UNIVERSITETET I AGDER

MAC Protocols for WuR Enabled WSNs: Design and Performance Evaluation

MIN ZHANG

SUPERVISOR
Frank Y. Li

University of Agder, 2017
Faculty of Engineering and Science
Department of Information and Communication Technology

MAC Protocols for WuR Enabled WSNs: Design and Performance Evaluation

by

Min Zhang

Supervised by
Professor Frank Y. Li

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information and Communication Technology

Department of Information and Communication Technology
Faculty of Engineering and Science
University of Agder

Grimstad, May 21, 2017

Abstract

Increasing energy efficiency is a challenging task for protocol design in wireless sensor networks (WSNs) as well as in Internet of things (IoT). Traditionally, duty-cycled (DC) protocols have been widely adopted for data transmissions in WSNs for energy conservation by reducing idle listening and overhearing. Recently, wake-up radio (WuR) has merged as a promising technique to replace DC protocols thanks to its superior performance in both network lifetime and transmission latency. This thesis work focuses on the design and performance evaluation of WuR-enabled MAC protocols considering various traffic conditions and network topologies.

As the first step, we investigate the niche of WuR by putting forward a question: Does WuR *always* consume lower energy than DC protocols? Through in-depth analysis, we ascertain the outstanding energy performance of WuR at light traffic loads. At the same time, we reveal its disadvantages at heavy traffic loads.

Secondly, we propose a WuR protocol that is capable of avoiding WuC collisions by enabling a contention-based collision avoidance mechanism for WuC transmissions. The performance of the proposed protocol is evaluated by a Markov chain based mathematical model. Numerical results indicate that our proposed protocol achieves higher packet delivery radio (PDR) and network throughput, with the cost of slightly longer packet delay, compared with an existing WuR protocol.

Thirdly, we propose another WuR protocol, referred to as EHA-WuR, which is designed to avoid energy hole in multi-hop networks for multipoint-to-point transmissions. Three operation modes are designed for EHA-WuR. The proposed protocol is implemented in Omnet++ simulator. Numerical results indicate that EHA-WuR significantly extends network lifetime compared with the traditional hop-by-hop operation mode.

Key words: Wireless sensor networks, Internet of things, Wake-up radio, Collision avoidance, Energy hole problem.

Preface

This thesis is based on the work performed for the IKT591 final year Master's thesis project which corresponds to 60 ECTS credits as part of the Master program in Information and Communication Technology at the University of Agder. The thesis work started from 29 Augest, 2016 and ended on 21 May, 2017. As a result of the study, a paper based on the first research question has been accepted by IEEE 85th Vehicular Technology Conference.

I would like to express my sincere thanks to my supervisor professor Frank Y. Li for his patient guidance and support throughout this project. His supervision leads me to open the door of doing research. I also would like to express my gratitude to a PhD candidate, Debasish Ghose, who helped me to better understand the research questions. Last but not least, I would like to thanks all my friends and family for their help and encouragement during my Master's studies.

Min Zhang
Grimstad
May 21, 2017

Contents

List of Figures

List of Tables

List of Abbreviations

IoT	Internet of Things
WSN	Wireless Sensor Network
MAC	Media Access Control
WuR	Wake-up Radio
WuRx	Wake-up Receiver
WuTx	Wake-up Transmitter
MCU	Microcontroller Unit
DC	Duty-cycled
WuC	Wake-up call
SCM-WuR	Sub-carrier Modulation WuR
ACK	Acknowledgment
DTMC	Discrete-time Markov Chain
DES	Discrete Event Simulation
LPL	Low Power Listening
RI	Receiver Initiated
TI	Transmitter Initiated
RN	Reference Node
CA-WuR	WuR with Collision Avoidance
PDR	Packet Delivery Ratio
LR-WPANs	Low-Rate Wireless Personal Area Networks
SIFS	Short Interframe Space
CCA	Clear Channel Assessment
EHA-WuR	WuR with Energy Hole Avoidance

Chapter 1

Introduction

Wake-up radio is an emerging technique for the next generation of wireless sensor networks (WSNs) and the Internet of things (IoT). This thesis focuses on the design and the performance evaluation of media access control (MAC) protocols in wake-up radio (WuR) enabled WSNs. In this chapter, we explore the background overview of this study and provide our motivation for the thesis work. In additional, the research approaches to ascertain the three proposed research questions are also outlined in this chapter.

1.1 Background

As an evolutionary technology, IoT refers to the ever-growing network of physical objects, in which “things” such as sensors and devices are connected with one another for collecting and exchanging information. Various applications depend on IoT such as e-health, smart grids and smart cities. The success of IoT is further encouraged by both the emergence of various standards and protocols, particularly in the field of resource-constrained wireless networks (e.g., BLE, 802.15.4 / ZigBee).

WSN is the fundamental technology of IoT, which performs like a bridge that connects the real world to the digital world. WSN is usually built with low-cost, low-power, and energy-constrained sensors responsible for monitoring physical or environmental conditions. The obtained monitoring information is further transmitted and forwarded by sensor nodes until it reaches the sink and eventually enter the outside world through a gateway.

Since wireless sensors are normally battery operated and due to the difficulty or infeasibility of replacing or recharging sensors, energy efficiency is a fundamental criterion in WSNs. The amount of energy consumed by radio communication is usually substantial compared to other components (e.g., microcontroller unit (MCU), sensor unit), which is controlled by the MAC protocols.

The MAC protocol defined in IEEE 802.15.4 standard is wildly used in WSNs, which suffers from idle listening and overhearing problems. Idle listening refers to the node listens to the channel when there is no traffic. Since many measurements have shown that idle listening consumes 50-100% of the energy required for receiving [6], reducing idle listening is an effective approach to save energy. DC protocols are considered as a traditional aporaoches of reducing idle listening. In DC mechanisms, the radio transceivers is switched off and on regularly, and listen to the radio channel for possible incoming communication during its on-state. Nevertheless, idle listening can not be completely suppressed since nodes have to check for possible communication. DC protocols also suffer from implicit additional latency since no packet is neither sent nor received until the nodes enter their on-state.

As an alternative technology, or some say, the best candidate for replacing traditional DC protocols in WSNs, WuR is merging [6]. Such a technique requires to couple a sensor node with a wake-up receiver (WuRx) which has the role of listening to the channel continuously at an extremely low energy. While the main radio that used for data transmission resides in a deep-sleep mode. Upon the WuRx detects a wake-up call (WuC) generated by a wake-up transmitter (WuTx), the receiver device will turn on its main for the data packet. Therefore, WuR is able to work in an on-demand manner and achieve very low energy consumption and latency.

The author in [1] evaluated the performance of sub-carrier modulation (SCM) WuR and proved the average power consumption of WuR was in the level of up to 1000 times lower than the main radio. An ultra-low power WuR consuming power in the order of nanowatt was designed in [3] while the traditional DC protocols usually have a consuming power in the order of milliwatt. Experimental results indicate that their implemented WuR achieves up to around 70 times longer lifetime then DC protocols.

1.2 Motivation and Problem Statement

As an excellent solution for prolonging the lifetime of WSNs, as well as an encouragement of IoT, WuR has been a very popular topic in recent years. Some of the literatures focused on evaluating the WuRx hardware, while others work on the MAC protocol design for WuR. Since WuR is a technique under active development, there exist many unanswered questions and unsolved problems.

The main idea behind the concept of WuR is to diminish the energy consumption due to idle listening and the envisaged scenarios for applying WuRs are primarily targeted at low traffic load conditions. However, traffic load may gets heavy or even saturated in some type of event-driven data reporting based WSNs. For example when multiple sensor nodes detect an event such as fire detection at the same time. Indeed, most existing studies on WuR did not consider network scenarios with heavy and/or saturated traffic conditions. Although there are some studies on the energy consumption analysis by simulation or experimental results, in-depth analysis rarely exists, especially when comparing with synchronous DC protocols.

WuR-enabled MAC protocols has been considered as a general subclass of WSNs' MAC protocol. Although many literatures have proposed WuR-enabled protocols for some different network scenarios, there is a lack of MAC protocol with more detailed consideration in a WuR-enabled WSN. For example, few WuR-enabled protocol design has the consideration of collision avoidance, which has much concern with the network performance.

On the other hand, in multi-hop WSNs, in order to prolong the overall network lifetime, reducing the energy consumption of sensor nodes is not the only issue. We also need to avoid the exhaustion of a single node since it may cause a failure of the network. Energy hole problem is a major limitation for multi-hop WSNs, especially in multipoint-to-point transmission scenarios regarding the network lifetime. Nodes nearer to the sink will die sooner then other nodes because they forward more data to the sink. Hence, death of the node near the sink causes failure in the communication even if many nodes at the outer regions are with sufficient energy. When the network lifetime is over, up to 90% of the total initial energy is left unused [17]. There are a large number of analytical models and mitigation techniques for the energy hole problem on literature. Despite of extensive efforts in recent years on WuR, the energy hole problem in WuR enabled multi-hop WSNs has not been explored.

The all above observation triggered our motivation to study the following research questions:

- Does WuR always lead to lower energy consumption than DC protocols in WSNs under various traffic conditions?
- Considering a distributed WuR-enabled WSN where multiple sensor nodes are within the transmission range of one another, how can we design a protocol to avoid collision of WuCs? How about the performance of such a protocol?
- Considering the energy hole problem in a multi-hop network scenario, how can we propose a eliminated WuR-enabled protocol so that the energy hole problem can be avoid and the network lifetime can be prolonged.

1.3 Research Approach

To find the solutions of the research questions posed above, a qualitative research approaches based on investigation, analysis, modelling, simulation and evaluation are required.

- To answer the first research question, we compare the energy consumption of a popular WuR implementation, SCM-WuR, with two representative DC protocols, synchronous MAC (S-MAC) protocol [6] and asynchronous MAC (X-MAC) protocol [7], on the consideration of different traffic loads. In our analysis, both short and long WuC range with different transmission power are also compared. The energy analysis is based on a mathematical energy model extended from what proposed in [23].
- To find the solution for the second research question, we investigated the state-of-the-art collision avoidance mechanisms. We propose a contention-based WuR-enabled protocol with exponential back-off. Considering that WuC is sent in a on-demand manner, we apply a collision avoidance mechanism to WuC, which is similar to un-slotted CSMA/CA that defined in non-beacon enabled IEEE 802.15.4. A Markov chain based model is developed to evaluate its performance, including throughput, packet delivery radio, packet delay and power consumption.

- The third research question is addressed by investigating the state-of-the-art strategies on literature to diminish energy hole problem in WSNs. Considering a clustered network scenario which is consist of a sink node, a coordinator node and multiple child nodes. In order to avoid the coordinator node becomimg “energy hole”, we propose a WuR-enabled protocol, in which the coordinator node is only responsible for forwarding WuCs while the data packets are sent directly from the child node to the sink. Since the coordinator node may still die earlier due to its overburden of forwarding WuCs, another operation mode is proposed, which can be used after the coordinator node drains its energy. The designed protocol is implemented in Omnet++ simulator and the performance is evaluated according to the simulation results.

1.4 Report Outline

The rest of the report is structured as follows.

- Chapter 2 presents the related work and enabling technologies, as well as the introduction of the mathematical models and the simulator.
- Chapter 3 investigates the theoretical analysis on the energy consumption of WuR and compares its energy performance with two typical DC protocols.
- In Chapter 4, a WuR-enabled MAC protocol with collision avoidance is proposed. A contention-based mechanism for the WuC transmission is also described in this chapter.
- In Chapter 5, another WuR-enabled MAC protocol for dinimishing the energy hole problem is proposed. The implementation of this protocol in a discrete event simulator is illustrated. In addition, numerical results from simulations are presented and discussed.
- Chapter 6 concludes our thesis work and gives a suggestion of the future works on this topic.

Chapter 2

Related Work and Enabling Technologies

In order to find the solutions of the research questions, studies on the related concepts and existing literatures are required. In this chapter, we summarize the related concepts, principles and existing solutions. Mathematical models and the selection of the network simulators that can be used in our work are also presented.

2.1 Related Work

This section introduces the related concepts of IoT, WSNs and the general working principle of DC protocols. The related work of WuR is categorized into two groups, WuR hardware implementation and the WuR-enabled MAC protocol design. CSMA-CA back-off mechanism is also described briefly in this section.

2.1.1 IoT and WSNs

IoT is a paradigm where everyday objects can be equipped with identifying, sensing, networking and processing capabilities that will allow them not only to communicate with one another, but also with the outside world via Internet [27]. The interconnection of these embedded devices, is expected to usher in automation in nearly all fields, while also enabling advanced applications like a smart grid and smart cities. Experts estimate that the IoT will consist of about 30 billion objects by 2020 [27].

As a fundamental technology of IoT, WSN contains spatially distributed devices using sensors to monitor physical or environmental conditions. Each a device has typically several parts: a radio transceiver with an antenna, a microcontroller, an electronic circuit for interfacing with the sensors and an energy source, usually a battery. Due to limited battery life and the difficulty or infeasibility of replacing or recharging sensors, energy efficiency is a fundamental criterion of WSNs and IoT.

2.1.2 DC protocols in WSNs

The purpose of duty-cycled protocols is to reduce the time a node is idle or spends overhearing an unnecessary activity by periodically putting the node in the sleep state. In DC protocol enabled WSNs, a node is asleep most of the time and wakes up periodically to transmit or receive packets. One duty cycle consists of a sleep period and an active period. DC protocols are divided into two categories: synchronous and asynchronous approaches.

- Synchronous DC protocols require local time synchronisation by exchanging some kind of schedule control messages such that neighbouring nodes share a common sleep schedule. The data packets will only be sent during the active period. The state-of-the-art synchronous protocols include S-MAC, T-MAC, RMAC, etc.

- While a node in asynchronous DC protocols chooses its schedule independently without awareness of its neighbours' schedules. To indicate that there is an impending data transmission, a sender precedes its data with a preamble that is long enough to be detected by all potential receivers. A key advantage of asynchronous DC protocols is that the sender and receiver can be completely decoupled in their duty cycles. The state-of-the-art asynchronous protocols include B-MAC, X-MAC, RIMAC, etc.

2.1.3 The concept of WuR and its hardware implementation

The most ideal condition of the MAC protocols in WSNs is to put the radio in the sleep mode for as long period as possible, and wake it up only when necessary, e.g., to exchange packets [6]. This can be achieved by WuR technique, which couples a sensor node with a WuRx, listening to the channel continuously at an extremely low energy consumption level. While the main radio that used for data packet transmission is put into a deep sleep mode. In a WuR-enabled WSN, before transmitting a data packet, the sender will first transmit a WuC for the purpose of waking up the receiver node. When the WuC is detected by the receiver's WuRx, the receiver will turn on its main radio for receiving data.

One type of the state-of-the-art WuR platforms requires two separate transceivers for the main radio and wake-up radio, i.e., out-of-band WuR. While in-band WuR platform is based on a re-configurable way of operation where the node's radio can be used as both main data radio and WuR.

Although WuR achieves an excellent performance on energy efficiency, it has a drawback of short WuC transmission range. A range extension WuR has been proposed to prolong the transmission range [7]. This WuR hardware design has been further evaluated and implemented as sub-carrier modulation (SCM) WuR protocol in [5] and [1]. The energy consumption analysis, protocol design and simulation implementation of WuR in this thesis work are mainly based on SCM-WuR.

In the hardware of SCM-WuR protocol is based on the low-frequency integrated circuit AS3932 [9] which works at 125 kHz with integrated address correlation. As shown in Figure 2.1, upon receiving a valid wake-up signal, it triggers the MCU to switch the transceiver from sleep to active mode. A 125 kHz wake-up signal is modulated on an 868 MHz carrier frequency at the sender using on-off keying and demodulated at the receiver using a Schottky diode followed by a low-pass filter. Only the envelope signal is then passed to the 125 kHz receiver. The SCM-WuR is implemented to cover the transmission range up to 100 meters using an incorporated RF front-end at the transmitter to enable output power up to +20 dBm.

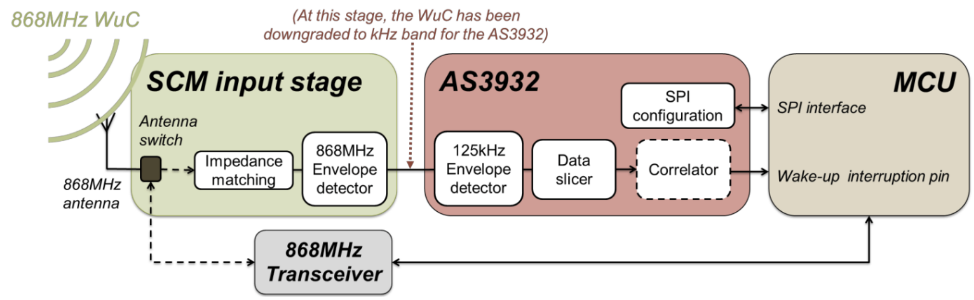


Figure 2.1: Block diagram of the SCM-WuRx/WuTx [5].

2.1.4 WuR-enabled MAC protocols in WSNs and the performance evaluation

The WuR-enabled MAC protocols can be divided into three categories, as shown in Figure 2.2.

- One type of the WuR-enabled protocols is based on path reservation [29] [5]. Data communication and channel reservation is performed in parallel, as on the one hand, communications in WSNs are generally in multi-hop links, and on the other hand the wake-up procedure is independent to data transmission and uses a separate radio channel.
- Another type of WuR-enabled protocols adopted the concept used in DC protocol and implement it to the WuR. Instead of duty-cycling the main radio for data transmission, they duty-cycle the WuR [25] [31][24]. In this case, WuR listens to the channel only in its active period and wakes up the main radio when detecting an incoming WuC.
- The third category is non cycled wake-up MAC protocols, which is based on an always-on low power WuR that is able to continuously listen to the channel, as well as transmit and receive WuCs [4] [10] [19]. In this thesis, we mainly focus on the non cycled WuR-enable MAC protocols.

When evaluating the performance of WuR-enabled protocols, the energy consumption is usually compared with DC protocols. The results of this comparison which can be found in [6] are mostly based on simulations or testbed-based evaluations and only a few of them are based on mathematical analysis.

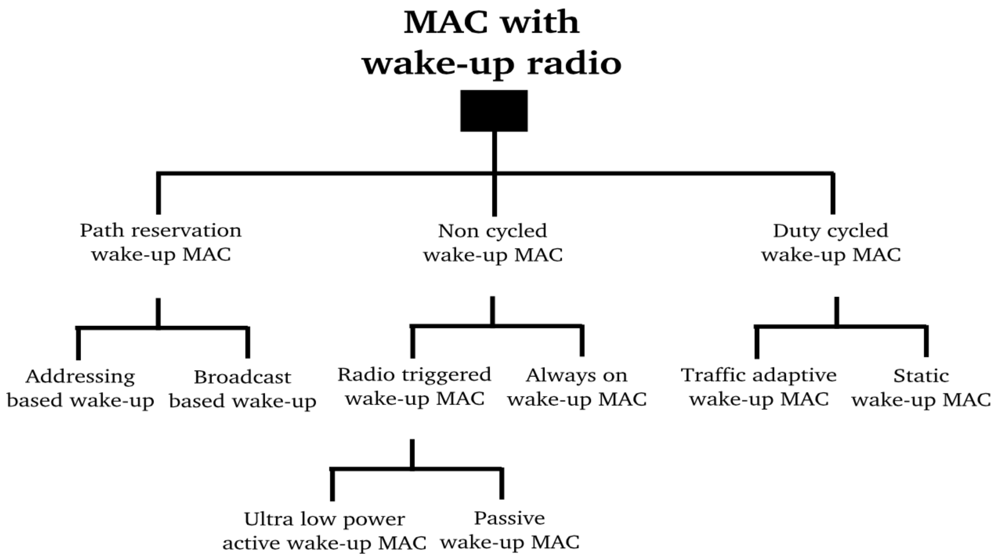


Figure 2.2: Categories of WuR-enabled MAC protocols [6].

2.1.5 IEEE 802.15.4 and the CSMA-CA backoff mechanism

The IEEE 802.15.4 standard uses two types of channel access mechanisms: beacon-enabled and Nonbeacon-enabled, which corresponds to slotted CSMA-CA and unslotted CSMA-CA , depending on the network configuration. The flow chart of the two mechanisms is shown in Figure 2.3.

- Nonbeacon-enabled uses an unslotted CSMA-CA channel access mechanism. Each time a device wishes to transmit data frames or MAC commands, it waits for a random backoff period. If the channel is found to be idle after the random backoff, the device transmits its data. Otherwise, the device waits for another random period before trying to access the channel again. Acknowledgment (ACK) frames are sent without using a CSMA-CA mechanism.

- Beacon mode employs two periods: active (divided into 16 time slots) and inactive (devices enter a low-power mode). At the beginning of the active period, the coordinator sends beacon frames with information regarding the period duration so the duty cycle can vary. The contention access period (CAP) follows the beacon, allowing devices to send frames using slotted CSMA-CA. In slotted CSMA-CA, the backoff periods are aligned with the start of

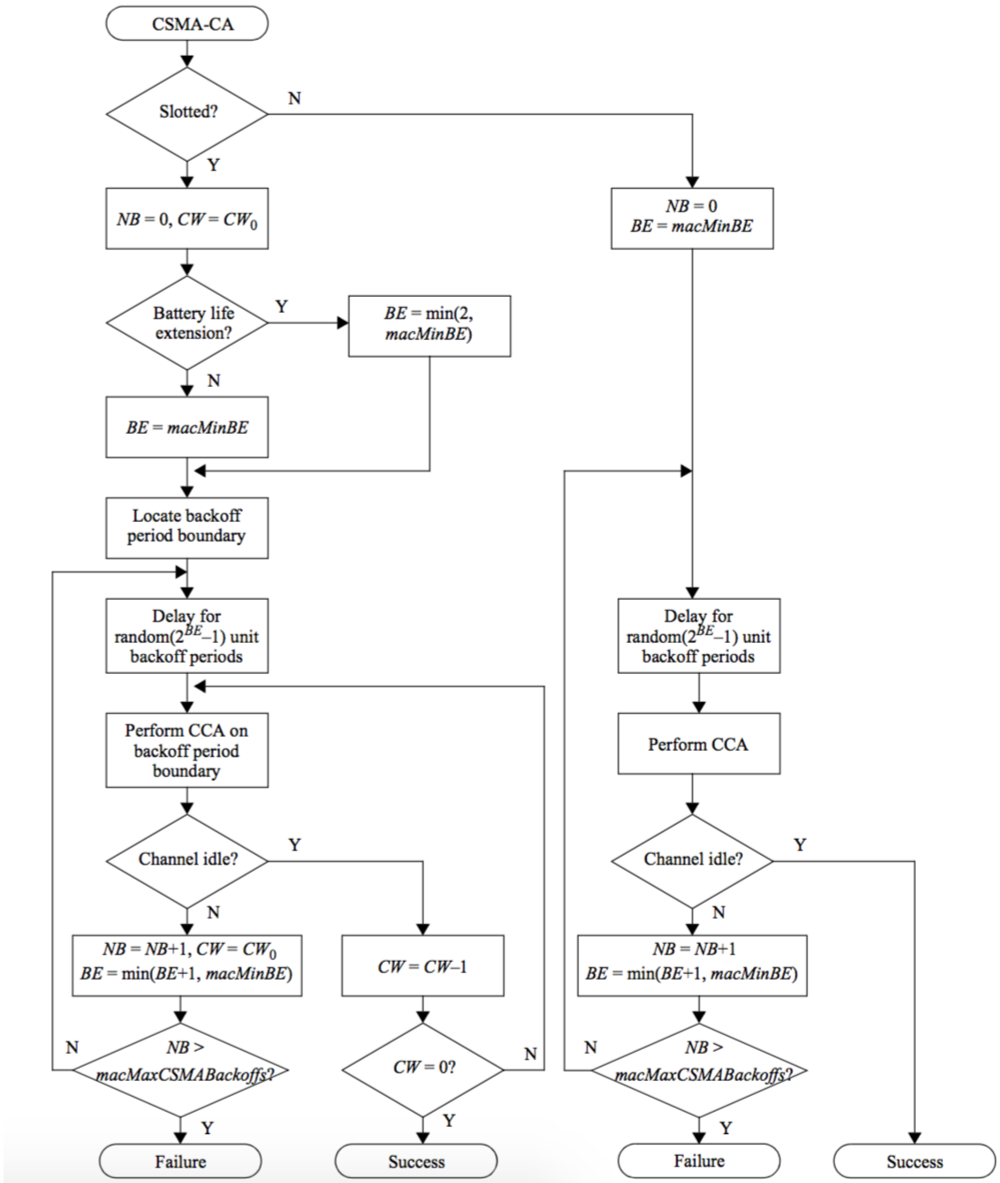


Figure 2.3: Algorithm of CSMA-CA mechanism [32].

the beacon transmission. Each time a device wishes to transmit data frames during the CAP, it locates the boundary of the next backoff period and then waits for a random number of backoff periods. If the channel is sensed busy after backoff, the device waits for another backoff periods before trying to access the channel again. If the channel is idle, the device begins transmitting on the next available backoff period boundary. ACK and beacon frames are sent without using a CSMA-CA mechanism [32].

2.2 The Energy Hole Problem in WSNs and Existing Solutions

2.2.1 Concept of the energy hole problem

Under the condition of multiple hops, sensor nodes close to the sink have larger energy consumption because they are burdened with heavier relay traffic. Therefore, these nodes consume energy at a high rate and deplete early, which usually cause the failure of connection in the network. Due to its effect, network dies early. Consequently, the network lifetime is determined by the weakest node. Figure 2.4 presents an example of the scenario that energy hole could happen. In the figure, sensor node A and B, as the one-hop neighbour of the sink node, will end up as the energy hole in the network.

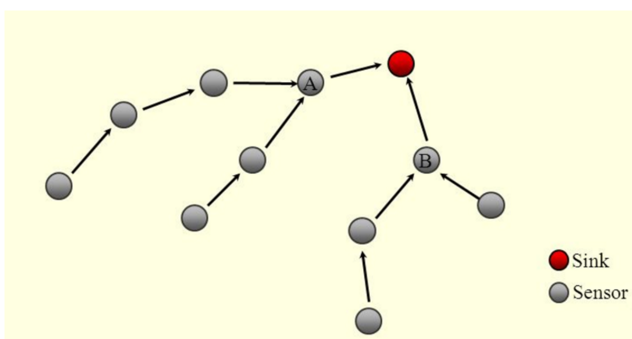


Figure 2.4: An example of the energy hole problem.

