher ist.

Evaluierung von HMAC-SHA1

Der Energiebedarf des VaultIC420 bei der Berechnung und Verifizierung von HMAC-SHA1 Message Authentication Codes (MACs) ist in Abbildung 6.8 und 6.9 dargestellt.

Bei der Berechnung der MACs, Abbildung 6.8, ist zu erkennen, dass die Berechnung in Software für beide Testvektoren einen $1,2mJ$ höheren Energiebedarf aufweist als die Berechnung mit dem VaultIC420. Die Berechnung des

[16]Die Ergebnisse sind spezifisch für den VaultIC420, das Betriebssystem TinyOS und den I^2C-Bus Treiber. Beim Einsatz anderer Hardware oder Software muss insbesondere die Berechnungsdauer nochmals evaluiert werden. Unter Umständen könnte somit auch die Berechnung von AES mit Hilfe des Hardwarebausteins schneller und energie-effizienter durchgeführt werden.

Funktion	Durchschnitt in s
Verschlüsseln (16 Byte)	0,00163
Verschlüsseln (96 Byte)	0,00630
Entschlüsseln (16 Byte)	0,00192
Entschlüsseln (96 Byte)	0,00856

Tabelle 6.4 Durchschnittliche Berechnungszeit von AES in Software

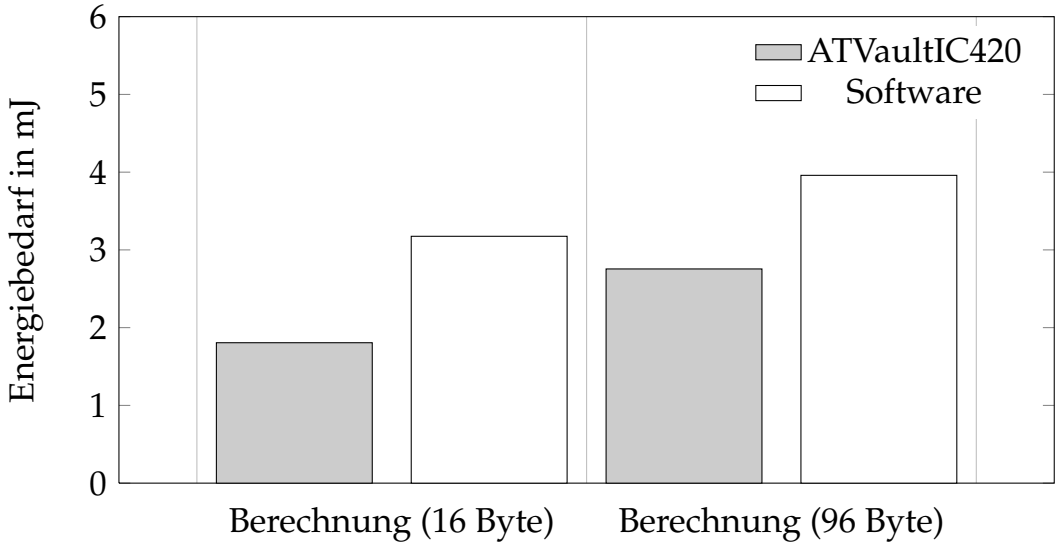


Abbildung 6.8 Energiebedarf bei der Berechnung eines Message Digests mit HMAC-SHA1.

MACs für den 96 Byte Testvektor ist sowohl in Software als auch in Hardware etwa $0,8mJ$ teurer als die Berechnung des MACs des kleineren Testvektors.

Die Verifikation der MACs, Abbildung 6.9, hat in etwa den gleichen Energiebedarf wie die Berechnung der MACs, die Abweichung des Energiebedarfs zwischen Berechnung und Verifikation beträgt unter 1%.

Die durchschnittlichen Berechnungszeiten für HMAC-SHA1 sind in den Tabellen 6.5 und 6.6 abgebildet. Die Verifikation eines MACs dauert für beide Testvektoren und sowohl mit dem VaultIC420 als auch in Software geringfügig länger als die Berechnung. Die Berechnung eines MACs mit dem VaultIC420 dauert $0,02079$ Sekunden bzw. $0,03112$ Sekunden. In Software ergibt sich eine Berechnungszeit von etwa $0,04294$ Sekunden bzw. $0,05357$ Sekunden. Somit ist der VaultIC420 bei der Berechnung eines MACs sowohl schnel-

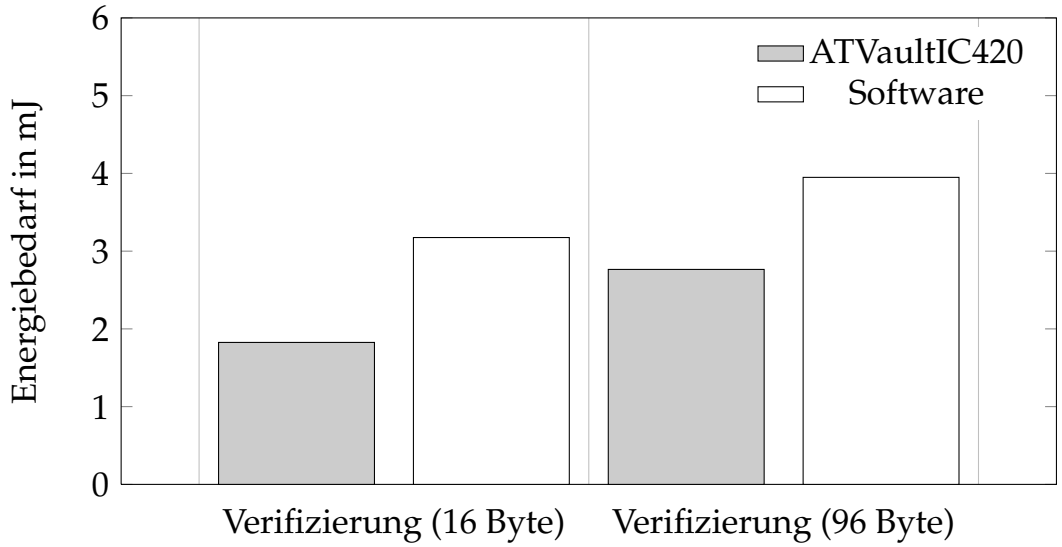


Abbildung 6.9 Energiebedarf bei der Verifizierung eines Message Digests mit HMAC-SHA1.

