Energy-Efficient Wireless Medium Access Control Protocols for Specknets

Kai Juan Wong

Doctor of Philosophy

Institute for Computing Systems Architecture
School of Informatics
University of Edinburgh

2007

ABSTRACT

A Speck is intended to be a miniature device (measuring 5x5x5mm) that integrates sensing, processing and wireless networking capabilities. Given the small form-factor, each Speck will have limited resources in terms of energy supply and memory. A network of Specks is called a Specknet, and the collaborative processing carried out on such a Specknet is termed as Speckled Computing. This thesis focuses on the communication issues in the Specknet and is divided into three main areas of research: the physical aspects, the Medium Access Control (MAC) layer and the networking layer.

For the physical aspects, as the miniature Speck is still being realised, research began with the selection of the wireless communication medium followed by the development of a larger physical Speck prototype - the ProSpeckz. The ProSpeckz allowed algorithms targeted at Specks to be developed and analysed in the absence of the actual devices. A number of demonstrators for the ProSpeckz were developed to validate the prototype followed by the construction of an experimental platform - the PerSpeckz-64, which consisted of 64 ProSpeckz placed in a 8x8 grid to enable researchers to remotely execute and monitor algorithms on a network of ProSpeckz.

In the case of the MAC layer, the focus was to develop and analyse MAC algorithms that enabled Specks to extend their lifetimes by duty-cycling the radio. A collection of novel MAC algorithms, called SpeckMAC, was proposed and compared against B-MAC, a well known power-aware unsynchronized random-access MAC algorithm. Evaluations carried out using both mathematical models as well as physical implementation on the ProSpeckz and PerSpeckz-64 demonstrated that SpeckMAC outperformed B-MAC in terms of energy efficiency, delivery ratio and latency under both broadcast and unicast types of traffic.

Finally, research in the networking layer explored the possibility of employing wireless Mobile Ad-hoc Network (MANET) routing protocols on Specks given the underlying SpeckMAC algorithms. A hybrid algorithm, SpeckMAC-H, was also proposed to enable nodes to switch on-the-fly between the two versions of the SpeckMAC algorithms to optimise energy efficiency. All the SpeckMAC algorithms

were analysed using the Qualnet network simulator with Dynamic Source Routing (DSR) and Ad-Hoc On-Demand Distance Vector Routing (AODV) selected as the target MANET routing protocols. The simulations successfully demonstrated that it was indeed possible to employ MANET routing protocols with SpeckMAC as the underlying energy-efficient MAC algorithm.

ACKNOWLEDGEMENT

I would like to thank my wife, Nicole, for the loads of love and support that she had showered upon me these past few years, without which, the stress would had caused me to obtain another form of PhD (Permanent head Damage) before I had a chance to submit this thesis. I would also like to thank her for assisting me in vetting through all my publications, including this thesis, for grammatical errors even though the contents were totally alien to her and would have most probably bored her to tears.

I would like to thank my mum for all her love and care as well as everything she has done for me. She had been a pillar of support for me, seeing me through all the rough times and ensuring that all my needs were met in order for me to concentrate on completing this thesis.

I would like to thank my supervisor, D.K Arvind, for his support these last couple of years. This thesis would not have existed if not for his grand vision of Speckled Computing and his determination to achieve it. I would also like to thank the members and ex-members of the Consortium in Speckled Computing for their invaluable advice and comments especially: Matt and Hugh, who had to endure non-stop gibberish from me whenever I am exploring new ideas and algorithms; Charmaine Wilson, who ensured that all administrative matters were taken care of; and Allan Patterson, for all his insights into battery technology.

I would like to thank my employer and sponsor, Nanyang Technological University (Singapore), for their generous offer to send me to Edinburgh for my post-graduate studies. I would also like to thank Assoc. Prof. Lau Chiew Tong, for giving me the job in the first place, and Assoc. Prof Francis Lee, for his support and guidance in preparing me for an academic career.

Last but not least, I would like to thank the examiners for agreeing to part with their valuable time to examine this thesis amidst their busy schedules.

DECLARATION

I declare that this thesis has been composed by me and that the work contained herein is my own except where explicitly stated otherwise in the text. This work has not been submitted for any other degree or professional qualification.

However, some contents of this thesis had been published or accepted for publication in the proceedings of the following peer-reviewed conferences:

1. K.J Wong, D.K Arvind, *"The Design, Verification and Usage of a Simulator for a Class of Low-Power MAC Protocols", to appear in Proceedings of The IEEE International Conference on Wireless Communications, Networking and Mobile Computing (IEEE), September 2007*
2. K.J Wong, D.K Arvind, *"Experiments with Periodic Channel Listening MAC Algorithms for Specknets", to appear in Proceedings of The International Wireless Communications and Mobile Computing Conference (ACM), August 2007*
3. K.J Wong, D.K Arvind, *"SpeckMAC: Low-power Decentralised MAC protocols for Low Data Rate Transmission in Specknets", in Proceedings of The Second International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality (ACM SIGMOBILE), pp. 71-78, Italy, May 2006 (Acceptance Rate ~17%)*
4. K.J Wong, D.K Arvind, *"Perspeckz-64: Physical Test-bed for Performance Evaluation of MAC and Networking Algorithms for Specknets", in Proceedings of The Second International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality (ACM SIGMOBILE), pp. 122-124, Italy, May 2006*
5. K.J Wong, D.K Arvind, *"Specknets: New Challenges for Wireless Communication Protocols", in Proceedings of The International Conference on Information Technology and Applications (IEEE), vol. 2, pp. 728-733, Australia, July 2005 (Acceptance Rate ~25%)*

6. *K.J Wong, D.K Arvind, , N. Sharwood-Smith, A. Smith, "Specknet-based Responsive Environments", in Proceedings of The International Symposium on Consumer Electronics (IEEE), pp. 334-338, Macau, June 2005*

7. *R.McNally, K.J Wong, D.K Arvind, "A Distributed Algorithm for Logical Location Estimation in Speckled Computing", in Proceedings of The Wireless Communications & Networking Conference 2005 (IEEE), U.S.A, March 2005*

8. *D.K. Arvind, K.J.Wong, "Speckled Computing: Disruptive Technology for Networked Information Appliances", in Proceedings of The International Symposium on Consumer Electronics (IEEE), pp. 219-223, U.K, September 2004*

Furthermore, some of the work described in this thesis had also been presented during the following talks:

- "SpeckMAC – Novel Medium Access Control Methods for Specknet", 4th Workshop in Speckled Computing, U.K, September 2006
- "ProSpeckz Applications: Two Case Studies", 3rd Workshop in Speckled Computing, U.K, May 2005
- "Maintaining Logical Location in Specknets", 6th IEEE workshop on Mobile Computing Systems and Applications, U.K, December 2004
- "Communication Protocols for Speckled Computing", 2nd Workshop in Speckled Computing, U.K, September 2004
- "ProSpeckz III – Design and Implementation", 2nd Workshop in Speckled Computing, U.K, September 2004
- "Wireless Communications in Speckled Networks 'For dreams to realization'", ICSA Lunchtime Talk, U.K, July 2004

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

"An introduction to Specks, Specknets and Speckled Computing as well as an overview of this thesis"

1.1 Background

As we enter into the digital age, many aspects of our lives are now integrated with computing capabilities. Personal Digital Assistants (PDA), mobile phones and Personal Computers (PC) have grown to become indispensable in our daily life, regardless of whether it is in the office or the home environment. These devices provide intelligent processing and storage of information that enhances our quality of life. Whether it is the PDA bulging in our pockets or the computer sitting on our desks, these platforms provide a form of 'visible' intelligence where the user is actively aware of the presence of such devices. The dawn of a new age in computing seeks to minimise or even hide the presence of such intelligent devices and a way forward to achieve the objective of the 'vanishing' or 'invisible' computer could be through the miniaturisation of the current computing devices such that they would be hardly noticeable to the human eye. These miniature devices, when embedded with sensors and wireless capabilities, would combine the once separated worlds of computing, sensing and communications, thus allowing the creations of ubiquitous and pervasive environments that would be filled with ambient intelligence. In the vision of the Research Consortium in Speckled Computing [1], it is envisaged that 'Specks', 'Specknets' and 'Speckled Computing' would be at the frontier of technology to enable the realisation of such environments.

