Simulation et optimisation de la consommation énergétique de réseaux de capteurs sans fil

Nanhao Zhu

UNIVERSITÉ DE LYON - ÉCOLE CENTRALE DE LYON

ÉCOLE DOCTORALE Electronique, Electrotechnique, Automatique

Institut des Nanotechnologies de Lyon (INL)

P H D T H È S E

pour obtenir le grade de

DOCTEUR DE L'ÉCOLE CENTRALE DE LYON

Discipline : ELECTRONIQUE

présentée et soutenue par

Nanhao ZHU

Simulation and Optimization of Energy Consumption on Wireless Sensor Networks

Directeur de Thèse : Ian O'CONNOR

préparé à INL, CONCEPTION DE SYSTÈMES HÉTÉROGÈNES

defended on Oct 11, 2013

Jury :

Rapporteur :	**Luc HÉBRARD**	-	Professeur, Université de Strasbourg-InESS
Rapporteur :	**Cécile BELLEUDY**	-	HDR, Université de Nice-Sophia Antipolis-LEAT
Directeur :	**Ian O'CONNOR**	-	Professeur,École Centrale de Lyon-INL
Président :	**Sophie CAVASSILA**	-	Professeur,Université Lyon 1-CREATIS

To my parents

Acknowledgments

I really appreciate that I have such an opportunity to express my great gratitude and respect to people who helped me when I was pursuing my Ph.D. Without their supports and encouragements I cannot go so far.

It is difficult to overstate my greatest gratitude to my thesis supervisor Prof. Ian O'Connor. First, I would like to thank him for his patient guiding and inspiring throughout my study period. Secondly, I highly appreciate his encouragement and support in my research work, which helped me build confidence and courage to overcome difficulties. Finally, I am grateful for his great insight and suggestions and sharing so much time in my thesis and paper writings. I would have been lost without him.

I would also like to thank China Scholarship Council (CSC) for the financial support in my Ph.D. studies. I would also like to express my grateful to Mrs Qiang Yapin (Office of Educational Affair of the Embassy of China) and Dr. Shang Liang (CSC coordinator) who helped me in my studies. I would like to thank Ecole Centrale de Lyon (ECL) for the tuition waiver and all the administrative and technical staffs in ECL, and I would like to give my thankfulness to Institute of Nanotechnology (INL) and the heterogeneous systems design group for providing me with travelling support to many international conferences to present my research work.

I would like to give a special thankfulness to Dr. Ning Sui and Dr. Zhugen Yang who worked in INL and Ph.D. student Tao Xu in MI lab, being my close friends, they were always ready to help me and stayed with me to give me encouragements. I also should thank my colleagues Dr. Vijayaragavan Visswanathan, Dr. Felipe Frantz, Dr. Du Wan, Zhenfu Feng, Huanhuan Liu, Dr. Taiping Zhang, Dr. Xianqin Meng, Zhen Li and Dr. Kevin Chen who are always willing to share their ideas not only in research but also in life which make INL such a great place to learn and grow. I am also pleased to thank engineer Laurent Carrel and secretary Patricia Dufaut for their kind help in my studies in heterogeneous system design group. I would also like to give my grateful for the Chinese-spoken community at ECL, and especially to Fanbing Ye, Xi Zhang, Tianli Huang, Changwei Zhou and Gang Huang who provide a fun environment in ECL after work.

I am also grateful for many peoples in my former University (Communication University of China-CUC), a special thank to Zhengxiang Li and Qibin Lu who guided me into research work, and I would also like to thank Prof. Rui Lv, Prof. Zhanxin Yang, Prof. Zhibin Zeng, Prof. Yahui Hou, Dr. Xiangran Sun who encouraged me to study aboard and invigorated me all the time. I also wish to thank staffs in administration department who helped me in the application to Ecole Centrale de Lyon. I would also express my gratitude to Wenbin Wang, Longcheng Yu, Xiaoyu Miao, Tong Bi, Xiaohui Fan, MengMeng Kang, Qunchao Feng and Houwei Zhu as they were always my close friends and gave me a happy and unforgettable memory in my six studies in CUC. I also would like thank my class teacher Mrs Wu Youping who was very strict on my study during my high school.

Finally, and most significantly, I wish to give my greatest gratitude to my parents Huaiwu Zhu and Dejun Shen, my financee Qiong Jian. They supported me, believed in me and loved me. Without them, I cannot complete this thesis.

Contents

Abbreviations

iWEEP	-	iNL@WSN Energy Evaluation Platform
iWEEP_SW	-	iNL@WSN Energy Evaluation Platform_Software
iWEEP_HW	-	iNL@WSN Energy Evaluation Platform_Hardware
MEMD	-	Multichannel Energy Measurement Device
EDMSP	-	Energy Data Management Software Platform
iHop@Node	-	iNL@High Date Rate and Ultra Low Power Based Testbed Node
iHop@250K	-	iHop@Node with 250Kbps
iHop@1M	-	iHop@Node with 1Mbps
iHop@2M	-	iHop@Node with 2Mbps
iMASKO	-	iNL@MATLAB Genetic Algorithm based Sensor Network Optimizer
ESB	-	Enhanced ShockBurst
SB	-	ShockBurst
WSNs	-	Wireless Sensor Networks
COTS	-	Commercial-off-the-shelf
MCU	-	Microcontroller
MAC	-	Medium Access Control
QoS	-	Quality-of-Service
TX / RX	-	Transmission / Reception
CCA	-	Clear Channel Assessment
LPL	-	Low Power Listening
CFP	-	Contention Free Period
CAP	-	Contention Access Period
GTS	-	Guaranteed Time Slots
CSMA/CA	-	Carrier Sense Multiple Access with Collision Avoidance
OS	-	Operating System
SCNSL	-	SystemC Network Simulation Library
TLM	-	Transaction Level Modeling
SPI	-	Serial Peripheral Interface
FSM	-	Finite State Machine
CAF	-	Channel Access Failure
CF	-	Collision Failure
PTX / PRX	-	Primary Transmitter / Receiver
LOS	-	Line Of Sight
IRQ	-	Interrupt
BER	-	Bit Error Rate
PER	-	Packet Error Rate
GA	-	Genetic Algorithm

List of Figures

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Brief Introduction of Wireless Sensor Networks

With the great developments in embedded systems as well as in wireless communication technologies, wireless sensor networks (WSNs) have gained worldwide attention and have been developing rapidly in the past decade. Wireless sensor networks are large-scale networks with low-cost, small-size, low-power and limited processing sensor nodes deployed in various kinds of environments. As the basic part of WSNs, a typical sensor node comprises several components: microcontroller, transceiver, sensor and power supply (Fig.2.2). By combining these different components into a miniaturized device, these sensor nodes are multi-functional and have been applied to a wide variety of application scenarios: medical and health care [1], environment and ecological monitoring [2], home and building automation [3], industrial monitoring [4] and battlefield [5]. Thus, such highly diverse scenarios impose application-specific requirements on WSN design and distinguish them from conventional networks. The advantages of WSNs are numerous: it is easy to achieve great scalability and a wide range of densities, due to the low cost and small dimensions of sensor nodes. Usually, specific sensor network protocols and algorithms with self-organizing capabilities are designed and applied to sensor nodes, so the whole sensor network is low-maintenance and tolerant against communication failure and topology changes that could be caused by node malfunction, node mobility or energy depletion of the nodes. Further, sensor nodes can be deployed in harsh environments to carry out a given task without any human interference.

However, further and potential applications are limited due to inherent WSN disadvantages. For instance, the limited processing ability and low data rate of sensor nodes cannot guarantee high performance in some scenarios especially for real-time applications. Short communication range can cause energy waste and network inefficiency, since multi-hop communications are always required for data transport between source node and sink node. The severe energy constraints also lead to a worse observation: the improvement of data processing ability using powerful processors are not expected, since the energy will be depleted very quickly and renders the network useless for the given task. Limited energy also means it is not possible to also maintain the operation of multi-hop communications for a long time. Meanwhile, sensor nodes are

expected to be used heavily in many remote area applications, where the frequent changing or recharging of batteries is inconvenient. Hence, an energy-efficient sensor network is necessary to carry out the work in such conditions.

Therefore, taking energy consumption as the major consideration, while not undermining other performances, is desired as the priority strategy in the design of current wireless sensor networks.

1.2 Motivation

The emerging field of wireless sensor networks and their applications promises a higher quality of life in many aspects of daily human activities, but the strict energy constrains stated above significantly limit their functionalities and possible applications, which makes energy consumption one of the most critical concerns in WSNs. Therefore, energy consumption evaluation and analysis is a critical process in the design and implementation of the energy-efficient sensor network.

Despite accurate results, energy consumption analysis from real deployment can suffer limitations, because measurements from large-scale sensor nodes are hard to acquire, and sometimes these nodes are placed in harsh and inaccessible areas. At the other end of the scale, mathematics based analytical/theoretical models offer greater exploration capability but can oversimplify through idealized hypotheses, and lead to inaccurate results for practical problems. As a tradeoff solution, the simulation based approach is widely accepted and commonly used in the analysis of WSNs, since it provides a good balance between efficiency and accuracy. General simulation tools [6] [7] [8], written in C++, Java or MATLAB, provide satisfactory efficiency at the early design stages, since the models are usually at a high abstraction level to facilitate the testing and verification of algorithms, protocols and strategies. However, only a coarse energy consumption evaluation can thus be expected due to the lack of realistic low level models. Emulation tools [9] [10] [11] can compensate this drawback on energy consumption evaluation but at the cost of efficiency, and they are basically limited to specific hardware platforms and operating systems. SystemC offering support for HW/SW co-simulation, and concurrency and, modeling at different abstraction levels as well as other flexible and diverse modeling advantages, is regarded as a good alternative to model and simulate WSNs for accurate, flexible and quick energy evaluation. Meanwhile, measurements from real world testbed nodes are also considered significant since they offer a more realistic environment to the real deployment and are able to calibrate and improve the accuracy of energy evaluation. Besides, a comprehensive energy performance investigation needs to be done with different sensor node platforms and under different case scenarios.

Since wireless communication is perceived to be the most energy hungry part in the sensor node [12], a huge number of communication protocols have been designed in recent year to directly and effectively handle wireless communication in the sensor network with the specific objective of trying to maximize the energy saving. With the diversity of application scenarios, the design requirements are usually application-specific that could include many other kinds of metrics, in addition to energy performance. Therefore, it is clear that a single common or default protocol cannot suit every application and choosing appropriate parameter settings for real deployment is key to balancing the energy consumption with other competing metrics. In spite of the efficiency provided by simulation, exhaustive and full simulations on all the possible configurations of the protocol are not only time-consuming, but also unnecessary most of the time. Thus,

an approach that can effectively and automatically select an appropriate protocol parameter set out of hundreds or thousands of possible options is urgently needed. The fast and intelligent search ability of genetic algorithms (GAs) [13] enables the fine tuning of parameter space of the protocol to find the most suitable configuration.

1.3 Contribution

The first contribution is to design and implement iWEEP (**i**NL@WSN **E**nergy **E**valuation **P**latform) in a mixed energy evaluation method that combines simulation and testbed node measurements. iWEEP is used for flexible and accurate energy consumption evaluation on sensor node and sensor network. It consists of two parts:

i. a SystemC-based simulation environment at the system-level and transaction-level which implements several popular commercial off-the-shelf (COTS) low data rate based sensor node models including the Telos series (e.g., TelosB [14], Tmote Sky [15], Shimmer node [16]), the MICA series (e.g., MICA2 [17], MICAz [18]), and a lab-built high data rate and ultra-low power testbed node iHop@Node [19] (PIC16F+nRF24L01p, maximum data rate up to 2 Mbps). Four MAC protocols are integrated and modeled in iWEEP's SystemC simulation environment (iWEEP_SW) which are unslotted and slotted CSMA/CA protocols in IEEE 802.15.4 [20], and Enhanced ShockBurst (ESB) and ShockBurst (SB) protocols [21], embedded in the high data rate iHop@Node.

ii. the second part of iWEEP is the hardware based energy measurement platform (iWEEP_HW) that is made up of a multi-channel energy measurement device (MEMD) and energy data management software platform (EDMSP) to be able to measure the energy related data of iHop@Node, MICA mote, Telos mote and other similar sensor motes. For MEMD, it can provide simultaneous measurements on different components of a sensor node or simultaneous measurements on several different nodes by using its multiple channel advantage. For EDMSP, it offers an easy to use GUI to help users manage and save energy data from the sensor node.

Real world measurements are used for both energy model and operation mechanism calibrations on the sensor node. The energy consumption performances on Telos based network, MICA based network and iHop@Node based network are evaluated and simulated under different scenarios: varying traffic loads, random sensing start time, different packet control overhead and different output powers under two channel models. Energy consumption performances of unslotted CSMA/CA based network are highlighted and evaluated under the high data rate scenarios (2 Mbps and 1 Mbps provided by iHop@Node) and comparisons are also made between this high data rate scenario and the commonly adopted low data rate scenario (e.g., 250 Kbps for Telos and MICAz, 38.4 Kbps for MICA2). In addition, the energy consumption of ESB and SB based networks are studied in terms of several aspects and then their energy performances are also compared with unslotted protocol based network.

The second contribution lies in the introduction and design of an online and genetic algorithm based optimization framework for WSNs, named as iMASKO (**i**NL@**M**ATLAB Genetic Algorithm based **S**ensor Networ**K** **O**ptimizer). iMASKO uses MATLAB to implement its optimization evaluation engine and it integrates iWEEP_SW as the simulation engine to generate performance metrics. As a generic optimization

framework, the simulation engine can be replaced by other simulation tools (e.g., NS-2 [6], OMNeT++ [7]) or even by mathematical analysis models. The use of iMASKO can help users find the optimal parameter set for protocols out of hundreds of potential solutions in a very short time without exhaustive simulations on all the possible scenarios. By employing a weighted sum cost function, iMASKO is able to evaluate the energy consumption, without undermining other metrics to achieve a tradeoff solution. When a multiple objective genetic algorithm is used, Pareto based results [22] generated by iMASKO can be applied to the problems where multi-scenario and multi-objective conditions are required to be taken into consideration.

1.4 Thesis Outline

The rest of the thesis is organized as follows:

Chapter 2 presents a state of the art of wireless sensor networks with respect to architecture, application scenarios, origins of energy consumption, energy consumption minimization strategies, potential energy conservation offered by the operating system as well as tools for the energy consumption of wireless sensor networks.

Chapter 3 describes the design of iWEEP_SW and iWEEP_HW. Detailed modeling implementations, design flow and framework architecture of iWEEP_SW as well as prototype building and measurement results in iWEEP_HW are explained in detail.

Chapter 4 investigates the energy consumption performance of different sensor motes under various case scenarios. Relevant energy results are presented and compared. Energy conservation strategies are given after each scenario.

Chapter 5 presents the design and implementation of iMASKO framework. With the fast and intelligent search ability, iMASKO is developed to integrate iWEEP_SW for optimization purposes. Two case studies are used to validate the usability and efficiency of iMASKO.

Chapter 6 concludes the thesis and discusses the possible future works.

Chapter 2

State of the Art

Contents

Wireless sensor networks (WSNs) consist of a large number of interconnected distributed sensor nodes to monitor and measure physical and environment conditions, such as humidity, temperature, motion, light, etc. The measured data can be sent to different terminals or end users via a wireless network. Compared to conventional networks, WSNs are small in size, low in power consumption and have been widely deployed to many applications. They also face constrained resource challenges in terms of storage, communication range, processing ability and energy. The potential uses of WSNs have been limited in particular due to the lack of energy, since sensor nodes in many application scenarios are required to run several months or years without human interference (e.g., battery changing or recharging), so increasingly energy conservation methods are

being developed with the aim of achieving increasingly energy-efficient networks. Various evaluation tools (simulators, emulators and testbeds) are available to facilitate and speed up energy consumption evaluation before real deployment.

In this chapter, the background and state of the art of WSNs are investigated. This chapter is organized as follows. Section 2.1 describes typical WSN architectures from the hardware platform and to the protocol stack. Five types of applications are introduced in section 2.2. Section 2.3 presents some challenges in the WSN design process. Section 2.4 analyzes the energy consumption both at the node hardware level and at the network level. Section 2.5 provides a survey on some typical and emerging energy conservation methods and energy-efficient MAC protocols are emphasized. Section 2.6 discusses the energy-saving possibility from the perspective of WSN operating systems. In section 2.7, evaluation tools that include simulator, emulator, testbed are carefully studied on their capabilities in the evaluation of energy consumption, and then a mixed energy evaluation approach that combines testbed measurements and simulation/emulation is highlighted. Finally, this chapter is concluded in section 2.8.

2.1 Architecture of Wireless Sensor Networks

Wireless sensor networks (WSNs) consist of a large number of distributed sensor nodes to monitor/measure environmental/physical conditions such as temperature, sound, humidity, vibration, pressure, etc. These measured data are then digitized and transferred over the network, and eventually to end users who are allowed to monitor the entire network and carry out some necessary management. Fig.2.1 shows a typical architecture of WSNs.

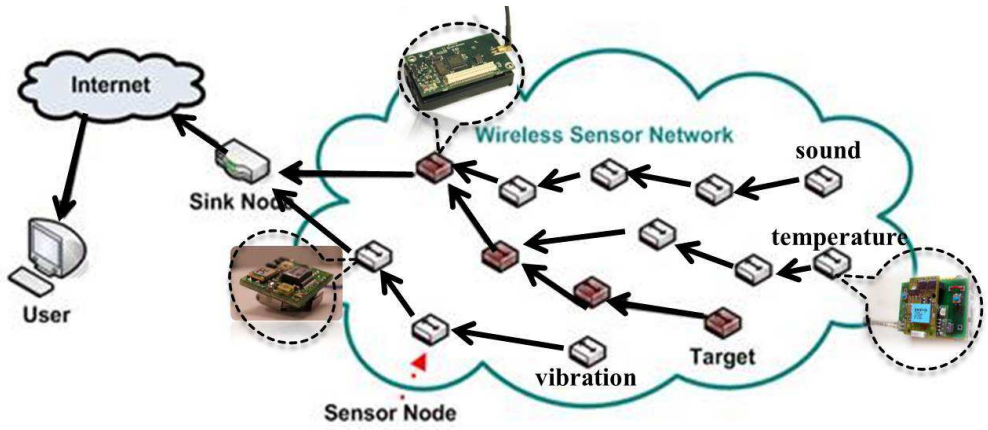

Figure 2.1: Typical architecture of WSNs

2.1.1 Hardware Architecture of Sensor Node

Since the sensor node is the core component to form WSNs, a general hardware architecture of it is presented in Fig.2.2. The architecture can be divided into four parts: processing unit, communication unit, sensing unit and power supply unit.

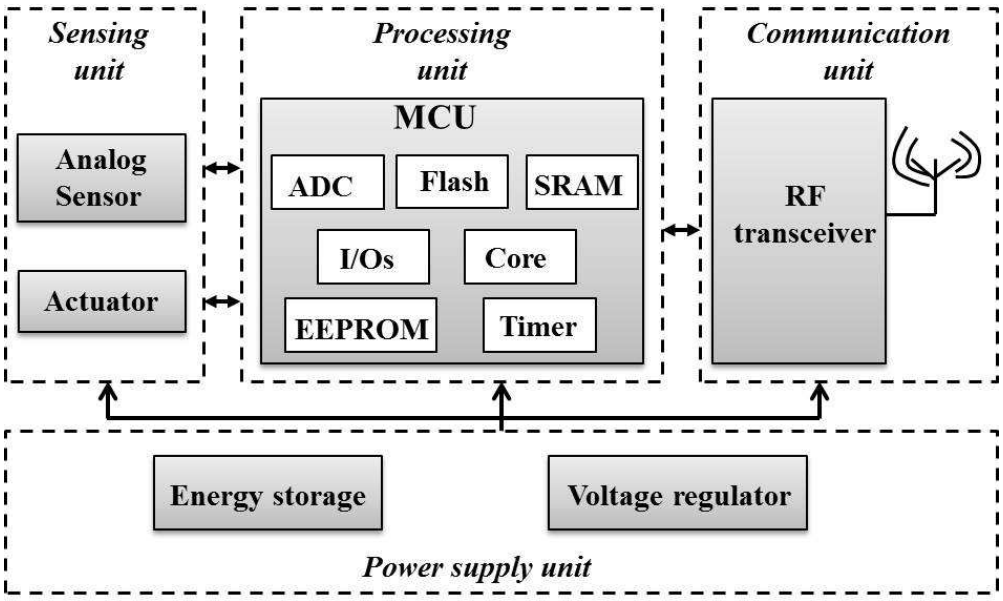

Figure 2.2: Sensor node hardware architecture

Usually, the sensing unit is made up of various kinds of sensors to measure a physical quantity and convert it into an electrical signal that can be read by the processing unit. The analogue signals produced by the sensing unit are converted to digital signals by an analogue to digital converter (ADC) in the processing unit, and further data processing can be done if necessary (e.g., data encoding, data compression). Besides, the processing unit is also responsible for the task management of the node. After that, the communication unit enables wireless communication between different nodes. The power supply unit provides the supply voltage for the whole sensor node to guarantee the running of all on-node components.

Currently, many commercial-off-the-shelf (COTS) sensor nodes offer feasible options. These nodes have already been widely used by many academic institutions and industrial departments. Some well-known and typical nodes are the MICA mote and Telos mote series. MICA2 [17] was designed by UC Berkeley and released by Crossbow Tech Inc. in 2002. The MICA2 node can be used for several different sensors (e.g., temperature, light, magnetisme, acoustics, etc), and Atmel ATmega128L MCU [23] is used to connect sensors and transceivers. The CC1000 transceiver [24] supports the maximum data rate of 76.8 Kbps and the default data rate of MICA2 is selected as 38.4 Kbps. As an improved version of MICA2, MICAz [18] replaces the CC1000 radio with CC2420 [25] which is an IEEE 802.15.4 [20] compatible transceiver and supports the maximum data rate of 250 Kbps. Compared with the MICA series, Telos [26] is a newly developed mote released in 2004. It still uses CC2420 radio, but replaces ATmega128L MCU with a very low-power MSP430 MCU. Other motes that share the same structure of Telos include TelosB [14], Tmote Sky [15] and Shimmer [16]. In addition to these low-power based platforms, some high-power based platforms with more powerful MCUs can be used for data processing and data routing tasks in WSNs. Examples are Stargate [27] and μAMPS [28].

2.1.2 Protocol Stack

To build an efficient wireless sensor network environment that can automatically handle various kinds of given tasks, the design of appropriate communication protocols plays a vital role. The development of the protocol stack aims to facilitate the implementation of protocol design. The widely accepted protocol stack architecture for WSNs is from [29] and presented in the following Fig.2.3.

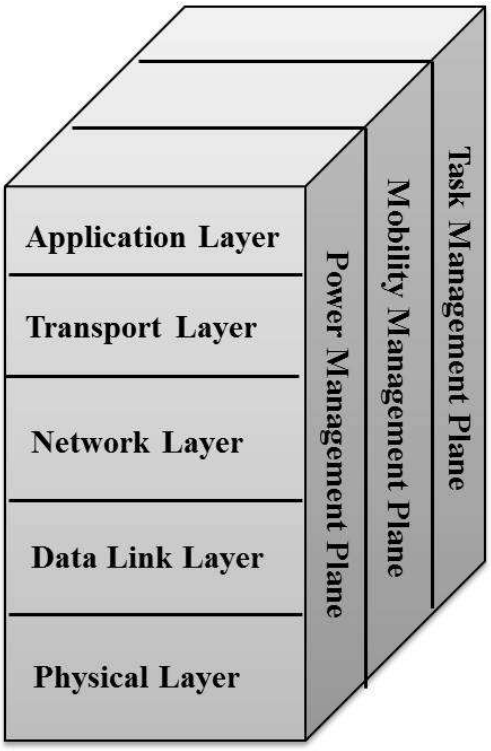

Figure 2.3: WSNs protocol stack [29]

This protocol stack is composed of five layers and three management planes. From a top-down order, the application layer provides an interface to receive user data and allows different types of application software to run on it. The transport layer can be used for data stream/flow management if necessary. Different routing mechanisms are implemented in the network layer to address the data supplied by the transport layer. The data link layer contains many medium access control (MAC) protocols to handle packet corruption and collision problems with the goals of minimizing energy consumption, reducing packet delay and improving packet transmission reliability. The physical layer offers various techniques such as modulation, transmission, reception as well as encoding/decoding for wireless communication. On the other hand, the uses of the three management planes are able to lower energy consumption, adapt the mobile network and share resources between different sensor nodes. Since these management services can take place through all five layers, their implementation and optimization on a cross layer could be considered.

2.2 Typical Applications of WSNs

The increasing interest in wireless sensor networks in recent years has greatly promoted their uses into diverse fields. The understanding of these application scenarios and their specific requirements/characteristics is important to the design of wireless sensor networks. An overview of five types of typical WSNs applications is presented in this section.

2.2.1 Medical and Health Care

One of the emerging applications of WSNs is their use in the healthcare domain. With advances in miniaturized electronic devices, leading to reduced sizes of battery, data processing, sensing and wireless communication technologies - each sensor mote in the network can be designed to be worn on the human body or implanted into the human body for the measurement and monitoring of corresponding physiological signals. Due to the features of small dimensions and light weight, the patients can be monitored with such devices whether in the hospital or at home. They can move freely without interfering with the daily routine, and the collected data can be automatically transmitted to medical staff to help them examine the status of patients. Numerous diseases have benefited from such a monitoring process.

The *Mercury* wearable system [1], designed by researchers at Harvard University, was aimed to monitor and analyze the motion performance of patients suffering from Parkinson's Disease, epilepsy and strokes. The *Mercury* network system consists of 8 low power based Shimmer nodes [16]. Each node is attached to a different part of the human body, and samples the movement signals associated with the diseases at 100Hz with a resolution of 16 bits. The collected data is stored in the local 2GB MicroSD flash memory. By tuning the operations of the sensor, the lab-based experimental measurements show that *Mercury* can achieve high data quality and long lifetime in a clinical study where the wearable sensor node can remain functional for about 12∼18 hours a day compared to a few hours with the previous approach.

In [30], implantable glucose sensors were placed in the subcutaneous tissue of the abdomen of 15 adult patients to continuously monitor their glucose levels, in the hope of finding appropriate solutions to reduce hypoglycemia. In the clinical experiment, glucose levels were measured every 30 seconds and the radio transmission of the glucose data to the receiver occurred every 5 minutes. The final experimental results showed that this system could satisfy the requirement of real-time continuous blood glucose monitoring, and simultaneously helps to reduce hyperglycemia excursions with type 1 diabetes. Besides, the use of WSNs in such continuous and prolonged healthcare monitoring can also found in fields such as hypertension monitoring, asthma monitoring as well as Alzheimer's disease monitoring.

2.2.2 Environment and Ecological Monitoring

In 2005, a volcano monitoring project [31] conducted by Werner-Allen et al. deployed a TMote Sky node [15] based network on Volcán Reventador in northern Ecuador to collect the seismic and acoustic data of the active volcanos. The network consists of 16 TMote Sky nodes with an average distance of 200∼400m between each adjacent node. The network was maintained by multihop data relay from the source node to the gateway node for data collecting and then these data were transmitted to a laptop at the base station

for storage and later evaluation. After over three weeks of field experiments, scientists successfully captured 230 volcanic related events.

The use of WSNs in a flood warning system was proposed by researchers at Lancaster University [32]. They designed and implemented a powerful *GridStix* sensor platform [32] which consisted of a 400MHz Intel PXA225 CPU, 64Mb of RAM, 16Mb of flash memory, radio modules with Bluetooth, IEEE 802.11b and GPRS for flood monitoring and prediction in UK. After lab measurements on energy consumption and reliability, this flood system has been used for the real-world experimental testing, which was deployed on the River Ribble in the Yorkshire Dales to cover approximately 1km of the river with the initial installation of 13 nodes.

The well-known ZebraNet [33] was adopted to study the behavior of zebras by attaching special GPS-equipped collars (nodes) to them. These nodes can track the animals across a large wild area via a peer-to-peer network to deliver the logged data back to researchers. A 30-node ZebraNet system according to the researchers was planned to be deployed at the Mpala Research Center in central Kenya. Likewise, for the study of animal behavior, the Great Duck Island project [34] was aimed at monitoring sea bird nesting behavior.

In agriculture, an image supported wireless sensor network was deployed for vineyard monitoring [2], where the data are collected and analyzed to provide a suitable diagnosis for the plants such as the choice of insecticide and fertilizer.

2.2.3 Home and Building Automation

With their small dimensions and increasing powerful data processing ability, wireless sensor nodes can be easily integrated into various home appliances or installed in smart buildings for control and monitoring tasks. For instance, a wireless sensor network based *iPower* energy conservation system [3] was designed to be used in intelligent buildings and was aimed at automatically adjusting and selectively turning off electric appliances to meet energy saving requirements. MICAz motes [18] were employed to collect sensing data from surroundings which including light, sound and temperature. One transmitter in this system was connected to a control server via an RS-232 interface to send specific commands to each independent node (with a unique address for each node). The system was evaluated by a smart desk scenario and proved that the use of the *iPower* system was able to save approximate 16.5%∼46.9% energy.

Besides, due to the utilization and deployment of precise climate sensors, air quality sensors and occupation sensors, the headquarters of the New York Times is regarded as a smart building and consumes 30% less energy than traditional office skyscrapers, while at the same time increases user comfort [35].

2.2.4 Industrial Applications

Industrial wireless sensor networks, also known as IWSNs, play a vital role in fields such as industrial monitoring, industrial control and automation. The objective of the *RealFusion* project [4] is to use wireless sensor networks in a real-life industrial environment for factory silos monitoring and factory storage monitoring. The observed data results can help control and adjust the temperature of the silo tower and the warehouse.

In vehicle automation [36], by integrating flow sensors, tire pressure sensors, light sensors, acceleration sensors onto different MICAz motes, and by placing these motes in different parts of the automobile to collect corresponding data, engineers and researchers are able to analyze the final acquired data from multiple angles and improve the automobile's interior comfort for the drivers. In addition, IWSNs also find their ways into chemical plants inventory management, pulp and paper mills as well as oil refineries [37].

2.2.5 Military Applications

Due to their rapid deployment and cost-effective characteristics, wireless sensor networks play a significant part in modern military engagement. Their applications cover a wide range, including tracking and targeting of friendly/enemy forces, battlefield monitoring and event localization.

In [5], Milica et al. presented some of the most recent advances of using wireless sensor networks in defense applications. For example, self-healing land mines are networked mines which can automatically rearrange themselves to ensure optimal coverage. Chemical, biological, and explosive vapor detection systems can measure and predict the threat of explosive, or toxic chemical and biological materials to warn and protect soldiers. An optical sensor system is used for collecting and storing vibration, shock, temperature, and damage events over the entire lifetime of a missile canister. For future military-based WSNs, sensor nodes with features of miniaturization, robustness, security and wide coverage are expected.

2.3 Design Challenges of WSNs

Constrained by the small dimensions and battery-driven features, the biggest problem of deployment of WSN frequently emphasized by researchers is the limited energy supply. The main design criterion is to reduce energy consumption for longer lifetime without compromising other application-specific metrics of the sensor network. The design of WSNs needs to find tradeoff solution for the following challenges.

Quality-of-Service (QoS): Despite variation in QoS requirements and specifications between different applications, the two main metrics in QoS are reliability and latency. Usually, the rate of successful data exchange between source nodes and the sink node must be above a certain level to guarantee the reliability and functionality of the network. The reliability can be further maximized but it may be at the cost of increasing energy consumption. Hence, a tradeoff is necessary. When comes to latency, for some kinds of applications the reception of the sensed data at the sink node is strictly timed, so long latency data may cause timeout problems and lead to wrong decisions especially in industrial monitoring applications.

Security and Privacy: Since the deployment of sensor nodes is sometimes in accessible areas, they risk encountering technical interference or intrusion by humans, leading to security privacy issues. Thus, there is a need to design robust algorithms for data encryption, authentication mechanisms for privacy protection and secure routing for data relay, in order to protects the entire network from various attacks such as passive attacks, active attacks and external denial-of-service (DoS) attacks [38].

Resource limitation: Constrained by the limited physical size, it is difficult to improve the energy supply and memory storage. The data processing capability is bottlenecked by the use of 8-bit and 16-bit microcontrollers. The short communication range and narrow coverage are unavoidable because of the low output transmission power. For example, MICA2 runs with 4∼8MHz clock rate and has an 8-bit based AVR

RISC-based microcontroller (ATmega128), 4KB of RAM, 128KB flash memory and ideally 916MHz of radio communication frequency. In addition, cost effectiveness is a key factor to be considered, since developers always strive to spend less money for the best optimal configurations.

Adaptability: The design of WSNs must be adaptive and flexible enough to be deployed in a wide range of application scenarios. The whole network should be maintained functional no matter whether the number of nodes is from hundreds to thousands or just a few. Due to the mobility of sensor nodes and observed events as well as the possibility of malfunctioning sensor nodes within the network, the overall topology of the network changes. Hence the design of WSNs must be intelligent and strong enough to cope with these dynamic topology scenarios.

Energy: One of the most serious concerns, the lifespan of the sensor node is determined completely by the amount of energy available and the rate of energy consumption (i.e. the average power consumption). Since the improvement of the energy capacity of small dimension batteries still requires much effort, the objective of a long lifetime is difficult to achieve by increasing the amount of energy available. Therefore, the design of energy-efficient MAC protocols, communication strategies, operating systems and energy saving routing mechanisms provide effective ways to extend the lifetime of the node. The use of energy harvesting technologies for sensor node power supplies also offers an alternative in recent years.

2.4 Energy Consumption in Sensor Networks

As sensor nodes are generally battery-operated devices and the scale of operation moves from laboratory testbeds to practical deployments, it is more important than ever to conserve energy to fulfil given tasks. This is especially valid in continuous monitoring scenarios where the (frequent) changing and recharging of the battery could interrupt and invalidate the task, while other application scenarios might require the sensor nodes to run for several months or years in hostile and unreachable places where the changing and recharging of batteries is impossible. In any cases, energy is a significant resource and must be used parsimoniously to extend the lifetime of the sensor node and network. In this section, the energy consumption of sensor networks is analyzed from the perspectives of the node and network (MAC layer) levels. We will then introduce some recent and emerging energy conservation techniques.

2.4.1 Energy Consumption at Node-Level

Since sensor nodes are component based, the node level energy consumption of commonly used hardware components in the sensor node are under investigation, i.e. microcontroller, radio chip (transceiver), memory and sensors.

Microcontroller (MCU): In each sensor node, the MCU consumes energy for data processing and control tasks. Basically speaking, multiple working states are implemented in current MCUs for energy saving purposes. For instance, the MSP430 MCU from TI, used in TelosB [39], features five low-power modes (LPM) and one full active mode. The active mode consumes about 6.75 mW (all power values given at 4 MHz and 3 V). Other five LPMs are LPM1 (CPU disabled), LPM2 (the loop control for the fast clock is disabled), LPM3 (the fast clock is disabled), LPM4 (DCO oscillator and its DC generator are also disabled) and LPM5 (crystal oscillator disabled) which consume 0.54 mW, 0.54 mW, 0.12 mW, 0.06 mW and 0.045

mW respectively (measurement results). In comparison, the ATmega128L in MICAz [18] consumes about 24 mW in active mode and 0.045 mW in sleep mode. One of the most popular energy saving methods used in MCUs is Dynamic Voltage Scaling (DVS) [40] which can dynamically adapt the supply voltage value to guarantee processor task performance while avoiding unnecessary power consumption. More power information of different MCUs can be derived from the current consumption figures shown in Tables 3.1 and 3.6.

Radio (Transceiver): A radio transceiver is responsible for over-the-air packet transmission and reception. Due to the complexity of wireless communication, the transceiver needs to contain many functional modules (modules for modulation, de-modulation, frequency synthesis, frequency conversion, filtering as well as other functions are included) and will consume a significant fraction of the total energy in a sensor node. The simplest energy saving strategy is to turn off the transceiver as much as possible and to turn it on only when necessary. The typical power consumption of the popular CC2420 transceiver [25] is 52.2 mW in transmission (TX) mode, 56.4 mW in reception (RX) mode and 3 μW in sleep mode. As an enhanced transceiver integrated with the IEEE 802.15.4 standard, the MRF24J40 [41] consumes more energy in all three states: 69 mW in TX mode, 57 mW in RX mode and 6 μW in sleep. The power consumption of some other transceivers can be found in Table 3.2.

Memory: With larger memory size than on-chip memory, off-chip RAM is usually used in sensor node for the data storage, and the energy consumed by related operations on the off-chip RAM cannot be ignored. Taking TelosB mote [39] as an example, measurements show that the three most relevant operations on flash memory are erase, write and read with energy consumption figures of 6.9 mW, 5.7 mW and 7.2 mW respectively. These figures are very close to the energy performance when MSP430 operates in active mode. Therefore, the energy used in these memory related operations cannot be considered to be a minor contribution to the overall energy consumption.

Sensor: The general belief is that sensing contributes very little to the overall energy consumption of the sensor node. This is mainly true for passive, low-quality and low sample rate sensors such as those used to sense light and temperature. However, other power hungry sensors, like CCDs/CMOS image sensors, high-rate high-resolution A/D converters and active sensors (e.g., sonar, radar, laser rangers), demonstrate energy consumption levels as high as that of the radio transceiver. Hence, the energy consumption of the sensing subsystem cannot be ignored.

2.4.2 Energy Consumption at Network-Level (MAC Level)

In WSNs, a group of sensor nodes are usually distributed in a specific area for various applications, such that fierce channel competition could occur and cause unnecessary energy consumption if no network strategies are applied. The use of the MAC layer aims to provide a reliable and efficient network condition for WSNs to reduce energy consumption, but some energy waste issues still exist in MAC and are listed as follows [42] [43]:

Collision: Collision occurs when two nodes transmit packets at the same time and interface with each other's transmission. Thus, reception energy is wasted at the destination node while transmission energy is wasted at the source node. Further energy might be consumed with packet retransmission.

Idle listening: A sensor node in idle listening mode is ready to receive possible incoming packets, but

the maintenance of the reception state is very costly, especially in low load networks. A periodical schedule for turning on receivers for incoming packets could be an energy saving method. However, such a period should be kept at a reasonable level to reduce the energy cost of changing modes.

Overhearing: With the broadcast feature of the wireless medium, a sensor node can waste energy in receiving packets that are not destined for it.

Overmitting: Energy can be wasted on the source node if it sends a packet to a destination node that is not ready for reception. Furthermore, since the ACK frame will not be sent by the destination node, more energy will be used for data packet retransmission.

Protocol overhead: To guarantee protocol operation, some control packets and mechanisms are employed, and the maintenance of these protocol overheads must consume large amounts of energy. Hence, an optimized method should be found to minimize the energy consumed by these overheads while avoiding any sacrifice in network performance.

2.5 Methods of Energy Saving

In this section, surveys of several recent and emerging energy conservation schemes that can address and alleviate energy consumption challenges are presented.

2.5.1 Hierarchical Sensor Node Architecture

Since it is the sensor node hardware that consumes energy, the design of an energy-efficient node platform is the basis for high-level energy conservation methods (e.g., protocols, strategies, software). Therefore, a hierarchical node architecture is proposed in [44]. Based on a surveillance application scenario, a two tiered architecture is used and shown in Fig.2.4. In the periodic 'low workload' based intruder detection phase, a low end microcontroller (e.g., ATmega128L in MICA2/z) is used for two points: (i) ultra-low power sleep mode, since the node spends most of the time in sleep mode during this phase, and (ii) rapid wakeup time to reduce the overhead in the frequent sleep-active operation. A high end embedded processor (e.g., Intel PXA-255 in Stargate [27]) is used after the presence of the intruder is detected. In this 'high workload' phase, the processor stays in active mode to perform computation and communication for intruder tracking and tracing. A high-end embedded processor is much more energy-efficient in this computation-based process than the low-end microcontroller (e.g., to execute a CRC-32 computation, the Stargate mote needs 13 mJ while MICA2 needs 367 mJ [44]). Therefore, this heterogeneous and hierarchical based architecture design provides a tradeoff in energy conversation.

2.5.2 Energy Harvesting

Due to the fact that the amount of energy stored in the battery is limited, the lifetime of the sensor node / network is also limited. Instead of putting efforts into the hardware platform or into the software that runs on it, researchers have increasingly focused on the energy supply system by using energy harvesting techniques to extract electrical energy from existing environmental sources such as solar light, thermal, vibration, motion, etc. As an emerging technology, the use of such techniques in WSNs has become popular

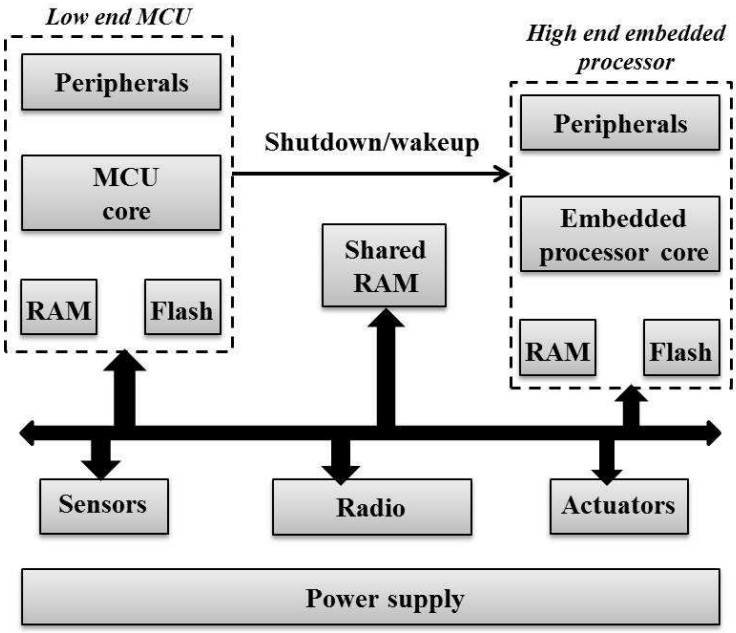

Figure 2.4: Sensor node architecture with two tiered processors [44]

in recent years, since they can help minimize or even eliminate battery dependency for individual sensor nodes. One of the major challenges is the energy storage problem for future use. Considering a solar light driven platform, extra energy needs to be stored to maintain the operation at night or in low light conditions. Thus, a better lifetime can be expected if an optimized energy saving strategy can be applied for night operation (e.g., lower duty cycle compared to daytime). A solar energy harvesting based sensor node named Heliomote is developed at UCLA and is shown in Fig.2.5.

Figure 2.5: Heliomote: a solar energy harvesting sensor node [44]

2.5.3 Sampling Related

With the rise of emerging sensing-constrained applications, the use of various power hungry sensors (section 2.4.1) can lead to high energy consumption. Therefore, energy conservation schemes need to reduce the

energy spent in acquiring sensed data by reducing the time spent in acquisition (correlated to sensed data sampling rate), which can also simultaneously reduce energy consumption in communication. Three types of methods presented in [44] [45] are described as follows:

Adaptive sampling: As the sensed sample data correlation can be found in some application scenarios, adaptive sampling can exploit correlations to reduce the number of acquisitions. This saves a considerable amount of energy without losing fidelity.

Dynamic hierarchical sampling: According to application-specific requirements, this sampling approach can dynamically select the most suitable sensor for a given task in different case scenarios based on the tradeoff between accuracy and energy-efficiency.

Model-based active sampling: This technique can reduce sampling times by using data prediction. A model of the sensed phenomenon is designed and built upon sample values, so that the model can use the anticipated data to measure the accuracy of the current sensed data and evaluate whether a new measurement is worth the effort without consuming extra energy [44].

2.5.4 In-Network Processing

When sensor nodes are distributed and collaborate in a network, in-network processing is responsible for such tasks as packet passing, application program execution and data processing to make energy-efficient network conditions. Several techniques are classified by [45] [46]:

Data aggregation: Data aggregation is performed in some applications that are only interested in average, maximum or minimum values. In such cases, the sensor nodes do not have to transport all the sensed data, since the sampled data generated in a period of time can be aggregated by the node for some necessary processing. Finally, only the required data is transported and a large amount of energy can be saved from the reduction of communication.

Data compression/processing: This approach is a conventional method that also aims to reduce data in the communication process. However, sometimes the computational efforts in data compression/processing before over-the-air transportation might be costly for a resource-limited sensor node microcontroller, which in turn can cause an increase in overall energy consumption. Thus, it is crucial to find different tradeoffs between local computation and wireless communication.

Mobile networking: In addition to a static network scenario, mobile elements in WSNs provide a new perspective for energy conservation. In a multi-hop based network, some paths could be more loaded than others depending on the route selection, and the nodes closer to the sink could suffer premature energy depletion because of the heavy relay tasks [47]. Assuming that nodes on the loaded path and nodes with heavy relay tasks are mobile, the traffic flow/load on these nodes can be altered and the energy consumption is dispersed throughout different sensor nodes. Hence, premature energy depletion can be mitigated to some extent and the network topology is able to be maintained for a longer time, so that an extended network lifetime is expected.

2.5.5 MAC Protocols

The design of Medium Access Control (MAC) protocols plays a significant role in achieving an energy-efficient network, since it can directly control wireless communication which is known to be the most energy consuming part in the sensor node [12]. In the following discussion three types of MAC protocols are surveyed mainly on their energy management and energy conservation issues rather than channel access mechanism. The taxonomy of these three types of MAC protocols are schedule-based, contention-based and hybrid-based [45].

2.5.5.1 Schedule-based

The operation of this type of protocol is time-scheduled so that each working state of the sensor node can be accurately computed to avoid unnecessary collision, overhearing and hidden node problems for energy saving. Usually, a TDMA scheme [42] is employed, where time is periodically divided into a certain number of time slots, and each node is assigned to one or more slots, in which a node can perform different channel access tasks (such as transmission, reception, listening) based on the schedule algorithm. Sometimes, such time slots can be dynamically assigned by the selected cluster-head in the multi-hop network scenario.

One well-known TDMA-scheduled protocol is TRAMA [48]. In TRAMA, time is divided into two parts, a random-access period and a schedule-access period. The random-access period is responsible for slot reservation and establishing two-hop neighborhood information where the channel access is contention-based. The schedule-access period is used to assign time slots to individual nodes aiming to provide a collision free schedule. After that, an adaptive election algorithm is started to decide the transmitting and receiving nodes in the current time slot based on the neighborhood and schedule information, while other unselected nodes are maintained in low power mode. Thus, energy consumption is reduced from both the collision free process and the low power maintenance strategy.

μ-MAC [49] is another schedule-based protocol similar to TRAMA which also has both contention and contention-free processes. Energy is saved from collisions and sleep-scheduled nodes, while at the same time reliability and packet latency can be at an acceptable level. However, the frequent use of a contention period could cause high energy consumption on the overhead, so energy-efficiency cannot be guaranteed if μ-MAC is used in a dynamic and frequently changing network environment.

DEE-MAC [50] is a cluster and TDMA based protocol that reduces energy consumption by using cluster head assigned slots to keep synchronization on the data transmission/reception schedule and force other idle nodes to sleep mode. In DEE-MAC, a periodic round is launched to select the cluster head based on the remaining power, and after successful selection the cluster head builds a TDMA schedule to assign data slots to all the nodes by broadcasting. Each node that has data to send or receive is awake during its data slot, otherwise it remains in sleep mode to save energy. On the other hand, the contention period in which nodes compete for cluster head could lead to overhead energy waste.

2.5.5.2 Contention-based

Contention-based protocols are the most commonly used MAC protocols in WSNs. They introduce random and flexible channel access mechanisms to reduce the energy consumption on collision.

B-MAC (Berkeley MAC) [51], one of the most famous contention-based MAC protocols, is designed based on CSMA mechanism [42] and especially for low power WSNs. In order to reduce energy consumption, B-MAC provides clear channel assessment (CCA), CSMA backoff scheme and optional ACK for the transmitter to effectively handle network conditions. At the receiver, a low power listening (LPL) mechanism is used to achieve a low duty cycle, so sensor nodes wake up periodically to check channel activities. If activity is detected, the sensor node can be awake for a specific time for the coming packet and then goes back to sleep. If no packet is received, then after a timeout the node can also return to sleep mode for energy conversation. To guarantee the reliability of data receiving, the B-MAC data packet consists of a long preamble and a payload. It is required that the preamble duration is bigger than the check interval so that the node can always detect ongoing channel activities. The long preamble frame can save synchronization costs, which in turn can increase energy consumption on packet transmission as well as the packet latency.

S-MAC (Sensor-MAC) [52] is another well-known contention-based protocol. It is based on a periodic duty cycle schedule that forces nodes to sleep instead of idle listening to conserve energy. Such a schedule is established among nodes and makes them organize in the form of a virtual cluster by means of exchanging *SYNC* packets. In S-MAC, the channel access process is divided into two parts: in the first time period (listen period) nodes exchange *SYNC* packets and some control packets to build a connection scheme for further collision avoidance. Real data transfer between correlated nodes takes place during the second time period (for other irrelevant nodes this period is a sleep period). Following the already built scheme, the sink node and related source node are able to wake up for simultaneous communication, while other source nodes can sleep until the next listening period. The basic operation scheme is briefly presented in Fig.2.6.

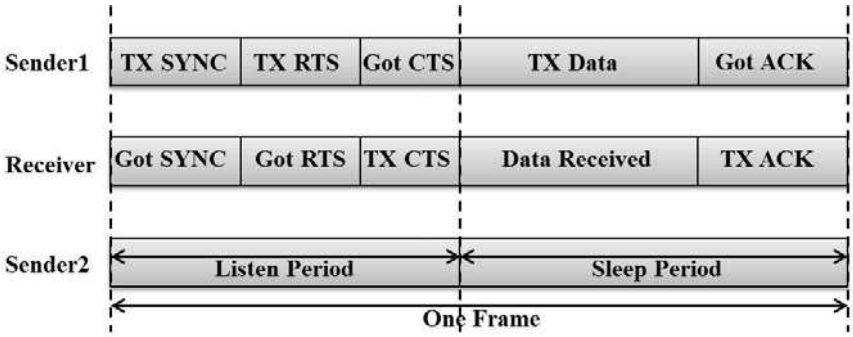

Figure 2.6: S-MAC operation mechanism

With the feature of message fragmentation, S-MAC can divide a long message into small frames and send it in a burst, which mitigates higher latency and storage problems. Besides, efforts should be done to make sure that two neighborhood nodes cannot be involved in two different schedules (two virtual clusters), which could cause unnecessary energy consumption in idle listening and overhearing [53].

T-MAC (Timeout MAC) [54] is proposed as an enhanced version of S-MAC. Since the parameters of S-MAC (such as listen and sleep periods) are constant and cannot be changed after deployment, it is not suitable for variable traffic load, which is common in WSNs. In addition, in networks without any traffic, S-MAC still has to wake up periodically and this causes significant energy waste. Hence, the design of T-MAC is to offer a dynamic and configurable duty cycle by modifying the listen-sleep period to improve

the S-MAC's poor performance on energy consumption and variable traffic support.

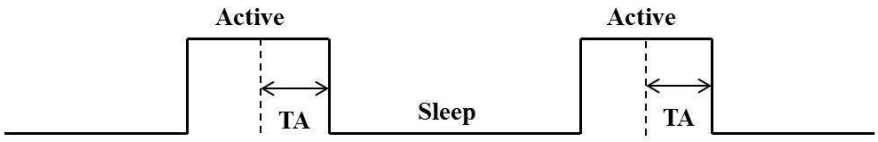

Figure 2.7: Dynamic duty cycle of T-MAC

In T-MAC, sensor nodes can automatically go to a sleep period even if no activation has occurred within a specific time period TA (shown in Fig.2.7). Energy conservation from more sleep nodes can therefore be achieved in a dynamic way. This dynamic duty cycle approach has also been used by DSMAC (Dynamic Sensor MAC) [55] and P-MAC (Pattern-MAC) [56] for energy-efficiency purposes.

2.5.5.3 Hybrid-based

The basic idea of hybrid-based MAC protocols is to achieve better energy performance by combining the advantages of schedule- and contention-based elements. The IEEE 802.15.4 standard [20] is one of the most interesting examples in this category.

The IEEE 802.15.4 standard is used for low-rate and low-power based Personal Area Networks (PANs), and defines three types of device:

- **PAN coordinator** is the core controller of PAN, and is responsible for the operation and management of the whole network.
- **Coordinators** collaborate with each other as well as with the PAN coordinator to maintain the connection and functionality of a subset of nodes in the network.
- **Nodes** can only communicate with (PAN) coordinators for data exchange without any network organization functions (e.g., routing).

With the above three devices, the IEEE 802.15.4 based network can be formed into a star topology, a cluster-tree topology as well as a mesh topology.

Two channel access methods are supported by IEEE 802.15.4 MAC layer, which are beacon enabled mode and non-beacon enabled mode.

In beacon enabled mode, a special duty cycle is scheduled by means of a superframe structure which is bounded by beacons. This superframe structure, shown in Fig.2.8, is periodically generated by the coordinator and broadcast to all nodes for synchronization purposes. Each superframe is composed of an active period and an inactive period. Communication between nodes and coordinators occurs during the active period. The active period can be further divided into a contention access period (CAP) and a contention free period (CFP). During CAP, nodes adopt a slotted CSMA/CA algorithm for channel access. During CFP, a certain number of guaranteed time slots (GTS) can be assigned to specific nodes so that the communication between nodes and coordinator cannot be interfered by any collisions in these GTS. During

the inactive period, nodes enter into a low power sleep state for energy saving and wake up at the beginning of the next superframe.

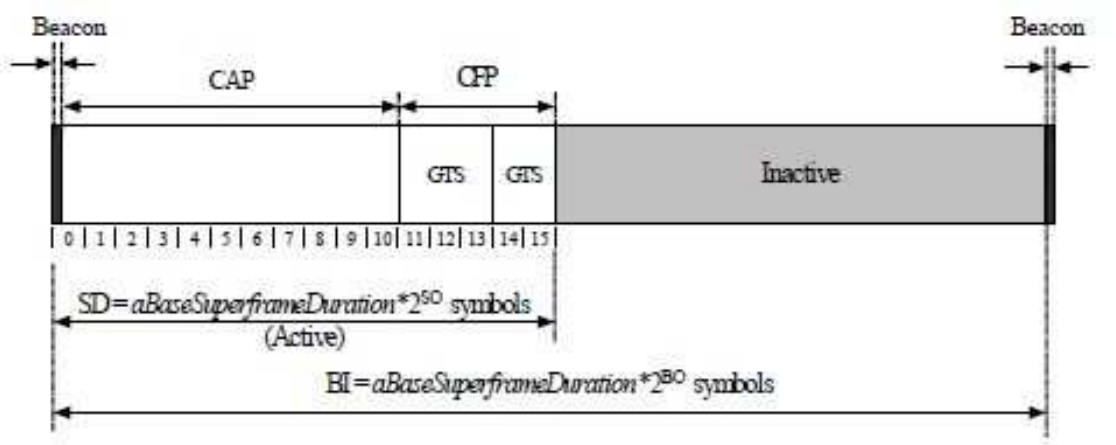

Figure 2.8: Superframe structure [20] (*aBaseSuperframeDuration* is fixed as 960 symbols and equals 15.36 ms under 250 Kbps data rate)

For the non-beacon enabled mode, this is totally contention-based, since the nodes are always in the active state and use an unslotted CSMA/CA algorithm for channel competition and data communication. Compared with the non-beacon enabled mode, the beacon enabled mode is more energy-efficient due to the low power consumption in the sleep schedule but at the cost of network latency and scalability. Thus, the tradeoff should be carefully selected based on specific application requirements. The operation of slotted and unslotted CSMA/CA algorithms is presented in Fig.2.9. More detailed description of these two algorithms can be found in section 3.3.2 or the standard document [20].