Funktion	Durchschnitt in s
Berechnung (16 Byte)	0,02079
Berechnung (96 Byte)	0,03112
Verifizierung (16 Byte)	0,02084
Verifizierung (96 Byte)	0,03099

Tabelle 6.5 Durchschnittliche Berechnungszeit von HMAC-SHA1 mit dem ATVaultIC420

Funktion	Durchschnitt in s
Berechnung (16 Byte)	0,04294
Berechnung (96 Byte)	0,05357
Verifizierung (16 Byte)	0,04301
Verifizierung (96 Byte)	0,05543

Tabelle 6.6 Durchschnittliche Berechnungszeit von HMAC-SHA1 in Software

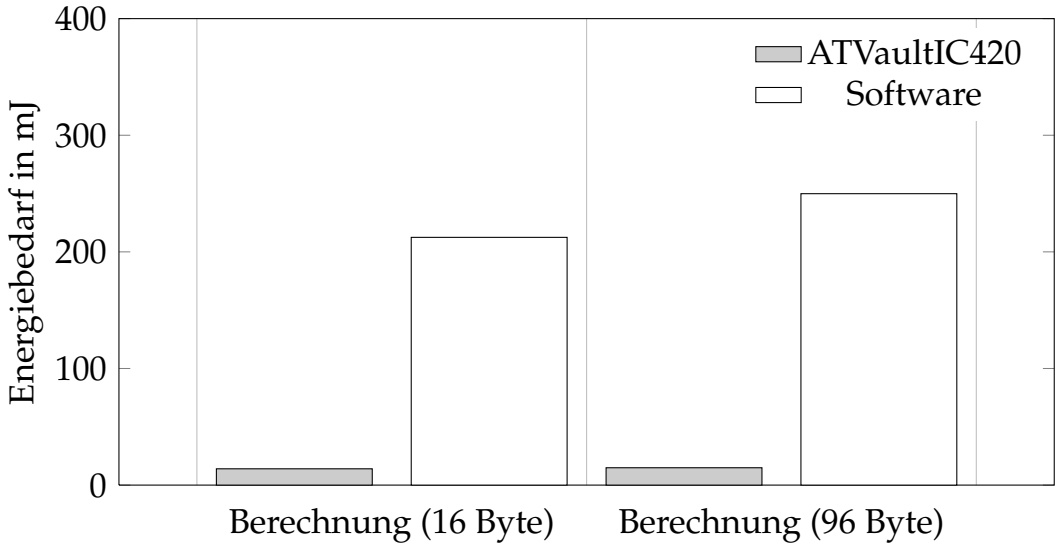


Abbildung 6.10 Energiebedarf der Berechnung einer Signatur mit ECDSA.

ler als auch energie-effizienter. Auch bei HMAC-SHA1 ergibt sich somit ein Zusammenhang zwischen Berechnungszeit und Energiebedarf.

Für die Verifikation der MACs ergibt sich ein ähnliches Verhältnis zwischen Berechnungsdauer und Energiebedarf.

6.3.4 Evaluierung von ECDSA

Der Energiebedarf der Berechnung und Verifikation von digitalen Signaturen mit ECDSA ist in Abbildung 6.10 und 6.11 dargestellt. Bei den Energiebedarfsmessungen zu ECDSA kam ein Schlüsselpaar mit 192-Bit zum Einsatz, es wurde jeweils eine Signatur des 16 Byte Testvektors sowie eine Signatur des 96 Byte Testvektors berechnet und verifiziert. Auf der x-Achse ist jeweils der verwendete Testvektor, auf der y-Achse der Energiebedarf der Operation in mJ dargestellt.

Die Berechnung einer ECDSA Signatur, Abbildung 6.10, mit dem VaultIC420 hat einen wesentlich geringeren Energiebedarf als die Berechnung in Software. Für den 16 Byte Testvektor hat die Softwareimplementierung einen etwa 15-mal höheren Energiebedarf (212,44 mJ gegenüber 13,98 mJ) als die Berechnung mit dem VaultIC420. Bei dem 96 Byte Testvektor zeigt sich ein etwa 17-mal größerer Energiebedarf der Softwareimplementierung (249,88 mJ gegenüber 14,88 mJ).

Die Berechnung der Signatur des 96 Byte Testvektors zeigt einen leicht erhöhten Energiebedarf im Vergleich zu dem kleineren Testvektor. Der VaultIC420 zeigt einen um 37 mJ erhöhten Energiebedarf (etwa 17%), die Softwareimplementierung einen 0,9 mJ erhöhten Energiebedarf (6, 5%).

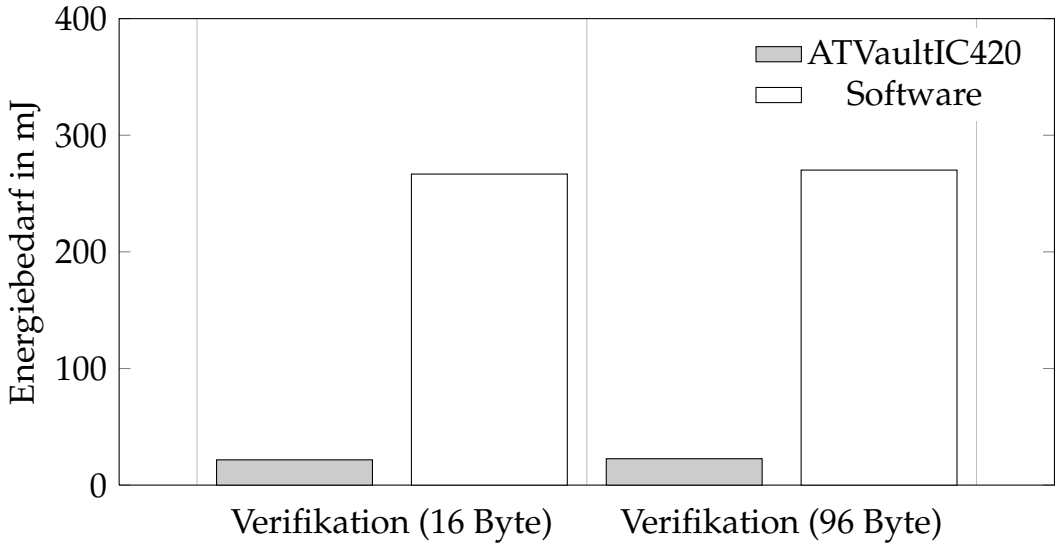


Abbildung 6.11 Energiebedarf der Verifikiation einer Signatur mit ECDSA.

Funktion	Durchschnitt in s
Berechnung (16 Byte)	0,16227
Berechnung (96 Byte)	0,1734
Verifikation (16 Byte)	0,25085
Verifikation (96 Byte)	0,26144

Tabelle 6.7 Durchschnittliche Berechnungszeit von ECDSA mit dem VaultIC420

Die Verifikation der Signaturen ist in allen hier durchgeführten Messungen teurer als die Berechnung. Insgesamt hat die Verifikation der Signatur des 16 Byte Vektors einen Energiebedarf von 21,66 mJ mit dem VaultIC420 und 266,72 mJ in Software. Die Verfikation des 96 Byte Testvektors hat einen Energiebedarf von 22,64 mJ in Software und 270,11 mJ mit dem VaultIC420. Die Softwareimplementierung hat somit für beide Testvektoren einen etwa 12-mal höheren Energiebedarf.

Die Berechnungs- bzw. Verifikationsdauer in Software (Tabelle 6.8) unterscheidet sich erheblich von den gemessenen Werten mit dem VaultIC420 (Tabelle 6.7).