A Speck is designed to integrate sensing, processing and wireless networking capabilities in a minute (ultimately 5mm x 5mm x 5mm) semiconductor grain. Specks are intended to be autonomous, each with a renewable energy source, and can be mobile if needed. Thousands of Specks, scattered or sprayed on any person or

surface, will collaborate as programmable computational networks called Specknets. Computing with Specknets, or Speckled computing, will enable linkages between the material and digital worlds with a finer degree of spatial resolution than hitherto possible.

Specknets are intended to be a generic technology for ubiquitous computing, where data is sensed, processed and information extracted in situ in a collaborative fashion. Sensing and processing of information will be highly diffused in a Specknet; the person, the artefacts and the surrounding space become at the same time computational resources and interfaces to those resources. Surfaces, walls, floors, ceilings, articles, and clothes, when sprayed with Specks (or "Speckled"), will be invested with a 'computational aura' and sensitised post hoc as props for rich interactions with the computational resources. Given their minute sizes, Specks will have the ability to bring sensing and computation to places hitherto unreachable and wireless communication will enable Specknets to be the first (last) millimetre of the world-wide web.

Figure 1-1 gives a system-level overview of the individual Speck and the Specknet. It is intended that Specks will be programmable, with the Specknet operating as a fine-grained distributed computation network, employing lightweight and energy-efficient communication protocols. The Speck, though modest in terms of processing and storage resources, can be powerful as part of a collective system when harnessed as a Specknet. Specks would process their own sensor data and report only results and summaries externally. Limited individual processing power implies that Specks would need to organise the required processing collectively within a Specknet. Thus, a new model of distributed computation would have to be developed which will take into account some specific attributes of Specknets, such as unreliability of communication, a higher than normal failure rate of Specks due to harsh operating environment and very large volume manufacturing.

The means of wireless communication between Specks would be supported by either optics (for example, laser and infrared) or radio, or a combination of both. Specknet presents unique problems for communication protocols due to the need to support

networks with high nodal density while taking into consideration the extremely limited energy resources available on each Speck. Novel distributed algorithms that are highly energy-efficient would have to be designed at the Medium Access Control (MAC) layer to allow wireless communications to be sustained over an extended period of time given the limited energy resources.

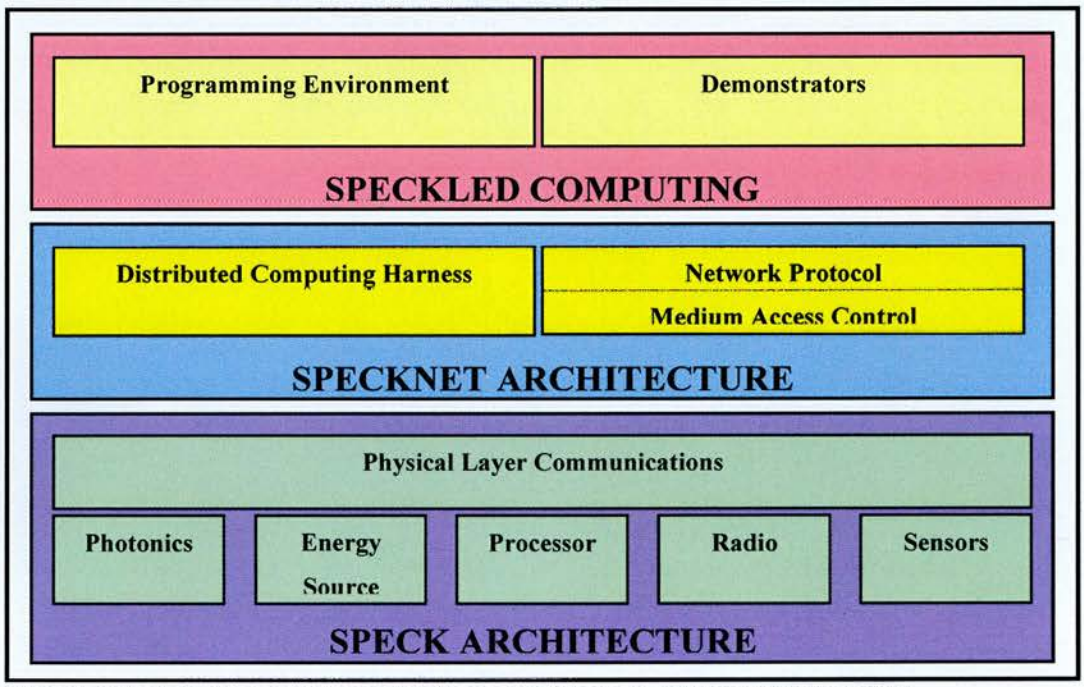


Figure 1-1: System-level overview of the Speck, the Specknet and Speckled Computing

Speckled Computing differs from the Smart Dust [2-4] project at the University of California (Berkeley) in several ways. Speckled computing focuses on providing a platform for distributed computing in situ within the network (thus, Speckled Computing), whereas Smart Dust focuses on providing a platform for sensing applications (thus, Sensor Networks). Due to this, Smart Dust performs more data-centric operations whereas Specknets will perform more program-centric operations. Data-centric operations usually require periodic transfers of sensor data between the sensor nodes and some centralised sink node, whereas program-centric operations would require aperiodic transmission of program, control and data packets between Specks. Speckled computing also requires a higher degree of re-programmability than Smart Dust. This is because sensor networks are often task-specific where task

types and data attributes are known at the time of deployment, whereas in the case of Speckled computing, the goal is to enable the flexibility of deploying a Specknet as a programmable platform where the task is undefined at the time of deployment and new tasks can be programmed into the network post-deployment. Lastly, Smart Dust are sparse networks where the sensor nodes would communicate over distances of a few meters or longer, whereas Specknets are designed to be highly dense networks as Specks would only be communicating over small distances up to 10cm.

One example in which Specks will be useful is in the area of ambient intelligence for escape-and-rescue applications. For example, Specks can be deployed for fire detection system in a building and assist in the planning of escape routes for the trapped occupants as well enable rescue and fire extinguishing operations to be carried out in an efficient manner. When there is no fire, minimal communications occurs between Specks and the radios on the Specks are placed in a sleep state to conserve energy. However, once a fire is detected by one or more Specks, the network of Specks will have to respond very quickly and communicate with one another to plan for escape routes for the occupants. Actuators, such as optical emitters, embedded in the Specks will indicate the safest path out of the building taking into consideration the spread of the fire as well as the number and locations of the occupants. Fire-fighters could also make use of the wireless devices with wearable displays to communicate with the deployed Specks to determine the best way to extinguish the fire, detect for trapped occupants and monitor the spread of the fire. New Specks can also be scattered by the fire-fighters to replace Specks that were destroyed in the fire.

This intelligent fire system application demonstrates several requirements that have to be supported by the algorithms and protocols designed for the Specks. Specks must have small form-factors so that they can be deployed easily and non-intrusively in the building. With small form-factors, fire-fighters could just spray Specks onto the walls or scatter them onto the floor to assist in their operations. However, having a small form-factor limit the resources available on each Speck; therefore, algorithms developed for Specknets must be able to operate under limited memory resources

and processing capability. The memory constraint limits the size of the packets that can be transmitted while the processing capability limits the complexity of the algorithms that can be supported. Specks have to be battery powered as any main electrical supplies in the building might be destroyed in the fire. Therefore, the communication protocols must be able to function in an energy-efficient manner to maximise the battery lifetimes. It can be assumed that communications between Specks are aperiodic as the need for communications changes based on the environment. For example, when no fire is detected; minimal communication occurs, however, when a fire is detected, the rate of communication will increase to support the rescue and evacuation operations. Given their small form factors, a large number of Specks can be used to monitor the building and determine the best escape routes for the occupants in a distributed fashion. Finally, Specks must be able to join a Specknet rapidly in an ad-hoc fashion as Specks can be destroyed by the fire and new Specks can be deployed to assist the fire-fighters. Wireless communication must also be maintained between the Specks and the mobile devices worn by the fire-fighters. Therefore, the communication protocols must support the mobile and ad-hoc nature of this Specknet application.