2.2.2 Existing solutions for the energy hole problem

There are many analytical models and mitigation techniques for the energy hole problem on literature such as cooperative transmission [14], transmission power adjustment [11] and data aggregation [22].

- Cooperative transmission focuses on using cooperative diversity gain to reduce the transmit power of the node. The author in[14] applied the range extension strategy of cooperative transmission to reduce the loads of the highly-burdened nodes by exploiting the energy of less burdened nodes. When an energy hole forms, instead of passing through the energy hole node, packets can be sent directly to the two-hop-away node by enable cooperative transmission with range extension strategy, as shown in Figure 2.5.

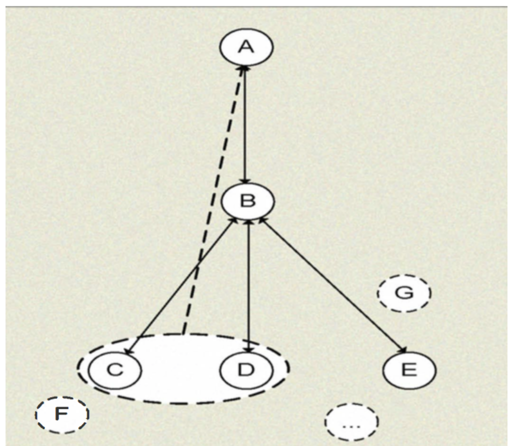


Figure 2.5: Principle of cooperative transmission [13].

- While in transmission power adjustment mechanism, the transmission range is extended by simply increasing the transmission power of a node, so that the packet can skip the energy hole node and be sent directly to the sink.
- The idea of data aggregation is to combine the data coming from multiple source nodes. By eliminating redundancy, minimizing the number of transmissions, the energy consumption of the relay node can be saved, which leads to longer lifetime.

2.3 Mathematical Models: Discrete-time Markov Chain

In probability theory and related fields, Markov process refers to a stochastic process that satisfies the Markov property or is memoryless. We call a stochastic process “memoryless” if the future of the process does not depend on its past, only on its present, as

$$P(X_{n+1} = \boldsymbol{x} \mid X_1 = \boldsymbol{x}_1, X_2 = \boldsymbol{x}_2 ...., X_n = \boldsymbol{x}_n) = P(X_{n+1} = \boldsymbol{x} \mid X_n = \boldsymbol{x}_n).$$

Discrete-time Markov chain (DTMC) is a sequence of random variables $X_1, X_2, X_3, ...$ with the Markov property. It appears as appropriate models in many fields: biological systems, queueing systems, telecommunication and so on. There are two main elements in solving a DTMC model: the transition matrix P and the stationary distribution π.

- The transition matrix is used to describe the transitions of a Markov chain and each of its entries is a nonnegative real number representing a probability. If the probability of moving from state i to j in one time step is $P_{j|i} = P_{i,j}$, the stochastic matrix P is given by using $P_{i,j}$ as the i^{th} row and j^{th} column element, as shown in Figure 2.6, where S denotes the finite state space of a Markov process.

$$\boldsymbol{P} = \begin{pmatrix} P_{1,1} & P_{1,2} & \dots & P_{1,j} & \dots & P_{1,S} \\ P_{2,1} & P_{2,2} & \dots & P_{2,j} & \dots & P_{2,S} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ P_{i,1} & P_{i,2} & \dots & P_{i,j} & \dots & P_{i,S} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ P_{S,1} & P_{S,2} & \dots & P_{S,j} & \dots & P_{S,S} \end{pmatrix}$$

Figure 2.6: Transition matrix of DTMC.

The total of transition probability from a state i to all other states must be 1, expressed as

$$\sum_{j=1}^{S} P_{i,j} = 1. \tag{2.1}$$

- A stationary distribution, or the steady state probability π is a vector whose entries are non-negative and sum to 1. The stationary distribution unchanged by the operation of transition matrix P on it, which can be expressed as

$$\begin{cases} \sum_{j=0}^{\infty} \pi_j = 1, \\ \pi P = \pi. \end{cases} \tag{2.2}$$

2.4 Selection of the Network Simulators

A discrete event simulation (DES) models the operation of a system as a discrete sequence of events in time. Each event occurs at a particular instant in time and marks a change of state in the system. Network designers or researchers usually use the discrete event network simulators to test new networking protocols or to change the existing protocols in a controlled and reproducible manner. There are quite a few current network simulators, which makes it difficult to find the appropriate network simulators for the research or practical requirements. In this section, we introduce the main features of three popular network simulators.

- As an open source, discreet event network simulator, network simulator 3 (NS-3) is an extension of NS-2, which is written completely in C++ and Python bindings for public API's provided. It supports coupling, interoperability, good memory management, debugging of split language object, coding in C++ and object oriented concepts. NS-3 is rapidly developing into a flexible and easy-to-use tool suitable for wireless network simulation.

- Optimized network engineering tools (OPNET) simulator is a well-established commercial discrete-event simulator which can be used free of charge by researchers applying to university program of the product. OPNET simulator defines a network as a collection of sub-models representing sub-networks or nodes and employs hierarchical modeling.

- OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators. OMNeT++ offers an Eclipse-based IDE, a graphical runtime environment, and a host of other tools. As an object-oriented modular DES framework, OMNeT++ provides the solutions for modeling of wired and wireless communication networks, including protocol evaluation, queueing method, hardware architecture validation, etc. In OMNET++, model frameworks are developed as independent projects.

MiXiM is a simulation framework for wireless and mobile networks using the OMNeT++ simulation engine. It offers detailed models of radio wave propagation, interference estimation, radio transceiver power consumption and wireless MAC protocols (e.g. Zigbee) and provides primitives specific to wireless sensor networks. These include characterizations of known hardware radio transceivers such as CC2420 or CC1100, as well as implementations of two largely recognized MAC protocols used for WSN, i.e., B-MAC and un-slotted IEEE 802.15.4. We therefore choose Omnet++ as our network simulator and employ MiXiM framework on top of OMNET++.

Chapter 3

The Niche of WuR: An Energy Comparison Analysis

In order to answer the first research question whether WuR always leads to lower energy consumption than DC protocols, we analyse in-depth the energy consumption of WuR in various traffic loads and compare the results with two state-of-the-art DC protocols in this chapter. As a result of the study, a paper based on this topic had been accepted by IEEE 85th Vehicular Technology Conference, which is presented in Appendix A.

3.1 The Principle of Three Studied Protocols

To make our comparison convincible, we compare WuR with a typical synchronous MAC protocol: S-MAC and a wild-known asynchronous MAC protocol: X-MAC.

3.1.1 S-MAC

As a synchronous MAC protocol, S-MAC enables periodic listen and sleep, and collision avoidance. In S-MAC, each duty cycle is divided into an active period and a sleep period, where packets are only transmitted during the active period. By exchanging SYNC messages, nodes in same cluster share the same sleep schedule. Collision avoidance is implemented by adopting the contention-based scheme used in IEEE.802.11, including both virtual and physical carrier sense, and RTS/CTS exchange. Idle listening happens when a node wakes up and listen to the channel for incoming packets but no packet sent. Since the idle listening current is assumed the same as the current in radio receive state in most literatures, the amount of energy consumption of S-MAC has a great concern with the active percentage of a cycle. The principle of S-MAC is shown in Figure 3.1.

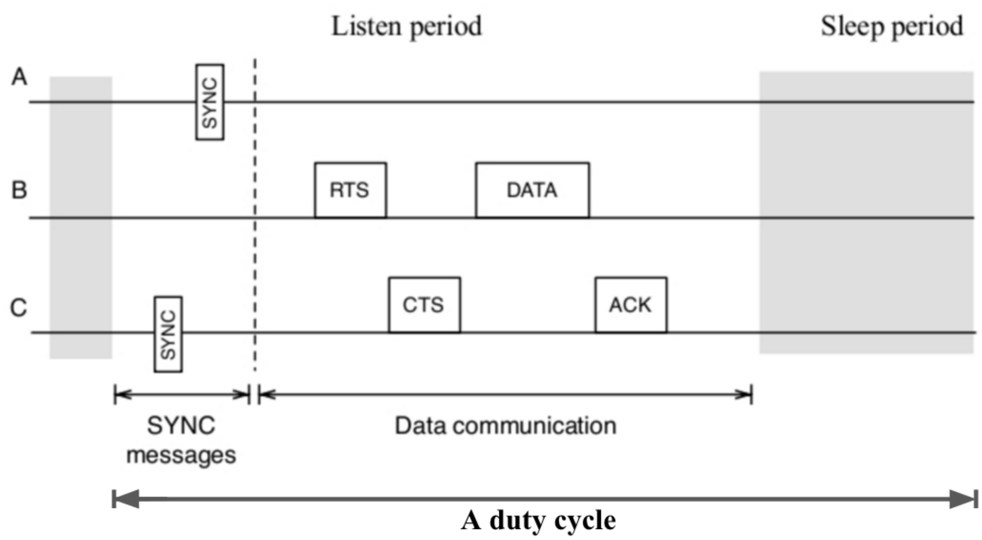


Figure 3.1: Principle of S-MAC protocol.

3.1.2 X-MAC

X-MAC is an asynchronous MAC protocol relying on low power listening (LPL). Instead of transmitting a long preamble before the data packet, it transmits a series of short preambles , each containing the address of the target receiver. Small pauses between preambles permit the target receiver to send back an ACK that stops the preamble sequence. When the node doesn't have any packet to send, it wakes up periodically and samples the channel for possible incoming preambles. Idle listening happens when the node samples the channel, but there is no traffic. After each data communication, node will be set to sleep mode for a predefined constant duration. The energy varies greatly as the receiver's sleep duration, since the sleep duration of the receiver influences the number of preambles the transmitter needs to send before the receiver wakes up. The principle of X-MAC is shown in Figure 3.2.

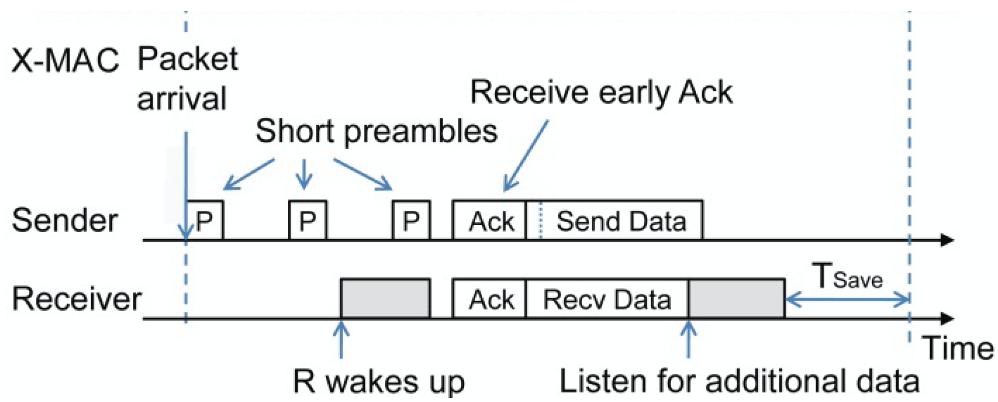


Figure 3.2: Principle of X-MAC protocol.

3.1.3 SCW-WuR

SCW-WuR protocol [5] is based on an in-band WuR hardware platform [7] that the main radio for data transmission also performs as the WuTx. Two operation modes are designed in SCW-WuR protocol: the receiver initiated (RI) mode and the transmitter initiated (TI) mode. In this chapter, only the TI mode is considered. To start a data communication, a WuC with the intended receiver's address is initiated by the WuTx when there is a packet to send. After detecting a WuC, the receiver node turns on its main radio. Assuming that IEEE 802.15.4 is adopted for the main radio operation [5], an ACK is sent back to the transmitter when the data packet is successfully received. Then both nodes will turn off their main radio and enter the sleep mode, with the WuRx keeping listening to the channel. Since the power consumption of WuRx in the listening mode is in the order of μW, compared with traditional radios which are in the order of mW, a huge amount of energy can be saved.

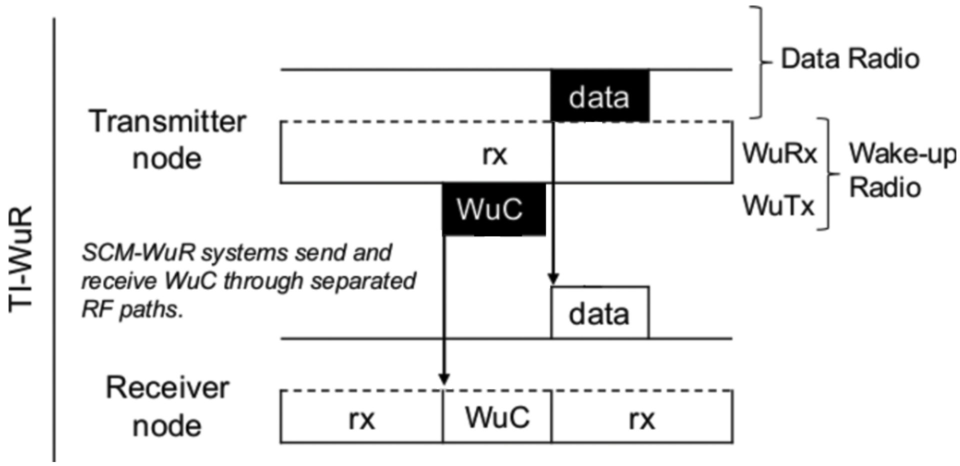


Figure 3.3: Principle of SCW-WuR protocol [1].

3.2 Network Scenario and Energy Analysis

3.2.1 Network scenario

Assume there are N sensor nodes in an one-hop distributed collision-free network where all nodes are within the transmission range of each other. It is assumed that all nodes have an identical constant packet arrival rate λ. Each node sends the same number of packets to one of its neighbours with equal chance. The generated packets are assumed to be unicast. Since the energy consumption of radio communication is the dominant component among energy consuming sources for a node, we focus solely on energy spent in radio communications in our analysis. For the fairness of comparison, ACK for data transmission is assumed enabled for all three protocols.

3.2.2 An outline of the energy analysis

In our calculation, we select arbitrarily one of the N nodes and refer to it as a reference node (RN) and concentrate on calculating the total energy consumption of the RN during the observation time T for three protocols, S-MAC, X-MAC, and SCM-WuR. The energy consumed by the RN in radio communication can be calculated based on its radio operations: data transmission and reception, overhearing, idle listening, and sleeping, denoted as E^d_{tx}, E^d_{rx}, E_{oh}, E_i, and E_s respectively.

- In S-MAC, the energy consumed in SYNC message exchanging is considered. The total energy consumption of a node using S-MAC protocol during T can be expressed as:

$$E_{smac} = E_{tx}^{s} + E_{rx}^{s} + E_{tx}^{d} + E_{rx}^{d} + E_{oh} + E_{i} + E_{s}, \tag{3.1}$$

  where E_{tx}^{s} and E_{rx}^{s} denote the energy used for SYNC message transmission and reception.

- In X-MAC, when a node wakes up, instead of traditional idle listening, it will sample the channel for incoming preamble periodically. The energy consumption of a node using X-MAC during T is denoted as:

$$E_{xmac} = E_{tx}^{p} + E_{rx}^{p} + E_{tx}^{d} + E_{rx}^{d} + E_{sam} + E_{oh} + E_{s}, \tag{3.2}$$

  where E_{tx}^{p}, E_{rx}^{p} and E_{sam} denote the energy used in preamble transmission, reception and channel sampling respectively.

- In SCW-WuR, energy used in WuC transmission and the WuRx listening are the extra energy consumption. Note that there is no idle listening in the main radio since it always keeps in the sleep state when not transmitting or receiving data packet. The idle listening is only exists in WuRx as it always listens to the channel for WuCs all the time in an extremely low energy. The energy consumption of a node using SCW-WuR protocol during T is denoted as:

$$E_{wur} = E_{tx}^{w} + E_{rx}^{w} + E_{tx}^{d} + E_{rx}^{d} + E_{s} + E_{oh} + E_{i}, \tag{3.3}$$

  where E_{tx}^{w} and E_{rx}^{w} denote as the energy used for WuC transmission and reception, respectively.

In this section, only an outline of the energy analysis is presented. More details of the calculations on energy consumption can be found in the paper shown in Appendix A.

3.3 Numerical Results and Discussions

Consider a small-scale one-hop distributed network with 6 sensor nodes. The observation time is set to be 10 minutes. The energy performance of RN at various traffic loads (λ= 0.01, λ =1 and saturated) is analysed in the paper shown in Appendix A. The results include the total energy consumption, the energy efficiency of RN, and the network lifetime. Network scenarios with different periodical sleep time of DC protocols and WuC transmission ranges of SCW-WUR are considered in that paper. Figure 3.4 and 3.5 present the numerical results that are not shown in the paper.

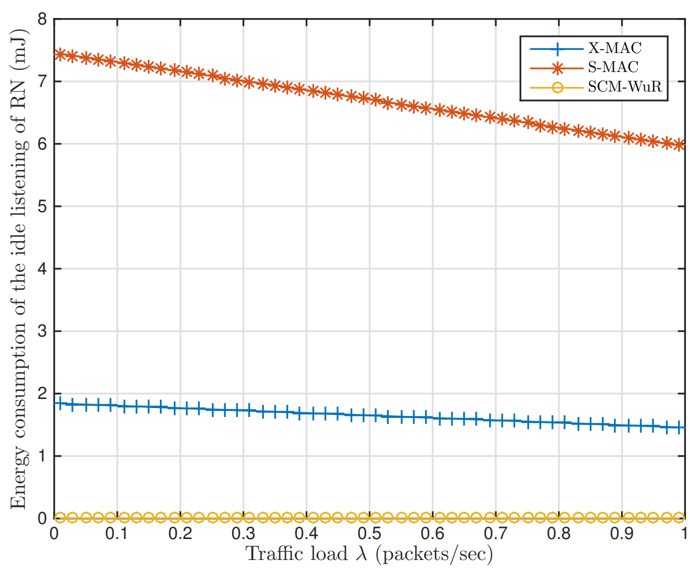
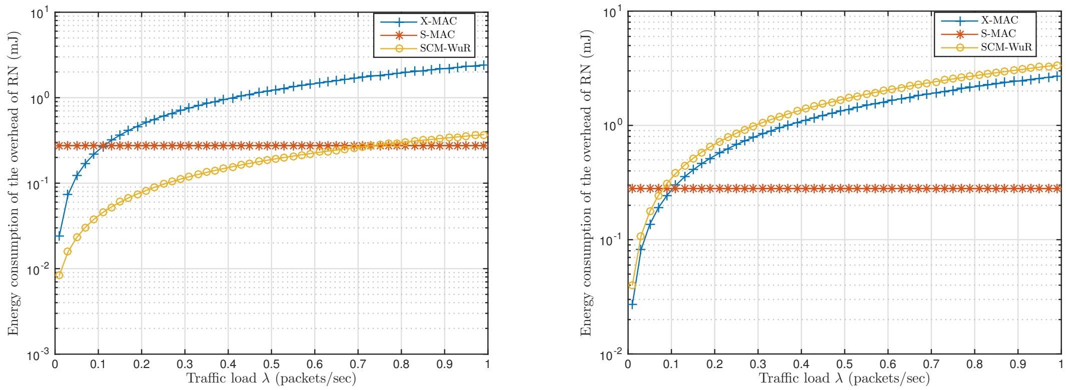


Figure 3.4: Energy consumption of idle listening at various traffic loads.

Figure 3.4 shows the energy of the RN consumed at idle listening of the three protocols during observation time. For S-MAC and X-MAC, when increasing the traffic load, less energy is consumed at idle listening since more time is used in data transmission. In SCM-WuR the energy consumption in idle listening is approximately 0 since idle listening only occurs to the WuRx that consumes power in the order of μW.

Figure 3.5 shows the energy consumption of RN used in protocol overhead. In S-MAC and X-MAC, the protocol overheads are SYNC messages and preambles, respectively. While in SCM-WuR, WuCs are the main overheads. From the figures we can see that in S-MAC, the energy con-



(a) Short WuC transmission range

(b) Long WuC transmission range

Figure 3.5: Energy consumption of the protocol overhead. Duty cycle of the DC protocols is 0.25. The WuC transmission range.

sumed at overhead is not related to traffic load, since the number of the transmitted SYNC messages only depends on the cycle for SYNC message exchanging, not the traffic load. While in X-MAC and SCM-WuR, the energy consumption of overhead increases as the traffic load increases, since the number of transmitted preambles or WuCs are related to the number of transmitted packets. The energy consumed at protocol overheads of SCM-WuR with short range WuC is lower than that of the X-MAC. While the SCM-WuR with long range WuC consumes higher energy on overheads, compared with S-MAC.

Through numerical results (including the results in Figure 3.4 and 3.5 and the figures in the paper), the excellent energy performance of SCM-WuR when traffic load is light is ascertained. When the traffic load gets heavier, SCM-WuR gradually loses its advantage against DC protocols for idle listening. While SCM-WuR always achieves better energy performance than X-MAC. It may not perform better than S-MAC under saturated or close-to-saturation traffic conditions. We also conclude that SCM-WuR is able to handle much higher traffic volume than the other two protocols. The results also indicate that it is not beneficial to enable long range WuCs with respect to energy efficiency, since the power required for long range WuC is much higher than that of the short range.

3.4 Chapter Summary

In this chapter, we addressed the first research question by analysing and comparing the energy consumption of S-MAC, X-MAC and SCW-WuR. The question has been answered based on in-depth analysis and considerable numerical results. Only portion of the contents and results were presented in this chapter and more details has been explained in the paper shown in Appendix A.

Chapter 4

Design and Performance Evaluation of the CA-WuR Protocol

In this chapter, we propose a WuR protocol with collision avoidance (CA-WuR) by enabling a contention-based collision avoidance mechanism for WuC transmissions. The performance of CA-WuR is evaluated in a star topology by a developed DTMC model. Numerical results on the packet delivery ratio (PDR), average packet delay, network throughput and energy consumption are presented and discussed in this chapter.

4.1 Preliminaries on CSMA/CA Versus CSMA-CA

Before explaining our designed WuR protocol, it is worthy mentioning the difference between two mechanisms: CSMA/CA and CSMA-CA.

- CSMA-CA is defined in IEEE 802.15.4 standard that used in low-rate wireless personal area networks (LR-WPANs), as well as WSNs. The principles for beacon-enabled mode and non beacon-enabled mode have been introduced in Chapter 2.
- While CSMA/CA is used in IEEE 802.11 that provides the basis for wireless network products using the Wi-Fi brand. Similar to slotted CSMA-CA, CSMA/CA is based on a super-

frame structure that requires synchronization between the nodes, which can be maintained using beacon messages sent from the access point.

In slotted CSMA/CA, the CCA procedure is performed before the backoff procedure, while in CSMA/CA the backoff procedure is performed firstly. This is the main difference between the two mechanisms.

4.2 CA-WuR Design

Our design CA-WuR protocol is based on the an in-band WuR design: SCW-WuR, which has been introduced in Chapter 2.

4.2.1 WuC transmission and the collision avoidance

In many WuR protocols, there is no collision avoidance mechanism for sending WuCs. In other words, the WuC is transmitted in an ALOHA manner. When a node intends to transmit a WuC, it will send it directly without checking if the channel is idle. This may lead to a serious collision problem when the traffic is heavy.

The authors in [1] proposed a WuR-enabled protocol, in which node will perform CCA before sending WuCs in order to avoid WuC collisions. Recently, a WuR protocol that enables a backoff procedure for sending WuCs was proposed by our research group in [8]. In the designed protocol, the backoff procedure is similar to the CSMA/CA mechanism, in which synchronization is required in order to enable a surperframe structure. However, since WuR is designed to work in an on-demand manner and the main radio will be put into sleep mode most of the time, it is challenging to maintain synchronization of the network, especially for the in-band WuR platforms. This triggers us to propose a WuR protocol with a proper collision avoidance mechanism for sending WuCs that doesn't require network synchronization and meets the feature of WuR.

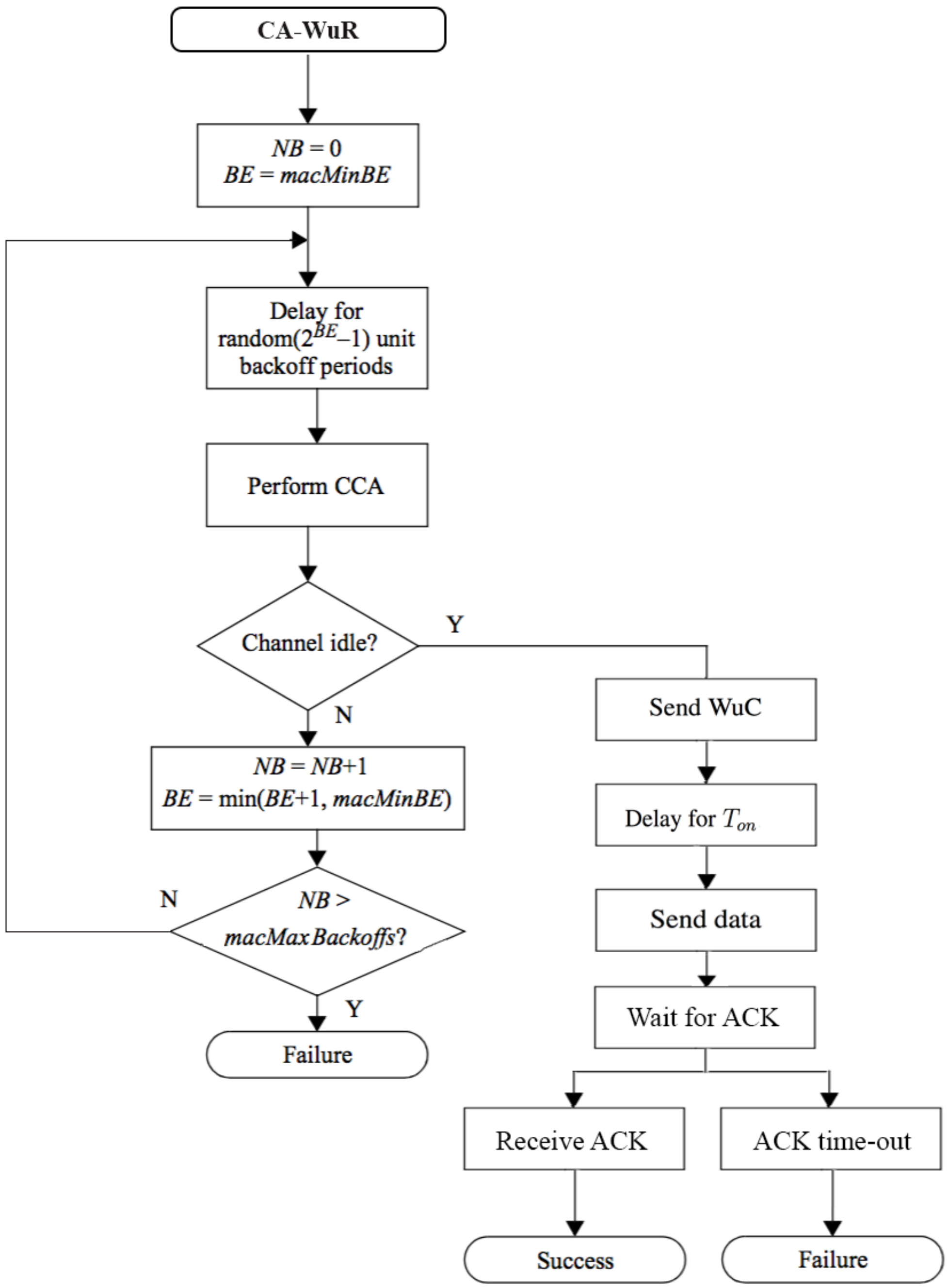


Figure 4.1: Algorithm of the WuC collision avoidance mechanism for CA-WuR in a packet transmission process.