Während die Softwareimplementierung eine Berechungsdauer von etwa $2,5$ Sekunden und eine Verifikationsdauer von etwa $3,6$ Sekunden aufweist, benötigt der VaultIC420 nur etwa $0,17$ Sekunden bzw. $0,26$ Sekunden. Der hö-

Funktion	Durchschnitt in s
Berechnung (16 Byte)	2,86808
Berechnung (96 Byte)	2,86808
Verifikation (16 Byte)	3,59252
Verifikation (96 Byte)	3,63912

Tabelle 6.8 Durchschnittliche Berechnungszeit und Energiebedarf von ECDSA in Software

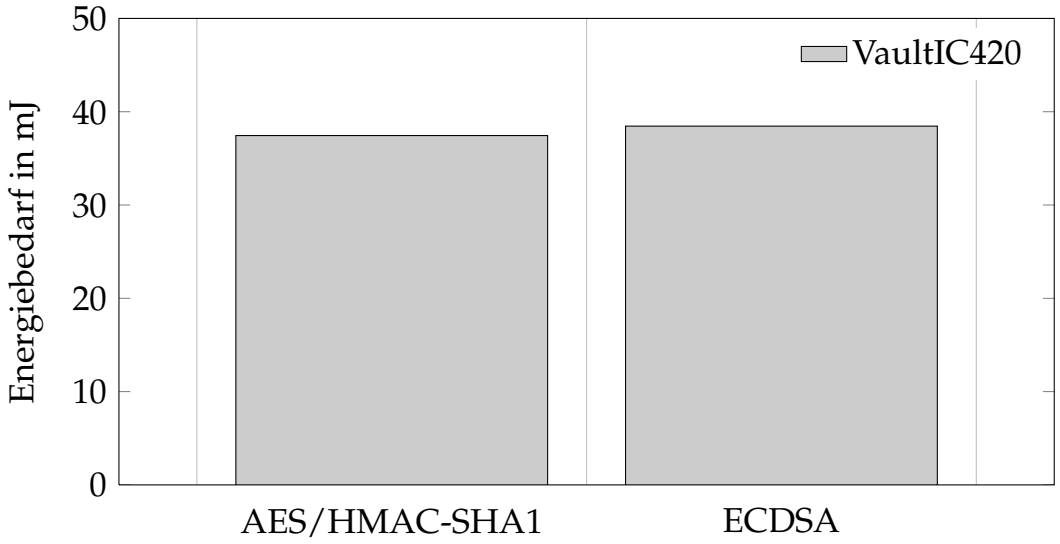


Abbildung 6.12 Energiebedarf der Initialisierungsphase.

here Energiebedarf der Softwareimplementierung ist somit auf die Berechnungsdauer zurückzuführen.

6.3.5 Initialisierungsphase

Der Energiebedarf der Initialisierungsphase ist in Abbildung 6.12 abgebildet. Die Initialisierungsphase für AES und HMAC-SHA1 weist einen Energiebedarf von 37,44 mJ, die Initialsierungsphase von ECDSA mit 38,46 mJ einen leicht höheren Energiebedarf auf. Die Dauer der Initialisierungsphase beträgt für AES und HMAC-SHA1 0,57685 s, für ECDSA 0,58696 s.

6.3.6 Vergleich HMAC-SHA1 und ECDSA mit Duty-Cycling des VaultIC420

Nachdem in den vorigen Abschnitten der Energiebedarf der reinen kryptografischen Berechnungen der Algorithmen AES, HMAC-SHA1 und ECDSA

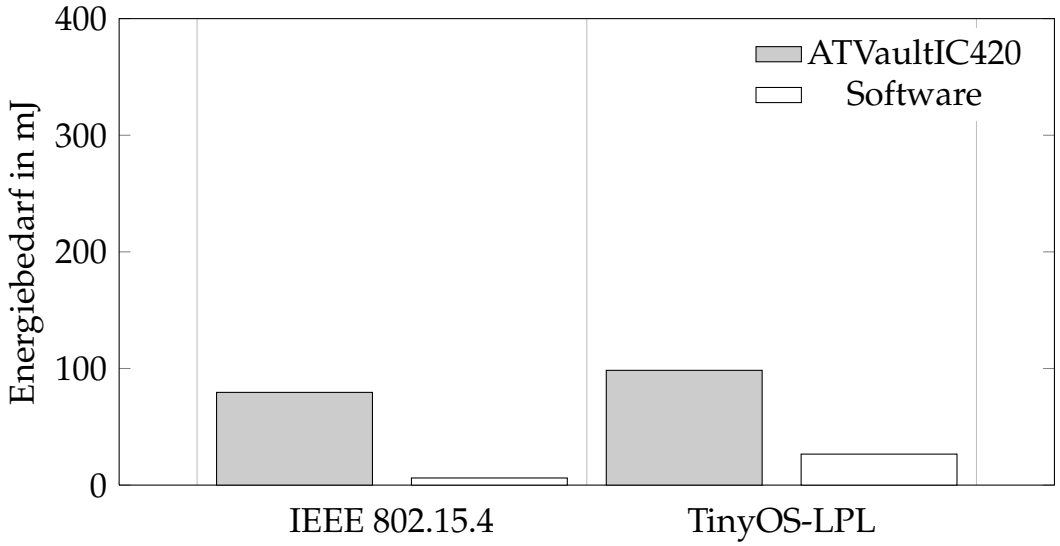


Abbildung 6.13 Vergleich des Energiebedarfs von HMAC-SHA1 im FleG-Sens-Demonstrator.

vorgestellt wurde, wird in diesem Abschnitt ein Vergleich des Energiebedarfs der Algorithmen HMAC-SHA1 und ECDSA im Demonstrator-Szenario angestellt. Hierbei wird ein PIR-Event auf beiden Sensorknoten ausgelöst, mit einem MAC (HMAC-SHA1) oder einer Signatur (ECDSA) versehen und an die Basisstation kommuniziert. Für beide Algorithmen wird der Energiebedarf der Softwareimplementierung mit dem Energiebedarf des VaultIC420 verglichen.

Als Medienzugriffsverfahren kommen IEEE 802.15.4 und TinyOS-LPL zum Einsatz. Insgesamt werden die Messungen sowohl für die Softwareimplementierung als auch für den VaultIC420 für jedes Medienzugriffsverfahren 20-mal wiederholt.

Der gemessene Energiebedarf umfasst somit die Erfassung und Verarbeitung des PIR-Events, die kryptografische Verarbeitung inklusive Duty-Cycling des VaultIC420 und die Kommunikation mit der Basisstation. Bei TinyOS-LPL wurde ein Aufwachintervall von $t_{LPL} = 1000ms$ genutzt, die Dauer eines CCA Checks beträgt wie in den vorigen Abschnitten 12 ms.

Der durchschnittliche Energiebedarf pro erzeugtem und versendetem PIR-Event bei der Verwendung von HMAC-SHA1 ist in Abbildung 6.13 dargestellt. Auf der x-Achse sind die beiden verwendeten Medienzugriffsverfahren unterschieden, auf der y-Achse ist der Energiebedarf in mJ abgebildet. Für beide Medienzugriffsverfahren ist der Energiebedarf der Softwareimplementierung geringer als der Energiebedarf des VaultIC420.