Speckled Computing is communication research at the extreme; it presents enormous challenges to the design of communication protocols for Specknets. The protocols would have to be highly adaptive, scalable and energy-conscious for dense networks comprised of nodes with very limited energy, computational and memory resources.

1.2 Objective and Motivation

The main objective of this thesis is the design, implementation and evaluation of energy-efficient communication protocols for Specknets that would extend the lifetime of the batteries in the Specks. For this thesis, the main source of energy savings would come from the algorithms at the MAC layer. The MAC layer was chosen because the MAC layer isolates the hardware details at the physical layer from the software/firmware at the network layer and beyond. This allows the energy-efficient MAC algorithms proposed to be ported easily to other physical platforms without the need to modify the protocols, middleware and programs at the higher layers.

Given that all Specks have energy resources in the same order of magnitude and there will be no central controller, thus neither synchronous nor centralised MAC protocols would be suitable as any Speck chosen as the controller would deplete its energy resources faster than the others. Therefore, the class of MAC protocols to be designed will be distributed and asynchronous. Furthermore, since communications between Specks are not necessarily periodic, random-access MAC algorithms would be more suitable for Specknets.

At the time when this research was carried out, the only energy-saving, distributed, asynchronous and random-access MAC algorithm designed for sensor networks that was suitable for supporting communication in Specknets was the B-MAC protocol [5]. B-MAC employs the idea of 'preamble listening' to conserve energy whereby nodes would duty-cycle the radio receivers by periodically sampling the channel for activity. In order to transmit a packet, a node would transmit a long preamble to inform its neighbours of impending data. This allows B-MAC to support asynchronous and random-access transmission of data packets. However, a long preamble contains no useful data and the main motivation of this thesis is to investigate alternative approaches to replace the long preamble transmitted by B-MAC. A collection of novel MAC protocols, SpeckMAC, has been proposed where the long preamble has been replaced with either wakeup frames or retransmissions of the data frame. The advantages and improvements in energy-efficiency obtained

using such an approach over transmission of a long preamble is presented over the course of this thesis. An insight into the ability of the proposed SpeckMAC protocols to support routing algorithms at the network layer will also be investigated.

1.3 Methodology and Approach

The research in this thesis was carried out using a collection of methods including hardware implementation, mathematical modelling and software simulations. The hardware implementation, using the Speck prototypes and test-bed is described in Chapter 2. Evaluations using these prototypes and test-bed would provide feedback to researchers on the difficulties prior to the actual implementation of the communication protocols, thus reducing the amount of idealistic assumptions made during the design and evaluation process. The physical implementation also provides stronger correlation between the measured results presented in this thesis to the possible results that could be seen on the actual Specks. Mathematical models were developed to allow rapid comparison between different algorithms such that the feasibility analysis could be performed before the exhaustive task of physical implementation was carried out. The physical implementation and the mathematical models were closely coupled to ensure better accuracy of the evaluations performed. Results and operations observed from the physical implementation were used to improve the accuracy of the mathematical models, and at the same time, the results from the mathematical models were also used to verify the operations performed by the physical implementation. Both physical implementation and mathematical modelling were used extensively to measure the performances of the MAC protocols, SpeckMAC and B-MAC which are, described in Chapter 3. Software simulations, on the other hand, were used to allow performance evaluations to be performed in situations where it is not feasible or suitable for physical implementation to be carried out and are also in situations where it is difficult for mathematical equations to be correctly modelled. This is normally the case when performance evaluations are needed to be performed at the network layer. The need to perform evaluations based on a large number of nodes makes physical implementation impractical and expensive, while the presence of node mobility and the dynamic connectivity

between nodes makes verification of mathematical model difficult. Because of these reasons, software simulations were used for the network layer, as described in Chapter 4, to provide an insight on the ability for SpeckMAC in supporting selected reactive routing algorithms. To provide assurances on the correctness of the simulator, some MAC results from the physical implementation and the mathematical model were used to verify the output of the simulator. The simulation results obtained for the network layer would be described in Chapter 4.

1.4 Overview of Thesis

The thesis is organised in five chapters, with the middle three chapters corresponding to the bottom three layers of the seven-layer Open Systems Interconnection (OSI) Model [6], i.e., the physical, the MAC (data-link) and the network layers. A brief overview of the chapters is as follows:

Chapter 1: Introduction

Chapter 2: The Physical Layer, Prototypes and Test-Bed:

- a comparison between the use of radio and optical communication media
- the construction of the Speck prototype
- the development of some demonstrators using the Speck prototype
- the description of the 64-node physical test-bed constructed using the Speck prototype
- an insight into the implications of energy limitations on the Specks on supporting wireless communications

Chapter 3: The Medium Access Control Layer:

- the background information on the MAC layer and related work on energy-efficient MAC algorithms
- the development of the physical implementations and the mathematical models for B-MAC and SpeckMAC

- the description and performance evaluation of B-MAC and SpeckMAC using physical implementation and mathematical models
- battery lifetime measurements, based on two cells with different drain profiles, of the Speck prototypes running B-MAC and SpeckMAC protocols and executing a location-maintenance algorithm

Chapter 4: The Network Layer:

- the background information on networking routing algorithms for Mobile Ad-hoc Networks (MANETS)
- validation of the output of the simulator with the results from the physical implementation and the mathematical models of the MAC layer
- the performance evaluation SpeckMAC protocols supporting two reactive routing algorithms in static and mobile scenarios
- the description and evaluation of a hybrid version of the SpeckMAC protocol

Chapter 5: Conclusion

In addition, Chapters 2 to 4 contain sections covering the introduction, objective, scope, summary, discussion and future work for each specific layer.

1.5 Contribution of the thesis

The key contributions of this thesis were in the design, implementation and evaluation of a collection of energy-efficient communication MAC protocols for Specknets using the motivation of replacing the long preambles transmitted by B-MAC with either the wakeup-frames (SpeckMAC-B) or the retransmissions of the data frame (SpeckMAC-D). It has been successfully demonstrated that both SpeckMAC algorithms managed to achieve higher energy-efficiency than B-MAC

based on results obtained from the analytical models as well as the experiments performed on the physical test-bed. It was also shown that the increased energy-efficiency was realised without sacrificing other performance attributes, such as the transmission latency and delivery ratios, when compared against B-MAC: SpeckMAC-D had achieve higher delivery ratios and lower latencies than B-MAC whereas SpeckMAC-B had achieved lower energy wastage due to overhearing.

Other contributions presented in this thesis are summarized as follow:

- the design and implementation of ProSpeckz, the first Speck prototype, using commercially off-the-shelf components
- the implementations of demonstrators using the ProSpeckz platform
- the design and implementation of PerSpeckz-64, a test-bed to provide functionalities for evaluating potential Specknet algorithms and protocols in a controlled and systematic manner
- the description on the impact that drain profiles of the Polymer Li-ion cells and the coin cells have on the lifetime of the Specks
- the implementation of the SpeckMAC-B and SpeckMAC-D protocols onto the Qualnet simulator
- the verification of the simulator using results obtained from the ProSpeckz implementation
- the evaluation of a proposed cross-layer algorithm, SpeckMAC-H, which switches between the use of wake-up frames and data frame retransmissions based on the decision made on the higher layers of the communication stack
- the performance evaluations demonstrating that all variations of the SpeckMAC algorithms were able to support the DSR and AODV network protocols in an energy-efficient manner

Chapter 2

PHYSICAL LAYER, PROTOTYPES AND TEST-BED

"Selection of a suitable wireless medium for the Specks as well as the hardware design of platforms to assist in the development and validation of Specknet algorithms"

2.1 Introduction

Specks in a Specknet communicate wirelessly. The choice of the wireless media could be free-space optics or radio. Wireless prototypes and test-beds have to be implemented to allow researchers to design, develop and evaluate Specknet applications, algorithms and protocols while the miniature 5x5x5mm Speck is being realised.