2.5.5.4 Discussion of MAC Protocols

In summary, schedule-based MAC protocols are inherently energy-efficient because the duty cycle and time slot assignment algorithm can make sure that wireless communication of each node only happens in its own allocated time slot in the active time period, while for the rest of the time the nodes are in the sleep state for energy conservation. WSNs are usually widely deployed and have a dynamic network topology, so schedule-based protocols, with various predefined parameters, lack the necessary flexibility and scalability to address the complexity of WSNs and hence are not commonly used in real WSNs in spite of the energy-saving features.

In comparison to schedule-based protocols, contention-based MAC protocols could cause higher energy consumption because of channel competition and collision. Nevertheless, their advantages in robustness, flexibility and scalability compensate the drawback in energy consumption and this type of protocols has been applied in widespread practical deployment and the challenge is how to make them more energy-efficient without sacrificing other performance metrics.

To combine both strengths of schedule- and contention- based protocols, hybrid protocols are very helpful in finding a tradeoff between energy and other performance. The disadvantage is that it could sometimes

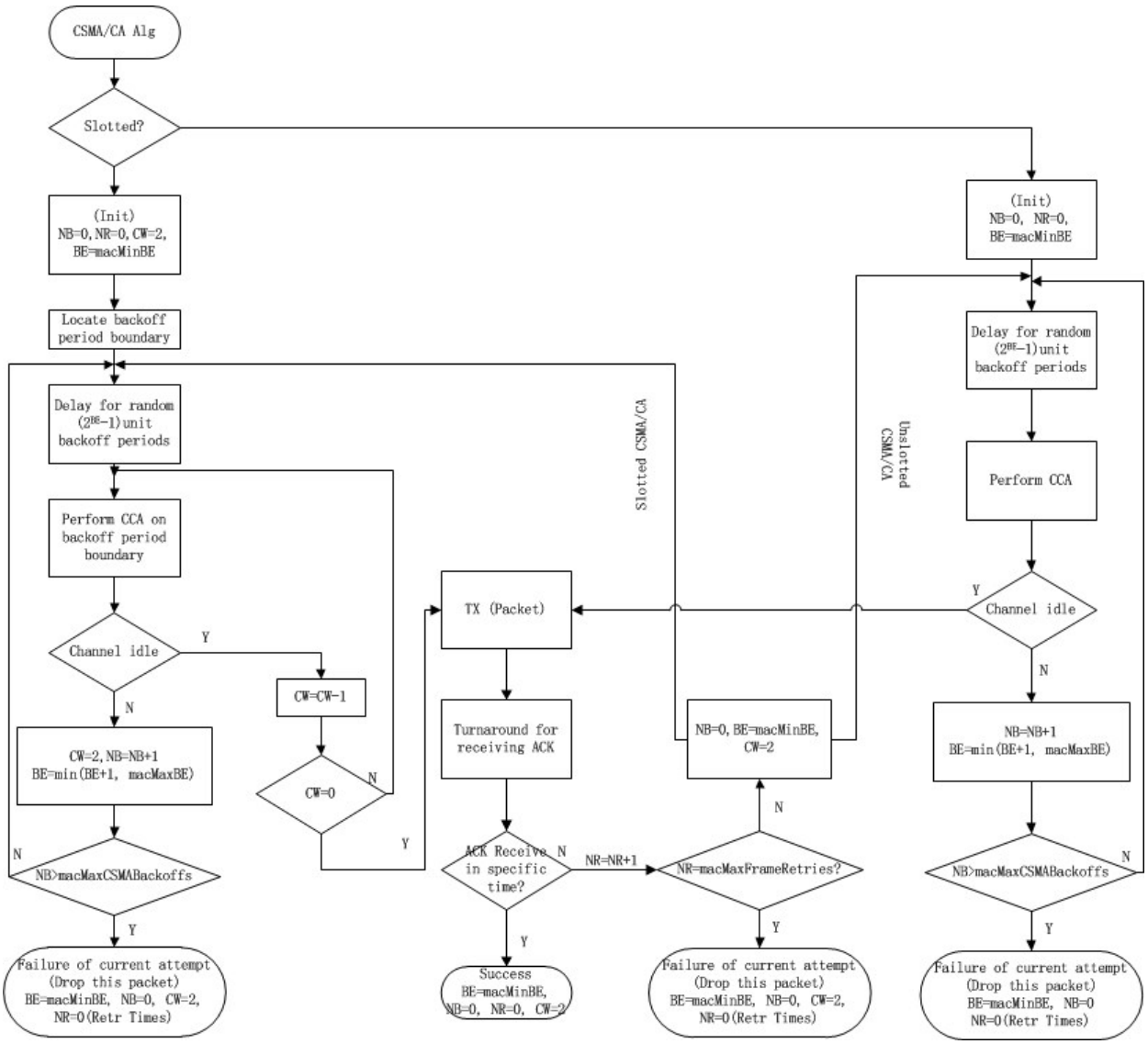

Figure 2.9: Slotted and unslotted CSMA/CA algorithm of IEEE 802.15.4

be too complex to be applied for a quick and large scale of deployment. The emerging IEEE 802.15.4 based networks provide a valuable research direction which might help a fast development of this kind of protocol.

2.6 WSNs Operating Systems Related to Potential Energy Saving

With wide application diversity and different types of sensor platforms, an operating system (OS) is needed to provide processor management, memory management, device management, scheduling, operation mechanism and other services for the sensor node. The OS aims to hide low-level hardware details and provide clear and friendly interfaces to facilitate users' application development process [57].

Since WSNs distinguish themselves from traditional wired and wireless networks, WSN OS needs to meet some specific requirements to satisfy the constraints of WSNs, which are listed as follows [58]:

Small in size: Since sensor nodes often have limited memory resources (from tens to hundreds of KB), the OS should be compact enough to run in the available memory.

Easy to program: The OS needs to provide reprogramming capabilities and a generic programming interface. Reprogramming is essential because the functionality performed by the sensor node can be updated even after deployment, and the generic programming interface may help access and control hardware more directly.

Efficient management mechanism: Microcontroller time and limited memory could be appropriately allocated and scheduled with the help of a resource management mechanism. Power management mechanisms can be used to handle sleep and wake up schedules for energy saving purposes.

Real time: Mentioned by many research works, this requirement is especially significant in industrial monitoring applications where outdated information is always rendered useless.

In summary, these requirements not only give an overview of WSN OS characteristics, but also provide good insight as to where optimization strategies could be. When compared to energy-efficient protocols and application strategies the OS itself does not have much direct impact on energy consumption of WSNs but it does exist and cannot be ignored. For instance, a stronger operating system with relatively large memory size, more services and more resource usage would certainly have higher energy consumption than a light operating system. However, a light operating system might not be capable of fulfilling the given tasks. Thus, a tradeoff is very important and some valuable information for such tradeoff solutions could be acquired from the description and optimization of some existing WSN OSs in the following context.

TinyOS: TinyOS [59] is the most well-known event-driven based operating system designed especially for wireless sensor network. It is written in nesC (network embedded systems C) programming language (an extension to the C programming language) and supports concurrent applications in a small memory that fits in 400 bytes. The component based design makes TinyOS easy to access various components by function calls and the component library includes network protocols, device divers and some services. The earlier versions of TinyOS were only event based without providing multithreading support until the release of recent 2.1 version. Despite the fact that these so called TOS threads are more familiar to developers, they might not be well suited to the resource constrained sensor node, and also add an additional burden on the developer to explicitly manage concurrency [60]. Thus, the development of a clear and light-weight based multithreading model is highlighted in WSN OSs. Other optimization challenges can be in the scheduling

and memory management. By using a FIFO scheduling algorithm, earlier versions of TinyOS are not suitable for real-time applications since the short task operations could be blocked by longer ones, while in [59], the authors claimed that by adopting an Earliest Deadline First (EDF) scheduling algorithm, the unfairness of waiting durations can be greatly improved. Without a dynamic memory management mechanism, the memory cannot be flexibly allocated or de-allocated at run-time. Finally, the direct application of existing C code to the development of TinyOS applications still needs some effort.

PicOS: PicOS [61] is another event-driven OS, is written in the C language for limited memory microcontrollers. PicOS operates as a finite state machine (FSM) and the state transition is triggered by events. The FSM can respond well to an event-driven based mechanism but does not show any advantages in real-time applications with a flat processing structure and limited resources.

Contiki: Contiki [62] is an open source WSN OS written in C. It combines the advantages of both events and multithreading but it is primarily event-driven. Multithreading is considered as an optional library implemented on top of the event-driven kernel. Asynchronous and synchronous events are supported in Contiki. Synchronous events are dispatched immediately while asynchronous event are scheduled for later. A polling mechanism is used to avoid race conditions and handle asynchronous events with high priority. All things in Contiki (e.g., communication, device driver, sensor data handling) are implemented in the form of services and each service has its own interface and implementation. With a dynamic memory management mechanism, it is sufficiently flexible to process memory related operations at run time. Due to the multiple types of services it provides (such as GUI, TCP/IP, IPv6, web browser and other services), when compared with TinyOS and PicOS, a typical Contiki configuration consumes about 2KB RAM and 40KB of ROM. Hence, the cutting and optimization of the OS kernel for a small size and at the same time satisfying application-specific requirements becomes a challenge.

MANTIS: MANTIS [63] is a lightweight multithreaded operating system, with a footprint of 500 bytes which includes kernel, thread scheduler and network stack. MANTIS is written in the standard C language, so its application development can be tested across multiple platforms. A thread table is maintained by the kernel to handle the data structure, and 10 bytes fixed memory is allocated to each thread table for storing thread-related information. The thread scheduler in MANTIS is priority based and is triggered by timer interrupts. In addition, MANTIS provides a remote shell and reprogramming ability to enhance users' control and inspection on the sensor node. Two of the major optimization directions are the reduction of the context switch time and stack memory overhead [64].

LiteOS: LiteOS [65] is also considered as a multithreaded WSN OS but with Unix-like abstractions to provide system and application developers with a familiar programming paradigm. Event handlers can also be registered in LiteOS by using a callback facility and dynamic memory allocation is also supported. The drawback is the scheduling algorithm provided by LiteOS, because once a process is scheduled it runs to completion, which may lead to the missing of a deadline for a higher priority process [60].

Nano-RK: Nano-RK [66] is an energy-aware real-time OS for WSNs. Compared to LiteOS, a rate monotonic scheduling is used in Nano-RK to guarantee that the job with a shorter period of time always has the higher priority, so that the deadlines of the given tasks can be satisfied. Besides, the implementation of a rate-harmonized scheduling in [67] is energy-efficient by eliminating CPU idle cycles. With a size of 2KB RAM and 18KB ROM, the optimization and cutting of the Nano-RK kernel can reserve more memory space

for real-time related purposes. Meanwhile, the capability of the entire OS can be improved if a dynamic memory management mechanism is integrated.

2.7 Evaluation Tools for Energy Exploration of WSNs

As described in section 2.4, due to the features of long-term operation without human interference, the issue of how to achieve energy-efficient WSNs has been frequently highlighted and emphasized in many recent works. Researchers and designers in the WSN community need to evaluate energy consumption efficiently and accurately for lifetime prediction. Otherwise, erroneous energy evaluation can cause energy waste since the sensor nodes might be rendered useless long before energy depletion. In addition to energy waste, network malfunction could be another potential issue, because the nodes might 'die' in advance of the expected lifespan.

However, measurement experiments on real deployments may suffer some limitations since the placement may involve hundreds and thousands of nodes, and are sometimes deployed in harsh and inaccessible environments. On the other hand, mathematical analytical/theoretical models might be idealized and unrealistic for real world scenarios especially considering the complexity of WSNs system. Therefore, energy results from simulation, emulation and testbed measurements provide an alternative on accuracy, time and cost. Besides, designers can also examine and evaluate their proposed algorithms, protocols, strategies and topologies with simulation/emulation tools as well as testbed devices. In this section, a survey of popular evaluation tools for WSN energy performance is presented from simulators, emulators and testbeds. After that a method requiring mixed simulation is highlighted in order to achieve the tradeoff between accuracy and efficiency for energy consumption evaluation of WSNs.

2.7.1 Simulation Tools

Some of the most popularly used WSN simulators are described in the following and characteristics of these simulators are compared in terms of several different aspects in addition to the energy evaluation ability.

NS-2 [6], the most popular and most widely used general-purpose network simulator, supports simulations for various kinds of wired and wireless networks, such as an IP based network protocol and a 802.11 standard based wireless network. An IEEE 802.15.4 model for sensor networks is proposed in [68]. NS-2 is a discrete-event and object-oriented simulator, written in C++ and OTcl (an object-oriented version of Tcl). OTcl is used as a script language to create the simulation environment and control and simulation processes. The trace file used for data storage is generated during the simulation and the results in the trace file can be graphically observed and analyzed by the Network Animator (NAM). With the excellent extensibility, NS-2 is maintained by a large number of researchers and many third party add-ons can be adopted to address some new problems. However, NS-2 still lacks appropriate application models, packet formats, sensing models, protocols and a configurable GUI [69], and the requirement of expert skills is especially unsuitable for beginners. Also, the exponential time and execution overhead make NS-2 perform not very well in terms of scalability for large sensor networks [70]. The energy consumption model in NS-2 is very simple, an initial amount of energy is given to each node, and after packet transmission and reception, corresponding energy

will be subtracted from the total energy until the later value reaches zero [6]. Two important and frequently mentioned NS-2 based WSN simulation environments are SensorSim [71] and Mannasim [72].

OMNeT++ [7] is another popular public source network simulator with the discrete event feature and component-based architecture. Like NS-2, OMNeT++ is extensible and actively maintained by its research community, which can be seen from the increase in related publications from less than 50 in 2004 to more than 250 in 2011 [7]. It offers an Eclipse-based IDE and runtime environment, so programming, configuring, debugging, tracing and analyzing are relatively easy. Simulation models of OMNeT++ are programmed in C++, and together with the high-level declarative language NED, these models (modules) are assembled into larger components to represent a greater system. In addition, OMNeT++ supports extensions for real-time simulation, network emulation, several programming languages (Java, C#, SystemC), database integration and other types of function. Thanks to the contributions of researchers, several commonly used simulators for WSNs have been built in the OMNeT++ simulation framework. MiXiM [73] is an OMNeT++ based simulation tool that supports WSNs, body area networks (BAN), mobile networks, etc. It consists of four parts: ChSim (channel simulator), Mac Simulator, Mobility Framework and Positif Framework [74]. With limited protocol support, MiXiM only implements CSMA based IEEE 802.11 and IEEE 802.15.4 [75] as well as T-MAC. The current version of MiXiM offers a simple battery model for coarse evaluation of available energy [76]. Castalia [77] is another simulator for WSNs built with OMNeT++. The main features are the advanced wireless channel, radio propagation models as well as the implementation of MAC and routing protocols, but the simple energy model can only take into account CPU power consumption [78]. In summary, the lack of both accurate HW models of the sensor node and the precise battery models are problems common to both MiXiM and Castalia. In addition to OMNeT++ based simulator, NesCT [79] is an extension that can translate nesC code into C++ classes to make TinyOS applications functional on OMNeT++.

Prowler [8] [80] stands out as the most well-known MATLAB-based WSNs simulators. Implemented by MATLAB's m files, Prowler is developed as a discrete event driven simulator and it is originally designed to target Berkeley MICA mote running TinyOS in addition to more general-purpose sensor networks. Prowler is capable of simulating in either deterministic mode (e.g., application test) or probabilistic mode (e.g., wireless communication channel, low level communication protocols). Besides, Prowler can also manage arbitrary numbers of sensor motes, and dynamic topologies. It benefits from the MATLAB environment, and the integration of application models and radio propagation models become flexible and easier using the m-file plug-ins. Meanwhile, the use of such kind of plug-in offers great convenience for debugging, optimization, parameter tuning and application prototyping. With the provided MATLAB based GUI, Prowler shows a good visualization capability and facilitates the development and operation process especially for beginners. However, one of the biggest limitations is that the function for energy consumption is not included and an extension for this could be necessary. Another problem is the limited protocol since only one TinyOS MAC protocol has been implemented by default. Note that a Java based simulator named as JProwler [81] provides the same capabilities as Prowler but no energy model is provided.

2.7.2 Emulation Tools

Emulation is an approach that combines hardware and software in which some components come from real hardware (e.g., running code in a real sensor node) and others adopt simulation (e.g., channel modeling, network traffic etc). Some commonly used emulators are listed as follows:

TOSSIN [9] is an emulator designed to run actual code of TinyOS. The application code of TinyOS is compiled to the TOSSIN framework that runs on a PC by only replacing a few low-level components with the simulation implementation. After successful tests for specific algorithms or applications, these implementations can be deployed directly to the real world TinyOS-based mote with only slight modifications. This facilitates the whole development process since a more controllable environment is provided for users' debugging and analysis. HW models and wireless channel models are abstracted in TOSSIN, and TOSSIN scales well to networks with a large number (thousands) of sensor nodes. TinyViz [9] is a Java-based GUI that can offer good visualization and controlling interfaces to users. However, since the design of TOSSIN requires all simulated nodes to run the same application code, the flexibility of the simulation is limited. The biggest problem is that TOSSIN does not have an energy consumption model, and the abstract hardware models also fail to capture low-level details of timing or interrupt issues. Hence, precise energy evaluation cannot be guaranteed. PowerTOSSIN [82] works as an add-on and can help deal with the energy consumption evaluation issue of TOSSIN. The lack of a decent user guide document can be another drawback.

ATEMU [10] (ATmel EMUlator, written in C) is a fine grained instruction-level emulator for AVR based node platform. ATEMU is binary compatible with MICA2 sensor mote by default but its architecture is extensible to support different hardware platforms according to [10]. When compared with TOSSIN, different parameter configurations are allowed on each single mote in a hierarchical manner in ATEMU by using an XML file. At the same time, ATEMU also comes with a graphical debugger called XATDB which is able to offer breaking setting, variable tracking, step-based debugging, etc. However, the instruction-level based high accuracy of ATEMU is achieved at the cost of simulation speed and scalability, which is 30 times slower than TOSSIN [11]. Furthermore, lacking an energy model, the study of energy performance is not supported by ATEMU.

Avrora [11] is a well-known open-source cycle accurate emulator designed for AVR microcontrollers and has good support for MICA2 and MICAz motes. Avrora operates in an instruction-level manner, so actual microcontroller programs can run directly in the emulator rather than employing software simulated models. During emulation, one thread is assigned to each node and in order to improve efficiency, Avrora only processes synchronization when necessary (e.g., ensure global time, the order of radio communication) instead of synchronizing all nodes after each instruction. The experimental results show that a network made up of 10,000 nodes under Avrora can be evaluated as much as 20 times faster than most other emulators (e.g., ATMEU) but is still 50% slower than TOSSIM [11]. Written in Java, Avrora has good flexibility and portability but everything has to be done manually by command lines since no GUI is provided. Although Avrora is developed without an energy model for lifetime prediction, AEON [83] is implemented on top of Avrora for energy consumption. Real world energy data are used in AEON to calibrate the energy model and cycle accurate executions are performed with the combination of these measurements for energy evaluation. Despite having higher accuracy than PowerTOSSIN, the design focus of AEON is still on TinyOS based

nodes, which excludes its use from simulating a heterogeneous network system.

COOJA [84] is a Java-based emulation framework for the Contiki operating system. Its cross-level simulation contains the simulation of high-level abstractions and emulation of low-level hardware behaviors. COOJA's code level simulation is achieved by compiling the Contiki core and then executing the object code. During the process of code execution, COOJA supports simultaneous simulation at both operating system level and network level. The offer of a configuration GUI should be considered in the further work. MSPsim [85] is an additional add-on that can be integrated into COOJA for WSN nodes with TI MSP430 microcontrollers. It combines cycle accurate interpretation of microcontroller instructions with discrete event simulation. MSPsim also provides a Java-based GUI for sensor board visual representation, break point setting, stepping, data tracking, etc. Similarly, MPLAB SIM [86] (one of the emulation engines from Microchip's MPLAB IDE [87]), can be used to emulate WSN nodes with 8/16/32 bit based PIC microcontrollers. As a program code emulator, MPLAB SIM can only emulate microcontroller application code for algorithm and implementation evaluation, while the modeling and abstraction of the channel is impossible. For the three emulators, the study of energy consumption is not allowed.

2.7.3 Emerging SystemC based Simulation Tools

As an emerging simulation tool, SystemC [88] (details in section 3.1) can provide very good hardware/software co-design, hardware/software concurrency, hierarchy modeling and synchronization. Therefore, SystemC suits well system abstraction and cross-level based system design. In recent years, considering the complexity of wireless sensor networks, researchers increasingly start to use SystemC for sensor network simulation. Some of the typical works are described as follows:

SCNSL (SystemC Network Simulation Library) [89], to the best of our knowledge, is the first well designed open source SystemC-based networked embedded system for WSN simulation. SCNSL introduces a joint design of HW, SW and network to simplify the exploration and evaluation of complex sensor network systems. Node modules (written in SystemC) can be modeled at different abstraction levels and sub-modules can be easily added (e.g., transceiver, sensor). Before simulation, each node instance is connected to *NodeProxy* (written in SystemC) that acts as the interface between node and network. Meanwhile, it decouples node implementation from network simulation to help exploit the advantages of HW/SW co-design on node modeling. The network module written in C++ can perform such functions as communication channel, propagation delay, collision, interference, path loss, etc. The lack of specific hardware platforms, communication protocols as well as energy models are the biggest limitations for SCNSL.

ATLeS-SN (Arizona Transaction-Level Simulator for Sensor Network) [90] is a typical SystemC and TLM (Transaction Level Modeling-section 3.1.4) based sensor network simulation framework. ATLeS-SN contains Sensor Node-level Modeling and Physical Channel-level Modeling. In Sensor Node modeling, an App component for specific functionality, a Sensor component for sensor structure modeling, and a NetStack component for network communication are included. Physical Channel is used for real physical medium modeling, shared by wired or wireless nodes. Benefiting from the modular design and TLM, ATLeS-SN can offer tremendous flexibility and cross-level simulation support. Its availability has been proved in building monitoring, forest fire detection and propagation tracking application. However, ATLeS-SN has no standard protocol implemented and the lack of energy consumption model could be a severe drawback.

In [91], a complete TLM model of the IEEE 802.15.4 standard for WSNs has been implemented in the SystemC environment and some experiments have also been made to analyze its performance, which helps bridge the gap between the lack of standard communication protocols in the previous described approaches and the following SystemC based simulation frameworks. However, no energy consumption is considered in this work, as like SCNSL and ATLeS-SN.

In the following, some SystemC-based WSN simulation tools with energy aware capabilities are listed and described.

A high level SystemC TLM based WSNs simulation platform was proposed by D. Serna et al. [92]. Based on a component design methodology, like ATLeS-SN, it can offer good flexibility and generality. It consists of five components:

- MCU component uses ISS (Instruction Set Simulator) to address CPU instructions and cycles of TI MSP430, and transaction level functions are used for the communication among different peripherals (e.g., RAM, Flash, ADC, I/O etc).
- Radio component provides functional implementations to simulate the configuration of registers and packet reading/writing in RAM for CC2420.
- The power consumption and processor cycle monitor component is designed to track and record the detailed executions of CPU to perform energy evaluation.
- Sensor module component is responsible for the periodic human body measurements upload.
- Finally, the network component is built on SCNSL for communication channel modeling.

Since wireless communication is always the most consumption part, energy monitoring only on MCU could be a major drawback in this work. Furthermore, no communication protocols are implemented except for a channel model.

Jan Haase et al. [93] adopted SystemC with TLM 2.0 (Transaction Level Modeling) and ISS to design an approach for the energy consumption on ultra-low power wireless sensor networks. The researchers modeled node internal structure and communication among nodes with SystemC and TLM. The whole simulation framework is modified from PAWiS (Power Aware Wireless Sensor) [94]. The core design of reduced power consumption on the node is due to the proposal and implementation of dedicated Reconfigurable Hardware Blocks (RHB), which function as an independent part and help deal with simple tasks to save power, since frequent and periodic CPU activations can be avoided. However, the evaluation of power consumption in this work focuses only on a single node without the network environment since no protocols or channel are modeled.

Another SystemC based simulation methodology for WSNs energy consumption was proposed by Andrey Somov et al [95]. In this work, the authors offered a hierarchical framework. The layers, from the top down, are: application layer, service layer, platform layer and energy source layer. Users can describe specific application cases in the C/C++ language in the application layer and interact with hardware by means of services that are provided by service layer. The platform layer defines a set of available sensor platforms which have already been implemented, otherwise a compilation error will occur with an unknown node. A

PSpice compatible NiMH battery model [96] is built in the energy source layer for power evaluation. The shortcoming of this framework is that the power consumption perspective focuses mainly on the TinyOS based node, and the power consumption performance in a heterogeneous network would only be possible when an additional network layer with specific protocols can be integrated.

IDEA1 [97] (hIerarchical DEsign plAtform for sensOr Networks Exploration) is a SystemC-based system-level simulation environment that built on SCNSL. As an enhanced version of SCNSL, IDEA1 extends the node module to include several different mote platforms which are MICAz, MICA2 and N@L [97], and the slotted/unslotted CSMA/CA algorithms of IEEE 802.15.4 are also implemented. A battery model that can trace the current power consumption of different device components is designed. However, during the simulation of IDEA1, each node has the same configuration so the energy evaluation is limited to some simple scenarios. Thus, the evaluation of energy consumption via IDEA1 is unavailable for relatively complex sensor network environments (e.g., different configurations on different nodes) and for environment where dynamic configuration is required (e.g., some adaptive protocols/strategies where the node configurations might be changed automatically in the simulation based on environment conditions).

2.7.4 Testbeds

Simulation and emulation tools provide a quick and efficient way to develop and evaluate the algorithms, protocols, strategies and applications for WSNs. However, due to the complex environment conditions of the real world, simulators and emulators may sometimes offer oversimplified models, so testbed-based experiments are essential before practical deployment since more realistic results can be provided and can also be used to calibrate the simulation/emulation models. In this section, only some testbeds are described. More comprehensive details about testbeds are available in [78] [98].

MoteLab [99] [100] stands out as one of the first fully developed WSN testbeds. It consists of 190 TMote Sky motes running with TinyOS and provides a public, permanent indoor-based testbed environment for developing and testing sensor network applications via a web-based interface. Registered users are allowed to upload their executable files to run on real mote. Each mote is wall powered and is connected to a central server that offers scheduling, reprogramming and data logging. During the process of job execution, a set of simple visualization tools is provided by the web interface for data tracing and viewing. However, once the uploaded test file is scheduled, no more changes can be done. Thus, using TOSSIN for corresponding simulations at the very beginning is highly recommended before uploading the final program onto the MoteLab. The design of MoteLab facilitates and accelerates research in sensor network programming, communication protocols, system design as well as application strategies [100]. A similar testbed called INDRIYA [101] is developed by researchers at National University of Singapore. INDRIYA comprises 139 TelosB sensor motes and more functional sensors are attached to each mote when compared to MoteLab. Instead of employing single-board and wall powered computers, INDRIYA testbeds can be reliably built over USB infrastructures and with low maintenance that under $500 for 2 years. Although sharing the same design goal with MoteLab, INDRIYA's further contribution is to provide extensive measurement and analysis of performance differences and correlations that might exist among different channels, which can help researchers in the WSN community design dynamic channel-switching based MAC or routing protocols.

TWIST [102] is a hierarchical based WSN testbed for indoor experiments. It has the architecture of

three layers. Starting from the lowest tier, sensor motes are deployed to sample and monitor required environment signals and they are connected to control servers at the second tier by USB infrastructures. Active and passive USB cables are both used to extend the size of the network, and message communication between the sensor node and control server is bidirectional. At the top tier, a central server can connect all the control servers via an Ethernet backbone and it is used to store uploaded programs, debugging and data logging. Other features of TWIST include heterogeneous mote platform support, active power supply, self-configuration, standardized interfaces and open-source software, which make TWIST scalable and reconfigurable. Taking the same architecture of TWIST, WUSTL WSN testbed [103] currently consists of 79 sensor motes placed throughout several office buildings of the college. In the hope of further advances and to facilitate research in WSNs, the WUSTL WSN testbed is well designed and maintained. Its developers claim that it is possible to run sophisticated experiments without manually changing the physical layout. Specific configuration of each individual node can be done automatically by the central server with the required format script.

SensLAB [104] [105] is a newly developed large scale open wireless sensor network testbed, which consists of 1024 WSN430 nodes [104] distributed over four different sites in France. Two sites can also offer mobile testbed nodes. The design of each SensLAB testbed node is composed of three sub function nodes which are an open node, a control node and a gateway node. An open node is the node that is involved in WSN experiments. Software on the control node is in charge of activity control of the open node as well as the programming task. The gateway node provides the message communication interface between the open node and the control node. The setup and control of experiments can be done by inputting specific SensLAB command lines in the virtual machine. The most significant aim of SensLAB is to offer an accurate and efficient scientific tool to help in the design, development, tuning, and experimentation of real large-scale sensor network applications [104].

2.7.5 Summary

Based on studies of evaluation tools for WSNs, it is easy to conclude that general WSNs simulators written in C++, Java or MATLAB can be a good choice at the early design stage for high efficiency requirements, since the system can be abstracted and modeled at the high-level for validating and testing algorithms, protocols and strategies. In turn, they might not be suitable for precise energy consumption scenarios due to the lack of realistic models and low level details.

In comparison, emulators for WSNs can be more effective and useful in both development effort and energy consumption, because emulators can run the code that are almost the same in the real node, so efforts in code rewriting are avoided. With cycle-accurate or bit-level operation mechanisms of emulators, more detailed information on resource usage can be provided, which is very helpful for a more realistic energy evaluation. The problem with emulators is that they are constrained to specific hardware platforms or operating systems, and thus heterogeneous sensor networks are not allowed. Besides, too much low level details come at the cost of efficiency and scalability.

The emerging SystemC-based simulation provides an alternative to model increasingly complex sensor network systems. This is because with the natural supports of concurrency modeling, pipelining modeling, hierarchical architecture modeling and synchronization modeling, SystemC can offer various abstraction

level based system models and required details on specific hardware/software operations, which can satisfy the requirements of different energy investigations. Nevertheless, the incompatibility of the program code with the real node and invalid hardware models are left unresolved.

Therefore, the idea of mixed-mode simulation [78] [106] that involves testbed node experiments are regarded as the tradeoff between accuracy and efficiency for the energy consumption of WSNs. A physical testbed node is not only a low cost prototyping solution, but also provides a controllable and real-world like environment that is similar to actual deployment. With the appropriate measurement platform, experimental results from testbed nodes can be used to calibrate both the energy model and HW/SW operating mechanism model of the simulation. Such a mixed-mode based energy simulation environment called iWEEP (**i**NL@**W**SNs **E**nergy **E**valuation **P**latform) will be described in the following Chapter 3.

2.8 Conclusion

This chapter investigates the state of the art of wireless sensor networks (WSNs), which includes the general aspects of architecture, applications and design challenges. In particular, much more efforts are based on the research of the energy performance of WSNs. First, elaborate energy consumption is analyzed at the hardware and network levels. Then, some typical and emerging energy conservation techniques are presented with the purpose of achieving an energy-saving sensor network, and MAC protocols with energy-efficient characteristics are emphasized. Operating systems (OS) of WSNs are described, since the optimized utilization of these OS might offer potential energy conservation. Finally, tools for the evaluation of the energy consumption of WSNs are surveyed from simulators, emulators, testbeds and a mixed-mode energy evaluation method that involves SystemC simulation and testbed measurements are highlighted. All investigations offer the most relevant materials that form the background of the research presented in this doctorate thesis.

Chapter 3

The Design and Implementation of iWEEP

Contents

In this chapter, a complete sensor network energy evaluation platform, named iWEEP (**i**NL@**W**SNs **E**nergy **E**valuation **P**latform), is presented. iWEEP consists of iWEEP_SW in software and iWEEP_HW in hardware. iWEEP_SW is a system-level transaction-level and energy-aware sensor network simulation environment that built on SystemC, and it focuses on the modular design of separate hardware components on sensor motes at different abstraction levels. Since SystemC is a system-level and hardware description language for SW and HW co-design, so SystemC-based iWEEP_SW can accurately evaluate the energy consumption performance of the target sensor networks according to the application requirements, the implemented network protocols and communication strategies at an early design stage of various abstraction levels. On the other hand, iWEEP_HW is a real-world energy measurements system made up of a measurement platform **M**ultichannel **E**nergy **M**easurement **D**evice (MEMD) and an **E**nergy **D**ata **M**anagement **S**oftware

Platform (EDMSP). The combination of MEMD and EDMSP can provide accurate and synchronous energy consumption monitoring for the current widely used sensor motes to calibrate their energy models as well as their operation models for the realistic energy performance estimation. In this thesis, the energy consumption of our lab-built high data rate and ultra-low power based testbed node iHop@Node (**i**NL@**H**igh data rate and ultra l**o**w **p**ower testbed **Node**) is measured by iWEEP_HW. For iWEEP_HW platform, it can also be considered as a flexible and cost-effective alternative for most expensive equipment and tools in sensor mote energy measurement research.

This chapter is organized as follows. Section 3.1 briefly introduces the SystemC based modeling concept. Section 3.2 gives an overview of iWEEP. Section 3.3 and section 3.4 describe iWEEP_SW and iWEEP_HW in details respectively. Finally, section 3.5 concludes this chapter.

3.1 SystemC Modeling

Due to the complexity of the wireless sensor networks design process, methodologies using high levels of abstraction are considered to be very necessary, since they allow the exploration of hardware and software architecture at an early design stage even though the hardware is still under development. Some system-level design methodologies have been proposed in recent years, such as model-driven design [107] and platform-based design (PBD) [108]. Both methods involve software and hardware design and support early design verification, and detailed implementations are tested on the real WSN system application. In this work, a generic approach using SystemC is adopted.

3.1.1 Brief Introduction to SystemC

SystemC is a modeling language based on standard C++ that is designed for system-level modeling, design and verification. It is created and maintained by Open SystemC Initiative (OSCI), and has been approved by the IEEE Std. 1666^{TM}-2005 [88]. As the complexity of ICs increases, the traditional software and hardware separated design method is no longer viable. Since the C/C++ language and existing hardware description languages (HDLs) such as VHDL or Verilog are entirely different, the setup of a proper testbench for both is difficult [109]. On the other hand, the use of clock cycle level based HDLs for complex IC design becomes increasing unfeasible, such as the system-on-chip (SoC) technology, because it consumes a much longer simulation time and a large amount of generated information are also unmanageable. Further, the HDLs cannot provide a good validation support in such a complex system design process [110]. SystemC simplifies and solves many of these limitations and provides and promotes excellent support for hardware and software co-design techniques [111].

3.1.2 Advantages of SystemC

SystemC is a C++ class library that implements hardware related concepts, and the following lists three significant advantages of using it.

Ease of use/Generic: Since C++ is a stable and well developed language for (embedded) software programming all over the world, it is familiar to most designers. SystemC is a class library of C++ and

therefore it is easy and quick to manage and can be associated with plenty of other development tools. Besides, since SystemC is entirely based on C++, it inherits all the features of C++, such that it can accept all the data-types offered by C++ and user-defined data structures, and also supports mechanism such as function inheritance and the use of templates [112] [89], all of which facilitates program writing with less effort.

HW/SW co-simulation: By introducing some constructs such as the notion of time, concurrency and reactive behavior that are missing in standard C++ but required in HDLs, SystemC is able to simulate synchronous hardware design and concurrent event modeling. HW/SW co-design can thus be developed in a single common language, which meets the requirement of fast executable specifications to validate system concepts and make a good time-to-market opportunity [109]. Furthermore, it releases designers from the need to be an expert in multiple languages, and much development time can be saved by the reuse of testbenches [113].

Efficiency: SystemC supports the design at different abstraction levels, such as transaction-level-modeling (TLM) and register-transfer-level modeling (RTL), which effectively meets the various modeling requirements. On the other hand, the description of hardware by SystemC can achieve an order of magnitude faster simulation time than VHDL [110].

3.1.3 SystemC Library

As a set of C++ classes, the SystemC library provides its own modeling components which consist of five major parts, which are Module, Process, Channel/Port, Hardware Data Type and Time model [113]. The SystemC library also provides an event-driven simulation kernel.

Module: A SystemC model usually consists of several modules, which represent small and manageable pieces of a complex system. Each piece of these modules is modeled by the *SC_MODULE* class and acts like the SW/HW component built in real world IC system. Besides, a module can also use another module in a hierarchy.

Process: Processes are contained inside the modules to simulate the function/behavior of the target system, and these processes are designed for concurrent execution irrespective of the calling order. *Methods*, *Threads* and *CThreads* are three types of processes provided by SystemC and they are the member functions of an *SC_MODULE*.

Channel/Port: In SystemC, the interconnections between modules use channels, and channels connect to modules via ports. Ports of different modules can provide input and output interfaces or the interfaces with bidirectional characteristics (Input and output) for the communication with other modules.

Hardware Data Type: Several new hardware data types are introduced in SystemC, such as bits, bit vector, arbitrary sized integers and fixed point type, which are useful for modeling and evaluating complex hardware functions.

Time Model: SystemC library also provides a time model for different abstraction level modeling. A class known as *sc_time* gives a time resolution from *SC_FSEC* (femtosecond) to *SC_SEC* (second), and the *sc_clock* class can provide a user defined clock.

Simulation Kernel: The SystemC simulation process has two operation phases, which are elaboration and execution, and after that a cleanup phase occurs [114]. The elaboration process happens at the beginning

of the *sc_main()* function, which is the simulation entry point of SystemC, and elaboration handles data structure initialization, module instance creation, and interface connection in modules. Then by calling *sc_start()*, execution starts and processes in modules are triggered concurrently to run when events to which the processes are sensitive occur. After all processes have been invoked during one simulation cycle, the simulation time is advanced to the next cycle with all the updated information until the stop of simulation. Finally, cleanup is responsible for releasing the memory by deleting module objects.

3.1.4 SystemC TLM

Despite the fact that SystemC supports designs with different abstraction levels, the primary goal of using SystemC is for system-level modeling (i.e. the modeling of systems above RTL level). No matter whether there are two [115], three [116] or four [117] levels, all agree that there exists an abstraction of transaction level based modeling above RTL. Starting from SystemC 2.0, transaction-level modeling (TLM) is supported to facilitate this upper level system design for the higher productivity of software and hardware co-design. With TLM, more emphasis is focused on the system functionalities while the detailed implementations of the function units are hidden. This makes it easier for system designers, because they can use function calls for the communications among different modules without considering actual protocol details to speed up their design process.

3.2 iWEEP Overview

With the widespread development of embedded systems and wireless communication technologies, wireless sensor networks (WSNs) have gained attention of industrial and research groups all over the world in recent years and their applications are found in various areas such as target tracking, building/environment monitoring, and also emerging medical and health care. Wireless sensor nodes need to be widely deployed in tracking and monitoring fields for long-term operational purposes. For instance, small-scale dimension size and lightweight nodes are employed to meet wearable or implantable requirements in medical applications and with years of functionality. Thus, energy consumption is one of the most constraining concerns for the design and implementation of these sensor nodes and the whole network, since most of these nodes are battery-driven with limited energy supply.

The provision of accurate energy consumption profiling is significant for evaluating the performance of wireless sensor networks in various applications. However, theoretical model analysis method is always unrealistic and leads results that are untrustworthy. In such cases, the energy source of sensor networks may prove to be inadequate and exhaust itself before completion of the required task, or conversely the node might become redundant long before the depletion of the energy. Only hardware-based measurements have the required accuracy, but are also extremely time-consuming requiring repeated experiments and calibration for each application scenario. Hence, a mixed energy evaluation approach that combines simulation and accurate extracted energy measurements of the testbed nodes/target sensor nodes can provide a good tradeoff between accuracy and efficiency.

The design of iWEEP (**i**NL@**W**SN **E**nergy **E**valuation **P**latform) consists of two parts. iWEEP_SW is a system-level, transaction-level and energy-aware SystemC based generic sensor network simulation

environment focusing on modular design for components on sensor nodes and sensor networks. iWEEP_HW is a real-world energy measurement system made up of a measurement platform MEMD (**M**ultichannel **E**nergy **M**easurement **D**evice) and an energy management platform EDMSP (**E**nergy **D**ata **M**anagement **S**oftware **P**latform), which can provide realistic and accurate measurements for the target sensor nodes. Simulation results from iWEEP_SW are calibrated by iWEEP_HW measurements to achieve a much more realistic and trustworthy energy performance evaluation. In this section, the design framework and implementation of iWEEP are presented.

3.3 iWEEP_SW Overview

iWEEP_SW is a system-level energy-aware and component-based simulation/emulation framework built by SystemC transaction-level modeling (SystemC TLM) that supports the evaluation of energy performance at different abstraction levels. Fig.3.1 provides an overview of the iWEEP_SW framework. Detailed modeling information will be presented from section 3.3.2 to section 3.3.6.

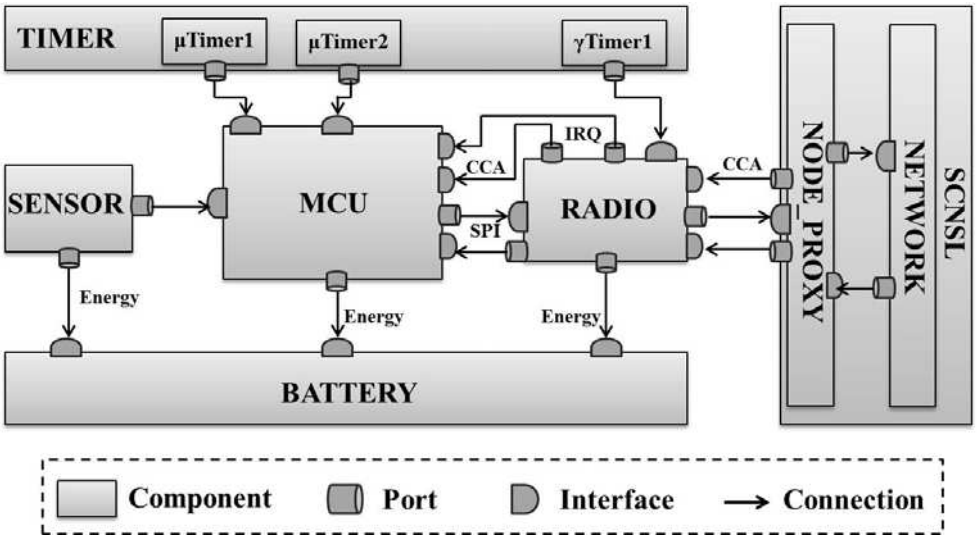

Figure 3.1: iWEEP_SW framework

In Fig.3.1, the iWEEP_SW is composed of components, ports, channels, interfaces and connections. Components correspond to basic hardware modules such as microcontroller, transceiver, sensor and battery as well as various kinds of peripherals. Each component in iWEEP_SW is modeled as an individual module of SystemC. In these hardware component models, the sensor is modeled to generate periodical sensed data from physical or environment conditions. The microcontroller (MCU) collects these analog data and converts them into digital signals by the built-in Analog to Digital Converter (ADC). The converted digital data are then sent to the transceiver according to different application scenarios and communication strategies. The transceiver (Radio) takes the responsibility of over-the-air transmission of packets, reception of packets (interrupt to MCU), and clear channel assessment (CCA). The energy module functions as a power consumption monitor, and can track and assess the energy consumption of all other hardware components

at different abstraction levels from bit-accurate/cycle-accurate to transaction-accurate based on designer's requirements. In the timer module, several sub-timer components can be defined for different purposes such as timer for sample interval, timer interval/delay for each transmission, protocol related timer, etc. For some other peripherals such as the memory model and UART model inside the MCU, the models are not mandatory and their modeling levels are optional, all of which depend on the detailed design requirements. In addition, the network module is also modeled as a component since it is inherited from the TLM-based SystemC Network Simulation Library (SCNSL) [89], and it is used to establish network topology, implement packet transmission and handle network collisions.

Channels are responsible for the communications between different components, such as the internal bus of the microcontroller connecting CPU core and peripherals, as well as the SPI (Serial Peripheral Interface) bus connecting microcontroller and transceiver. The channels' implementations are not mandatory and they are always abstracted to simplify the development process based on the designer's requirements.

Each component and channel can have one or more interfaces to distinguish and specify a set of functions (or *transactions*), which are used to interact with the relevant components or channels. Like interfaces, multiple ports can be defined by components and channels, and they are utilized to specify the types of the interfaces, so each function-specific port can be connected to the related component or channel as long as the corresponding interface is implemented by that component or channel [90].

This generic transaction-level modeling framework provides designers the requisite extensibility and flexibility in the modeling, refining and simulation a complex sensor network system. Developers and designers are allowed to focus on the modeling and the refining of individual components of the system at various levels with required accuracy. At an early design stage, high-level modules, without detailed communication and time information, can be used to verify the functional correctness of the specific components. With excellent scalability of transaction-level modeling (TLM), such a component module is able to be extended to an approximate time function module [118] by incorporating related timing description elements that provide an approximate estimation of time that required by the given tasks being performed, or a high-level abstraction channel can also be extended for realistic emulation by implementing the detailed communication protocols. Therefore, more elaborate performance information on the implementation and refinement of the final target system are available for the evaluations and optimizations. When moving forward to the RTL design stage, the approximate time module can be replaced by a cycle-accurate based implementation, like [92], [119] and [120]. A cycle-based instruction-set simulator (ISS) is modeled to emulate the detailed behaviors of the processor for precise time and energy consumption evaluation. However, this requires a long simulation/emulation time for a single node, and even a relatively small sensor network with tens of nodes could be extremely time-consuming. As previously mentioned, by adopting the approximate-time modeling measured in terms such as *ns/instruction* or *μs/instruction* for code execution processes in target hardware components, relatively accurate energy consumption values can be acquired at this high level of abstraction while significantly reducing simulation time. We based our work on real world measurements from [19] and [39] as well as from section 3.4.6, where existing widely used 8/16 bits microcontrollers consume similar power in instruction executions [121]. Hence, in iWEEP_SW the approximate-time mechanism is adopted for achieving the tradeoff between efficiency and accuracy on energy consumption evaluation.

In addition, in iWEEP_SW framework, the whole sensor network system maintains two shared data

items for packet frames. One is a generic packet format and the other is IEEE 802.15.4 based packet format, which are presented in Fig.3.2 and Fig.3.3 respectively.

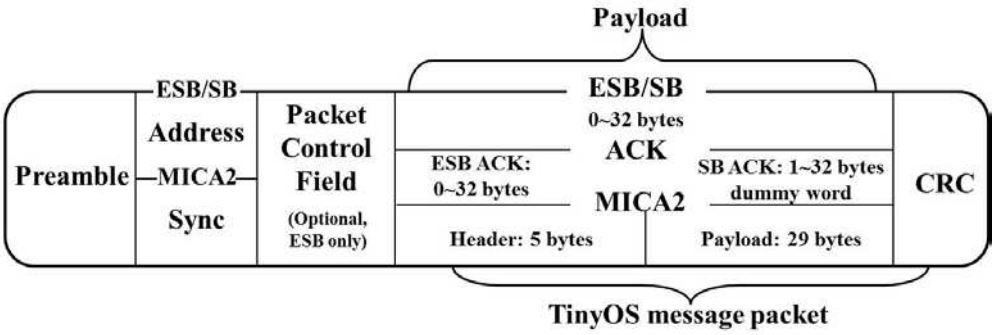

Figure 3.2: Generic packet format

The generic packet format frame can be used for Enhanced ShockBurst (ESB) / ShockBurst (SB) [21] protocol-based sensor node network (described in section 3.3.3.2), which is based on nRF24 series transceivers [122]. A preamble sequence is used to synchronize the receiver demodulator for the incoming data. The address is to ensure packet data can be detected and received by the correct receiver. The packet control field (PCF) is optional and it is required only by the ESB based network. The payload saves user defined content and its size depends on application scenarios without exceeding limitations of hardware (transceiver) buffer space or operating system (TinyOS) defined packet structure length, but it is not mandatory for the ACK frame. A CRC (Cyclic Redundancy Check) is utilized for error detection and correction in the packet. The PCF in ESB mode is divided into three fields, which are payload length, packet identity and ACK request flags. Further, this generic frame format can also be extended or modified for other networks, such as for MICA2 TinyOS based packet format [123] [124]. Another IEEE 802.15.4 based packet frame is shown in Fig.3.3.

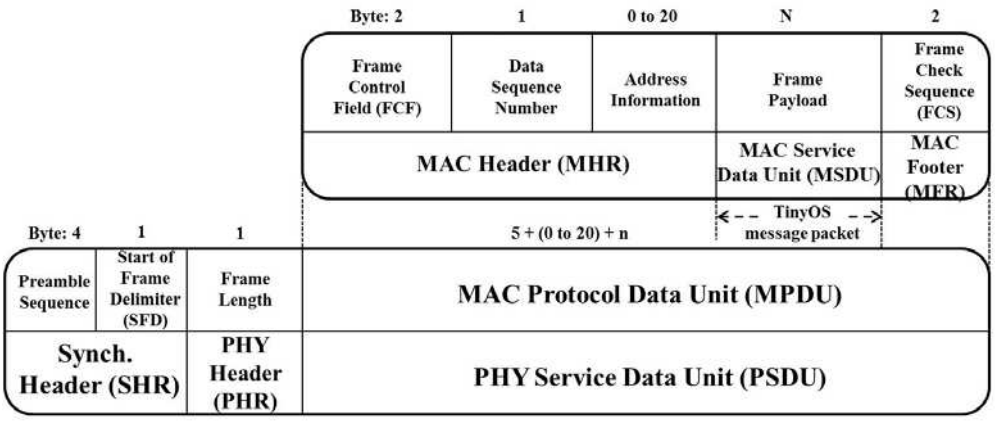

Figure 3.3: IEEE 802.15.4 packet frame format

The preamble sequence, frame length, address information, payload and frame check sequence (for CRC check) have similar functionalities as described above. SFD (likes Sync in MICA2 TinyOS packet) acts to indicate the arrival of the packet. The frame control field includes some flags for specifying the packet type (e.g. beacon, data and ACK).

On the other hand, a shared data structure used for statistics purposes is maintained in each sensor node. This shared data structure is used to record some simulation results such as packet loss, latency, energy consumption, number of sent packets, number of received packets, etc. This structure is shown in Fig.3.4, and if the whole network is under investigation rather than a single node, a static data structure can be defined to fulfill this global statistic purpose.

```
//Data structure to record simulation results
typedef struct {
        long double latency;     //do not include ACK
        long double bi_latency;  //include ACK
        long double energy_mAh;  //measured in mAh
        long double energy_mW;
        long double energy_mJ;
        int packet_overflow;
        int packet_loss;
        int packet_send;
        int packet_received;
        int packet_retr;         //retransmission
        int CAF;                 //channel access failure
        int CF;                  //collision failure
        ..........................
} Sim_Results;
```

Figure 3.4: Shared data structure for recording simulation results

After the overview of the iWEEP_SW framework in this section, the following sub-sections will provide details of the implementation of iWEEP_SW by presenting the component-based design and modeling methods.

3.3.1 Library of iWEEP_SW

Two types of sensor mote networks are utilized as the case studies in iWEEP_SW framework. One is based on an in-house testbed node, named iHop@Node [19] (Its detailed information is also presented in section 3.4.3), which is our INL (Institut des Nanotechnologies de Lyon) lab-built high data rate and ultra low power testbed node. Another node system under investigation within iWEEP_SW modeling framework is the well-known Telos mote series [26] [125] (including Tmote Sky (TelosB) [15] or Shimmer node [16]), which is widely used in many medical/health monitoring applications [126] [127] and mainly consists of a 16-bit TI MSP430 microprocessor [128] with five low power modes and an IEEE 802.15.4 compatible transceiver chip CC2420 [25]. In addition, iWEEP_SW also integrates MICA2 [17], MICAz [18] and N@L from [97].

3.3.2 Microcontroller Modeling

The detailed model of the microcontroller is presented in Fig.3.5.

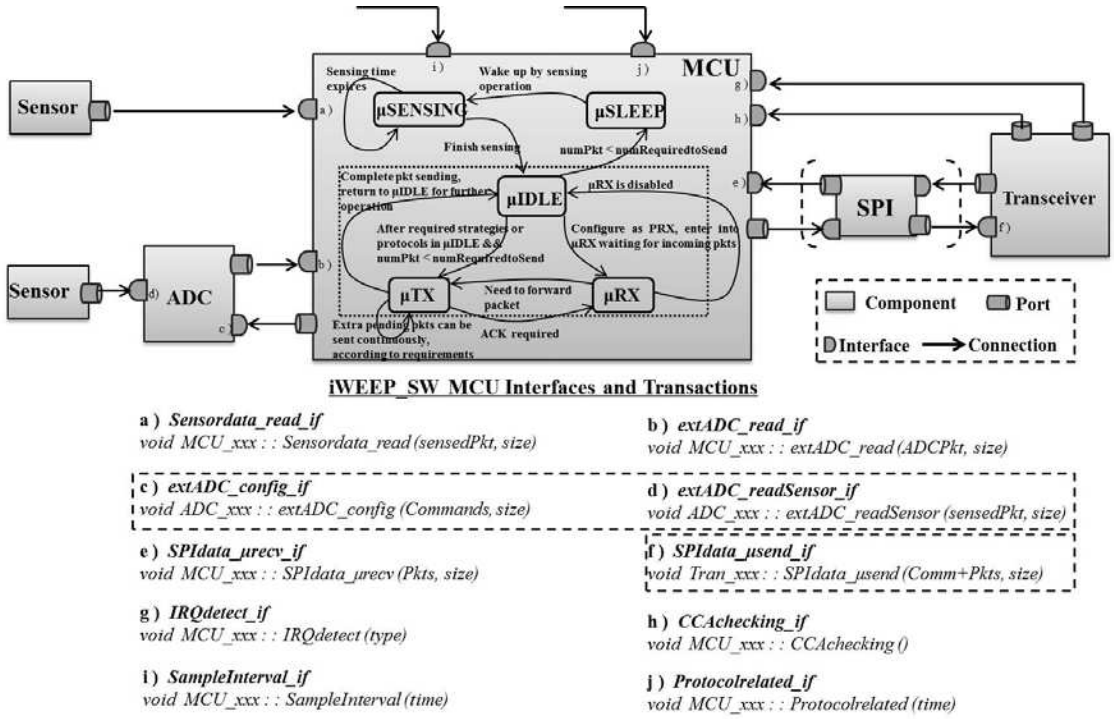

Figure 3.5: iWEEP_SW MCU model

The microcontroller component defines the *sensordata_read_if* interface consisting of a single *SensorDataRead* transaction (or function) which allows the microcontroller to first read data from the target sensor module and then convert these analog data into digital data by the built-in Analog to Digital Converter (ADC). In some application scenarios, if an external ADC is needed for a more accurate and quicker conversion requirement, the microcontroller defines another *extADC_read_if* interface. This is maintained by an *extADC_read* transaction (function) that emulates the process whereby the microcontroller reads converted ADC values through its digital I/O port pins or through SPI pins from an external ADC chip. The microcontroller component also links ADC's *extADC_config_if* interface by the defined output port on the microcontroller side, and this external ADC interface is composed of a single *extADC_config* transaction to receive configuration information from microcontroller. Moreover, the *extADC_readSensor_if* interface and *extADC_readSensor* transaction defined on the external ADC are used to acquire sampled sensor data. Considering that cycle-by-cycle based energy consumption on the sensor node will be evaluated in later work, the above interfaces and transactions can be easily extended to incorporate detailed communication mechanism and device driver code implementation. Another two transactions named *IRQdetect* and *CCAchecking*, which are defined by *IRQdetect_if* and *CCAchecking_if*, are responsible for interrupts detecting from transceiver and clear channel assessment processes respectively. Several timer transactions can be defined for varying functions such as the examples that are shown in Fig.3.5. In addition, interactions

between the microcontroller and transceiver are handled via SPI, considering that SPI communications between the microcontroller and transceiver are bidirectional. Hence, the SPI communication interface defines two transactions named *SPIdata_µrecv* and *SPIdata_µsend* on the microcontroller and transceiver components respectively. The *SPIdata_µrecv* transaction is called when the microcontroller needs to read data out from the transceiver's RAM (e.g., TX/RX FIFO, Buffer) and the *SPIdata_µsend* transaction provides specific command, data size, and address to configure the corresponding registers in transceiver and send data payload with the required length to the transceiver buffer space.