Die Softwareimplementierung hat einen Energiebedarf von $6,11mJ$ (IEEE 802.15.4) und $26,57mJ$ (TinyOS-LPL). Der Energiebedarf des VaultIC420

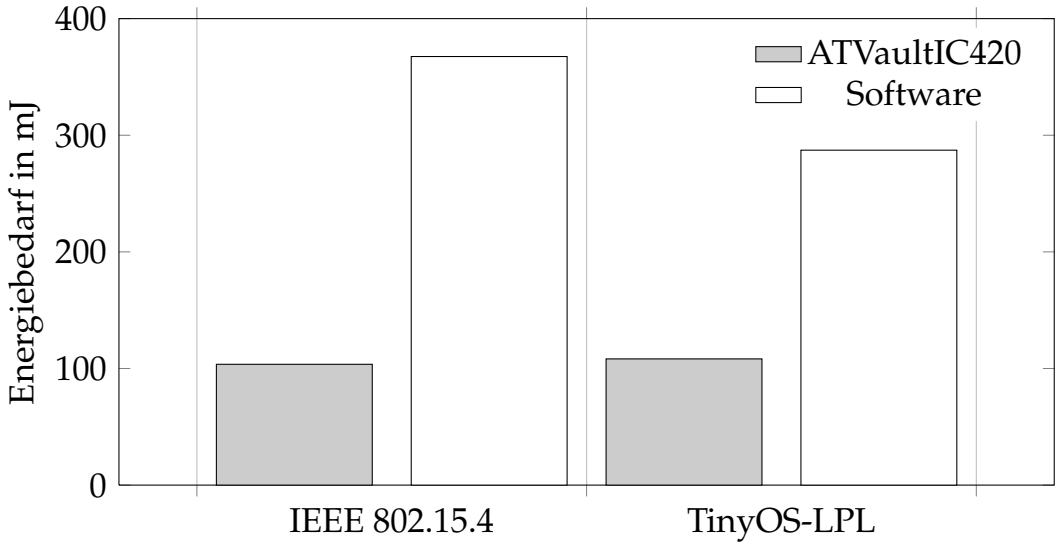


Abbildung 6.14 Vergleich des Energiebedarfs von ECDSA im FleGSens-Demonstrator.

beträgt $79,51mJ$ mit IEEE 802.15.4 und $98,41mJ$ mit TinyOS-LPL. Hiervon entfallen, wie in den vorigen Messungen gezeigt, etwa $40mJ$ auf die Initialisierungsphase des VaultIC420 und $2mJ$ auf die Berechnung mit HMAC-SHA1. Der restliche Energiebedarf wird durch die Kommunikation verursacht. Da der hier vermessene Sensorknoten nur Nachrichten sendet, bedeutet dies bei der Verwendung von TinyOS-LPL einen erhöhten Energiebedarf aufgrund der Präambel (maximal 1 Sekunde).

Diese Messergebnisse stehen im Widerspruch zu den Messungen des Energiebedarfs für die reine kryptografische Verarbeitung im vorigen Abschnitt, dort hatte die Hardwareimplementierung einen geringeren Energiebedarf. Dieser Unterschied lässt sich auf den Energiebedarf der Initialisierungsphase zurückführen, welche jedesmal beim Anschalten des VaultIC420 durchgeführt werden muss. Der VaultIC420 hat deutlich mehr Energiebedarf durch die Initialisierungsphase (etwa 40 mJ) im Vergleich zu der Ersparnis durch die schnellere Berechnung des HMAC-SHA1 MACs (etwa 1 mJ).

Der durchschnittliche Energiebedarf pro erzeugtem und versendeten PIR-Event bei der Verwendung von ECDSA ist in Abbildung 6.14 dargestellt. Auf der x-Achse sind die beiden verwendeten Medienzugriffsverfahren unterschieden, auf der y-Achse ist der Energiebedarf in mJ abgebildet.

Zu erkennen ist der geringere Energiebedarf bei der Verwendung des VaultIC420 im Vergleich zu der Softwareimplementierung. Bei der Verwendung des VaultIC420 zeigt sich ein Energiebedarf von $103mJ$ mit IEEE802.15.4 bzw. $108mJ$ mit TinyOS-LPL. Für die Softwareimplementierung zeigt sich ein 3,5-

mal höherer Energiebedarf mit IEEE802.15.4 ($367mJ$) und ein 2,7-mal höherer Energiebedarf mit TinyOS-LPL.

Insgesamt zeigt der VaultIC420 somit einen geringeren Energiebedarf bei der Verwendung von ECDSA, unabhängig vom eingesetzten Medienzugriffsverfahren. Der zusätzliche Energiebedarf der Initialisierungsphase wird durch die deutlich kürzere (und somit im Energiebedarf günstigere) Berechnung der Signatur ausgeglichen.

6.3.7 Diskussion der Ergebnisse

Die Evaluierung des Energiebedarfs hardwarebasierter Sicherheitsmechanismen hat gezeigt, dass je nach Anwendungsszenario und nach eingesetztem kryptografischen Algorithmus der Energiebedarf deutlich gesenkt werden kann. Dies gilt insbesondere bei der Verwendung von asymmetrische Algorithmen wie beispielsweise ECDSA. Für die im Rahmen dieser Arbeit verglichenen symmetrischen Algorithmen zeigte sich, dass der Energiebedarf beim Einsatz des VaultIC im Gegensatz zu den Erwartungen nicht immer geringer war. Die Berechnung von AES zeigte mit der Softwareimplementierung einen deutlich geringeren Energiebedarf.

Im Rahmen der Evaluierung wurde auch gezeigt, dass der Overhead für den Einsatz des Hardwaremoduls ein Duty-Cycling des Moduls notwendig macht. Weiterhin zeigt sich, dass die notwendige Initialisierungsphase im Rahmen der Evaluierung des Energiebedarfs mit einbezogen werden muss. So zeigt sich beispielsweise, dass die reine Berechnung von HMAC zwar mit dem VaultIC420 einen geringeren Energiebedarf aufweist, im Zusammenspiel mit der Initialisierungsphase der Energiebedarf jedoch größer als die im Vergleich genutzte Softwareimplementierung ist.

Ein weiterer Vorteil des Einsatzes des VaultIC420 ist der deutlich geringere Speicherbedarf. Auf dem hier verwendeten IRIS-Sensorknoten stünde bei der Verwendung der Softwareimplementierungen lediglich etwa 2000 Byte an RAM für die Anwendung zur Verfügung. Mit dem VaultIC420 vergrößert sich der verfügbare Speicher um etwa das Dreifache.

Aufgrund der deutlich kürzeren Berechnungszeiten mit Hilfe des Hardwarebausteins für die asymmetrischen Algorithmen und auch HMAC-SHA1 kann die Nutzung des VaultIC420 neben dem geringeren Energiebedarf hilfreich sein. Insbesondere wenn eine zeitnahe kryptografische Verarbeitung von Daten notwendig ist, kann der Einsatz des VaultIC420 je nach Anwendungsszenario sinnvoll sein.