2.1.1 Objective and Scope

The objective is to realise one or more physical platforms, assembled using off-the-shelf components, so that algorithms and protocols intended for Specknets can be designed, tested and evaluated before the miniature Speck is realised. The scope of this chapter does not include the development of novel wireless media [7-10]; instead, the interest was to explore the feasibility of existing wireless media and technologies for the Specks.

2.1.2 Chapter Overview

In this chapter, two widely used wireless media, radio and free-space optics, were introduced and radio was selected as the means of wireless communication for the initial Speck prototype. Next, several commercially-available sensor network nodes were considered and found to be unsuitable for the Speck prototypes. Therefore, the ProSpeckz (Programmable Specks using Zigbee Radio), was designed and

constructed, and several applications were developed to demonstrate its usability. To facilitate the evaluation of wireless applications and communication algorithms on the ProSpeckz platform, an array of 64 ProSpeckz were assembled in an 8 x 8 grid to form the PerSpeckz-64 test-bed. This test-bed enabled researchers to perform experiments from a remote site with the ability to repeat the experiments if required. The PerSpeckz-64 was also used extensively to evaluate the performance of the MAC algorithms presented in the Chapter 3. Finally, this chapter concludes by introducing the latest Speck prototype, the 5cube-OTS, designed by other members of the Research Consortium in Speckled Computing, and provides an insight into the power constraints for the Specks in the future.

2.2 Physical communication media for Specks

Two different wireless media, namely radio and free-space optics, could be used to support communications between Specks. Carrier-based radio communication systems communications depends on sinusoidal waves being transmitted continuously at some known frequency (the carrier frequency) and data is modulated onto this carrier frequency. Examples of such communication systems can be found in the implementations of Zigbee [11], Bluetooth [12] and the IEEE 802.11 standards [13]. Carrier-less radio systems, such as those utilising Pulse-Position Modulation (PPM) in Ultra-Wideband (UWB) [14, 15] systems, on the other hand does not require a constant carrier frequency; instead, data is transmitted by sending "impulses" of radio frequency signals that occupy very high bandwidth.. The Federal Communications Commission (FCC) [16] define UWB systems as "any device where the fractional bandwidth is greater than 0.2 or occupies 500MHz or more of the spectrum operating in a radio frequency band of between 3.1GHz and 10GHz". Carrier-less UWB is designed primarily for Personal Area Networks (PAN) [17-19] and allows for higher data rates at lower power consumption using simpler transceiver hardware architecture. At first glance, carrier-less UWB seems to be suitable for however, at the time of starting this thesis, this technology was still in the early stages of development and no standards had been finalised.

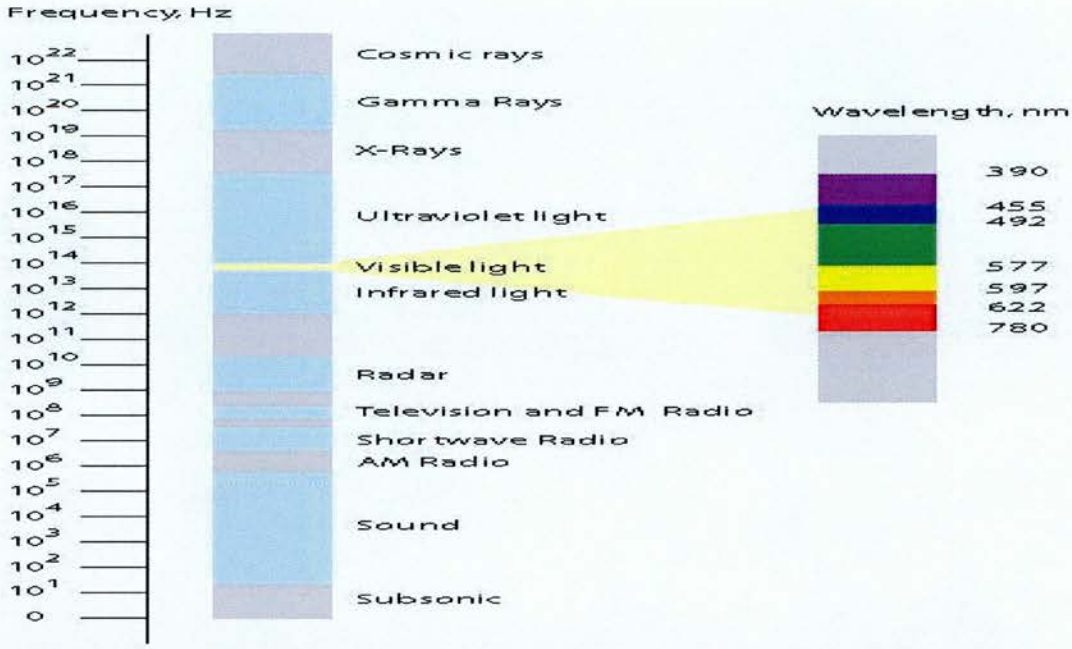


Figure 2-1: The frequencies utilised by the various transmission mediums (source: USByte.com)

Optical communication makes use of light pulses to transmit data. This provides an alternative means for Specks to communicate. Theoretically, the use of light as the communication medium should provide for higher data rates. This is because light, including infrared, visible light and ultraviolet light, has a shorter wavelength than radio. Given that the frequency of a media is the product of its wavelength and the speed of light, optical links have a higher frequency than radio links as shown in Figure 2-1. This allows optical communications to use a higher carrier frequency than radio, and therefore more data can be modulated using light as a medium. Optical systems can then run at lower duty-cycling rates to transmit the same amount of data as radio, thereby reducing the power consumed. However, having a shorter wavelength also has its disadvantages. This is because, the shorter the wavelength a medium has, the more susceptive the medium is to obstruction. This property allows radio to communicate through obstructions such as walls, whereas optical signals are reflected, diffused or adsorbed (blocked). Therefore, line-of-sight between the optical transmitter and receiver is a necessary requirement. Nevertheless, optical communications do have their advantages over radio. Firstly, in radio communication systems, the frequencies used are regulated very strictly by agencies

such as the International Telecommunication Union (ITU) [20], the European Telecommunications Standards Institute (ETSI) [21], and the Federal Communications Commission (FCC) [22]. 'Unregulated' frequency bands, such as the Industrial, Scientific and Medical (ISM) band [23], have very limited bandwidth and are congested with many applications, such as 802.11 and Bluetooth. On the other hand, there is an abundance of unregulated spectrum that can be used in optics. Furthermore, radio systems must make great efforts to overcome or avoid the effects of multi-path fading. Optical systems, on the other hand, do not suffer from these time-varying fades due to the line-of-sight requirement at the receiver. This simplifies design and increases operational reliability of optical links.

For this thesis, radio systems were selected as the preferred media for wireless communications as free-space optics could only support unidirectional communications and requires line-of-sight between the transmitter and receiver. Furthermore, preliminary analysis between the communication coverage supported by radio and optical systems (presented in Appendix A) indicated that radio communications suffered from less link changes due to the "circular" emission patterns of the transmitted signals, which in turn resulted in higher route stability between nodes when compared to the communication support by optical systems. Because of these reasons, radio was selected as the means of wireless communication for the initial Speck prototype.

2.3 ProSpeckz – The first Speck prototype

The need to construct a new Speck prototype instead of using existing commercially-available sensor network nodes was driven by the differences in the technical requirements of a Speck prototype from that of a typical sensor node. The Speck prototype should allow for sufficient hardware flexibility and at the same time, model as closely as possible the resources that would be available in the 5mm-cube Speck. It should provide facilities to easily emulate real-life Specknet environments and gather statistical results for analysing performances of different network protocols. It should also provide sufficient hardware peripherals to allow possible Speck applications to be developed for feasibility analysis.

Table 2-1 shows the details on some of the existing sensor nodes that were considered as possible candidates for the Speck prototype. The two major components of these devices; the radio and the processor, had to be carefully evaluated. The radio module had to model closely to what would be used on the Specks, while the processor must have sufficient hardware peripherals for interfacing with external devices such as different sensors.