In the high abstraction level TLM-based framework, although detailed communication is hidden, both previously described SPI transaction contents can be easily extended with real SPI communication mechanisms for bit-accurate/cycle-accurate performance evaluation. iWEEP_SW hides SPI communication details and replaces by approximate-timed modeling. For example, in iHop@Node, approximate-time estimations are acquired from the implemented HW/SW SPI code (C code) execution time between PIC16F [129] and nRF24L01+ [19]. The built-in function *spi_read()* and *spi_write()* from CCS IDE [130] (version 4.057) are used to program HW SPI code. For SW SPI code, the microcontroller's GPIOs are used to emulate the SPI communication mechanism, and it is therefore available to any microcontrollers to establish communication with the nRF transceivers. The C code example of SW SPI bidirectional communication is presented in Fig.3.6.

SW SPI code execution time for all devices across PIC16, PIC18, PIC24 and dsPIC33 [131] can be accurately measured and evaluated by MPLAB IDE Stopwatch function [132]. For example, the use of SW SPI for one byte length register configuration (1 byte command followed by 1 byte configuration data) on iHop@Node takes 181.5 μs compared to 101.5 μs via HW SPI. In the iWEEP_SW framework, both SW and HW based SPI communication are integrated. For other non-PIC based sensor motes, SW SPI also provides a generic implementation of SPI communication between microcontroller and transceiver, and HW SPI can be implemented by directly using assembly code or built-in communication functions in the specific IDEs. Both HW and SW SPI approximation time are defined in *SPIdata_µsend* and *SPIdata_µrecv* transactions for more accurate system performance emulation. With all the above characteristics, this microcontroller framework can provide good scalability and flexibility for the further system refinement and performance evaluation.

In addition, since iWEEP_SW focuses on high level abstraction evaluation, the microcontroller internal communications between the CPU core and various peripheral modules, as well as instruction and code execution processes are out of scope of this work. However, the iWEEP_SW framework can be easily extended to support these cycle-accurate emulations, and modeling spaces for detailed programming has been reserved for these further refinements.

On the other hand, the operation of the microcontroller is performed as a finite state machine (FSM) with approximate-timed mechanism. For the most common periodic sensing scenario, the microcontroller is designed as a generic model shown in Fig.3.5. This generic FSM model is the basis for the modeling of other more specific models on different microcontrollers. In this generic model, the microcontroller wakes up periodically from μSLEEP state to the active state to sample data from sensor according to application requirements. The microcontroller's active working state is then divided into several sub-states (μIDLE, μRX, μTX and μSENSING) for different missions. The microcontroller enters into the μIDLE state after

```
//Send N bytes data and receive N bytes at the same time
unsigned char spi_Send_Read(unsigned char bytein)
{
  unsigned char byteout = 0x00;            // return byte

   output_low(SCK);                        // Ensure SCK is low
   if (bytein & 0x80)                      // Check if next outbit is a 1
   {
       output_high(SDO);                   // If a 1, pull SO high
   }
   if (input(SDI) == 1)                    // Check if next inbit is a 1
   {
       byteout |= 0x80;                    // If a 1, set next bit to 1
   }
   output_high(SCK);                       // Bring SCK high to latch bit
   output_low(SDO);                        // Reset SO to zero

   output_low(SCK);                        // Bring SCK low for next bit
   if (bytein & 0x40)                      // Check if next outbit is a 1
   {
       output_high(SDO);                   // If a 1, pull SO high
   }
   if (input(SDI) == 1)                    // Check if next inbit is a 1
   {
       byteout |= 0x40;                    // If a 1, set next bit to 1
   }
   output_high(SCK);                       // Bring SCK high to latch bit
   output_low(SDO);                        // Reset SO to zero

   ..............................
   ..............................

   output_low(SCK);                        // Bring SCK low for next bit
   if (bytein & 0x01)                      // Check if next outbit is a 1
   {
       output_high(SDO);                   // If a 1, pull SO high
   }
   if (input(SDI) == 1)                    // Check if next inbit is a 1
   {
       byteout |= 0x01;                    // If a 1, set next bit to 1
   }
   output_high(SCK);                       // Bring SCK high to latch bit
   output_low(SDO);                        // Reset SO to zero

   output_low(SCK);                        // Bring SCK low for next bit

  return byteout;
}
```

Figure 3.6: Software SPI C code

μSENSING if there are enough data packets pending to be sent out, otherwise the microcontroller returns to the μSLEEP state for energy saving. Various specific requirements such as sample time delay, communication strategy or complex protocol algorithm are implemented via software by combining μIDLE, μTX and μRX states. When there are enough data packets, the microcontroller will first go into the μTX state to send related data to the transceiver buffer, trigger over-the-air transmission and then wait until transmission completes by receiving an interrupt signal (PTX_IRQ) from transceiver or an acknowledgement. If there still remain extra packets pending to be sent, the microcontroller can start the μTX state again to perform another transmission process. When the node is enabled as a receiver, it can stay in the μRX state for the incoming packets, and the packets will be read out of the transceiver buffer by SPI when a reception interrupt (PRX_IRQ) is detected. Besides, μRX and μTX states are inter-convertible, which facilitates the switch process from transmitter to receiver or vice versa. Since large scale sensor network are deployed for various application scenarios, specific communication protocols are needed to help them access the channel and avoid packet collisions to guarantee transmission reliability. Many proposed protocols are listed in section 2.5.5 and their implementation type can be either transceiver hardware based or microcontroller software based. For hardware implemented types, the IEEE 802.15.4 MAC protocol is fully supported and implemented in MRF24J40 [41], and ESB/SB protocols are embedded in the nRF24L transceiver series. While for other motes like Telos, MICA2 and MICAz, users need to implement the MAC CSMA/CA algorithm of IEEE 802.15.4 manually by software programming in the microcontroller.

3.3.2.1 Modeling of PIC16

Developed by Microchip, MiWi [133] as a wireless networking protocol stack is especially designed for PIC microcontroller families from PIC16 to dsPIC33. Currently, only the non-beacon mode of IEEE 802.15.4 is implemented as MiWi's MAC layer, and our PIC16F microcontroller in iWEEP_SW is modeled following MiWi by using standard defined unslotted CSMA/CA algorithm in non-beacon mode. Our iHop@Node employs nRF24L01+ as the transceiver, which is integrated with another protocol engine-Enhanced ShockBurst (ESB)/ShockBurst (SB) (section 3.3.3). However, as stated in [134] and section 4.6, ESB only has simple channel access and collision avoidance mechanisms, which are not capable of handling a sensor node network with relatively high traffic loads, while SB is just a simple RX/TX protocol without any channel access or packet collision avoidance mechanisms. Hence, PIC16 modeled with the unslotted CSMA/CA algorithm is combined with the transceiver's SB mode for the better channel use and effective collision avoidance. As the comparison, the ESB mode in the transceiver model is used along with the microcontroller's generic FSM model (Fig.3.5). The unslotted CSMA/CA algorithm of IEEE 802.15.4 MAC protocol is modeled in the previously mentioned μIDLE, μTX and μRX states as FSM, and divides these three states into many sub-states. Its workflow is presented in Fig.3.7.

In Fig.3.7, the microcontroller first performs a random backoff duration and checks the channel status. If the channel is detected to be busy, and the number of CCA attempts is larger than the protocol parameter *macMaxCSMABackoffs*, then a Channel Access Failure (CAF) is reported. If on the other hand the number of CCA attempts is smaller than *macMaxCSMABackoffs*, the microcontroller will go back for a new round of random backoff process. When the channel is indicated as free by CCA, the microcontroller will send a packet to the transceiver and trigger over the air transmission. After the transmission of an ACK required

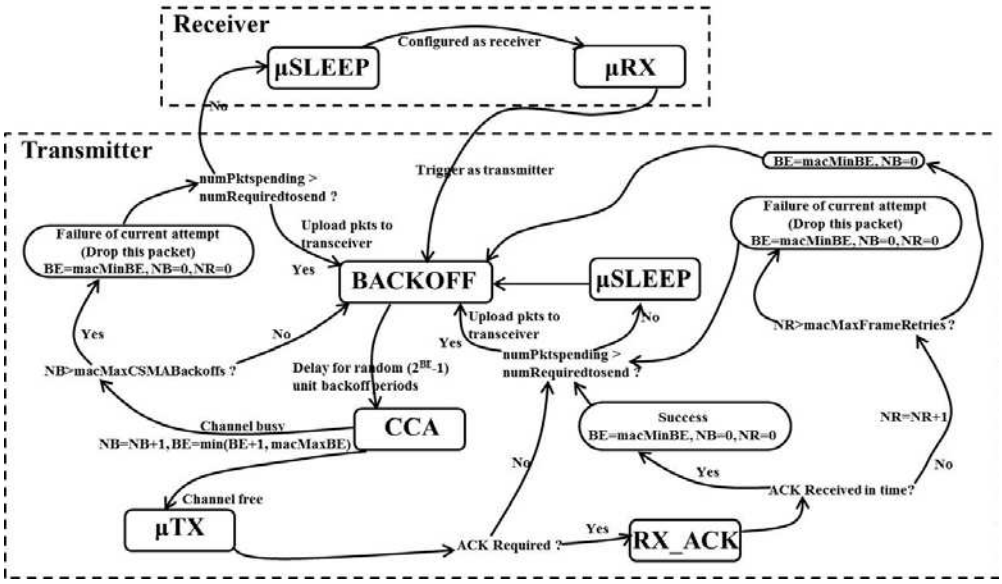

Figure 3.7: Unslotted CSMA/CA algorithm model

data packet, the protocol will make the microcontroller wait within a fixed time period (0.864 ms or 54 symbols) for the ACK frame confirmation. If ACK is received in time, this transmission is regarded as successful. Otherwise, the packet retransmission process will start. A Collision Failure (CF) occurs only after failure to receive ACK frame *macMaxFrameRetries* times, which might be caused by collisions of data packets or collisions between data packets and the ACK frame. In addition, as long as there are pending data packets in the MCU, they will be uploaded to the transceiver immediately, once the transmission process is over (no matter whether a success or a failure), which is to guarantee the timeliness and reliability of data transmission.

3.3.2.2 Modeling MSP430

Compared with PIC16F, a powerful 16-bit MSP430 microcontroller is widely used in some current COTS (commercial-off-the-shelf) sensor motes such as the previous mentioned Telos, Tmote Sky and Shimmer nodes. MSP430 has a relatively low power consumption in sleep and active modes than most existing products from Microchip and Atmel [125]. Table 3.1 lists the power consumption of some of these microcontrollers.

The large on-chip RAM (up to 10KB) means that more efficient on-chip data processing can be achieved, programming code for complex applications is allowed and complex communication protocol stacks can be integrated and implemented by software alone. Therefore, nowadays a complete MAC protocol stack of IEEE 802.15.4 can be implemented in the TinyOS operating system on these popular sensor motes such as TKN15.4 [142], which integrates both slotted and unslotted CSMA/CA algorithms (used in beacon-enable and non-beacon enable modes respectively). Hence, the MSP430 microcontroller component in iWEEP_SW is modeled as a slotted and unslotted compatible FSM in the μIDLE, μTX and μRX states and is shown in

Table 3.1: Comparison of low power consumption MCUs

MCU	Active (mA)	Sleep (μA)	RAM (KB)	Flash (KB)
Atmel ATmega128 [23]	5∼17	5∼25	4	128
Atmel ATmega103L(AVR) [135]	5.5	1	4	128
Atmel AT91FR40162S(ARM) [136]	25	400	256	2048
Intel XScale PX A2/X [125]	39	574	256	N/A
Motarola HCS08 [125]	6.5	1	4	60
Cypress CY8C29666 [137]	8∼14	5	2	32
Freescale M68HC08 [138]	10∼13	22	2	61
Semtech XE8802 [139]	0.3	1.9	1	22
Microchip PIC18LF8722 [140]	6.0	2.32	3.9	128
Microchip PIC16F87/88 [129]	1.5	0.1	0.368	7
Microchip PIC18F2525 [141]	1.3∼3.0	0.1	3.968	48
TI MSP430F1611 [15] [128]	1.8	5.1	10	48

Fig.3.8.

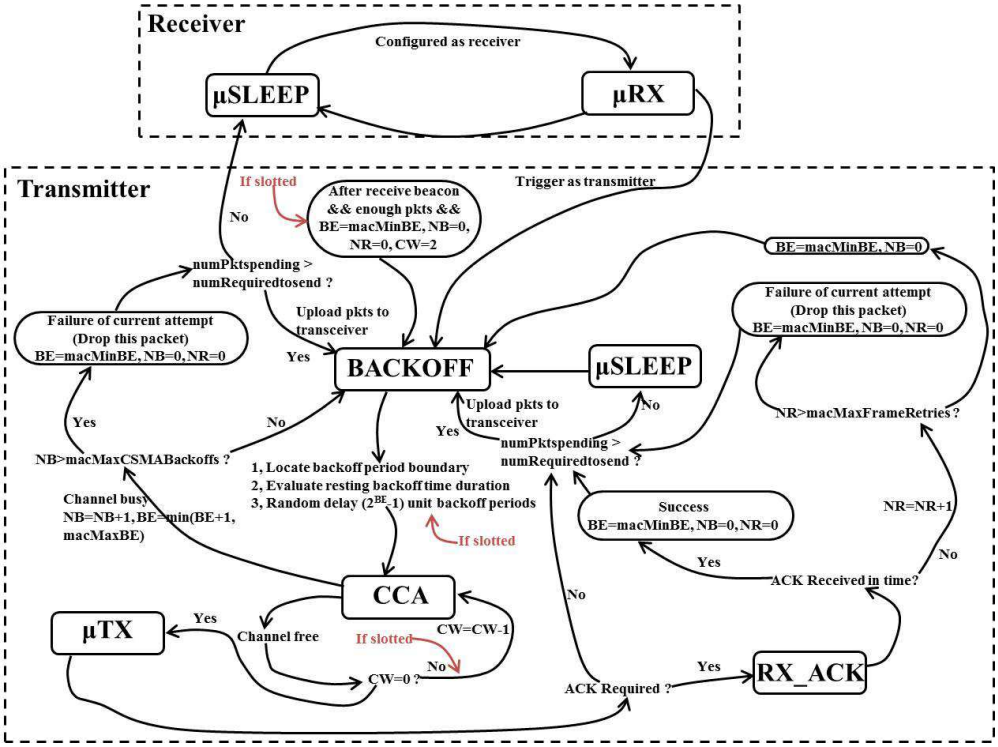

Figure 3.8: Slotted and unslotted CSMA/CA model

Compared with the unslotted algorithm, the slotted CSMA/CA algorithm needs to be accurately timed, where a beacon packet broadcast by the coordinator at the beginning of the defined superframe [20] time period is used for synchronization between coordinator and sensor motes. After the reception of the beacon packet, sensor motes with enough packets will start a slotted-based CSMA/CA algorithm. This requires

that the backoff period boundaries should be aligned with the superframe slot boundaries (simulation logs starting with '#####' in Fig.3.14 show that the backoff period boundaries are accurately aligned with slot boundaries). Further, before each new backoff period there is a time evaluation process of checking whether remaining CSMA/CA can be undertaken before the end of the contention access period (CAP) (2.8). If the number of backoff periods is less than the remaining number of backoff periods, the slotted algorithm will continue this backoff delay under the condition that two CCA analyses, the frame transmission and acknowledgement can be completed before the end of the CAP. Otherwise, it must wait until the start of the next superframe CAP. If the number of backoff periods is greater than the number of remaining backoff periods, the slotted algorithm shall first pause the backoff countdown at the end of CAP and then resume it at the beginning of the CAP of next superframe. Except for the aforementioned time evaluation mechanism, another difference between the slotted and unslotted algorithms is that two idle channel checks before packet transmission are needed for the slotted algorithm. For CAF and CF, both algorithms follow the same mechanism.

3.3.3 Transceiver Modeling

The behavior of the RF transceiver is handled via two interfaces. As mentioned previously, the *SPIdata_µsend_if* interface is defined on the transceiver component and consists of a single transaction *SPIdata_µsend*, which manages events from the microcontroller such as receiving configuration commands and payload data. The *Netdata_rInput_if* interface defines the transaction *Netdata_rInput* that allows the transceiver to detect network related events such as the validation of incoming packets from the RF channel, and the transaction *Netdata_channelstatus* can perform CCA operation. If the communication protocol is embedded in hardware, then some protocol-related timer transactions are also required such as ACK waiting transaction and auto retransmission delay (ESB mode). A typical transceiver is modeled shown in Fig.3.9.

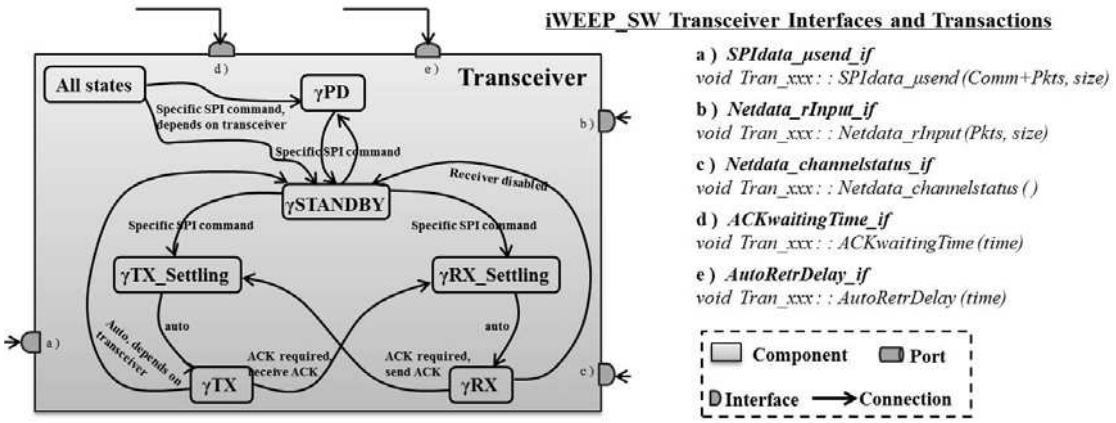

Figure 3.9: Typical transceiver model

In γPD, the transceiver operates with minimum current consumption with active SPI for register configurations. After the related register is set by specific SPI commands from the microcontroller, the transceiver will wait for one period of the crystal oscillator start-up time and then enter into γSTANDBY (γIDLE). γTX

and γRX states are triggered by commands from the microcontroller, and a chip enable (CE) signal is also needed for some specific chips such as nRF24L01 and nRF24L01+. Their transition states γTX_Settling and γRX_Settling are responsible for the preparation and activation of some internal components of the transceiver to enable transmission and reception functions. If ACK is requested by a received data packet, the transceiver can go to γTX from γRX to return the ACK frame.

To be a generic model, this model also inherits the characteristic of flexible energy consumption evaluations at different abstraction levels. If bit/cycle-accurate energy evaluation is required, an SPI transaction can be added with a bit-based communication mechanism; while the network transaction level can also be extended to incorporate detailed CRC implementation, bit stream based packet reception and validation.

In addition, another feature in the transceiver model is that its configuration is based on register settings like real hardware, which means that related registers are defined in each specific transceiver model. They can either be defined as *char* type, *short int* type or *int* type, depending on their actual size in hardware. The register configuration process is based on bit operation as real hardware does (the way of changing one or several bit values in the specific register of hardware). These defined registers can be set as default values and they can also be set as user required values during the SystemC elaboration process, or they can be configured by corresponding commands from the microcontroller via an SPI transaction during the SystemC simulation process. Besides, for the convenience of register configuration, a configuration library is built for each transceiver in which are defined, by hex, or binary values in straight C code: register address information, field information within the register, register default value, definition of some commands and register bitwise definitions. Therefore, it also works well with any C compiler in hardware development. Using the nRF24L01+ transceiver as an example, Fig.3.10a) is the register data structure defined in its transceiver model, and Fig.3.10b) shows a small part of the configuration library.

```
//Definition of nRF24L01+ registers
class Transceiver_NRF24L01p : public Scnsl::Transceiver_t
{
        public:
        ..............................
        ..............................
        unsigned char config;
        unsigned char en_aa;
        unsigned char en_rxaddr;
        unsigned char setup_aw;
        unsigned char setup_retr;
        unsigned char status;
        ..............................
        ..............................
};
```

a)

```
#ifndef NRF24L01P_DEF_HPP_
#define NRF24L01P_DEF_HPP_

//SPI command defines
#define nrf24l01p_R_REGISTER    0x00
.....................................

//register address
#define nrf24l01p_SETUP_AW      0x03
.....................................

//register default value
#define nrf24l01p_STATUS_DEF    0x0E
.....................................

//register bitwise definitions
#define nrf24l01p_EN_AA_PIPE5   0x20
.....................................

#endif
```

b)

Figure 3.10: Register definitions and configuration library

With the register configuration mechanism, the simulation of the network becomes flexible because different nodes could keep their own settings during the whole simulation process shown in Fig.3.11b).

The final results could be acquired from each node to evaluate the performance of such a heterogeneous

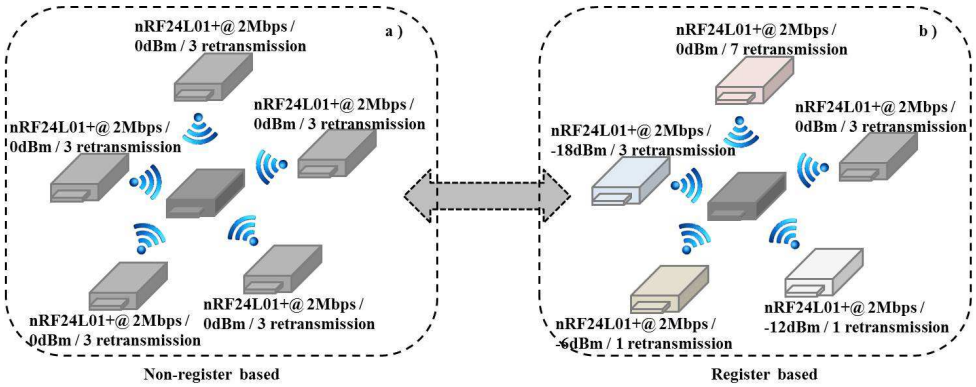

Figure 3.11: Advantage of register configuration mechanism

configured network scenario.

3.3.3.1 Modeling of CC2420

The CC2420 is a 2.4 GHz IEEE 802.15.4 compliant RF transceiver chip with a maximum data rate of 250 Kbps. CC2420 is designed to support low power applications, it provides on-chip packet handling by automatically adding a preamble sequence, frame check sequence and frame delimiter to the data packet. It is functional for data buffering, clear channel assessment, link quality indication and packet timing information, all of which reduce the load and power consumption on the host microcontroller. Since for CC2420 based sensor motes (like Telos, Tmote Sky and Shimmer), MAC algorithms for channel access are programmed by software and embedded in the microcontroller, the CC2420 acts only as a TX/RX device with transmission and reception of the IEEE 804.15.4 format based data packet. Therefore, the model of CC2420 in iWEEP_SW shares the generic model presented in Fig.3.9.

3.3.3.2 Modeling of nRF24L Transceiver Series

Designed by Nordic Semiconductor, the nRF24 transceiver series [122] represent ultra-low-power RF chips that provide available solutions for 2.4 GHz ISM band wireless applications. The nRF24 transceiver series can be further divided into two sub-families nRF24x and nRF24L. The nRF24x family includes nRF2401A [143], nRF2402 [144] and nRF24E1/2 [145], all of which only support ShockBurst (SB) [21] with a maximum data rate of 1Mbps (nRF2402 and nRF24E2 they are only transmitter chips). Compared with the nRF24x family, the nRF24L family includes more recent Nordic transceiver chips such as nRF24L01 [146], nRF24L01+, nRF24LE1 [147] and nRF24LU+ [148], all of which support Enhanced ShockBurst (ESB) [21] with a maximum data rate of 2Mbps. Designed by Nordic, Enhanced ShockBurst is a baseband MAC protocol engine embedded in transceiver chips, while ShockBurst is only a simple TX/RX protocol without any channel access algorithm, using the packet format presented in Fig.3.2 for transmission and reception.

In this section, the transceiver nRF24L01+ used in iHop@Node is taken as an example for modeling. The first reason for this choice is the hardware supported automatic packet assembly (automatic assembly of preamble, address, packet control field and CRC) and auto-acknowledgement, which can reduce load

and average current consumption of the host microcontroller. Indeed, the nRF24L01+ is ultra-low power consumption as compared to most existing widely-used transceiver chips shown in Table 3.2 (section 3.4.3). The second reason is that it supports a high data rate up to 2Mbps which can guarantees the high performance of the network. Thirdly, it provides configurable packet size, output power and data rate as well as configurable ESB-related parameters, all of which provides potential solutions for specific application scenarios. Finally, it is a transceiver chip embedded with both ESB and SB protocol engines, which makes it a good example for generic ESB/SB model development. Therefore, a universal hardware embedded ESB and SB protocol has been modeled in iWEEP_SW by a finite state machine shown in Fig.3.12.

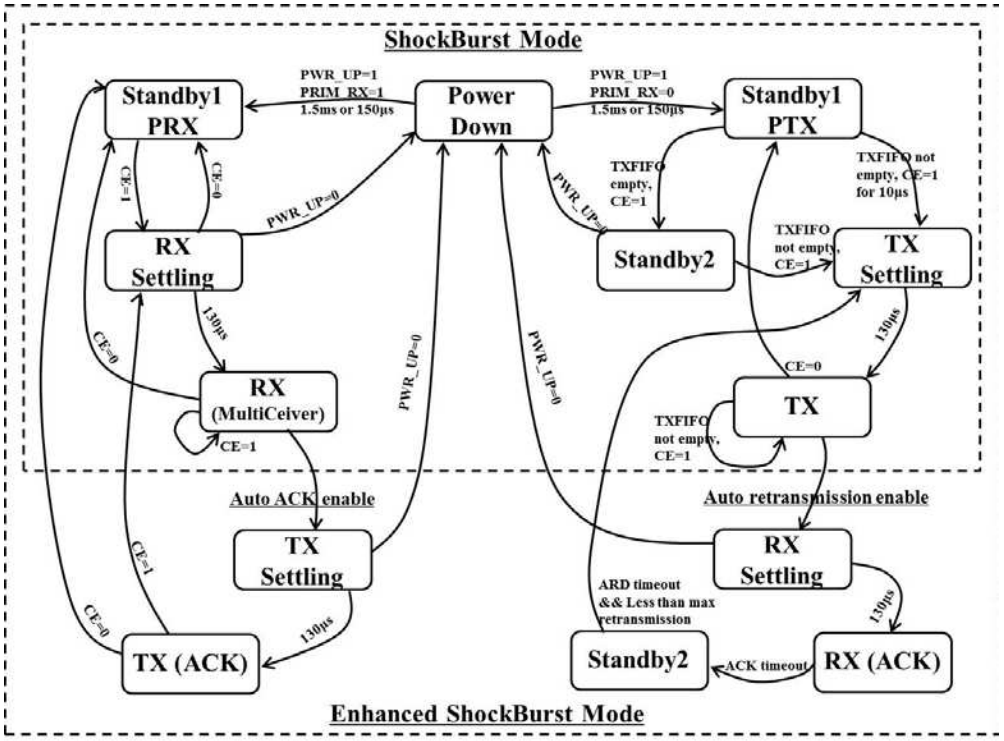

Figure 3.12: Enhanced ShockBurst and ShockBurst model

The transceiver is woken up from Power Down (PD) state to Standby1 PTX (Primary Transmitter) state or Standby1 PRX (Primary Receiver) state by the microcontroller's SPI configuration command. After the chip enable signal CE is set high at least 10μs, and with data packet in TX FIFO buffer, then the transceiver will be triggered from Standby1 PTX state to TX state for packet transmission. Before entering into the TX state, the transceiver should be in TX_Settling transition state for 130μs to complete packet data assembly and activate internal components for over the air transmission. If the TX_FIFO is empty when CE is set, the transceiver will go into Standby2 state, waiting for the incoming packet payload. Once a new packet payload is uploaded to the TX_FIFO and CE is held high, transmission will start after 130μs settling time. The transceiver will generate a transmission interrupt (PTX_IRQ) to the microcontroller after the completion of packet transmission if the ESB mode is not activated, and in such a case transceiver will go back to Standby1 PTX when CE is set low. Otherwise, the transceiver will stay in TX state for the

next packet transmission if TX_FIFO is not empty, or it will return to Standby2 state waiting for a new payload to be uploaded into TX_FIFO. For the PTX device, the ESB mode is activated by enabling the auto-retransmission function, so after the finish of packet transmission, the transceiver can automatically enter into the RX state for the reception of ACK frame. If the ACK frame is acknowledged within the maximum waiting time (250μs for 2Mbps and 1Mbps cases, and 500μs in the 250Kbps case), the transceiver will return to Standby1 PTX state. If not, the transceiver goes into Standby2 state for energy saving and waits to start a retransmission process until the Auto Retransmission Delay (ARD) time elapses. For the PRX device, CE must be kept high to guarantee that the transceiver stays in the RX state for channel listening and packet data reception. In the RX state, the PRX device can perform a similar CCA operation using its Receive Power Detector (RPD), which is a signal that is set high when an RF signal is detected inside its receiving frequency channel. The status of RPD can be read out as a correct value when the RX mode is enabled after a wait period of 130μs + 40μs, which offers a snapshot of the current received power level in the channel. PRX also has a MultiCeiver [21] function enabled in the RX mode, which allows one PRX device to receive data addressed to six different data pipes in one frequency channel. If the auto-ACK function is activated, the PRX device can switch to the TX state for sending back the ACK frame to the corresponding PTX after the received packet is validated. After this, the transceiver may either stay in the RX state or just return to Standby1 PRX, depending on the status of the CE signal.

Like nRF24L01+, other transceiver models in the nRF24 series can be easily added to iWEEP_SW just by making some simple modifications on the state transition time, the energy consumption values of different states, and the configuration information of the corresponding registers. To sum up, this generic model can be easily adopted by all transceivers in the nRF24 series as long as they are ESB/SB driven, and it helps to perform a wide range of energy consumption evaluations for ESB/SB based sensor mote, products and applications.

3.3.4 Network Modeling

The network model is derived and modified from the SystemC Network Simulation Library (SCNSL) [89]. In network modeling, SystemC provides primitives such as concurrency mechanism as well as events that are used to simulate network transmission and reception behaviors. Before the start of simulation (in the elaboration process), *nodeProxy* class objects are instantiated with the same number of node instances in the SystemC entrance function *sc_main()*. The use of *nodeProxy* is to decouple each node's implementation of the network simulation, and each *nodeProxy* instance uploads some significant information to the network object, including node identity, two-dimensional position information, TX output power and receiver sensitivity. Combined with the channel model, these parameters are used to reproduce the network scenario and perform network connection assessment. At present, two types of channel model are supported in iWEEP_SW, which are free space propagation model [149] and human's on-body **L**ine **O**f **S**ight channel model (LOS, in section 4.4). The most fundamental free space model in iWEEP_SW is presented as follows.

$$P_r = \frac{P_t \cdot G_t \cdot G_r \cdot \lambda^2}{(4\pi D)^2} \tag{3.1}$$

where P_r is the receiver power sensitivity measured in dBm, P_t represents TX output power at transmitter,

D is the distance (m) between the transmitter and receiver in two dimensional space, G_r and G_t are antenna gains of the receiver and transmitter respectively (it is common to select G_r=G_t=1), and λ denotes the wavelength, calculated as $\lambda = \frac{c}{f_c} = \frac{3*10^8(m/s)}{2.4*10^9(Hz)} = 0.125(m)$.

When simulation is initiated by *sc_start()* function, the over-the-air transmission time of the data packet and ACK frame will be calculated in the network model, and a waiting period is thus required for each transmission. After this time, the network model will distribute the packets to the receiver nodes within the radio transmission range. However, if collisions occur, no packets will be received. The collision scenario is simulated by checking the numbers of active transmitters that are interfering at a given receiver. If the number is greater than one, then packet transmission is considered as a failure because of collision.

3.3.5 Sensor Modeling

In order to adjust energy evaluation requirements for multiple levels, in iWEEP_SW the sensor component is allowed to be modeled at various abstraction levels ranging from a generic sensor model at the high-level, to a specific sensor model integrating device driver code for a physical sensor. In this work, sensor component modeling is mainly focused on generic model development consistent with the high-level design philosophy of iWEEP_SW.

A generic sensor model can be designed in 3 ways. The first is periodic data generation, which samples the signals of the physical environment by an application defined time interval (sampling period). The second is to read data from an input file where this file can be a designer-specific data input file at the early design stage, if the actual sensor has not yet been chosen. This input file can also contain actual measurements from experiments. The third way is to model the sensor in a specific function. In [90] a temperature sensor component *ExpSensor*, used in a fire detection and tracking application, is modeled as an exponential function based on the node's two-dimensional physical location. The received data from each sensor mote can be stored separately and compared with original data to evaluate network performance metrics such as network delivery rate and congestion performance.

3.3.6 Energy Modeling

A significant part of iWEEP_SW, the energy model plays an important role in energy consumption calculation and it must therefore be well designed and optimized to adjust accurate, realistic and flexible energy consumption evaluation. The following sections illustrate the design method of the energy model.

Elaborate Lib and Register Based: An energy model is developed to incorporate an elaborate library that includes the current consumption of different operation modes of each hardware component. The working mechanism of this energy model is based on register configurations. Most changes in hardware operation state are triggered by the configuration of specific registers, such as the five low power modes of MSP430, TX output power and data rate selection of nRF24L01. Immediately or shortly after the configuration of a specific register changes, the hardware operation state changes depending on the current status of the hardware. For example (operation state changes after a short period of time), the register configuration on the nRF24L01+ transceiver is available at any time, but the change can trigger a new operation state only when it is in power down mode or standby mode. In the iWEEP_SW energy model,

the hardware registers of each component are mapped into the energy model as the mirrors, and the update of their values is synchronous with the corresponding register prototypes in related hardware components, as shown Fig.3.13. Developers and researchers can benefit greatly from this design method. Since the index (tracing) of the correct energy value is always consistent with the change of component configuration during the whole simulation process, the energy model is flexible enough to adjust scenarios in which hardware configurations need to be changed during the simulation. In that sense, the energy model is able to proceed without any interruption to the whole simulation process, and also no re-compilations are required for new scenarios after configuration changes. Therefore, energy evaluation for complex sensor networks is well supported, such as when using dynamic networks and adaptive communication protocols, where the configuration of specific components must be changed continuously according to current network conditions to guarantee an optimized performance.

Energy Evaluation for Multi-abstraction levels: The energy model in iWEEP_SW is able to trace the energy consumption at different levels of abstraction. If the energy evaluation is focused at the high-level, the change in operation state of each hardware component is uploaded to the energy model, and then the duration and current consumption of each state can be identified for the energy consumption calculation by applying the basic formula as follows.

$$E = P * t = (V * I) * t \tag{3.2}$$

If the energy evaluation is at low-level, the energy model can track each executed instruction in different operation states, record their execution time and associate them with corresponding current loads in the energy library for accurate energy estimation. Compared with energy evaluation at high-level, the use of the same calculation mechanism at the low level is also possible, because the real measurements on instructions from [39] and [19] show that commonly-used instructions, such as AND, OR, XOR, ADD, MUL, DIV, FOR loop and WHILE loop, consume similar power levels in both MSP430 and PIC16. Thus, the energy consumption in a specific time period can be easily and directly calculated by (3.2), where both voltage (V) and current (I) are steady. The energy consumption for instruction execution is thus linearly proportional to execution time.

Calibrated Energy Support: The energy model incorporates calibrated energy values from real-world measurements for realistic and accurate energy evaluation. These measurements can be acquired from any COTS sensor motes or in-house testbed mote. The power consumption figures are not easy to extract from product specifications, since a sensor mote is composed of various chips and some chips need additional peripherals to guarantee operation. These peripherals on chip modules and sensor mote board can cause extra energy consumption (e.g., nRF24L01+ module in section 3.4.6). Thus, the calibrated energy model in iWEEP_SW is necessary.

Transition State Energy Consumption: The energy model considers state transitions in each hardware component, because each transition introduces additional cost to the overall energy consumption that cannot be ignored. So the total energy consumption is calculated by adding the individual energy consumption of each operation state of every hardware component as well as the energy consumption of each transition state of every hardware component. The detailed energy calculation formula is presented in Fig.3.13. Compared with the energy calculation in [97], this calculation is extended to support energy e-

valuations at different abstraction levels in iWEEP_SW. At the high level, both t_{ijk} and t_{ijl} represent the time duration of each operation state and transition state respectively, while at the low level, t_{ijk} and t_{ijl} can represent the time units of executing an instruction or the handling of one bit of data.

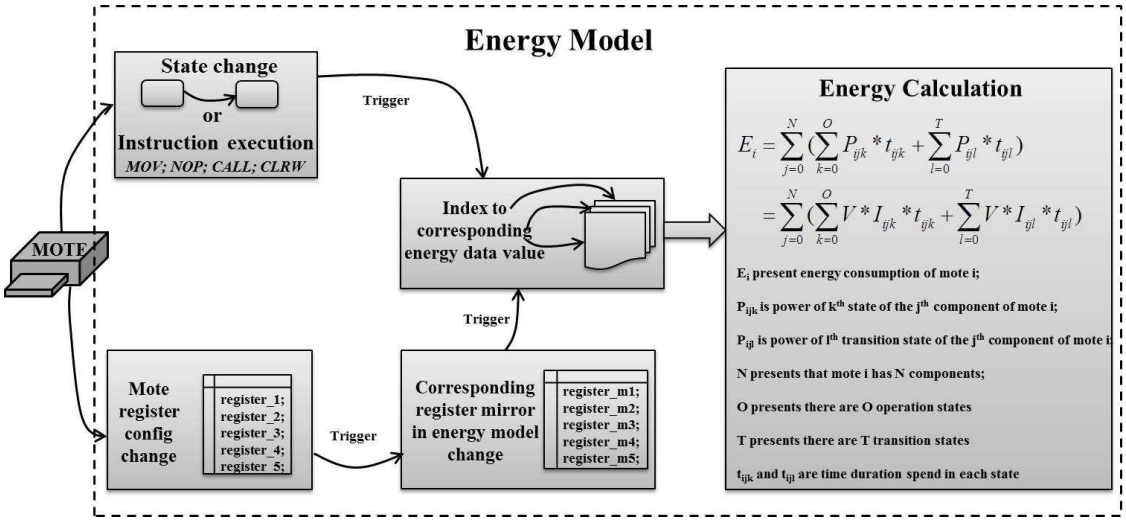

Figure 3.13: Workflow of energy model

Multi Performance Metrics: The energy model supports various performance metrics and can be calculated in *mW*, *mJ*, *mAh*. If given a specific battery module measured in *mAh*, then lifetime performance can be estimated in minutes, hours, days, months and years via the following calculation.

$$LifeTime = \frac{Cap}{Av\_power * 24}(days) \tag{3.3}$$

where *Av_power=EnSim / (SimT * 1000 * 3600)* is the average power consumption of the sensor node per hour, *SimT* (*ms*) is the defined simulation time, *EnSim* (*mAh*) presents the total energy consumed in the simulation process, and *Cap* (*mAh*) denotes the capacity of the battery module.

3.3.7 MATLAB based iWEEP_SW_GUI

The graphical user interface iWEEP_SW_GUI is designed for iWEEP_SW to simplify sensor network configurations and help researchers and developers quickly access and explore various kinds of network scenarios with more efficiency by using iWEEP_SW.

Compared with JAVA-based GUI [7] and QT-based GUI [97], iWEEP_SW_GUI is developed under MATLAB. The design of such a GUI in MATLAB has several advantages: firstly, MATLAB has very strong and flexible numerical calculation abilities, and secondly the visualization of results is easier to carry out without significant programming effort. Both advantages are necessary for subsequent data processing since many kinds of performance metrics can be acquired after simulation. In the evaluation process, all kinds of data processing functions can be easily and directly integrated into iWEEP_SW_GUI. As MATLAB has an extensive set of built-in functions as well as additional toolboxes that can provide specific solutions

for the given scenarios, so these specialized functions can also be incorporated in iWEEP_SW_GUI as the optimization strategies for various purposes such as the exploration of network performance or the tuning of network configurations. For example, in chapter 5 the optimization toolbox is integrated into iWEEP_SW_GUI and is used for the search of optimized settings of the target sensor network. With such excellent flexibility and extendability, the MATLAB-based GUI design is chosen for iWEEP_SW (and iMASKO's GUI in chapter 5) and it is shown in Fig.3.14.

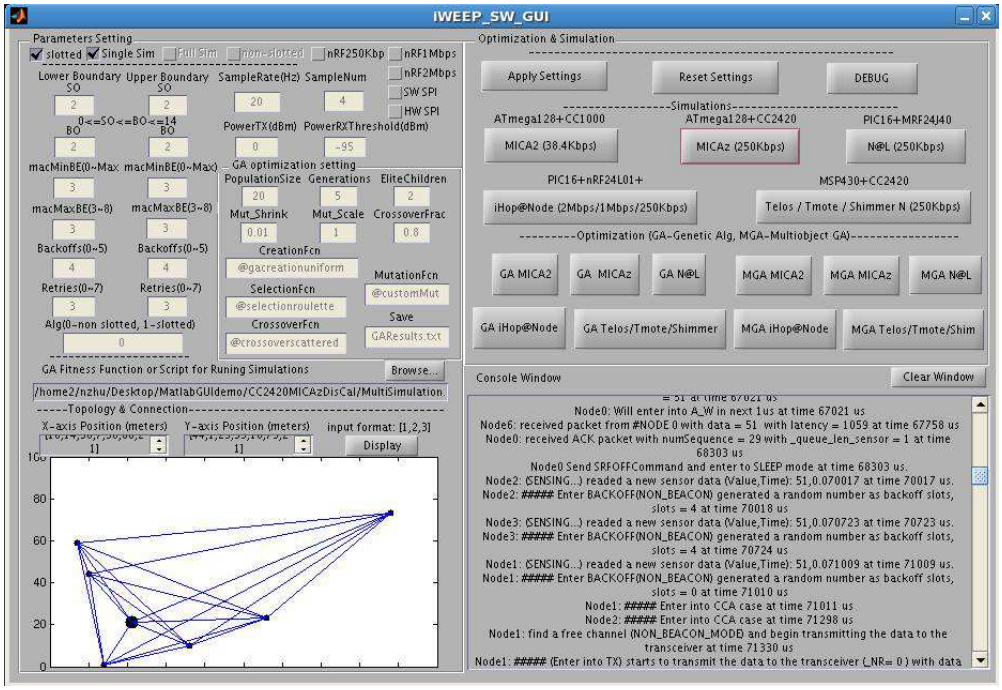

Figure 3.14: MATLAB-based iWEEP_SW_GUI

On the upper-left of the iWEEP_SW_GUI main window is the network configuration zone, where users can set various input parameters of different levels. As for the MAC layer, since the slotted and unslotted CSMA/CA algorithms of IEEE 802.15.4 are implemented by iWEEP_SW, so the parameters including *SO*, *BO* in the slotted algorithm and *macMinBE*, *macMaxBE*, *macMaxCSMABackoffs*, *macMaxFrameRetries* in both slotted and unslotted algorithms are all allowed to be set via iWEEP_SW_GUI. Application-level parameters such as sampling rate and node number can be set, and the types of the sensor mote platforms under simulation are also selectable on the mid-right of the GUI's main window. Hardware-level parameters like data rate, HW/SW SPI, transmitter TX output power and receiver sensitivity can also be configured. The parameter settings on the genetic algorithm for sensor network optimization, which contains generation size, population size, upper/lower boundaries, fitness function selection, selection function, creation function, mutation function and crossover function, will be presented in detail in chapter 5. In addition, iWEEP_SW_GUI not only supports single simulations with a specific configuration, but also supports multiple simulations by setting the parameters' upper and lower boundaries. Then simulations that cover all possible parameter combinations will be tried, so in such conditions, iWEEP_SW_GUI can

automatically provide designers a comprehensive exploration of various network scenarios. The network connection and topology display window is located at the bottom-left, where the relative positions of all given nodes and their connection status are shown. The connection between two nodes are calculated by the radio propagation model (section 3.3.4) that contains several impact factors such as node position, output power, sensitivity, etc. In the display window, each node is plotted as a black spot while the connections are indicated by solid blue lines.

After the checking network connections and finishing the network configurations, these parameters are written to an XML (eXtensible Markup Language) file or a TXT file by pressing the 'Apply' button. At the same time, all the parameter edit boxes will be disabled. If any modifications are required, the 'Reset' button can enable parameter edit boxes for the reconfiguration. The simulation and optimization process can be launched by pressing the corresponding target sensor mote button on the mid-right of the GUI window. Since the SystemC simulation of each sensor mote platform is compiled as an executable file, MATLAB is able to use a *unix command* that calls upon the Linux operating system to execute these SystemC-based executable simulation files in the shell. Once the SystemC simulation starts, the parameters will be read from the XML or TXT file to the SystemC simulation environment. In this sense, MATLAB-based iWEEP_SW_GUI is well linked to SystemC simulation.

After simulation or optimization, there are three types of simulation logs. The first one is the real-time simulation logs that are displayed in the console window of iWEEP_SW_GUI during the simulation process, which shows designer required debug information on some specific events to help designers monitor mote and network behaviors. The second type is the Value Change Dump (VCD) file, VCD file is continually updated through the simulation process and the required variables in this VCD can be displayed graphically by using waveform-viewing tools like [150]. The third type is the generated TXT file after simulation or optimization, which suit detailed evaluations very well. Based on designer requirements, the generated TXT file could be simulation logs of the whole network or specific sensor mote, or they could also be specific performance metrics such as packet loss, latency and energy consumption. Besides, the contents in the file can be written as any required formats to facilitate the later data processing by using MATLAB or other data analysis tools like Origin [151].

3.4 iWEEP_HW

3.4.1 iWEEP_HW Overview

Due to their reliance on batteries, energy consumption has always been of a significant concern for sensor node networks. In recent years, many energy-efficient protocols and energy-aware applications have been proposed. However, due to the lack of real-world measurement tools, most of these protocols and applications are based on idealized and unrealistic theoretical analysis or simplified simulation models, which lead to erroneous energy consumption predictions. Therefore, the energy sources of the sensor network may be exhausted before completion of required tasks. Even if energy-efficient protocols and applications have been prototyped in testbeds, only some simple metrics such as packet loss rate, source-to-sink latency or network throughput can be determined and measured, while the measurements of energy consumption on the sensor mote is still a challenge: expensive high-resolution digital multimeters and costly oscilloscope need to be

connected to the sensor motes to sample the varying low currents and voltages [152].

The design and implementation of iWEEP_HW provides a solution with cost-effective, side-effect free and flexibility to access reliable, precise and detailed information on energy consumption without using high-frequency multimeters and oscilloscopes. iWEEP_HW is a lab-built real-world energy measurement system that mainly consists of a hardware measurement platform named MEMD (**M**ultichannel **E**nergy **M**easurement **D**evice) and an energy data management platform called EDMSP (Energy Data Management Software Platform). iWEEP_HW can provide realistic and accurate measurements for the target sensor motes, and it can also calibrate and validate energy models and node operation models from iWEEP_SW to achieve a much more reliable and realistic energy performance evaluation.

3.4.2 Experimental Methods for Measurements

Several current experimental methods available for measuring the energy consumption of wireless sensor networks are listed and described in this section. Four types of the methods are commonly used: (a) the clamp-on current probe based method [153], (b) the widely-used shunt resistor and amplifier based method, (c) SPOT (scalable power observation tool) based method [154], and (d) SuperCaps based lifetime estimation method [155]. The first three methods provide real-time energy data monitoring while the last one is used for predicting network longevity. Fig.3.15 presents these four types of measurement methods.

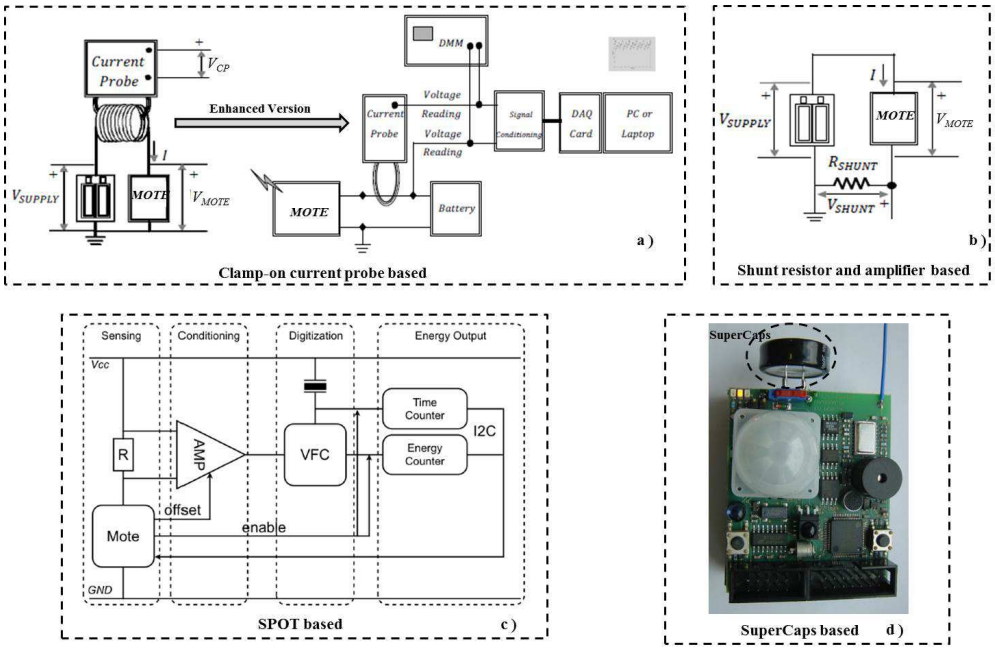

Figure 3.15: Wireless sensor node measurement technologies

3.4.2.1 Clamp-on Current Probe Based Method

As shown in Fig.3.15a) [153], a non-contact current probe is used in this method between the power supply and the target mote under test. The output voltage from current probes is sampled, and is usually a linear

function of current through the clamp (equation 3.4). The use of the current probe makes the method unobtrusive.

$$P_{MOTE} = V_{MOTE} * I = V_{SUPPLY} * I, I = f(V_{CP}) \tag{3.4}$$

An enhanced version of current probe measurement is also presented in Fig.3.15a) by A. Milenkovic et al. [153]. Compared with the previous approach, in this method the linear dependency between current load and voltage of the current probe is calibrated by using a simple circuit with the power supply and a resistor. The current load I_{TEST} and power supply V_{TEST} are measured for a range of resistors with resistances from 70Ω to 3KΩ by the digital multimeter. The results show that instead of linear dependency, current and voltage can be better approximated with the following second order polynomial and the results are shown in Fig.3.16.

$$I(mA) = -20 * {V_{CP}}^2(mV) + 99.1 * V_{CP}(mV) + 0.1576 \tag{3.5}$$

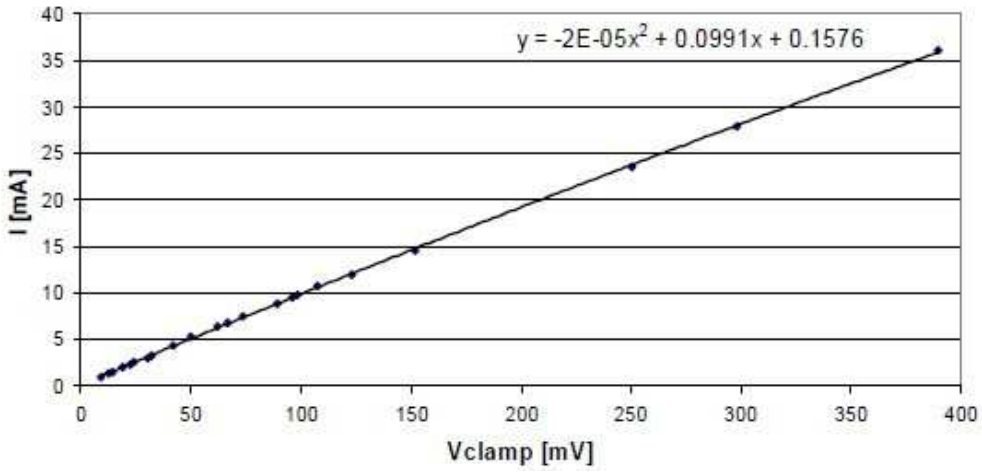

Figure 3.16: Current probe calibration [153]

The combination of the current probe (Extech 380946) and data acquisition card (DAQCard-AI-16XE-50) in this approach guarantees high accuracy and sampling rates. However, with the current price of over $150 for the current probe and about $400 for the acquisition card, this approach cannot be considered as a cost-effective means for sensor mote measurements.

3.4.2.2 Shunt Resistor and Amplifier Based Method

This is a widely used method for measurement testing and its basic idea is to place a low-impedance shunt resistor between the power supply and target sensor mote which is shown in Fig.3.15b). The voltage drop across the shunt resistor is proportional to the current flowing through this resistor. Therefore, the current consumption of the target sensor mote can be calculated by the measured voltage.

$$P_{MOTE} = V_{MOTE} * I = (V_{SUPPLY} - V_{SHUNT}) * \frac{V_{SHUNT}}{R_{SHUNT}} \tag{3.6}$$

The advantage of this approach is that it is less susceptible to noise (few components, low resistance and low thermal noise). Thus, it has been applied by many studies. However, costly measurement equipment (oscilloscope, acquisition card) has limited its large-scale deployment.

3.4.2.3 SPOT Based Method

Designed by J. Xiaofan et al. [154], with an architecture as presented in Fig.3.15c), this SPOT (Scalable Power Observation Tool) based method uses the standard current sensing technique that places a shunt resistor between the power supply and the target sensor mote. The voltage drop across the shunt resistor is amplified and sent to a voltage-to-frequency converter (VFC) for analog to digital conversion. With a large dynamic range of 10000:1, it is able to precisely monitor tiny sleep currents. However, there are some disadvantages despite its good accuracy and high sampling rate. Firstly, it is more costly and complex than the simple shunt resistor and amplifier based solution, which makes it difficult for wide deployment. Secondly, measurements need to be processed by the host sensor mote that the SPOT board is attached to, so the normal operation of the mote can be affected during the SPOT processing stage. Thirdly, the measurements can be interfered due to the noise introduced by the oscillator and VFC.

3.4.2.4 SuperCaps Based Method

This method, proposed by H. Ritter et al. [155], is adopted mainly for lifetime estimation rather than real-time energy data monitoring. A real experimental prototype is shown in Fig.3.15d). Since some energy-effective protocols, communication strategies and applications might be functional for weeks or months by using batteries or rechargeable batteries, it is time consuming and impractical for short term experiment validation and comparison, and it is also less efficient in providing appropriate data to mathematical models for lifetime prediction. Therefore, the SuperCaps method is introduced by this approach. The SuperCaps device has a high capacity (up to the order of 1F) and can be charged very quickly to power a sensor mote for a reasonable amount of time T (e.g., tens of minutes) as shown in equation (3.7). This speeds up the energy evaluation process, while realistic and reliable results can also be guaranteed with real world measurement data.

$$T = C * (U_{max} - U_{min})/I \tag{3.7}$$

Here, U_{max} is the maximum voltage of SuperCaps, U_{min} is the minimum voltage (no operation below U_{min} possible), C denotes the capacity of the SuperCaps, and I represents the current load of the target sensor mote. However, this method only provides the overall energy usage and lacks detailed energy information. This cannot therefore provide comprehensive insights for the design of energy-aware protocols and application scenarios.