6.4 Zusammenfassung

Im Rahmen dieses Kapitels wurde der Energiebedarf des VaultIC420 im Vergleich zu reinen Softwareimplementierungen verglichen. Die Messungen des Energiebedarfs wurden hierfür im Testbed SANDbed durchgeführt. Es konnte gezeigt werden, dass der Einsatz von hardwarebasierten Sicherheitsmechanismen den Energiebedarf von kryptografischen Algorithmen deutlich gesenkt werden kann. Dies gilt insbesondere für asymmetrische kryptografische Algorithmen wie das hier untersuchte ECDSA. Für symmetrische Algorithmen gilt, dass der Energiebedarf aufgrund der hohen Leistungsaufnahme des VaultIC420 und der kurzen Brechnungszeit, sowohl in Software als auch in Hardware, im Allgemeinen nicht geringer ist.

7. Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde der Einfluss verschiedener Faktoren auf den Energiebedarf von Sicherheitsmechanismen in drahtlosen Sensornetzen am Beispiels FleGSens evaluiert. Eine der Herausforderungen beim Entwurf des FleGSens-Gesamtsystem war die geforderte Mindestlaufzeit von bis zu 7 Tagen. Um dies zu erreichen, mussten verschiedene Maßnahmen zur Senkung des Energiebedarfs der eingesetzten Sicherheitsmechanismen und Protokolle durchgeführt werden. Im Rahmen des Projekts gestaltete sich insbesondere die Evaluation des Energiebedarfs schwierig, da die zum damaligen Zeitpunkt zur Verfügung stehenden Evaluationswerkzeuge hierfür nicht geeignet waren.

Im Mittelpunkt der Untersuchungen dieser Dissertation stand insbesondere der Einfluss des Medienzugriffsverfahrens bzw. der dort eingesetzten Duty-Cycling Strategie auf den Energiebedarf. Als Evaluierungswerkzeuge kam dabei neben dem Testbed SANDbed auch der Simulator AVRORA+ zum Einsatz, dessen Energiemodell im Rahmen dieser Dissertation entstand. Am Beispiel des Schlüsselaustauschs wurde untersucht, welchen Einfluss die Wahl und die Parametrisierung des Medienzugriffsverfahren auf den Energiebedarf und die Dauer des Schlüsselaustauschs hat. Weiterhin wurde evaluiert, inwiefern sich hardwarebasierte Sicherheitsmechanismen in drahtlosen Sensornetzen eignen und ob deren Einsatz den Energiebedarf im Vergleich zu Softwareimplementierungen verringern kann.

7.1 Beiträge dieser Arbeit

Die Beiträge dieser Arbeit lassen sich wie folgt zusammenfassen:

- Mit AVRORA+ wurde eine Evaluationswerkzeug vorgestellt, mit dessen Hilfe der Energiebedarf von Sensornetzanwendungen möglichst realitätsnah evaluiert werden kann. Hierfür wurde das Energiemodell von AVRORA im Vergleich zu Energiebedarfsmessungen in SANDbed analysiert und evaluiert. Hierfür wurden die elementaren Hardwareelemente eines Sensorknotens, Prozessor und Funkschnittstelle, mit Hilfe sogenannter Mikrobenchmarks vermessen. Neben den Mikrobenchmarks wurde auch anhand einer einfachen Testanwendung der Energiebedarf der Kommunikation zwischen unterschiedlichen Sensorknoten untersucht. Mit Hilfe der Untersuchungen konnte das Energiemodell derart erweitert und parametrisiert werden, dass die simulierten Energiebedarfswerte vergleichbar mit den in SANDbed gemessen Werten sind. Das Energiemodell ist dabei spezifisch für die MICAz-Plattform. Das generelle Vorgehen zum Entwurf eines Energiemodells, also die Kombination von Testbed-Messungen und Ausführung im Simulator, kann auch für andere Komponenten genutzt werden. Beispielsweise ist es denkbar, die Simulationsumgebung des Contiki-Betriebssystems mit dem in dieser Dissertation genutzten Vorgehen um ein Energiemodell zu erweitern. Weiterhin ist es einfach möglich, auch AVRORA+ um weitere Energiemodelle für andere Plattformen zu erweitern. Beispielsweise wurde in [82] der Simulator AVRORA+ um ein Energiemodell der IRIS-Sensorknoten erweitert.

- Im Rahmen der Evaluierung wurde mit Hilfe einer einfachen Anwendung der Energiebedarf und die Latenz verschiedener Medienzugriffsverfahren evaluiert. Mit Hilfe der Experimente konnte gezeigt werden, dass gängige Annahmen im Bezug auf den Energiebedarf nicht immer gültig oder verallgemeinerbar sind. Beispielsweise konnte gezeigt werden, dass eine Vergrößerung der zu übertragenen Datenmenge nicht automatisch zu einem erhöhten Energiebedarf führt. Weiterhin konnte gezeigt werden, dass der Energiebedarf der Anwendung, je nach Medienzugriffsverfahren, nicht gleichmäßig auf Sender- und Empfängerknoten verteilt ist. Beim Einsatz von TinyOS-LPL ist das Senden von Daten für fast alle Parametrisierungen deutlich teurer als das Empfangen. Ein weiterer Gegenstand der Evaluierung war die Latenz der Datenübertragung. Hier konnte gezeigt werden, dass die Parametrisierung des Medienzugriffsverfahrens und des Duty-Cycles einen erheblich Einfluss auf die Latenz der Übertragungen hat.

- Am Beispiel des Schlüsselaustausches in drahtlosen Sensornetzen wurde gezeigt, dass Seiteneffekte zwischen relativ langen Berechnungen und der Kommunikation den Energiebedarf des Schlüsselaustauschprotokolls stark beeinflussen. Beispielsweise konnte bei der Evaluierung von ECDH-ECDSA mit TinyOS-LPL gezeigt werden, dass Timing-Effekte bei der Ausführung des Schlüsselaustauschs im Zusammenspiel mit dem Medienzugriffsverfahren den Energiebedarf deutlich beeinflussen. Je nach eingesetztem Medienzugriffsverfahren und kryptografischem Algorithmus besteht ein Zusammenhang zwischen Latenz, Dauer eines Schlüsselaustauschs und dem Energiebedarf aufgrund des Duty-Cyclings der Funkschnittstelle. Weiterhin konnte gezeigt werden, dass die Wahl des Schlüsselaustauschprotokolls weniger Einfluss auf den Energiebedarf hat als die Parametrisierung des Duty-Cycles. Durch die Verwendung verschiedener Evaluationsmetriken wurde deutlich, dass die Wahl der Metrik je nach Evaluierungsziel entscheidend sein kann.

- Im Rahmen der Evaluierung der hardwarebasierten Sicherheitsmechanismen konnte gezeigt werden, dass diese den Energiebedarf einer Anwendung im Vergleich zu einer Softwareimplementierung deutlich senken können. Dies gilt insbesondere für die asymmetrischen Algorithmen. Für die symmetrischen Algorithmen wurde deutlich, dass die Verwendung von Softwareimplementierung in vielen Fällen weniger Energiebedarf verursachte. Dies lässt sich durch den Overhead für die Kommunikation und der Leistungsaufnahme des Hardwarebausteins erklären. Es wurde somit deutlich, dass ein Duty-Cycling des Hardwarebausteins aufgrund der hohen Leistungsaufnahme notwendig ist.