	MICA2DOT	**Smart-its**	**IMote**	**TelosB**
Processor	ATmega128L	ATmega103L	ARM7TDMI	TI MSP430 16-bit RISC
Flash	128Kb	128Kb	512Kb	48Kb
RAM	4Kb	4Kb	64Kb	10Kb
Radio Freq	315/433/868/916 MHz	2.4GHz Bluetooth	2.4Ghz Bluetooth	2.4GHz IEEE 802.15.4
Power: **Sleep** **Idle** **Tx** **Rx**	@3V <48uW 24mW 99mW 48mW	@3.3V 11mW 50mW 94mW 94mW	@3.3V 0.83mW 10.9mW 112-122mW 112-122mW	@3V 18.3uW 5.4mW 35-71mW 69mW
Analog Functions	6 channel 10-bit ADC	8 channel 10-bit ADC	Add-on boards needed	8 channel 12-bit ADC, 2 DAC
Digital Functions	UART, 9 I/O	16-bit Timer, UART, 2 x 8 bit I/O	UART, USB, 8 bit I/O, SPI, I^2C	I2C, SPI, UART, 2 16-bit timers
Size	25mm Diameter	40mm x 60mm	28mm x 28mm	65mm x 32mm

Table 2-1: Attributes of some of the existing sensor nodes (circa 2003)

Given that Specks are targeted to have a form-factor of only 5x5x5mm, communicating at frequencies of 315-916MHz would not be appropriate due to the antenna size required as well as the larger components needed at the RF front-end.

Therefore, sensor nodes that utilise the Megahertz bandwidth such as the MICA2DOT [24] are unsuitable candidates for a Speck prototype. A more suitable frequency to be used for Specks is at 2.4GHz, as it is an ISM band and the gigahertz carrier allows for smaller antenna designs. Even though the Smart-its [25] communicate using a 2.4GHz radio, it is also an unsuitable candidate for a Speck prototype. This is because Bluetooth transceivers were used and requires a master node to coordinate communications in a piconet of seven slaves. This not only limits the topology that could be supported by the wireless network, it also imposes a hierarchical requirement for coordinator nodes (or master nodes) that will not be feasible in Specknets due to the ad-hoc and mobile nature of the network. This is because Specks that were selected as master nodes will deplete its limited energy resources quickly. Likewise, the IMotes [26, 27] uses Bluetooth communications and was therefore unsuitable as a Speck prototype. Furthermore, there are no analogue interfaces on the IMotes and additional add-on modules will have to be built to facilitate the implementation of possible Specknet applications.

The lack of a suitable Speck prototype, when the thesis work began in 2003, made the development of a new hardware prototype necessary. The ProSpeckz (Programmable Specks using Zigbee Radio) was designed with flexibility and ease of development in mind, and Figure 2-2 shows the different versions of the ProSpeckz during the development phase. The current version of the prototype, the ProSpeckz IIK-mini, uniquely combines the following components into a versatile development platform:

- An 802.15.4 compliant CC2420 [28] Zigbee radio chipset from Chipcon provides wireless communications up to data rates of 250kbps over 16 channels.
- An embedded 2.4GHz matched antenna [29] and filter circuitries allows software adjustable ranges from less than 30 centimetres to over 20 meters.

- A Programmable System-on-Chip (PSoC) [30] from Cypress Microsystems enables ProSpeckz to provide software reconfigurable analogue circuitries to external interfaces and components.
- The PSoC is also the processing core of the ProSpeckz providing an 8-bit micro-controller with 32Kbytes of FLASH and 2Kbytes of RAM.

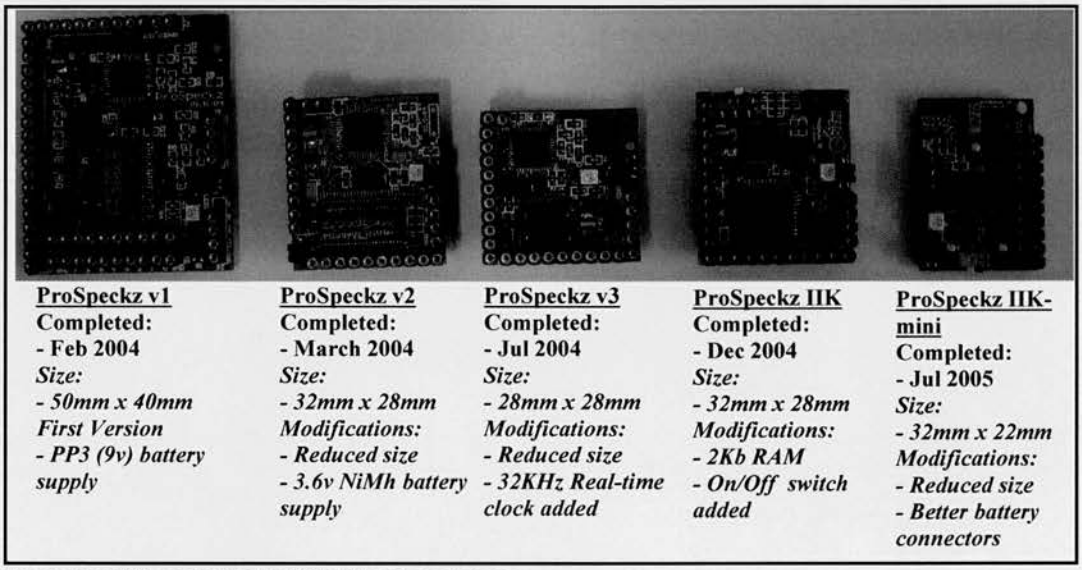


Figure 2-2: The evolution of the Speck prototypes

To allow MAC and networking protocols to be developed for Specknets, the wireless communication capabilities of the ProSpeckz is facilitated by the CC2420 Zigbee radio; however, the Zigbee standards (IEEE 802.15.4) for the MAC and networking layers were not used. The CC2420 was also used in another product, the TelosB [31], which was only commercially available only in the last quarter of 2004. Even though both TelosB and ProSpeckz use the same radio chipset, the main difference between the two lies in choice of the embedded processor. TelosB uses the TI MSP430F1611 [32] 16-bit RISC processor that has 48Kb of program Flash and 10Kb RAM. This low power processor appears to be an ideal processor for a Speck prototype; however, the embedded hardware peripherals on the processor are fixed and thus lack the flexibility needed for it to be an ideal experimental platform for rapidly exploring Speck applications and evaluating the performances of novel communication protocols. The ProSpeckz, on the other hand, uses a programmable

system-on-chip that provides greater hardware flexibility to the programmers by allowing the physical hardware modules and interconnections to be modified via software. An easy to use graphical user interface, the PSoC Designer® as shown in Figure 2-3, allows programmers to select different hardware modules to be inserted into the various analogue and digital blocks embedded within the PSoC.