3.4.2.5 Summary of Measurement Methods

To sum up, the shunt resistor and amplifier based methodology is considered as the cleanest way for energy measurement, as it does not introduce any side-effects on the target sensor mote system either in software or hardware. A number of research works are based on this method and extend this method for more accurate energy measurement. Leonardo et al. [156] used a high side current shunt monitor (INA139) to amplify the voltage across the shunt resistor. Besides, a charge counter was employed to characterize the consumed charge of the node, and it also expressed this charge information in the units of mAh or μAh to flow from the battery into the MICAz mote. Moslem [157] used an oscilloscope (Tek TDS2012B) to monitor the amplified shunt resistor voltage (AD620AN) for the measurements of Tmote Sky mote. Rachit et al. [158] adopted an acquisition card (NI PCI-6071) to collect shunt resistor voltages and calculate the current consumption for a ZigBee based sensor mote. Since expensive experimental equipment like oscilloscopes and acquisition cards, costing over $500, were used in the above three works, they reduce the chance of potential wide deployments.

Leo et al. [159] implemented an external power-meter module to measure the consumed power of a MICAz mote. The voltage across the shunt resistor was amplified by a current sense amplifier (LT1787) and was converted to a digital value by the built-in ADC of MICAz's ATmega128. The energy results could be displayed graphically on an LCD screen or stored in the external memory. Ivaylo et al. [160] designed a single-chip successive approximation ADC solution. This integrated a shunt resistor / amplifier (INA122) based method into the testbed, and the results were sent to an embedded Linux platform for subsequent evaluation. Without using costly measurement tools, the above two methods are cost-effective. However, there are still drawbacks. Firstly, each energy test module can only be attached to one mote for the measurements, while sometimes multi mote monitoring is needed for specific energy efficient protocols or applications. Secondly, energy measurements are for the entire sensor mote without any detailed energy information on each hardware component. This is because every hardware component is integrated onto a main board and supplied by the same power source. In addition, for [160] there might be some side effect since the testbed mote and energy measurement module are designed on one single-chip.

In order to acquire a thorough insight of detailed energy information on each hardware component, according to [39], the decomposition of the mote system into main functional components by software provides a good solution.

iWEEP_HW is designed and implemented as a cost-effective system platform to overcome the aforementioned drawbacks in current sensor mote energy measurement methods. iWEEP_HW consists of a hardware energy measurement platform MEMD (**M**ultichannel **E**nergy **M**easurement **D**evice) and an energy data management platform EDMSP (**E**nergy **D**ata **M**anagement **S**oftware **P**latform), as well as a high data rate and ultra low power testbed iHop@Node with a multi-layer architecture to provide elaborate energy consumption information on each hardware component. In the following sections, the detailed design and implementation of iWEEP_HW will be presented.

3.4.3 iHop@Node Testbed Node

iHop@Node (**i**NL **H**igh data rate and ultra l**o**w **p**ower testbed **Node**) shown in Fig.3.18b) is our lab-built testbed node that is mainly composed of a power efficient Microchip PIC16F88 [129] microcontroller with 0.93∼1.30 mA current load in active mode and 0.3∼0.5 μA in sleep mode, and high data rate (2Mbps/1Mbps) low power transceiver Nordic nRF24L01+ [21]. Table 3.2 provides a comparison of power consumption figures between the nRF24 transceiver series and other widely used transceivers. With such a low-energy microcontroller and transceiver, the whole iHop@Node testbed is ultra low power consumption.

The design of iHop@Node is for three reasons. The first is that high data rate based iHop@Node can provide better network performance (e.g., packet delivery, latency, etc) than most existing low rate based motes, and it is the tendency for the future sensor network [167]. Second, due to its low power consumption, iHop@Node is a good prototype for energy-efficient research exploration (e.g., energy-saving protocols, strategies and applications), since energy has always been a great concern in sensor networks. Finally, iHop@Node is the target mote built under the framework of iWEEP_HW, which validates the availability and usability of both MEMD and EDMSP. The following Table 3.3 lists some characteristics of iHop@Node.

Besides, it has a small-dimension of 4.1" * 2.4" which can be used as the wearable mote for medical/health applications in wireless body area network (WBANs) [168] which is an emerging sub-field for WSNs. It can also be used as a tiny WSN mote for generic purposes such as temperature, sound, vibration, humidity, etc. Another feature of iHop@Node is its multi-layer architecture design, which means that the microcontroller layer, transceiver layer, sensor layer and other components layer (e.g., LEDs, external memory) are all separate and can be fed either by the same power supply or by different supplies. When different power supplies are adopted, each hardware component is considered as an independent load that is under test and detailed energy information can be measured respectively and accurately from each of these components synchronously by using MEMD. Compared with the software based functional components decomposition approach [39], this means is at hardware level and provide much more reliable results.

3.4.4 MEMD

A dedicated Multichannel Energy Measurement Device (MEMD) is designed and implemented to measure and retrieve sensor mote energy consumption. Fig.3.17 provides the schematic of one channel of MEMD.

A shunt resistor is placed in series between the power supply and the sensor mote, and the voltage drop across this resistor R1 is amplified by a high-side current-sense amplifier U1A (MAX4172) [169] with a gain of 77.3. A voltage divider, consisting of a voltage follower U2A and U2B (LM358) [170], is used to expand the test range of the amplified output voltage. The voltage is then fed to a 10-bit ADC U3 (ADC10664) [171] that is sampled and held by an internal circuit. A microcontroller controls the analog to digital conversion, reads the converted sample data and saves them into the buffer. Finally, the buffered data can be sent out to data terminal (e.g., PC or server) in two ways. First, the sampled data can be saved continuously in the whole buffer space and then sent out in a buffered sampling mode. Alternatively, the

Table 3.2: Comparison of low power transceivers

Radio	**Wireless (Tech)**	**Data Rate (Kbps)**	**Band (MHz)**	**Sleep (μA)**	**RX (mA)**	**TX (mA)**	**RX (nJ/bit)**	**TX (nJ/bit)**
MRF24J40 [128]	802.15.4	250	2400	2	19	23	250	303
CC2420 [25]	802.15.4	250	2400	1	18.8	17.4	248	229
CC1000 [24]	N/A	76.8	433∼915	0.2	7.4	10.4	319	446
CC1100 [161]	N/A	250	433∼915	0.4	16.4	16.9	216	223
CC2500 [162]	N/A	500	2400	0.4	17.0	21.2	112	139
CC2480 [163]	ZigBee	250	2400	0.75	26.7	26.9	352	355
TR1001 [164]	N/A	115.2	868	0.7	3.8	12	108	343
TR3100 [165]	N/A	576	433	0.7	7.0	10	40	57
nRF905 [166]	Nordic SB	50	433∼915	2.5	14.0	12.5	924	825
nRF2401A [143]	Nordic SB	1000	2400	0.9	19.0	13.0	62	42
nRF24L01 [146]	Nordic ESB/SB	2000	2400	0.9	12.3	11.3	20	18
nRF24L01+ [21]	Nordic ESB/SB	2000	2400	0.9	11.3	13.5	18	22

Table 3.3: iHop@Node characteristics

Characteristics:
√ Automatic packet assembly and disassembly
√ Automatic retransmission with delay from 250μs∼4000μs
√ Automatic acknowledgement mechanism
√ Configurable packet format: address length, CRC, payload and packet control field
√ Configurable output power: 0dBm, -6dBm, -12dBm, -18dBm
√ Configurable data rate: 250Kbps, 1Mbps, 2Mbps
√ Configurable carrier frequency: 2.4GHz∼2.525GHz
√ Configurable automatic retransmission: 0∼15 times
√ MultiCeiver function: a PRX device can receive data from six PTX devices
√ HW / SW SPI communication supported
√ DC supply powered / 2 AA battery powered

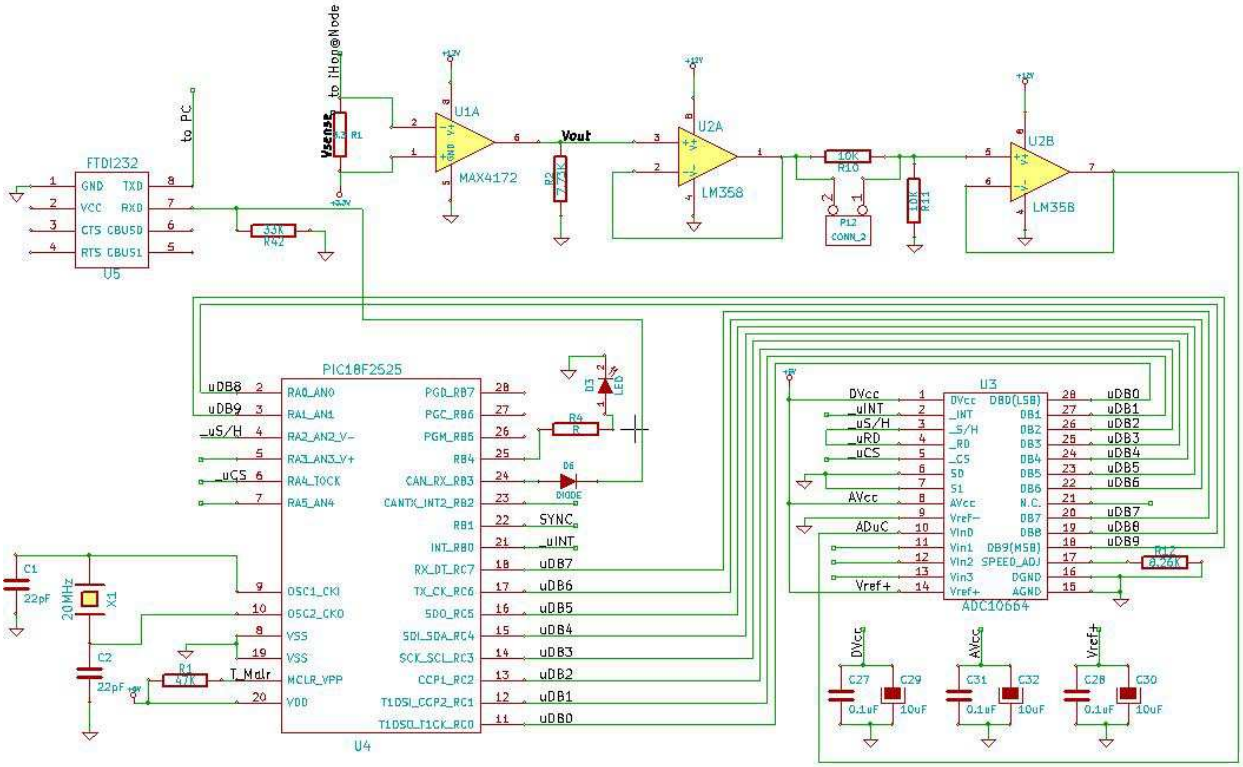

Figure 3.17: Schematic map of MEMD (one channel)

data can be sent out immediately after each sampling in an unbuffered sampling mode. An USB to Serial UART module U5 (UR232R) [172] also known as a virtual COM port module, is used for communication between microcontroller and data terminal. As shown in Fig.3.18, MEMD has four power supply channels for the sensor mote energy measurements and they can be applied in two cases (multi-component test and multi-mote test).

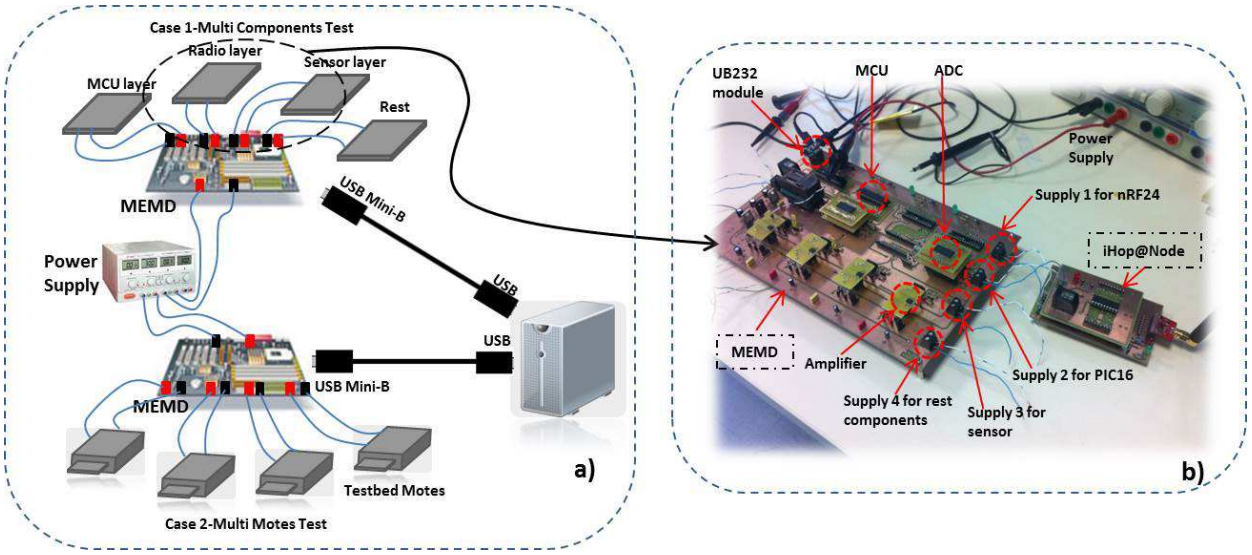

Figure 3.18: MEMD prototype

With the multi-layer architecture design of iHop@Node, MEMD can measure energy consumption on each hardware component separately and synchronously as shown in case 1. For case 2, instead of separate hardware components, MEMD is used to measure the energy consumption of four independent sensor motes. In Fig.3.18b), the MEMD prototype is presented. Channel 1 of MEMD is responsible for the energy measurement of nRF24L01+ transceiver of iHop@Node, channel 2 is for PIC16 microcontroller energy measurements, channel 3 can be attached to the sensor chip for energy measurement, and channel 4 is used for other hardware components such as LEDs or external memory. All the channels run samplings simultaneously, and measurements are stored in respective microcontrollers via buffered sampling mode for high speed. These data will be sent by data terminal and saved separately according to the channel number for later evaluation. In addition, the cost of this MEMD is much lower than an oscilloscope, an acquisition card and recently proposed Sensor Node Management Device (SNMD) built by KIT [173]. The following Table 3.4 lists the detailed measurement capabilities of MEMD.

3.4.5 EDMSP

Compared with [174] and [159], which provide SD or MMC card memory interfaces for energy measurement storage, iWEEP_HW provides an alternative means for energy data saving and management. A GUI-based Energy Data Management Software Platform (EDMSP) is designed and presented in Fig.3.19.

Table 3.4: MEMD capabilities

MEMD Capabilities:
√ Four channels for sampling energy usage data
√ Synchronous energy usage data sampling
√ 10-bit resolution for ADC value
√ 1900 samples buffers (each channel)
√ Configurable Gain (by tuning R1 and R2 in schematics)
√ 150 KHz buffered sampling rate
√ 5.5 KHz unbuffered sampling rate
√ Measurement range 0∼40 mA
√ Configurable data rate via virtual COM port interface (UB232R module)

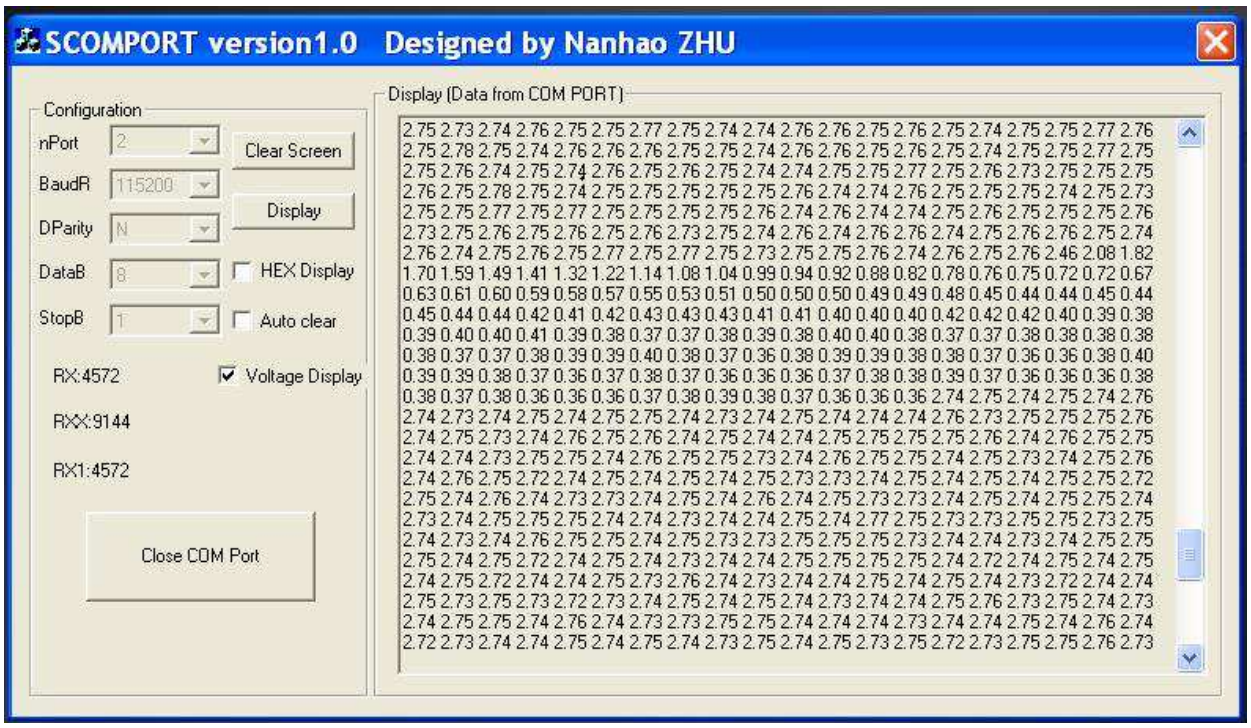

Figure 3.19: EDMSP

Due to huge storage space being needed for the collection of raw measurements, the on-board memory card based method cannot even support the experiment running for several minutes or hours. Therefore, saving energy measurements onto hard disk would be a better choice, since TB capacity level is available nowadays. Thus, when large amount of raw measurements can be sampled and sent to the PC for storage and future evaluation, EDMSP is built to facilitate this collection process. The implementation of EDMSP is based on the well-known CSerialPort class [175] developed by MFC and Windows API. Many configurable parameters can be set via EDMSP GUI. Before opening the specific COM port for data collection, parameters such as port number, baud rate, data bit and stop bit are provided. Once the selected COM port receives data, the number of received data will be recorded according to the configuration table, and the data values will also be shown in the display window. As EDMSP supports multi-type display functions, the received data can be shown in decimal value, hex value, voltage value, current value and even power consumption when the corresponding display check-box is selected. Other specific functions can also be easily extended to EDMSP according to user requirements. Further, all the displayed energy values are printed out into a TXT file for the convenience of subsequent data processing and evaluation. At present, EDMSP is implemented in Visual Studio 6.0, Visual Studio 2010 [176] and Dev C++ [177].

3.4.6 Measurements and Calibration of iHop@Node

In this section, MEMD and EDMSP are used for the measurements and calibration of the iHop@Node testbed. In the first part, the current consumption measurements of iHop@Node are presented in Fig.3.20 on both PTX and PRX devices. Since the microcontroller and transceiver are typically the most power consuming parts, the measurement results are mainly focused on these two parts in order to simply the investigations. After the energy measurement calibration, two case studies are employed to compare the calibrated iWEEP_SW model with the MEMD/EDMSP measurement model.

3.4.6.1 Energy Measurement and Calibration

Fig.3.20a) represents the process of PTX device under Enhanced ShockBurst mode, where the iHop@Node testbed is configured as 2Mbps and 0dBm output power. Each operation step is marked with a corresponding number for the clarity.

Interval 1: The PIC16 microcontroller is in active mode, while the nRF transceiver is in power down mode. The microcontroller demands about 1.4mA current consumption, and the transceiver consumes about 0.8∼0.9mA current. During the whole experiment, the PIC16 microcontroller is in active mode all the time.

Interval 2 and 4: During interval 2, the microcontroller sends commands to the transceiver to configure it from the power down mode to the standby mode by SPI communication. For interval 4, also by SPI communication, the microcontroller sends a 10 byte payload to the nRF transceiver's TXFIFO. Both intervals show that SPI communication will not increase the current consumption of PIC16 microcontroller.

Interval 3: on the receipt of SPI commands from microcontroller, the nRF transceiver crystal starts up and the transceiver enters into standby mode from power down mode. The maximum current consumption for this interval can reach about 1.83mA.

Interval 5: The CE (Chip Enable) signal is set high for at least 10μs, which triggers the transceiver first into the transition state TX_Settling. After 130μs in this TX_Settling state, the transceiver will

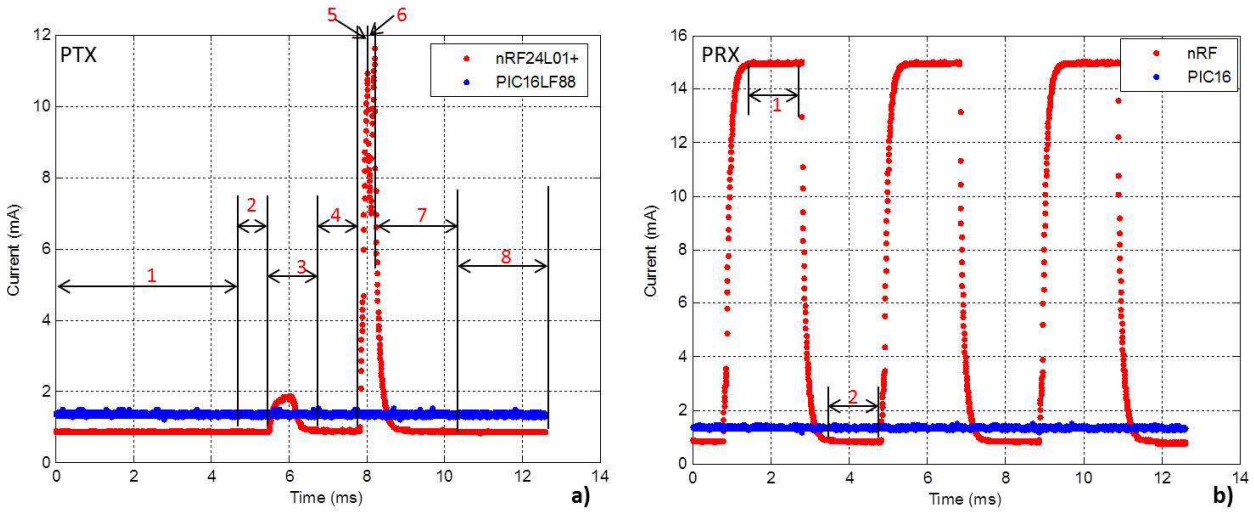

Figure 3.20: PTX and PRX measurements

automatically enter into the TX mode for over the air transmission. An average current consumption is given in the product specification for this 130μs settling state, and according to the measurement results there are actually two steps consuming different currents in this transition state. During the first 50μs, the current consumption increase rate is slower than the last 80μs, which is probably because in the first 50μs the transceiver only performs packet assembly, and then for the next 80μs many internal components of the transceiver are turned on for over the air packet transmission. Therefore, a much more rapid increase in current consumption can be found during the second step.

Interval 6: According to the ESB mode, the nRF transceiver will enter into RX_ACK mode waiting for the acknowledgement packet, and before this a transition state RX_Settling is required to make the transceiver from PTX to PRX. Since the current consumption in RX_Settling is lower than that consumed in TX and RX states, so the consumed current during interval 6 will first reduce and then increase.

Interval 7 and 8: On the successful receipt of the ACK frame, the nRF transceiver automatically returns to standby mode and during interval 8, the microcontroller sends a command via SPI to configure the transceiver into power down mode.

Fig.3.20b) represents the process of periodical listening of PRX device.

Interval 1: The transceiver is in RX mode for channel listening and is ready to receive possible incoming packets. The microcontroller works is in active mode.

Interval 2: CE is set low by the microcontroller, and the transceiver returns to standby mode.

The nRF24L01+ transceiver is integrated onto a module with an onboard 3.3V regulator, RP-SMA 2.4GHz antenna and some peripheral circuits. Hence, as shown in Table 3.5, the measured current consumption values of nRF24L01+ are much greater than those listed in the product specification. This is especially true in power down and standby modes, because significant current loads are required by other hardware components on the module except for the nRF24L01+ chip itself. Table 3.5 also shows that SPI and UART communications in the microcontroller do not consume extra energy. This is also true for the

instruction execution of microcontroller, where the instructions under investigation are AND, OR, XOR, ADD, DIV, MUL, WHILE and FOR loop [121]. Besides, the measured current consumption in the sleep mode of microcontroller is found to be significantly different (three orders of magnitude) from the values in the specification. This discrepancy could be related to some other scenarios: (1) The enable/disable of interrupt, SPI, ADC and Timer functions on the MCU chip (about 0.2mA current consumption can be saved if these functions are disabled). (2) The enable/disable of watchdog function. (3) The configuration of I/O ports.

For the other popular sensor motes, such as Telos, Shimmer Node, Tmote (TelosB), MicaZ, Mica2 and N@L, the measurement range of the proposed MEMD and EDMSP is still valid. The buffered sampling mode is fast enough to collect energy data, while multichannel design can also provide accurate and detailed energy measurements of several nodes simultaneously. MEMD and EDMSP are of very generic use in the energy measurement of WSN motes, and the following Table 3.6 also lists some current consumption figures of the above mentioned sensor motes in comparison.

3.4.6.2 Calibrated iWEEP_SW Model and Measurement Model

After energy consumption calibration, measurement results are also used for testbed operation calibration. Measurements provide detailed time duration information on each working state of the testbed, including chip enable signal duration time, delay of interrupt, duration of transition states, SPI communication time, etc. So with the calibrated model, an ESB mode-based calibrated iWEEP_SW model is simulated via iWEEP_SW and is compared with the real world measurement model. Detailed information is presented in Fig.3.21. In this experiment, the packet-based data rate is configured to 2Mbps, and the packet format contains a 5 byte payload while other packet fields are maintained as default values (1 byte preamble, 1 byte CRC, 5 bytes address, and 9 bits packet control field). The experiment shows that the calibrated simulation model can provide very close results to measurements, so more trusted and accurate energy consumption can be predicted when calculated with the measurement energy data value.

The second case study is about evaluating iHop@Node maximum continuous packet stream data rate for transmission and reception.

The maximum transmission packet stream rate can be achieved in the following ways. Firstly, the single packet-based data rate is configured as 2Mbps. Secondly, SB mode is used rather than ESB mode, because SB can send three packets continuously without waiting for the ACK frame after each packet transmission. For the packet format, each packet contains a 32 byte payload and default values for the other overheads (1 byte preamble, 1 byte CRC and 5 bytes address). In this case, three times of transmission are used in both simulation and experiment. The detailed result is presented in Fig.3.22. In this figure, the CE signal is set high for about 444μs to guarantee the sending of three continuous packets.

According to the results, the continuous packet stream rate for simulation and experiment can be calculated respectively as follows.

Measurement:

*Number of bits to send = (39 * 3 * 8) * 3=2808 bits = 2808 * 10^{-6} Mbit*

Table 3.5: Measurements of iHop@Node (oscilloscope : Tek 1012B)

Hardware State (iHop@Node)	V_{out} **(Scope)**	V_{out} **(MEMD)**	**Error of** V_{out} **(Vol;%)**	**Current (Scope)**	**Current (MEMD)**	**Current (Datasheet)**
PIC16(Sleep)	0.048V	0.047V	0.001V; 2.08%	0.13mA	0.12mA	0.5μA
PIC16(Active)	0.36V	0.35V	0.01V; 2.80%	1.399mA	1.34mA	1.2mA
PIC16(SPI)	0.36V	0.35V	0.01V; 2.80%	1.399mA	1.34mA	N/A
PIC16(UART)	0.36V	0.35V	0.01V; 2.80%	1.399mA	1.34mA	N/A
PIC16(Instructions)	0.36V	0.35V	0.01V; 2.80%	1.399mA	1.34mA	N/A
nRF(Power Down)	0.23V	0.21V	0.02V; 8.70%	0.89mA	0.82mA	0.9μA
nRF(Standby)	0.24V	0.22V	0.02V; 8.33%	0.93mA	0.85mA	26μA
nRF(Startup@max)	0.48V	0.47V	0.01V; 2.08%	1.86mA	1.83mA	0.4mA(average)
nRF(TX_Settling@max)	2.24V	2.23V	0.01V; 0.45%	8.70mA	8.66mA	8.0mA(average)
nRF(TX@0dBm)	2.84V	2.80V	0.04V; 1.41%	11.03mA	10.88mA	11.3mA
nRF(RX_Settling@max)	2.76V	2.75V	0.01V; 0.36%	10.72mA	10.68mA	8.9mA(average)
nRF(RX@2Mbps)	3.85V	3.82V	0.03V; 0.78%	14.96mA	14.84mA	13.5mA

Table 3.6: Current comparison of widely used sensor motes

Sensor Motes	**Telos** [125]	**Tmote(TelosB)** [39]	**N@L** [97]	**MICAz** [18] [125]	**MICA2** [17] [125] [83]	**CC2480** [178]	**CC2520** [178]	**MC1322x** [178]
MCU(Standby)	5.1∼21.0μA	15μA	7μA	27.0μA	19.0μA	0.75μA	N/A	0.3μA
MCU(Idle)	54.5∼1200μA	2.25mA	N/A	3.2mA	3.2∼3.3mA	N/A	N/A	N/A
MCU(Active)	1.8∼2.4mA	2.4mA	1.386mA	8.0mA	7.6∼8.0mA	15.5mA	N/A	10mA
MCU(RX)	21.8∼23mA	22.8mA	N/A	23.3mA	15.1mA	32.5mA	27mA	25.5mA
MCU(TX@0dBm)	19.5∼21mA	21.7mA	N/A	21.0mA	25.4mA	30.5mA	25.6mA	32mA
Radio(RX)	19.7mA	19.7mA	23.5mA	19.7mA	8.0∼9.6mA	27mA	22.3mA	21mA
Radio(TX@0dBm)	17.4mA	17.4mA	23.96mA	17.4mA	17∼25mA	27mA	25.8mA	29mA
Radio(Idle)	29μA	29μA	N/A	29μA	N/A	N/A	N/A	N/A
Radio(Sleep)	1μA	1μA	17μA	1μA	1μA	0.5μA	0.03μA	0.4μA

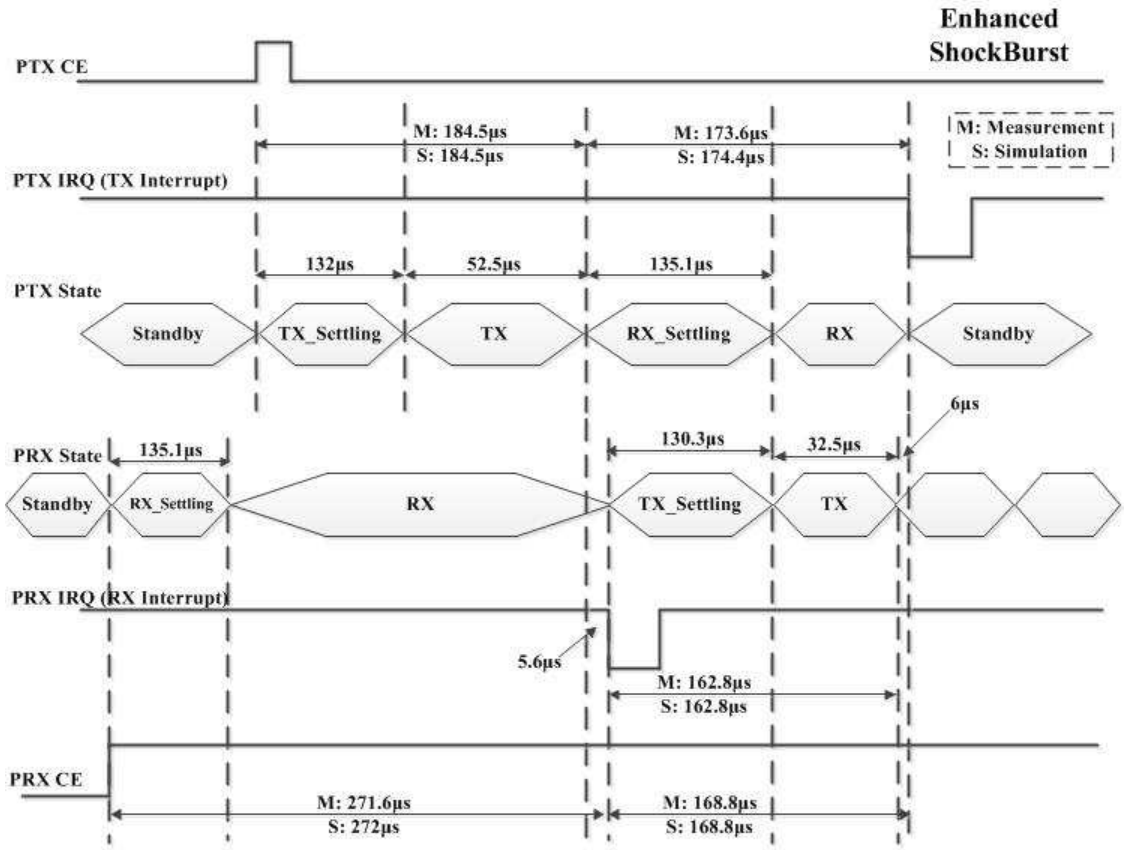

Figure 3.21: Calibrated model ESB model and measurements

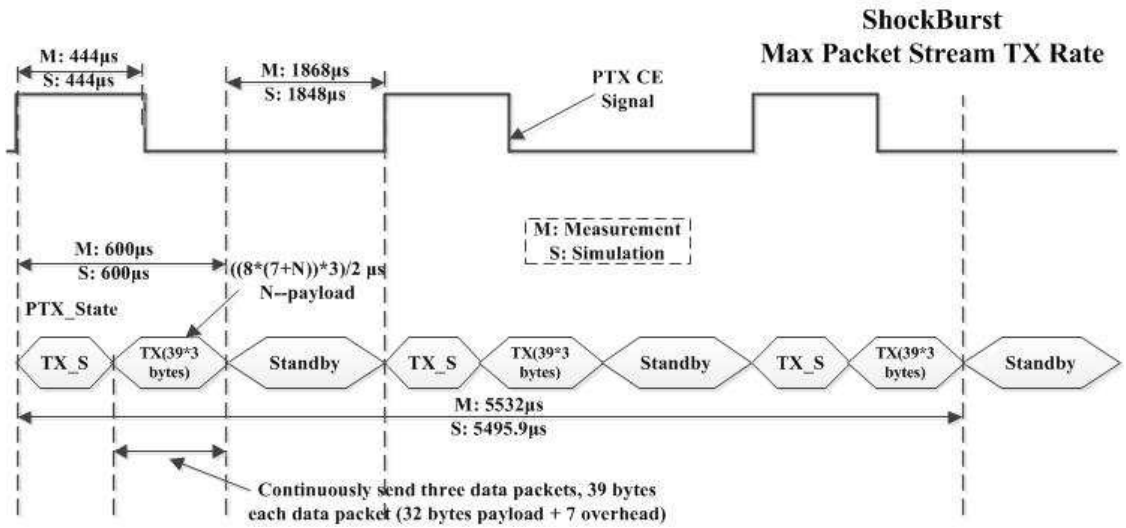

Figure 3.22: Maximum TX rate of packet stream

*Transmission time = 5532 μs = 5532 * 10^{-6} s*

*Continuous packet stream rate = (2808 * 10^{-6}) / (5532 * 10^{-6}) = 0.5076 Mbps*

Simulation:

*Number of bits to send = (39 * 3 * 8) * 3=2808 bits = 2808 * 10^{-6} Mbit*

*Transmission time = 5495.9 μs = 5495.9 * 10^{-6} s*

*Continuous packet stream rate = (2808 * 10^{-6}) / (5495.9 * 10^{-6}) = 0.5109 Mbps*

However, based on the measurements, the receiver cannot receive the packet stream at this 0.5Mbps rate without losing packets, because the SPI speed of payload reading is slower than its payload writing speed. This means that before reading out all the payloads from the three RXFIFO spaces (each RXFIFO contains 32 bytes payload), a new data packet will come from the transmitter to make the RXFIFOs full. Therefore, the transceiver cannot receive any new data packets with the saturated RXFIFOs, which will lead to packet loss. In order to make sure that the receiver can successfully receive all the packets without any loss, the receiver should first read out all the packets from RXFIFOs and have PRX_IRQ (RX interrupt) cleared before the transmitter finishes the first data packet sending during each transmission process. This is to guarantee that the next PRX_IRQ can be triggered, and that in its interrupt service routine the PIC16 microcontroller can continuously read out all the packets from RXFIFOs and then clear PRX_IRQ for the preparation of new incoming packets. Thus, measurements are run to find the maximum packet stream reception rate, and the experimental measurement is compared with its corresponding simulation result shown in Fig.3.23.

Based on the results, the maximum packet stream reception rate is calculated respectively as follows.

Measurement:

*Number of bits to send = (39 * 3 * 8) * 3=2808 bits = 2808 * 10^{-6} Mbit*

*Transmission time = 7100 μs = 7100 * 10^{-6} s*

*Continuous packet stream rate = (2808 * 10^{-6}) / (7100 * 10^{-6}) = 0.395 Mbps*

Simulation:

*Number of bits to send = (39 * 3 * 8) * 3=2808 bits = 2808 * 10^{-6} Mbit*

*Transmission time = 7038 μs = 7038 * 10^{-6} s*

*Continuous packet stream rate = (2808 * 10^{-6}) / (7038 * 10^{-6}) = 0.399 Mbps*

Despite the fact that reception rate is lower than transmission rate, it is high enough to guarantee a much better performance of many sensor network application scenarios.

3.5 Conclusion

In this chapter, the design and implementation of a complete WSN energy evaluation platform, named iWEEP, is presented in detail. For its software part, iWEEP_SW is a system-level transaction-level and energy-aware SystemC based sensor network simulation environment. iWEEP_SW focuses on modular design for components on sensor motes at different abstraction levels and can provide accurate and realistic energy consumption evaluations with a calibrated measurement model. On the hardware part, iWEEP_HW is a real-world energy measurement system made up of a measurement platform MEMD and an energy management platform EDMSP. Due to the capability of simultaneous energy data monitoring at experiment

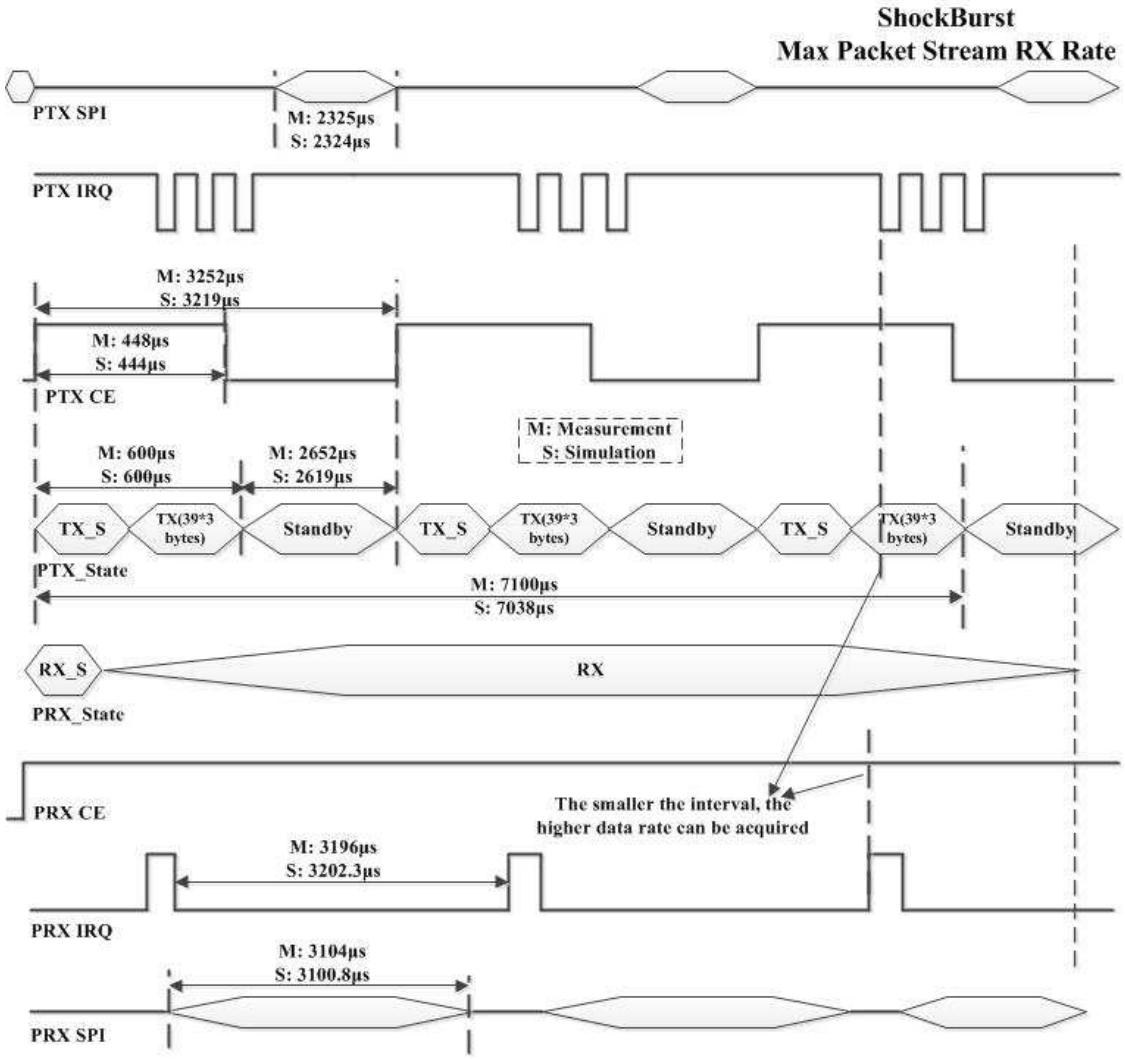

Figure 3.23: Maximum RX rate of packet stream

runtime, MEMD and EDMSP not only provide accurate measurements on each hardware component of the target sensor mote testbed, but also help to explore the interaction between different components during the experiment process. The use of iWEEP for the energy evaluation of sensor network by combining iWEEP_SW and iWEEP_HW is proved to be an efficient, accurate and cost-saving method.

Chapter 4

Energy Performance Evaluations of WSNs under Different Scenarios

Contents

In this chapter, the simulation based method is used for energy evaluation. The energy consumption evaluation and exploration of four mote type based networks (Telos, MICAz, MICA2 and iHop@Node) are investigated under different case scenarios via iWEEP simulation. In each case scenario, detailed energy information and comparison data are presented and analyzed for the corresponding mote platform. After the energy evaluation of each scenario, suggestions are given to designers and researchers to help them effectively reduce energy consumption and achieve an energy-aware wireless sensor network.

This chapter is organized as follows. Section 4.1 examines the energy consumption under different traffic load. Section 4.2 studies the impact of sensing start time on energy consumption. Section 4.3 investigates energy consumption under a payload collecting strategy. Section 4.4 analyzes the impacts of output power and channel on energy consumption. Section 4.5 uses the principal component analysis method to evaluate and explore data rate and unslotted algorithm parameter impacts on energy consumption. Section 4.6 evaluates the energy consumption of Enhanced ShockBurst (ESB) and ShockBurst (SB), and then the energy consumption of ESB is compared with the unslotted algorithm energy consumption on iHop@Node. Finally, section 4.7 concludes this chapter.

4.1 Traffic Load Impact

Depending on the different application scenarios, the deployment of a wireless sensor networks may range from the light and steady traffic load in applications such as domestic smart metering/automation [179], environment monitoring, and target tracking [180], to the heavy and burst traffic loads in areas such as vehicle automation/adaptation [181] and health/medical monitoring on the human body [168]. With varying number of nodes as well as different sampling intervals of each sensor mote in these different applications, the overall traffic load has a large dynamic range, and the impact of traffic load on the energy consumption of the sensor mote will be discussed in this section.

Since the sensor motes in the aforementioned applications are required to run for a long period of time for a given task without human interference, the energy consumption has always been a significant concern. With a large dynamic range of traffic load, a typical health/medical monitoring based network for human body area [1] [182] (Fig.4.4) is under investigation to explore the impact of traffic load on the energy consumption of sensor network . Consisting of 3 to 10 nodes, the network is attached to the human body for different physiological signal monitoring purpose (10Hz for temperature sensor, 50Hz for glucose sensor, 100Hz for ECG-electrocardiogram). Hence, this scenario has a wide dynamic range of traffic loads from 30 pkts/s (packets/s) to 1000 pkts/s. Three types of sensor mote platforms applied in the simulation experiments are iHop250K (iHop@Node with 250Kbps), Telos (250Kbps) and MICAz (250Kbps). The average energy consumption of each sensor node under different traffic load rates are presented in Fig.4.1. For the sake of clarity, only 4-node, 7-node and 10-node cases are given for each platform. For other parameter settings, a 2 byte payload is used to store 12 bits of sensed data [168]. The default 0dBm output power is selected on the transceiver for over-the-air packet transmission. The packets format of Telos and MICAz used for transmission are shown in Fig.3.3, which is the standard IEEE 802.15.4 based packet format. The packet format shown in Fig.3.2 is applied to iHop250K, which is defined by the ShockBurst (SB) mode of the nRF24L transceiver (1 byte preamble, 3 byte address as network id, 1 byte CRC, 4 bytes payload field including 2 bytes source node id and 2 bytes real payload data). Since only the unslotted CSMA/CA based MiWi communication protocol is supported by iHop@Node (cf. section 3.3.2.1), both Telos and MICAz motes also apply the unslotted CSMA/CA in the comparison. Four parameters of the unslotted algorithm are set as default values, which are 3, 5, 4 and 3 for *macMinBE*, *macMaxBE*, *macMaxCSMABackoffs* and *macMaxFrameRetries* respectively. In addition, each sensor node starts the sensing at the same time. Each case runs for the same simulation time (4 seconds in this work) with different seeds to generate random backoff slot numbers, and the average value of 50 times of runs are used.

From Fig.4.1, it can be seen that more energy will be consumed on each sensor node with the increase of sample rate and number of nodes. When the sample rate is fixed, more packets are required to be transmitted for increasing node number. Therefore, increasing competition for channel access directly results in increasing probability of packet collision and channel access failure. Thus, the communication protocol (unslotted CSMA/CA) supports repeated channel access attempts and multiple retransmission times to guarantee the functionality and performance of the network. This means that the sensor node consumes extra energy for this overhead. In the case with fewer nodes, more energy can be saved from the overhead of communication protocol, since the transmission task can be completed with less or without any channel access attempts or retransmissions due to the good channel conditions.

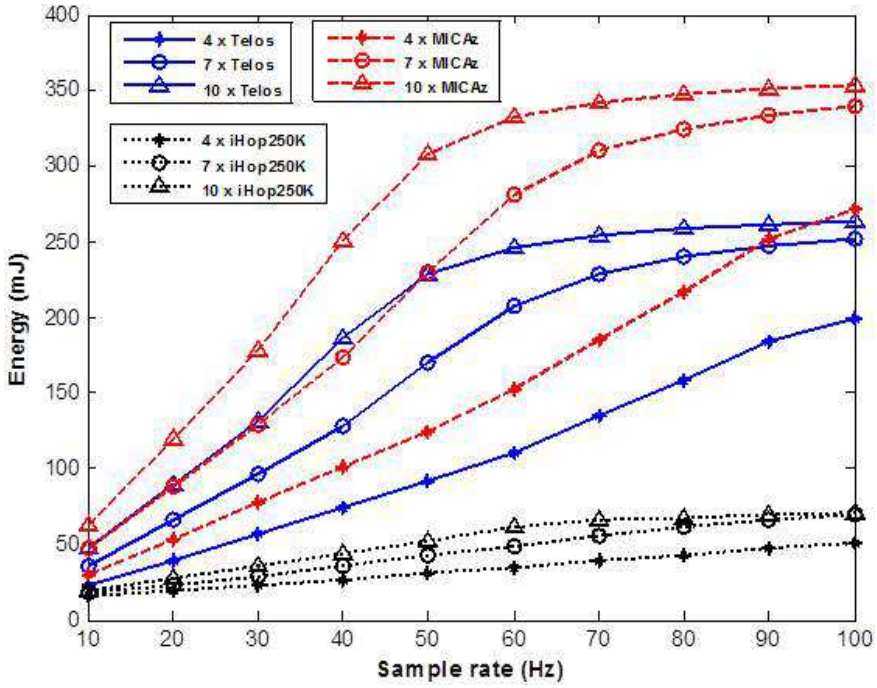

Figure 4.1: Traffic load impact on energy consumption

When it comes to the scenario with a fixed number of nodes and increasing sample rates, the energy consumption follows the increase of sample rate. Except for the energy consumption used for the protocol overhead analyzed above, another reason is that the increase of sampling rate will result in packet saturation problems, which is to say that there are always pending packets waiting to be sent out on each sensor node. Hence, more time will be spent in active mode for packet transmission while less time will be in sleep mode for energy saving.

The results from Fig.4.1 also show a tendency that, in the 250Kbps based network, the energy consumption remains stable when the traffic load is high (above 560pkts/s) and the energy consumption value has low dependency on the number of nodes and the sample rate. The reasons for this are based on the following two aspects. Firstly, when traffic load is heavy, the channel condition becomes too bad to carry out packet transmission. Thus, more times of repeated channel checks and retransmission mechanisms will be applied by the unslotted protocol for each packet transmission until it uses up all times of predefined CCA and retransmissions. It is especially true that each sensor mote based network has a similar probability of encountering Channel Access Failure (CAF - after $1+macMaxCSMABackoffs$ consecutive CCA failures) and Collision Failure (CF - when $1+macMaxFrameRetries$ collisions take place) under high traffic load according to the simulation experiments. This means that for each specific sensor node based network the energy consumed on its protocol overhead is basically the same under high traffic load conditions. The second reason is the previously mentioned packet saturation problem. With a high traffic load, after the end of transmission of one packet, irrespectively of whether it is a success (success of receiving ACK) or a failure (CAF / CF), the next packet is always pending for transmission. Thus, the node remains in active mode all

the time and consumes nearly the same amount of energy. On the other hand, under the same traffic load conditions, the scenario with fewer nodes and higher sample rate consumes a slightly more energy than the scenario with more nodes and lower sample rate. This is because lower sample rate can effectively avoid the packet saturation problem and provide more chances for entering sleep mode to save more energy.

Compared with MICAz and Telos, iHop250K has much lower energy consumption under each traffic load scenario, which is due to its short packet format and low current consumption of the hardware components. With the comparison between standard IEEE 802.15.4 packet (17 bytes data packet, 11 bytes ACK frame in this experiment) and the short packet (9 bytes data packet, 8 bytes ACK frame in this experiment), it is easy to find that the transmission time for both data packet and ACK frame of iHop250K can be greatly reduced. This not only saves energy used in TX and TX_ACK modes, but also improves the channel condition and saves the energy from the protocol overhead. Further, the ultra-low power consumption of iHop@Node (section 3.4.6) in transmission, reception, and active states also make iHop@Node more energy-efficient than the other two platforms. Note that despite the use of the same transceiver (CC2420) for both Telos and MICAz, Telos is more energy-saving because MSP430F used on Telos is a new low-power microcontroller in active, idle and sleep modes when compared to ATmega128 microcontroller on MICAz (Table 3.6).

In addition, when adopting a typical AA battery with a capacity of 2600mAh for the lifetime estimation, and assuming all the capacity will be used to power the node, the lifetime under minimum traffic load (30pkts/s) and maximum traffic load (1000pkts/s) are 90.4 days and 5.6 days for Telos, 80.0 days and 4.1 days for MICAz, 95.5 days and 20.4 days for iHop250K respectively.

4.2 Impact of Sensing Start Time

Due to the spatially distributed characteristic of sensor nodes, whether their deployments are dense or sparse over the area of interest, it is difficult to make sure each node can start working at the same time to retain the synchronization. Secondly, due to the fact of cheap manufacturing of hardware components and lack of calibrations after fabrication, it is also difficult to guarantee the same start up time or clock period for each sensor node. In this section, the impact of sensing start time on energy consumption is examined. Comparison of energy consumption between simultaneous sensing start time and random sensing start time (each node starts sensing first data at a random instant between the beginning of the simulation experiment and the first sample interval) are made as follow.

By employing the same experiment scenario from section 4.1, we compare the energy consumption on sensor node as follows. Taking the Telos mote as an example, Fig.4.2 reveals that the case with random start time is more energy-efficient. Compared with the random start sensing case, simultaneous sensing consumes more energy due to protocol overhead. This is because all sensor nodes start data sensing at the same time under the simultaneous case, and before over-the air-transmission 2^{BE}-1 (BE initialize as $macMinBE$) unit backoff periods are required. However, when $macMinBE$ is initialized to a smaller value (i.e. default: 3), it is easy for different sensor nodes to get the same number of backoff slots (0∼7). Thus, with the same sensing start point and the same/similar backoff period, the transmission time of many packets can overlap with each other, which causes serious packet collisions during a short period of time. Even if the transmissions do not

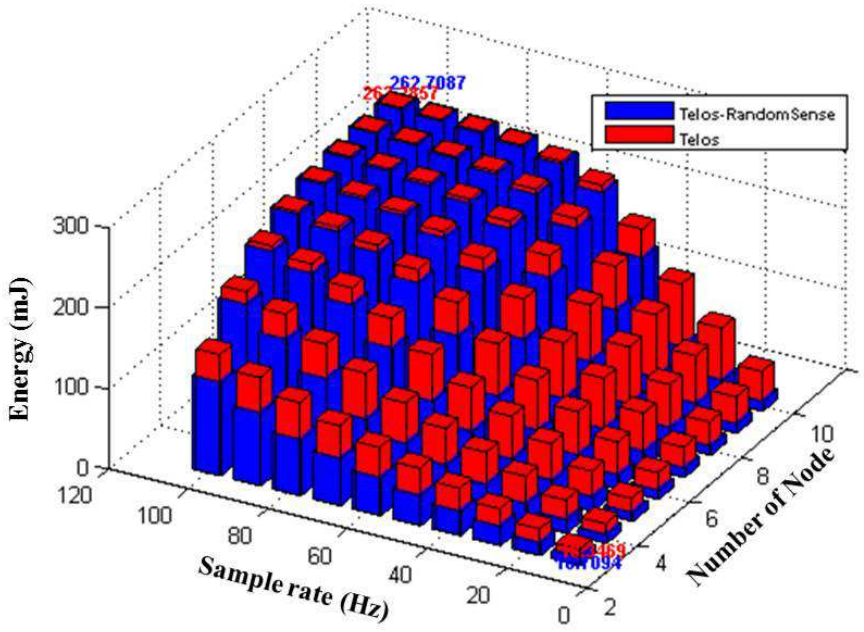

Figure 4.2: Impact of random sensing start time on energy comparison

overlap, with the close value of the backoff period, when new nodes join in the transmission process, these nodes will always find the channel busy because the transmission of the last node packet has not finished. So the processes of repeated CCA and retransmissions are launched by the communication protocol and therefore extra energy is consumed.