7.2 Ausblick

Aus den Ergebnisse dieser Arbeit lassen sich einige weitere Ansätze ableiten:

- Im Rahmen dieser Arbeit wurden keine Multi-Hop Netze untersucht. Insbesondere bei der Untersuchung der Schlüsselaustauschprotokolle ist der Einfluss der Topologie und des unter Umständen notwendigen Routingprotokolls ein weiterer Untersuchungsgegenstand. Auch hier ist zu erwarten, dass Timing-Effekte zwischen dem Medienzugriffsverfahren und beispielsweise dem Rountingprotokoll bestehen.

- Bei der Standardisierung der 6LoWPAN-Protokollfamilie durch die IETF wurden bisher keine Untersuchungen zum Einfluss des Medienzugriffsverfahrens auf den Energiebedarf der Protokolle oder die Latenz der

Nachrichtenübertragung durchgeführt. Die im Rahmen dieser Dissertation gewonnen Erkenntnisse legen nahe, dass auch bei diesen Protokollen Nebenwirkungen oder Timing-Effekte auftreten können, die so bisher nicht erwartet wurden. Dies könnte in zukünftigen Arbeiten untersucht werden.

- Der Einsatz der hardwarebasierten Sicherheitsbausteine zur effizienteren Implementierung von Sicherheitsprotokollen für das Internet der Dinge wurde bisher nicht evaluiert. Aktuelle Arbeiten der IETF zur Implementierung von DTLS für das Internet der Dinge lassen vermuten, dass diese Protokolle vom Einsatz der Hardwarebausteine profitieren könnten. Die Erkenntnisse dieser Arbeit können daher in die Spezifizierung dieser Protokolle einfließen.

Literaturverzeichnis

[1] MEMSIC: *MicaZ Sensorknoten*. http://www.memsic.com/wireless-sensor-networks/TPR2420, überprüft im April 2013

[2] WEISER, M. : Human-computer interaction. In: *The computer for the 21st century*. 1995, S. 933–940

[3] ASHTON, K. : That 'Internet of Things' Thing. In: *RFID Journal* (Juli 2009). http://www.rfidjournal.com/articles/view?4986

[4] MORELLI, B. : IHS Research: ARM to bring the Internet of Things to life at its Cambridge Campus. (2013). http://www.arm.com/about/newsroom/

[40] HAAS, C. ; KRÜGER, D. ; PFISTERER, D. ; FISCHER, S. ; DUDEK, D. ; KUNTZ, A. ; ZITTERBART, M. ; ROTHENPIELER, P. : FleGSens - secure area monitoring using wireless sensor networks. In: *Proceedings of the 4th Safety and Security Systems in Europe*. Potsdam, Deutschland, April 2009

[41] HAAS, C. ; DUDEK, D. ; KUNTZ, A. ; ZITTERBART, M. ; KRÜGER, D. ; ROTHENPIELER, P. ; PFISTERER, D. ; FISCHER, S. : *A Wireless Sensor Network for Border Surveillance*. Demonstrator, November 2009

[5] HERGENRÖDER, A. ; HORNEBER, J. ; WILKE, J. : SANDbed: A WSAN Testbed for Network Management and Energy Monitoring. In: *Proceedings of the 8. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze*. Hamburg, Deutschland, August 2009, S. 71–73

[6] TITZER, B. L. ; LEE, D. K. ; PALSBERG, J. : Avrora: Scalable Sensor Network Simulation with Precise Timing. In: *Proceedings of the 4th international symposium on Information processing in sensor networks*. Los Angeles, California, USA, April 2005 (IPSN '05)

[7] HAAS, C. ; WILKE, J. ; STÖHR, V. : Realistic Simulation of Energy Consumption in Wireless Sensor Networks. In: *Proceedings of the 9th European Conference on Wireless Sensor Networks (EWSN)*. Trento, Italien, Februar 2012, S. 82–97

[8] HAAS, C. ; WILKE, J. : Energy Evaluations in Wireless Sensor Networks - A Reality Check. In: *Proceedings of the 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*. Miami, Florida, USA, November 2011, S. 27–30

[9] HAAS, C. ; WILKE, J. : Evaluating the Energy-Efficiency of Key Exchange Protocols in Wireless Sensor Networks. In: *Proceedings of the 7th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks - PM2HW2N '12*. Zypern, November 2012, S. 133–140

[10] HAAS, C. ; WILKE, J. ; KNITTEL, F. : Evaluating the Energy-Efficiency of the Rich Uncle Key Exchange Protocol in WSNs. In: *Proceedings of the 38th IEEE Conference on Local Computer Networks (LCN)*. Sydney, Australien, Oktober 2013

[11] HAAS, C. ; HERGENRÖDER, A. ; WILKE, J. ; WISKOT, T. ; NIEDERMANN, M. : Evaluating Energy-Efficiency of Hardware-based Security Mechanisms . In: *Proceedings of the 9th European Conference on Wireless Sensor Networks*. Trento, Italien, Februar 2012, S. 80–81

[12] HAAS, C. ; MUNZ, S. ; WILKE, J. ; HERGENRÖDER, A. : Evaluating Energy-Efficiency of Hardware-based Security Mechanisms. In: *Proceedings of the 9th IEEE International Conference on Pervasive Computing and Communications Workshops (PerSeNS)*. San Diego, Californien, USA, März 2013, S. 560–565

[13] HEALY, M. ; NEWE, T. ; LEWIS, E. : Wireless Sensor Node Hardware: A Review. In: *Proceedings of IEEE Sensors* IEEE, 2008, S. 621–624

[14] MEMSIC: *Mica Sensorknoten*. http://www.memsic.com/wireless-sensor-networks/MCS410CA, überprüft im April 2013

[15] MEMSIC: *IRIS Sensorknoten*. http://www.memsic.com/wireless-sensor-networks/XM2110CA, überprüft im April 2013

[16] IEEE: Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). In: *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)* (2006), S. 1–305

[17] THOMAS KUNZ, M. O. F.: Operating Systems for Wireless Sensor Networks: a Survey. In: *IEEE Sensors Journal* 6 (2011), S. 1–31

[18] LEVIS, P. ; MADDEN, S. ; POLASTRE, J. ; SZEWCZYK, R. ; WHITEHOUSE, K. ; WOO, A. ; GAY, D. ; HILL, J. ; WELSH, M. ; BREWER, E. ; CULLER, D. ; WEBER, W. ; RABAEY, J. ; AARTS, E. : TinyOS: An Operating System for Wireless Sensor Networks. In: *Ambient Intelligence*, 2005, S. 115–148

[19] GAY, D. ; LEVIS, P. ; BEHREN ANDMATTWELSH, R. von ; BREWER, E. ; CULLER, D. : The nesC Language: A Holistic Approach to Networked. In: *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*. San Diego, California, USA, 2003, S. 1–11

[20] REZAEI, Z. ; MOBININEJAD, S. : Energy Saving in Wireless Sensor Networks. In: *International Journal of Computer Science&Engineering Survey* 3(1) (2012), S. 23–37