4.2.2 The principle of CA-WuR

In our designed CA-WuR, the collision avoidance mechanism is contention-based and is similar to unslotted CSMA-CA, but applies to the WuCs. More specifically, each time a device intends to transmit WuC, it will wait for a random backoff period. If the channel is found to be idle after the random backoff, the device transmits its WuC. Otherwise, it will wait for another random period before attempting to access the channel again. Data and ACK frames are sent without using a collision avoidance mechanism.

Exponential backoff algorithm is used for the contention procedure. The device shall maintain two variables: NB and BE, representing the backoff stage counter and the backoff exponent, which are initiated 0 and $macMinBE$, respectively. After waiting for a random backoff period (chosen uniformly between 0 and $2^{BE} - 1$), the device performs a CCA. The value of NB will be increased by 1 after each failure of CCA until it reaches maximum retry limit $macMaxBackoffs$. The algorithm of the mechanism in a packet transmission process is shown in Figure 4.1.

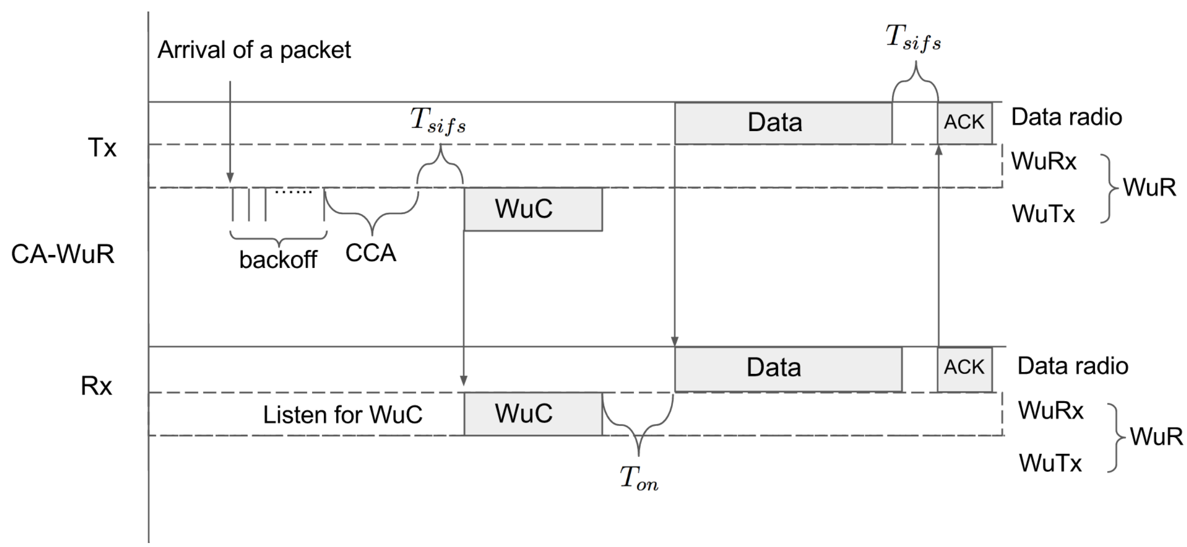


Figure 4.2: Principle of CA-WuR protocol.

After sending a WuC, the transmitter will delay for a T_{on} duration, during which the receiver will turn on its main radio. After receiving the data packet, the receiver will wait for a short interframe space (SIFS) duration that used for radio switching, and send back an ACK. Eventually, a successful transmission ended after the transmitter device receives the ACK message. Figure 4.2 shows the work principle of CA-WuR.

In CA-WuR we use a similar WuC frame structure to what proposed in [8], which is shown in Figure 4.3. A WuC frame is of 48 bits long in which one byte is reserved for the network allocation vector field and two bytes are dedicated to allocating WuR addresses [8].

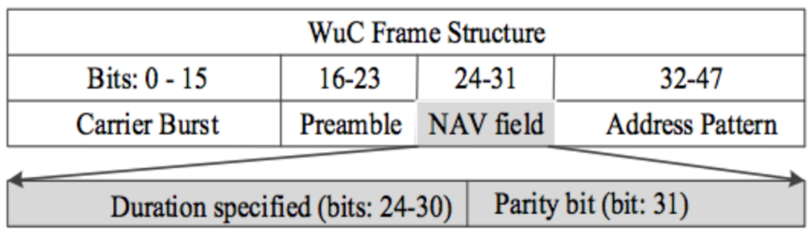


Figure 4.3: WuC frame structure [8].

4.2.3 Configuration of CCA duration to avoid WuC collisions

In the standard of IEEE 802.15.4, CCA is performed for 8 symbol durations while a SIFS period is last for 12 symbol durations. However, for the unlsotted CSMA-CA mechanism, this could cause a serious collision problem if a device senses the channel idle for 8 symbol durations during another device's SIFS period. An assumption that CCA duration is longer than T_{sifs} is made in [15] to prevent collisions.

The T_{on} duration which used for turning on the main radio is 1.792 milliseconds if a CC1101 transceiver [10] is implemented. A similar collision problem could occur during T_{on}, which is shown in Figure 4.4. Therefore, in our designed CA-WuR protocol, CCA duration is initialized to be longer than T_{on}. Then this type of collision can be avoided, as shown in Figure 4.5.

Although longer CCA duration avoids the collision during T_{on}, collisions can take place in the beginning of a WuC transmission caused by the difference of the CCA start epochs of different nodes. This is an inherent problem in an unslotted mechanism, which is proposed in [16]. As shown in Figure 4.6, suppose Node A finishes its backoff phase at time t. Then it performs a CCA for the channel access. Upon sensing the channel idle for T_{cca}, it will take a T_{sifs} durations to

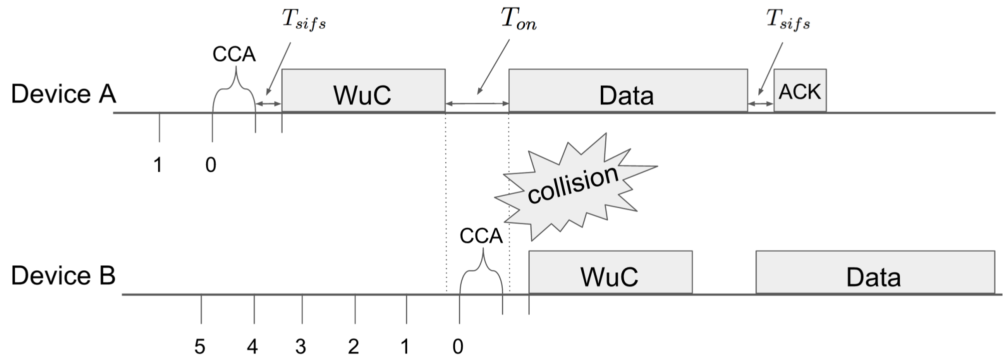


Figure 4.4: Possible collisions during the receiver's main radio setup time.

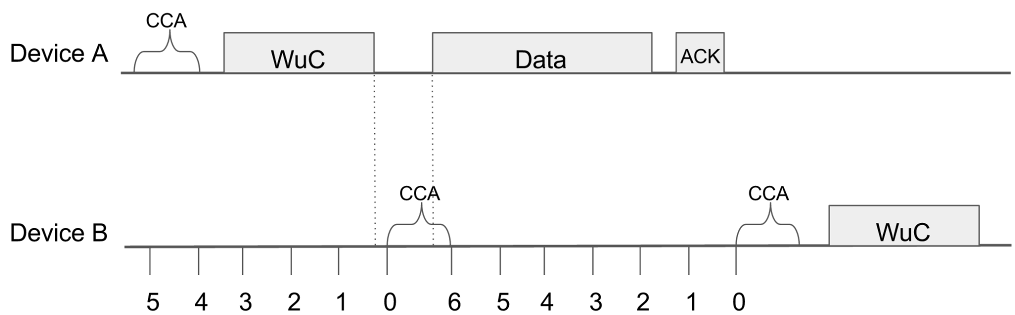


Figure 4.5: Longer CCA duration to avoid WuC collisions.

switch the radio from receive to transmit mode. Therefore, the channel will remain idle until time $(t + T_{cca} + T_{sifs})$ at which Node A begins its frame transmission. However, if Node B starts its CCA at time t', where $t \leq t' \leq (t + T_{sifs})$, it would also sense the channel idle and its WuC frame transmission would collide with that of Node A. The possible time window to encounter a beginning of collision for a given frame is equal to T_{sifs} and will be denoted as "Collision Window" hereafter.

4.3 Network Scenario

Consider a CA-WuR-enabled WSN with a star topology that consists of N sensor nodes and a center node that acts as a common receiver for all other sensor nodes. All nodes are within the

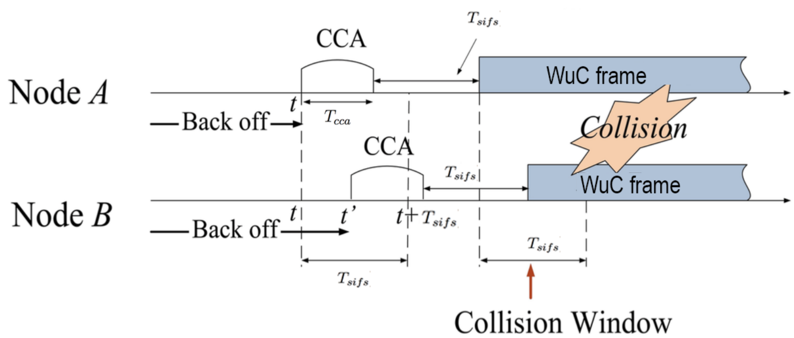


Figure 4.6: Inherent collision problem in unslotted mechanisms.

carrier sensing range of each other. All data packets are assumed to have common length. Data packets arrive at each node according to a Poisson distribution with an arrival rate of λ. Buffering at the nodes is not considered and the channel is assumed to be error-free.

To model the behaviour of CA-WuR, the continuous time evolution of the protocol is approximated by a discrete time evolution where the discrete time unit is equal to one symbol duration ($16\mu s$), which is referred to as mini-slots. All nodes are assumed to be synchronised at mini-slot boundaries. Similar approximations can be found in [9] and [26].

The DTMC model is adopted from [28] that is used for evaluating the performance of unslotted CSMA-CA. The model has been verified by NS-2 simulation in [28]. Two Markov chains are formulated to model the behaviour of an individual node and the common channel. The collision caused by the difference of the CCA start epochs which is mentioned in Section 4.3 is considered in the model. We need to find two key parameters by solving the two Markov chains:

- The probability that a node begins a CCA, denoted by p_{cca}.
- The probability that the channel is sensed idle at the CCA, denoted by p_{idle}.

4.3.1 System model: node state model

In our analysis, we select arbitrarily a sender node as the RN. Retransmission is disabled in the analysis. Figure 4.7 shows the Markov chain model for the behaviour of the RN. The description

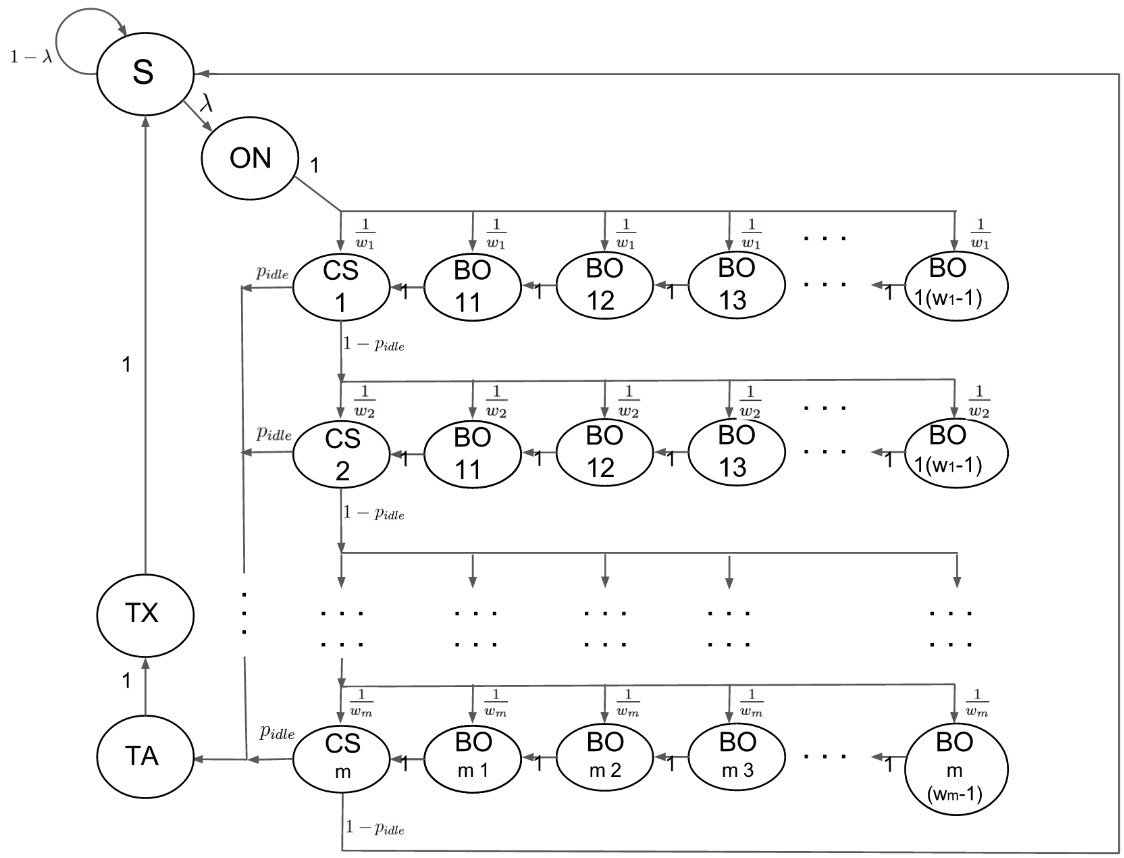


Figure 4.7: DTMC model for an individual node.

for each state in the model is listed in Table 4.1.

When RN does not have a packet to transmit, it resides in state S, where the main radio is off and the WuRx is listening to the channel. Upon an arrival of a packet, RN moves from S state to ON state, during which it turns on its main radio. Then RN moves to the first backoff stage. In Figure 4.7, state BO_{ji} denotes the state in the j^{th} backoff stage with a backoff counter i, where $j \in \{1, 2, ..., macMaxBackoffs{+}1\}$, $i \in \{1, 2, ..., w_j{-}1\}$ and $w_j = 2^{min[(j-1)+macMinBE,\ macMaxBE]}$.

When the first backing off expires, the node moves from state BO_{11} to the first channel sensing state CS_1. If the channel is found to be idle, which occurs with the probability p_{idle}, RN will enter TA state. After radio switching, the node enters state TX and starts the transmission session, which includes WuC transmission, waiting for receiver turning on the main radio, data transmis-

Table 4.1: Description of the symbols in the DTMC model of an individual node.

State symble	Description
S	The state that RN does not have a packet to transmit
ON	The state that RN is turning on the main radio
BO_{ji}	The state that RN is in the j^{th} backoff stage with a backoff counter i
CS_j	The state that RN is in CCA
TA	The state that RN is in radio switching
TX	The state that RN is in a transmission session, including WuC transmission, waiting for receiver to turn on main radio, data transmission, radio switching and ACK reception

sion, radio switching and ACK reception. Note that since the retransmission is disabled, no matter RN receives an ACK or not, at the end of the ACK reception or the ACK timeout, RN will turn off its main radio and moves back to state S. Conversely, if the channel is sensed busy at CS_1 state, the node will move to the next backoff stage BO_{2i} and repeat this procedure until the channel is sensed idle or reach the $macMaxBackoffs$ retry time. If the channel has been sensed busy for $macMaxBackoffs + 1$ times, the packet will be dropped. The steady state equations for the Markov chain model of RN can be obtained as

$$\begin{aligned}
\pi(S) &= (1-\lambda)\pi(S) + (1-p_{idle})\pi(CS_j) + \pi(TX), \\
\pi(ON) &= \lambda\pi(S), \\
\pi(BO_{1i}) &= \lambda\pi(S)/w_1 + h\pi(BO_{1(i+1)}), \\
\pi(CS_1) &= \pi(BO_{11}), \\
\pi(BO_{ji}) &= (1-p_{idle})\pi(CS_{j-1})/w_j + h\pi(BO_{j(i+1)}), where\ 2 \le j \le m \\
\pi(CS_j) &= (1-p_{idle})\pi(CS_{j-1})/w_j + \pi(BO_{j1}), where\ 2 \le j \le m \\
\pi(TX) &= p_{idle}\sum_{j=1}^{m}\pi(CS_j),
\end{aligned} \tag{4.1}$$

where $1 \le j \le m$, $w_j = 2^{BE_j}$ and $h = \begin{cases} 1, & 1 \le i < w_j - 1 \\ 0, & i = w_j - 1 \end{cases}$ and the notation $\pi(STATE)$

represents the ‘long term proportion of transitions into STATE’. The normalisation condition of the DTMC can be expressed as

$$\pi(S)+\pi(ON)+\sum_{j=1}^{m}\sum_{i=1}^{w_j-1}\pi(BO_{ji})+\sum_{j=1}^{m}\pi(CS_j)+\pi(TA)+\pi(TX)=1. \tag{4.2}$$

The equations in (4.1) can be expressed as a function of $\pi(S)$, which can be rearranged to obtain

$$\begin{aligned}\pi(ON)&=\lambda\pi(S),\\ \sum_{i=1}^{w_j-1}\pi(BO_{ji})&=\frac{w_j-1}{2}(1-p_{idle})^{j-1}\lambda\pi(S),\ \ where\ 1\le j\le m,\\ \pi(CS_j)&=(1-p_{idle})^{j-1}\lambda\pi(S),\ \ where\ 1\le j\le m,\\ \pi(TA)&=\pi(TX)=[1-(1-p_{idle})^m]\lambda\pi(S),\end{aligned} \tag{4.3}$$

Thus, (4.2) can be simplified as

$$\left[1+\lambda+\sum_{j-1}^{m}\frac{w_j-1}{2}(1-p_{idle})^{j-1}\lambda+\frac{1-(1-p_{idle})^m}{p_{idle}}\lambda+2p_{idle}^m\lambda\right]\pi(S)=1. \tag{4.4}$$

The steady state probability p_{cca} that a node begins a CCA can be expressed as

$$p_{cca}=\frac{\sum_{j=1}^{m}\pi(CS_j)}{\Lambda} \tag{4.5}$$

where $\Lambda=\pi(S)+T_{on}\pi(ON)+T_{bo}\sum_{j=1}^{m}\sum_{i=1}^{w_j-1}\pi(BO_{ji})+T_{cca}\sum_{j=1}^{m}\pi(CS_j)+T_{sifs}\pi(TA)+(T_{wuc}+T_{on}+T_{data}+T_{sifs}+T_{ack})\pi(TX)$. Here $T_{on}, T_{bo}, T_{cca}, T_{sifs}, T_{wuc}$ and T_{data} are the duration of a node used for turning on main radio, one-slot backoff, CCA, SIFS, WuC and data transmission, respectively. While T_{ack} represents the duration of receiving ACK, which is set to be equal to the ACK time-out duration.

Therefore, p_{cca} can be expressed as a function of $\pi(S)$ and p_{idle}. Since $\pi(S)$ is also a function of p_{idle}, p_{cca} can be completely determined by the p_{idle}.

4.3.2 System model: channel state model

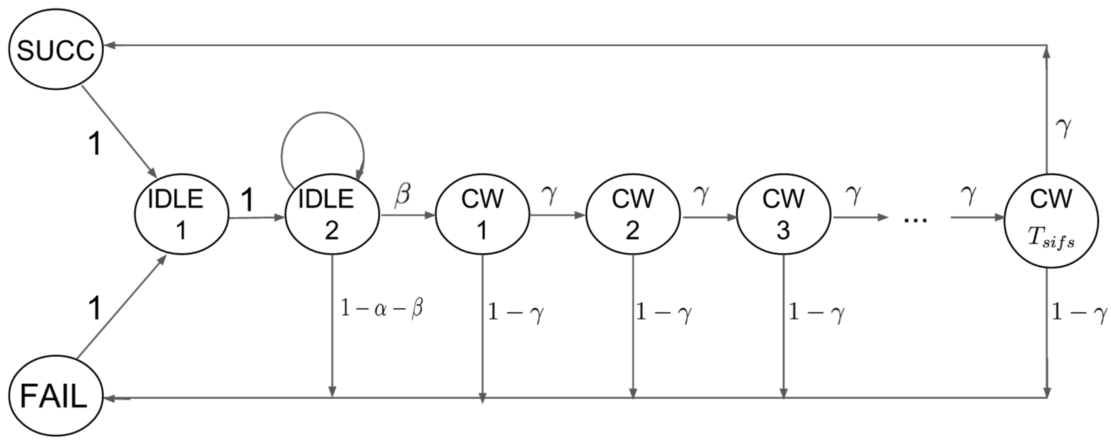


Figure 4.8: DTMC model for the channel.

Table 4.2: Description of symbols in the DTMC model of the channel.

State symble	Description
$SUCC$	The state that the channel is in a successful transmission
$FAIL$	The state that the channel is in collision
$IDLE_1/IDLE_2$	The states that the channel is idle
CW_i	The state that the channel is in the i^{th} collision window

The behaviour of the channel can be modelled using a Markov chain as shown in Figure 4.8. $SUCC$ and $FAIL$ states represent the channel states during a successful transmission and a failed transmission, respectively. The channel will at least be idle for a $T_{cca} + T_{sifs}$ mini-slot before next transmission. Therefore, the state that the channel is idle can be split into two states: $IDLE_1$ and $IDLE_2$. The $IDLE_1$ state represents the first $T_{cca} + T_{sifs} - 1$ mini-slots of the channel idleness, during which WuC or data transmission will not begin at the end of a mini-slot. $IDLE_2$ represents the remaining duration of the channel idleness. The channel remains in the $IDLE_2$ state if none of the nodes begins a transmission, which occurs with probability α and $\alpha = (1 - p_{cca})^N$. The channel will move to CW_1 state if only one node begins a transmission, which occurs with probability $\beta = Np_{cca}(1 - p_{cca})^{N-1}$. Collision happens if more than one nodes start transmission during state $IDLE_2$ and therefore the channel will move to $FAIL$ state. CW_i represents the state that the channel is in the i^{th} mini-slot of the collision window, where $1 \leq i \leq T_{sifs}$. Collision window refers to the duration that a collision might occur in the channel, which has been described in Section 4.3. After expiring the CW_1 state the channel enters state CW_2 if none of the remaining nodes begins a transmission, which occurs with probability $\gamma = (1 - p_{cca})^{N-1}$. The channel will finally get to $SUCC$ state if none of the remaining nodes begins a transmission during the entire collision window. Otherwise the channel will move to $FAIL$ state due to collision. The steady state equations and the normalisation condition are given by

$$\begin{aligned}
\pi(IDLE_1) &= \pi(SUCC) + \pi(FAIL) \\
\pi(IDLE_2) &= \alpha\pi(IDLE_2) + \pi(IDLE_1) \\
\pi(CW_1) &= \beta\pi(IDLE_2) \\
\pi(CW_i) &= \gamma\pi(CW_{i-1}), \quad where 2 \leq i \leq 12 \\
\pi(SUCC) &= \gamma\pi(CW_{T_{sifs}}) \\
\pi(FAIL) &= (1 - \alpha - \beta)\pi(IDLE_2) + (1 - \gamma)\sum_{i=1}^{T_{sifs}} \pi(CW_i)
\end{aligned} \tag{4.6}$$

$$\pi(IDLE_1) + \pi(IDLE_2) + \sum_{i=1}^{T_{sifs}} \pi(CW_i) + \pi(SUCC) + \pi(FAIL) = 1 \tag{4.7}$$

The steady state probability that the CCA process successes equals the probability that the channel is idle for T_{cca} consecutive mini-slots, which can be expressed as

$$p_{idle} = \frac{\frac{T_{sifs}}{T_{cca}+T_{sifs}-1}(T_{cca} + T_{sifs} - 1)\pi(IDLE_1) + \pi(IDLE_2)}{\Gamma} \tag{4.8}$$

where, $\Gamma = (T_{cca} + T_{sifs} - 1)\pi(IDLE_1) + \pi(IDLE_2) + \sum_{i=1}^{T_{sifs}} \pi(CW_i) + (T_{wuc} + T_{on} + T_{data} + T_{sifs} + T_{ack} - T_{sifs})\pi(SUCC) + (T_{wuc} + T_{on} + T_{data} + T_{sifs} + T_{ack} - \frac{T_{sifs}}{2})\pi(FAIL)$. Here $T_{cca} + T_{sifs} - 1$ is the dwell time of the the $IDLE_1^c$ state in mini-slots and $\frac{T_{sifs}}{T_{cca}+T_{sifs}-1}$ denotes the probability that the channel has T_{cca} consecutive mini-slots in state $IDLE_1$.