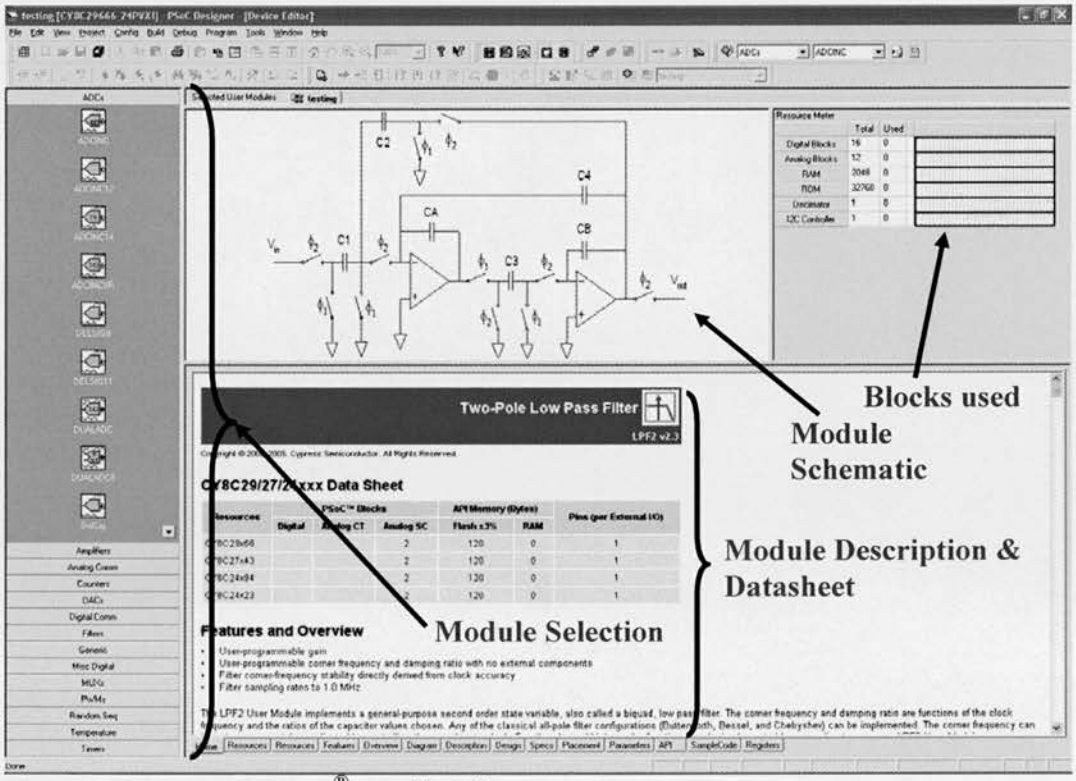


Figure 2-3: The PSoC Designer® user interface

There are 12 analogue blocks provided by the PSoC that could be configured to provide a combination of Analogue-to-Digital Converters (ADC), Digital-to-Analogue Converters (DAC), programmable gain amplifiers, filters and comparators that should be sufficient to satisfy the analogue signal processing requirements for a large number of Speck applications. For the digital logic and communication requirements, the PSoC provides 16 reconfigurable digital blocks that allows the programmer to easily construct one or more 8-32 bits timer/counters, Pulse Width Modulators (PWM), Cyclic-Redundancy Checks (CRC) modules, Pseudo Random Sequence (PRS) generators, 'Universal Asynchronous Receiver Transmitter'

(UARTs) and Serial Peripheral Interfaces (SPI). Furthermore, the additional ability to clock the various modules using different clock sources is especially useful as it allows the ProSpeckz to easily maintain different timers to simulate virtual battery lifetimes on the devices and at the same time, monitor networking statistics such as the processing times and the delays in packet delivery.

The use of a PSoC is unique in the design of wireless sensor devices. Platforms built by other research groups, after and during the development of the ProSpeckz, generally use processors that offer higher processing and memory capacities. Examples of such platforms include SunSPOT [33] (ARM 920T microprocessor, 512KB RAM, 4MB Flash), MicaZ [34] (ATMEGA-128 microprocessor, 4KB RAM, 128KB Flash), Eco [35] (8051 microprocessor, 4KB RAM, 32KB EERPOM), KMote [36] (MSP430 microcontroller, 10KB RAM, 48KB Flash), T-Mote Sky [37] (MSP430 microcontroller, 10KB RAM, 48KB Flash) and XYZ [38] (ARM/THUMB microcontroller, 32KB RAM, 256KB Flash). Using similar radio systems as the ProSpeckz, these platforms may surpass the ProSpeckz in terms of the processing capability and memory capacities; however these platforms lack the flexibility of the software-defined hardware modules provided by the ProSpeckz.

2.4 Interactive Environments: Demonstrators using the ProSpeckz

To demonstrate the usefulness of the ProSpeckz and validate it as a feasible tool for application development, several demonstrators were built. One of these demonstrators was the construction of interactive environments in consumer items such as furniture and electrical appliances. These demonstrators were designed in collaboration with Nick Sharwood-Smith and Andrew Smith from the Edinburgh College of Art and were showcased at an exhibition in Edinburgh in December 2004. A user survey was also carried out during the exhibition and the results from that survey are presented in Appendix D.4. Nick and Andrew were responsible for the artistic aspects of the demonstrators, such as the construction of the tables, mood cloud, etc, while the author of this dissertation was responsible for the engineering aspects, such as the electronics/electrical interfaces, designs and programming of the ProSpeckz. Other than the interactive environments, other ProSpeckz demonstrators

were also constructed. These demonstrators will not be discussed in this dissertation but details can be found in Appendix D.6.

2.4.1 Smart Furniture – A Behaviourally Responsive Environment

Normally passive artefacts such as furniture and appliances, such as the radio, were incorporated with ProSpeckz to enable them to interact with each other and respond to the users. The scenario of a living room shown in Figure 2-4 consists of the following furniture with enhanced capabilities:

(i) A reading table – when the user removes a book from the book rest, the table lamp would automatically turn on.

(ii) A smart-throw – when the user sits on the smart throw placed on a sofa or a chair, electrical appliances such as the TV and lamps, would turn on automatically.

(iii) A radio table – The radio embedded within the table automatically turns on when the remote-control is removed from the table. As the user moves away from the table with the remote control, the volume of the radio automatically increases, as vice versa.

Figure 2-4 also shows a simplified schematic for the smart furniture and the interactions between the different parts. Simple contact switches are used on the reading table and smart-throw to allow the ProSpeckz to detect the presence of a book and a person, respectively. Appliances, such as the reading lamp, television, and radio are controlled by relay switches activated by a remote ProSpeckz. This allows the user to easily switch between the appliances that are controlled via the reading table or smart-throw.

The radio embedded in the table is turned on directly using a normally-closed push switch that detects when the remote control is removed from the table. The remote control system for the radio is implemented using two ProSpeckz devices: one is interfaced to the radio which changes the volume as well as the radio station, and a remote ProSpeckz allows the user to remotely control the radio by sending wireless

commands to the ProSpeckz that is local to the radio. In addition, the remote ProSpeckz broadcasts a beacon every second which allows the local ProSpeckz to gauge the distance of the remote ProSpeckz based on the received signal strength information of the beacon, and modifies the volume of the radio accordingly.

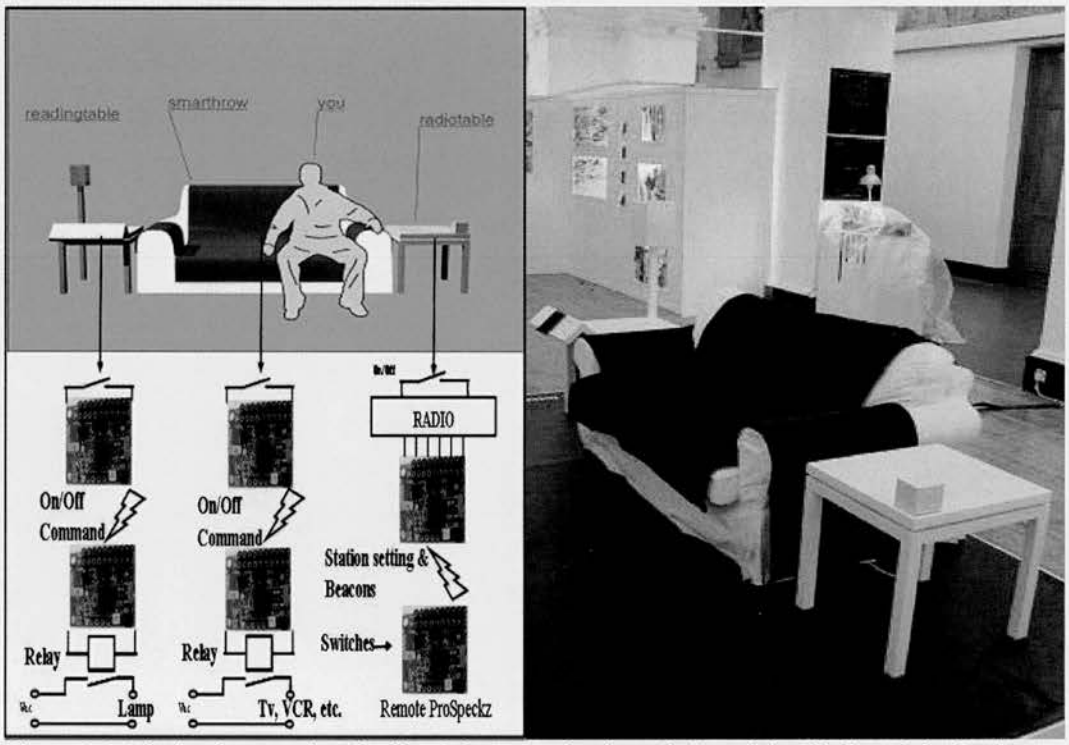


Figure 2-4: The implementation details of the smart furniture (left) and the display of the furniture at an exhibition in Edinburgh (right)