[21] MOSS, D. ; HUI, J. ; KLUES, K. : *TinyOS Low Power Listening*. http://www.tinyos.net/tinyos-2.x/doc/html/tep105.html, überprüft im Oktober 2013

[22] MOSS, D. ; LEVIS, P. : Box-MACs: Exploiting Physical and Link Layer Boundaries in Low-Power Networking / Stanford University. 2008. – Forschungsbericht

[23] YE, W. ; HEIDEMANN, J. ; ESTRIN, D. : An energy-efficient MAC protocol for wireless sensor networks. In: *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* Bd. 3. New York City, New York, USA, 2002, S. 1567–1576

[24] KARN, P. : MACA: A New Channel Access Method for Packet Radio. In: *Proceedings of the 9th ARRL Computer Networking Conference, Ontario, Kanada*, 1990

[25] HERGENRÖDER, A. ; HORNEBER, J. ; MEIER, D. ; ZITTERBART, M. ; ARMBRUSTER, P. : *Demo Abstract: Distributed Energy Measurements in Wireless Sensor Networks*. Demonstrator, November 2009

[26] HERGENRÖDER, A. ; WILKE, J. ; MEIER, D. : Distributed Energy Measurements in WSN Testbeds with a Sensor Node Management Device (SNMD). In: *Workshop Proceedings of the 23th International Conference on Architecture of Computing Systems (ARCS)*. Hannover, Februar 2010, S. 341–438

[27] LANDSIEDEL, O. ; WEHRLE, K. ; GOTZ, S. : Accurate prediction of power consumption in sensor networks. In: *Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors*. Washington, DC, USA, 2005 (EmNets '05), S. 37–44

[28] PAZ ALBEROLA, R. de ; PESCH, D. : AvroraZ: extending Avrora with an IEEE 802.15.4 compliant radio chip model. In: *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*. Vancouver, Kanada, 2008 (PM2HW2N '08), S. 43–50

[29] CHEN, X. ; MAKKI, K. ; YEN, K. ; PISSINOU, N. : Sensor Network Security: a Survey. In: *Communications Surveys Tutorials, IEEE* 11 (2009), Nr. 2, S. 52–73

[30] STANDARDS, N. I. ; TECHNOLOGY: Advanced Encryption Standard. In: *NIST FIPS PUB 197* (2001)

[31] RIVEST, R. ; BALDWIN, R. : *The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms, RFC 2040*. `http://tools.ietf.org/html/rfc2040`

[32] RIVEST, R. L. ; SHAMIR, A. ; ADLEMAN, L. : A Method for Obtaining Digital Signatures and Public-key Cryptosystems. In: *Commun. ACM* 21 (1978), Februar, Nr. 2, S. 120–126

[33] *Standards for efficient cryptography, SEC 1: Elliptic Curve Cryptography*. `http://www.secg.org/download/aid-780/sec1-v2.pdf`. Version: Mai 2009

[34] DU, W. ; DENG, J. ; HAN, Y. ; CHEN, S. ; VARSHNEY, P. : A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. In: *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies* Bd. 1. Honk Kong, China, 2004, S. 586–597

[35] CHAN, H. ; PERRIG, A. ; SONG, D. : Wireless Sensor Networks. 2004, Kapitel Key Distribution Techniques for Sensor Networks, S. 277–303

[36] PARK, S. ; KIM, K. ; (ED.), W. H. ; CHAKRABARTI, S. ; LAGANIER, J. : *IPv6 over Low Power WPAN Security Analysis*. https://tools.ietf.org/html/draft-daniel-6lowpan-security-analysis-05, March 2011

[37] GROSSSCHÄDL, J. ; SZEKELY, A. ; TILLICH, S. : The energy cost of cryptographic key establishment in wireless sensor networks. In: *Proceedings of the 2nd ACM symposium on Information, computer and communications security*. Singapore, 2007 (ASIACCS '07), S. 380–382

[38] MEULENAER, G. de ; GOSSET, F. ; STANDAERT, F.-X. ; PEREIRA, O. : On the Energy Cost of Communication and Cryptography in Wireless Sensor Networks. In: *Proceedings of the 2008 IEEE International Conference on Wireless & Mobile Computing, Networking & Communication*. Avignon, Frankreich, 2008, S. 580–585

[39] GALINDO, D. ; ROMAN, R. ; LOPEZ, J. : On the energy cost of authenticated key agreement in wireless sensor networks. In: *Wirel. Commun. Mob. Comput.* 12 (2012), S. 133–143

[42] COALESENSES: *http://www.coalesenses.com/*. November 2013

[43] JENNIC: *JN5139 Datenblatt*. `http://www.jennic.com/files/product_briefs/JN-DS-JN5139-001-1v8.pdf`. Version: November 2013

[44] COALESENSES: *iSense Betriebssystem*. `http://www.coalesenses.com/index.php/products/solutions/isense-software/os-net-firmware/`. Version: November 2013

[45] FEKETE, S. ; KROLLER, A. ; FISCHER, S. ; PFISTERER, D. : Shawn: The fast, highly customizable sensor network simulator. In: *Networked Sensing Systems, 2007. INSS '07. Fourth International Conference on*. Braunschweig, Deutschland, 2007, S. 299–299

[46] STÖHR, V. : *Energiemessung in drahtlosen Sensornetzen - Simulation vs. Realität*, Institut für Telematik, Universität Karlsruhe, Diplomarbeit, 2011. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[47] MEMSIC: *TelosB Sensorknoten*. http://www.memsic.com/wireless-sensor-networks/TPR2420, überprüft im April 2013

[48] LEVIS, P. ; LEE, N. ; WELSH, M. ; CULLER, D. : TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In: *Proceedings of the 1st international conference on Embedded networked sensor systems*. Los Angeles, California, USA, 2003 (SenSys '03), S. 126–137

[49] SHNAYDER, V. ; HEMPSTEAD, M. ; CHEN, B.-r. ; ALLEN, G. W. ; WELSH, M. : Simulating the power consumption of large-scale sensor network applications. In: *Proceedings of the 2nd international conference on Embedded*

networked sensor systems. Baltimore, MD, USA, 2004 (SenSys '04), S. 188–200

[50] PERLA, E. ; CATHÁIN, A. O. ; CARBAJO, R. ; HUGGARD, M. ; GOLDRICK, C. M.: PowerTOSSIMz: Realistic Energy Modelling for Wireless Sensor Network Environments. In: *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*. Vancouver, Canada, 2008 (PM2HW2N '08), S. 35–42

[51] VARGA, A. ; HORNIG, R. : An overview of the OMNeT++ simulation environment. In: *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. Marseille, Frankreich, 2008 (Simutools '08), S. 1–10

[52] KÖPKE, A. ; SWIGULSKI, M. ; WESSEL, K. ; WILLKOMM, D. ; HANEVELD, P. K. ; PARKER, T. ; VISSER, O. ; LICHTE, H. S. ; VALENTIN, S. : Simulating Wireless and Mobile Networks in OMNeT++ – The MiXiM Vision. In: *OMNeT++ 2008: Proceedings of the 1st International Workshop on OMNeT++ (hosted by SIMUTools 2008)*. Marseille, Frankreich, 2008