From equation (4.6) and (4.7), $\pi(IDLE_2)$ can be obtained as

$$\pi(IDLE_2) = \frac{1}{2(1-\alpha)+1+\beta(\frac{1-\gamma^{12}}{1-\gamma})} \tag{4.9}$$

Then the steady state probability of the channel being in all other states can be expressed using $\pi(IDLE_2)$ as

$$\begin{aligned} \pi(IDLE_1) &= (1-\alpha)\pi(IDLE_2) \\ \pi(CW_i) &= \beta\gamma^{i-1}\pi(IDLE_2), \;\; where 1 \leq i \leq T_{sifs} \\ \pi(SUCC) &= \beta\gamma^{T_{sifs}}\pi(IDLE_2) \\ \pi(FAIL) &= (1-\alpha-\beta\gamma^{T_{sifs}})\pi(IDLE_2) \end{aligned} \tag{4.10}$$

Since p_{idle} can be expressed using $\pi(IDLE_2)$, which is a function of p_{cca} through $\alpha, \; \beta$ and γ, the value of p_{idle} and p_{cca} can be obtained by solving the system of equations, which consists of (4.3)-(4.5) and (4.8)-(4.10). Eventually, the steady state probability of all states in both Markov chains can be obtained.

4.4 Performance Evaluation

4.4.1 Parameters of the network performance

The parameters of the network performance are calculated based on the result from the DTMC models, including PDR, average packet delay, network throughput and energy consumption.

Aggregate network throughput

The aggregate network throughput S is defined as the fraction of time the channel spends in successful data transmission, which can be calculated by

$$\begin{aligned} S &= \frac{(T_{wuc} + T_{on} + T_{data} + T_{sifs} + T_{ack})\pi(SUCC)}{\Gamma} \\ &= \frac{(T_{wuc} + T_{on} + T_{data} + T_{sifs} + T_{ack})\beta\gamma^{T_{sifs}}}{(T_{wuc} + T_{on} + T_{data} + T_{sifs} + T_{ack} + T_{cca} + 3/2T_{sifs} - 1)(1-\alpha) + \beta(\frac{1-\gamma^{T_{sifs}}}{1-\gamma} - 3/2T_{sifs}\gamma^{T_{sifs}}) + 1} \end{aligned} \tag{4.11}$$

Packet drop probability and PDR

The probability that a packet is dropped due to $macMaxBackoffs + 1$ times of CCA failure can be expressed as

$$P_{drop} = (1 - p_{idle})^{macMaxBackoffs+1}. \tag{4.12}$$

PDR is defined as the ratio of packets successfully received to the total generated. Since the channel is assumed to be error-free, PDR is equal to the probability that a node wins the contention within $macMaxBackoffs$ retry times and doesn't experience collision during the transmission process, which can be expressed as

$$PDR = (1 - P_{drop}) \cdot \gamma^{T_{sifs}+1} \tag{4.13}$$

Average delay

In our analysis, The average delay of a packet is counted from the time that a data packet arrives at a node until it start being transmitted, which can be calculated by

$$D = \sum_{j=1}^{m} \left[(1 - p_{idle})^{j-1} p_{idle} (\sum_{i=1}^{j} \frac{w_i - 1}{2} + jT_{cca}) \right] \tag{4.14}$$

The delay of the dropped packets is not considered.

Average energy consumption

Let E be the average energy consumed by a sensor node in one mini-slot duration and it can be computed by the sum of the energy consumption of all states in the node state model, which is expressed as

$$\begin{aligned} E = \rho_s \pi(S) + \rho_{on} T_{on} \pi(ON) + \rho_{bo} T_{bo} \sum_{j=1}^{m} \sum_{i=1}^{w_j - 1} \pi(BO_{ji}) + \rho_{cca} T_{cca} \sum_{j=1}^{m} \pi(CS_j) \\ + \rho_x T_{sifs} \pi(TA) + (\rho_{tx}^w T_{wuc} + \rho_i T_{on} + \rho_{tx}^d T_{data} + \rho_x T_{sifs} + \rho_{tx}^d T_{ack}) \pi(TX) \end{aligned} \tag{4.15}$$

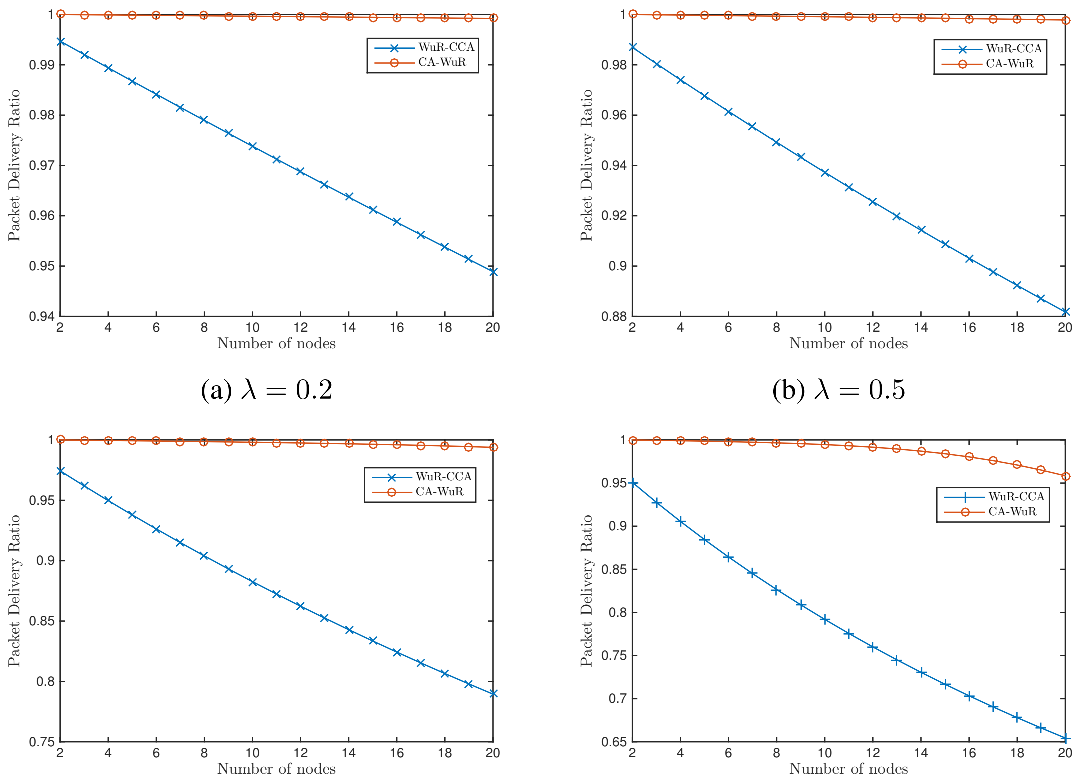
where $\rho_s, \rho_{on}, \rho_{bo}, \rho_{cca}, \rho_x, \rho_{tx}^w$ and ρ_{tx}^d denotes the power consumed at sleep, main radio setup , backoff, CCA, radio switch, WuC and data transmission, respectively.

4.4.2 Numerical results and discussions

We assign default values 3, 5, 4 for $macMinBE, macMaxBE,$ and $macMaxBackoffs$. The parameters $T_{CCA}, T_{setup}, T_{wuc}, T_{data}, T_{sifs}$ and T_{ack} are set to be 113, 112, 375, 200, 12 and 22 mini-slot, respectively. The value of the power consumption at each radio refers to the Table A.1 in Appendix A. We plot the aforementioned performance measures with respect to different arrival rates ($\lambda = 0.2, 0.5, 1$ and 2 packets per second). The results are compared with the WuR protocol proposed in [1], in which the device performs CCA before sending WuCs.

Note that protocol proposed in [1] uses a radio-triggered WuR detailed design. In the comparison, we mainly focus on the performance with different collision avoidance mechanisms of WuC transmissions. For the fairness of the comparison, we assume the value of all the related parameters of the two compared protocols are the same. The results of to various traffic loads are illustrated in Figure 4.9- 4.12. The protocol proposed in [1] is referred to as WuR-CCA hereafter.

Figure 4.9 presents the PDR of the RN at various traffic loads. As shown in the figure, the PDR of CA-WuR is always higher than that of the WuR-CCA. When the number of nodes increases, the PDR of WuR-CCA decreases dramatically. Higher traffic load also leads to lower PDR for WuR-CCA. While the PDR of CA-WuR hardly influenced by the traffic load or the number of nodes.

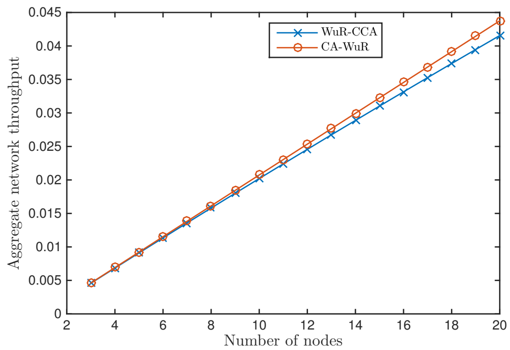


(a) $\lambda = 0.2$ (b) $\lambda = 0.5$

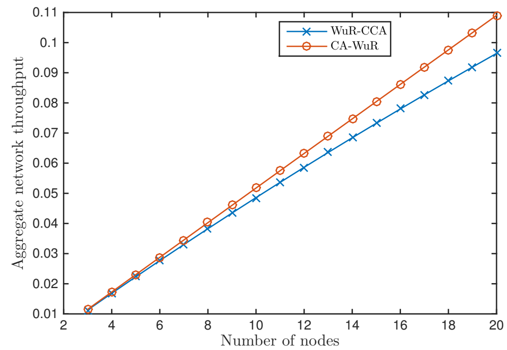
(c) $\lambda = 1$ (d) $\lambda = 2$y

Figure 4.9: PDR of the RN at different traffic loads.

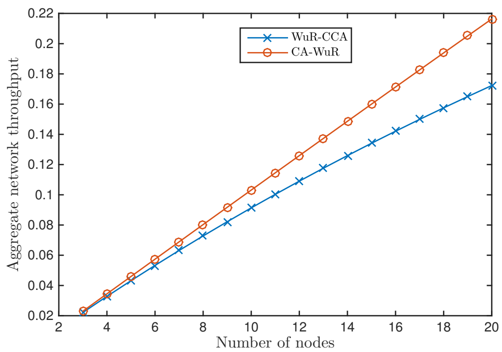
Figure 4.10 shows the aggregated network throughput at various traffic loads. From the figure we can conclude that CA-WuR always achieves higher network throughput at various traffic load conditions, compared with WuR-CCA. The reason is that CA-WuR has higher PDR so that more packets will be delivered successfully, compared with WuR-CCA.



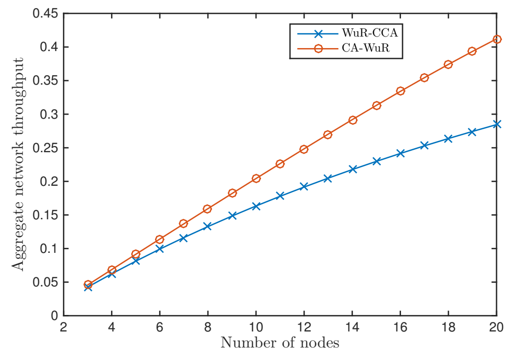
(a) $\lambda = 0.2$



(b) $\lambda = 0.5$



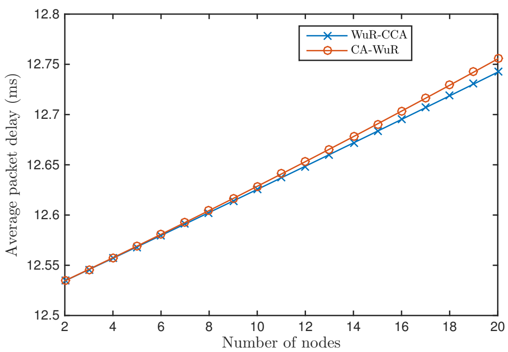
(c) $\lambda = 1$



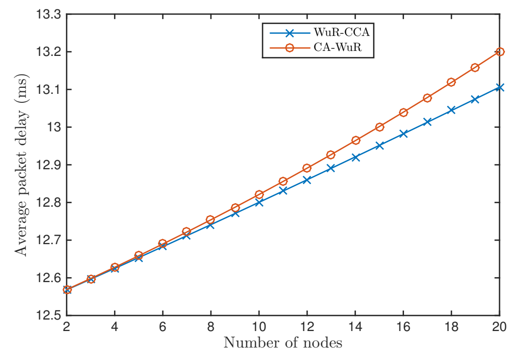
(d) $\lambda = 2$

Figure 4.10: Aggregated network throughput at different traffic loads.

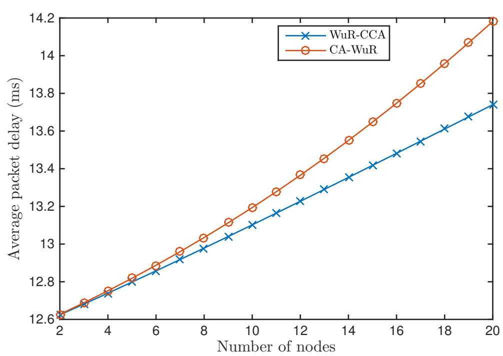
Figure 4.11 illustrates the average packet delay of RN at various traffic loads. As shown is the figure, average packet delay of CA-WuR is always longer than that of WuR-CCA at different traffic loads. When the packet arrival rate increases, the gap between the two compared protocols increases. This is due to the time that CA-WuR used for contention procedure increases. In CA-WuR, when the traffic load is heavy, there is a less probability to sense the channel idle and therefore the average retry time of the backoff increases before sending WuCs.



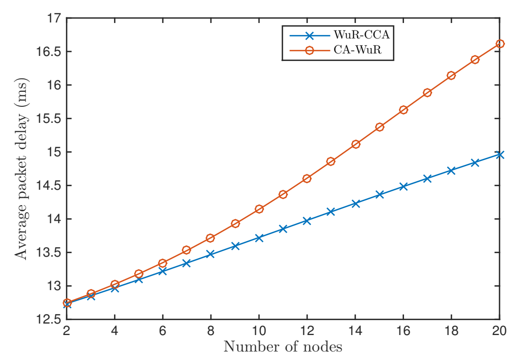
(a) $\lambda = 0.2$



(b) $\lambda = 0.5$



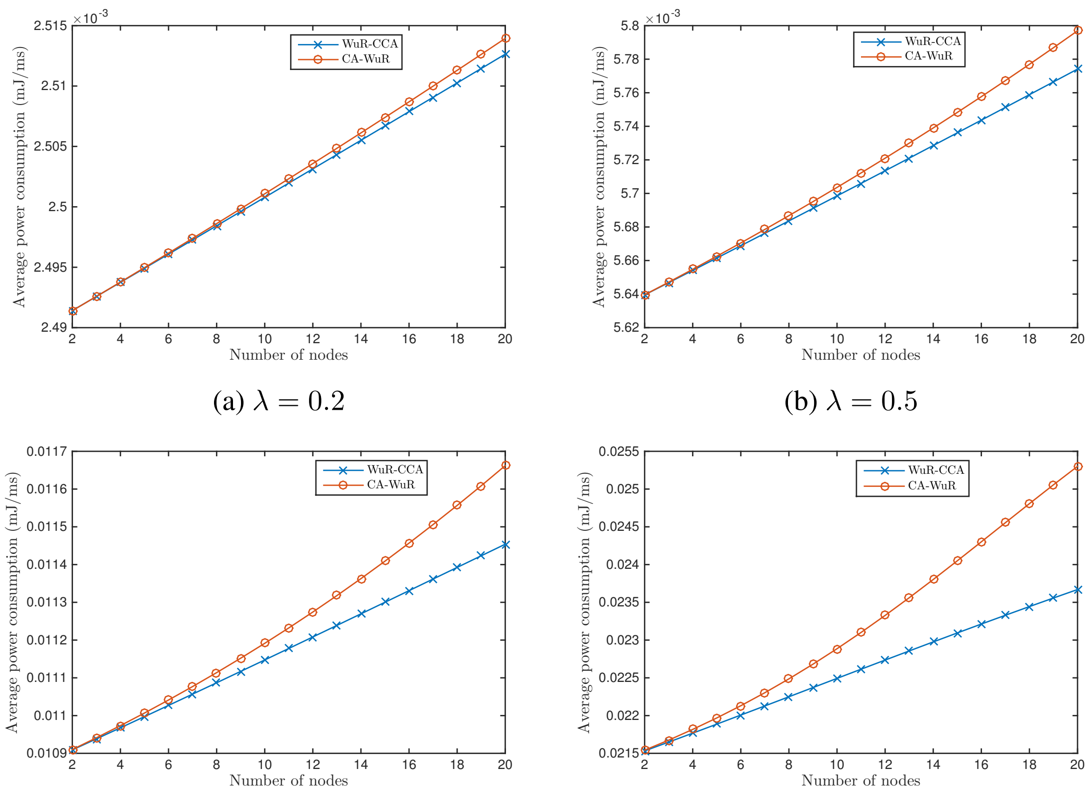
(c) $\lambda = 1$



(d) $\lambda = 2$

Figure 4.11: Average packet delay of RN at various traffic loads.

The average power consumption of RN at various traffic loads are shown in Figure 4.12. From the figure we can see that CA-WuR has higher power consumption than that of WuR-CCA. This is because more energy is consumed at the backoff procedure in CA-WuR, compared with WuR-CCA. On the other hand, when the number of node increases from 2 to 20, the average power consumption only increased by up to 0.004 mJ/ms. This indicates that the average energy consumption basically does not depend on the number of devices.



(a) $\lambda = 0.2$ (b) $\lambda = 0.5$

(c) $\lambda = 1$ (d) $\lambda = 2$

Figure 4.12: Average power consumption of RN at various traffic loads

4.5 Chapter Summary

In this chapter, CA-WuR has been proposed with detailed descriptions. The performance is analysed using a developed DTMC model and compared with a WuR protocol that only performs CCA before sending WuCs. Numerical results indicate that our proposed protocol always has better performance on the PDR and network throughput at various traffic loads. The results reveal at the same time that CA-WuR has longer packet delay and higher power consumption, compared with WuR-CCA.

Chapter 5

EHA-WuR Protocol: A Solution for Energy Hole Problem in WSNs

In this chapter, we propose another WuR protocol with energy hole avoidance (EHA-WuR) in a multipoint-to-point WSN scenario. Three operation modes of EHA-WuR will be described in detail. In addition, the implementation of the proposed protocol in Omnet++ is illustrated. By comparing the performance of EHA-WuR with traditional hop-by-hop operation mode of WuR protocols, we prove our designed protocol is able to avoid the energy hole problem in a multipoint-to-point WSN scenario.

5.1 Related Background and the Network Scenario

5.1.1 Background on WuC and data transmission ranges

The main disadvantage of the state-of-the-art WuR platforms is their short WuC transmission ranges. The transmission range of Zigbee product that used for data transmission has been measured to reach 70-100 meters at the output power of 0 dBm. However, most out-of-band WuR can only achieve a distance of several meters. Even for the in-band WuR, in which the WuC is sent via main radio, the transmission range varies from 20 to 40 meters. For example, the SCW-WuR hardware platform, which has been discussed in the previous chapters, is an in-band WuR system and has been tested achieving a WuC range of 24 meters at the output power of 0 dBm.

Since a higher output power leads to higher energy consumption, the most energy efficient way is to enable different output power for WuC and data transmissions so that their transmission ranges are basiclly the same. This is a premise of our designed EHA-WuR. For example, when using a SCW-WuR, if a node has a WuC range of 24 meters at 0 dBm, the output power of the data packet can be configured at a lower value, (e.g., -10 dBm), which only needs to be enough for reaching a transmission range of 24 meters.

5.1.2 Network Scenario

Clustering is an effective and practical way to enhance the system performance of WSNs. In our study, we simplify the clustered WSN scenario by considering a single cluster with multipoint-to-point transmission in a WuR-enabled WSN, as shown in Figure 5.1. One node (Node A) in the network performs as the sink node and all other child nodes (Node C-H) in the network transmit data to the sink via a coordinator node (Node B). All the nodes in the network are WuR-enabled. It is assumed that the output power of the WuC and data are able to be configured differently and can be adjusted. Initially, all the child nodes and the coordinator node are in the same WuC and data transmission ranges, while the sink node is only in the range of the coordinator node. In the network, data packets arrive at each child node according to a Poisson distribution with an arrival rate of λ. The coordinator node B doesn't generate traffic itself.

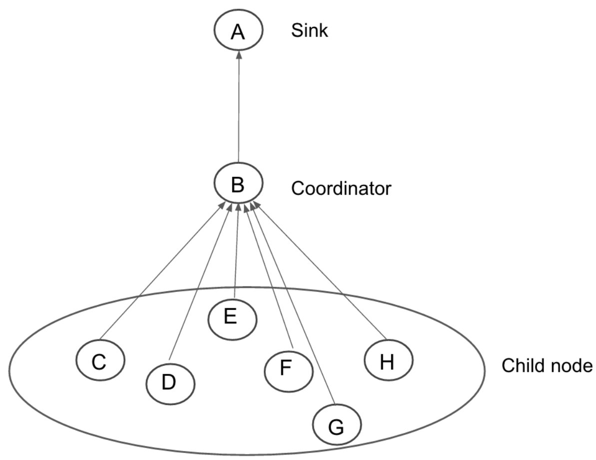


Figure 5.1: Network topology with multipoint-to-point transmission.

5.2 EHA-WuR Design

In our designed EHA-WuR protocol, the first hop WuC transmissions are based on the principle of CA-WuR, in which a contention-based collision avoidance procedure will be performed before WuC transmission.

5.2.1 Preliminary: traditional hop-by-hop operation mode

Traditional hop-by-hop operation mode is used in many state-of-the-art WuR protocols. When sending data packet to the sink that is multiple hops away, both of the WuC and data packets will pass through each middle-node. For the network topology shown in Figure 5.1, when the child node C intends to send data to the Node A, it will first wake up Node B by sending B a WuC. Then it will again send its data packet to Node B. Node B will then repeat the same procedure by sending a WuC to Node A and forward the data packet. The principle of this traditional hop-by-hop operation mode is shown in Figure 5.2.

Since Node B forwards all the packets from its child nodes, it would consume energy at a high rate and deplete early. Therefore, Node B becomes an energy hole of the network. No matter how much residual energy is left in the rest of the nodes, the network becomes disconnected due to this

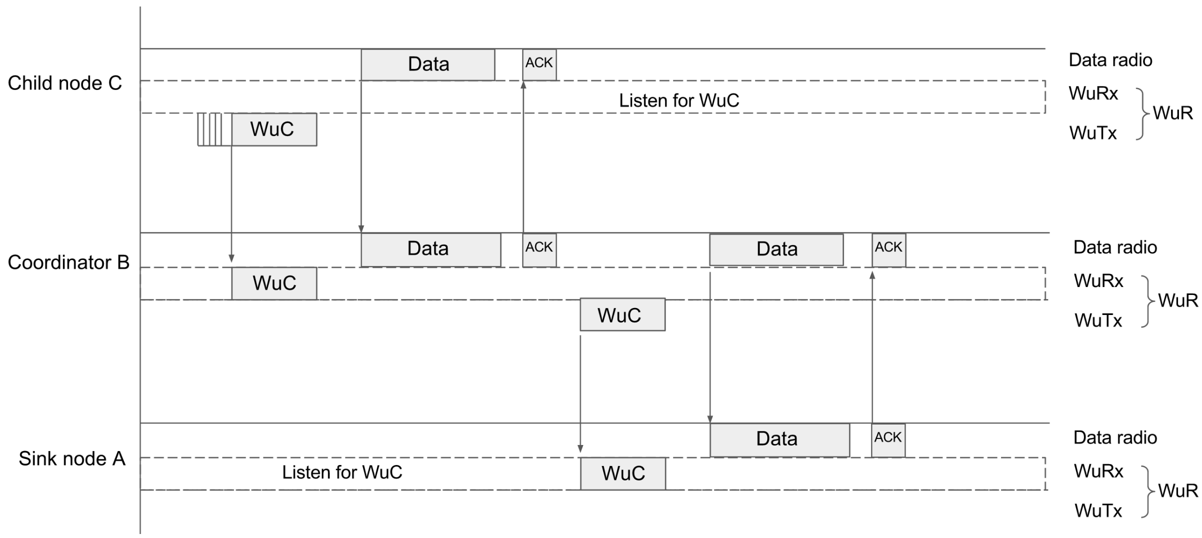


Figure 5.2: Principle of traditional hop-by-hop operation mode

energy hole.

5.2.2 EHA-WuR: operation mode 1

To diminish the energy hole problem that appears in the traditional hop-by-hop mode, it is important to release the traffic burden on the coordinator node. In EHA-WuR operation mode 1, the coordinator node only forward WuCs, while the data packets of the child nodes are sent directly to the sink, without passed through Node B. Therefore, the transmission range of data packets is twice that of the WuCs by configuring a higher output power in Mode 1.

As shown in Figure 5.3, when Node C intends to send a packet to the sink, it first performs the contention procedure for the channel access. Upon succeeding, it sends a WuC to Node B. Then Node B will wake up and forward the WuC to the sink node without performing contention. When finishing WuC transmission, Node B will turn off the main radio and go back to sleep mode. Meanwhile, Node C delays for a duration until Node A is waked up by Node B. Since Node B does not perform contention procedure, this delay time of Node C is a constant duration ($2T_{on} + T_{wuc}$). After the delay, Node C will send data packet directly to the sink node.