Due to random sensing start time, the transmissions of the packets are spread out and channel competition is released, so more energy is saved since it is more likely that packets can be successfully delivered with no collisions and therefore low protocol overhead. The energy-efficiency of the random sensing start time is obvious under low traffic load. At high traffic load, as previously analyzed, the whole network is in a saturated state, and therefore the spreading of packet transmissions by the random sensing start time strategy does not help. The results also show that for random start time case under the same traffic load conditions, the case with a larger number of nodes and lower sample rate can save more energy than the case with a smaller number of nodes and higher sample rate, to which two reasons can be attributed. Firstly, a higher sample rate reduces sleep time and causes more energy consumption despite the use of random sensing start time. Secondly, a higher sample rate narrows the interval of data generation, which in turn increases the probability of the overlapping of packet transmissions and hence leads to additional energy consumption due to protocol overhead. According to the experiments of the random sensing start time, the maximum and minimum energy saving rates of the Telos mote are 73.8% and 0.16%, and for MICAz and iHop250K the rate are 73.4%, 0.01% and 40.4%, 0.15% respectively.

4.3 Packet Control Overhead (Packet Collecting Strategy)

Data packets used for over-the-air transmission consist of two parts: packet payload and packet overhead. Payload bytes, as mentioned previously, are the data values sampled/sensed by the sensor and assembled into a transmission packet. The ratio of successful delivery of these data values is used to judge the network reliability. Other functional parts can be considered to be part of the packet control overhead since they do not contain any payload data. For a standard IEEE 802.15.4 packet, packet control overhead includes Synch Header (SHR), PHY Header (PHR), MAC Header (MHR) and MAC Footer (MFR). For iHop@Node packet and MICA2's TinyOS packet, the packet control overhead consists of preamble, destination address, packet control field (only for ESB mode) and CRC. In this section, the impact of these packet control overheads on energy consumption will be examined.

A typical lab-sized network (with a size of about 20 sensor nodes [183] [127]) is under investigation. The comparison experiments are made on iHop250K, MICAz and Telos-based networks. In the first experiment, a packet with a 2 byte payload is transmitted every 10ms, then a 4 byte payload packet every 20ms, a 6 byte payload packet every 30ms until a 20 byte payload packet every 100ms is reached. Each experiment runs for the same time (10s) and the average value of 50 runs is used. The results, presented in Fig.4.3, reveal the significant impact of packet control overhead.

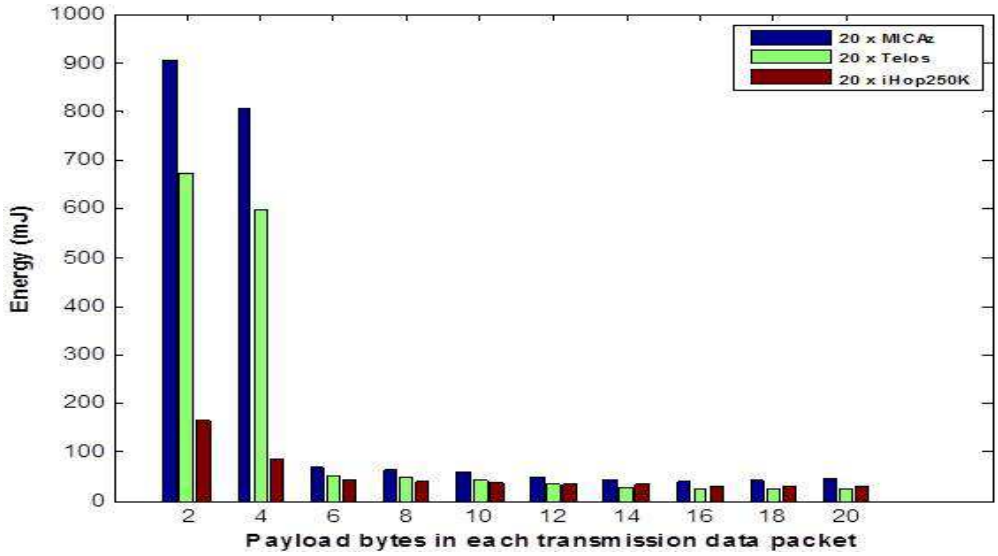

Figure 4.3: Energy consumption under payload collecting strategy

Since the increasing payload data can be attached to the same size of packet control overhead and when this data collecting strategy is used, fewer data packets are transmitted when the same amount of payload data is required. Therefore, the energy wasted in packet control overhead under frequent transmission conditions can be greatly saved, while energy cost is also reduced with fewer ACK frames. In addition, despite the fact that the increase on the total length of each transmission packet occupies the channel for a longer time, the increase in the transmission interval is large enough to release channel competition and save energy from channel checks and protocol retransmissions. It further saves the energy wasted in active

mode due to packet overflow preventing the sensor node entering sleep mode. According to the experimental results, the packet loss probability can be drastically reduced from 96.3% and 90.1% for the 2-byte and 4-byte cases respectively to 3.98% in the 20-byte case for MICAz and Telos (MICAz and Telos share the same protocol model). For iHop250K, this rate is from 93.9% in the 2-byte case, 36.0% in the 4-byte case, and is reduced to 1.97% in the 20-byte case. In addition, starting from the 14-byte case, the energy consumption of iHop250K exceeds that of Telos. This is because when the transmission interval increases, sensor nodes can spend more time in the sleep mode. Hence, the microcontroller of Telos saves more energy in sleep mode (5.1μA from product specification) than the iHop250K microcontroller (120μA from measurement) and this eliminates the energy disadvantage of its transceiver, as well as the packet control overhead and active current consumption of the microcontroller.

Although the payload collecting method can conserve energy, it has some drawbacks. When the channel is of low quality with a high bit error rate (BER) / packet error rate (PER), longer data packets are more likely to subsume errors and necessitate retransmission. This will definitely cause extra energy consumption and longer packet latency. This channel impact is examined in the following section 4.4.

4.4 Output Power and Channel Impact

The use of various energy-aware communication protocols provides a flexible means of energy-saving for the sensor network at the software level. At the hardware level, the power consumption depends on the working state of hardware components, which is governed by the characteristics of the component itself and cannot be changed in the field, unless some power management or configuration strategy has been designed into the component. For instance, the configurable output power on the node transceiver can impact the energy consumption, which may provide a potential energy-efficient alternative for special cases, and can also help to develop energy-efficient communication strategies that involve both hardware elements and software design. As in [184], the authors propose and implement an adaptive power control algorithm by automatically configuring the programmable output power on the nRF24L01 transceiver according to the node's distance information.

Lower output power results in lower current consumption but also shortens transmission distance and increase the likelihood of higher PER over the channel. Therefore, low output power may not overall yield energy saving, and the selection of the output power should be specific to the application and channel condition. In this section, the energy consumption of various output power configurations are examined under both free space conditions and in a real-world application: an on-body line of sight (LOS) channel [185].

By extending the free space propagation model (section 3.3.4), the received power in an on-body LOS channel at a distance of d between the transmitter and receiver can be described as follows.

$$P_{rcv}(d) = \frac{P_{tx} \cdot G_t \cdot G_r \cdot \lambda^2}{(4\pi)^2 \cdot d^n} \tag{4.1}$$

where n=3.11 [185], G_t=G_r=1, λ=12.5cm.

Moreover, the energy per bit E_b in relation to the noise energy N_o is presented in (4.2) via Signal-to-Noise

(SNR).

$$\frac{E_b}{N_o} = SNR \cdot \frac{1}{R} = \frac{P_{rcv}(d)}{N_o} \cdot \frac{1}{R} = \frac{1}{N_o \cdot R} \cdot \frac{P_{tx} \cdot G_t \cdot G_r \cdot \lambda^2}{(4\pi)^2 \cdot d^n} \tag{4.2}$$

where N_o=-100dBm [185], and R is the bit rate (250kbits per second).

Then, the bit error rate (BER) is given as

$$BER = \frac{1}{2} \cdot erfc\left(\sqrt{\frac{E_b}{N_o}}\right) \tag{4.3}$$

And the packet error rate (PER) of a single transmission for a packet of q bits is computed as

$$PER = 1 - (1 - BER)^q \tag{4.4}$$

In an application study, simulations are set to run for the following scenario: eight sensor nodes with different functions are deployed on a human body to collect data and send them to a common destination for the evaluation, as briefly shown in Fig.4.4. In the following experiments, iHop250K, MICAz and Telos motes are under investigation, and the simulations were performed over the whole range of output power values. For clarity, in this section the results with output power of 0dBm and -15dBm (depends on CC2420) are shown for MICAz/Telos, and the results of 0dBm and -18dBm (depends on nRF24L01+) are presented for iHop250K. The results are shown in Fig.4.5 (other simulation parameters are kept the same as in section 4.1).

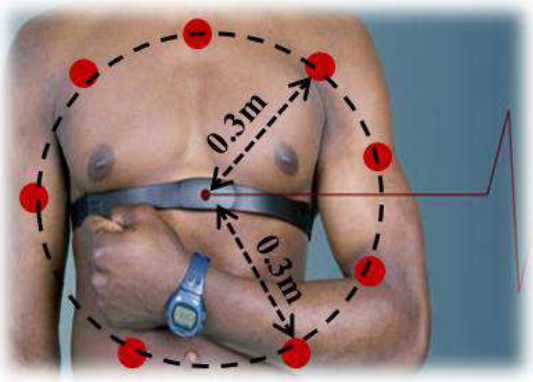

Figure 4.4: Sensor node deployment on human body

According to the results, when the free space channel condition is considered, despite the lower current consumption for lower output power, the overall advantage of using low output power is not obvious. This is because the packet transmission time itself is a short period and the total amount of time takes only a small part of the whole task. Since the current consumption values for various output power values are actually quite close, the output power has little impact on the total energy consumption. The difference does increase under heavy traffic load, since more energy is saved from the increase in retransmissions against the worse channel conditions.

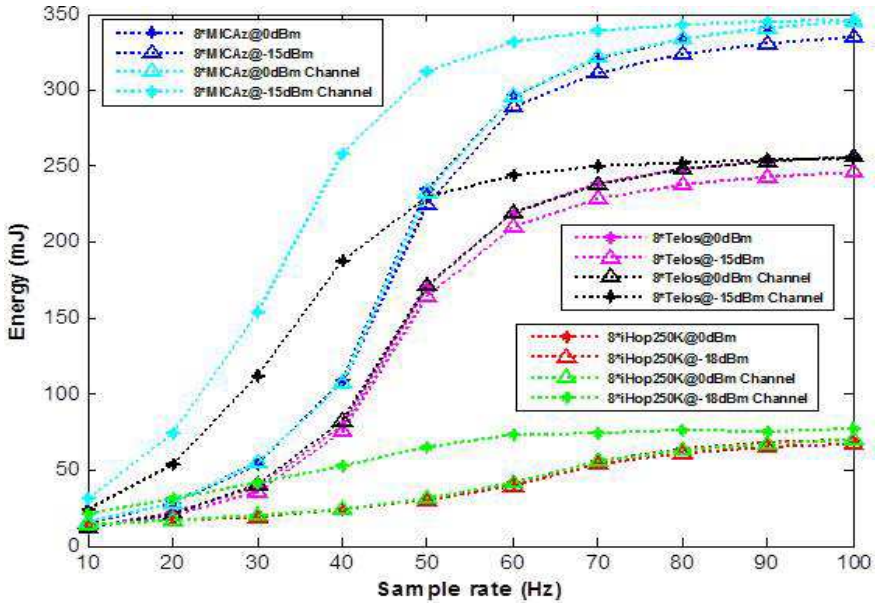

Figure 4.5: Output power impact on energy consumption under two channel conditions

As the comparison, when the on-body line-of-sight (LOS) channel is considered, the low output power scenario in fact consumes more energy than the high output power scenario, due to the high packet error rate. According to the equations from (4.1) to (4.4) and the distance (between the sensor node and the receiver terminal) shown in Fig.4.4, when the output power is set as -15dBm and -18dBm for MICAz/Telos and iHop250K respectively, the PER for the MICAz/Telos based network can reach 46.95%, and the PER for the iHop@Node based network can reach 86.55%. If 0dBm output power is used for both MICAz/Telos and iHop@Node based networks, the receive power at a distance of 0.3m (Fig.4.4) is sufficiently high to ensure no packet errors. Therefore, extra energy will be consumed by the protocol to overcome the high PER under LOS channel for low output power. Since 0dBm output power scenario in this study can effectively overcome packet error to save energy, the energy consumption is the same under both free space and noisy channel conditions. On the other hand, under the high traffic load of LOS channel the energy consumption of both low and high output power scenarios become stable and close. This is due to the saturated channel conditions where the impacts of different PERs to the channel status can be ignored, and nearly same amount of energy is consumed by the protocol overhead to handle the busy channel.

4.5 Data Rate and Unslotted CSMA/CA

As a leading and well-known standard in wireless sensor networks, IEEE 802.15.4 based networks have been well-adopted in various applications over the past few years such as environment monitoring, event tracing and target tracking, in which sensor nodes are always densely deployed with intense channel competition. However, the use of 250Kbps low data rate (MICAz/Telos/iHop250K) and 38.4Kbps low data rate (MICA2) based networks not only can be inefficient and unsuitable for these application scenarios in terms of energy,

but also with negative impact on network reliability as well as other performance metrics. In this section, with the configurable data rate characteristic of iHop@Node (1Mbps and 2Mbps high data rate based sensor nodes - iHop1Mb and iHop2Mb), we investigate these performance metrics and compare them to those of low data rate networks. The impact of unslotted algorithm parameters *macMinBE/macMaxBE (BE)*, *macMaxCSMABackoffs (Backoff)* and *macMaxFrameRetries (Retries)* on the energy consumption and network reliability are evaluated under increasing traffic load conditions for the aforementioned platforms (i.e. with a data rate of 38.4Kbps/250Kbps/1Mbps/2Mbps). The traffic load is varied by changing the number of nodes over the range 10 through 80 for the simulation of a dense deployment based network scenario. Each node senses data at the rate of 10pkts/s, for which the sample interval of 100ms is greater than the typical packet latency. Hence, the network in low data rate can operate in unsaturated conditions when the traffic load is not heavy and with a short backoff wait period, or alternatively by making use of a high data rate to guarantee unsaturated conditions of the network. Packets with a 32 byte payload (the maximum size for iHop@Node packet frame) is used for intense channel competition under different traffic loads. Strategies of how to select the unslotted algorithm parameters to achieve an energy-aware and highly reliable network are given after each evaluation analysis. The following sub-sections examine and analyze the impact of each protocol parameter under different data rate and traffic load scenarios.

4.5.1 Impact of Data Rate and Traffic Load

Results from Fig.4.6 to Fig 4.17 reveal that the 1Mbps and 2Mbps based high data rate networks are able to provide both lower energy consumption and better network reliability under different traffic loads. Since the time spent in packet/ACK transmission and reception at high data rate is reduced, the channel can be left in an idle status for more time and therefore more packets can be sent out, reducing the packet loss and improving network reliability. With better channel conditions under a high data rate, packets are more likely to be successfully transmitted with fewer or no CCA failures or collisions, so the time saved from the protocol overhead helps the nodes to be in sleep mode for energy conservation for longer. Despite the fact that individual packet transmission and reception energy at high data rates is higher [21], the total time spent in the reception and transmission process is shorter, so the final overall energy consumption is reduced and gives better lifetime. In terms of traffic load, as analyzed in the previous sections, the strong channel competition leads to more packet loss and the increase in protocol overhead causes extra energy consumption.

4.5.2 Impact of *BE*

For the unslotted algorithm, the BE value controls the range of backoff wait time duration and is initialized as *macMinBE*. After each CCA failure it will increase by 1 until it reaches *macMaxBE*. In order to make this backoff duration independent of CCA failures, *macMinBE* and *macMaxBE* are set to the same value [186] and they are simplified to be both equal to *BE*. The results, with the values of 3, 5 and 7, are shown for clarity and the investigations are examined for all platforms (iHop2Mb, iHop1Mb, iHop250K, MICAz, MICA2 and Telos). On the other hand, parameters *macMaxCSMABackoffs (Backoff)* and *macMaxFrameRetries (Retries)* are kept at their default values of 4 and 3 respectively during the experiments. This principal

component analysis (PCA) based method is also used for experiments that investigates the impacts of *Backoff* and *Retries*.

Results from Fig.4.6, 4.8, 4.7 and 4.9 reveal that with the three data rates of iHop@Node, a greater *BE* can reduce packet loss but increases energy consumption. Firstly, a greater *BE* increases the range of backoff wait duration, which spreads out the time that nodes compete for the channel, such that more packets can be successfully sent out and received, so better network reliability can be achieved.

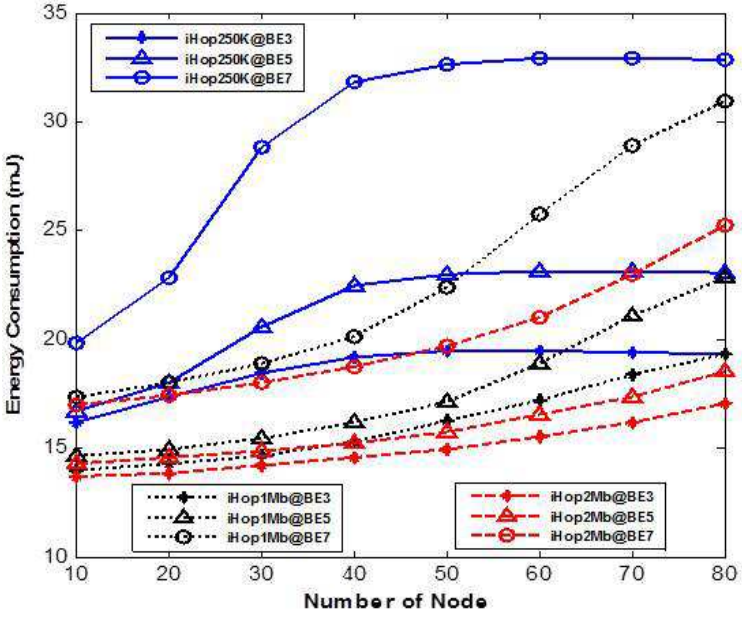

Figure 4.6: Impact of *BE* @energy consumption (iHop@Node)

However, for energy consumption a greater value of *BE* leads to a significant increase in backoff wait duration, which means that the node will spend more time in the active mode for this wait period and less time in sleep mode for energy conservation. Thus, the overall energy consumption increases with greater *BE*. According to the experimental results of the 250Kbps low data-rate based networks (iHop250K, MICAz, Telos), the channel competition introduced with increasing numbers of nodes will cause each node to use up all defined quotas of channel checks and retransmissions (some overflows happen only when *BE*=7) before dropping a packet. So with such a saturated channel, even though the number of nodes continues to increase, the energy consumption used on the protocol overhead still remains constant. Therefore, for each *BE* value the energy consumption becomes stable under the saturated channel condition. This is especially true for the MICA2 mote due to its 38.4Kbps based low data rate and long TinyOS based data packet format (27 bytes packet overhead + 32 bytes payload). Compared to the low data rate scenario, the energy consumption in the 1Mbps and 2Mbps scenarios do not become stable under larger sensor node conditions. This is because the high data rate can significantly reduce channel competition, and even for large number of nodes the channel cannot be saturated. To handle the stronger channel competition for large number of nodes, more channel access attempts and retransmissions will be used compared to the small node case. Thus, more energy will be consumed with the increase of nodes, and the 1Mbps and 2Mbps based high data

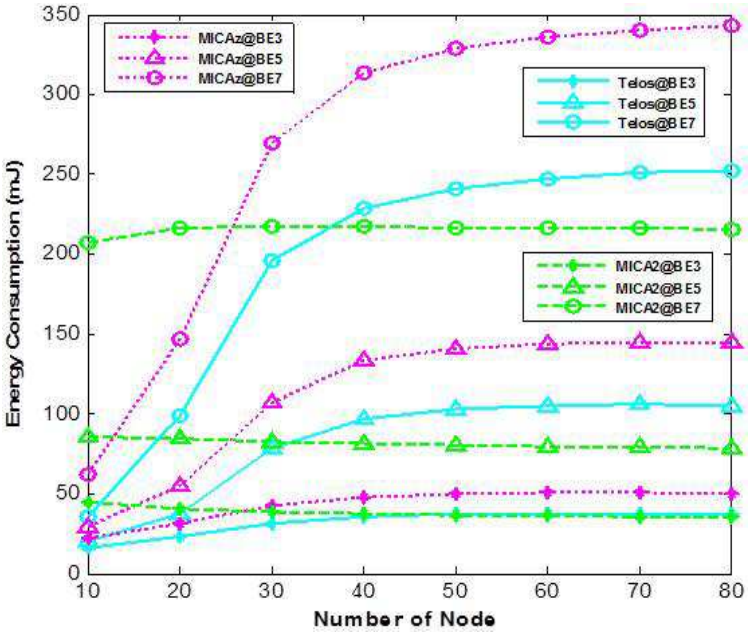

Figure 4.7: Impact of *BE* @energy consumption (MICA2, MICAz, Telos)

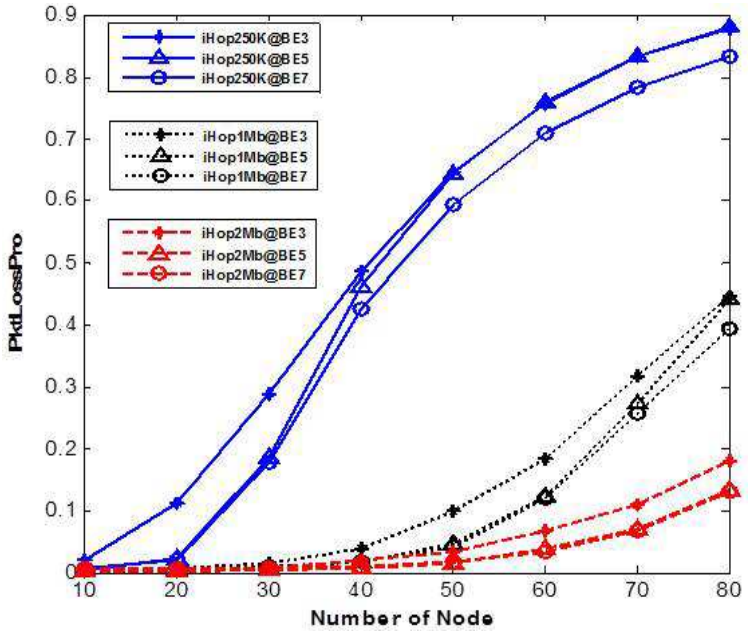

Figure 4.8: Impact of *BE* @packet loss probability (iHop@Node)

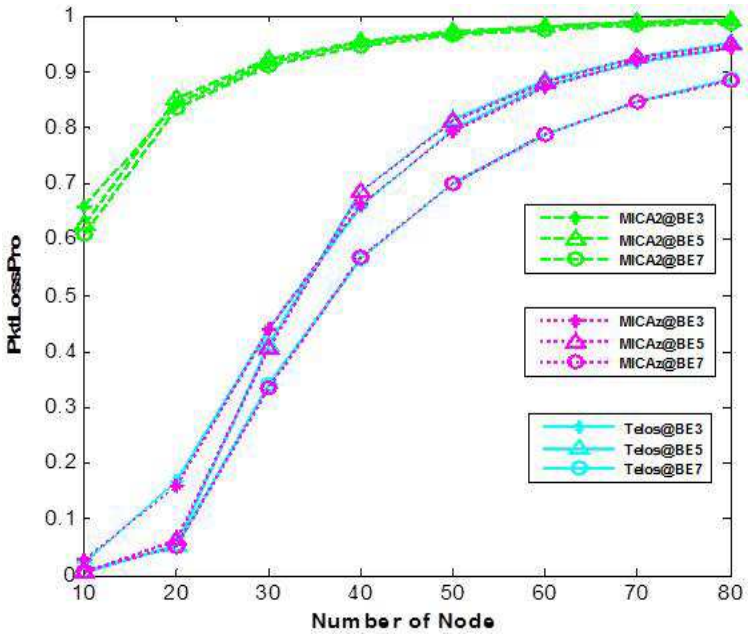

Figure 4.9: Impact of *BE* @packet loss probability (MICA2, MICAz, Telos)

rate scenarios follow the same trend in the following *Backoff* and *Retries* investigations.

Note that MICAz and Telos share the same CC2420 transceiver model and the same functional model of the unslotted algorithm in the MCU model, so the packet loss probabilities for both are basically the same. In addition, the packet latency will inevitably increase with a greater *BE* value.

In summary, if the packet transmission latency of the application is not a significant consideration, then a smaller *BE* could be a suitable choice since it consumes less energy and has shorter latency, On the other hand, a greater *BE* could be an alternative for the tradeoff between network reliability and energy consumption.

4.5.3 Impact of *Backoff*

In this sub-section, *macMaxCSMABackoffs* is simplified to *Backoff*, and the values under investigation are 2, 4 and 5 (*BE* and *macMaxFrameRetries* are kept at the default value of 5 and 3 respectively). The results shown in Fig.4.10, 4.12, 4.11 and 4.13 are explained as follows.

For 250Kbps based networks (iHop250K, MICAz, Telos), the increment of the *Backoff* value under low traffic loads can help improve network reliability, because it provides more transmission attempt opportunities for each packet. Starting from 300pkts/s traffic load, greater *Backoff* causes more packet loss. This is because when many nodes compete for the channel, the channel is saturated, and increasing backoff transmission attempts cannot improve saturation since this results the channel being occupied for even longer and reduces the transmission probability of new packets. Further, more attempts for a packet transmission with a greater *Backoff* value will of course consume more energy as shown in the Figure. For a 38.4Kbps based MICA2, better energy consumption and reliability is expected with smaller *Backoff* value.

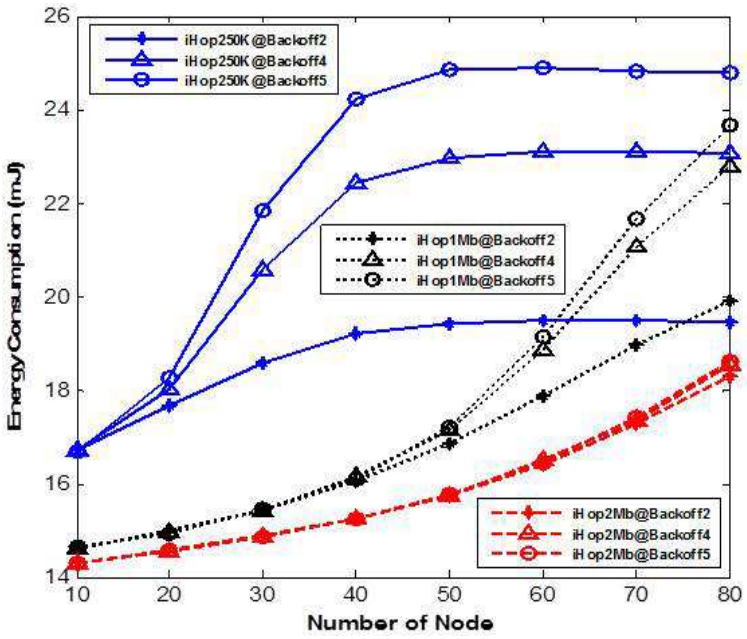

Figure 4.10: Impact of *Backoff* @energy consumption (iHop@Node)

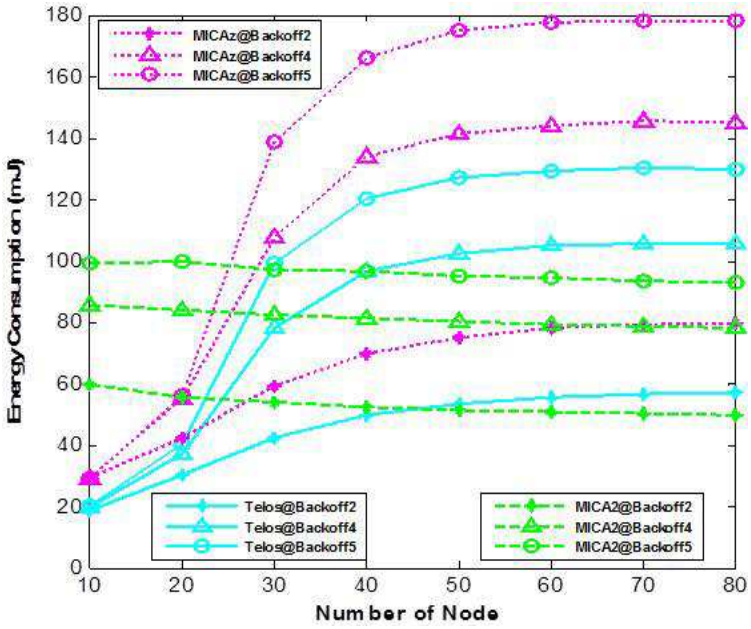

Figure 4.11: Impact of *Backoff* @energy consumption (MICA2, MICAz, Telos)

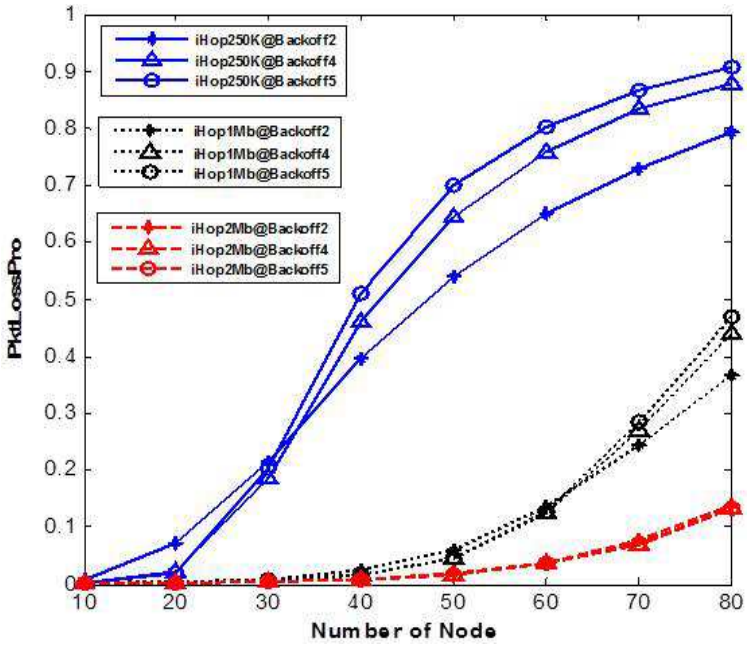

Figure 4.12: Impact of *Backoff* @packet loss probability (iHop@Node)

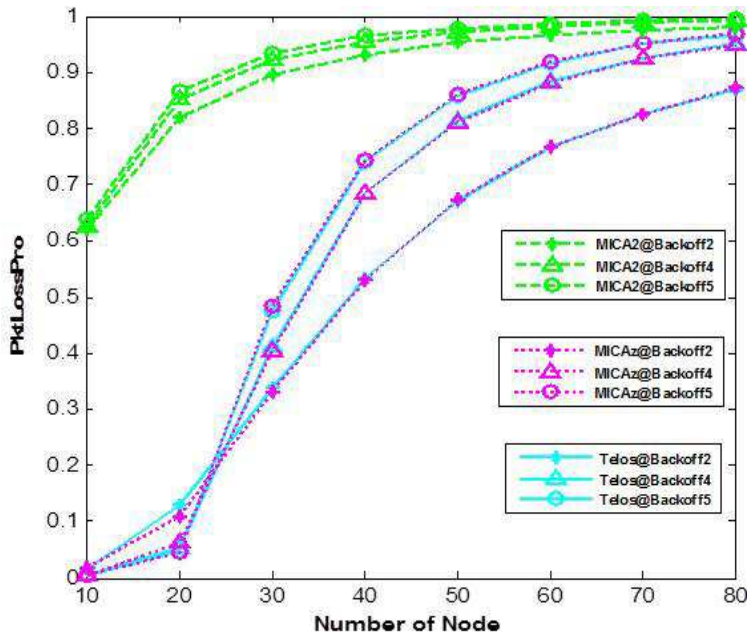

Figure 4.13: Impact of *Backoff* @packet loss probability (MICA2, MICAz, Telos)

At the data rate of 1Mbps, it follows a similar trend to that of the 250Kbps scenario. Compared to the 250Kbps scenarios, the changing point for energy and packet loss starts at about 600pkts/s traffic load, which is because the high data rate can effectively reduce the channel occupation time and hence provides better channel condition for packet transmission.

At the 2Mbps data rate, different *Backoff* values have little impacts on the energy and reliability performance. With a 2Mbps high data rate, the greatly reduced packet/ACK transmission and reception time can provide a very good channel condition, so that the packet can be successfully transmitted after just one attempt, without starting the repeated channel check mechanism. Hence, the same amount of energy is consumed under different *Backoff* values. A smaller packet loss for larger numbers of sensor nodes is introduced by continuous 1+*macMaxFrameRetries* collision failures. According to experimental results about 96.8% packet loss are from such collision failures. The *macMaxFrameRetries* value is set to the same value in this *Backoff* investigation, so the extra energy consumed in such cases are also close and the overall energy consumption under the three *Backoff* values do not have a significant difference.

To sum up, it is suggested that that for smaller sensor network scenarios at the data rate of 250Kbps and 1Mbps, the choice of the *Backoff* value is application-specific for the trade-off between energy consumption and packet delivery performance. For the larger sensor network case as well as for the 38.4Kbps based network, the selection of a smaller *Backoff* value would be better. Finally, for the energy and reliability performance at 2Mbps, the impact of *Backoff* can be ignored.

4.5.4 Impact of *Retries*

In this part, *macMaxFrameRetries* value is represented by *Retries*, and Fig.4.14, 4.16, 4.15 and 4.17 show the results with the *Retries* value of 3, 5 and 7 (*BE* and *Backoff* take the default values of 5 and 4 respectively).

For the 250Kbps and 38.4Kbps data rate based networks, the change of retransmission time only has a slight impact on the performance of packet loss. Due to the node competition for the channel, and according to the simulation results, starting from the 20 nodes case over 90% packet loss is caused by Channel Access Failure (CAF), so different *Retries* values do not make much changes on the reliability. However, for energy consumption, it is clear that the use of smaller *Retries* values across all traffic loads (number of nodes) can save some energy, which is because with lower retransmission times the nodes can reduce the time spent in the active state while sleeping longer for energy-saving purposes.

At the rate of 1Mbps, above a traffic load of 500pkts/s, the increase in *Retries* value does not improve packet transmission. In fact, it leads to a more occupied channel. Since more retransmission attempts for a packet will prolong the packet transmission process, new packets from new nodes have less chances of being able to send out their packets, so the total packet loss could as well as energy consumption.

For the 2Mbps data rate, the same analysis can be applied from 1Mbps, except that energy consumption and packet loss rate start to change at 700pkts/s traffic load, which is due to a better channel conditions compared with the 1Mbps scenario.

In general then, the use of a smaller *Retries* value for the low data rate case would be a better solution in terms of energy and packet loss performance. A greater *Retries* value can be considered for 1Mbps and 2Mbps based networks under lower traffic load, while under higher traffic load a smaller value would be more suitable.

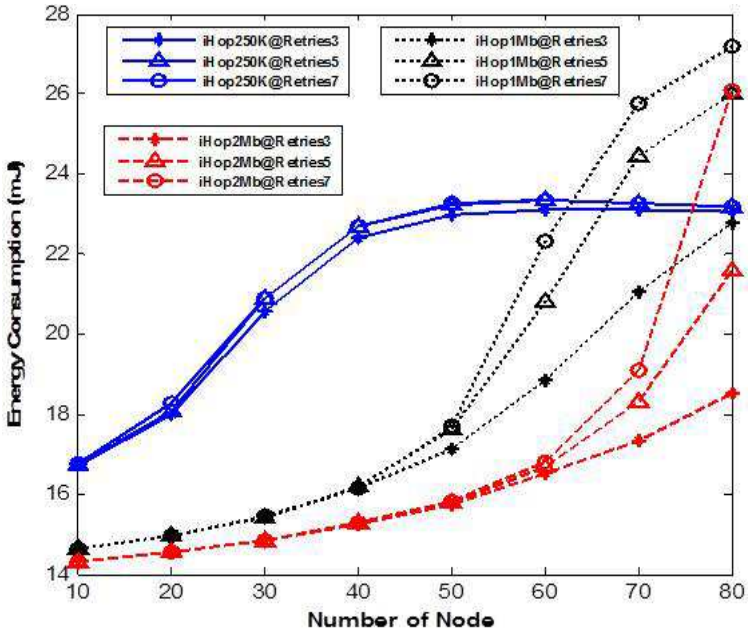

Figure 4.14: Impact of *Retries* @energy consumption (iHop@Node)

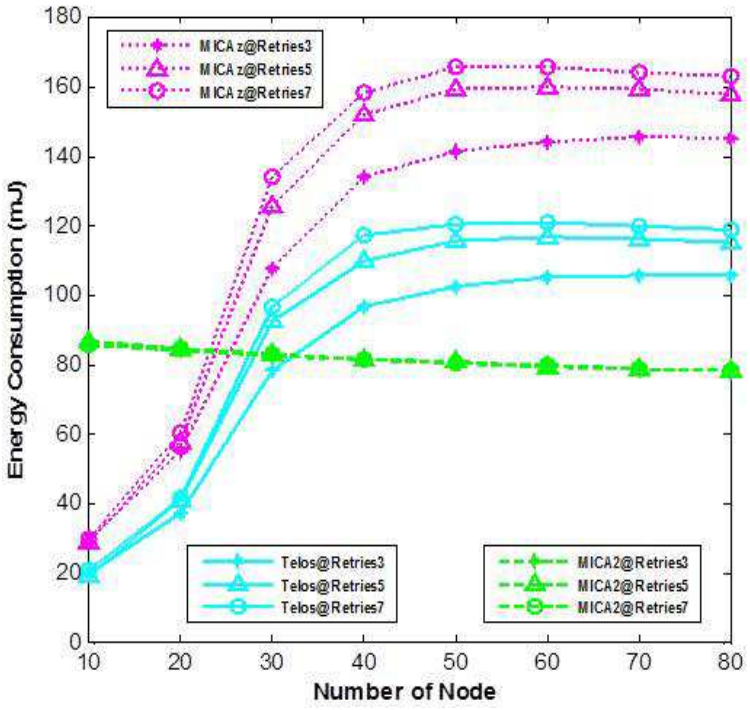

Figure 4.15: Impact of *Retries* @energy consumption (MICA2, MICAz, Telos)

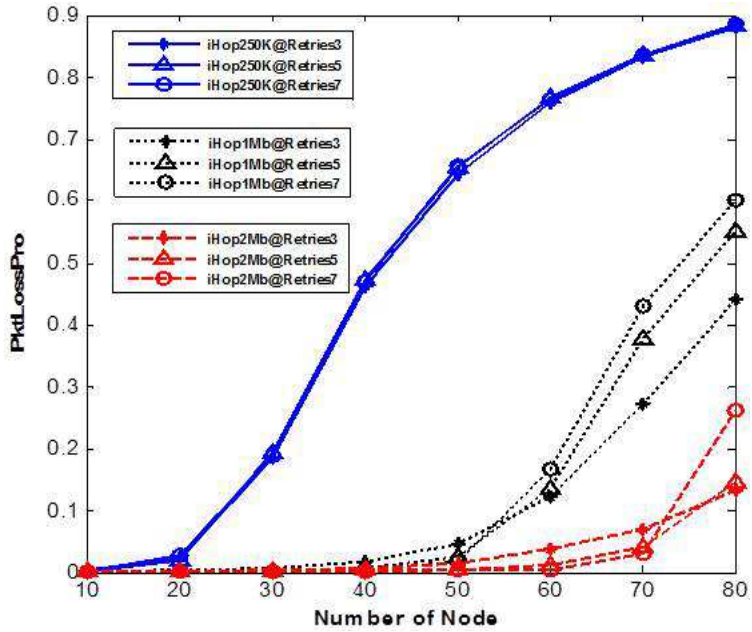

Figure 4.16: Impact of *Retries* @packet loss probability (iHop@Node)

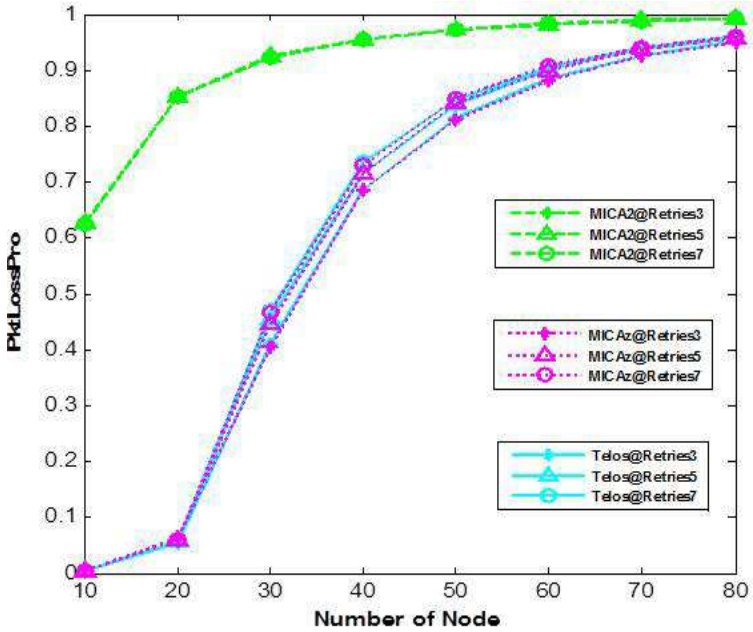

Figure 4.17: Impact of *Retries* @packet loss probability (MICA2, MICAz, Telos)

4.6 Enhanced ShockBurst (ESB) / ShockBurst (SB)

As mentioned in section 3.3.3.2, Enhanced ShockBurst (ESB) is a baseband MAC protocol engine embedded in transceiver chips (nRF24L series), while ShockBurst is only a simple TX/RX protocol without any channel access algorithm. Energy consumption explorations in this section are divided into two parts.

In the first part, the energy consumption performance of both embedded ESB- and SB- modes are examined via the nRF24L01+ transceiver of the iHop@Node. With a high data rate and ultra low power consumption (section 3.4.3), the nRF24L transceiver series based platforms are widely used applied in the health/medical monitoring field, such as ECG (electrocardiogram [187]) application scenarios as described in [188] [189]. Therefore, the investigation of the energy consumption of ESB- and SB- modes is performed via a typical ECG application, where the sensor node is equipped with an ECG lead sampling at 200 Hz [190], and the payload size of each data is 2 bytes. The energy consumption performances are evaluated from data rate, output power, task profiling and packet overhead.

In the second part, the energy consumption and network reliability of the ESB mode are chosen to be compared with the unslotted CSMA/CA algorithm. This is because ESB is not just a TX/RX protocol, it has a simple channel algorithm for auto ACK and retransmission that can guarantee the network performance. To some extent, we can consider that it functions as a simple MAC level algorithm as the unslotted algorithm does.

4.6.1 Energy Consumption of ESB and SB

4.6.1.1 Output Power and Data Rate

This sub-section analyzes the energy consumption of ESB and SB modes under varying output power and data rate scenarios (assume that no PER in the channel). Based on the above mentioned ECG case, a packet containing 2 bytes of payload data is transmitted every 5ms by PTX (Primary Transmitter) and the simulation experiment runs for the transmission of 500 packets. The PRX (Primary Receiver) is configured in the continuous RX mode. For other packet parameters, the default values are used. The results are shown in Fig.4.18.

In each case, the energy consumption of SB-PRX is higher than ESB-PRX, which is because SB-PRX is in the RX mode (highest power consumption mode) all the time, while ESB-PRX needs to switch into a lower consumption TX mode for ACK transmission from time to time, so the total energy consumption for ESB-PRX is reduced. In turn, PTX in SB is more energy-efficient than PTX in ESB, which is due to the fact that ESB-PTX must be in RX mode waiting for ACK after each packet transmission (but this is not required for SB-PTX). Therefore, extra energy is consumed by ESB-PTX. On the other hand, since the higher the data rate, the higher the current consumption in RX mode, energy consumption of both ESB-PRX and SB-PRX increase when the data rate is set high. The situation is however different for PTX in ESB: even though the PTX must be in the highest consumption RX mode for ACK frame, the high data rate can reduce the duration of RX and TX modes, so the overall energy consumption is reduced. Some detailed result comparisons on PTX device are made in Table 4.1.

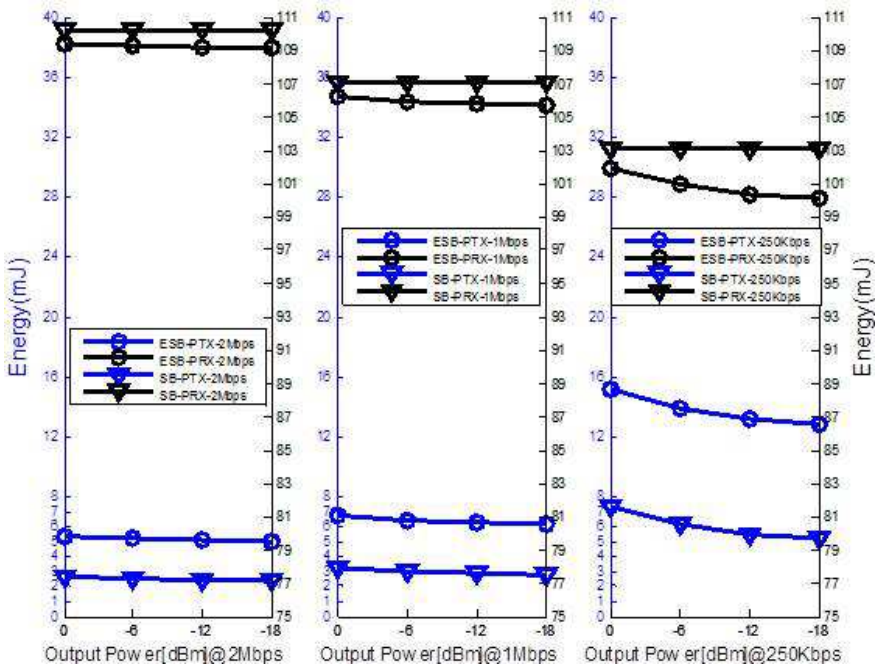

Figure 4.18: Impact of output power and data rate on energy consumption of ESB/SB

Table 4.1: Energy consumption of ESB-PTX and SB-PTX

Date Rate	**ESB(PTX)** **0dBm VS -18dBm**	**SB(PTX)** **0dBm VS -18dBm**
2Mbps	5.35% higher	9.84% higher
1Mbps	8.45% higher	15.65% higher
250Kbps	15.02% higher	28.06% higher
2Mbps-Most energy efficient *250Kbps-Most energy consumption*		
Date Rate	**ESB(PTX)**	**SB(PTX)**
2Mbps@PTX **VS** 250Kbps@PTX	65% total energy is save at the data rate of 2Mbps	83% total energy is saved at the data rate of 2Mbps

4.6.1.2 Detailed Energy Consumption and Task Time Profiling

By using the same application scenario of section 4.6.1.1, Fig.4.19 shows the detailed energy consumption information on 2Mbps data rate and 0dBm output power based ESB and SB mode, and Fig.4.20 gives the time information of each corresponding state, is presented in Fig.4.19.

The results show that for PTX in ESB and SB modes, a large amount of energy can be saved in low-power *STANDBY1* mode (ST1), which takes over 90% task time while contributing less than 4% of the total energy. For PRX in ESB and SB modes, the RX mode takes over 90% task time and also consumes over 95% total energy. In addition, the total energy consumption of PRX in ESB mode is about 20x that of PTX, while in SB mode, this ratio rises to about 42x.

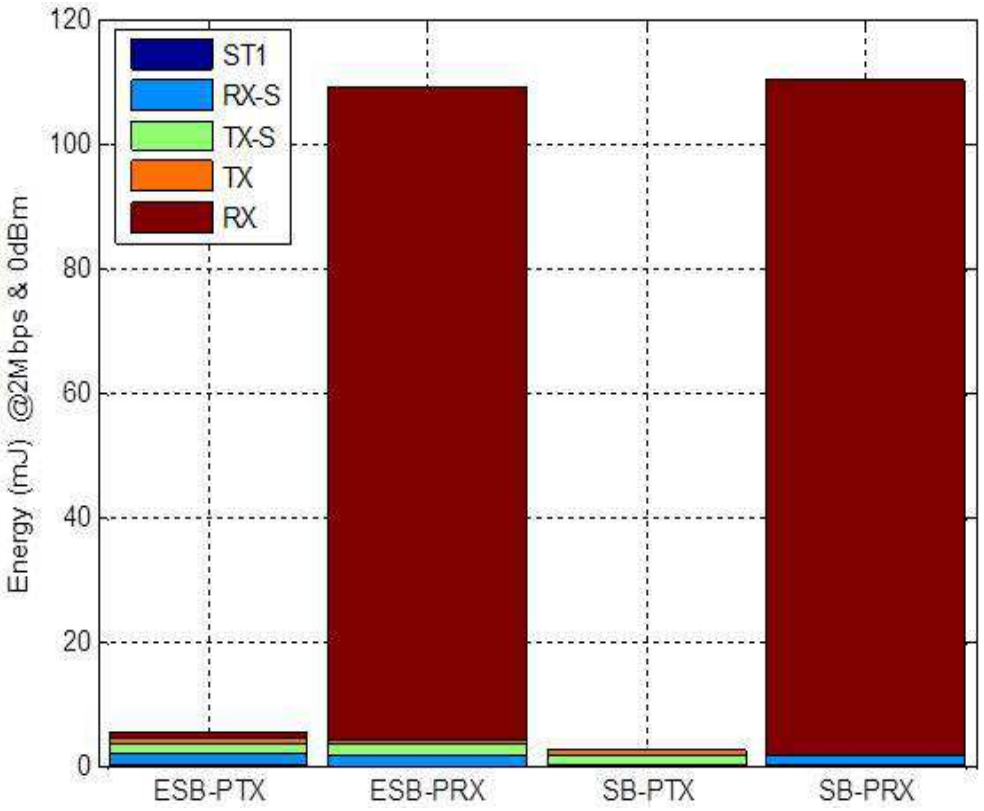

Figure 4.19: Energy consumption of ESB and SB (@2Mbps and 0dBm)

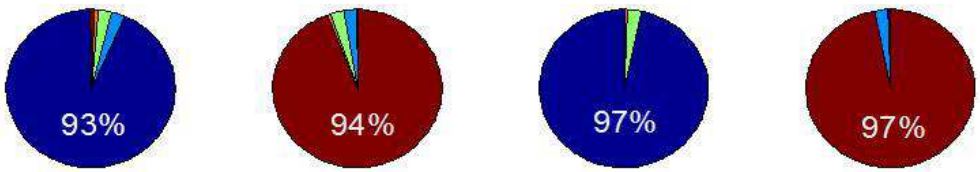

Figure 4.20: Corresponding task time

4.6.1.3 Payload Collection

The payload collection strategy is used for energy-saving in the ESB mode under data rates of 2Mbps, 1Mbps and 250Kbps in this sub-section. For the payload collecting process, each packet with 2 bytes payload is transmitted every 5ms, then 4 bytes payload every 10ms until a 30 byte payload is reached every 75ms. Thus, the total payload during transmission is the same as the overall rate of 0.4bytes/ms. In addition, due to the impact of cutaneous tissue as well as body movement, PER is common in such application scenarios. To simplify the investigation in this section, a simple PER case is introduced into the experiments. In Fig.4.21, 5% PER and 10% PER are considered in the channel model (If packet errors occur, the default maximum 3 times of retransmissions can be used).

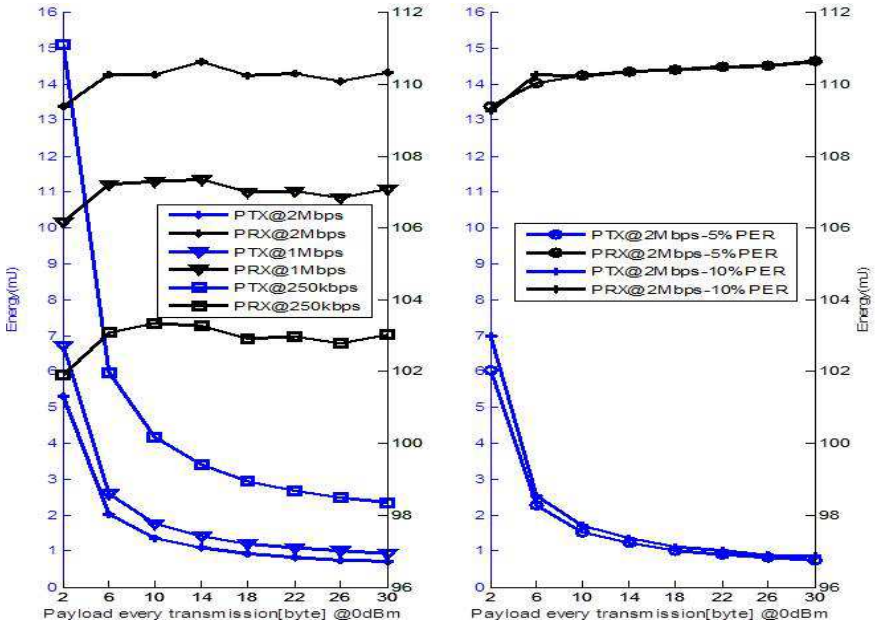

Figure 4.21: Energy consumption with payload collection and PER

The results reveal that the 2Mbps scenario is the most energy-efficient way of using payload collecting for PTX, which can save from 64.9% (2 bytes payload) to 70.3% (30 bytes payload) energy when compared to 250Kbps. For the 1Mbps scenario, the ratios are from 21.0% to 25.3%. Even with introduced PER, the 2Mbps based high data rate can still provide better energy performance than the 1Mbps and 250Kbps scenarios. When considering a 5% PER in the 2Mbps network, 6.6% (30 bytes payload) to 12.1% (2 bytes) more energy will be consumed compared with the ideal 2Mbps scenario, and with 10% PER the ratio rises to 17.7% to 23.9%.

4.6.2 Comparison of ESB and Unslotted Algorithm

Compared with the unslotted algorithm, the ESB mode does not have a random backoff wait period (depending on *BE*) and repeated times of channel check (depending on *macMaxCSMABackoff*). It only has a retransmission mechanism that can be compared. By employing the same scenario from section 4.5, i-

Hop@Node configured as ESB mode are deployed with the traffic load from 100pkt/s to 800pkts/s (10 to 80 nodes), and the impact of 3, 5 and 7 retransmission attempts are evaluated at the data rate of 250Kbps, 1Mbps and 2Mbps. The energy consumption and packet loss performance are shown in Fig.4.22 and Fig.4.23 respectively.