[53] KUNTZ, A. ; SCHMIDT-EISENLOHR, F. ; GRAUTE, O. ; HARTENSTEIN, H. ; ZITTERBART, M. : Introducing probabilistic radio propagation models in OMNeT++ mobility framework and cross validation check with NS-2. In: *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. Marseille, Frankreich, 2008 (Simutools '08), S. 72:1–72:7

[54] CHEN, F. ; DIETRICH, I. ; GERMAN, R. ; DRESSLER, F. : An Energy Model for Simulation Studies of Wireless Sensor Networks using OMNeT++. In: *Praxis der Informationsverarbeitung und Kommunikation (PIK)* 32 (2009), June, S. 133–138

[55] BOULIS, A. : Castalia: Revealing Pitfalls in Designing Distributed Algorithms in WSN. In: *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*. Sydney, Australien, 2007, S. 407–408

[56] OSTERLIND, F. ; DUNKELS, A. ; ERIKSSON, J. ; FINNE, N. ; VOIGT, T. : Cross-Level Sensor Network Simulation with COOJA. In: *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*. Tampa, Florida, USA, nov. 2006, S. 641 –648

[57] DUNKELS, A. ; GRONVALL, B. ; VOIGT, T. : Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors. In: *Proceedings of*

the 29th Annual IEEE International Conference on Local Computer Networks. Tampa, Florida, USA, 2004 (LCN '04), S. 455–462

[58] ERIKSSON, J. ; ÖSTERLIND, F. ; FINNE, N. ; TSIFTES, N. ; DUNKELS, A. ; VOIGT, T. ; SAUTER, R. ; MARRÓN, P. J.: COOJA/MSPSim: interoperability testing for wireless sensor networks. In: *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. Rom Italien, 2009 (Simutools '09), S. 27:1–27:7

[59] ATMEL: *Atmega128(L) datasheet*. `http://www.atmel.com/Images/doc2467.pdf`. Version: April 2013

[60] WILKE, J. : *Energieeffiziente Concast-Kommunikation in drahtlosen Sensornetzen*, Karlsruhe Institute für Technologie (KIT), Diss., 2013

[61] ZEPF, T. : *Reality Check: Evaluierung von TinyOS-LPL auf MicaZ Sensorknoten* , Institut für Telematik, Universität Karlsruhe, Studienarbeit, 2011. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[62] WISKOT, T. : *Portierung und Evaluierung des Medienzugriffsverfahrens Sensor-MAC (S-MAC)*, Institut für Telematik, Karlsruhe Institut für Technologie (KIT), Diplomarbeit, 2012. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[63] KNITTEL, F. : *Evaluierung der Energieeffizienz des Rich Uncle Protokolls*, Institut für Telematik, Universität Karlsruhe, Diplomarbeit, 2012. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[64] WILKE, J. ; HAAS, C. : Energy-Efficiency of Concast Communication in Wireless Sensor Networks. In: *Proceedings of the 10th Annual Conference on Wireless On-Demand Network Systems and Services (WONS)*. Banff, Kanada, März 2013, S. 34–38

[65] KLOCK, T. : *Evaluierung der Energieeffizienz von Schlüsselaustauschmechanismen in Concast-Szenarien*, Institut für Telematik, Karlsruher Institut für Technologie, Diplomarbeit, 2012. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[66] HOF, H. ; BAUMGART, I. ; ZITTERBART, M. : Key Exchange for Service Discovery in Secure Content Addressable Sensor Networks. In: *Proc. Kommunikation in Verteilten Systemen (KiVS 2007)*, Springer, Februar 2007

[67] PERRIG, A. ; SZEWCZYK, R. ; TYGAR, J. D. ; WEN, V. ; CULLER, D. E.: SPINS: Security Protocols for Sensor Networks. In: *Wireless Networking* 8 (2002), September, Nr. 5, S. 521–534

[68] TINYOS COMMUNITY: *TinyOS-Contrib*. `https://github.com/tyll/tinyos-2.x-contrib`. Version: April 2013

[69] LIU, A. ; NING, P. : TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks. In: *Proceedings of the 7th international conference on Information processing in sensor networks*. St. Louis , Missouri , USA, 2008, S. 245–256

[71] TRUSTED COMPUTING GROUP: *Trusted Platform Module (TPM) Specifications*. https://www.trustedcomputinggroup.org/specs/TPM, 2007

[72] WISKOT, T. : *Implementierung von hardwarebasierten Sicherheitsmechanismen in WSNs*, Institut für Telematik, Universität Karlsruhe (TH), Studienarbeit, 2012. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[73] MUNZ, S. : *Evaluierung von hardware-basierten Sicherheitsmechanismen in WSNs*, Institut für Telematik, Karslruher Institut für Technologie (KIT), Studienarbeit, 2013. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart

[74] HU, W. ; CORKE, P. ; SHIH, W. C. ; OVERS, L. : secFleck: A Public Key Technology Platform for Wireless Sensor Networks. In: *Proceedings of the 6th European Conference on Wireless Sensor Networks*. Cork, Ireland, 2009 (EWSN '09), S. 296–311

[75] HU, W. ; CORKE, P. ; SHI, W. S. ; OVERS, L. : secFleck: A Public Key Technology Platform for Wireless Sensor Networks. In: *Lecture Notes in Computer Science* (2009), February, S. 296–311

[76] SITKA, P. ; CORKE, P. ; OVERS, L. ; VALENCIA, P. ; WARK, T. : Fleck - A platform for real-world outdoor sensor networks. In: *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*. Melbourne, Australien, 2007, S. 709–714

[77] KOTHMAYR, T. ; SCHMITT, C. ; HU, W. ; BRÜNIG, M. ; CARLE, G. : DTLS based security and two-way authentication for the Internet of Things. In: *Ad Hoc Networks* 11 (2013), November, Nr. 8, S. 2710–2723

[78] INSIDE SECURE: *ATVaultIC420 Sicherheitsmodul*. `http://www.insidesecure.com/`. Version: November 2013

[79] NXP SEMICONDUCTORS (Hrsg.): *UM10204, I2C bus specification and user manual*. NXP Semiconductors, `http://www.nxp.com/documents/user_manual/UM10204.pdf`

[80] OTHMAN, S. ; TRAD, A. ; YOUSSEF, H. : Performance Evaluation of Encryption Algorithms for Wireless Sensor Networks. In: *Information Technology and e-Services (ICITeS), 2012 International Conference on*. Changhua, Taiwan, 2012, S. 1–8

[81] OTHMAN, S. B. ; TRAD, A. ; ALZAID, H. ; YOUSSEF, H. : Performance Evaluation of EC-ElGamal Encryption Algorithm for Wireless Sensor Networks. In: *Wireless Mobile Communication and Healthcare* Bd. 61. 2013

[82] JUNG, M. : *Integration und Evaluation von IRIS-Sensorknoten in den Simulator Avrora*, Institut für Telematik, Universität Karlsruhe, Bachelorarbeit, 2013. – Betreuer: Joachim Wilke, Christian Haas und Martina Zitterbart