In this case, the coordinator node B simply acts as a WuC-relay that only forwards WuCs. No

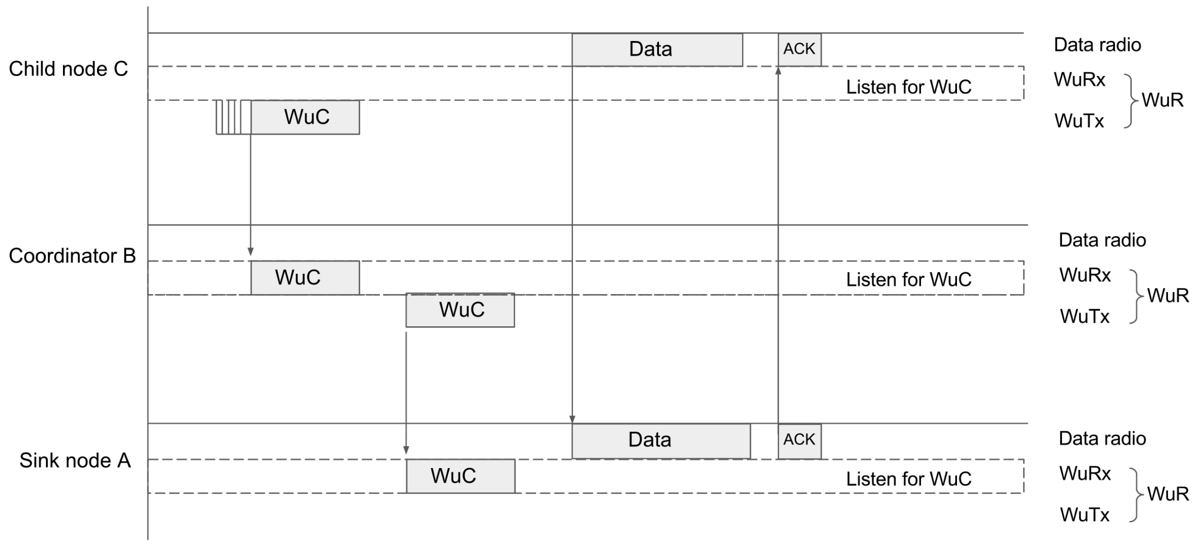


Figure 5.3: Principle of EHA-WuR at operation mode 1.

data will pass through Node B and the energy consumption of Node B is reduced.

5.2.3 EHA-WuR: operation mode 2

Although the first operation mode releases the traffic burden on the coordinator node compared to the traditional operation mode, the energy hole problem is not completely resolved. The coordinator node may still end up as an energy hole due to extensive WuC forwarding. To further diminish the energy hole problem, we propose anther operation mode that enables long transmission range of WuCs by increasing the WuC output power for the child nodes. In this case, both of the WuCs and the data packets can be sent directly to the sink node and the coordinator node is not required anymore.

As what has been introduced in Chapter 2, the hardware of SCM-WuR is designed to cover the transmission range up to 100 meters using an incorporated RF front-end [5] at the transmitter to enable transmission power of up to +20 dBm. This design makes it possible for long transmission range of WuCs. However, the required current for WuC transmission is much higher than that of the short range. In Chapter 3, we have concluded that enabling long range WuC is not an energy efficient option by numerical results. Therefore, the EHA-WuR is only proposed to be used in the scenario where the coordinator has exhausted its energy. Although the long range WuC is not energy efficient, it gives benefit to the network lifetime by skipping the energy hole node.

5.2.4 EHA-WuR: the hybrid operation mode

A hybrid operation mode of EHA-WuR enables both Mode 1 and Mode 2. At the hybrid mode, the network is configured at Mode 1 initially. When the residential energy is detected less than a certain value e, Node B will broadcast a beacon message via the same frequency band as WuC. The collision avoidance mechanism is not implemented for beacon transmissions since the beacon frame might be dropped during the contention procedure. As what explained in Chapter 4, after each successful transmission, the channel will remain idle for at least a duration of $T_{cca} + T_{sifs}$. Therefore, when detected the residential energy is less than e, the coordinator node will wait until the end of the next data transmission, as shown in Figure 5.4. Note that the value of e should be configured enough for the coordinator node remaining alive for a period of time, as well as performing a WuC forwarding process and a beacon transmission.

Upon receiving the beacon message, the child nodes will switch their operation mode to Mode 2 and increasing the WuC output power. From then on, all the WuCs and data packets will be sent directly to the sink.

The frame structure of the beacon message is designed the same as the WuC, which is presented in Figure 4.3. Since the beacon message contains a broadcast address, it can be distinguished from WuC frames.

5.3 EHA-WuR Protocol Implementation

The designed EHA-WuR protocol with at hybrid operation mode is implemented in Omnet++ simulator for the performance evaluation purpose. In the implementation, we follow the hardware design of SCW-WuR. Retransmission is disabled in our implementation.

Corresponding to different role of nodes, the implementation of EHA-WuR can be divided into three categories based on their roles and behaviours: the child node implementation, the coordinator

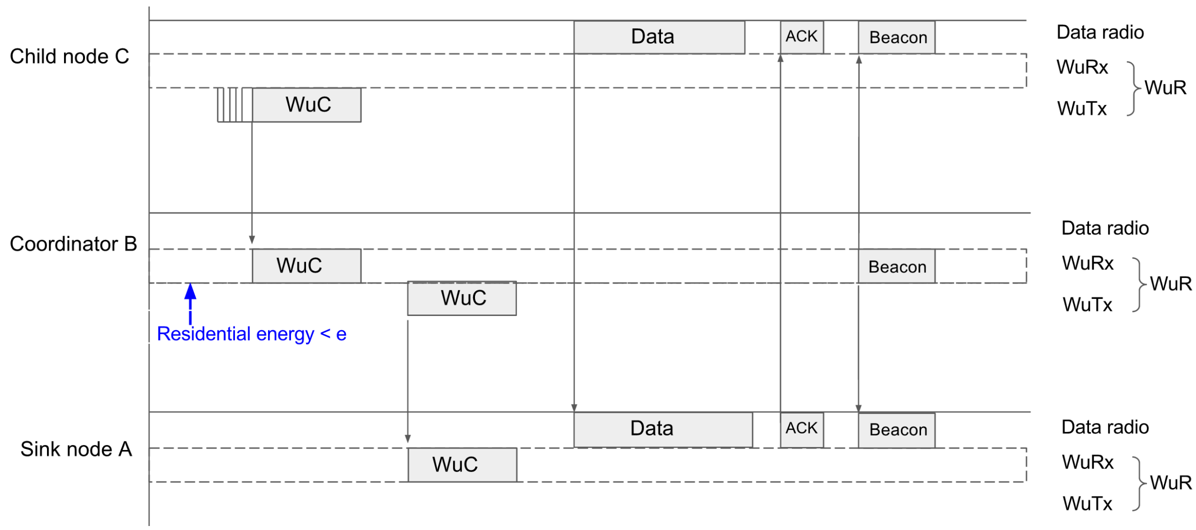


Figure 5.4: Principle of EHA-WuR at the hybrid operation mode.

node implementation and the sink node implementation.

5.3.1 Omnet++ implementation framework

An OMNeT++ model consists of hierarchically nested modules that communicate by passing messages to each other. A network is built as a *compound module* in OMNeT++, which is grouped by multiple *simple modules* or *submodules*. Messages can be sent either via connections that span modules or directly to other modules. Figure 5.5 shows the structure of a single WuR-enabled sensor node in our implementation.

The components "appl" and "netwl" in the left side of the figure represent the submodules which define the behaviour of the application layer and network layer, respectively. The submodule "dual" defines the message control functions of EHA-WuR protocol. Module "nic" and "wur_nic" contains the submodule which defines the behaviour of a node's main transceiver and WuR transceiver, respectively. The main radio is controlled by a "controller" module, which acts as an MCU that takes charge of switching the main radio on and off. The rest of the modules: "mobility", "arp", "battery" and "battery state" defines the network mobility, address resolution, battery consumption and battery monitoring, respectively. The main functions of the implementation on the protocol

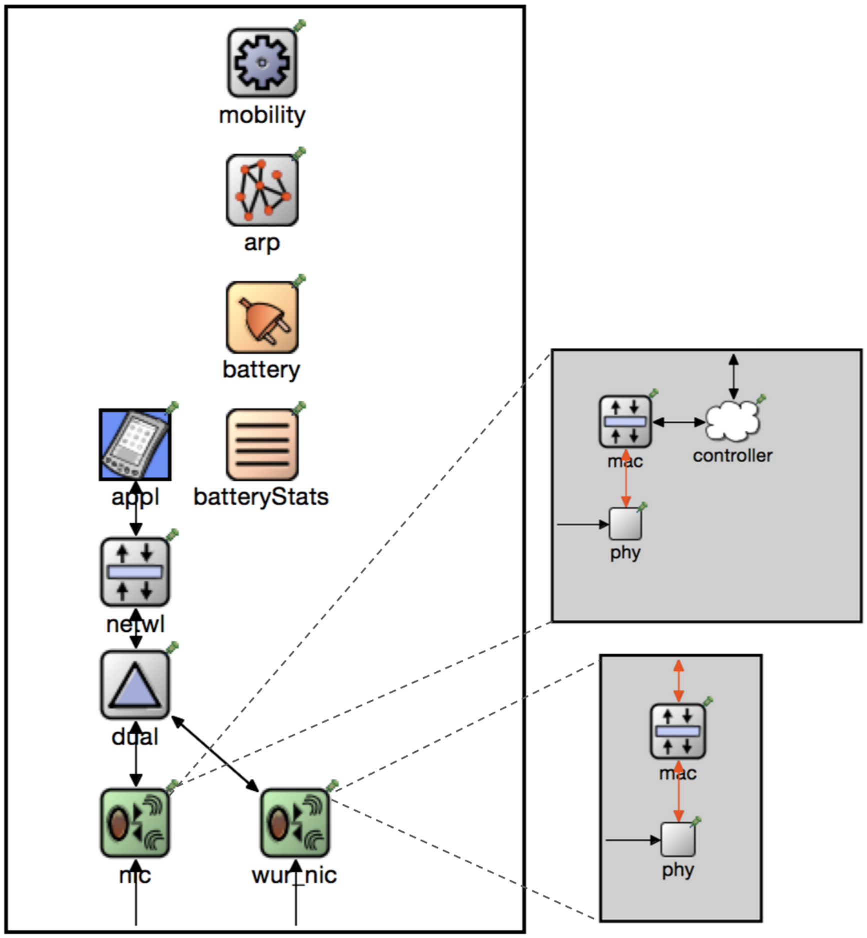


Figure 5.5: The simulation module structure of a single sensor node.

principle are presented in Appendix B.

5.3.2 Protocol implementation of the child node.

Figure 5.6 presents the flow chart of the child node implementation. After initiation process, the WuRx will keep listening to the channel for incoming WuCs or beacon message. Since the node initially operates at Mode 1, the parameter $Mode\_state$ is default as 1. When receiving a beacon message, $Mode\_state$ will be set to 2, and the WuC output power will be increased to new_power, which enables the WuCs to be sent directly to the sink.

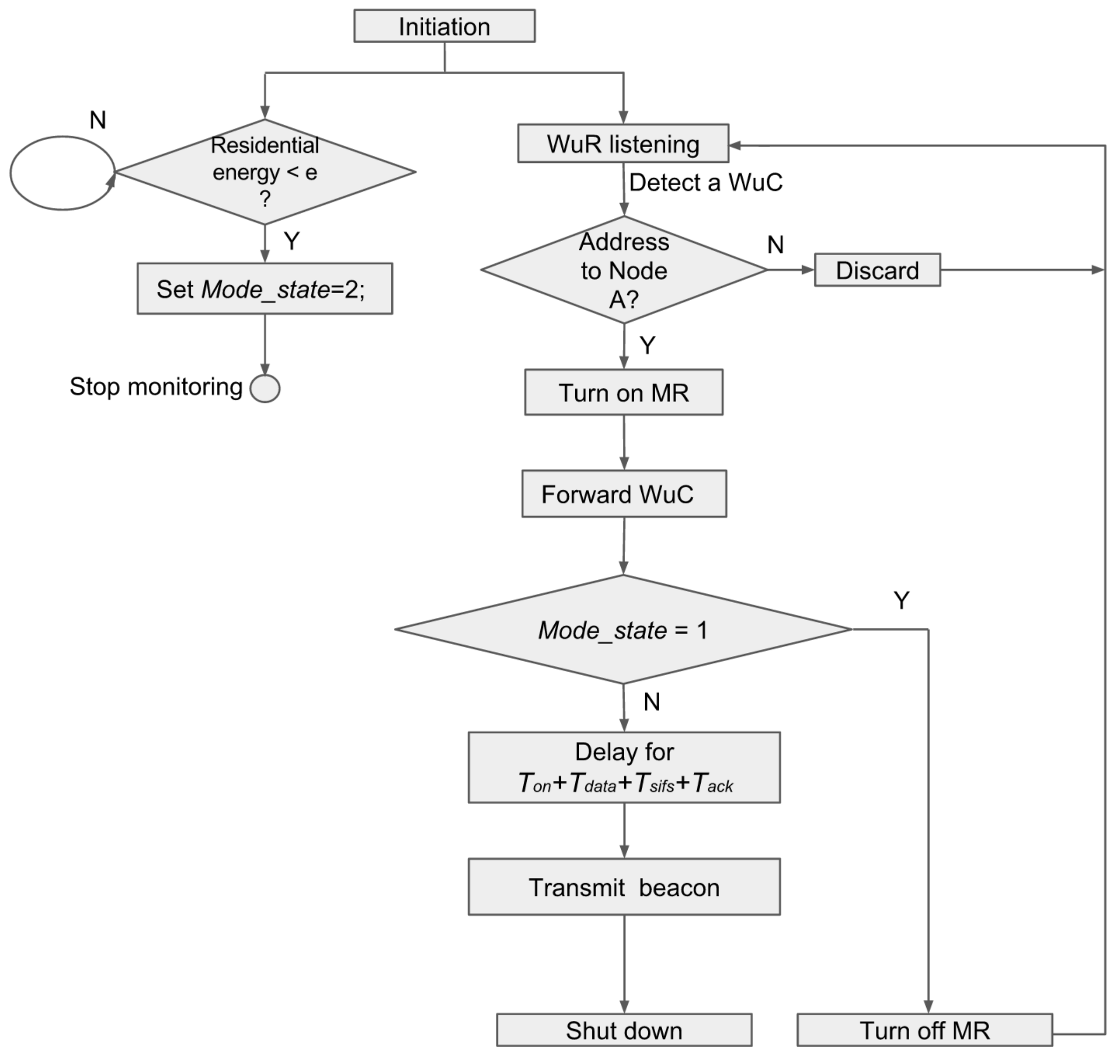


Figure 5.6: Flow chart of the child node implementation.

- When the child node is in Mode 1 and has a packet to sent, it turns on the main radio and performs contention procedure. The WuC will be sent if the node wins the contention. After sending WuC, it will delay for a $2T_{on} + T_{wuc}$ duration, during which the sink node will be waked up by the coordinator node. The data packet will be sent directly to the sink node after the delay time. When the transmission ended, the child node will turn off its main radio and again start to listen for WuCs on its WuRx.

- When the child node is in Mode 2, the WuCs are sent directly to the sink node with output power of new_power. The delay time that waits for the sink waking up becomes T_{on}.

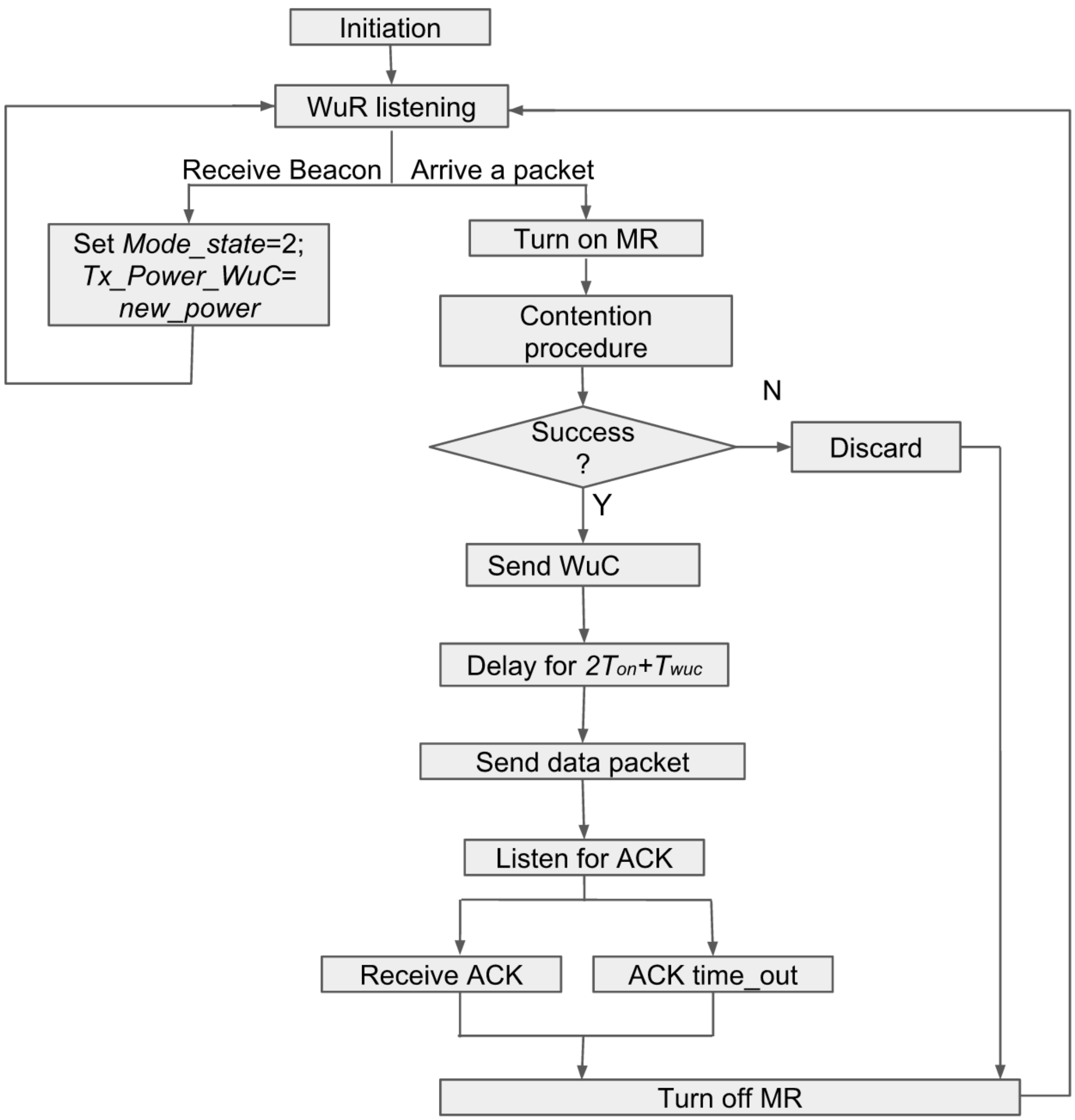


Figure 5.7: Flow chart of the coordinator node implementation.

5.3.3 Protocol implementation of the coordinator node

Figure 5.7 presents the flow chart of the coordinator node. After initiation process, the node will keep monitoring the residential energy of its battery. If the residential energy is less than e, the value of $Mode\_state$ will be set to 2. When the WuRx detected a WuC which is addressed to the sink node, the node will turn on its main radio and forward the WuC.

After the WuC forwarding, the coordinator node will check the value of $Mode\_state$. When the value is checked to be 2, the node will delay for a duration of $T_{on} + T_{data} + T_{sifs} + T_{ack}$, during which the child node will finish its data transmission. Then the coordinator node will broadcast a beacon message and shut down itself.

5.3.4 Protocol implementation of the sink node

Figure 5.8 presents the flow chart of the sink node implementation. As presented in the figure, the behaviour of the sink node is not influenced by the operation mode. After initiation process, the WuRx keeps listening to the channel for incoming WuCs. Upon detecting a WuC that is addressed to itself, the sink node will turn on its main radio and listen for the data packet. After received the data packet, it will send an ACK back and then turn off its main radio, leaving the WuRx listening to the channel again.

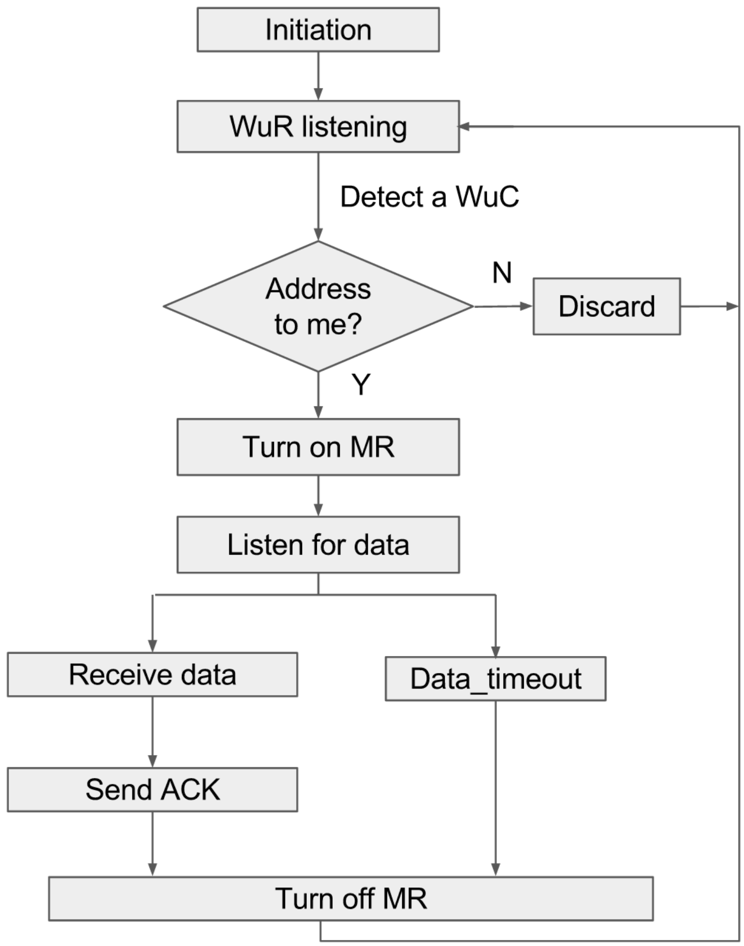


Figure 5.8: Flow chart of the sink node implementation.

5.4 Numerical Results and Discussions

5.4.1 Network parameter configuration

The related parameter configurations are shown in Table 5.1[1] and the rest of the network parameters refer to Table A.1. The network lifetime is defined as the time that the first child node drains its energy. The network topology is the same as shown in Figure fig:topology. In our testing scenario,

[1]Note that we use -6 dBm as the short range data output power because it is the smallest output power listed in the CC1101 data sheet [10].

Table 5.1: Parameter configurations for the test scenarios

Parameter	Short range ($\leq$ 24 m)	Long range ($\leq$ 70 m)	Unit
Data output power	-6	0	dBm
Data transmission current	16.4	16.8	mW
Data frame length	100	100	Byte
Main radio data rate	250	250	Kbps
WuC output power	0	+20	dBm
WuC transmission current	16.8	152	mW
WuC frame length	48	48	bit
WuC data rate	8	8	Kbps

there is always one sink node (Node) A and one coordinator node (Node B). The number of the child nodes in the network and the packet arrival rate varies.

5.4.2 Numerical results and discussions at different test scenarios

Test scenario 1: varied number of child node

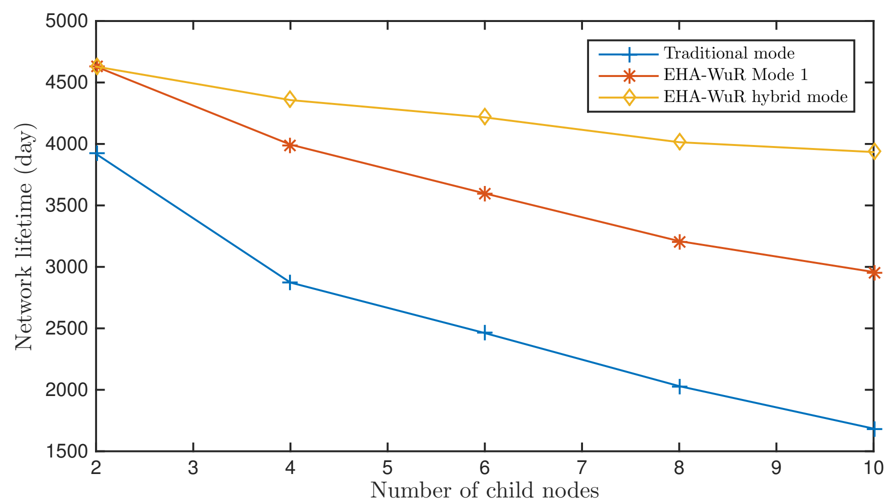


Figure 5.9: Network lifetime with different number of child node. $\lambda = 0.01$.

Figure 5.9 shows the network lifetime at the scenario with different number of child nodes with the packet arrival rate $\lambda = 0.01$. The results of three operation modes are plotted in the figure including the traditional hop-by-hop operation mode, EHA-WuR Mode1 and EHA-WuR hybrid mode. From the figure we can see that the in traditional mode, the network achieves a shortest network lifetime. Compared with the traditional mode, our proposed MCA-WuR Mode 1 prolongs the overall network lifetime by around 1000 days. Meanwhile, in EHA-WuR hybrid mode, the lifetime is further increased by around 500 days compared with EHA-WuR Mode 1.