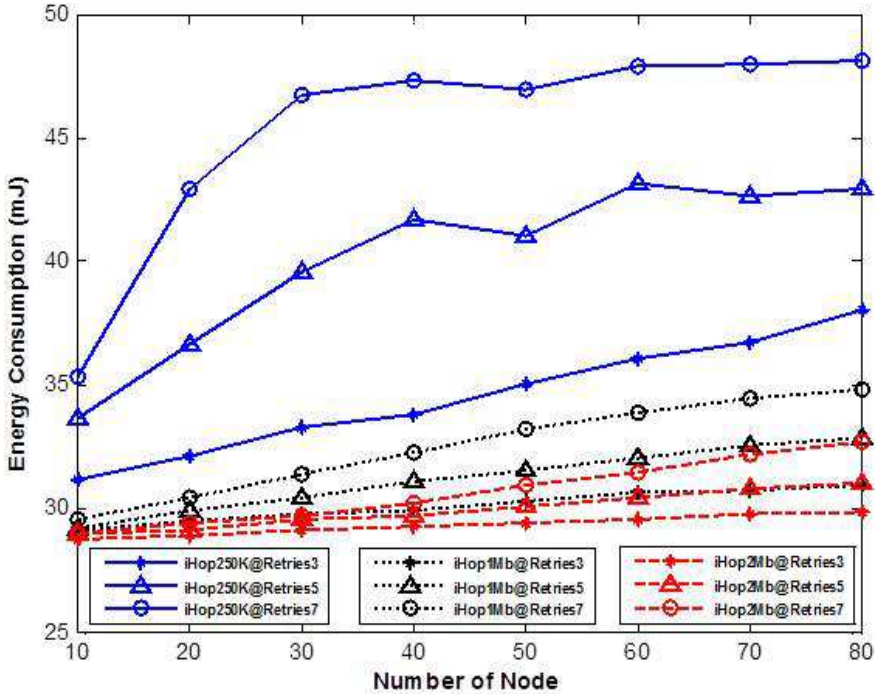

Figure 4.22: Impact of *Retries* @energy consumption of ESB

As a result of lacking an effective CSMA/CA algorithm, ESB does not demonstrate any advantage in terms of energy consumption and reliability. For reliability, the results in Fig.4.23 show that for all three data rates based ESB modes, the packet loss rate under each traffic load is worse than the cases where the iHop@Node uses an unslotted algorithm in section 4.5 (Fig.4.8, Fig.4.12 and Fig.4.16). The reason is that the ESB mode itself has a significant potential for encountering packet loss, which is analyzed and explained as follows. When many sensor nodes are densely deployed and compete for the common channel, the initial transmissions of two nodes are easy to overlap. Hence, data packet collisions will occur and neither of the nodes can receive the ACK frame, since the ESB mode embedded in these two nodes use the same mechanism. After the collision, there are no operations such as random backoff wait or repeated CCA, so such collision problems will happen again during the retransmission process and this kind of situation will continue until the defined maximum number of *Retries*. Ultimately, both of the packets will result in transmission failures. Furthermore, such a case will still appear for the new packet transmission, because both nodes have the same sampling interval, so the transmission time of the new packets will overlap again. Thus, in such conditions, the increase of *Retries* times is useless not only in terms of packet loss, but also in terms of energy consumption. A greater *Retries* value will only result in the transmission time for overlapping packets to be in the retransmission process for a longer time, so extra energy will be consumed

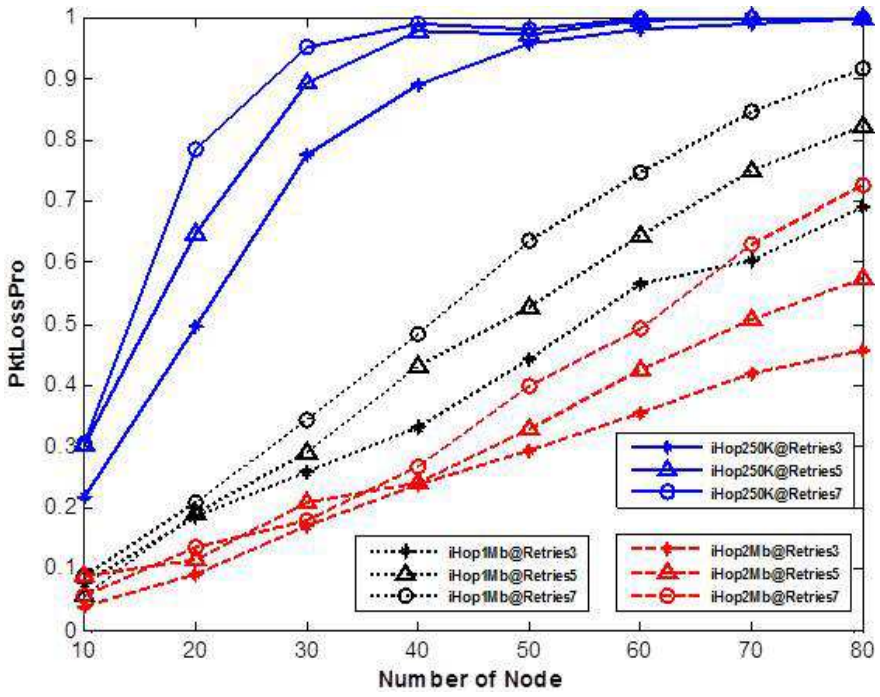

Figure 4.23: Impact of *Retries* @packet loss probability of ESB

for all the three data rate scenarios, as shown in Fig.4.22. The energy consumption in Fig.4.22 is higher than the energy consumption shown in section 4.5 (Fig.4.6, Fig.4.10 and Fig.4.14) under each corresponding case, because most of the energy consumed by the protocol is in the CAF process, which does not involve the power-consuming over-the-air packet transmission process. However, the extra energy consumed by ESB is all based on the retransmission process, which includes many rounds of over-the-air packet transmission, so more energy is consumed under ESB mode. Another reason is that unslotted algorithm has an efficient channel access mechanism, so more sleep time can be expected on each node for the energy saving. While for ESB mode, each sensor node needs to stay in active state for a longer time before sleep because of its simple and ineffective channel access capability. In addition, the above condition could become even worse when the packet transmission of several nodes overlap at the initial round, which can happen in the case of larger numbers of nodes. With a greater *Retries* value, the performance of both energy consumption and packet loss becomes worse compared to the case of smaller numbers of nodes. On the other hand, no improvements can be achieved in both energy consumption and reliability, even by adding a 2^{BE}-1 slot unit based random backoff wait time before each new packet transmission (BE=3, 5, 7) [134].

Overall, after the comparison between ESB and unslotted based networks, it is easy to conclude that an embedded ESB mode has no advantages as concerns the improvement in energy consumption and network reliability under multi sensor node conditions, especially for large numbers of nodes. The simple automatic retransmission and auto-acknowledgement cannot effectively guarantee network performance and also has a high potential for causing large collisions if the packet/ACK transmission time of many nodes overlap.

4.7 Conclusion

In this chapter, the energy consumption performance along with network reliability of low data rate (MICA2/MICAz/Telos/iHop250K) and high data rate (iHop1Mb/iHop2Mb) based networks are evaluated under several different scenarios. Results acquired from the strategy level (random sensing start time, payload collecting), hardware configuration level (data rate, output power) and protocol level (unslotted algorithm, ESB, SB) reveal that:

- the use of random sensing start time and payload collecting can effectively reduce energy consumption.
- the use of low output power can save small amounts of energy if the channel PER is not considered.
- the use of high data rates could be the most energy-efficient method under each scenario.
- there are more advantages in using an unslotted CSMA/CA algorithm for energy consumption as well as network reliability when compared to the use of ESB mode.

Finally, due to the low power consumption of nRF transceiver and PIC16 microcontroller, the iHop@Node always proves to be the most energy-efficient platform when compared to MICA2, MICAz and Telos.

Chapter 5

iMASKO:A GA based Optimization Framework for WSNs

Contents

In this chapter, the design and implementation of a generic GA-based optimization framework iMASKO (**i**NL@**M**ATLAB Genetic **A**lgorithm-based **S**ensor Networ**K** **O**ptimizer) are described. Due to the global search property of genetic algorithms, the framework is able to automatically fine tune hundreds of possible solutions to find the best suitable tradeoff solution. A weighted sum based cost function is explored to measure the optimization efficiency and capability of the framework, while another experiment tests its multi-scenario and multi-objective optimization ability.

This chapter is organized as follows. Section 5.1 is an overview of optimization methods used in the design of wireless sensor networks. Section 5.2 gives an introduction to genetic algorithms (GAs). Section 5.3 presents the motivation for proposing this optimization framework. Section 5.4 shows the architecture of the framework. Section 5.5 lists some performance metrics that are commonly used in the optimization. Section 5.6 illustrates the optimization in a weighted sum based method. Section 5.7 explores the use of optimization in the multiscenario and multiobjective based condition. Section 5.8 gives the setting options of the framework by using command lines. Section 5.9 designs a GUI to make the configuration of the framework more visualized and discusses the generic use of the framework. Section 5.10 is the experimental part. Finally, the conclusion is in section 5.11.

5.1 Optimization in Wireless Sensor Network Design

The integration of sensing, data processing and over-the-air transmission into a single miniaturized device enables the development of wireless sensor networks in many fields of application. However, the weak computation ability, limited storage, short communication range and severe energy constraints to some extent limit their use. Therefore, carrying out optimizations on these elements is very useful for the improvement of network lifetime, the reduction of packet loss, end-to-end delay and other related metrics, all of which are essential to guarantee adequate performance for specific applications. Typically, two classes of objects are optimized in the design of wireless sensor networks: hardware and software.

From the hardware perspective, better energy efficiency can be achieved by optimizing the power consumption of related hardware components. By employing an ultra low power based microcontroller (e.g., MSP430) and configuring it into five different low power modes, a significant amount of energy can be saved. Besides, the reduction of sensing tasks and simplification of data processing algorithms can also be helpful. For the transceiver, some factors such as the modulation scheme, transceiver packet frame and duty cycle can affect power consumption. The use of a high data rate as described in section 4.5, has proved to be not only as a very energy-effective choice, but also a strategy that greatly improves network reliability. In recent years, some emerging radio based technologies such as Bluetooth low energy technology [191], Ultra-wideband (UWB) [192] and ANT [193] also provide excellent choices in lifetime improvement, reliability enhancement as well as short network latency. From the viewpoint of energy supply, since most sensor motes are battery-driven, the battery type, capacity and size play an important role in the node lifetime, cost, weight and deployment ability. Emerging energy harvesting technologies [194] are also highly useful methods for overall energy optimization and power management. In addition, some hardware based algorithms have been proposed for optimizing energy consumption. An adaptive power control algorithm is presented and implemented in [184], by automatically configuring the programmable output power on a transceiver chip according to the distance information between nodes. This algorithm is tested via experiment for the validation of its energy-efficiency in increasing node lifetime. A Dynamic Voltage Scaling (DVS) algorithm [40] applied to microprocessors can also minimize the power consumption by dynamically scaling the supply voltage to match the required performance level. Finally, in the Dynamic Power Management (DPM) algorithm [195], a more traditional method is used by selectively turning off idle state components to save energy.

From the software perspective, the optimization method can be grouped into three categories: the development of new communication protocols (MAC and routing) for optimization, the adoption of energy-aware strategies for optimization, and the configuration and exploration of the optimal set of existing protocols for optimization. Firstly, building on the contention-based scheme for collision avoidance and reliable transmission, both S-MAC [196] and T-MAC [54] are proposed to synchronize communication schedules and listening periods to minimize latency, while reducing energy consumption by turning off the radios during sleep periods. Through the use of low-power listening approaches, WiseMAC [197] and B-MAC [51] can save more energy from idle listening. For most of the new proposed routing protocols, optimizations focus on how to select the shortest path for energy-saving, while at the same time guaranteeing the reliability of the network by reducing the number of communication hops. Secondly, the uses of strategies for optimization include in-network processing and data aggregation. In-network processing in wireless sensor networks typically

involves operations such as data compression and fusion. Despite the cost of increased latency (more time is spent in data processing), the impacts of this approach on energy reduction in [198] are always found to be significant. The idea of data aggregation is to gather the data from different source nodes to be forwarded onto further destination nodes, and is considered an essential part in routing protocols of wireless sensor networks. The aggregation process helps eliminate redundancy from packet overhead, minimize the number of transmissions and thus save the total energy [199]. For the final method, by exploring different parameter configurations of a beaconless IEEE 802.15.4 network via simulation, the optimal parameter settings of unslotted CSMA/CA algorithm under different traffic loads are suggested in [186].

However, when considering increasingly complex system designs, the above-mentioned optimizations cannot be considered to be comprehensive, since they only focus on optimizing a specific objective. Even when several objectives are under evaluation, their optimization procedures are separate. Thus, such optimizations are inefficient to provide a quick overview of problem solutions. In the meantime, they cannot satisfy the requirements of application-specific wireless sensor networks, in which many design criteria need to be optimized simultaneously.

Thus, in the following section 5.2 we give an overview of genetic algorithms (GAs) as an intelligent and efficient optimization technique, and present some of GA based optimizations in wireless sensor networks.

5.2 Introduction to Genetic Algorithm (GAs)

5.2.1 Genetic Algorithms (GAs)

Genetic Algorithms (GAs) are stochastic search engines that mimic natural selection and biological evolution processes. Initially, a population holds randomly selected individuals that are generated from the candidate solution space. Then, these individuals are made to evolve towards an optimal solution over successive generations via selection, crossover and mutation processes. For GAs, the fittest individuals are selected to generate a new population of individuals at each step in the hope of improving the solution quality [200]. Fig.5.1 shows a brief flowchart of GAs.

Apart from GAs, several other optimization algorithms are also widely used such as gradient-based local optimization, random search, stochastic hill climbing, simulated annealing and symbolic artificial intelligence [13]. However, when compared with these conventional optimization methods, GAs are considered to be highly efficient techniques and distinguish themselves from traditional approaches, since their research is based on a whole population of individuals in parallel calculations rather than a single point, which improves the chance of achieving the global optimum solution and helps to avoid local stationary points. Further, the use of a fitness function (rather than derivatives) for evaluation can extend GAs to any kinds of continuous or discrete optimization problems. As a domain-independent search technique, GAs are ideal for applications where domain knowledge and derivative information are difficult or impossible to provide [201]. In other words, tedious and knowledge-based processes can be greatly reduced, which is of especial interest for inexperienced designers, since only the fitness function and corresponding fitness levels influence the whole search process [202] [203]. Finally, some advantages of GAs are summarized and listed in Table 5.1.

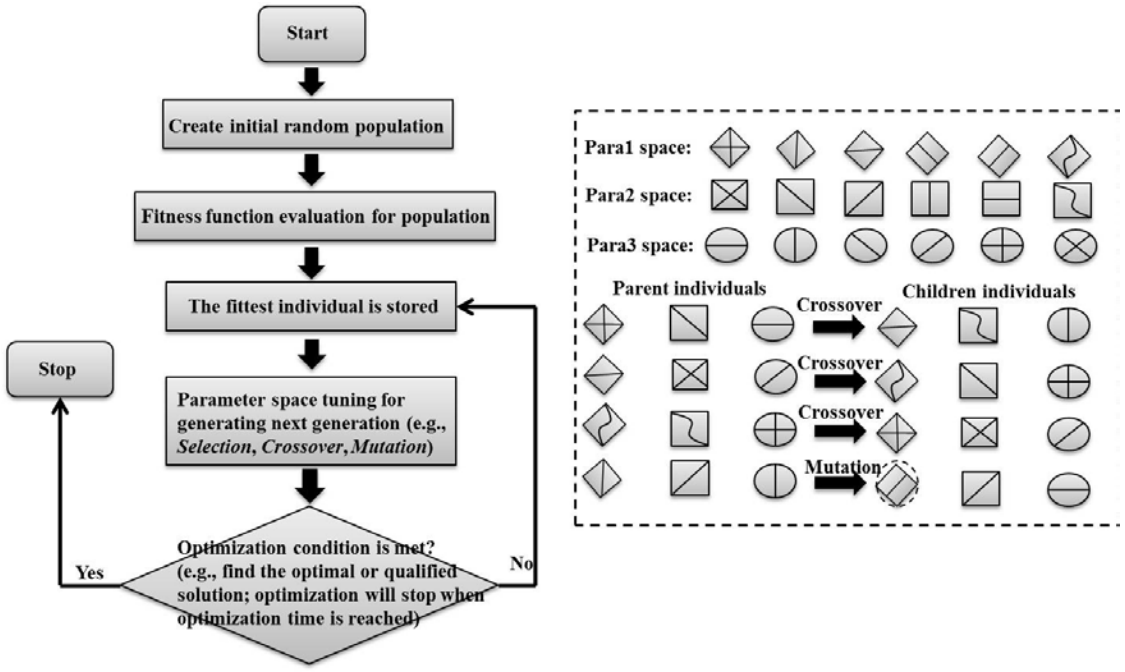

Figure 5.1: A brief flowchart of genetic algorithms

Table 5.1: Advantages of GAs compared to the conventional methods

Advantage of GAs:
√ Parallelism, efficiency, reliability, easily modified and adaptable to different problems
√ Large and wide solution space search ability
√ Non-knowledge based optimization process
√ Use of fitness function for evaluation
√ Easy to discover global optimum, avoid trapping in local optima
√ Capable of multi objective optimization, can return a suite of potential solutions
√ Good choice for large-scale/wide variety of optimization problems

5.2.2 GA based Optimization in WSNs

Genetic algorithms have been applied and to solve many engineering problems with GAs' excellent search abilities, and the use of GAs in wireless sensor network design has been proved to be successful by a number of works. The most common use of GAs is how to achieve an energy-aware network under specific routing conditions, which includes the finding of proper cluster heads for data aggregation in the network; the selection of the optimal path to reduce hops (and thus transmission energy); and at the same time the reduction of channel contention. Such efforts for energy-efficiency can be found in [204] [205] [206].

In [207], a GA-based methodology is proposed for adaptive wireless sensor network design. A fitness function that incorporates seven design parameters from the aspects of the specific application, network connectivity and energy consumption is used for the evaluation, so that during the optimization process many requirements (such as the status of the sensor nodes, the selection of the cluster heads as well as the distance between nodes) are taken into account for the design of a reliable and energy-aware wireless sensor network.

In recent years, the use of GAs to find the optimal placement of wireless sensor network nodes before real deployment has gained wide attention from many research groups and academic institutions. However, in the placement problem, the optimization process is required to tune multiple conflicting objectives, e.g., reducing the energy consumption while achieving large coverage and robust connectivity, so Multiobjective Genetic Algorithms (MOGA) are adopted in [208] and [209], aiming to find a deployed network arrangement to maximize area coverage, minimize energy cost (maximize lifetime), and maintain good connectivity.

5.3 Motivation for Proposing the Optimization Framework

While previous optimization efforts on hardware have been mentioned, the limited energy supply, low processing ability, low data rate and short communication range inevitably affect the performance behavior of the whole network and also limit their further potential utilizations. The solving of these hardware-based issues to improve performance quality takes some time, such that it is difficult to satisfy rapidly evolving applications and requirements. In comparison, efforts on non-hardware (software) aspects, (for instance the development of energy-efficient and robust protocols, the design of specific communication strategies or the exploration of configurations on existing protocols) can provide a quick alternative for performance optimization on given scenarios in terms of various metrics (section 5.5).

The widely accepted methods of evaluating the above solutions before real deployment can be done through software modeling (theoretical model and simulation) or real-world testbed experiments. The latter are very time-consuming, with repeated experiments and large amounts of raw data required for subsequent evaluation, while the use of simulations/theoretical models can provide a more efficient way. When compared with simulation, theoretical models are faster to evaluate but are also sometimes idealized and inaccurate for realistic conditions. Therefore, simulation is considered to be a more suitable trade-off between accuracy and efficiency. Nevertheless, in order to get a satisfactory performance for the given scenario, the selection of the optimal WSN infrastructure, out of hundreds of potential solutions, is still required before real deployment. Thus, the use of full and exhaustive simulation to explore all possible solutions is also regarded as a very time-consuming and inefficient method.

Thus, how to effectively and accurately achieve an optimized simulation result from a large solution space is a challenge. As a fast and efficient search engine, the use of GA gives a solution. However, almost all GA-based optimization applications in wireless sensor networks mentioned in section 5.2.2 are based on theoretical and analytical models. In addition to the issues of inaccuracy for realistic situations, such models always require expert knowledge in related domains as well as strong mathematical background. Therefore, it prevents many non-expert researchers and some ordinary users without much knowledge about sensor networks from quickly accessing the optimization process. So, in order to reduce time spent on evaluating all possible solutions and avoiding expert knowledge based analysis process, a generic optimization framework is proposed and will be described in detail in the following sections.

5.4 Architecture of the Optimization Framework (iMASKO)

In this section, a generic GA-based optimization framework called iMASKO (**i**NL@**M**ATLAB Genetic **A**lgorithm based **S**ensor Networ**K** **O**ptimizer) is proposed. It is able to quickly and efficiently explore simulation-based results or the theoretical model-based results. Note that in this context, iMASKO is applied to automatically fine-tune the results from iWEEP's SystemC-based simulation. In practical use, the optimization of related parameters in the design space can help find the application-specific solution to achieve the most optimal network performance for the given scenario before real deployment. By using iMASKO, only two things are needed: (i) a set of input parameters in the design space, and (ii) a user defined fitness function. As long as the system/model under evaluation can return the required data metrics to the fitness function, the detailed implementation of the system/model (simulators / emulators / mathematical models) can be ignored, which is of especial interest for non-experts in wireless sensor network design. Finally, the configuration of the GA optimization process is very simple in iMASKO for users, since the interfaces of the corresponding algorithm parameters are provided and the configurations can be done via command lines and a MATLAB-based GUI.

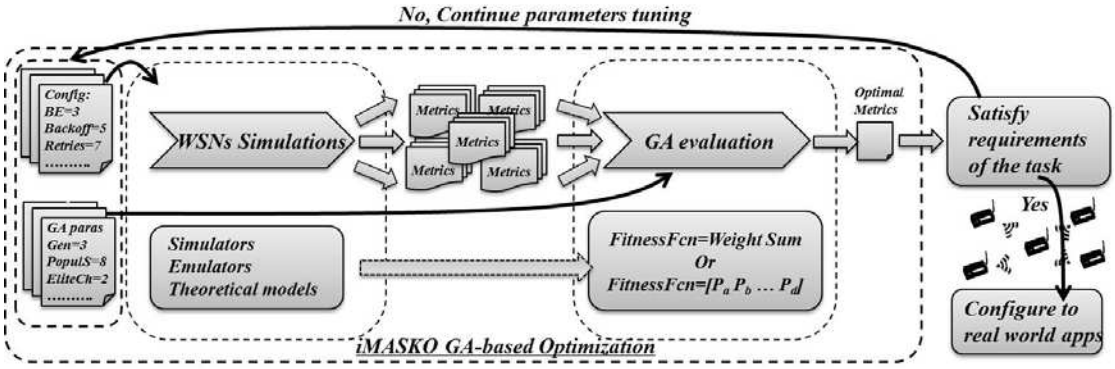

Figure 5.2: Workflow of iMASKO

Firstly, a brief workflow of iMASKO is presented in Fig.5.2 and the detailed optimization process of iMASKO is given in Table 5.2. Before starting the optimization, relevant parameters are fed to the simulation environment and to the GA evaluation engine respectively. Once the corresponding metrics have been

Table 5.2: Pseudo code of iMASKO optimization process

iMASKO Optimization Process:

Input initial solution: *X,P #X-parameter space, P-performance space*
1: *assume that:* $X_{best} \leftarrow X$, $P_{best} \leftarrow P$
 # Starting optimization until stop condition met
2: **while** stop condition not met then **do** *#condition (e.g., max generation exceeded)*
3: $X_{next} \leftarrow$ *generate(X) #GA operations: selection, crossover, mutation*
 # Evaluations are proceeded in parallel
4: $P_{next} \leftarrow$ *evaluate*(X_{next}) *#evaluate simulation results from iWEEP*
5: $\triangle cost \leftarrow$ *compare*(P_{next},P) *#*$\triangle cost = P_{next}$-$P$
6: **if**($\triangle cost \leq 0$)
7: $X \leftarrow X_{next}$, $P \leftarrow P_{next}$
8: **if**($P \leq P_{best}$)
9: $X_{best} \leftarrow X$, $P_{best} \leftarrow P$
10: **end if**
11: **end if**
12: **end while** *# when condition is met*
13: **return/output:** X_{best}, P_{best}

*__Note:__ *process 5 is not necessary, when considering the case in section 5.10.1, where one population size is used*

generated from simulation and processed within the fitness function, the GA evaluation engine will decide whether the final result can satisfy the criteria of the given task. If so, the solution can be considered and applied in real applications. Otherwise, another cycle of the parameter tuning will start until the qualified solution for the task is found.

In Table 5.2, the typical steps of optimization are given by easily identified syntax: generate, evaluate and compare. The parameter space X could be a set of communication strategies/protocols, different configurations of a given protocol, or different application scenarios as long as the parameters that under the investigation can be represented as the qualified individuals in GAs. For the performance space P, it contains the metrics that can reflect network performance presented in section 5.5. The metrics are returned by the fitness function in two ways: one is to combine all related metrics into a weighted sum (section 5.6), while another is to return separated and independent metrics for multi-objective optimization problems (section 5.7).

5.5 Performance Metrics

Despite the widespread use of wireless sensor networks and the variability in various application-specific requirements, the performance metrics that reflect the most fundamental characteristics of wireless sensor networks have always been the same, and can be classified into three major aspects: energy consumption, network reliability and network delay.

Energy Consumption: in most application scenarios, the sensor network must run for a long period

of time to fulfill the given task without human interference (e.g., for battery recharge), so energy saving has always been a significant concern for extending the lifetime of the sensor network/node. Otherwise, the network will not remain operational until the required task is completed. Since the energy consumption of the microcontroller ($\mu CEnergy$) and transceiver ($TraEnergy$) are the most power consuming parts, most works focus on saving the energy from these two parts at both hardware and software levels. In hardware, ultra low power devices are used, especially in medical/health-care applications [188] [189] [191]. In software, as mentioned in section 5.1, various kinds of energy-efficient MAC protocols, duty-cycling based communication strategies and data aggregation routings are proposed and measured. All kinds of efforts are made to achieve an energy-aware sensor network, and according to different requirements, energy performance could also be measured and expressed in many ways (e.g., power consumption-mW, μW, lifetime-*days, months, years*).

Network Reliability: for this metric, packet loss probability (*PktLossPro*) or packet delivery rate (*PDR*) are often adopted as the evaluation standard. *PDR* here is defined as the ratio of the number of packets successfully received (successful receiving of ACK) to the total number of packets that are sensed for transmitting. *PktLossPro* is the probability that a packet sensed for sending will be dropped or fail to be transmitted (*PktLossPro=1-PDR*). If a MAC layer algorithm is used, such as the unslotted/slotted CSMA/CA algorithm of IEEE 802.15.4, the packet loss can take place due to the packet drop in channel access failures (*PktCAFs*) and collision failures (*PktCFs*). *PktCAFs* denotes that a packet encounters (1 + *macMaxCSMABackoffs*) consecutive CCA failures, while *PktCFs* occurs when it suffers (1 + *macMaxFrameBackoffs*) times of collision failures, which take place during the packet transmission or transmission of the ACK frame. Besides, the loss of packet can also be caused by packet overflow (*PktOverflw*), which means that before starting to transmit the pending data packet (pending in MCU), a new data packet is sensed and replaces the pending packet. Therefore, evaluations can be made in detail with *PktCAFs*, *PktCFs* and *PktOverflw* in addition to *PktLossPro* and *PDR*.

Network Delay: packet latency is usually used to evaluate network delay. Packet latency can be divided into three sub-types: successful packet latency (*SucPktLatency*), all packet latency (*AllPktLatency*) and data packet latency (*DataPktLatency*). *SucPktLatency* is defined as the time interval between the instant at which the data packet is sensed and the instant at which is successfully transmitted by receiving its ACK frame. Compared to *SucPktLatency*, *AllPktLatency* takes both successful packet transmission and failure packet transmission (caused by *PktCAFs* or *PktCFs*) into consideration. *DataPktLatency* is calculated as the time interval from sensing the data packet to the receipt of this packet by the sink node (or coordinator).

To sum up, the fundamental features of varying WSNs applications can be evaluated from the above three major aspects, and the detailed performance metrics presented in each of the three major aspects can be used simultaneously to provide a more comprehensive insight for the given scenario.

5.6 Cost Function (Weighted Sum)

As the metrics listed above, the performance evaluation of the specific application scenario is an overall and comprehensive process which must consider multiple objectives at the same time. The evaluation engine of iMASKO returns a set of performance metrics $\mathbf{p}$ (section 5.5) that need to be optimized/improved simultaneously and is given as:

$$min_{(X)}\mathbf{p} = [p_1, p_2, p_3, ...p_n] \tag{5.1}$$

where X is the parameter space. However, the solutions that can simultaneously optimize every metric are difficult to find for such a multi-objective problem. Therefore, a commonly used trade-off method combines different objectives into a linear cost function presented as a weighted sum:

$$min_{(X)} = cost(\mathbf{p}) = \sum_{i=1}^{n} w_i \cdot p_i \tag{5.2}$$

where w_i is the weight vector that is designed to emphasize the importance of each performance metric p_i. X is the parameter space, which can consist of different types of design parameters for the sensor network as shown in [207], or it can be just a set of protocol configurations if only the communication protocol is under investigation.

Thus, the multi-objective problem becomes single objective and the optimization can proceed by directly evaluating the value of the cost function. Note that this cost function in GA-based optimization is also known as the fitness function. Based on the application-specific requirements at hand, the designers can decide whether to choose all or only part of the most competing performance metrics for the fitness function. An example of a fitness function in terms of energy consumption, network reliability and network delay can be presented as follows:

$$f_{performance} = w_1 \cdot nodeEnergy + w_2 \cdot PktLoss + w_3 \cdot SucPktLatency \tag{5.3}$$

However, even with the same importance on each metric, each contribution to the weighted sum could also be different, since the scale of each metric is not the same. Hence, percentage data values are preferred to be used to reduce the impacts caused by the metrics' scale. The above formula is thus modified as:

$$\begin{aligned} f_{performance} =& w_1 \cdot (nodeConsumedEnergy/TotalEnergy) + \\ & w_2 \cdot (PktLoss/TotalSensedPkt) + \\ & w_3 \cdot (SucPktLatency/SampleInterval) \end{aligned} \tag{5.4}$$

where *nodeConsumedEnergy* denotes the average energy consumption on every sensor node. *TotalEnergy* represents the maximum energy volume that can be used on each node. Other metrics are self-explanatory and some have been defined in section 5.5.

5.7 Multi-objective GA for Pareto-front Optimization

Despite the fact that the evaluation by a weighted sum cost function facilitates the optimization process to a great extent, the most significant limitation of this process is that the metrics (e.g., energy consumption-*mJ*, latency-*ms*, packet loss-*how many*) used in the cost function formula are sometimes not in the same scale.

Even with the use of probability values in the experimental part, the equality of scale and avoidance of implicit weighting cannot be strictly guaranteed, considering that if the given energy volume for each node is set to be a large value (e.g., 10000 mJ) or that if a high maximum latency is used as the dividend, then the importance of energy and latency impacts will be reduced.

A more suitable solution to consider trade-offs among different metrics is Pareto-front based multi-objective optimization. As mentioned previously in section 5.6, simultaneous multi-objective (equation 5.1) based techniques rarely exist in conventional optimization methods. However, the GA based multi-objective optimization provides an alternative, since the parallelizable characteristics of GA (represented by the population size) can help evaluate many different sets of solutions in the parameter space simultaneously, which greatly improves the efficiency. The selection of the best individuals is based on the Pareto-front ($\partial F \prec$, F-solution space), which can be described as a solution $\mathbf{f} = [f_1, ..., f_n]$ dominating $\mathbf{f}^* = [f_1^*, ..., f_n^*]$. This condition is true if each parameter of $\mathbf{f}$ is not greater than the corresponding parameter in $\mathbf{f}^*$ and there is at least one parameter that is less, i.e. $\mathbf{f_i} \leq \mathbf{f_i^*}$ for each i and $\mathbf{f_i} < \mathbf{f_i^*}$ for some i. This is presented as $\mathbf{f} \prec \mathbf{f}^*$ to mean $\mathbf{f}$ dominates $\mathbf{f}^*$, and the total description in mathematics can be given as follows:

$$\forall_{(i\in 1,...,n)}(f_i \leq f_i^*) \wedge \exists_{(i\in 1,...,n)}(f_i < f_i^*) \tag{5.5}$$

In other words, the Pareto-front is part of the boundary of performance space. On this boundary, no solution is better in criteria that makes at least one performance metric better without making other metrics worse. The Pareto front captures the trade-offs between competing performance metrics and identifies the solutions that are non-dominated, as shown in Fig.5.3.

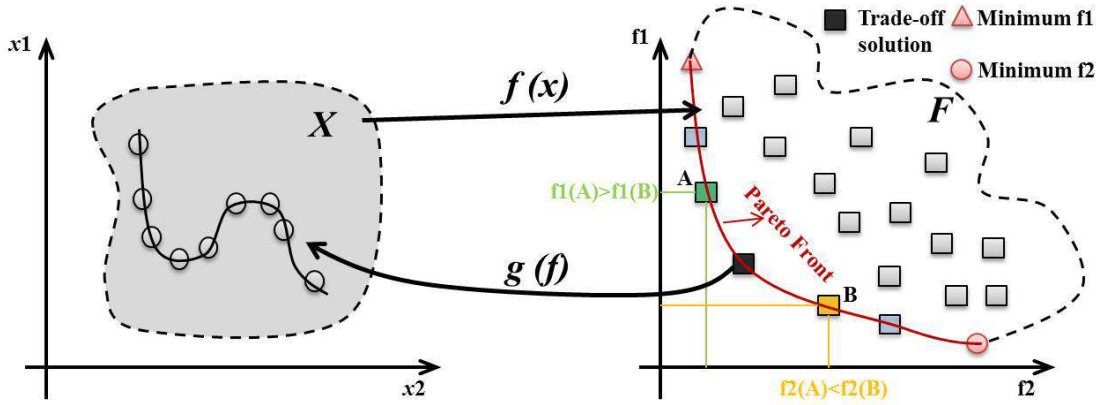

Figure 5.3: Pareto-front optimality

In addition, with design problems becoming more complex and the system under investigation integrating multiple functionalities, performance metrics of several different cases need to be evaluated at the same time. This cannot be achieved if multi-objective optimization only focuses on a specific condition. Therefore, a multi-scenario optimization method is proposed and already has applications in many engineering problems like [210] [211]. The use of this multi-scenario based multi-objective optimization is to find a robust solution which can give the optimal performance for all possible case scenarios and it can be presented as:

$$min_{(X)}\mathbf{p} = [p_{a1}, p_{a2}, ..., p_{an}, ..., p_{m1}, p_{m2}, ..., p_{mn}] \tag{5.6}$$

where X is the vector of design parameters, $\mathbf{p}$ denotes a set of related performance metrics of several scenarios which are returned by iMASKO evaluation engine from the simulations, p_{a1} *to* p_{an} and p_{m1} *to* p_{mn} represent the performance metrics of scenario a and scenario m respectively.

Take two primary objectives, energy and reliability (packet loss), as the example in the optimization of network performance. If these two metrics need to be evaluated under different scenarios before practical use, the following four-dimensional (4D) front in Fig.5.4 is able to transpose the concept for the case of multiple scenarios and help explore trade-offs for the metrics.

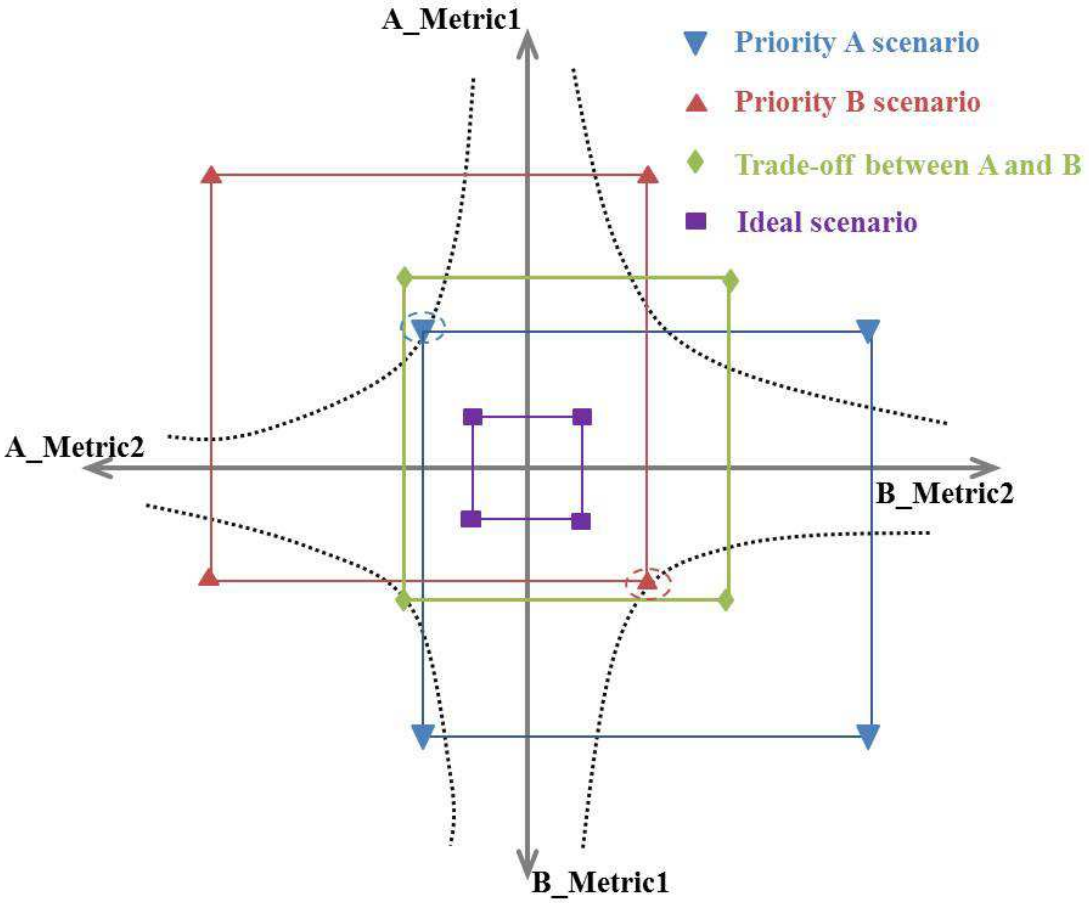

Figure 5.4: 4D Plot for multi-objective based multi-scenario optimization [202]

In this 4D coordinate, the dotted lines are plotted in quadrant one and four are the Pareto fronts. The rectangles are signs of metrics balance for different scenario requirements according to the decision maker. In this 4D Pareto front, *A_Metric2* and *B_Metric1* axes are reversed to make the plot clearer, and quadrant two and three are used as the auxiliary quadrant to help locate the rectangle tradeoff markers.

5.8 Optimization Framework (iMASKO) Options

iMASKO provides a series of interfaces to configure the optimization/evaluation process by command lines, which are shown in *Italics* and **bold** letters in this section. iMASKO command line options consist of five parts which are platform selection, simulation parameter configuration, GA configuration, seed generator and result saving respectively.

Platform selection: this part is to load the mote platforms. Executable files of SystemC simulation are placed under the same folder, which contains all the mote platforms used in Chapter 4 (iHop@Node, Telos, MICAz, MICA2). The option ***-n <platform>*** is adopted to select the specific mote platform for the optimization experiments. The example command line can be given as: > ***./imasko -n Telos***.

Simulation parameters config: this part is used to set the corresponding parameters of SystemC simulation, such as sample rate, sample times, CSMA/CA algorithm, output power, receiver sensitivity and independent simulation run times. The corresponding options in iMASKO for the above parameter settings are ***-sr <samplerate>, -st <simulationtimes>, -alg <algorithm>, -op <outputpwr>, -rs <sensitivity>*** and ***-r <runs>***. Among the options, ***samplerate, simulationtimes, algorithm (0-unslotted, 1-slotted), outputpwr, sensitivity*** and ***runs*** are all integers. An example command line would be: > ***./imasko -sr 10 -st 100 -alg 0 -op 0 -rs -95 -r 50***, which means that the sample rate is set as 10Hz, the simulation runs for 100ms, unslotted CSMA/CA is selected, output power is set as 0dBm, receiver sensitivity is configured as -95dBm and finally 50 independent simulations (different seeds) are required for average results. Default values will be used if the corresponding parameters are not set in the command line.

GA config: GA related parameters are configured in this part. Parameter space under GA optimization can be configured by setting their upper and lower boundaries. In the experiment of this work, the options ***-lbminBE <minBElb>, -ubminBE <minBEub>, -lbmaxBE <maxBElb>, -ubmaxBE <maxBEub>, -lbBackoff <Backofflb>, -ubBackoff<Backoffub>, -lbRetries<Retrieslb>*** and ***-ubRetries<Retriesub>*** are used to specify the lower and upper bounds of the unslotted algorithm's four parameters, which are *macMinBE*, *macMaxBE*, *macMaxCSMABackoffs* and *macMaxFrameRetries*. These boundary values are used to limit the parameter space range during the whole optimization process and they can also be employed to initialize the *PopInitRange* parameter in GA at the very beginning of the optimization. Besides, other commonly used GA parameters are also configurable by command lines, the provided parameters include GA's *PopulationSize*, *Generations*, *CrossoverFraction*, *CreationFcn*, *SelectionFcn*, *MutationFcn*, *CrossoverFcn*. The corresponding options in iMASKO are ***-ps <populationsize>, -g <generations>, -cf <crossoverfraction>, -creatf <@creationfunction>, -selectf <@selectionfunction>, -mutf <@mutationfunction>,*** and ***-crossf <@crossoverfunction>***. Among these options, *populationsize* and *generations* are integers, *crossoverfraction* is a float number within the range 0 through 1. Finally, *creationfunction*, *selectionfunction*, *mutationfunction* and *crossoverfunction* are the names of the functions which are provided by the GA library (e.g., *@gacreationuniform*, *@selectionstochunif*, *@mutationgaussian*, *@crossoverscattered*) or the user custom functions. Likewise, if the parameters are not set via the command line, default values will be used. An example command line can be: > ***./imasko -ps 20 -g 100 -cf 0.8 -mutf @mutationgaussian***.

Seeds Generator: due to the quad-core CPUs of the server, each simulation actually consists of four

parallel and independent (independent seeds) runs for average results, so the use of ***-s1 <seeds1>, -s2 <seeds2>, -s3 <seeds3> and -s4 <seeds5>*** (integers for *seeds1, seeds2, seeds3* and *seeds4*) can specify different seeds for the four independent runs, and all the generated seeds can be stored and used in the optimization process to guarantee reproducibility of results.

Result Saving: the use of ***-save <filename>*** can save the final result file with the user defined file name and with any suffix. As an example: > ***./imasko -save GAresults.log***.

Since the requirements of different tasks are application-specific and vary all the time, the complete configuration of iMASKO options via command lines not only satisfies the varying requirements, but also provides a flexible and effective way to set related parameters in the optimization/evaluation process for the given scenario.

5.9 GUI of iMASKO and Genetic Use of iMASKO

Since iMASKO command line options are not intuitive especially for non-experience designers and users, a MATLAB based GUI has been developed to link simulation and optimization, which facilitates configurations on both sides and makes the evaluation process visualizable. The GUI is shown in Fig.3.14, where all parameters mentioned in the previous section can be set in the corresponding edit box. Default values are used if the parameters are not set. All these parameters are passed to the simulation and GA evaluation engine by pressing the 'Apply' button, and the evaluation will start when a specific platform is chosen on the right side of the GUI.

Except for the multiple and detailed configuration interfaces support, iMASKO is generic in its use of the optimization/evaluation process. In this work, the fitness function uses iWEEP's SystemC-based simulation results. However, the fitness function in iMASKO can be of multiple types as long as it provides parameter space inputs and performance metrics outputs. Thus, results from other well-known WSN simulators such as NS-2, OMNeT++ [212] and Prowler [8] can also be used under the evaluation of iMASKO, even if the detailed implementation and knowledge of such simulations are unknown. For instance, an executable C++ file of NS-2 and OMNeT++, as well as MATLAB's m-file of Prowler are all able to act as the fitness function, since the required performance metrics in the fitness function can be generated after the execution of these files. Besides, in some cases if the simulation process is controlled by a shell script, then such a shell script represents the fitness function. Furthermore, if the simulation is implemented directly in MATLAB as a theoretical model, then this m-file function can be used as the fitness function. Therefore, iMASKO GUI provides for path loading to help select different types of fitness functions (in the mid-left of GUI) according to each specific case.

5.10 Experimental Results

In this experimental results section, the optimization framework is verified by using the following two test cases. The test cases are carefully chosen to prove the framework's ability to search quickly for optimal performances, as well as its capability for fine tuning the parameter space in wireless sensor networks. For this work, the optimization process is based on the tuning of parameter configurations of existing protocols

to achieve the best optimal behavior. In both cases, the parameter space under investigation consists of the four parameters of the unslotted CSMA/CA algorithm, which are *macMinBE, macMaxBE, macMaxCSMABackoffs* and *macMaxFrameRetries*. With the range from 3 to 8 for *macMaxBE*, 0 to *macMaxBE* for *macMinBE*, 0 to 5 for *macMaxCSMABackoffs*, and 0 to 8 for *macMaxFrameRetries*, the total number of solutions has a wide possibility of 1872 combinations in the parameter space. The first case applies a weighted sum fitness function as the trade-off for evaluating multiple performance metrics. The second case is for a simultaneous multi-objective optimization under multi-scenario conditions. All the experiments performed in this section were executed on an Intel Xeon server operating with Linux CentOS (4 CPUs, 8GB RAM, 2.4GHz).

5.10.1 Part I - Results of Weighted Sum Optimziation

In this part, the search efficiency and reliability of this GA-based framework is tested. The performance metrics in terms of energy consumption, packet loss and packet delay are from iWEEP's SystemC-based simulation which has 30 Telos nodes, 10 Hz sampling rate, 32 bytes payload in each data packet and 4 seconds of simulation time (one of the scenarios in section 4.5). The three performance metrics are modeled into a linear weighted sum cost function as the trade-off optimization method, to find the best optimal performance weighted sum value via the tuning of unslotted algorithm's parameter configuration. Finally, the GA-based optimizations are compared with the time-consuming full/exhaustive simulations (simulations on all possible solutions) in terms of efficiency and results. The weighted sum cost function is derived from (equation 5.4) and given as follows:

$$\begin{aligned} CostFcn_{(performance)} =& 1 \cdot (nodeConsumedEnergy/TotalEnergy) + \\ & 1 \cdot (PktLoss/TotalSensedPkt) + \\ & 1 \cdot (SucPktLatency/(2 * SampleInterval)) \end{aligned} \tag{5.7}$$

In this test, the same importance is given to the three metrics with each weight vector as 1. Percentage numbers are used for the performance metrics to reduce the impact of the metrics' contribution in the weighted sum. Here, *TotalEnergy* is given as 250 *mJ* (in the experiment, the maximum energy will not exceed 250 *mJ*). The use of twice the sample interval is because latency could sometimes exceed 100 *ms* (1*Sample Interval) when longer backoff wait time happens, which can be caused by the use of larger *BE* value.

The GA implemented in MATLAB R2012a has been integrated in the iMASKO framework and used for the parameter tuning in the experiment. The parameters of the unslotted algorithm are all integers rather than real numbers. Although MATLAB R2012a has a built-in integer number based optimization support, it provides limited configuration options. Namely, if integer optimization is applied, many commonly used options cannot be configured, such as the settings of *CreationFcn, SelectionFcn, CrossoverFcn, MutationFcn, EliteCount* and *CrossoverFraction* are all unavailable in integer problems according to the user guide. Therefore, the default real number optimization is used in our experiments but with custom functions like

Table 5.3: Exhaustive/Full simulation results

Fixed *macMaxBE*	**Parameter Combinations**	**Weighted Sum Range**	**Simulation Time (minutes)**
3	192	0.504349∼0.812070	34.397
4	240	0.498976∼0.982988	43.622
5	288	0.497643∼1.292583	53.144
6	336	0.507912∼1.747466	61.901
7	384	0.507912∼1.834899	69.886
8	432	0.507912∼1.935943	76.702
Total →	1872	0.497637 1.935943	339

Table 5.4: Weighted sum range area

Weighed Sum Range: 0.497637 ⟶ 1.935943									
Ratio	0.5%	1%	2%	3%	4%	5%	6%	7%	8%
Threshold	0.5048	0.5120	0.5264	0.5408	0.5522	0.5696	0.5839	0.5983	0.6127

CreationFcn and *MutationFcn* to produce integer individual parents and integer mutation children, in which the evaluation can proceed totally based on integer optimization while at the same time all the options in GA can be configured according to the problems at designers' hands.

In addition, the typical data sensing start point of each node is randomly generated (as in section 4.2, random start sensing case) by feeding with different seed values, but it could also be fixed with the same sensing start time. Since the different random sensing start time of the data can affect the network condition (even with the same combination of protocol parameters), the final performance results could be different. Hence, fixing the same data start time can avoid such unpredictable elements and helps GA to effectively search the parameter space for the best performance exploration. The fixed starting point can be controlled by employing a set of fixed seeds (four different seeds are used for four parallel simulations, four runs for the average simulation result), which ensures that under each parameter combination case every node starts to generate the data at the same instant. This approach is typically used to reproduce the given result, and here enables the comparison between optimization results and exhaustive/full simulation results to validate the efficiency of GA's searching ability under the proposed iMASKO framework.

The comparison results are shown in the following part. At first, the weighted sum results of exhaustive/full simulations are presented in Table 5.3. In the following table, since *macMinBE* must be less than *macMaxBE*, the value of *macMaxBE* is fixed as the maximum value for *macMinBE* in each row of the Table. The range of weighted sum results are given in each fixed *macMaxBE* case and the total simulation time is also provided.

The comparison between full simulation results and optimization results is done by checking the probability that GA based results fall into different parts of the weighted sum range, and Table 5.4 shows the detailed information of the weighted sum range.

The GA optimizations were launched with the settings of 2 elitecount, 0.8 crossoverfraction, and the

Table 5.5: Test results of GA-based optimization

Area	0.5%	1%	2%	3%	4%	5%	6%	7%	8%
Value	0.5048	0.5120	0.5264	0.5408	0.5522	0.5696	0.5839	0.5983	0.6127
Population size (Opt Time) ↓	The following statistics indicate the probabilities that the optimization results fall within each specific area (Average optimization time for 5,10,20,30 are shown next to the population size)								
5 (1.28min)	2%	19%	45%	74%	85%	91%	94%	97%	98%
10 (2.74min)	3%	35%	72%	94%	100%	100%	100%	100%	100%
20 (5.25min)	5%	63%	94%	100%	100%	100%	100%	100%	100%
30 (8.00min)	7%	76%	99%	100%	100%	100%	100%	100%	100%

roulette based selection process. A larger population size for individuals and more generations can certainly provide better optimization results, but at the cost of time. In this experiment, with the general efficiency and availability of the GA-based framework under the test, four population sizes with only one generation were measured (5/10/20/30 populationsize, 1 generation), since they were able to provide good results. Under each population size case, GA ran 100 times to determine the average results, because different seed values were generated according to the current time in every GA run. Hence, the randomly formed initial parent individuals, the children after crossover and mutation could be different, which led to different final results. Table 5.5 gives the efficiency information in terms of time and availability information in terms of the statistics probability of results falling into the weighted sum range area.

The results show that even with a small population size of 5, the probability of achieving cost values inside 7% of the weighted sum range area is over 95%, while the optimization time is only about 1/264 compared to the exhaustive simulation. When the population size is raised to 10, there is over a 90% probability of achieving the area, which is 3% in the weighted sum range area, while the time is about 1/123 of the exhaustive simulation. Performances are further improved when the population size reaches 20 and 30, with all the results within the 3% of the weight sum area at a small cost of optimization time, which is about 1/64 and 1/42 respectively. On the other hand, for non-expert researchers and users who do not have much knowledge in how the configuration of the unslotted algorithm can affect performances, the default configuration ([*macMinBE=3, macMaxBE=5, macMaxCSMABackoffs=4, macMaxFrameRetries=3*]) of the algorithm could always be the priority of their choice for the comparison. In this experiment, the default configuration weighted sum value is 0.737, so even with 5 population sized GA optimization all solutions can have better performance than this default configuration. Note that if another set of seed values are used, the above results could be different, but not significantly so, according to our tests. In addition, similar efficiency and availability can be achieved for MICAz, MICA2, iHop@Node based networks under the iMASKO framework.

5.10.2 Part II - Results of Multi-scenario and Multi-objective Optimization

The main goal of this test case is to verify the capability of the iMASKO framework under the conditions of multi-scenario and multi-objective optimization. The case applied in this experiment is the typical wireless body area networks (WBANs) [168], which are widely used in medical and health care applications, as

shown in Fig.5.5.

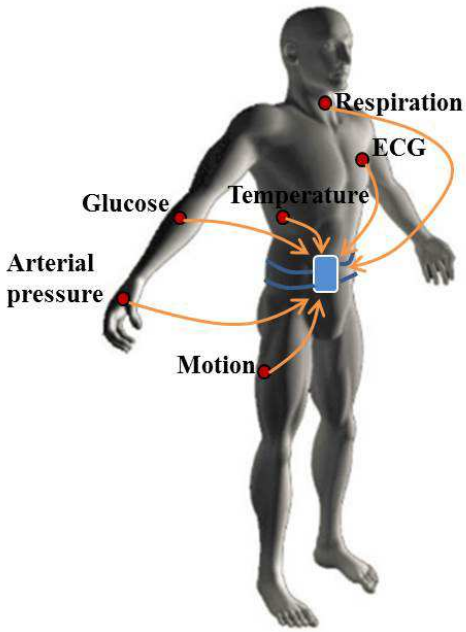

Figure 5.5: Typical WBANs for medical and health care

This typical on-body sensor network consists of six different functional sensor nodes for physiological signals monitoring and measurement, which are ECG (electrocardiogram, 125 Hz sampling rate) sensor node for measuring the rate and regularity of heartbeats, Arterial (125 Hz) and Glucose (50 Hz) nodes for pressure measurements, Motion node (100 Hz) for movement strength recording, Respiration node (25 Hz) for the test of respiration frequency and Temperature node (10 Hz) for the real-time temperature monitoring of human body. In the experiments, both Telos node and iHop@Node based networks are under investigation, the typical payload length in the packet is set to 2 bytes, and each simulation is set to 2 seconds. Fixed seed values are also adopted to mitigate the influence of the starting time of data sensing.

For such a network, each sensor node represents a specific scenario, and the use of different configurations of the unslotted CSMA/CA algorithm under this multi-scenario case could cause different performances at each sensor node. Note that in this experiment during the algorithm tuning process, the same algorithm parameter combination is used to configure all sensor nodes. As mentioned previously, performance optimization problems always involve multiple objectives to be met simultaneously in each specific scenario. These objectives are usually conflicting such as achieving maximum network reliability and at the same time minimizing the energy consumption. However, this kind of situation cannot happen most of the time, since under channel competition packet loss can be prevented if more channel access attempts, retransmission attempts or longer backoff wait time are used, and hence more energy will be consumed in these processes. Therefore, there is no single solution to a multi-objective optimization problem such as this one, in addition to the use of the Pareto front method to find a set of mathematically equal solutions (section 5.7).

In the test, two objectives (energy consumption and packet loss rate) are subjected to optimization

within the framework to achieve the solution trade-offs. In the multi-scenario situation of the network, the fitness function of each individual is composed of two pairs of energy consumption and packet loss rate for two different sensor nodes. The detailed fitness function is given as:

$$min_{(X)}\mathbf{p} = [EnergyNode1, PktLossProNode1, EnergyNode2, PktLossProNode2] \tag{5.8}$$

where X is the parameter space of the unslotted algorithm (four parameters). Node1 and Node2 are different nodes that are selected from ECG, Arterial, Motion, Glucose, Respiration and Temperature sensor nodes. The multi-objective genetic algorithm in MATLAB R2012a optimization toolbox (ver. 6.2) was integrated into the iMASKO framework and was used to generate the Pareto front for energy consumption and packet loss probability. The GA was configured for 60 initial integer individuals, 30 generations, scattered crossover, 0.8 crossover fraction as well as 2 elite children. Fig.5.6 shows the obtained four-dimensional Pareto front from the Telos mote based network.