2.4.2 The Mood Cloud – An Emotionally Responsive Environment

The concept behind an emotionally responsive environment is to allow the immediate environment surrounding a human to react accordingly to the person's mood. In this particular scenario, a ProSpeckz in the computer keyboard monitors the heartbeat of the user, and the lighting in a remote mood cloud adjusts itself relative to the stress level inferred from the user's heartbeat. As depicted in Figure 2-5, as the stress level of the user increases, the light in the mood cloud brightens and at a prefixed maximum stress level, the mood cloud starts to flash suggesting a break away from the computer. It could also be used as part of a bio-feedback system to reduce the person's stress level by playing soothing music.

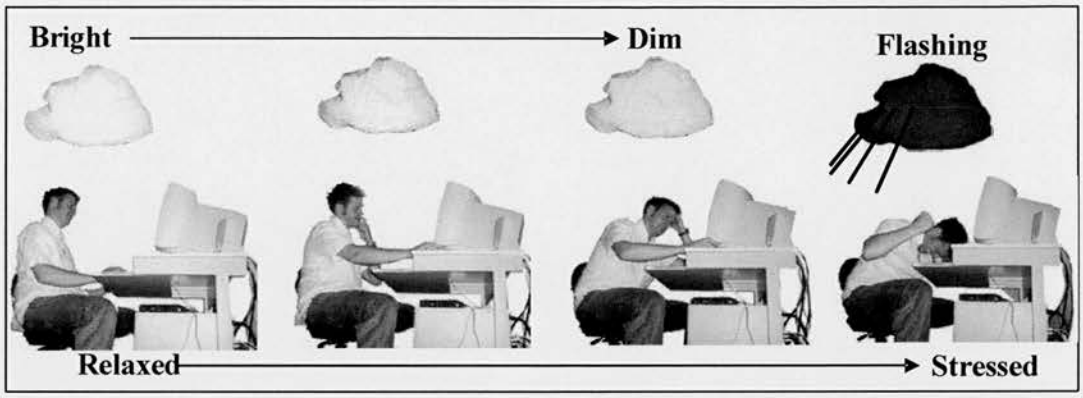


Figure 2-5: The operations of the Mood Cloud

Figure 2-6 shows the schematic of this application. Two copper pads on the palm rest of the keyboard are connected to an inexpensive heartbeat monitor. These monitors are usually tied across the body of the user to determine the heartbeat when exercising. As these monitors are used to detect the occurrence of a heartbeat instead of providing an accurate electrocardiograph (ECG), there is no third lead or 'right leg drive' to reduce the common-mode noise. Instead, a simple peak detector is used to determine the presence of a pulse. The output of this detector is sampled by a ProSpeckz that monitors the changes in the rate of the user's heartbeat. For different thresholds, a command packet is sent to the remote ProSpeckz to change the level of lighting in the mood cloud provided by three white lamps and a flashing strobe. These lights are powered directly from the mains and are turned on using relays activated by the remote ProSpeckz.

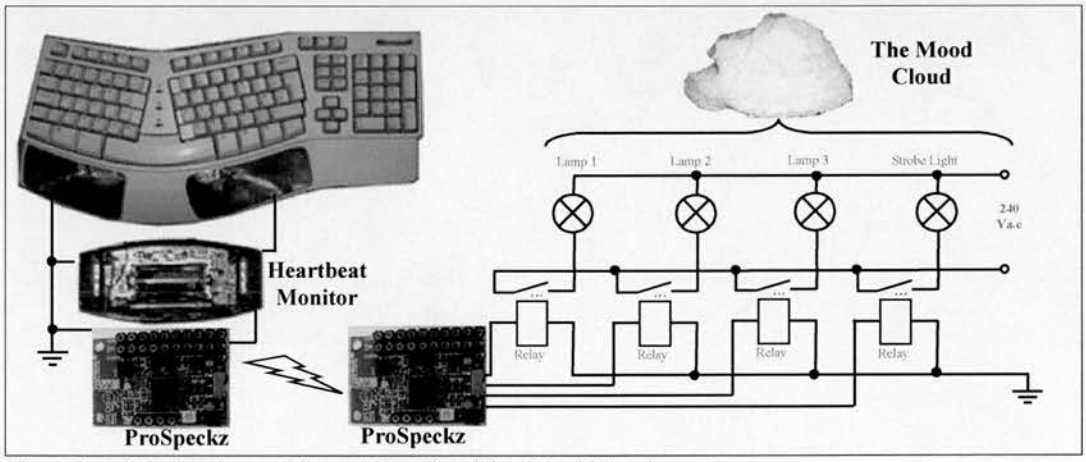


Figure 2-6: The implementation schematic of the Mood Cloud

2.5 PerSpeckz-64: A wireless test-bed for Specknet applications and algorithms

Section 2.4 described two early demonstrators employing the ProSpeckz prototype. However, in order for the ProSpeckz to be used to measure the performances of applications and algorithms for Specknets, a test-bed had to be developed so that realistic experiments could be performed on a wireless network of ProSpeckz in a controlled and repeatable fashion. To facilitate this, an array of 64 ProSpeckz nodes arranged as an 8 x 8 grid, with a 10cm separation and housed in a Perspex enclosure, as shown in Figure 2-7. This test-bed, PerSpeckz-64, can be accessed over the internet to carry out experiments from remote sites and the test-bed has the following features:

- The size of the physical network can be varied from 1 to 64 ProSpeckz, where each ProSpeckz can be individually selected to be turned on or off
- The connectivity of the network and the network topology can be changed by altering the radio transmission ranges of the individual ProSpeckz
- New network scenarios and algorithms can be programmed wirelessly onto the ProSpeckz
- Statistics, such as radio and processor usage information, can be collected systematically for the MAC and networking algorithms
- The power consumed by the network can be accurately measured and stored via a PC-based oscilloscope.
- Batch files to automate the experiments to be performed