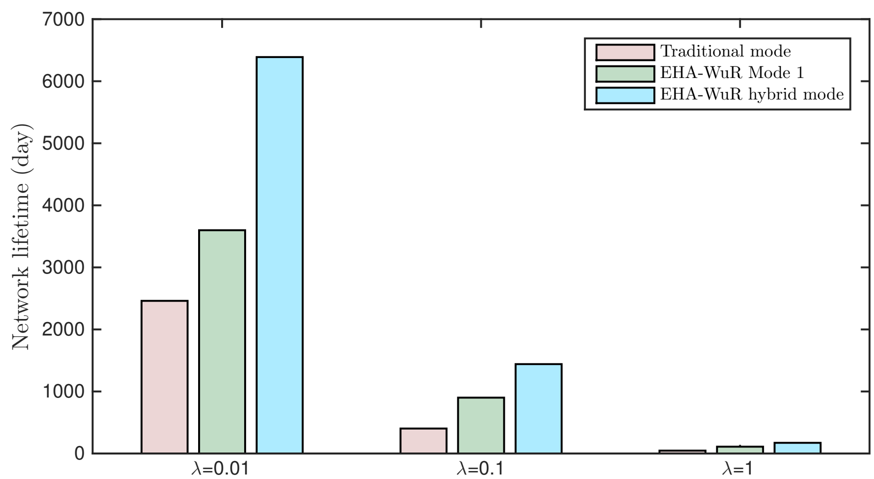


Figure 5.10: Network lifetime at various traffic load.

Figure 5.10 shows the network lifetime at various traffic loads when there are 6 child nodes in the network. The results prove that EHA-WuR Mode1 and hybrid mode always achieves longer network lifetime. The lifetime of EHA-WuR hybrid mode is twice that of the traditional mode.

Test scenario 2: number of child node=2, $\lambda = 0.01$

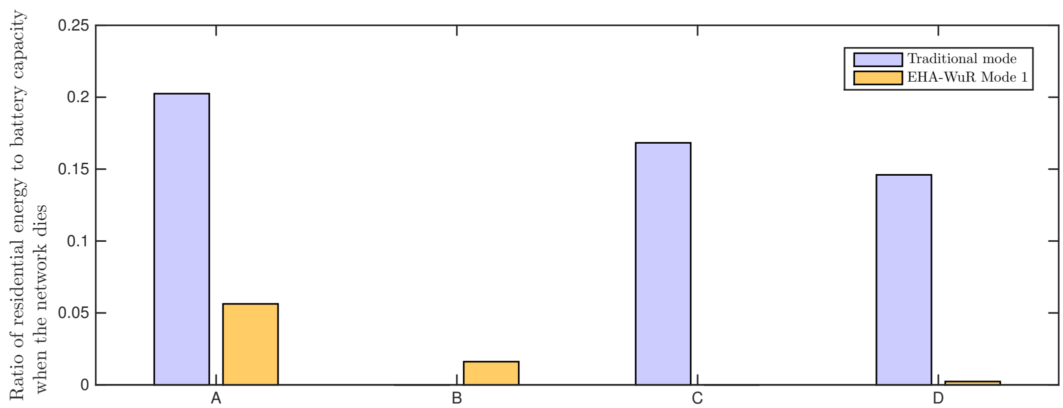


Figure 5.11: Percentage of residential energy to the battery capacity of all the nodes when the network dies. Number of the child nodes=2.

Figure 5.11 presents the percentage of the residential energy to the battery capacity of all the nodes when the network dies. As what shown in the figure, at the traditional mode, the coordinator node B is the energy hole that causing the network disconnected. When Node B or the network dies, there are more than 15% percent of energy left in other nodes. When implementing the EHA-WuR Mode 1, it is the child node C who dies first in the network. Both of the child nodes (Node C and Node D) have almost used up their energy and there is only a small percentage of energy left in the coordinator and sink node. In this case, EHA-WuR Mode 1 successfully avoids the energy hole problem and the EHA-WuR hybrid mode is not needed.

Figure 5.12 presents the average power consumption of all the nodes. The power consumption of the coordinator node B at EHA-WuR Mode 1 is lower than that of the traditional mode. While other nodes have almost same level of power consumption at both traditional mode and EHA-WuR Mode 1. The reason is that we use CC1101 transceiver for our implementation and the current required for the output power of -6 dBm is very close to that of 0 dBm (i.e., 16.4 mA and 16.8 mA, respectively), as shown in Table 5.1.

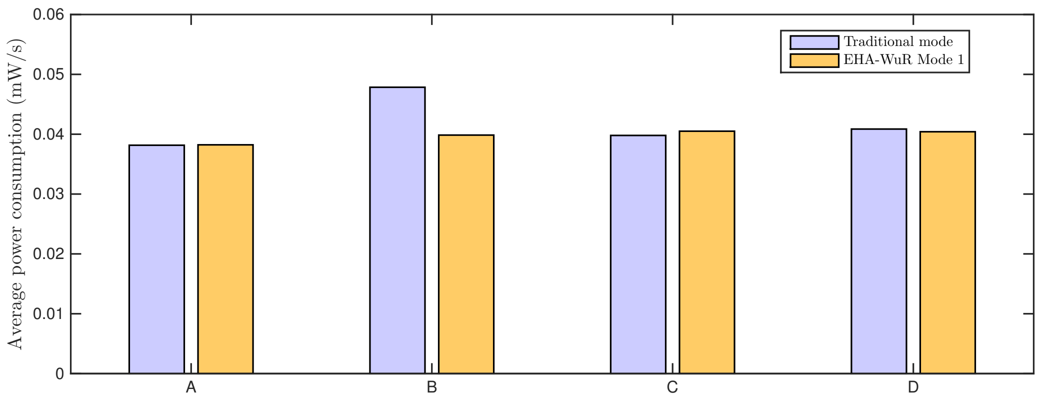


Figure 5.12: Average power consumption of all the nodes. Number of the child nodes=2.

Test scenario 3: number of child node=4, $\lambda = 0.01$

Figure 5.13 illustrates the percentage of the residential energy to the battery capacity of all the nodes when the network dies. Compared the result with that of the test scenario 2, the energy hole problem becomes more serious at the traditional mode since more energy left at the nodes when the network dies. As shown in Figure 5.13, in EHA-WuR Mode 1, although the residential energy (around 10%) is less than that of the traditional mode (around 30%), Node B drains its energy before other nodes. The energy hole problem is still unsolved completely at EHA-WuR Mode 1 in this test scenario. Therefore, EHA-WuR hybrid mode is required. As shown in the figure, in EHA-WuR hybrid mode, there is almost no energy left in the network when Node C dies and nearly all the energy has been taken advantage.

The power consumption of all the nodes at three operation modes are presented in Figure 5.14. The traditional mode and EHA-WuR Mode 1 have similar performance at test scenario 2. While in EHA-WuR Mode 2, the nodes have extremely high power consumption due to their high WuC transmission current. This is also the reason that the nodes in EHA-WuR hybrid mode have higher power consumption than that of EHA-WuR Mode 1.

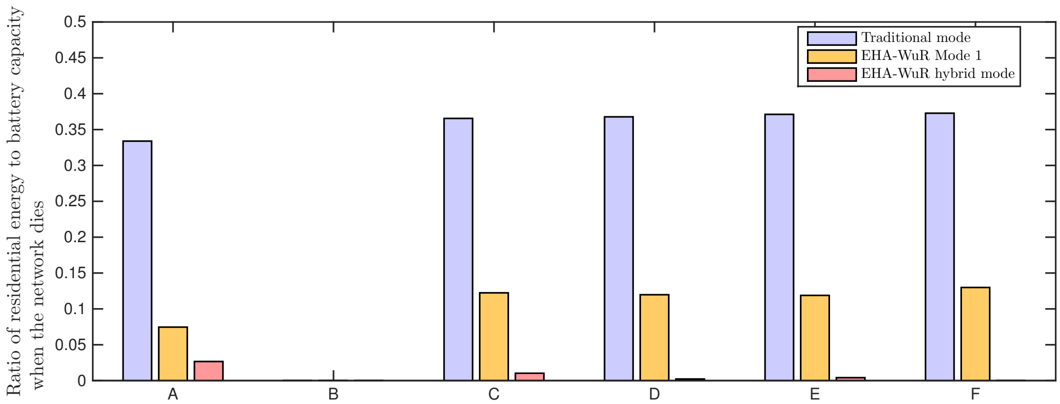


Figure 5.13: Percentage of residential energy to the battery capacity of all the nodes when the network dies. Number of the child nodes=4.

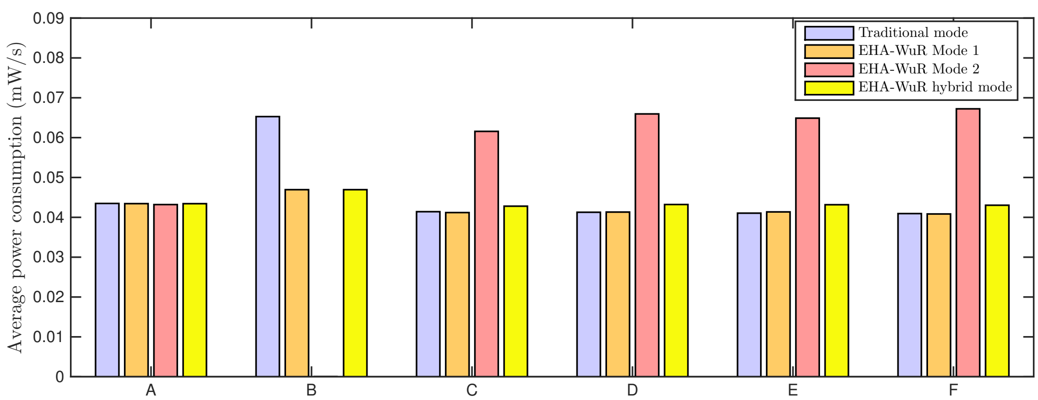


Figure 5.14: Average power consumption of all the nodes. Number of the child nodes=4.

Test scenario 4: number of child node=6 and 8, $\lambda = 0.01$

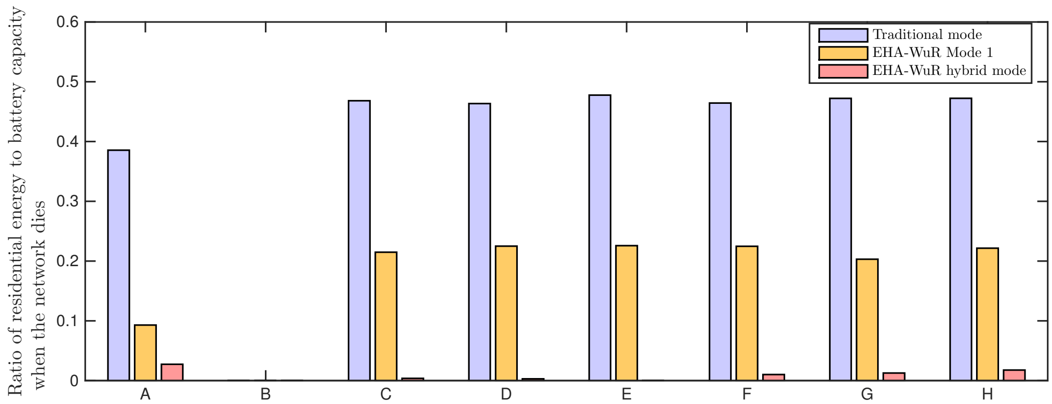


Figure 5.15: Percentage of residential energy to the battery capacity of all the nodes when the network dies.The number of child nodes=6.

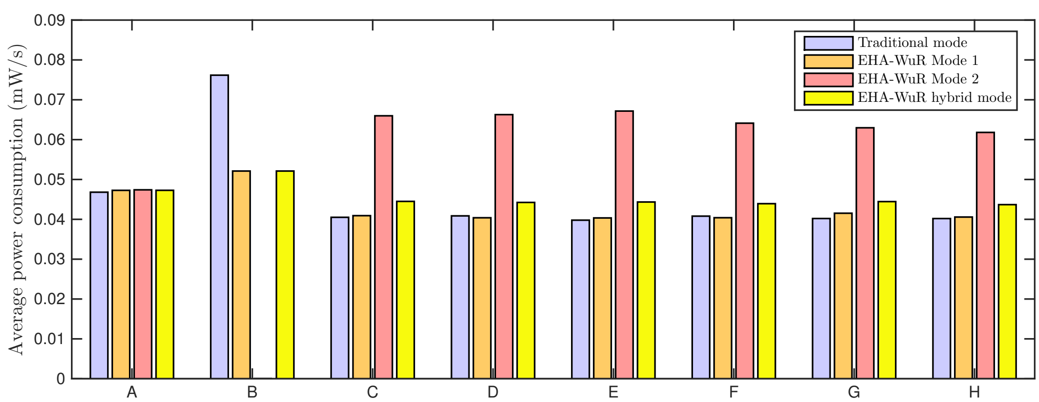


Figure 5.16: Average power consumption of all the nodes. The number of child nodes=6.

Figure 5.15 - 5.18 presents the percentage of the residential energy to the battery capacity and the average power consumption of all the nodes when there are 6 and 8 child nodes in the network. From the figures we can see that in the traditional mode, around half of the energy is left when the network dies. EHA-WuR hybrid mode always has the best performance among three modes, and is always capable of avoiding energy hole in the network.

When comparing the results of the four figures as well as the previous results, we conclude that in the traditional mode and EHA-WuR Mode 1, when the number of child node increases, the

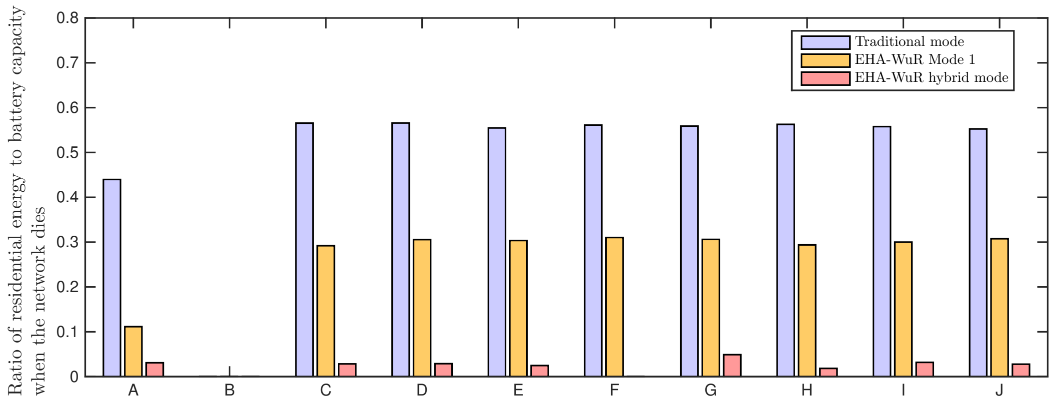


Figure 5.17: Percentage of residential energy to the battery capacity of all the nodes when the network dies. The number of child nodes=8.

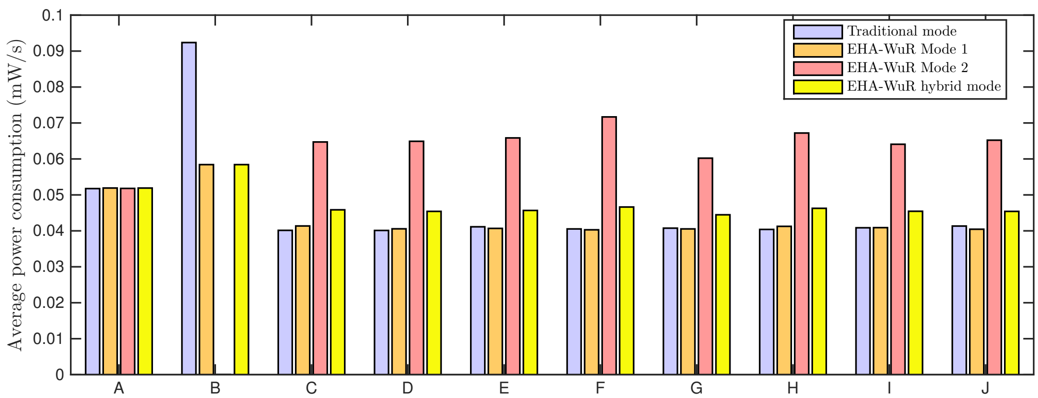


Figure 5.18: Average power consumption of all the nodes. The number of child nodes=8.

residential energy of the network increases. This is due to the heavier burden of the coordinator node since there is more child node generating traffic. We also conclude that although the average power consumption at the EHA-WuR hybrid mode is always higher than the other two modes, it always achieves better performance on network lifetime.

5.5 Chapter Summary

In this chapter, we proposed EHA-WuR protocol with energy hole avoidance in a multipoint-to-point WSN scenario. The design and implementation of the protocol have been presented in detail. The performance of EHA-WuR was evaluated from the simulation results. Numerical results indicate that our proposed EHA-WuR protocol is capable of avoiding energy hole problem in a multipoint-to-point WSN. By comparing its performance with traditional hop-by-hop operation mode, we proved our designed protocol achieves longer network lifetime. In addition, we also concluded that although the average power consumption at the EHA-WuR hybrid mode is higher than EHA-WuR Mode 1, it achieves longer network lifetime and balances the energy consumption of the network.

Chapter 6

Conclusions and Future Work

After extensive in-depth analysis and the simulation implementation of WuR, we now present the conclusions on our thesis work in this chapter. The main points of our study are summarized in the conclusions, including the energy analysis and comparison of three state-of-the-art MAC protocols in WSNs, the design and analysis of a contention-based WuR-enabled MAC protocol with collision avoidance, and the proposal and Omnet++ implementation of a WuR-enabled MAC protocol with energy hole avoidance. In addition, the contributions made by this thesis are presented in this chapter as well as the proposed future work.

6.1 Conclusions

This thesis focuses on the WuR technique, analysing its performance in WSNs and proposing two MAC layer protocols that could improve the network performance in the aspect of PDR and network lifetime. For achieving this goal, the existing WuR-enabled MAC protocols and WuR hardware platforms have been investigated and studied. Three research questions are proposed at the beginning of the thesis and the corresponding solutions are proposed and presented.

As a starting point, an existing WuR-enable protocol (SCW-WuR) has been analysed, together with two typical DC protocols. By comparing their energy performance at various traffic load conditions, we ascertained that SCW-WuR is able to achieve an excellent energy performance at a low traffic load, but when the traffic load gets heavy, WuR will lose its advantage on the aspect of energy saving. In addition, we also concluded that it may not be beneficial to enable long range WuCs with respect to energy efficiency.

Although WuR is suitable to the low traffic WSNs, the traffic can get heavy in some type of event-driven WSNs applications such as fire detection. It is necessary to enable a collision avoidance mechanism to WuC for WuR-enabled protocol in WSNs. Therefore, we proposed CA-WuR protocol with collision avoidance. The novel point of this proposal is to apply an unslotted collision avoidance mechanism to WuC transmissions. The performance of CA-WuR was evaluated by a DTMC model and numerical results indicate that our proposed protocol has better performance on the PDR and network throughput, with the cost of slightly longer delay, compared with an existing WuR protocol.

In addition, we proposed EHA-WuR protocol with energy hole avoidance, which contains three operation modes. The proposed protocol has been implemented in Omnet++ simulator. The performance of EHA-WuR protocol is evaluated and compared with a traditional hop-by-hop operation mode. We proved our designed protocol is capable of avoiding the energy hole problem in a multipoint-to-point WSN scenario. We also concluded that EHA-WuR protocol achieves longer network lifetime than that of the traditional mode.

6.2 Contributions

The main contributions of this thesis are summarized below.

- The question whether WuR always leads to lower energy consumption is answered by comparing an existing WuR-enabled protocol with two representative DC protocols.
- A contention-based WuR-enabled MAC protocol with collision avoidance is proposed. The performance of the proposed protocol is evaluated by a developed DTMC model. Better performance on PDR of the proposed protocol is demonstrated.
- A WuR-enabled MAC protocol with energy hole avoidance is proposed for multipoint-to-point transmissions in WSNs. The proposed protocol is implemented in Omnet++ simulator

and its performance is proved achieving longer lifetime than that of the traditional operation mode.

6.3 Future Work

Beyond the research questions and analysis that we have discussed in this thesis work, there are several possible directions.

- Our analysis of the existing WuR protocols is mainly based on the energy consumption. An overall performance of WuR on more aspects, such as packet delay, PDR and network throughput is remained to be further analysed, comparing with synchronous and asynchronous DC MAC protocols.
- Our designed MCA-WuR is only suitable for specific small-scaled network topology. A WuR-enabled protocol that fits large-scaled network, regardless the network protocol can be further designed. A cross-layer protocol that between the network layer and the MAC layer could be a direction for achieving this goal.

Bibliography

[1] J. Ansari, D. Pankin, and P. Mähönen, "Radio-triggered wake-ups with addressing capabilities for extremely low power sensor network applications," *International Journal of Wireless Information Networks*, vol. 16, no. 3, p. 118, 2009.

[2] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Proc. 4th Int. Conf. Embedded Netw. Sensor Syst.*, 2006, pp. 307-320.

[3] T. R. Burchfield, S. Venkatesan, and D. Weiner, "Maximizing throughput in zigbee wireless networks through analysis, simulations and implementations," in *Proc. 1st Int. Workshop Localized Algorithms Protocols Wireless Sens. Netw.*, Santa Fe, NM, Jun. 18-20, 2007.

[4] J. Da Silva, J. Shamberger, M. J. Ammer, C. Guo, S. Li, R. Shah, T. Tuan, M. Sheets, J. M. Rabaey, B. Nikolic, *et al.*, "Design methodology for picoradio networks," in *Proc. Design, Automation and Test In Europe Conf. Exhib.*, 2001, pp.314-323.

[5] M. Dhanaraj, B. Manoj, and C. S. R. Murthy, "A new energy efficient protocol for minimizing multi-hop latency in wireless sensor networks," in *Proc. 3rd IEEE PERCOM*, Washington, DC, 2005, pp. 117-26.

[6] F. Z. Djiroun and D. Djenouri, "Mac protocols with wake-up radio for wireless sensor networks: A review," *IEEE Communications Surveys & Tutorials*, Sep, 2016.

[7] G. U. Gamm and L. M. Reindl, "Range extension for wireless wake-up receivers," in *Proc. 9th SSD*, 2012, pp. 1-4.

[8] D. Ghose and F. Y. Li, "Enabling backoff for SCM wake-up radio: Protocol and modeling," *IEEE Communications Letters*, 2017.

[9] M. Gribaudo, D. Manini, A. Nordio, and C.-F. Chiasserini, "Transient analysis of IEEE 802.15. 4 sensor networks," *IEEE Transactions on wireless communications*, vol. 10, no. 4, pp. 1165-1175, 2011.

[10] C. Guo, L. C. Zhong, and J. M. Rabaey, "Low power distributed MAC for ad hoc sensor radio networks," in *Proc. IEEE GlobeCom*, 2001.

[11] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *ACM SIGOPS operating systems review*, vol. 34, no. 5, pp. 93-104, 2000.

[12] V. Jelicic, M. Magno, D. Brunelli, V. Bilas, and L. Benini, "Benefits of wake-up radio in energy-efficient multimodal surveillance wireless sensor network," *IEEE Sensors Journal*, vol. 14, no. 9, pp. 3210-3220, 2014.

[13] H. Jiao, M. A. Ingram, and F. Y. Li, "A cooperative lifetime extension MAC protocol in duty cycle enabled wireless sensor networks," in *MILCOM*, 2011, pp. 896-901.

[14] J. W. Jung and M. A. Ingram, "Residual-energy-activated cooperative transmission (react) to avoid the energy hole," in *IEEE ICC*, 2010, pp. 1-5.

[15] T. O. Kim, J. S. Park, K. J. Kim, and B. D. Choi, "Performance analysis of IEEE 802.15. 4 non-beacon mode with both uplink and downlink traffic in non-saturated condition," in *Mobile Lightweight Wireless Systems*, 2009, pp. 357-371.

[16] P. Le-Huy and S. Roy, "Low-power 2.4 ghz wake-up radio for wireless sensor networks," in *IEEE/WIMOB on Wireless and Mobile Computing*, 2008, pp. 13-18.

[17] J. Lian, K. Naik, and G. B. Agnew, "Data capacity improvement of wireless sensor networks using non-uniform sensor distribution," *International Journal of Distributed Sensor Networks*, vol. 2, no. 2, pp. 121-145, 2006.

[18] M. Magno, V. Jelicic, B. Srbinovski, V. Bilas, E. Popovici, and L. Benini, "Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 2, pp. 633-644, 2016.

[19] W. Nosovic and T. D. Todd, "Scheduled rendezvous and rfid wakeup in embedded wireless networks," in *IEEE ICC*, vol. 5, 2002, pp. 3325-3329.

[20] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl, "Performance evaluation and comparative analysis of subcarrier modulation wake-up radio systems for energy-efficient wireless sensor networks," *Sensors*, vol. 14, no. 1, pp. 22-51, 2013.

[21] ——, "Has time come to switch from duty-cycled mac protocols to wake-up radio for wireless sensor networks?" *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 674-687, 2016.

[22] R. Rajagopalan and P. K. Varshney, "Data aggregation techniques in sensor networks: A survey," *IEEE Commun. Surv. Tuts.*, vol. 8, no. 4, pp. 48- 63, 2006.

[23] M. Ram and S. Kumar, "Analytical energy consumption model for mac protocols in wireless sensor networks," in *SPIN*, 2014, pp. 444-447.

[24] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Optimizing sensor networks in the energy-latency-density design space," *IEEE Transactions on mobile computing*, vol. 99, no. 1, pp. 70-80, 2002.

[25] ——, "Topology management for sensor networks: Exploiting latency and density," in *Proc. 3rd. ACM*, 2002, pp. 135-145.

[26] F. Wang, D. Li, and Y. Zhao, "Analysis of CSMA/CA in IEEE 802.15. 4," *IET communications*, vol. 5, no. 15, pp. 2187-2195, 2011.

[27] A. Whitmore, A. Agarwal, and L. Da Xu, "The Internet of things—a survey of topics and trends," *Information Systems Frontiers*, vol. 17, no. 2, pp. 261–274, 2015.

[28] S. Wijetunge, U. Gunawardana, and R. Liyanapathirana, "Throughput analysis of non-beacon enabled IEEE 802.15. 4 networks with unsaturated traffic," in *ISCIT*, 2012, pp. 1177–1182.

[29] X. Yang and N. H. Vaidya, "A wakeup scheme for sensor networks: Achieving balance between energy saving and end-to-end delay," in *RTAS*, 2004, pp. 19–26.

[30] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient mac protocol for wireless sensor networks," in *Proc. INFOCOM*, vol. 3, 2002, pp. 1567–1576.