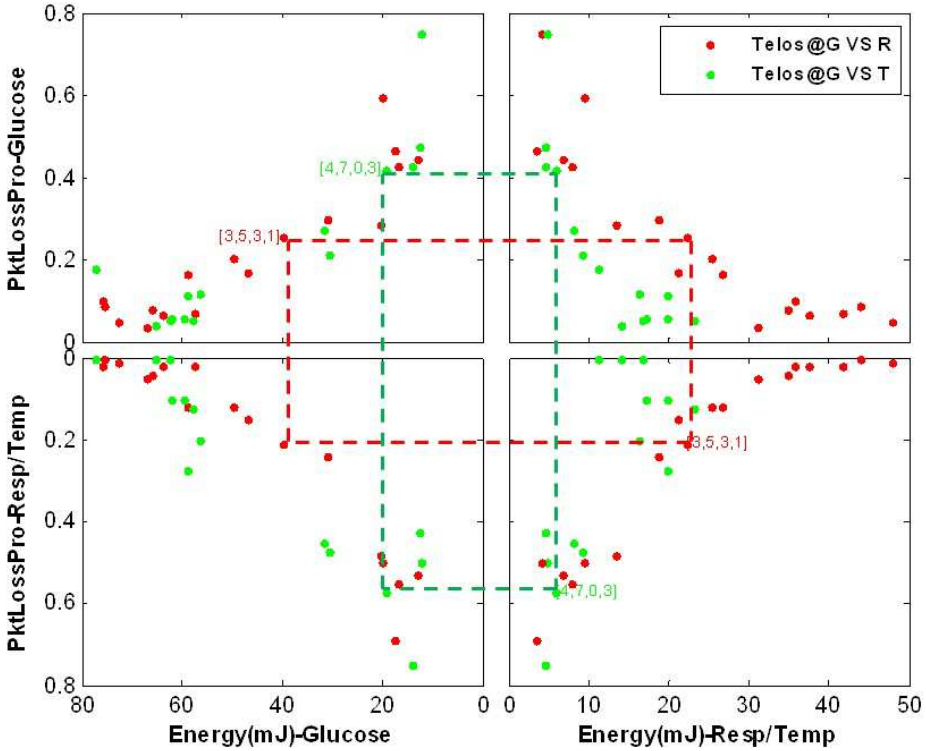

Figure 5.6: Energy and packet loss Pareto front (Glucose VS Respiration/Temperature)

In the Figure, the node energy consumption and network packet loss rate of the glucose sensor node are compared with the corresponding performance metrics of the respiration and temperature nodes respectively. According to the problem requirements, a good tradeoff solution for both energy and reliability can be manually selected by the decision maker based on their knowledge and intuitive experience (e.g., dashed rectangle in the Fig.5.6). Algorithm configurations are also presented next to the tradeoff point in the Fig.5.6 (e.g. [*macMinBE*,*macMaxBE*,*macMaxFrameBackoffs*,*macMaxFrameRetries*] = [3,5,3,1]). On the

other hand, while the glucose sensor node is in both experiments, the results show that under such multi-scenario network conditions, the best tradeoff solution for two specific cases might not guarantee the best tradeoff for another two cases. Therefore, if an overall tradeoff solution is required for the whole network then a more comprehensive optimization on all possible scenarios needs to be evaluated.

In addition, the performances (energy, reliability) of the highest sample rate based ECG node are also tested and compared with the glucose node for the iHop@Node based network. Two cases were under investigation, 1Mbps based high data rate and 250Kbps based low data rate. Fig.5.7 shows the 4D Pareto front between energy cost and packet loss for two data rate cases.

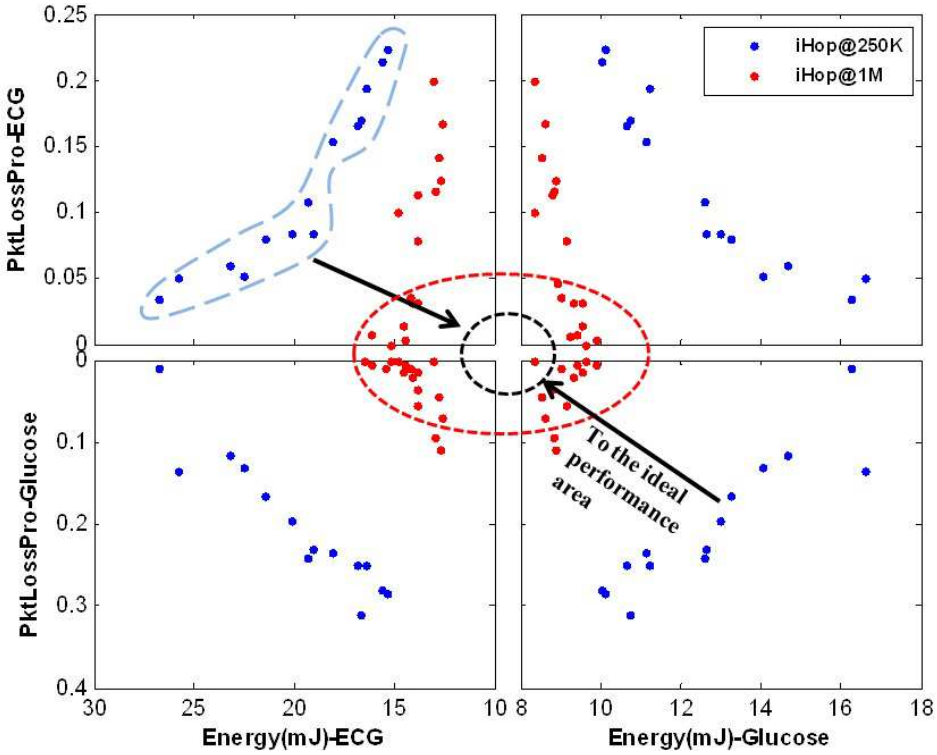

Figure 5.7: Energy and packet loss Pareto front (ECG VS Glucose @ 1Mb and 250Kb)

The results in Fig.5.7 show that the balance of tradeoffs between energy consumption and packet loss still needs to be found on the Pareto front for both data rate cases. As compared to the commonly used 250K low data rate, the high data rate based network can provide both lower energy consumption and packet loss, which brings it closer to the ideal performance area. This is because the high data rate reduces the channel occupation condition, so more packets can be successfully transmitted. For energy saving, this comes from two parts. Firstly, with good channel conditions under high data rate, mechanisms like repeated channel check and retransmission in the unslotted algorithm are no longer necessary, since almost all the packets can be successfully transmitted with one attempt, and energy is saved from protocol overhead. Secondly, despite the fact that a high data rate case consumes more current during the packet and ACK frame transmission, the time spent in both transmissions are greatly reduced due to the high data rate, and ultimately a large amount of energy can be saved from this part. (All above explanations have already been illustrated in

Chapter 4).

5.11 Conclusion

In this chapter, a generic genetic algorithm-based optimization framework iMASKO is proposed for the performance metrics optimization of wireless sensor networks. It supports the optimization of both theoretical analysis based models and simulation based models. The high efficiency and availability of the framework have been proved by modeling the required performance metrics into a weighted sum based cost function and comparing these results with exhaustive simulations. In the meantime, with the study of a typical health-care network on human body, the simultaneous optimization of four performance metrics on different node platforms and different data rates has proved that the framework supports multi-scenario and multi-objective based optimization. On the other hand, in the weighted sum based optimization, how to make each metric belong to strictly the same scale is still a challenge. Also, SystemC-based simulation could improve the whole optimization process of the framework even when larger population size and generations are employed.

Chapter 6

Conclusions and Future Works

Contents

6.1 Summary

This thesis first investigated the state of the art pertaining to energy consumption in WSNs, which mainly consists of the requirements for energy-efficient sensor networks in various application scenarios, the origins of energy consumption in a sensor network, some typical and emerging energy conservation schemes, and a survey of existing evaluation methods for energy consumption in the sensor network. A mixed energy evaluation method is adopted by combining simulation and testbed measurements, and iWEEP (**i**NL@**W**SN **E**nergy **E**valuation **P**latform) is designed and implemented based on this method for accurate evaluation of energy consumption in sensor networks. iWEEP consists of a SystemC-based simulation environment and a hardware energy measurement platform that includes a multichannel energy measurement device (MEMD), a lab-built testbed node iHop@Node (PIC16F+nRF24L01+) and an energy data management software platform (EDMSP).

Benefiting from SystemC's native support of concurrency, hierarchical modeling and synchronization, iWEEP's SystemC-based simulation environment (iWEEP_SW) enables HW/SW co-simulation and is built at the system-level and transaction-level for heterogeneous sensor network energy consumption evaluation and exploration. From the HW perspective, several COT sensor mote platforms (like Tmote Sky, MICAz and MICA2) are modeled and our lab-built high data rate and ultra-low power based testbed node iHop@Node is also simulated in the SystemC model. At the SW level, MAC layer protocols (unslotted and slotted CSMA/CA algorithms) of IEEE 802.15.4 standard are modeled, and another new MAC protocol engine Enhanced ShockBurst (ESB)/ShockBurst (SB), used as the embedded protocol in iHop@Node (embedded protocol for nRF24L transceiver series), is also implemented. An energy model is designed to automatically record hardware component register settings and calculates energy consumption by tracking every hardware component's working state and transition state that are triggered by register settings. In addition, if different output power, data rate or some other related parameters on the sensor node are configured via register settings, the energy model can index the calculation process to the right energy value according to the specific register setting. With such a flexible design strategy for energy evaluation, the heterogeneous network scenario (where each node has different configurations) and the dynamic network scenario (where configurations change during operation) can be well supported in the simulation without interrupting the

entire simulation process. This can save significant effort, especially in complex network simulation.

In the hardware energy measurement platform (iWEEP_HW), MEMD can provide detailed and accurate energy information on the sensor node, since it has multiple channels to simultaneously measure energy consumption from different components on a sensor node, and it can also be used for simultaneous multiple sensor nodes measurements. The usability and functionality of MEMD have been tested together with the iHop@Node, and its generic approach also allows it to be applied to other COT motes (e.g., Telos, MICAz, MICA2, etc). EDMSP offers a simple to use GUI to help users manage and save energy related data from the sensor node. Acquired measurements are employed not only to calibrate the energy model but also the operation mechanism of the node.

Energy consumption performances of different sensor mote based networks were investigated under several scenarios. Telos based network, MICAz based network and iHop250K (iHop@Node with 250Kbps data rate) based network were tested under varying traffic loads for energy consumption evaluation. Random sensing start time on each node can be a more energy-efficient strategy, as can payload collecting to reduce packet overhead. The use of low output power does not guarantee energy saving, which is also related to channel conditions. A comprehensive energy evaluation was given under unslotted CSMA/CA network by using a high data rate (1 Mbps and 2 Mbps data rate can be provided by iHop@Node). The principal component analysis method was used to tune each parameter of the unslotted CSMA/CA algorithm and observe these parameters' impact on the energy consumption of sensor network. The energy consumption of ESB and SB based networks were studied from several aspects. Due to the simple channel access mechanism, the ESB network, in comparison to unslotted CSMA/CA based network, was observed to have no advantages in energy consumption and network reliability, especially under a large scale of network.

On the other hand, a genetic algorithm based optimization framework, named iMASKO (**i**NL@**M**ATLAB Genetic **A**lgorithm based **S**ensor Networ**K** **O**ptimizer), is designed and implemented in this thesis. With the fast and effective search ability of genetic algorithms (GAs), iMASKO is able to find a good solution out of hundreds of possible configurations of a protocol for the given scenario. The performance metrics under the evaluation of iMASKO were generated by iWEEP's SystemC-based simulation environment. The usage of iMASKO is demonstrated through detailed case scenarios. The first case is a typical lab-scaled and unslotted CSMA/CA based network (Telos based network). By using a weighted sum cost function, iMASKO can automatically and intelligently tune the unslotted algorithm's four parameters (1872 combinations) to achieve a good tradeoff between energy consumption and other metrics like network delay and reliability in a very short time when compared to exhaustive simulation (1872 simulation runs). The second case is a human body network where iMASKO's multi-scenario and multi-objective optimization capability is tested using both Telos mote and iHop@Node. The Pareto optimization results can offer a series of available tradeoff solutions for the given scenario and it is the designer's decision to select the most suitable one according to the empirical knowledge or application-specific requirements. Besides, iMASKO is not only capable of tuning simulation or emulation based results but is also available for various mathematical and theoretical analysis models.

6.2 Future Works

Despite the fact that the research works implemented in this thesis has achieved encouraging results, they are still not ideal and there are many additional works can be done to further improve research.

- It is worth of building an operating system abstraction level in iWEEP_SW for energy consumption since many nodes are driven and controlled by an OS instead of a few lines of program code. Thus, as the basis for software development, energy performance taking the OS into account would be necessary. On the other hand, if more accurate power analysis is required, the integration of instruction set simulation provides a good choice. In addition, the modeling of a more complex channel model would be helpful for a more realistic energy evaluation.

- Apart from energy tracking and calculation functions in the battery model, more sub-modules should be added, such as an energy scavenging module which can enable the energy harvesting of many different kinds of sources (e.g., solar, light, temperature, vibration). Furthermore, the implementation of a customizable battery model could make for more efficient analyses on the energy consumption and lifetime prediction.

- For real world hardware experiments, first the MEMD and iHop@Node need to be re-designed with a smaller size. Then MEMD can be put into a package leaving only the necessary and accessible pins/interfaces in order to make it more portable. Energy measurements of MICA and Telos series need to be directly tested by MEMD and used for the calibration of the software model in iWEEP_SW. Specific functions can be added to EDMSP according users' requirements.

- For the GA based optimization framework, more optimization algorithms can be integrated and experimental results from these algorithms can be used to compare with GA results. Research work in this thesis selects a set of specific GA parameters, such as roulette wheel selection, 0.8 crossover fraction, 2 elite children, fixed random seeds, etc. So it would be interesting to explore other parameter values in future works.

- In addition to the unslotted protocol, other types of protocols are worth of being investigated. For example, it could be more challenging to evaluate the slotted protocol since its operation mechanism is more complex than unslotted protocol and it provides additional two parameters which are *SO* and *BO*.

- Since iMASKO is a framework, it does not have any protocol implementations, so the detailed modeling and running of the protocol is provided by the underlying simulation environment. In this work, we adopted the house-built iWEEP_SW as the fundamental simulation engine, which only has the implementation of unslotted/slotted MAC protocols and ESB/SB protocols. Hence, in future research work, if other types of MAC protocols or routing protocols are required for iMASKO optimization, there are two ways: (i) model detailed protocol implementation by designers themselves, and (ii) replace iWEEP_SW with other simulators or emulators. For instance, the IEEE 802.11 protocol is available for optimization if NS-2 is used as the simulation engine in iMASKO. Besides, various mathematical based theoretical and analytical models can also be used for evaluation in the

future. Finally, note that if the simulation based approach is used in the framework, the increase of simulation speed can directly accelerate optimization process.

Chapter 7

Résumé Français

Contents

7.1 Chapitre 1

Avec les grands développements dans les systèmes embarqués ainsi que dans les technologies de communication sans fil, les réseaux de capteurs sans fil (WSNs) ont attiré l'attention du monde entier et ils ont été rapidement développés dans la dernière décennie. Les réseaux de capteurs sans fil sont des réseaux à grande échelle avec des nœuds de capteurs sous faible coût, petite taille, faible puissance et traitement limitée qui sont déployés dans les différents types d'environnements. Un typique nœud de capteur qui est une partie de base de WSNs inclut plusieurs éléments comme microcontrôleur, émetteur-récepteur, capteur et source de courant (Fig. 2.2). En combinant ces différents éléments dans un dispositif miniaturisé, les nœuds de capteurs sont multi-fonctionnelles et ont été appliquées à une grande variété d'application : la médicineet la santé, l'environnement et la surveillance écologique, la maison et l'automatisation des bâtiments, le contrôle industriel et le champ de bataille. Ainsi, tels scénarios divers imposent des exigences spécifiques à l'application sur la conception de WSN et aux distinguents des réseaux conventionnels. Les avantages de WSNs sont nombreux : il est facile de réaliser une grande évolutivité et une large gamme de densités, grâce à sa petite taille et son faible coût de nœuds de capteurs. Généralement, les protocoles des réseaux de capteurs et des algorithmes spécifiques avec des capacités d'auto-organisation sont conçus et appliqués aux nœuds de capteurs, donc l'ensemble du réseau de capteurs est rarement maintenu et tolérant contre les pannes de communication et les changements de topologie qui pourraient être causés par le dysfonctionnement du nœud, la mobilité d'un nœud ou l'épuisement de l'énergie des nœuds. De plus, les nœuds de capteurs peuvent être déployés dans des environnements durs pour effectuer une tâche donnée sans aucune intervention humaine.

Toutefois, les nouvelles et potentielles applications sont limitées en raison d'inconvénients inhérents de WSN. Par exemple, la capacité de traitement limitée et du faible débit de données de nœuds de capteurs ne peut que garantir de hautes performances dans certains scénarios, particulièrement pas pour les applications en temps réel. La portée de communication courte peut causer le gaspillage d'énergie et l'inefficacité du

réseau, car les communications multi-sauts sont toujours nécessaires pour le transport de données entre le nœud source et nœud récepteur. Les contraintes de l'énergie sévère conduisent également à une observation pire : l'amélioration de la capacité de traitement des données en utilisant des processeurs puissants n'est pas attendue, puisque l'énergie sera épuisée très rapidement et elle rend le réseau inutile pour la tâche donnée. L'énergie limitée signifie aussi qu'il n'est pas possible de maintenir le fonctionnement des communications de multi-hop pendant une longue période. Pendant cette période, les nœuds de capteurs devraient être largement utilisés dans de nombreuses applications de région éloignée où le changement fréquent ou la recharge des batteries est gênant. Par conséquent, un réseau de capteurs d'énergie efficace est nécessaire pour effectuer le travail dans des telles conditions.

Du coup, en prenant la consommation d'énergie comme la considération principale, sans miner d'autres performances, est souhaitée que la stratégie prioritaire dans la conception des réseaux de capteurs sans fil actuels.

Le domaine émergent des réseaux de capteurs sans fil et leurs applications promet une meilleure qualité de vie dans de nombreux aspects des activités quotidiennes de l'homme, mais les contraintes de l'énergie stricte indiquées ci-dessus limitent considérablement leurs fonctionnalités et les applications possibles, ce qui rend la consommation d'énergie l'un des problèmes les plus critiques dans les réseaux de capteurs. Pour cette raison, l'évaluation de la consommation d'énergie et d'analyse est un processus essentiel dans la conception et la mise en œuvre du réseau de capteurs d'énergie efficace.

Malgré des résultats précis, l'analyse de la consommation d'énergie de déploiement réel peut souffrir des limites, parce que les mesures de nœuds de capteurs à grande échelle sont difficiles à acquérir, et ces nœuds sont parfois placés dans des zones difficiles et inaccessibles. D'autre part, les modèles analytiques/théoriques basés sur les mathématiques offrent une plus grande capacité d'exploration, mais ils peuvent être simplifiés par des hypothèses idéalisées, et conduire à des résultats erronés pour des problèmes pratiques. Comme une solution de compromis, l'approche basée sur la simulation est largement acceptée et couramment utilisée dans l'analyse des réseaux de capteurs, car elle offre un bon équilibre entre l'efficacité et la précision. Les outils de simulation générales, écrites en C++, Java ou MATLAB, offrent une efficacité satisfaisante dans les premières étapes de conception, puisque les modèles sont généralement à un niveau d'abstraction élevée pour faciliter l'essai et la vérification des algorithmes, protocoles et stratégies. Cependant, une seule évaluation de la consommation d'énergie grossière peut donc s'attendre en raison de l'absence de modèles réalistes de bas niveau. Les outils d'émulation peuvent compenser cet inconvénient sur l'évaluation de la consommation d'énergie, mais au prix de l'efficacité, et ils sont essentiellement limités aux plateformes matérielles spécifiques et aux systèmes d'exploitation. SystemC offrant un soutien pour HW / SW co-simulation, la concurrence et la modélisation à différents niveaux d'abstraction ainsi que d'autres avantages de la modélisation souples et diversifiées, est considéré comme une bonne alternative pour modéliser et simuler des réseaux de capteurs pour l'évaluation de l'énergie précisée, souple et rapide. Pendant ce temps, les mesures de nœuds de banc d'essai dans le monde réel sont également considérés importants, car ils offrent un environnement plus réaliste pour le déploiement réel et sont capables de calibrer et améliorer la précision de l'évaluation de l'énergie. Par ailleurs, une enquête approfondie de la performance de l'énergie doit être faite avec des différentes plateformes de nœuds de capteurs et sous différents scénarios.

Puisque la communication sans fil est perçue comme le plus faim d'énergie dans le nœud de capteur,

un grand nombre de protocoles de communication ont été conçus à l'année dernière pour gérer directement et efficacement la communication sans fil dans le réseau de capteur avec l'objectif spécifique d'essayer de maximiser les économies d'énergie. Avec la diversité des scénarios d'application, les exigences de conception sont habituellement application spécifique qui pourrait inclure de nombreux autres types de mesures, en plus de la performance énergétique. Par conséquent, il est clair que une seul protocole commun ou default ne peut pas convenir à toutes les applications et le choix des réglages avec les paramètres appropriés pour le vrai déploiement est essentiel pour équilibrer la consommation d'énergie avec d'autres mesures concurrentes. Malgré l'efficacité fournie par simulation, les simulations exhaustives et complètes sur toutes les configurations possibles du protocole consomment non seulement beaucoup de temps, mais également inutile la plupart du temps. Donc, une approche qui peut effectivement et automatiquement sélectionner un ensemble de paramètres de protocole approprié sur des centaines ou des milliers d'options possibles est urgente. La capacité de recherche rapide et l'intelligente des algorithmes génétiques (GA) permet le réglage du paramètre espace du protocole pour trouver la configuration la plus adaptée.

La première contribution est de concevoir et de mettre en œuvre iWEEP (**i**NL@**W**SN **E**nergy **E**valuation **P**latform) dans une méthode mixte d'évaluation de l'énergie qui combine la simulation et les mesures de nœuds de bancs d'essai. iWEEP est utilisé pour l'évaluation de la consommation d'énergie souple et précisée sur le nœud de capteur et un réseau de capteurs. Il se compose de deux parties :

i. un environnement de simulation basé sur SystemC au niveau du système et au niveau des transactions qui mettent en œuvre plusieurs COTS (commercial off-the-shelf) modèles de nœuds de capteurs avec faible débit de données y compris la série Telos (par exemple, TelosB, Tmote Sky, Shimmer), la série de MICA (par exemple, MICA2, MICAz), et un nœud de banc d'essai sous laboratoire construit basé sur le débit de données élevé et d'ultra-faible puissance, nommé iHop@Node (PIC16F + nRF24L01p, débit de données maximal jusqu'à 2 Mbps). Quatre protocoles MAC sont intégrés et modélisés dans SystemC l'environnement de simulation de iWEEP (iWEEP_SW) qui sont protocoles des unslotted and slotted CSMA/CA dans la norme IEEE 802.15.4, protocoles des Enhanced ShockBurst (ESB) et ShockBurst (SB) intégrés dans le haut débit de données basées iHop@Node.

ii. la deuxième partie de iWEEP est la plateforme de mesure d'énergie basé sur le matériel (iWEEP_HW) qui est constitué MEMD (Multi-channel Energy Measurement Device) et EDMSP (Energy Data Management Software Platform) pour pouvoir mesurer les données relatives à l'énergie de iHop@Node, MICA mote, Telos mote et autres motes de capteurs similaires. Pour MEMD, il peut fournir des mesures simultanées sur les différents éléments d'un nœud de capteur ou des mesures simultanées sur plusieurs nœuds différents en utilisant son avantage de canaux multiples. Pour EDMSP, il offre un GUI facile à utiliser pour aider les utilisateurs à gérer et enregistrer les données de l'énergie à partir du nœud de capteur.

Mesures du monde réel sont utilisés à la fois pour le modèle d'énergie et des calibrations du mécanisme de fonctionnement sur le nœud de capteur. Les performances de la consommation d'énergie sur les réseaux à base de Telos, MICA et iHop@Node sont évalués et simulés dans différents scénarios : variations du volume de trafic, détection aléatoire, différent tête de paquet et des différentes puissances de sortie dans deux modèles de canaux. Performances de consommation d'énergie de réseau basé unslotted CSMA / CA sont mis en évidence et évalués dans les scénarios de haut débit de données (2 Mbps et 1 Mbps fournies par iHop@Node) Les comparaisons sont également faites entre ce scénario de haut débit de données et le

scénario de faible débit de données couramment adopté (par exemple, 250 Kbps pour Telos et MICAz, 38.4 Kbps pour MICA2). De plus, la consommation d'énergie des réseaux basé sur ESB et SB sont étudiés en fonction de plusieurs aspects et leurs performances énergétiques sont également comparées avec le réseau à base d'unslotted protocole.

La deuxième contribution réside dans l'introduction et la conception d'un cadre d'optimisation en ligne basé sur l'algorithme génétique pour les réseaux de capteurs, nommé iMASKO (**i**NL@**M**ATLAB Genetic **A**lgorithm based **S**ensor Networ**K** **O**ptimizer). iMASKO utilise MATLAB pour mettre en œuvre son moteur d'évaluation de l'optimisation et il intègre iWEEP_SW comme le moteur de simulation pour générer des indicateurs de performance. Comme un cadre d'optimisation générique, le moteur de simulation peut être remplacé par d'autres outils de simulation (par exemple, NS-2, OMNeT++) ou par des modèles mathématiques d'analyse. L'utilisation de iMASKO peut aider les utilisateurs à trouver les paramètres optimaux pour les protocoles sur des centaines de solutions possibles dans un temps très court sans simulations exhaustives sur tous les scénarios possibles. En utilisant une fonction de coût de la somme pondérée, iMASKO est capable d'évaluer la consommation d'énergie, sans porter problème à d'autres mesures pour parvenir à une solution de compromis. Quand un algorithme génétique de l'objectif multiple est utilisée, les résultats basés Pareto générés par iMASKO peuvent être appliquées à des problèmes aù multi-scénarios et multi-objectifs conditions sont requis pour être pris en considération.

Le reste de la thèse est organisé comme suit :

Le chapitre 2 présente un état de l'art des réseaux de capteurs sans fil en ce qui concerne l'architecture, des scénarios d'application, les origines de la consommation d'énergie, les stratégies de réduction de la consommation d'énergie, les économies d'énergie potentiel offert par le système d'exploitation ainsi que des outils pour la consommation d'énergie des réseaux de capteurs sans fil.

Le chapitre 3 décrit la conception de iWEEP_SW et iWEEP_HW. Détaillées implémentations de modélisation, les flux de conception et de l'architecture de cadre de iWEEP_SW ainsi que la construction de prototype et des résultats de mesure dans iWEEP_HW sont expliqués en détail.

Le chapitre 4 examine la performance de la consommation d'énergie de différents nœuds de capteurs dans divers scénarios. Résultats énergétiques pertinents sont présentés et comparés. Stratégies de conservation de l'énergie sont donnés après chaque scénario.

Le chapitre 5 présente la conception et la mise en œuvre d'un cadre nommé iMASKO. Avec la capacité de recherche rapide et intelligente, iMASKO intègre iWEEP_SW pour les optimisations. Deux études de cas sont utilisées pour valider l'utilisabilité et l'efficacité de iMASKO.

Le chapitre 6 conclut la thèse et discute des travaux futurs possibles.

7.2 Chapitre 2

Les réseaux de capteurs sans fil (WSN) sont constitués par la grande échelle de nœuds de capteurs distribués pour surveiller et mesurer les conditions physiques et environnementales, tels que l'humidité, la température, mouvement, lumière, etc. Les données mesurées sont envoyées aux différents terminaux ou aux utilisateurs finaux via le réseau sans fil. Par rapport au réseau conventionnel, les réseaux de capteurs sont de petite taille, de faible consommation d'énergie et ils ont déjà été largement déployée dans de nombreuses applications,

mais WSNs face à des défis de ressources limitées dans le stockage, de la gamme de communication, de la capacité de traitement et de l'énergie. Les utilisations potentielles de WSNs ont été limités notamment en raison du manque d'énergie. Puisque les nœuds de capteurs dans de nombreux scénarios d'application sont requis à fonctionner plusieurs mois ou années sans intervention humaine (par exemple, changement de batterie et la recharge), l'augmentation des méthodes de conservation de l'énergie sont proposées visant à réaliser un réseau efficace de l'énergie. Différents outils d'évaluation (simulateurs, émulateurs et testbeds) sont disponibles pour faciliter et accélérer l'évaluation de la consommation d'énergie de WSNs avant les vrais déploiements.

Dans la section 2.1, l'architecture de WSNs est décrite brièvement dans deux aspects : du point de vue de la plateforme matérielle, les fonctions de différents composants sur le nœud de capteur sont présentées et plusieurs prototypes de nœuds de capteurs populaires sont donnés. D'autre part, la pile de protocole de WSN avec cinq-couches est également décrite comme la stratégie de conception de logiciel.

Dans la section 2.2, afin d'aider à comprendre les exigences et les caractéristiques spécifiques de WSNs dans le processus de conception, un aperçu des cinq types d'applications WSN est présenté qui comprennent médicale et les soins de santé, la surveillance de l'environnement et écologique, l'automatisation de la maison et du bâtiment, les applications industrielles et les applications militaires.

Dans la section 2.3, plusieurs défis de conception de WSNs sont décrits, y compris la qualité de service (QoS), sécurité et confidentialité, la limitation des ressources, l'adaptabilité et l'énergie. La consommation d'énergie est souligné et mis en évidence dans cette thèse, et un bon critère de conception est de réduire la consommation d'énergie sans compromettre d'autres métriques. La solution de compromis est toujours requise.

Section 2.4 analyse la consommation d'énergie du réseau de capteurs. Information générale de la consommation d'énergie est donnée pour les principales composantes du nœud capteur où microcontrôleur, radio, mémoire et capteur sont inclus. A partir du niveau de réseau, le gaspillage d'énergie dans collision, l'écoute ralenti, entendant, overmitting et les frais généraux de paquets de protocole sont mentionnées.

Section 2.5 est une enquête sur certaines méthodes typiques et émergentes de conservation de l'énergie. En utilisant l'architecture de nœud hiérarchique pour différentes tâches, la source d'énergie peut être effectivement affectée à l'événement spécifique. Technique de récupération d'énergie est utilisée à la conversion de sources d'environnement à énergie électrique. L'énergie peut aussi être conservée en réglant les temps de transmission et des temps d'acquisition des données détectées, donc trois types de stratégies d'échantillonnage sont mentionnés pour la réduction de l'énergie. In-network traitement que l'inclusion de l'agrégation de données, la compression des données et des réseaux mobiles est considérée comme la façon traditionnelle de conversion d'énergie. La conception du protocole MAC est mise en évidence comme le mécanisme l'efficacité énergétique et la plus importante dans la conception de WSNs, car il gère directement la communication sans fil, qui est la partie la plus consommatrice d'énergie.

Section 2.6 discute sur les caractéristiques et les performances de plusieurs WSN systèmes d'exploitation, car les possibilités d'économie d'énergie pourraient être réalisées via les optimisations de ces systèmes d'exploitation. Les systèmes d'exploitation qui sous enquête sont TinyOS, PicOS, Contiki, MANTIS, LiteOS and Nano-RK.

Dans la section 2.7, une enquête détaillée et élaborée est donnée sur les outils d'évaluation de WSN

existantes au sujet de leurs capacités dans l'évaluation de la consommation d'énergie. De nombreux outils de simulation, des outils d'émulation et d'outils de testbed sont étudiés à la fois des avantages et des désavantages. La méthode de simulation émergents basée sur SystemC peut fournir une évaluation de la consommation d'énergie souple et précise pour les réseaux de capteurs de différents niveaux d'abstraction, en raison de ses soutiens naturels de HW / SW co-simulation et la conception basé sur les composants. Pour obtenir des résultats plus réalistes de consommation d'énergie, une méthode d'évaluation de l'énergie mixte combinant les mesures de nœud de testbed et de simulation de SystemC est soulignée et mise en évidence dans cette thèse.

7.3 Chapitre 3

Dans ce chapitre, une plateforme d'évaluation de l'énergie du réseau de capteur complète, nommée iWEEP (**i**NL @ **W**SNs **E**nergy **E**vaulation **P**latform), est présentée. iWEEP se compose de iWEEP_SW dans le logiciel et iWEEP_HW dans le matériel. iWEEP_SW est un environnement de simulation réseau de capteurs d'énergie conscients qui est construit sur SystemC au niveau du système et des transactions. Il se concentre sur la conception modulaire des composants matériels distincts sur nœuds de capteurs à différents niveaux d'abstraction. iWEEP_SW basé sur SystemC peut précisément évaluer la performance de la consommation d'énergie des objectif réseaux de capteurs / nœuds conformément aux exigences des applications, protocoles réseau mis en place et des stratégies de communication à un stade précoce de la conception sur différents niveaux d'abstraction, car SystemC est un niveau du système et langage de description matérielle pour SW et HW co-conception. iWEEP_HW est un système de mesure d'énergie dans le monde réel constitué d'une plateforme **M**ultichannel **E**nergy **E**vaulation **D**evice (MEMD) et une plateforme **E**nergy **D**ata **M**anagement **S**oftware **P**latform (EDMSP). La combinaison de MEMD et EDMSP peut fournir précise et synchrone surveillance de la consommation d'énergie pour les nœuds de capteur de courant largement utilisés pour calibrer leur modèle d'énergie et le modèle d'opération pour l'estimation de la performance énergétique réaliste. Dans cette thèse, la consommation d'énergie de notre laboratoire construit testbed nœud iHop@Node (**i**NL@**H**igh data rate and ultra l**o**w **p**ower testbed **Node**) basé débit de données élevé et ultra faible puissance est mesurée par iWEEP_HW. Pour iWEEP_HW plateforme, il peut être considéré comme une alternative flexible et rentable pour la plupart des équipements et des outils coûteux dans la recherche de la mesure de l'énergie de nœud de capteur courant.

Section 3.1 est une brève introduction sur SystemC. SystemC est un nouveau langage de modélisation basé sur le standard C++ et spécifié pour le niveau du système de modélisation, la conception et la vérification. Le né de SystemC favorise et optimise beaucoup la co-conception entre matériel et logiciel. Dans cette section, les caractéristiques de SystemC, la structure de la bibliothèque de SystemC et le soutien de SystemC pour la modélisation de niveau de transaction sont présentés.

Dans la section 3.2, la motivation de la conception des iWEEP et la vue d'ensemble de l'architecture de iWEEP sont présentés.

La section 3.3 décrit la conception de iWEEP_SW en détail. iWEEP_SW est composé de composants, les ports, les canaux, les interfaces et les connexions fournies par SystemC. En iWEEP_SW, différents modèles tels que microcontrôleur, émetteur-récepteur, capteur, timer, réseau et de l'énergie sont modélisés

comme module individuel de SystemC. Les communications entre ces différents modules sont simulées par des canaux, des ports et les interfaces fournis par SystemC. Pour simplifier l'utilisation de iWEEP_SW pour les utilisateurs non-experts, une interface graphique basée sur MATLAB est conçue pour faciliter la configuration de simulation et d'évaluation des résultats. La version actuelle de iWEEP_SW prend en charge de diverses platesformes de nœuds de capteurs : MICA série (MICAz, MICA2), Telos série (Telos, TelosB, Tmote Sky, Shimmer node) and iHop@Node. Protocoles pris en charge par iWEEP_SW comprennent unslotted et slotted CSMA/CA couche de protocole MAC de la norme IEEE 802.15.4, ainsi que Enhanced ShockBurst (ESB) et ShockBurst (SB) moteur de protocole intégré dans le iHop@Node.

Dans la section 3.4, la conception et la mise en œuvre de iWEEP_HW sont présentés. iWEEP_HW fournit une solution, qui est rentable, effet secondaire libre et flexible, pour accéder aux informations fiables et détaillées sur la consommation d'énergie sans utiliser de coûteux haute fréquence multimètres et oscilloscopes. iWEEP_HW est un système de mesure d'énergie construit par la laboratoire qui est constitué d'une plateforme matérielle de mesure appelée MEMD (**M**ultichannel **E**nergy **M**easurement **D**evice) et une plateforme de gestion de données d'énergie appelé EDMSP (**E**nergy **D**ata **M**anagement **S**oftware **P**latform). iWEEP_HW peut fournir des mesures réalistes et précis pour les nœuds de capteurs destinés, et il peut également calibrer et valider les modèles d'énergie et les modèles de fonctionnement de iWEEP_SW pour parvenir à une évaluation de la performance énergétique beaucoup plus fiable et réaliste. Dans cette section, iHop@Node est utilisé pour les expériences de mesure.

7.4 Chapitre 4

Dans ce chapitre, la méthode de simulation est utilisée pour l'évaluation de l'énergie. La consommation d'énergie détaillée de quatre types de réseaux de nœuds de capteurs (Telos, MICAz, MICA2 et iHop@Node) est étudiée sous différents scénarios via des simulations iWEEP. Dans chaque cas, des informations détaillées sur la consommation d'énergie et de données de comparaison sont présentées et analysées pour chaque plateforme de nœud de capteur correspondant. Après l'évaluation complète de l'énergie dans chaque scénario, des suggestions sont données aux développeurs et chercheurs pour les aider à concevoir un réseau de capteurs sans fil conscient de l'énergie.

La section 4.1 discute des impacts de la charge de trafic sur la consommation d'énergie du réseau basé sur Telos nœud, MICAz nœud et iHop250K (iHop@Node avec 250Kbps débit de données). Dans une étude de cas de la santé / de la surveillance typique ($3 \sim 10$ nœuds), iHop@Node a été prouvé être le plus économe en énergie sur l'ensemble des charges de trafic, en raison des périphériques de faible consommation d'énergie et des frais généraux de contrôle de paquet court (moins d'énergie consommée sur les frais généraux de paquets). Malgré l'utilisation de CC2420 à la fois sur Telos et MICAz, une meilleure performance de l'énergie peut être prévue sur Telos, car un ultra-faible puissant microcontrôleur TI MSP430 est employé sur Telos. D'autre part, dans les mêmes conditions du charge de trafic, il est montré que le cas avec plus des nœuds et fréquence d'échantillonnage plus faible est plus économe en énergie que le cas avec moins des nœuds et fréquence d'échantillonnage plus élevée.

La section 4.2 examine l'impact de la stratégie d'échantillonnage sur la consommation d'énergie des réseaux de capteurs. L'échantillonnage avec le même temps de départ des premières données de détection

et l'échantillonnage du temps aléatoire des premières données de détection sont étudiés dans cette section. Les résultats montrent la stratégie d'échantillonnage de temps aléatoire est plus économe en énergie. En employant le même scénario de réseau de la section 4.1, le taux d'épargne maximum et minimum de la stratégie aléatoire sur Telos sont respectivement 73.8% et 0.16%, pour MICAz le taux est 73.4% et 0.01%, et le taux de iHop250K sont 40.4% et 0.15%.

Section 4.3 explore la perte d'énergie provoquée par les frais généraux de paquets contrôlés (d'autres parties dans le paquet de données sauf la charge utile). Les résultats expérimentaux montrent que l'économie d'énergie des frais généraux de paquets contrôlés a un impact significatif sur la consommation d'énergie totale du nœud de capteur. Si la latence du réseau n'est pas une considération stricte, la stratégie de collecter la charge utile peut être une approche efficace pour réduire considérablement la consommation d'énergie et la perte de paquets du réseau de capteurs.

Dans la section 4.4, les différentes puissances de sortie du nœud sont sous l'évaluation de la consommation d'énergie des réseaux de capteurs. Dans le processus d'évaluation, un scénario de réseau du corps réel d'humain est étudié. Expériences de simulation de la consommation d'énergie sont effectuées dans le modèle du canal de l'espace libre et le modèle réaliste du canal sur le corps de la ligne du regard. Les résultats des comparaisons montrent que la faible puissance de sortie ne peut pas garantir l'exigence d'économie d'énergie et pour le but de l'efficacité énergétique, le choix de la puissance de sortie serait spécifique a l'application.

La section 4.5 évalue les impacts des taux de données et configuration de l'algorithme de unslotted C-SMA / CA sur la consommation d'énergie des réseaux de capteurs. L'utilisation des réseaux basés à haut débit de données (iHop2M / iHop1M) peut offrir beaucoup mieux performances d'énergie et la fiabilité du réseau. Méthode basée sur l'analyse des composantes principales est adoptée pour évaluer les impacts des paramètres de l'algorithme de unslotted CSMA / CA (*macMinBE = macMaxBE = BE, macMaxCSMABackoffs = Backoff, macMaxFrameRetries = Retries*) sur la consommation d'énergie des réseaux de capteurs. Pour atteindre l'efficacité énergétique, la sélection des appropriée *BE, Backoff* et *Retries* sont spécifique à l'application.

Section 4.6 présente la consommation d'énergie détaillée des réseaux basés sur Enhanced ShockBurst (ESB) et ShockBurst (SB). Ensuite, le réseau basé sur ESB est comparé au réseau basé sur unslotted CSMA/CA dans trois débits de données et les variations de la charge de trafic. Les résultats montrent que ESB a non seulement un grand potentiel d'avoir grande perte de paquets, mais aussi impuissant dans la consommation d'énergie en particulier dans le scénario où grand nombre de nœuds de capteurs sont déployés.

7.5 Chapitre 5

Dans ce chapitre, la conception et la mise en œuvre d'une optimisation générique cadre basée sur GA (algorithme génétique) appelé iMASKO (**i**NL@**M**ATLAB Genetic **A**lgorithm-based **S**ensor Networ**K** **O**ptimizer) sont minutieusement décrits. En raison de la propriété de recherche globale des algorithmes génétiques, le cadre permet de syntoniser automatiquement des centaines de solutions possibles pour trouver les meilleures solutions de compromis appropriés (dans ce travail, iMASKO est utilisé pour syntoniser et évaluer la simulation de iWEEP_SW). L'efficacité de l'optimisation et de disponibilité des iMASKO ont été vérifiées par l'expérience avec une fonction de coût basée sur la somme pondérée. En outre, iMASKO montre aus-

si sa bonne capacité de soutenir l'optimisation basé sur multi-scénarios et multi-objectif dans une autre expérience.

Section 5.1 décrit la nécessité de faire l'optimisation au réseau de capteurs sans fil. Les méthodes d'optimisation sont présentées à partir respectivement du matériel et des logiciels perspectives. L'utilisation d'algorithmes génétiques (GAs) de la méthode dans l'optimisation des réseaux de capteurs sans fil est proposée, car GAs peut fournir une capacité de recherche rapide et complète.

Section 5.2 introduit GAs et donne une brève comparaison entre GAs et les autres méthodes conventionnelles. Certains travaux qui utilisent GAs dans l'optimisation de WSNs sont également décrits dans cette section.

Dans la section 5.3, la motivation de proposer iMASKO cadre optimisation est présentée, qui vise à sélectionner automatiquement et intelligemment des solutions adaptées sur des centaines ou des milliersposs-ibilités dans l'espace de conception du problème, au même titre que l'évitement des simulations inutiles pour garantir l'efficacité de cette méthode d'optimisation.

Section 5.4 montre l'architecture de iMASKO cadre optimisation et répertorie le processus d'optimisation en détail.

Dans la section 5.5, trois types d'importants métriques de performance couramment utilisés sont donnés, qui comprennent métriques liés à l'énergie, métriques liées à la fiabilité de réseau et métriques liées au retard de réseau. Ces métriques de performance sont utilisées dans l'évaluation de l'optimisation d'iMASKO.

Section 5.6 présente une méthode sur la fonction de coût de la somme pondérée pour WSN optimisation, qui vise à trouver une solution de compromis pour l'équilibre entre les différentes métriques de performance.

Section 5.7 présente la méthodologie de faire l'optimisation dans les conditions basée sur complexe multi-scénario et multi-objectif où l'optimisation de Pareto-front fournit les critères de compromis pour les métriques de performance dans différents cas.

Section 5.8 et 5.9 décrivent les options de iMASKO, fournit la façon de l'utiliser par commande et interface graphique, et l'utilisation générique de iMASKO dans le processus d'optimisation / d'évaluation est également soulignée.

Section 5.10 est la section de résultat expérimental. En utilisant iMASKO et la méthode de la somme pondérée, dans les expériences iMASKO peut atteindre des valeurs de la somme pondérée très faibles dans un temps extrêmement court (détails dans la section 5.10.1), qui prend seulement 1/42 du temps de la simulation complète. Pour le scénario avec les caractéristiques de complexe multi-scénario et multi-objectif, iMASKO montre son excellente adaptation et les limites basées sur Pareto sont trouvés après l'optimisation pour les réseaux basés sur Telos et iHop@Node dans un scénario de réseau de corps humain sans fil.

7.6 Chapitre 6

Cette thèse d'abord étudie l'état de l'art concernant la consommation d'énergie dans les réseaux de capteurs, qui se composent principalement des exigences des réseaux de capteurs d'efficacité énergétique dans divers scénarios d'application, des origines de la consommation d'énergie dans un réseau de capteurs, des certains régimes typiques et émergents d'économie d'énergie, et une enquête sur les méthodes d'évaluation existantes pour la consommation d'énergie dans le réseau de capteurs. Une méthode d'évaluation de l'énergie

mixte est adoptée en combinant simulation et les mesures de banc d'essai, et iWEEP (**i**NL@**W**SN **E**nergy **E**valuation **P**latform) est conçu et mis en œuvre basé sur cette méthode pour l'évaluation précise de la consommation d'énergie dans les réseaux de capteurs. iWEEP se compose d'un environnement de simulation basée sur SystemC et une plateforme matérielle pour mesurer de l'énergie qui comprend à un dispositif MEMD (**M**ultichannel **E**nergy **M**easurement **D**evice), à un laboratoire construit nœud de banc d'essai iHop@Node (PIC16F+nRF24L01p) et à une plateforme logicielle EDMSP (**E**nergy **D**ata **M**anagement **S**oftware **P**latform).

Bénéficiant d'un support natif de SystemC de la concurrence, la modélisation hiérarchique et la synchronisation, l'environnement de simulation (iWEEP_SW) de iWEEP basée sur SystemC permet HW / SW de co-simulation et est construit au niveau du système et au niveau des transactions pour l'évaluation et à l'exploration de la consommation d'énergie des réseaux de capteurs hétérog¨¨nes. Du point de vue HW, plusieurs COT platesformes de nœud de capteur (comme Tmote Sky, MICAz et MICA2) sont modélisés et notre laboratoire construit nœud de banc d'essai iHop@Node basé sur débit de données élevé et d'ultra-faible puissance est également simulé dans le modèle SystemC. Au niveau de la SW, protocoles de couche MAC (algorithmes des unslotted et slotted CSMA / CA) de la norme IEEE 802.15.4 sont modélisés, et un nouveau moteur de protocole MAC ¨C Enhanced ShockBurst (ESB) / ShockBurst (SB), utilisant comme protocole intégré dans iHop@Node (protocole intégré pour nRF24L série émetteur-récepteur), est également mis en œuvre. Un modèle d'énergie est conçu pour enregistrer automatiquement les paramètres de registre de composants matériels et calculer la consommation d'énergie par le suivi de l'état fonctionnement et de l'état de transition qui sont déclenchés par les paramètres de registre. En outre, si la puissance de sortie différent, débit de données ou d'autres paramètres associés sur le nœud de capteur sont configurés via les paramètres de registre, le modèle énergétique peut indexer le processus de calcul avec valeur de l'énergie correspondante en fonction du réglage spécifique de registre. Avec une telle stratégie de conception flexible pour l'évaluation de l'énergie, le scénario de réseau hétérogène (où chaque nœud a différentes configurations) et le scénario de réseau dynamique (où les configurations changent pendant le fonctionnement) peuvent être bien pris en charge dans la simulation sans interrompre le processus de simulation. Cela peut sauver des efforts significatifs, en particulier dans la simulation de réseau complexe.

Dans la plateforme matériel de mesure d'énergie (iWEEP_HW), MEMD peut fournir de l'information détaillée et précise de l'énergie sur le nœud de capteur, puisqu'il a plusieurs canaux pour mesurer simultanément la consommation d'énergie à partir de différents composants sur un nœud de capteur, et il peut également être utilisé pour simultanés multiples nœuds de capteurs de mesures. La convivialité et la fonctionnalité de MEMD ont été testées avec le iHop@Node, et son approche générique permet également d'être appliquée à d'autres COT motes (par exemple, Telos, MICAz, MICA2, etc.). EDMSP offre une GUI simple pour aider les utilisateurs à gérer et sauvegarder les données liées à l'énergie à partir du nœud de capteur. Mesures acquises sont utilisées non seulement pour calibrer le modèle d'énergie mais aussi le mécanisme de fonctionnement du nœud.

Les performances de consommation d'énergie de réseaux à base de différents mote capteur ont été étudiées dans les plusieurs scénarios. Le réseau basé sur Telos, le réseau basé sur MICAz et le réseau basé sur iHop250K (iHop@Node avec débit de données 250Kbps) ont été testés dans les différentes charges de trafic pour l'évaluation de la consommation d'énergie. La détection de démarrage aléatoire sur chaque nœud peut

être une stratégie énergétique plus efficace, puisque la charge utile collective peut réduire les frais généraux de paquets. L'utilisation de la puissance de sortie faible ne garantit pas l'économie d'énergie, qui est liée à des conditions de canal. Une évaluation complète de l'énergie a été donnée dans le réseau d'unslotted CSMA / CA en utilisant un débit de données élevé (1 Mbps et 2 Mbps peuvent être fournis par iHop@Node). La méthode d'analyse en composantes principales a été utilisée pour régler chaque paramètre de l'algorithme unslotted CSMA / CA et d'observer l'impact de ces param¨¨tres sur la consommation d'énergie du réseau de capteurs. La consommation d'énergie des réseaux ESB et SB ont été étudiés à partir de plusieurs aspects. En raison du mécanisme d'accès de canal simple, le réseau ESB, en comparant le réseau basé sur unslotted CSMA/CA, a été observé à avoir aucun avantage dans la consommation d'énergie et la fiabilité du réseau, en particulier sous une grande échelle de réseau.

D'autre part, un cadre d'optimisation basé sur algorithme génétique, nommé iMASKO (**i**NL@**M**ATLAB Genetic **A**lgorithm based **S**ensor Networ**K** **O**ptimizer), est conçu et mis en œuvre dans cette thèse. Avec la capacité de recherche rapide et efficace des algorithmes génétiques, iMASKO est capable de trouver une bonne solution sur centaines configurations possibles d'un protocole pour un scénario donné. Les mesures de performances sous l'évaluation des iMASKO ont été générées par SystemC l'environnement de simulation de iWEEP. L'utilisation de iMASKO est démontrée par des scénarios détaillés. Le premier cas est un réseau basé sur laboratoire échelle et algorithme d'unslotted CSMA / CA (réseau basé sur Telos). En utilisant une fonction de coût de la somme pondérée, iMASKO peut automatiquement et intelligemment régler quatre paramètres de l'algorithme unslotted (1872 combinaisons) pour obtenir un bon compromis entre la consommation d'énergie et d'autres mesures tels que le retard et la fiabilité de réseau dans un temps très court comparatif à la simulation exhaustive (1872 exécutions de la simulation). Le deuxième cas est un réseau de corps humain où la capacité d'optimisation de iMASKO en multi-scénario et multi-objectif est testée en utilisant à la fois Telos mote et iHop@Node. Les résultats de l'optimisation de Pareto peuvent offrir une série de solutions de compromis disponibles pour le scénario donné et c'est la décision du concepteur pour sélectionner le plus approprié en fonction de la connaissance empirique ou exigences spécifiques à l'application. De plus, iMASKO est non seulement capable de régler les résultats basé sur simulation ou émulation, mais aussi disponible pour différents modèles basés sur mathématique et analyse théorique.

Malgré le fait que les travaux de recherche mis en œuvre dans cette thèse a obtenu des résultats encourageants, ils ne sont pas encore idéaux et beaucoup de travaux supplémentaires pourraient être prises pour améliorer encore la recherche.

- Il vaut la peine de construire un niveau d'abstraction du système d'exploitation dans iWEEP_SW pour la consommation d'énergie puisque de nombreux nœuds sont entraînés et commandés par un système d'exploitation au lieu de quelques lignes de code de programme. Ainsi, comme la base pour le développement de logiciels, la performance énergétique en considération le système d'exploitation serait nécessaire. D'autre part, si l'analyse de puissance plus précise est requise, la simulation de l'intégration de jeu d'instructions est un bon choix. Par ailleurs, la modélisation d'un modèle de canal plus complexe serait utile pour l'évaluation de l'énergie plus réaliste.

- À partir des fonctions de suivi et de calcul de l'énergie dans le modèle de batterie, plus de sous-modules doivent être ajoutés, par exemple un module de récupération d'énergie qui peut permettre

la récupération d'énergie de différents types de source (par exemple, l'énergie solaire, la lumière, la température, vibrations). De plus, la mise en œuvre d'un modèle de batterie personnalisable peut être utilisée pour des analyses plus efficaces sur la consommation d'énergie et la prédiction de durée de vie.

- Pour les expériences matérielles du monde réel, MEMD et iHop@Node doivent être reconçue avec une taille plus petite. Ensuite, MEMD peut être mis dans un emballage laissant seulement les repères nécessaires afin de rendre plus portable. Les mesures de l'énergie de MICA et Telos série doivent être directement testées par MEMD et utilisée pour la calibration du modèle de logiciel dans iWEEP_SW. Les fonctions spécifiques peuvent être ajoutées à EDMSP selon les besoins des utilisateurs.

- Pour le cadre de l'optimisation (iMASKO) basée sur GA, plus des algorithmes d'optimisation peuvent être intégrés et les résultats expérimentaux de ces algorithmes peuvent être utilisés pour comparer les résultats de GA. Les travaux de recherche dans cette thèse sélectionnent un ensemble de paramètres spécifiques de GA, comme la sélection de roulette, 0.8 fraction crossover, 2 enfants d'élite, les graines aléatoires fixes, etc. Donc, il serait intéressant d'explorer d'autres valeurs de paramètres dans les travaux futurs.

- En plus du protocole unslotted, les autres types de protocoles valent de l'étude. Par exemple, il pourrait être plus stimulant d'évaluer le protocole slotted depuis son mécanisme de fonctionnement est plus complexe que le protocole unslotted et il fournit deux paramètres supplémentaires qui sont *SO* et *BO*.

Puisque iMASKO est un cadre, il n'a pas d'implémentations du protocole, donc la modélisation détaillée et le fonctionnement du protocole est fourni par l'environnement de simulation sous-jacent. Dans ce travail, nous avons adopté la maison construite iWEEP_SW comme le moteur de simulation fondamentale, Dans ce travail, nous avons adopté la maison construite iWEEP_SW comme le moteur de simulation fondamentale, qui a seulement la mise en œuvre de protocoles MAC de unslotted / slotted et les protocoles ESB / SB. Ainsi, dans les futurs travaux de recherche, si d'autres types de protocoles MAC ou protocoles de routage sont requis dans l'optimisation iMASKO, il y a deux façons: (i) pour modéliser la mise en œuvre de protocole détaillé par les designers eux-mêmes, et (ii) remplacer iWEEP_SW avec d'autres simulateurs ou émulateurs. Par exemple, le protocole IEEE 802.11 est disponible pour l'optimisation si NS-2 est utilisé comme le moteur de simulation dans iMASKO. Par ailleurs, différents modèles théoriques et analytiques basées mathématiques peuvent également être utilisées pour l'évaluation à l'avenir. Enfin, notez que si l'approche basée sur la simulation est utilisée dans le cadre iMASKO, l'augmentation de la vitesse de simulation peut directement accélérer le processus d'optimisation.

Bibliography

[1] Konrad Lorincz, Bor-rong Chen, Geoffrey Werner Challen, Atanu Roy Chowdhury, Shyamal Patel, Paolo Bonato, and Matt Welsh, "Mercury: A wearable sensor network platform for high-fidelity motion analysis," *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09)*, pp. 183–196, 2009. (Cited on pages 1, 9 and 76.)

[2] Jaime Lloret, Ignacio Bosch, Sandra Sendra, and Arturo Serrano, "A wireless sensor network for vineyard monitoring that uses image processing," *Sensors*, vol. 11(6), pp. 6165–6196, June 2011. (Cited on pages 1 and 10.)

[3] Lun-Wu Yeh, You-Chiun Wang, and Yu-Chee Tseng, "ipower: an energy conservation system for intelligent buildings by wireless sensor networks," *International Journal of Sensor Networks*, vol. 5(1), pp. 1–10, Feb 2009. (Cited on pages 1 and 10.)