[31] G. Zheng, J. Fu, S. Tang, Y. Li, and Z. Dong, "A dual channel-based energy efficient and low latency mac protocol for wsns," in *NSWCTC*, vol. 1, 2010, pp. 466–469.

[32] LAN/MAN Standards Committee, "IEEE Standard for Local and metropolitan area networks", New York, September, 2011.

Appendix A

Title:	Does Wake-up Radio Always Consume Lower Energy Than Duty-Cycled Protocols?
Authors:	Min Zhang, Debasish Ghose, and Frank Y. Li
Affiliation	Dept. of Information and Communication Technology, University of Agder (UiA), N-4898 Grimstad, Norway
Conference	IEEE Vehicular Technology Conference (VTC), Jun 2017 (accepted)
Copyright©	IEEE

Abstract

Many recent studies anticipate that wake-up radio (WuR) will replace traditional duty-cycled (DC) protocols given its overwhelming performance superiority on energy consumption. Meanwhile, the question on whether WuR performs *always* better than DC protocols has not been answered explicitly. In this paper, we investigate in-depth the energy consumption performance of WuR by considering various levels of traffic load in a wireless sensor network. By comparing SCM-WuR with both synchronous MAC (S-MAC) and asynchronous MAC (X-MAC), we ascertain that SCM-WuR does consume orders of magnitude lower energy than DC protocols when traffic load is low. However, our numerical results reveal at the same time that SCM-WuR does not have an absolute advantage when traffic load is heavy or saturated, especially when long range wake-up call is targeted.

A.1 Introduction

As an enabling technology of the Internet of things (IoT), wireless sensor networks (WSNs) have a unique requirement on energy efficiency in order to maintain network lifetime. Traditionally duty-cycled (DC) protocols have been a dominant solution for energy conservation in WSNs. Such a solution provides significant energy consumption benefits over an always-on approach by regularly switching on and off radio transceivers to diminish idle listening and overhearing. However, idle listening and overhearing cannot be completely suppressed since nodes have to check channel status and exchange data during their on-state. As an alternative technique, the concept of wake-up radio (WuR) has emerged in recent years and it is evident that WuR-enabled WSNs consume much lower energy than DC WSNs [1]-[3]. Such a technique requires to couple a sensor node with a wake-up receiver (WuRx). The WuRx has the role of listening to the channel *continuously* at an extremely low energy consumption level for incoming wake-up calls (WuCs) generated by a wake-up transmitter (WuTx). Upon the detection of a WuC, the receiver device will generate an interrupt and turn on its main radio (MR) and then a data communication is performed. In other words, WuR works in an on-demand manner.

A comprehensive survey on the state-of-the-art WuR implementations and protocols can be found in [4]. Among existing WuR solutions, the on-off keying (OOK) modulation based ultra-low power WuRs have gained popularity because of their superior performance and extended WuC ranges. The benefits of WuR have been revealed in many studies. The authors in [1] evaluated the performance of sub-carrier modulation (SCM) WuR and showed that its average power consumption was at the level of 1000 times lower than the MR. SCM-WuR builds its WuRx based on the off-the-shelf low-frequency integrated circuit AS3932 [9] which works at 125 kHz with integrated address correlation. Upon receiving a valid wake-up signal, it triggers the micro-controller to switch from sleep to active mode. A 125 kHz wake-up signal is modulated on an 868 MHz carrier frequency at the sender using OOK and demodulated at the receiver using a Schottky diode followed by a low-pass filter. Only the envelope signal is then passed to the 125 kHz receiver. The SCM-WuR is also designed to cover the transmission range up to 100 meters using an incorporated RF front-end [5] at the transmitter to enable transmission power up to +20 dBm. The benefits of WuR were also demonstrated in [2] by investigating the tradeoff between energy consumption and latency through measurement-based simulations. An ultra-low power WuR consuming power in the order of nanowatt was designed in [3]. Experimental results show that their implemented WuR achieves up to around 70 times longer lifetime against DC protocols (1% duty cycling) in low traffic load conditions.

The main idea behind the concept of WuR is to diminish the energy consumption for idle listening required in DC MAC protocols. The envisaged scenarios for applying WuRs are primarily targeted at low traffic load conditions. However, traffic load may get higher or even saturated in event-driven data reporting based WSNs when multiple sensor nodes detect an event such as fire detection at the same time. Indeed, most existing studies on WuR did not consider network scenarios *with heavy and/or saturated traffic conditions*. Although these simulation or experimental results are convincing, in-depth analysis on energy consumption rarely exists, especially when comparing with synchronous DC protocols. In this study, we compare the energy consumption of a popular WuR implementation, SCM-WuR, with two representative DC protocols, synchronous medium access control (S-MAC) [6] and asynchronous MAC (X-MAC) [7], considering both short and long transmission ranges.

The remaining of the paper is organized as follows. In Sec. II, the network scenario and assumptions are presented briefly. The energy consumptions of the three studied protocols are analyzed in Sec. III, followed by numerical results presented in Sec. IV. Finally the paper is concluded in Sec. V.

A.2 Network Scenario and Assumptions

Consider a one-hop network with N sensor nodes in which all nodes are within the transmission range of each other. Assume an identical constant packet arrival rate, λ (packets/sec), for all nodes. Since the energy consumption of radio communication is the dominant component among energy consuming sources for a node, we focus solely on the energy consumption of data transmission, data reception, and protocol handshakes in our analysis.

During a fixed observation time, T, each node sends the same number of packets to one of its neighbours with equal chance. The generated packets are assumed to be unicast messages with identical packet length. Each successful packet transmission is followed by an acknowledgment

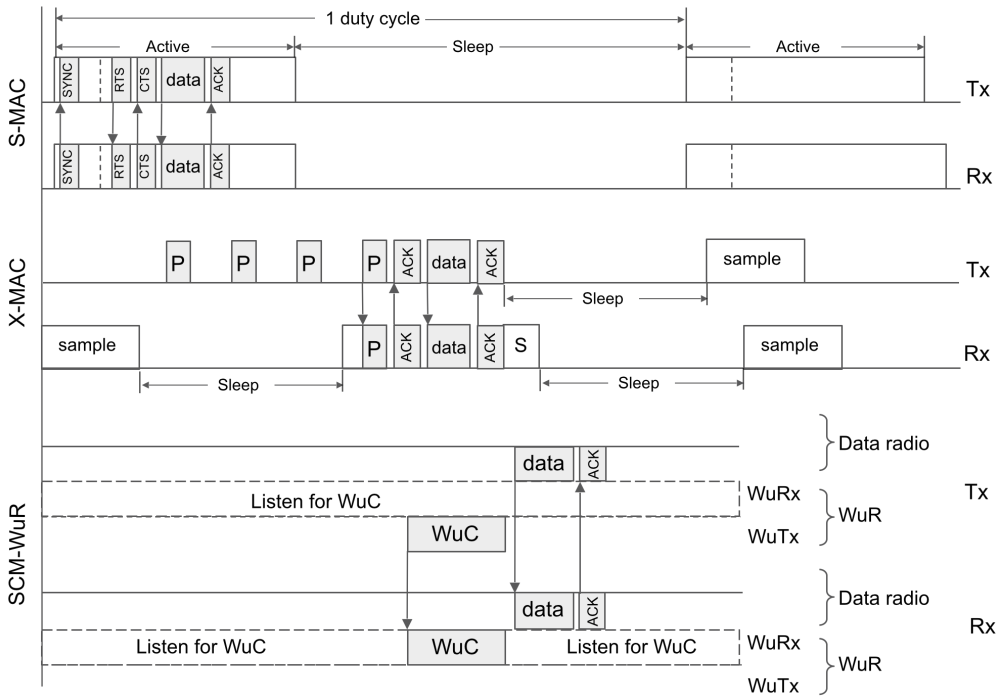


Figure A.1: The operation of S-MAC, X-MAC, and SCM-WuR.

(ACK) from the receiver. No collision is considered in our energy consumption calculation. During the observation time, each node transmits and receives $T \cdot \lambda$ packets. Thus, the total number of packets that are transmitted in the network is $N \cdot T \cdot \lambda$.

A.3 Energy Analysis for Three Protocols

Select arbitrarily one of the N nodes and refer to it as a reference node (RN). In this study, we concentrate on calculating *the total energy consumption of the RN* during the observation time for three protocols, S-MAC, X-MAC, and SCM-WuR, denoted as E_{smac}, E_{xmac}, and E_{wur} respectively. The notations and their explanations used in our energy consumption calculations are listed in Table 1, together with their values that are obtained from each protocol's original design and device data sheets.

A.3.1 S-MAC

S-MAC is a representative synchronous MAC protocol proposed for DC WSNs [6]. In S-MAC, time is partitioned into multiple cycles of equal length and each cycle includes an active period T_a and a sleep period T_s. The active period is dedicated for SYNC message exchanges and data communication. By exchanging SYNC messages periodically, neighboring nodes share the same wake-up and sleep schedule. Idle listening happens when a node is in the active state listening to channel for incoming data or SYNC message while no message is transmitted. The behavior of a sender node and a receiver node according to S-MAC is shown in Fig. 1.

E_{smac} can be calculated by summing up the energy consumed at every state, i.e., for SYNC message exchange, data communication, overhearing, idle listening, and sleeping.

$$E_{smac} = E_{tx}^{s} + E_{rx}^{s} + E_{tx}^{d} + E_{rx}^{d} + E_{oh} + E_i + E_s, \tag{A.1}$$

where $E_{tx}^{s}, E_{rx}^{s}, E_{tx}^{d}, E_{rx}^{d}, E_{oh}, E_i$, and E_s, denote the energy that the RN consumed for SYNC message transmission and reception, data transmission and reception, overhearing, idle listening, and sleeping respectively during T.

Since SYNC messages in S-MAC are exchanged at a rate of r_{syn} (per duty cycle), we have $E_{tx}^{s} = T \cdot (E_{cca} + E_{bo} + P_{tx} \cdot T_{syn}) \cdot \frac{r_{syn}}{T_s+T_a}$, $E_{rx}^{s} = T \cdot (N-1) \cdot (P_{rx} \cdot T_{syn}) \cdot \frac{r_{syn}}{T_s+T_a}$. Furthermore, E_{tx}^{d} and E_{rx}^{d} can be calculated by: $E_{tx}^{d} = T \cdot \lambda \cdot (E_{cca} + E_{bo} + P_{tx} \cdot T_{rts} + P_{rx} \cdot T_{cts} + P_{tx} \cdot T_d + P_{rx} \cdot T_{ack} + P_x \cdot T_{SIFS} \cdot 3)$, and $E_{rx}^{d} = T \cdot \lambda \cdot (P_{rx} \cdot T_{rts} + P_{tx} \cdot T_{cts} + P_{rx} \cdot T_d + P_{tx} \cdot T_{ack} + P_x \cdot T_{SIFS} \cdot 3)$, where E_{cca} and E_{bo} are the energy consumption of clear channel assessment (CCA) and contention back-off (BO) procedures respectively, which can be calculated by: $E_{cca} = P_{cca} \cdot T_{cca}$ and $E_{bo} = P_{bo} \cdot T_{slot} \cdot (CW-1)/2$. During the data transmission period, nodes which are not the intended receivers will follow the network allocation vector (NAV) notification and go to sleep after receiving a request to send (RTS) message. Since there are N nodes in the network, we have $E_{oh} = T \cdot \lambda \cdot (N-2) \cdot P_{rx} \cdot T_{rts}$. The NAV duration is set to be the remaining time of the current transmission. The accumulated NAV duration after the RN overhears the RTS message is therefore $D_{nav} = T \cdot \lambda \cdot (N-2) \cdot (T_{cts} + T_d + T_{ack} + T_{SIFS} \cdot 3)$. Similarly, the idle listening energy can by calculated by: $E_i = P_i \cdot (T \cdot \frac{T_a}{T_a+T_s} - D_{tx}^{s} - D_{rx}^{s} - D_{oh} - D_{nav} - D_{tx}^{d} - D_{rx}^{d})$, where D_{tx}^{s}, D_{rx}^{s}, D_{tx}^{d}, D_{rx}^{d}, and D_{oh} denote the accumulated duration that the RN used for SYNC message transmission and reception, data transmission and reception ($T \cdot \lambda$ packets in total), and overhearing respectively, and they can be easily derived from the aforementioned expressions. Lastly, the energy consumption during the sleep period can be calculated by: $E_s = P_s \cdot (T \cdot \frac{T_s}{T_a+T_s} + D_{nav})$.

A.3.2 X-MAC

X-MAC is a popular asynchronous DC MAC protocol proposed in [7]. It employs a strobed preamble mechanism by transmitting a sequence of short preamble packets, each containing the address of the target receiver. The intervals between preamble packets permit the target receiver to send an ACK that suspends transmitting subsequent preamble packets. As shown in Fig.1, the transmitter keeps sending short preambles until the receiver wakes up and sends back an ACK. The transmitter will start to send data right after receiving the ACK and it goes to sleep for a T_s duration after data communication. The receiver will instead keep listening to the channel for any possible queued packet for a short duration (marked as S in Fig. 1, during which the energy consumption is negligible) before it goes to sleep. When a node wakes up and stays awake for a sample period without capturing any incoming preamble, it will go to sleep again after a T_a duration.

E_{xmac} is obtained by the sum of the energy consumption at every state, i.e., for preamble communication, data communication, channel sampling, overhearing, and sleeping.

$$E_{xmac} = E^p_{tx} + E^p_{rx} + E^d_{tx} + E^d_{rx} + E_{sam} + E_{oh} + E_s, \tag{A.2}$$

where E^p_{tx}, E^p_{rx} and E_{sam} denote the accumulated energy consumed by the RN for preamble transmission and reception, and channel sampling. Since the transmitter will keep sending short preambles then listening to the channel for a D_{ack} duration until it receives an ACK[1], we have $E^p_{tx} = T \cdot \lambda \cdot [(E_{cca} + E_{bo}) \cdot 1 + (P_{tx} \cdot D_p + P_x \cdot T_{SIFS} + P_i \cdot D_{ack}) \cdot N_{pre}]$, and $E^p_{rx} = T \cdot \lambda \cdot (P_{rx} \cdot D_p + P_x \cdot T_{SIFS} + P_{tx} \cdot T_{ack})$. Here $N_{pre} = 1/\left(\frac{T_a - D_p}{T_a+T_s}\right)$ is the average number of preambles that a transmitter needs to send before getting an ACK [7]. E^d_{tx} and E^d_{rx} are calculated by $E^d_{tx} = T \cdot \lambda \cdot (P_{tx} \cdot T_d + P_{rx} \cdot T_{ack} + P_x \cdot T_{SIFS})$ and $E^d_{rx} = T \cdot \lambda \cdot (P_{rx} \cdot T_d + P_{tx} \cdot T_{ack} + P_x \cdot T_{SIFS})$. Furthermore, $E_{sam} = P_{sam} \cdot [T - D^d_{tx} - D^d_{rx} - T \cdot \lambda \cdot (2 \cdot T_s + \frac{T_a}{T_a+T_s} \cdot (N-2) \cdot (D_p + T_s))] \cdot \frac{T_a}{T_a+T_s}$. Overhearing occurs when a node receives a preamble which is not intended to it and $E_{oh} = P_{rx} \cdot \frac{T_a}{T_a+T_s} \cdot T \cdot \lambda \cdot (N-2) \cdot D_p$.

[1]Note that a node performs CCA only before sending its first preamble.

Lastly E_s is calculated by $E_s = P_s \cdot \{T \cdot \lambda \cdot N \cdot T_s + [T - D^d_{tx} - D^d_{rx} - T \cdot \lambda \cdot (2T_s + \frac{T_a}{T_a+T_s}(N - 2)(D_p + T_s))] \cdot \frac{T_s}{T_a+T_s}\}$.

A.3.3 SCM-WuR

SCM-WuR is an in-band WuR and its principle is also shown in Fig. 1. Consider the transmitter initiated (TI) transmission mode in SCM-WuR [1]. A WuC with the intended receiver's address is initiated by the WuTx when there is a packet to send. In SCM-WuR, the WuC is transmitted by the MR to the intended WuRx [1]. After detecting a WuC by the WuRx, the receiver turns on its MR. Assume that IEEE 802.15.4 is adopted for MR operation [1]. An ACK is sent back to the transmitter when the data packet is successfully received. Then both nodes will turn their MR off to the sleep mode but their WuRx will keep idle listening. Since the power consumption of WuRx in the listening state is in the order of μW or even nW, compared with traditional radios which are in the order of mW, a huge amount of energy can be saved.

Similarly, E_{wur} is calculated by the sum of the energy consumption at every state for both MR and WuR, including WuC transmission and reception, data communication, MR sleep state, overhearing, and idle listening of the WuR.

$$E_{wur} = E^w_{tx} + E^w_{rx} + E^d_{tx} + E^d_{rx} + E_s + E_{oh} + E_i, \tag{A.3}$$

where E^w_{tx} and E^w_{rx} denote the energy consumed by the RN for sending and receiving WuCs during T. Accordingly, we have $E^w_{tx} = T \cdot \lambda \cdot P^w_{tx} \cdot T_{wuc}$ and $E^w_{rx} = T \cdot \lambda \cdot P^w_{rx} \cdot T_{wuc}$ respectively. For data transmission E^d_{tx} and E^d_{rx} are calculated by $E^d_{tx} = T \cdot \lambda \cdot (E_{cca} + E_{bo} + P_{tx} \cdot T_d + P_{rx} \cdot T_{ack} + P_x \cdot T_{SIFS})$ and $E^d_{rx} = T \cdot \lambda \cdot (P_{rx} \cdot T_d + P_{tx} \cdot T_{ack} + P_x \cdot T_{SIFS})$. The MR will keep sleeping if there is no data communication. Thus $E_s = P_s \cdot (T - D^w_{tx} - D^d_{tx} - D^d_{rx})$, where D^w_{tx} is the accumulated duration during which the RN is sending in total T$\cdot\lambda$ WuCs. Overhearing happens when the WuRx receives a WuC which is not intended to it. Thus $E_{oh} = T \cdot \lambda \cdot P^w_{rx} \cdot (N - 2) \cdot T_{wuc}$. The WuR of the RN keeps listening to the channel at a very low power level although there is no ongoing traffic on the channel. Therefore $E_i = P^w_i \cdot (T - D^w_{rx} - D_{oh})$, where D^w_{rx} denotes the accumulated duration that the RN used for receiving WuCs.

A.4 Numerical Results and Discussions

Consider a small-scale one-hop network with $N = 6$ nodes. We present below the numerical results obtained via MATLAB. Three per-node traffic load levels are considered, as light ($\lambda = 0.01$ packet/s), heavy ($\lambda = 1$ packet/s), and saturated (with different saturation points). The WuC transmission ranges are configured as 24 m (short range) and 100 m (long range) [1] respectively.

For illustration clarity, the active period for both S-MAC and X-MAC is configured as a fixed duration $T_a = 10$ ms, but the sleep period T_s varies from 10 ms to 200 ms. Note that for S-MAC, T_a is the SYNC and data transmission duration while for X-MAC, T_a is the maximum duration for channel sampling in each duty cycle. The observation time is configured to be $T = 10$ minutes.

For comparison fairness, we consider the same type of sensor node, CC1101 [10], in all three protocols (as the MR in SCM-WuR). The corresponding parameters are configured based on the data sheet, as shown in Table I. It is worth mentioning that the maximum transmission power of CC1101 is +12 dBm and a WuC sent directly by CC1101 cannot reach 100 m. To achieve a long WuC coverage of 100 m, an RF front-end which enables an output transmission power level up to +20 dBm is adopted [1]. Correspondingly, the transmission current needed would be 152 mA and the transmission power is 456 mW.

A.4.1 Energy Comparison under Light Traffic Load

Fig. 2 illustrates the energy consumption levels at a light traffic load with $\lambda = 0.01$ packet/s. As shown in the figure, the energy consumption of both S-MAC and X-MAC decreases monotonically as the sleep time, T_s, increases. This is a typical behavior of DC protocols since longer sleep time leads to less idle listening. As SCM-WuR works in an on-demand manner, its energy consumption is not dependent on the length of the sleep time. Thus, the SCM-WuR curve is a horizontal line. The total energy consumption for SCM-WuR depends on the traffic load and the observation time, i.e., $T \cdot \lambda$, not on T_s.

At a light traffic load, S-MAC always consumes highest energy among these three protocols, whereas SCM-WuR presents an extremely low value for both short and long WuC ranges. With a short WuC range, the energy consumption of SCM-WuR is several hundred times lower than that

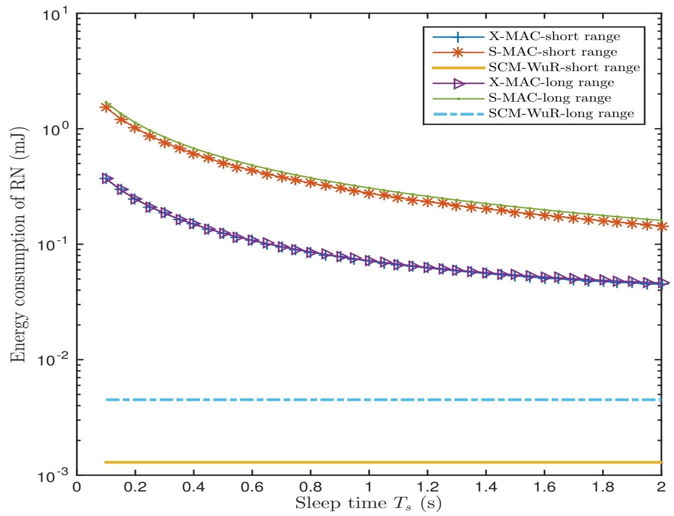


Figure A.2: Energy consumption of the RN at light traffic load $\lambda = 0.01$ packet/s.

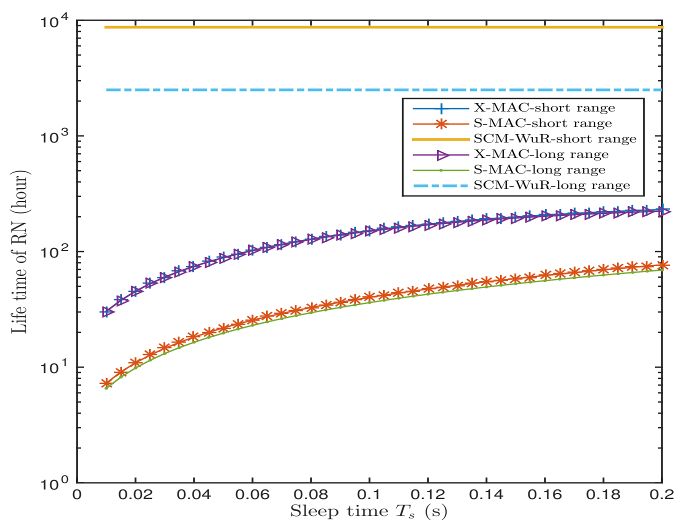


Figure A.3: Lifetime of the RN at light traffic load $\lambda = 0.01$ packet/s.

of the other two protocols when the DC sleep time T_s is short. When $T_s < 0.02$ s, the energy consumption of SCM-WuR is even over 1000 times lower than that of S-MAC. Similar results have been observed when comparing SCM-WuR with 802.15.4, B-MAC, X-MAC, and RI-MAC [1].

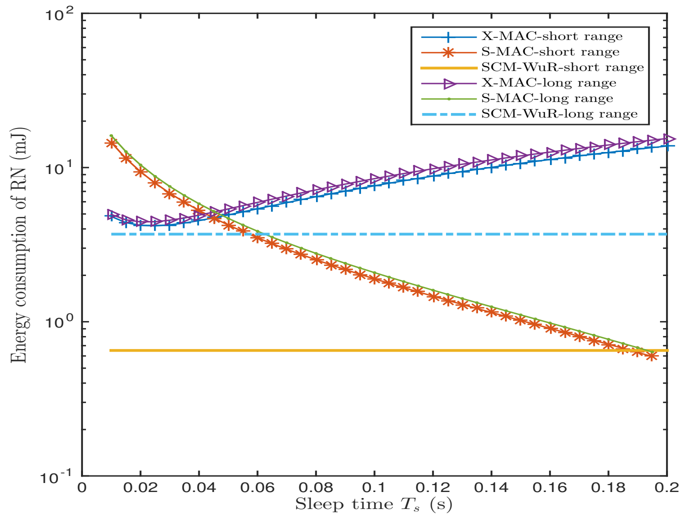


Figure A.4: Energy consumption of the RN at heavy traffic load $\lambda = 1$ packet/s.

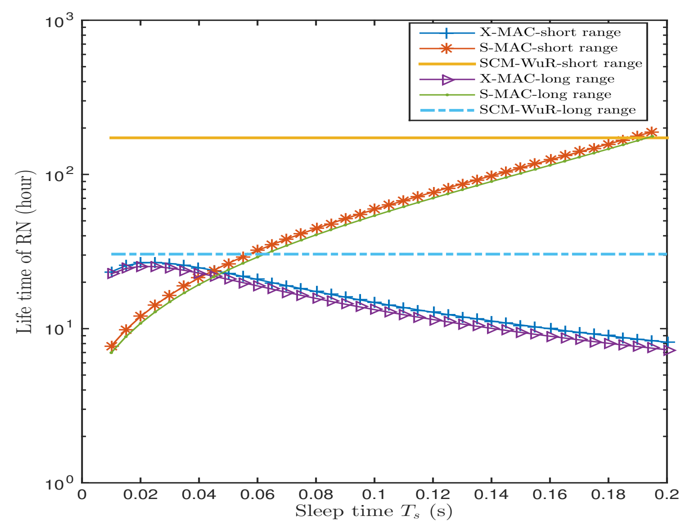


Figure A.5: Lifetime of the RN at heavy traffic load $\lambda = 1$ packet/s.

Despite the fact that it consumes much higher energy to reach long WuC coverage as discussed above, SCM-WuR still presents dozens or over one hundred times energy saving in comparison with S-MAC and X-MAC.

Fig. 3 shows the lifetime of the three protocols when the capacity of a battery is 1500 mAh. At a light traffic load, SCM-WuR is able to work for almost one year for short transmission range and 100 days for long transmission range. In comparison, X-MAC and S-MAC can only last for around 100 days and 3 days respectively with 5% duty cycling.

A.4.2 Energy Comparison under Heavy Traffic Load

Fig. 4 illustrates the energy consumption levels at a heavy traffic load with $\lambda = 1$ packet/s. For the DC protocols, when T_s increases, less idle listening is needed, leading to decreased energy consumption in S-MAC. However, the energy consumption of X-MAC increases beyond $T_s = 0.02$ s. The reason is that when T_s keeps increasing, the duration for a transmitter sending short preambles increases and more energy is wasted. Compared with the light traffic load case, SCM-WuR consumes a significantly higher amount of energy when $\lambda = 1$ packet/s, especially with the long WuC range. From Fig. 4, it is evident that the energy consumption of SCM-WuR is higher than that of S-MAC when $T_s > 0.19$ s for short range WuC or $T_s > 0.06$ s for long range WuC respectively. With both short and long ranges, SCM-WuR always consumes lower energy than X-MAC does.

The lifetimes of the RN for three protocols at heavy traffic load conditions are illustrated in Fig. 5. As mentioned earlier, the energy consumption for SCM-WuR is irrelevant to the sleep duration. For S-MAC, longer lifetime is achieved when nodes sleep longer. For X-MAC, the RN lifetime is decreasing as T_s becomes longer since more preamble messages are required. As a result, SCM-WuR loses its advantage against S-MAC since the energy wastage for idle listening in S-MAC is not severe at a heavy traffic load. To enable long range WuCs, the advantage of WuRs may disappear even faster.

A.4.3 Energy Comparison under Saturated Traffic Load

When the network becomes saturated, there are always data communications in the channel and correspondingly no idle listening time is needed. Fig. 6 illustrates the saturation points for the three protocols. In the figure, the value of each point is corresponding to a different packet arrival

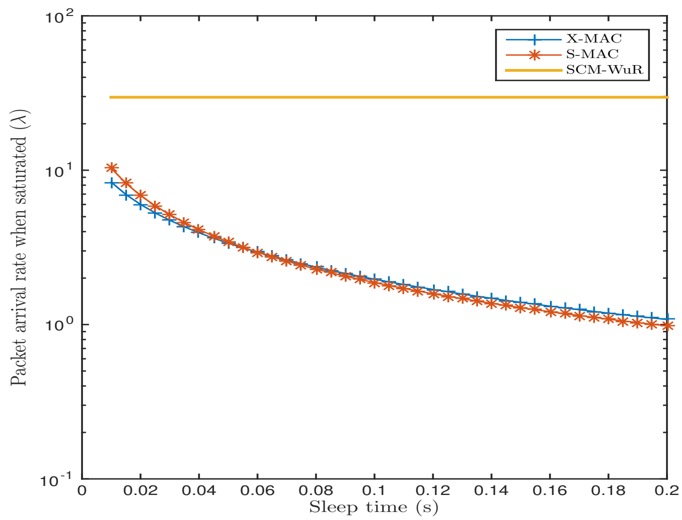


Figure A.6: Packet arrival rate at saturated traffic load.

rate λ which makes the network saturated.

From Fig. 6 we can conclude that SCM-WuR is able to handle the highest traffic load until the network becomes saturated. In other words, it is able to transmit a highest number of packets during T. X-MAC and S-MAC lead to their saturation points with much lower values of λ. This is because X-MAC will force nodes to sleep for a T_s duration after each data communication or overhearing and S-MAC suffers from its control message overhead. When nodes always have packets to transmit, WuR loses its benefit against S-MAC and X-MAC since there is no idle listening when the network is saturated.

Under saturation traffic conditions, we can observe from Fig. 7 that SCM-WuR always achieves lower energy efficiency in transmitted bits per mJ than S-MAC does. However, the energy performance of SCM-WuR is still better than X-MAC in short range and partially better in long range. The reason is that the energy consumed by the overhead of WuR (i.e., WuC) is lower than that of the X-MAC (i.e., preamble).

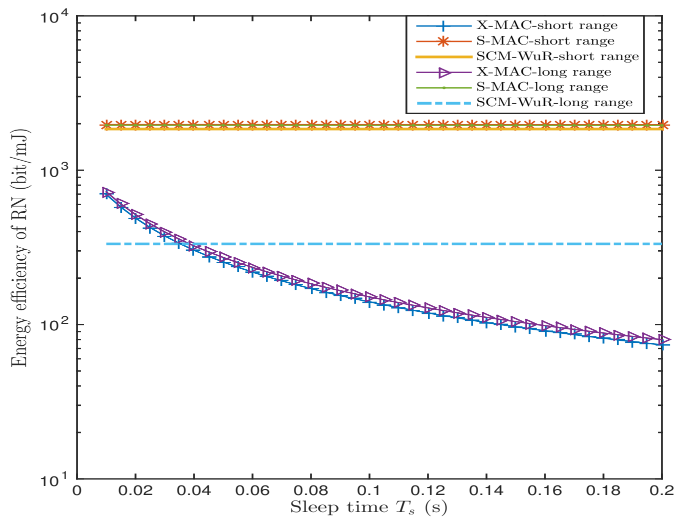


Figure A.7: Energy efficiency of the RN at saturated traffic load.

A.5 Conclusions

In this paper, we performed in-depth comparison of energy consumptions of SCM-WuR, S-MAC and X-MAC and analyzed their energy performance at various traffic loads in both short and long range scenarios. Through numerical results, we confirm the excellent energy consumption performance of SCM-WuR when traffic load is light. When traffic load gets heavier, SCM-WuR gradually loses its advantage against DC protocols for idle listening. While SCM-WuR always achieves better energy performance than X-MAC, it may not perform better than S-MAC under saturated or close-to-saturation traffic conditions. Furthermore, SCM-WuR is able to handle much higher traffic volume than the other two protocols. Our results imply also that it may not be beneficial to enable long range WuRs with respect to energy efficiency and power consumption.

Table A.1: Notations, descriptions and values [1][6][7][8][9][10]

Notation	Description	Value		Unit
		Short range (24 m)	Long range (100 m)	
P_{tx}	MR data transmission power	43.2	52.2	mW
P_{rx}	MR data reception power	56.4	56.4	mW
P_i	MR idle listening power	56.4	56.4	mW
P_{sam}	X-MAC channel sampling power	12.3	12.3	mW
P_s	MR sleeping power	3	3	μW
P_x	Power to switch MR between Rx and Tx	45	45	mW
P_{bo}	Back-off power	15.48	15.48	mW
P_{cca}	CCA power	56.4	56.4	mW
P_{tx}^w	WuC transmission power	43.2	456	mW
P_{rx}^w	WuC reception power	24.9	24.9	μW
P_i^w	WuR idle listening power	7.8	10.5	μW
T_{rts}	RTS duration	256		μs
T_{cts}	CTS duration	256		μs
T_{ack}	ACK duration	352		μs
T_{SIFS}	SIFS duration	192		μs
T_d	Packet duration	4.096		ms
T_{wuc}	WuC duration	12.2		ms
T_{cca}	CCA duration	X-MAC: 10; S-MAC, WuR:0.128		ms
T_{slot}	Slot time duration	320		μs
D_{ack}	X-MAC: listening for ACK duration	10		ms
D_p	X-MAC: short preamble duration	10		ms
λ	Data packet arrival rate	variable		packets/s
r_{syn}	SYNC message rate	0.1		number/cycle
CW	Contention window size	32		
R_d	MR data rate	250		kbps
R_{wur}	WuC data rate	2730		bps
L_d	Data frame length	128		byte
L_{wur}	WuC frame length	32		bit

Bibliography

[1] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl, "Has time come to switch from duty-cycled MAC protocols to wake-up radio for wireless sensor networks?" *IEEE Trans. Netw.*, vol. 24, no. 2, pp. 674-687, Apr. 2016.

[2] V. Jelicic, M. Magno, D. Brunelli, V. Bilas, and L. Benini, "Benefits of wake-up radio in energy-efficient multimodal surveillance wireless sensor network," *IEEE Sensors J.*, vol. 14, no. 9, pp. 3210–3220, Sep. 2014.

[3] M. Magno, V. Jelicic, B. Srbinovski, V. Bilas, E. Popovici, and L. Benini, "Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver," *IEEE Trans. Ind. Informat.*, vol. 12, no. 2, pp. 633-644, Apr. 2016.

[4] F. Z. Djiroun and D. Djenouri, "MAC protocols with wake-up radio for wireless sensor networks: A review," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 587-618, 2017.

[5] J. Oller, I. Demirkol, J. Casademont, J. Paradells, G. U. Gamm, and L. Reindl, "Performance evaluation and comparative analysis of subcarrier modulation wake-up radio systems for energy-efficient wireless sensor networks," *Sensors*, vol. 14, no. 1, pp. 22-51, 2014.

[6] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 1-10.

[7] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: A short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proc. ACM SenSys*, 2006, pp. 307–320.

[8] T. R. Burchfield, S. Venkatesan, and D. Weiner, “Maximizing throughput in Zigbee wireless networks through analysis, simulations and implementations,” in *Proc. Int. Workshop Localized Algorithms and Protocols for WSNs*, 2007, pp. 15-29.

[9] AMS, *AS3932 3D Low Frequency Wakeup Receiver Data Sheet*, Austria, Mar. 2015.

[10] Texas Instrument, *CC1101 Low-Power Sub-1 GHz RF Transceiver Data Sheet*, TX, USA, Dec. 2016.

Appendix B

Main Functions of EHA-WuR Protocol in Omnet++

B.1 Message Control Function

```
void Dual_Mode::handleSelfMsg(cMessage* msg) {
    __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
    // FSM for WuR acting as Transmitter-Initiated
    switch (FSM_STATE) {
            // INIT state
            // ************************************************
            case INIT_STATE:
                if (msg->getKind() == INIT_MSG_TYPE) {
                    __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                    __MACRO_RADIO_OFF__
                    FSM_STATE = SWITCHING_RADIO_TO_SLEEP_STATE;
                }
                else if (msg->getKind() == APP_DATA_MESSAGE||msg->getKind() ==TRANSMISSION_OVER) {
                    __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                    delete msg;
                }
                __SHOW_OTHER_MSG_RECEIVED_WHILE_IN__(FSM_STATE, debug)
                break;

            // WAIT_SWITCH_STATE_RX_TO_SLEEP state of main NIC
            // ***********************************************
            case SWITCHING_RADIO_TO_SLEEP_STATE:
                if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == RADIO_SWITCHED)) {
                    dual_to_phy = FindModule<MacToPhyInterface*>::findSubModule(this->getParentModule());
                    if(dual_to_phy->getRadioState() == MiximRadio::SLEEP) {
                        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                        FSM_STATE = SLEEP_STATE;
                    }
                    delete msg;
                }
```

```
        __SHOW_OTHER_MSG_RECEIVED_WHILE_IN__(FSM_STATE, debug)
        break;

    // SLEEP_STATE (send OR receive WuC)
    // **************************************************
    case SLEEP_STATE:
        if((msg->getArrivalGateId() == upperLayerIn_dual) && (msg->getKind() == APP_WUC_MESSAGE)) {
            __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
            sendDelayed(msg, TIME_TO_TXSETUP,lowerLayerOut_dual);
            __MACRO_RADIO_RX_ON__
            FSM_STATE = WUC_TO_DATA_TX_STATE;
        }

        else if((msg->getArrivalGateId() == upperLayerIn_dual) && (msg->getKind() == FW_WuC) ) {
            if( modeState==1){
                sendDelayed(msg,TIME_TO_SWITCH,lowerLayerOut_dual);
                __MACRO_RADIO_TX_ON__
                cancelEvent(rx_timeout_msg);
                scheduleAt(simTime() + RX_TIMEOUT, rx_timeout_msg);
                FSM_STATE = WUC_TO_DATA_TX_STATE;
        }
        else if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == TRANSMISSION_OVER)) {
                  __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                  __MACRO_RADIO_OFF__
                  delete msg;
              }

        else if((msg->getArrivalGateId() == upperLayerIn_dual) && (msg->getKind() == MODE_SWITCH)) {
             sendDelayed(msg,TIME_TO_TXSETUP,lowerLayerOut_dual);
             __MACRO_RADIO_TX_ON__
             FSM_STATE = TX_DATA_STATE;
         }

        else if(msg->getArrivalGateId() == lowerLayerInWuR_dual && (msg->getKind() == APP_WUC_MESSAGE) && isSink==false) {
            __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
            NetwPkt* incoming_wuc = static_cast<NetwPkt*>(msg);
            if (incoming_wuc->getDestAddr() == findHost()->getIndex()) {
                // main NIC on
                __MACRO_RADIO_RX_ON__
               cancelEvent(rx_timeout_msg);
                scheduleAt(simTime() + RX_TIMEOUT, rx_timeout_msg);
                FSM_STATE = RX_DATA_STATE;
                delete msg;
            }else if (incoming_wuc->getDestAddr() != findHost()->getIndex() && isCoordinator==true && modeState==1){
                send(msg, upperLayerOut_dual);
                __MACRO_RADIO_RX_ON__
            }
        }
        else if(msg->getArrivalGateId() == lowerLayerInWuR_dual && (msg->getKind() == FW_WuC) && isSink==true) {
            __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
              NetwPkt* incoming_wuc = static_cast<NetwPkt*>(msg);
             debugEV<<"incoming_wuc source address=incoming_wuc->getDestAddr()"<<endl;
             if (incoming_wuc->getDestAddr() == findHost()->getIndex()) {
                                // main NIC on
                __MACRO_RADIO_RX_ON__
                cancelEvent(rx_timeout_msg);
                scheduleAt(simTime() + RX_TIMEOUT, rx_timeout_msg);
                FSM_STATE = RX_DATA_STATE;
                delete msg;
             }
        }

        else if(msg->getArrivalGateId() == lowerLayerInWuR_dual && (msg->getKind() == MODE_SWITCH) && isCoordinator==false) {
           __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
```

```
            modeState=2;
                    send(new cMessage("MODE_SWITCH_MESSAGE",CHILD_MODE_SWITCH), lowerControlOut_dual );
            send(new cMessage("MODE_SWITCH_MESSAGE",CHILD_MODE_SWITCH),upperControlOut_dual);
                    }

    else if(msg->getArrivalGateId() == lowerLayerIn_dual && (msg->getKind() == APP_DATA_MESSAGE && isCoordinator==true)) {
        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
        send(msg, upperLayerOut_dual);
    }

    else if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == MODE_SWITCH)) {
            send(msg, upperControlOut_dual);
        }


    // WuR receiving DATA_MESSAGES
    // ---------------------------
    else if(msg->getArrivalGateId() == lowerLayerInWuR_dual && (msg->getKind() == APP_DATA_MESSAGE)) {
        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
        delete msg;
    }
    __SHOW_OTHER_MSG_RECEIVED_WHILE_IN__(FSM_STATE, debug)
    break;

// WUC_TO_DATA_TX_STATE
// ********************************
case WUC_TO_DATA_TX_STATE:
    // ----------------------------------
    if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == TRANSMISSION_OVER)) {
         __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
         __MACRO_RADIO_OFF__
         delete msg;
                    }
    else if (msg->getKind() == RX_TIMEOUT_MSG_TYPE) {
                        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                        FSM_STATE = INIT_STATE;
                        scheduleAt(simTime() + TIME_TO_GO_TO_SLEEP, init_msg);
         }
    else if((msg->getArrivalGateId() == upperLayerIn_dual) && (msg->getKind() == APP_DATA_MESSAGE)) {
        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
        __MACRO_RADIO_TX_ON__
        dual_to_phy = FindModule<MacToPhyInterface*>::findSubModule(this->getParentModule());
        sendDelayed(msg, TIME_TO_TXSETUP, lowerLayerOut_dual);
        FSM_STATE = TX_DATA_STATE;
    }
    else if((msg->getArrivalGateId() == upperLayerIn_dual) && (msg->getKind() == FW_DATA)) {
        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
            __MACRO_RADIO_TX_ON__
        cancelEvent(rx_timeout_msg);
        dual_to_phy = FindModule<MacToPhyInterface*>::findSubModule(this->getParentModule());
        sendDelayed(msg, TIME_TO_TXSETUP, lowerLayerOut_dual);
        FSM_STATE = TX_DATA_STATE;
    }

    else if(msg->getArrivalGateId() == lowerLayerInWuR_dual && (msg->getKind() == MODE_SWITCH) && isCoordinator==false) {
         __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
         send(msg, upperLayerOut_dual);
       }

    else if (msg->getKind() == WUC_START_TX) {
        send(msg, upperControlOut_dual);
    }
    else if (msg->getKind() == RADIO_SWITCHED) {
        delete msg;
```

```
    }
    else if (msg->getArrivalGateId() == lowerControlIn_dual &&msg->getKind() == MAC_ERROR) {

                        FSM_STATE=INIT_STATE;
                        scheduleAt(simTime() + TIME_TO_GO_TO_SLEEP, init_msg);
                        delete msg;
                    }
    else if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == MODE_SWITCH)) {
                            send(msg, upperControlOut_dual);
                        }
    __SHOW_OTHER_MSG_RECEIVED_WHILE_IN__(FSM_STATE, debug)
    break;

// TX_DATA_STATE
// *********************
case TX_DATA_STATE:
    // ----------------------------
    if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == TRANSMISSION_OVER)) {
        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)

        FSM_STATE = INIT_STATE;
        scheduleAt(simTime() + TIME_TO_GO_TO_SLEEP, init_msg);
        delete msg;
    }
    else if (msg->getKind() == RADIO_SWITCHED) {
        delete msg;
    }
    else if (msg->getKind() == MAC_ERROR) {
        __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
        FSM_STATE = INIT_STATE;
        // RESET
        scheduleAt(simTime() +TIME_TO_GO_TO_SLEEP, init_msg);
        delete msg;
    }
    else if (msg->getArrivalGateId() == lowerLayerInWuR_dual) {
                        delete msg;
                    }

    else if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == MODE_SWITCH)) {
                            send(msg, upperControlOut_dual);
                        }
    __SHOW_OTHER_MSG_RECEIVED_WHILE_IN__(FSM_STATE, debug)
    break;

// RX_DATA_STATE (to end)
// *********************
case RX_DATA_STATE:
    // -------------------
  if ((msg->getArrivalGateId() == lowerLayerIn_dual) && (msg->getKind() == APP_DATA_MESSAGE) && (modeState==2) && isSink==true) {
        NetwPkt* incoming_data = static_cast<NetwPkt*>(msg);
        if (incoming_data->getDestAddr() == findHost()->getIndex()) {
            __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
            cancelEvent(rx_timeout_msg);
            recordPacket(PassedMessage::INCOMING, PassedMessage::LOWER_DATA, msg);
            send(msg, upperLayerOut_dual);
            FSM_STATE = INIT_STATE;
            scheduleAt(simTime() + TIME_TO_GO_TO_SLEEP, init_msg);
        }

    }

  else if ((msg->getArrivalGateId() == lowerLayerIn_dual) && (msg->getKind() == FW_DATA) && (modeState==1) && isSink==true) {
       NetwPkt* incoming_data = static_cast<NetwPkt*>(msg);
       if (incoming_data->getDestAddr() == findHost()->getIndex()) {
```

```
                __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                cancelEvent(rx_timeout_msg);
                                recordPacket(PassedMessage::INCOMING, PassedMessage::LOWER_DATA, msg);
                                send(msg, upperLayerOut_dual);
                                FSM_STATE = INIT_STATE;
                                scheduleAt(simTime() + TIME_TO_GO_TO_SLEEP, init_msg);
                            }

                        }
            // ------------------
          else if ((msg->getArrivalGateId() == lowerLayerIn_dual) && (msg->getKind() == APP_WUC_MESSAGE)) {
                __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                delete msg;
            }
            // we have a timeout because no received DATA after being activated
            // ----------------------------------------------------------------
            else if (msg->getKind() == RX_TIMEOUT_MSG_TYPE) {
                __FSM_TRANSITION_ANNOUNCEMENT__(FSM_STATE)
                FSM_STATE = INIT_STATE;
                scheduleAt(simTime() + TIME_TO_GO_TO_SLEEP, init_msg);
            }
            else if (msg->getKind() == RADIO_SWITCHED) {

                delete msg;
            }
            else if (msg->getArrivalGateId() == lowerLayerInWuR_dual) {
                delete msg;
            }
            else if ((msg->getArrivalGateId() == lowerControlIn_dual) && (msg->getKind() == MODE_SWITCH)) {
                                send(msg, upperControlOut_dual);
                            }
            __SHOW_OTHER_MSG_RECEIVED_WHILE_IN__(FSM_STATE, debug)
            break;
        }
        return;
}
```

B.2 Residential Energy Monitoring Function

```
void SimpleBattery::deductAndCheck() {
        Enter_Method_Silent();
        // already depleted, devices should have stopped sending drawMsg,
        // but we catch any leftover messages in queue
        if (lessOrEqualNull(residualCapacity)) {
                return;
        }

        const double    dLastResCap= residualCapacity;
        const simtime_t now        = simTime();
        const simtime_t tDeltaT    = now - lastUpdateTime;
        const double    dVoltDelta = voltage * SIMTIME_DBL(tDeltaT);

        // If device[i] has never drawn current (e.g. because the device
        // hasn't been used yet or only uses ENERGY) the currentActivity is
        // still -1.  If the device is not drawing current at the moment,
        // draw has been reset to 0, so energy is also 0.  (It might perhaps
        // be wise to guard more carefully against fp issues later.)

        for (int i = 0; i < numDevices; i++) {
                int currentActivity = devices[i].currentActivity;
                if (currentActivity > -1) {
```

```
                        const double energy = devices[i].draw * dVoltDelta;
                        if (energy != 0) {
                                if (residualCapacity >= energy || residualCapacity != 0.0) {
                                        devices[i].accts[currentActivity] += energy;
                                }
                                devices[i].times[currentActivity] += tDeltaT;
                                residualCapacity                  -= energy;
                        }
                }
        }

        lastUpdateTime = now;

        if (residualCapacity > nominalCapacity)
                residualCapacity = nominalCapacity;
        else if (lessOrEqualNull(residualCapacity))
                residualCapacity = 0;

        if (dLastResCap != residualCapacity) {
                debugEV<< simTime() << " residual capacity = " << residualCapacity << " fill state is " << estimateResidualRelative()*100.0 << "%" << e
        }
        if(estimateResidualRelative() <=MODE_THRESHOLD_VALUE && isCoordinator){
                debugEV<<"host state changed to MODE_2"<<endl;
                hostState.set(HostState::MODE_2);
            }
            }
```

B.3 CCA Status Uptdate Function

```
void csma::updateStatusCCA(t_mac_event event, cMessage *msg) {
        switch (event) {
        case EV_TIMER_CCA:
        {
                 isIdle2 = phy->getChannelState().isIdle();
                 //The value for isIdle1 is set at the beginning of CCA duration : isIdle1 = phy->getChannelState().isIdle();
                 if(isIdle1&&isIdle2) {
                        updateMacState(TRANSMITFRAME_4);
                        phy->setRadioState(MiximRadio::TX);
                        macpkt_ptr_t mac = check_and_cast<macpkt_ptr_t>(macQueue.front()->dup());
                        attachSignal(mac, simTime()+aTurnaroundTime); //Min: there is no turn around for the radio, it is in TX state already
                        sendDelayed(mac, aTurnaroundTime, lowerLayerOut);
                        nbTxWuc++;
                        cMessage * m=macQueue.front();
                        m->setName("WUC_START_TX");
                        m->setKind(WUC_START_TX);
                        sendControlUp(m);
                } else {
                        // Channel was busy, increment 802.15.4 backoff timers as specified.
                        debugEV << "(7) FSM State CCA_3, EV_TIMER_CCA, [Channel Busy]: "
                        << " increment counters." << endl;
                        NB = NB+1;
                        if(NB> macMaxCSMABackoffs) {
                                // drop the frame
                                debugEV << "Tried " << NB << " backoffs, all reported a busy "
                                << "macMaxCSMABackoffs=" <<macMaxCSMABackoffs<< endl;
                                cMessage * mac = macQueue.front();
                                macQueue.pop_front();
                                txAttempts = 0;
                                nbDroppedFrames++;
                                mac->setName("MAC ERROR");
                                mac->setKind(PACKET_DROPPED);
```

```
                    debugEV<<"PACKET_DROPPED:"<<PACKET_DROPPED<<endl;
                    sendControlUp(mac);
                    manageQueue();
                } else {
                    // redo backoff
                    updateMacState(BACKOFF_2);
                    startTimer(TIMER_BACKOFF);
                }
            }
            break;
        }
        case EV_DUPLICATE_RECEIVED:
            debugEV << "(26) FSM State CCA_3, EV_DUPLICATE_RECEIVED:";
            if(useMACAcks) {
                debugEV << " setting up radio tx -> WAITSIFS." << endl;
                // suspend current transmission attempt,
                // transmit ack,
                // and resume transmission when entering manageQueue()
                transmissionAttemptInterruptedByRx = true;
                cancelEvent(ccaTimer);

                phy->setRadioState(MiximRadio::TX);
                updateMacState(WAITSIFS_6);
                startTimer(TIMER_SIFS);
            } else {
                debugEV << " Nothing to do." << endl;
            }
            //sendUp(decapsMsg(static_cast<macpkt_ptr_t>(msg)));
            delete msg;
            break;

        case EV_FRAME_RECEIVED:
            debugEV << "(26) FSM State CCA_3, EV_FRAME_RECEIVED:";
            if(useMACAcks) {
                debugEV << " setting up radio tx -> WAITSIFS." << endl;
                // suspend current transmission attempt,
                // transmit ack,
                // and resume transmission when entering manageQueue()
                transmissionAttemptInterruptedByRx = true;
                cancelEvent(ccaTimer);
                phy->setRadioState(MiximRadio::TX);
                updateMacState(WAITSIFS_6);
                startTimer(TIMER_SIFS);
            } else {
                debugEV << " Nothing to do." << endl;
            }
            sendUp(decapsMsg(static_cast<macpkt_ptr_t>(msg)));
            delete msg;
            break;
        case EV_BROADCAST_RECEIVED:
            debugEV << "(24) FSM State BACKOFF, EV_BROADCAST_RECEIVED:"
           << " Nothing to do." << endl;
            sendUp(decapsMsg(static_cast<macpkt_ptr_t>(msg)));
            delete msg;
            break;
        default:
            fsmError(event, msg);
            break;
        }
}
```