[4] Konstantin Mikhaylov, Jouni Tervonen, Joni Heikkila, and Janne Kansakoski, "Wireless sensor networks in industrial environment: Real-life evaluation results," *2nd Baltic Congress on Future Internet Communications (BCFIC 2012)*, pp. 1–7, April 2012. (Cited on pages 1 and 10.)

[5] Milica Pejanovic Durisic, Zhilbert Tafa, Goran Dimic, and Veljko Milutinovic, "A survey of military applications of wireless sensor networks," *2012 Mediterranean Conference on Embedded Computing (MECO 2012)*, pp. 196–199, June 2012. (Cited on pages 1 and 11.)

[6] Kevin Fall and Kannan Varadhan, "The ns manual (formerly ns notes and documentation)," *[Online], Available at: http://www.isi.edu/nsnam/ns/doc/ns_ doc.pdf*, Nov 2011. (Cited on pages 2, 4, 24 and 25.)

[7] OMNeT++ Network Simulation Framework, *[Online], Available at: http://www.omnetpp.org/*. (Cited on pages 2, 4, 25 and 54.)

[8] Gyula Simon, Peter Volgyesi, Miklos Maroti, and Akos Ledeczi, "Simulation-based optimization of communication protocols for large-scale wireless sensor networks," *Proceedings of 2003 IEEE Aerospace Conference*, vol. 3, pp. 1339–1346, March 2003. (Cited on pages 2, 25 and 113.)

[9] Philip Levis, Nelson Lee, Matt Welsh, and David Culler, "Tossim: Accurate and scalable simulation of entire tinyos applications," *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys'03)*, pp. 126–137, 2003. (Cited on pages 2 and 26.)

[10] Jonathan Polley, Dionysys Blazakis, Jonathan McGee, Dan Rusk, and John S. Baras, "Atemu: A fine-grained sensor network simulator," *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON'04)*, pp. 145–152, Oct 2004. (Cited on pages 2 and 26.)

[11] Ben L. Titzer, Daniel K. Lee, and Jens Palsberg, "Avrora: Scalable sensor network simulation with precise timing," *Processing of Fourth International Symposium on Information Sensor Networks (IPSN'05)*, pp. 477–482, April 2005. (Cited on pages 2 and 26.)

[12] Holger Karl and Andreas Willig, "Protocols and architectures for wireless sensor networks, chapter 2 single-node architecture," *John Wiley & Sons*, pp. 18–57, April 2005. (Cited on pages 2 and 17.)

[13] Sivanandam, S. N. and S. N. Deepa, "Introduction to genetic algorithms, chapter 2 genetic algorithms," *Springer Publishing Company, Incorporated, 2007*, pp. 15–36. (Cited on pages 3 and 103.)

[14] Crossbow Technology Inc, "Telosb datasheet," *[Online], Available at: http://www.willow.co.uk/TelosB_Datasheet.pdf.* (Cited on pages 3 and 7.)

[15] Moteiv Corporation, "Tmote sky datasheet," *[Online], Available at: http://www.eecs.harvard.edu/ konrad/projects/shimmer/references/tmote-sky-datasheet.pdf*, 2006. (Cited on pages 3, 7, 9, 40 and 46.)

[16] Shimmer Research, "Shimmer - wireless sensor platform for wearable applications," *[Online], Available at: http://www.shimmer-research.com/.* (Cited on pages 3, 7, 9 and 40.)

[17] Crossbow Technology Inc, "Mica2 datasheet," *[Online], Available at: https://www.eol.ucar.edu/rtf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf.* (Cited on pages 3, 7, 40 and 70.)

[18] ——, "Micaz datasheet," *[Online], Available at: http://www.openautomation.net/uploadsproductos/micaz_datasheet.pd* (Cited on pages 3, 7, 10, 13, 40 and 70.)

[19] Nanhao Zhu and Ian O'connor, "Energy measurements and evaluations on high data rate and ultra low power wsn node," *IEEE International Conference on Networking, Sensing and Control (ICNSC'13)*, pp. 232–236, April 2013. (Cited on pages 3, 38, 40, 42 and 53.)

[20] IEEE Computer Society, "Part 15.4: Wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (wpans)," *IEEE Std 802.15.4-2006*, 2006. (Cited on pages 3, 7, 19, 20 and 46.)

[21] Nordic Semiconductor Inc, "nrf24l01+ product specification," *Available at: http://www.nordicsemi.com/eng/content/download/2726/34069/file/nRF24L01P_Product_Specification_1_0.pdf*, 2008. (Cited on pages 3, 39, 49, 51, 61, 62 and 84.)

[22] Wikipedia, "Pareto efficiency," *[Online], Avaiable at: http://en.wikipedia.org/wiki/Pareto_efficiency.* (Cited on page 4.)

[23] Atmel Corporation, "Atmega128 datasheet," *[Online], Available at: http://www.atmel.com/Images/doc2467.pdf*, 2011. (Cited on pages 7 and 46.)

[24] Texas Instruments, Inc, "Cc1000 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/cc1000.pdf*, 2009. (Cited on pages 7 and 62.)

[25] ——, "Cc2420 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/cc2420.pdf*, 2013. (Cited on pages 7, 13, 40 and 62.)

[26] Moteiv Corporation, "Telos datasheet," *[Online], Available at: http://www.eecs.harvard.edu/ konrad/References/TinyOSDocs/telos-reva-datasheet-r.pdf*, 2004. (Cited on pages 7 and 40.)

[27] Crossbow Technology, Inc, "Stargate datasheet," *[Online], Available at: https://www.eol.ucar.edu/rtf/facilities/isa/internal/CrossBow/DataSheets/stargate.pdf.* (Cited on pages 7 and 14.)

[28] Rex Min, Manish Bhardwaj, Seong-Hwan Cho, Nathan Iickes, and Eugene Shin, "Energy-centric enabling tecumologies for wireless sensor networks," *IEEE Wireless Communications*, vol. 9(4), pp. 28–39, Aug 2002. (Cited on page 7.)

[29] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38(4), pp. 393–422, March 2002. (Cited on page 8.)

[30] Garg SK, Schwartz S, and Edelman SV, "Improved glucose excursions using an implantable real-time continuous glucose sensor in adults with type 1 diabetes," *Diabetes Care*, vol. 27(3), pp. 734–738, 2004. (Cited on page 9.)

[31] Geoffrey Werner-Allen, Konrad Lorincz, Matt Welsh, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees, "Deploying a wireless sensor network on an active volcano," *Internet Computing, IEEE*, vol. 10(2), pp. 18–25, 2006. (Cited on page 9.)

[32] Danny Hughes, Phil Greenwood, Gordon Blair, Geoff Coulson, Florian Pappenberger, Paul Smith, and Keith Beven, "An intelligent and adaptable grid-based flood monitoring and warning system," *Proceedings of the UK eScience All Hands Meeting*, 2006. (Cited on page 10.)

[33] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li Shiuan Peh, and Daniel Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet," *ACM SIGPLAN Notices*, vol. 37(10), pp. 96–107, 2002. (Cited on page 10.)

[34] Robert Szewczyk, Eric Osterweil, Joseph Polastre, Michael Hamilton, Alan Mainwaring, and Deborah Estrin, "Habitat monitoring with sensor networks," *Communications of the ACM - Wireless sensor networks*, vol. 47(6), pp. 34–40, June 2004. (Cited on page 10.)

[35] Verena Weber, "Smart sensor networks: Technologies and applications for green growth [oecd technical report]," *[Online], Available at: http://www.oecd.org/sti/44379113.pdf*, pp. 1–48, 2009. (Cited on page 10.)

[36] Jorge Tavares, Fernando J. Velez, and Joao M. Ferro, "Application of wireless sensor networks to automobiles," *Measurement Science Review*, vol. 8(3), pp. 65–70, 2008. (Cited on page 11.)

[37] Xingfa Shen, Zhi Wang, and Youxian Sun, "Wireless sensor networks for industrial applications," *Proceedings of the 5th World Congress on Intelligent Control and Automation (WCICA 2004)*, vol. 4, pp. 3636–3640, June 2004. (Cited on page 11.)

[38] Vehbi C. Gungor and Gerhard P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *IEEE Transactions on Industrial Electronics*, vol. 56(10), pp. 4258–4265, Oct 2009. (Cited on page 11.)

[39] A. Prayati, Ch. Antonopoulos, T. Stoyanova, C. Koulamas, and G. Papadopoulos, "A modeling approach on the telosb wsn platform power consumption," *The Journal of Systems and Software*, vol. 83(8), pp. 1355–1363, Aug 2010. (Cited on pages 12, 13, 38, 53, 60, 61 and 70.)

[40] Johan Pouwelse, Koen Langendoen, and Henk Sips, "Dynamic voltage scaling on a low-power microprocessor," *Proceedings of the 7th annual international conference on Mobile computing and networking*, pp. 251–259, 2001. (Cited on pages 13 and 102.)

[41] Microchip Technology Inc, "Mrf24j40 datasheet," *[Online], Available at: http://ww1.microchip.com/downloads/en/DeviceDoc/39776C.pdf*, 2010. (Cited on pages 13 and 44.)

[42] Holger Karl and Andreas Willig, "Protocols and architectures for wireless sensor networks, chapter 5 mac protocols," *John Wiley & Sons*, pp. 111–148, April 2005. (Cited on pages 13, 17 and 18.)

[43] Mahmood Ali, Annette Bohm, and Magnus Jonsson, "Wireless sensor networks for surveillance applications - a comparative survey of mac protocols," *Proceedings of the 4th International Conference on Wireless and Mobile Communications (ICWMC'08)*, pp. 399–403, 2008. (Cited on page 13.)

[44] Vijay Raghunathan, Saurabh Ganeriwal, and Mani Srivastava, "Emerging techniques for long lived wireless sensor networks," *IEEE Communications Magazine*, vol. 44(4), pp. 108–114, April 2006. (Cited on pages 14, 15 and 16.)

[45] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7(3), pp. 537–568, May 2009. (Cited on pages 16 and 17.)

[46] Holger Karl and Andreas Willig, "Protocols and architectures for wireless sensor networks, chapter 3 network architecture," *John Wiley & Sons*, pp. 60–81, April 2005. (Cited on page 16.)

[47] Chengfa Li, Mao Ye, Guihai Chen, and Jie Wu, "An energy-efficient unequal clustering mechanism for wireless sensor networks," *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference (MASS'05)*, Nov 2005. (Cited on page 16.)

[48] Venkatesh Rajendran, Katia Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient collision-free medium access control for wireless sensor networks," *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys'03)*, pp. 181–192, 2003. (Cited on page 17.)

[49] Andre M. Barroso, Utz Roedig, and Cormac J. Sreenan, "μ-mac: an energy-efficient medium access control for wireless sensor networks," *Proceeedings of the Second European Workshop on Wireless Sensor Networks*, pp. 70–80, Jan 2005. (Cited on page 17.)

[50] Sungrae Cho, K. Kanuri, Jin-Woong Cho, Jang-Yeon Lee, and S.-D. June, "Dynamic energy efficient tdma-based mac protocol for wireless sensor networks," *Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services (ICAS-ICNS 2005)*, p. 48, Oct 2005. (Cited on page 17.)

[51] Joseph Polastre, Jason Hill, and David Culler, "Versatile low power media access for wireless sensor networks," *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys'04)*, pp. 95–107, 2004. (Cited on pages 18 and 102.)

[52] Wei Ye, John Heidemann, and Deborah Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 12(3), pp. 493–506, June 2004. (Cited on page 18.)

[53] Ilker Demirkol, Cem Ersoy, and Fatih Alagoz, "Mac protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44(4), pp. 115–121, April 2006. (Cited on page 18.)

[54] Tijs van Dam and Koen Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys'03)*, pp. 171–180, 2003. (Cited on pages 18 and 102.)

[55] Peng Lin, Chunming Qiao, and Xin Wang, "Medium access control with a dynamic duty cycle for sensor networks," *2004 IEEE Wireless Communications and Networking Conference (WCNC'04)*, vol. 3, pp. 1534–1539, March 2004. (Cited on page 19.)

[56] Tao Zheng, Sridhar Radhakrishnan, and Venkatesh Sarangan, "Pmac: An adaptive energy-efficient mac protocol for wireless sensor networks," *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05)*, pp. 65–72, April 2005. (Cited on page 19.)

[57] Phani Kumar, D Janakiram, and G Ashok Kumar, "Operating systems for wireless sensor networks: A survey technical report," *International Journal of Sensor Networks (IJSNet)*, vol. 5(4), pp. 236–255, 2009. (Cited on page 22.)

[58] Kazem Sohraby, Daniel Minoli, and Taieb Znati, "Wireless sensor networks: Technology, protocols, and applications, chapter 10 operating system for wireless sensor networks," *JohnWiley and Sons, Inc*, pp. 273–282, 2007. (Cited on page 22.)

[59] Philip Levis, Sam Madden, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer, and David Culler, "Tinyos: An operating system for sensor networks," *In Ambient intelligence, Springer-Verlag*, pp. 115–148, 2005. (Cited on pages 22 and 23.)

[60] Muhammad Omer Farooq and Thomas Kunz, "Operating systems for wireless sensor networks: a survey," *Sensors*, vol. 11(6), pp. 5900–5930, 2011. (Cited on pages 22 and 23.)

[61] E. Akhmetshina, P. Gburzynski, and F. Vizeacoumar, "Picos: Atiny operation system for extremely small embedded platforms," *Proceedings of the Conference on Embedded Systems and Applications (ESA'03)*, pp. 116–122, June 2003. (Cited on page 23.)

[62] Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," *Proceedings of the 9th Annual IEEE International Conference on Local Computer Networks*, pp. 455–462, Oct 2004. (Cited on page 23.)

[63] Hector Abrach, Jim Carlson, Hui Dai, Jeff Rose, Anmol Sheth, Brian Shucker, and Richard Han, "Mantis: System support for multimodal networks of in-situ sensors," *Proceedings of the 2nd Workshop on Sensor Networks and Applications (WSNA'03)*, pp. 50–59, Sept 2003. (Cited on page 23.)

[64] Arslan Munir and Ann Gordon-Ross, "Optimization approaches in wireless sensor networks," *Sustainable Wireless Sensor Networks. InTech (2010)*, pp. 313–338, 2010. (Cited on page 23.)

[65] Qing Cao, Tarek Abdelzaher, and John Stankovic, "The liteos operating system: Towards unix like abstraction for wireless sensor networks," *Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008)*, pp. 233–244, April 2008. (Cited on page 23.)

[66] Anand Eswaran, Anthony Rowe, and Raj Rajkumar, "Nano-rk: An energy-aware resource-centric rtos for sensor networks," *Proceedings of the 26th IEEE Real-Time Systems Symposium (RTSS'05)*, pp. 10 pp.–265, Dec 2005. (Cited on page 23.)

[67] Anthony Rowe, Karthik Lakshmanan, Haifeng Zhu, and Ragunathan Rajkumar, "Rate-harmonized scheduling for saving energy," *Proceedings of the 29th IEEE Real-Time Systems Symposium (RTSS'08)*, pp. 113–122, Dec 2008. (Cited on page 23.)

[68] Jianliang Zheng and Myung J. Lee, "Will ieee 802.15.4 make ubiquitous networking a reality?: a discussion on a potential low power, low bit rate standard," *IEEE Communications Magazine*, vol. 42(6), pp. 140–146, June 2004. (Cited on page 24.)

[69] Harsh Sundani, Haoyue Li, Vijay Devabhaktuni, Mansoor Alam, and Prabir Bhattacharya, "Wireless sensor network simulators a survey and comparisons," *International Journal Of Computer Networks (IJCN)*, vol. 2(5), pp. 249–265, 2011. (Cited on page 24.)

[70] Valeri Naoumov and Thomas Gross, "Simulation of large ad hoc networks," *Proceedings of the 6th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems (MSWIM¡¯03)*, pp. 50–57, 2003. (Cited on page 24.)

[71] Sung Park, Andreas Savvides, and Mani B. Srivastava, "Sensorsim: a simulation framework for sensor networks," *Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (MSWIM'00)*, pp. 104–111, 2000. (Cited on page 25.)

[72] MannaSim Framework - Main Page, *[Online], Available at: http://www.mannasim.dcc.ufmg.br.* (Cited on page 25.)

[73] A. Kopke, M. Swigulski, K. Wessel, D. Willkomm, P.T. Klein Haneveld, T.E.V. Parker, O.W. Visser, H.S. Lichte, and S. Valentin, "Simulating wireless and mobile networks in omnet++the mixim vision," *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops (Simutools '08), Article No. 71*, 2008. (Cited on page 25.)

[74] MiXiM developers, *[Online], Available at: http://mixim.sourceforge.net/index.html*. (Cited on page 25.)

[75] Feng Chen, Nan Wangy, Reinhard Germany, and Falko Dresslery, "Performance evaluation of ieee 802.15.4 lr-wpan for industrial applications," *Fifth Annual Conference on Wireless on Demand Network Systems and Services (WONS 2008)*, pp. 89–96, Jan 2008. (Cited on page 25.)

[76] Laura Marie Feeney and Daniel Willkomm, "Energy framework: An extensible framework for simulating battery consumption in wireless networks," *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques (SIMUTools '10), Article No. 20*, 2010. (Cited on page 25.)

[77] Castalia-Wireless Sensor Network Simulator, *[Online], Available at: http://castalia.research.nicta.com.au/index.php/en/*. (Cited on page 25.)

[78] Muhammad Imran, Abas Md Said, and Halabi Hasbullah, "A survey of simulators, emulators and testbeds for wireless sensor networks," *2010 International Symposium in Information Technology (ITSim)*, vol. 2, pp. 897–902, June 2010. (Cited on pages 25, 29 and 31.)

[79] NesCT: A language translator, *[Online], Available at http://nesct.sourceforge.net*. (Cited on page 25.)

[80] Prowler: Probabilistic Wireless Network Simulator, *[Online], Available at: http://www.isis.vanderbilt.edu/projects/nest/prowler*. (Cited on page 25.)

[81] Jprowler: Java-based probablistic wireless network simulator, *[Online], Available at: http://w3.isis.vanderbilt.edu/projects/nest/jprowler/*. (Cited on page 25.)

[82] Enrico Perla, Art O Cathain, Ricardo Simon Carbajo, Meriel Huggard, and Ciaran Mc Goldrick, "Powertossim z: realistic energy modelling for wireless sensor network environments," *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks (PM2HW2N '08)*, pp. 35–42, 2008. (Cited on page 26.)

[83] Olaf Landsiedel, Klaus Wehrle, and Stefan Gotz, "Accurate prediction of power consumption in sensor networks," *The Second IEEE Workshop on Embedded Networked Sensors (EmNetS-II)*, pp. 37–44, May 2005. (Cited on pages 26 and 70.)

[84] Fredrik Osterlind, Adam Dunkels, Joakim Eriksson, Niclas Finne, and Thiemo Voigt, "Cross-level sensor network simulation with cooja," *IEEE Conference on Local Computer Networks (LCN'06)*, pp. 641–648, 2006. (Cited on page 27.)

[85] Joakim Eriksson, Adam Dunkels, Niclas Finne, Fredrik Osterlind, and Thiemo Voigt, "Mspsim-an extensible simulator for msp430-equipped sensor boards," *Proceedings of the European Conference on Wireless Sensor Networks (EWSN), Poster/Demo session*, 2007. (Cited on page 27.)

[86] Getting Started with MPLAB SIM, *[Online], Available at: http://www.microchip.com/stellent/idcplg?IdcService=SS_* (Cited on page 27.)

[87] MPLAB Integrated Development Environment, *[Online], Available at: http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469*. (Cited on page 27.)

[88] Accellera Systems Initiative, "About systemc," *[Online], Available at: http://www.accellera.org/downloads/standards/systemc/about_systemc/.* (Cited on pages 27 and 34.)

[89] F. Fummi, D. Quaglia, and F. Stefanni, "A systemc-based framework for modeling and simulation of networked embedded systems," *Forum on Specification, Verification and Design Languages (FDL'08)*, pp. 49–54, Sep 2008. (Cited on pages 27, 35, 38 and 51.)

[90] Jeff Hiner, Ashish Shenoy, Roman Lysecky, Susan Lysecky, and Ann Gordon Ross, "Transaction-level modeling for sensor networks using systemc," *2010 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'10)*, pp. 197–204, June 2010. (Cited on pages 27, 38 and 52.)

[91] A. Mignogna, M. Conti, M. D'angelo, M. Baleani, and A. Ferrari, "Transaction level modeling and performance analysis in systemc of ieee 802.15.4 wireless standard," *11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools (DSD'08)*, pp. 839–843, Sept 2008. (Cited on page 28.)

[92] D. Serna, S. Villa, F. Rivera, and J. Aedo, "A systemc-tlm platform for wireless sensor networks design exploration," *The 2011 International Conference on Embedded System and Applications (ESA'11)*, 2011. (Cited on pages 28 and 38.)

[93] Jan Haase, Markus Damm, Johann Glaser, Javier Moreno, and Christoph Grimm, "Systemc-based power simulation of wireless sensor networks," *Forum on Specification & Design Languages (FDL 2009)*, pp. 1–4, Sept 2009. (Cited on page 28.)

[94] Daniel Weber, Johann Glaser, and Stefan Mahlknecht, "Discrete event simulation framework for power aware wireless sensor networks," *5th IEEE International Conference on Industrial Informatics*, vol. 1, pp. 335–340, June 2007. (Cited on page 28.)

[95] Andrey Somov, Ivan Minakov, Alena Simalatsar, Giorgio Fontana, and Roberto Passerone, "A methodology for power consumption evaluation of wireless sensor networks," *Proceedings of the 14th IEEE international conference on Emerging technologies & factory automation (ETFA'09)*, pp. 1326–1333, Sep 2009. (Cited on page 28.)

[96] Min Chen and Gabriel A. Rincon-Mora, "Accurate electrical battery model capable of predicting runtime and i-v performance," *IEEE Transactions on Energy Conversion*, vol. 21(2), pp. 504–511, June 2006. (Cited on page 29.)

[97] Wan Du, Fabien Mieyeville, David Navarro, and Ian O Connor, "Idea1: A validated systemc-based system-level design and simulation environment for wireless sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 143, Oct 2011. (Cited on pages 29, 40, 53, 54 and 70.)

[98] L. P. Steyn and G. P. Hancke, "A survey of wireless sensor network testbeds," *IEEE Africon 2011*, pp. 1–6, Sep 2011. (Cited on page 29.)

[99] Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh, "Motelab: A wireless sensor network testbed," *Fourth International Symposium on Information Processing in Sensor Networks (IPSN'05)*, pp. 483–488, April 2005. (Cited on page 29.)

[100] MoteLab-Harvard University, *[Online], Available at: http://motelab.eecs.harvard.edu/.* (Cited on page 29.)

[101] Manjunath Doddavenkatappa, Mun Choon Chan, and Ananda A. L, "Indriya: A low-cost, 3d wireless sensor network testbed," *7th International ICST Conference,TridentCom 2011*, pp. 302–316, April 2011. (Cited on page 29.)

[102] Vlado Handziski, Andreas Kopke, Andreas Willig, and Adam Wolisz, "Twist: A scalable and reconfigurable testbed for wireless indoor experiments with sensor networks," *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks (REALMAN '06)*, pp. 63–70, 2006. (Cited on page 29.)

[103] The WUSTL Wireless Sensor Network Testbed, *[Online], Available at http://wsn.cse.wustl.edu/index.php/The_ WUSTL_ Wireless_ Sensor_ Network_ Testbed.* (Cited on page 30.)

[104] Clement Burin Des Rosiers, Guillaume Chelius, Eric Fleury, Antoine Fraboulet, Antoine Gallais, Nathalie Mitton, and Thomas Noel, "Senslab very large scale open wireless sensor network testbed," *Proceeding of 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCOM'11)*, April 2011. (Cited on page 30.)

[105] Senslab, "Senslab-very large scale open wireless sensor network testbed," *[Online], Available at: http://www.senslab.info/.* (Cited on page 30.)

[106] Abdelrahman Abuarqoub, Fayez Al-Fayez, Tariq Alsboui, Mohammad Hammoudeh, and Andrew Nisbet, "Simulation issues in wireless sensor networks: A survey," *The Sixth International Conference on Sensor Technologies and Applications (SENSORCOMM 2012)*, pp. 222–228, 2012. (Cited on page 31.)

[107] F. Losilla, C. Vicente-Chicote, B. Alvarez, A. Iborra, and P. Sanchez, "Wireless sensor network application development: An architecture-centric mde approach," *Proceedings of the First European conference on Software Architecture (ECSA'07)*, pp. 179–194, 2007. (Cited on page 34.)

[108] Alvise Bonivento, Luca P. Carloni, and Alberto Sangiovanni-Vincentelli, "Platform-based design of wireless sensor networks for industrial applications," *Proceedings of the conference on Design, automation and test in Europe (DATE'06)*, pp. 1103–1107, 2006. (Cited on page 34.)

[109] Stuart Swan, "An introduction to system level modeling in systemc 2.0," *[Online], Available at: http://www.es.ele.tue.nl/ heco/courses/EmbSystems/WhitePaper20.pdf*, pp. 1–10, 2001. (Cited on pages 34 and 35.)

[110] Ney Calazans, Edson Moreno, Fabiano Hessel, Vitor Rosa, Fernando Moraes, and Everton Carara, "From vhdl register transfer level to systemc transaction level modeling: a comparative case study," *Proceedings of 16th Symposium on Integrated Circuits and Systems Design (SBCCI'03)*, pp. 355–360, Sep 2003. (Cited on pages 34 and 35.)

[111] Giovanni De Micheli and Rajesh K. Gupta, "Hardware/software co-design," *Proceedings of the IEEE*, vol. 85(3), pp. 349–365, 1997. (Cited on page 34.)

[112] D. Quaglia and F. Stefanni, "Systemc network simulation library (scnsl)," *[Online], Available at: http://sourceforge.net/projects/scnsl/*, 2008. (Cited on page 35.)

[113] Accellera Systems Initiative, "Systemc version 2.0 users guide," *[Online], Available at: http://www.cse.iitd.ac.in/ panda/SYSTEMC/LangDocs/UserGuide20.pdf*, 2003. (Cited on page 35.)

[114] D. C. Black and J. Donovan, "Systemc: From the ground up," *Secaucus, NJ, USA: Springer-Verlag New York, Inc*, 2005. (Cited on page 35.)

[115] Sudeep Pasricha, "Transaction level modeling of soc with systemc 2.0," *Conference Synopsys Users Group (SNUG'02)*, vol. 3, 2002. (Cited on page 36.)

[116] Guido Arnout, "Eda moving up, again!" *Technical Report, OSCI technology symposium at DAC'02*, 2002. (Cited on page 36.)

[117] Anssi Haverinen, Maxime Leclercq, Norman Weyrich, and Drew Wingard, "White paper for systemc based soc communication modeling for the ocp protocol," *[Online], Available at: http://www.ocpip.org/uploads/documents/ocpip_wp_SystemC_Communication_Modeling_2002.pdf*, pp. 1–39, Oct 2002. (Cited on page 36.)

[118] Rabie Ben Atitallah, Smail Niar, Samy Meftali, and Jean-Luc Dekeyser, "An mpsoc performance estimation framework using transaction level modeling," *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'07)*, pp. 525–533, Aug 2007. (Cited on page 38.)

[119] J. Haase, M. Damm, J. Glaser, J. Moreno, and C. Grimm, "Systemc-based power simulation of wireless sensor networks," *Forum on Specification & Design Languages (FDL'09)*, pp. 1–4, Sep 2009. (Cited on page 38.)

[120] Mario Lang, Jan Haase, and Joseph Wenninger, "Distributed multi-level simulation of wireless sensor networks," *24th International Conference on Architecture of Computing Systems (ARCS'11)*, Feb 2011. (Cited on page 38.)

[121] Greg Morton, "Msp430 competitive benchmarking," *Application Report, [Online], Available at: http://www.gaw.ru/pdf/TI/app/msp430/slaa205.pdf*, 2004. (Cited on pages 38 and 68.)

[122] Nordic Semiconductor Inc, "2.4ghz rf ultra low power 2.4ghz rf ics/solutions," *[Online], Available at: http://www.nordicsemi.com/eng/Products/2.4GHz-RF*. (Cited on pages 39 and 49.)

[123] San Jose, "Wireless communications and the tinyos radio stack," *Technical Report*, 2005. (Cited on page 39.)

[124] Joe Polastre, "Radio stack iteration-how to improve the cc1000," *[Online], Available at: http://webs.cs.berkeley.edu/retreat-1-04/slides/joep-nest-cc1000.ppt*, 2004. (Cited on page 39.)

[125] Joseph Polastre, Robert Szewczyk, and David Culler, "Telos: enabling ultra-low power wireless research," *Fourth International Symposium on Information Processing in Sensor Networks (IPSN'05)*, pp. 364–369, April 2005. (Cited on pages 40, 45, 46 and 70.)

[126] Aleksandar Milenkovic, Chris Otto, and Emil Jovanov, "Wireless sensor networks for personal health monitoring: Issues and an implementation," *Journal Computer Communications*, vol. 29(13-14), pp. 2521–2533, 2006. (Cited on page 40.)

[127] Chris Otto, Aleksandar Milenkovic, Corey Sanders, and Emil Jovanov, "System architecture of a wireless body area sensor network for ubiquitous health monitoring," *Journal of Mobile Multimedia*, vol. 1(4), 2006. (Cited on pages 40 and 80.)

[128] Texas Instruments, Inc, "Msp430 datasheet," *[Online], Available at: www.ti.com/lit/ds/symlink/msp430f1611.pdf*, 2011. (Cited on pages 40, 46 and 62.)

[129] Microchip Technology Inc, "Pic16f87/88 data sheet," *[Online], Available at: http://ww1.microchip.com/downloads/en/devicedoc/30487c.pdf*, 2005. (Cited on pages 42, 46 and 61.)

[130] Custom Computer Service Inc, *[Online], Available at: http://www.ccsinfo.com/*. (Cited on page 42.)

[131] Microchip Technology Inc, "Microchip mcus," *[Online], Available at: http://www.microchip.com/pagehandler/en-us/products/picmicrocontrollers*. (Cited on page 42.)

[132] ——, "Mplab x integrated development environment (ide)," *[Online], Available at http://www.microchip.com/pagehandler/en-us/family/mplabx/*. (Cited on page 42.)

[133] ——, "Microchip miwi wireless networking protocol stack [online]," *Application Note, [Online], Available at: http://ww1.microchip.com/downloads/en/AppNotes/AN1066%20-%20MiWi%20App%20Note.pdf*, 2010. (Cited on page 44.)

[134] Nanhao Zhu and Ian O'connor, "Performance evaluations of unslotted csma/ca algorithm at high data rate wsns scenario," *The 9th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC'2013)*, pp. 406–411, July 2013. (Cited on pages 44 and 98.)

[135] Atmel Corporation, "Atmega103l datasheet," *[Online], Available at: http://www.atmel.com/Images/doc0945.pdf*, 2007. (Cited on page 46.)

[136] ——, "At91fr40162 datasheet," *[Online], Available at: http://www.keil.com/dd/docs/datashts/atmel/at91fr40162.pdf*, 2002. (Cited on page 46.)

[137] Cypress MicroSystems Inc, "Psoc mixed-signal array final data sheet," *[Online], Available at: ftp://ftp.efo.ru/pub/cypress/psoc/DataSheet_CY8C29x66_I.pdf*. (Cited on page 46.)

[138] Freescale Semiconductor Inc, "M68hc08 datasheet," *[Online], Available at: http://www.freescale.com/files/microcontrollers/doc/data_sheet/MC68HC908AP64.pdf*, 2007. (Cited on page 46.)

[139] Semtech Corporation, "Xe8802 sensing machine data acquisition mcu with zoomingadc and lcd driver," *[Online], Available at: http://www.semtech.com/images/datasheet/xe8802.pdf*, 2006. (Cited on page 46.)

[140] Microchip Technology Inc, "Pic18f8722 datasheet," *[Online], Available at: http://ww1.microchip.com/downloads/en/devicedoc/39646b.pdf*, 2004. (Cited on page 46.)

[141] ——, "Pic18f2525 datasheet," *[Online], Available at: http://ww1.microchip.com/downloads/en/devicedoc/39626b.pdf*, 2004. (Cited on page 46.)

[142] Jan-Hinrich Hauer, "Tkn15.4: An ieee 802.15.4 mac implementation for tinyos 2," *[Online], TKN Technical Report TKN-08-003, Available at: http://www.tkn.tu-berlin.de/fileadmin/fg112/Papers/TKN154.pdf*, 2009. (Cited on page 45.)

[143] Nordic Semiconductor Inc, "nrf2401a product specification," *[Online], Available at: http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF2401A*, 2006. (Cited on pages 49 and 62.)

[144] ——, "nrf2402 product specification," *[Online], Available at: http://www.nordicsemi.com/eng/content/download/2740/*
2006. (Cited on page 49.)

[145] ——, "nrf24e1 product specification," *Available at: http://www.nordicsemi.com/eng/content/download/2741/34218/file*
2006. (Cited on page 49.)

[146] ——, "nrf24l01 product specification," *[Online], Available at: http://www.nordicsemi.com/eng/content/download/2730/34105/file/nRF24L01_Product_Specification_v2_0.pdf*, 2010. (Cited on pages 49 and 62.)

[147] ——, "nrf24le1 product specification," *[Online], Available at: http://www.nordicsemi.com/eng/content/download/2443/29442/file/nRF24LE1_Product_Specification_rev1_6.pdf*, 2010. (Cited on page 49.)

[148] ——, "nrf24lu1+ product specification [online]," *Available at: http://www.nordicsemi.com/eng/content/download/2724/34051/file/nRF24LU1P_Product_Spec_v1_1.pdf*, 2010. (Cited on page 49.)

[149] Tom Henderson, "Free space model," *[Online], Available at: http://www.isi.edu/nsnam/ns/doc/node217.html.* (Cited on page 51.)

[150] SynaptiCAD Sales Inc, "Gigawave viewer: Vcd waveform reader," *[Online], Available at: http://www.syncad.com/vcd_waveform_viewer.htm.* (Cited on page 56.)

[151] OriginLab Corporation, *[Online], Available at: http://www.originlab.com.* (Cited on page 56.)

[152] Philipp Hurni, Benjamin Nyenegger, Torsten Braun, and Anton Hergenroeder, "On the accuracy of software-based energy estimation techniques," *Proceedings of the 8th European conference on Wireless sensor networks (EWSN'11)*, pp. 49–64, 2011. (Cited on page 57.)

[153] Aleksandar Milenkovic, Milena Milenkovic, Emil Jovanov, Dennis Hite, and Dejan Raskovic, "An environment for runtime power monitoring of wireless sensor network platforms," *Proceedings of the Thirty-Seventh Southeastern Symposium on System Theory (SSST'05)*, pp. 406–410, March 2005. (Cited on pages 57 and 58.)

[154] Xiaofan Jiang, Prabal Dutta, David Culler, and Ion Stoica, "Micro power meter for energy monitoring of wireless sensor networks at scale," *Proceedings of the 6th international conference on Information processing in sensor networks (IPSN'07)*, pp. 186–195, April 2007. (Cited on pages 57 and 59.)

[155] Hartmut Ritter, Jochen Schiller, Thiemo Voigt, Adam Dunkels, and Juan Alonso, "Experimental evaluation of lifetime bounds for wireless sensor networks," *Proceedings of the Second European Workshop on Wireless Sensor Networks (EWSN'05)*, pp. 25–32, Jan 2005. (Cited on pages 57 and 59.)

[156] Leonardo Barboni and Maurizio Valle, "Experimental analysis of wireless sensor nodes current consumption," *Second International Conference on Sensor Technologies and Applicationsn (SENSORCOMM'08)*, pp. 401–406, Aug 2008. (Cited on page 60.)

[157] Moslem Amiri, "Measurements of energy consumption and execution time of different operations on tmote sky sensor motes," *Master thesis, MASARYK UNIVERSITY FACULTY OF INFORMATICS*, 2010. (Cited on page 60.)

[158] Rachit Agarwal, Rafael V. Martinez-Catala, Sean Harte, Cedric Segard, and Brendan O'Flynn, "Modeling power in multi-functionality sensor network applications," *Proceedings of the 2008 Second International Conference on Sensor Technologies and Applications (SENSORCOMM'08)*, pp. 507–512, 2008. (Cited on page 60.)

[159] Leo Selavo, Gang Zhou, and John A. Stankovic, "Seemote: In-situ visualization and logging device for wireless sensor networks," *3rd International Conference on Broadband Communications, Networks and Systems (BROADNETS'06)*, pp. 1–9, Oct 2006. (Cited on pages 60 and 64.)

[160] Ivaylo Haratcherev, Gertjan Halkes, Tom Parker, Otto Visser, and Koen Langendoen, "Powerbench: A scalable testbed infrastructure for benchmarking power consumption," *Workshop on Sensor Network Engineering (IWSNE'06)*, pp. 37–44, 2006. (Cited on page 60.)

[161] Texas Instruments, Inc, "Cc1100 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/cc1100.pdf*, 2013. (Cited on page 62.)

[162] ——, "Cc2500 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/cc2500.pdf*, 2011. (Cited on page 62.)

[163] ——, "Cc2480 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/cc2480a1.pdf*, 2011. (Cited on page 62.)

[164] RF Monolithics Inc, "Tr1001 datasheet," *[Online], Available at: http://www.rfm.com/products/data/tr1001.pdf*, 2008. (Cited on page 62.)

[165] ——, "Tr3100 datasheet," *[Online], Available at: http://www.rfm.com/products/data/tr3100.pdf*, 2008. (Cited on page 62.)

[166] Nordic Semiconductor Inc, "nrf905 product specification," *[Online], Available at: http://www.nordicsemi.com/eng/content/download/2452/29528/file/Product_Specification_nRF905_v1.5.pdf*, 2008. (Cited on page 62.)

[167] Nanhao Zhu, Fabien Mieyeville, David Navarro, Wan Du, and Ian O'connor, "Research on high data rate wireless sensor networks," *14eme Journees Nationales du Reseau Doctoral de Micro et Nanoelectronique*, May 2011. (Cited on page 61.)

[168] Benoit Latre, Bart Braem, Ingrid Moerman, Chris Blondia, and Piet Demeester, "A survey on wireless body area networks," *Wireless Networks*, vol. 17(1), pp. 1–18, Jan 2011. (Cited on pages 61, 76 and 116.)

[169] Maxim Integrated Products Inc, "Max4172 datasheet," *[Online], Available at: http://datasheets.maximintegrated.com/en/ds/MAX4172.pdf*, 2012. (Cited on page 61.)

[170] Texas Instruments, Inc, "Lm358 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/lm358.pdf*, 2010. (Cited on page 61.)

[171] ——, "Adc10664 datasheet," *[Online], Available at: http://www.ti.com/lit/ds/symlink/adc10664.pdf*, 2013. (Cited on page 61.)

[172] Future Technology Devices International Limited, "Ft232r usb uart ic," *[Online], Available at: http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf*, 2010. (Cited on page 64.)

[173] Anton Hergenroder, Joachim Wilke, and Detlev Meier, "Distributed energy measurements in wsn testbeds with a sensor node management device (snmd)," *23rd International Conference on Architecture of Computing Systems (ARCS'10)*, pp. 1–7, Feb 2010. (Cited on page 64.)

[174] Anton Hergenroder, Jens Horneber, Detlev Meier, Patrick Armbruster, and Martina Zitterbart, "Distributed energy measurements in wireless sensor networks," *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys'09)*, pp. 299–300, 2009. (Cited on page 64.)

[175] Remon Spekreijse, "A communication class for serial port," *[Online], Available at: http://www.codeguru.com/cpp/i-n/network/serialcommunications/article.php/c2483/A-communication-class-for-serial-port.htm*. (Cited on page 66.)

[176] Microsoft Inc, "Microsoft visual studio," *[Online], Available at: http://www.microsoft.com/visualstudio/eng/team-foundation-service*. (Cited on page 66.)

[177] Bloodshed Software, "Dev-c++," *[Online], Available at: http://www.bloodshed.net/devcpp.html*. (Cited on page 66.)

[178] Eduardo Casilari, Jose M. Cano-Garcia, and Gonzalo Campos-Garrido, "Modeling of current consumption in 802.15.4/zigbee sensor motes," *Sensors*, vol. 10(6), pp. 5443–5468, June 2010. (Cited on page 70.)

[179] Kwang-Soo Kim, Hyunhak Kim, Tae-Wook Heo, Yoonmee Doh, and Jong-Arm Jun, "A smart grid testbed using wireless sensor networks in a building," *The Fifth International Conference on Sensor Technologies and Applications (SENSORCOMM'11)*, pp. 371–374, 2011. (Cited on page 76.)

[180] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52(12), pp. 2292–2330, 2008. (Cited on page 76.)

[181] Wan Du, David Navarro, and Fabien Mieyeville, "A simulation study of ieee 802.15.4 sensor networks in industrial applications by system-level modeling," *2010 Fourth International Conference on Sensor Technologies and Applications (SENSORCOMM'10)*, pp. 311–316, 2010. (Cited on page 76.)

[182] Emil Jovanov, Aleksandar Milenkovic, Chris Otto, and Piet C de Groen, "A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation," *Journal of NeuroEngineering and Rehabilitation*, vol. 2(6), 2005. (Cited on page 76.)

[183] Yu Ge, Liang Liang, Wei Ni, Aung Aung Phyo Wai, and Gang Feng, "A measurement study and implication for architecture design in wireless body area networks," *2012 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM'12)*, pp. 799–804, 2012. (Cited on page 80.)

[184] S. S. Sonavane, V. Kumar, and B. P. Patil, "Msp430 and nrf24l01 based wireless sensor network design with adaptive power control," *ICGST-CNIR Journal*, vol. 8(2), pp. 11–15, 2009. (Cited on pages 81 and 102.)

[185] Mari Carmen Domingo, "Packet size optimization for improving the energy efficiency in body sensor networks," *ETRI Journal*, vol. 33(3), pp. 299–309, June 2011. (Cited on pages 81 and 82.)

[186] D Rohm, M Goyal, H Hosseini, A Divjak, and Y Bashir, "Configuring beaconless ieee 802.15.4 networks under different traffic loads," *AINA '09 Proceedings of the 2009 International Conference on Advanced Information Networking and Applications*, pp. 921–928, 2009. (Cited on pages 84 and 103.)

[187] Min Chen, Sergio Gonzalez, Athanasios Vasilakos, Huasong Cao, and Victor C. Leung, "Body area networks: A survey," *Mobile Networks and Applications*, vol. 16(2), pp. 171–191, April 2011. (Cited on page 93.)

[188] Zhurong Chen, Chao Hu, Jingsheng Liao, and Shoubin Liu, "Protocol architecture for wireless body area network based on nrf24l01," *IEEE International Conference on Automation and Logistics (ICAL'08)*, pp. 3050–3054, 2008. (Cited on pages 93 and 108.)

[189] Andreas Weder, "An energy model of the ultra-low-power transceiver nrf24l01 for wireless body sensor networks," *2010 Second International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN'10)*, pp. 118–123, 2010. (Cited on pages 93 and 108.)

[190] Mark Hempstead, Michael J. Lyons, David Brooks, and Gu-Yeon Wei, "Survey of hardware systems for wireless sensor networks," *Journal of Low Power Electronics*, vol. 4, pp. 1–10, 2008. (Cited on page 93.)

[191] Mackensen, E, Lai, M, and Wendt, T.M, "Bluetooth low energy (ble) based wireless sensors," *Sensors, 2012 IEEE*, pp. 1–4, Oct 2012. (Cited on pages 102 and 108.)

[192] Jinyun Zhang, Philip V. Orlik, Zafer Sahinoglu, Andreas F. Molisch, and Patrick Kinney, "Uwb systems for wireless sensor networks," *Proceedings of the IEEE*, vol. 97(2), pp. 313–331, 2009. (Cited on page 102.)

[193] Chiara Buratti, Andrea Conti, Davide Dardari, and Roberto Verdone, "An overview on wireless sensor networks technology and evolution," *Sensors*, vol. 9(9), pp. 6869–6896, 2009. (Cited on page 102.)

[194] Andrea Castagnetti, Alain Pegatoquet, Cecile Belleudy, and Michel Auguin, "A framework for modeling and simulating energy harvesting wsn nodes with efficient power management policies," *EURASIP Journal on Embedded Systems 2012, no. 1 (2012): 8.*, 2012. (Cited on page 102.)

[195] Luca Benini, Alessandro Bogliolo, and Giovanni De Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8(3), pp. 299–316, June 2000. (Cited on page 102.)

[196] Wei Ye, John Heidemann, and Deborah Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 12(3), pp. 493–506, June 2004. (Cited on page 102.)

[197] A. EI-Hoiydi, J.-D. Decotignie, C. Enz, and E. Le Roux, "Wisemac: an ultra low power mac protocol for the wisenet wireless sensor network," *Proceedings of the first ACM SenSys Conference (SenSys'03)*, 2003. (Cited on page 102.)

[198] Naoto Kimura and Shahram Latifi, "A survey on data compression in wireless sensor networks," *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, vol. 2, pp. 8–13, 2005. (Cited on page 103.)

[199] Bhaskar Krishnamachari, Deborah Estrin, and Stephen Wicker, "The impact of data aggregation in wireless sensor networks," *Proceedings of 22nd International Conference on Distributed Computing Systems Workshops*, pp. 575–578, 2002. (Cited on page 103.)

[200] The MathWorks, Inc, "Genetic algorithm-find global minima for highly nonlinear problems," *[Online], Avaiable at: http://www.mathworks.cn/discovery/genetic-algorithm.html.* (Cited on page 103.)

[201] Sivanandam, S. N. and S. N. Deepa, "Introduction to genetic algorithms, chapter 10 applications of genetic algorithm," *Springer Publishing Company, Incorporated, 2007*, pp. 317–402. (Cited on page 103.)

[202] Felipe Frantz Ferreira, "Architectural exploration methods and tools for heterogeneous 3d-ic," *Ph.D. Thesis, EEA Dept., Ecole Centrale de Lyon*, 2012. (Cited on pages 103 and 111.)

[203] Andrew Chipperfield, Peter Fleming, Hartmut Pohlheim, and Carlos Fonseca, "Genetic algorithm toolbox user's guide," *[Online], Avaiable at: http://crystalgate.shef.ac.uk/code/manual.pdf.* (Cited on page 103.)

[204] Sajid Hussain, Abdul Wasey Matin, and Obidul Islam, "Genetic algorithm for hierarchical wireless sensor networks," *Journal of Network*, vol. 2(5), pp. 87–97, Sep 2007. (Cited on page 105.)

[205] Nesa Sudha, Dr. M. L Valarmathi, and T. Christopahpaul Neyandar, "Optimizing energy in wsn using evolutionary algorithm," *IJCA Proceedings on International Conference on VLSI, Communications and Instrumentation (ICVCI'11)*, vol. 12, pp. 26–29, 2011. (Cited on page 105.)

[206] Jenn-Long Liu and Chinya V. Ravishankar, "Leach-ga: Genetic algorithm-based energy-efficient adaptive clustering protocol for wireless sensor networks," *International Journal of Machine Learning and Computing*, vol. 1(1), pp. 79–85, April 2011. (Cited on page 105.)

[207] Konstantinos P. Ferentinos and Theodore A. Tsiligiridis, "Adaptive design optimization of wireless sensor networks using genetic algorithms," *Computer Networks*, vol. 51(4), pp. 1031–1051, March 2007. (Cited on pages 105 and 109.)

[208] Soumyadip Sengupta, Swagatam Das, M. D. Nasir, and B. K. Panigrahi, "Multi-objective node deployment in wsns: In search of an optimal trade-off among coverage, lifetime, energy consumption, and connectivity," *Engineering Applications of Artificial Intelligence*, vol. 26, pp. 405–416, 2013. (Cited on page 105.)

[209] Shafaq B. Chaudhry, Victor C. Hung, Ratan K. Guha, and Kenneth O. Stanley, "Pareto-based evolutionary computational approach for wireless sensor placement," *Engineering Applications of Artificial Intelligence*, vol. 24, pp. 409–425, 2011. (Cited on page 105.)

[210] Manpreet S. Bhatti, Dhriti Kapoor, Rajeev K. Kalia, Akepati S. Reddy, and Ashwani K. Thukral, "Rsm and ann modeling for electrocoagulation of copper from simulated wastewater: Multi objective optimization using genetic algorithm approach," *Desalination*, no. 274(1), pp. 74–80, 2011. (Cited on page 110.)

[211] Felipe Frantz, Lioua Labrak, and Ian O'Connor, "3d ic floorplanning: Automating optimization settings and exploring new thermal-aware management techniques," *Microelectronics Journal*, vol. 43(6), pp. 423–432, June 2012. (Cited on page 110.)

[212] Andras Varga and Rudolf Hornig, "An overview of the omnet++ simulation environment," *Proceedings of the 1st international conference on simulation tools and techniques for communications, networks and systems & workshops (ICST'08)*, 2008. (Cited on page 113.)

Simulation and Optimization of Energy Consumption on Wireless Sensor Networks

Abstract: The great technique developments of embedded system in recent years have successfully enabled the combination of sensing, data processing and various wireless communication technologies all in one node. Wireless sensor networks (WSNs) that consist of many of such node have gained worldwide attention from academic institutions and industrial communities, since their applications are widespread in such as environment monitoring, military fields, event tracing/tracking and disaster detection.

Due to the reliance on battery, energy consumption of WSNs has always been the most significant concern. In this paper, a mixed method is employed for the accurate energy evaluation on WSNs, which involves the design of a transaction-level system-level based SystemC simulation environment for energy exploration, and the building of an energy measurement system platform for the real world testbed node measurements to calibrate and validate both node energy simulation model and operation model. Elaborate energy consumption of several different node platform based networks are investigated and compared under different kinds of scenarios, and then comprehensive energy-saving strategies are also given after each case scenario for the developers and researchers who focus on the energy-efficient WSNs design.

A genetic algorithm (GA) based optimization framework is designed and implemented using MATLAB for the energy aware WSNs. Due to the global search property of genetic algorithms, the optimization framework is able to automatically and intelligently fine tune hundreds/thousands of potential solutions to find the most suitable tradeoff among energy consumption and other performance metrics. The framework's high efficiency and reliability of finding the tradeoff solutions among node energy, network packet loss and latency have been proved by tuning unslotted CSMA/CA algorithm parameters (used by non-beacon mode of IEEE 802.15.4) in our SystemC-based simulation via a weighted sum cost function. Furthermore, the framework is also available for the multi-scenario and multi-objective based optimization task by studying a typical medical application on human body.

Keywords: Wireless sensor networks (WSNs), energy consumption, simulation/emulation, SystemC, testbeds, measurements, calibration, optimization, genetic algorithms, performance metrics, weighted sum cost function, multi-scenario and multi-objective optimization, Pareto-front

Simulation et optimisation de la consommation énergétique de réseaux de capteurs sans fil

Résumé: Les grandes évolutions de la technique de systèmes embarqués au cours des dernières années ont permis avec succès la combinaison de la détection, le traitement des données, et diverses technologies de communication sans fil tout en un nœud. Les réseaux de capteurs sans fil (WSN) qui se composent d'un grand nombre de ces nœud ont attiré l'attention du monde entier sur les établissements scolaires et les communautés industrielles, puisque leurs applications sont très répandus dans telles que la surveillance de l'environnement, domaines militaire, suivi des événements et la détection des catastrophes.

En raison de la dépendance sur la batterie, la consommation d'énergie des réseaux de capteurs a toujours été la préoccupation la plus importante. Dans cet article, une méthode mixte est utilisé pour l'évaluation précise de l'énergie sur les réseaux de capteurs, ce qui inclut la conception d'un environnement de SystemC simulation base au niveau du système et au niveau des transactions pour l'exploration de l'énergie, et la construction d'une plate-forme de mesure d'énergie pour les mesures de nœud banc d'essai dans le monde réel pour calibrer et valider à la fois le modèle de simulation énergétique de nœud et le modèle de fonctionnement. La consommation d'énergie élaborée de plusieurs différents réseaux basés sur la plate-forme de nœud sont étudiées et comparées dans différents types de scénarios, et puis des stratégies globales d'économie d'énergie sont également donnés après chaque scénario pour les développeurs et les chercheurs qui se concentrent sur la conception des réseaux de capteurs efficacité énergétique.

Un cadre de l'optimisation basée sur un algorithme génétique est conçu et mis en œuvre à l'aide de MATLAB pour les réseaux de capteurs conscients de l'énergie. En raison de la propriété de recherche global des algorithmes génétiques, le cadre de l'optimisation peut automatiquement et intelligemment régler des centaines de solutions possibles pour trouver le compromis le plus approprié entre la consommation d'énergie et d'autres indicateurs de performance. Haute efficacité et la fiabilité du cadre de la recherche des solutions de compromis entre l'énergie de nœud, la perte de paquets réseau et la latence ont été prouvés par réglage paramètres de l'algorithme CSMA / CA de unslotted (le mode non-beacon de IEEE 802.15.4) dans notre simulation basé sur SystemC via une fonction de coût de la somme pondérée. En outre, le cadre est également disponible pour la tâche d'optimisation basée sur multi-scénarios et multi-objectif par l'étude d'une application médicale typique sur le corps humain.

Mots-clés: Réseaux de capteurs sans fil, consommation d'énergie, simulation/émulation, SystemC, bancs d'essai, mesures, calibration, optimisation, algorithmes génétiques, mesures de performance, fonction de coût de la somme pondérée, optimisations des multi-scénario et multi-objectif, Pareto-front

Publications related to this thesis:

Journal

[1] **Nanhao ZHU**, Ian O'CONNOR, "iMASKO: A Genetic Algorithm Based Optimization Framework for Wireless Sensor Networks," *Journal of Sensor and Actuator Networks. 2013, 2(4)*, 675-699.

Conferences

[1] **Nanhao ZHU**, Ian O'CONNOR, "Performance Evaluations of unslotted CSMA/CA Algorithm at High Data Rate WSNs Scenario," *The 9th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC 2013)*, pp. 406-411, July, 2013, Cagliari, Italy.

[2] **Nanhao ZHU**, Ian O'CONNOR, "A GA based Optimization Framework for the Energy Aware Wireless Sensor Networks," *Colloque du GDR SoC-SiP 2013*, July, 2013, Lyon, France.

[3] **Nanhao ZHU**, Ian O'CONNOR, "Energy Measurements and Evaluations on High Data Rate and Ultra Low Power WSN Node," *The 2013 IEEE International Conference on Networking, Sensing and Control (ICNSC 2013)*, pp. 232-236, April, 2013, Paris (Evry), France.

[4] **Nanhao ZHU**, Ian O'CONNOR, "Energy Performance of High Data Rate and Low Power Transceiver based Wireless Body Area Networks," *2nd International Conference on Sensor Networks (SENSORNETS 2013)*, pp. 141-144, Feb, 2013, Barcelona, Spain.

[5] **Nanhao ZHU**, Wan DU, David NAVARRO, Fabien MIEYEVILLE, Ian O CONNOR, "High data rate wireless sensor network research," *Colloque du GDR SoC-SiP 2011*, July, 2011, Lyon, France.

[6] **Nanhao ZHU**, Fabien MIEYEVILLE, David NAVARRO, Wan DU, Ian O CONNOR, "Research on High Data Rate Wireless Sensor Networks," 14ème Journées Nationales du Réseau Doctoral de Micro et Nanoélectronique (JNRDM 2011), May, 2011, Paris, France.

[7] David Navarro, Fabien Mieyeville, Wan Du, Mihai Galos, **Nanhao Zhu**, Ian O'Connor, "Design Framework for Heterogeneous Hardware and Software in Wireless Sensor Networks," *ICSNC 2011 : The Sixth International Conference on Systems and Networks Communications*, pp. 111-116, Oct, 2011, Barcelona, Spain.