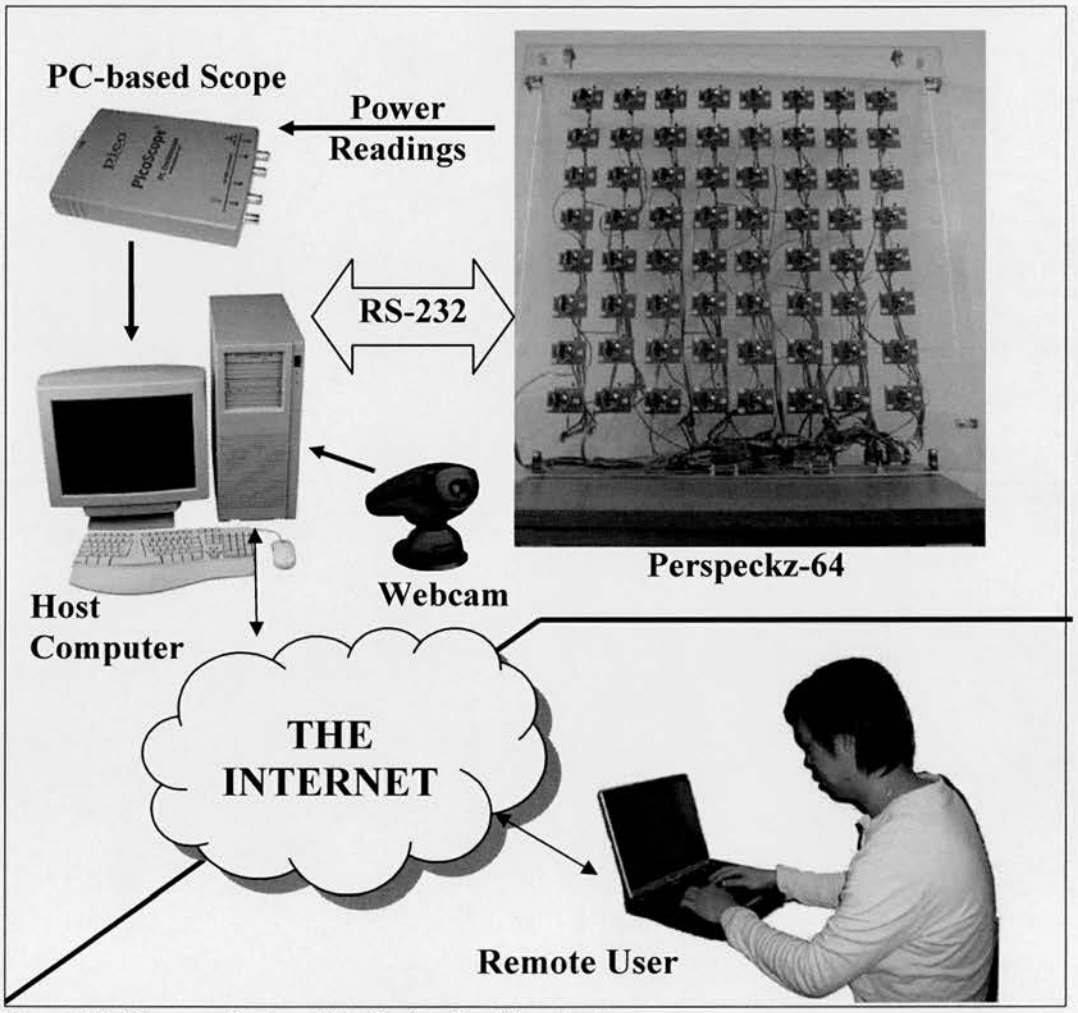


Figure 2-7: The organisation of the PerSpeckz-64 test-bed

Figure 2-8 gives an overview of the firmware programmed onto the array of ProSpeckz for performing experiments. The topmost layer, the traffic generator, is user-programmable and mimics different network traffic scenarios, such as a constant/variable packet generator. The network and MAC layers contain the algorithms and protocols under study. The statistics layer has a book-keeping role of noting the number of packets sent and received as well as tracking the radio and processor usage. A boot-loader enables modifications to be made to the MAC and networking algorithms, which is programmed wirelessly onto the ProSpeckz. Furthermore, it also allows the traffic generator and the statistics layer to be modified. The boot-loader occupies the highest 4 Kilobyte of the FLASH memory

(actual usage is 2628 bytes) and 84 bytes of RAM. As the boot-loader has its own MAC/networking layer, it is not dependent upon the MAC and networking algorithms under study.

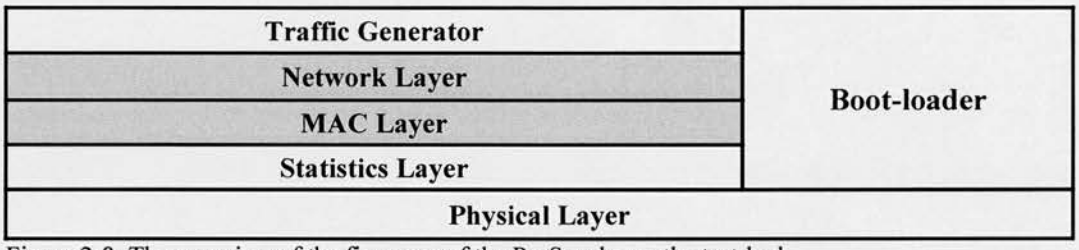


Figure 2-8: The overview of the firmware of the ProSpeckz on the test-bed

Figure 2-9 shows the different components of the ProSpeckz-64. Two different power supplies were used to isolate the control circuitry of the Perspeckz-64 from that of the ProSpeckz. This enables the current consumption of the ProSpeckz, either individually or as a network, to be monitored accurately using a PC-based oscilloscope. Each ProSpeckz was attached to a relay board and can be powered by toggling the relay that is controlled by a central controller which is capable of addressing each relay board individually using a multiplexing board. This allows scenarios with different network configurations and densities to be emulated. The network topology can also be modified by changing the wireless communication radius supported by each ProSpeckz. This is done, via software, by adjusting the radio transmission strength on each ProSpeckz to a value ranging from -25dbm to 0bdm, thus enabling the ProSpeckz on the test-bed to transmit packets with a range from 10cm up to 10m.

In addition to controlling the power supply, the central controller is also responsible for the wireless programming of all the ProSpeckz on the test-bed, as well as any other ProSpeckz nodes deployed in the vicinity. The central controller interfaces to a remotely accessible host computer through a RS-232 serial link, thereby allowing researchers to design, program and download MAC and networking algorithms onto the physical devices remotely. Finally, the central controller polls each ProSpeckz for statistical information at the end of each experiment and transfers them onto a log

file on the host computer. This log file can then be accessed remotely to assess the performance of the algorithms.

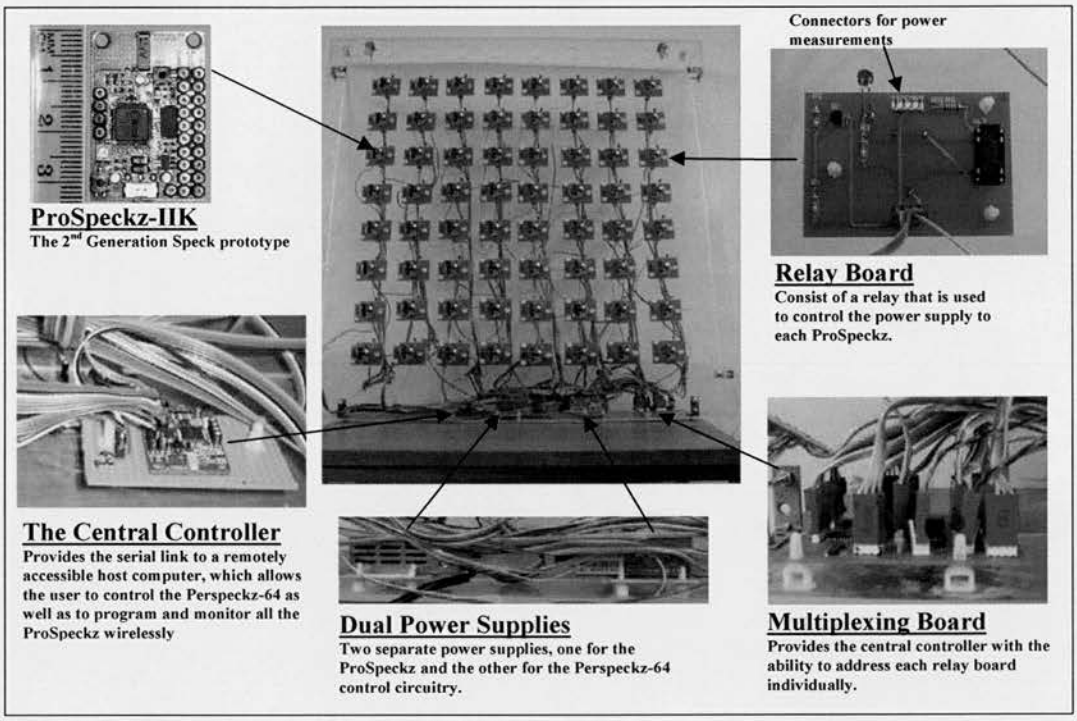


Figure 2-9: The Perspeckz-64 and its constituent components

Two DOS-based commands, 'progme.exe' and 'results.exe' are provided on the host computer to allow reprogramming, controlling and monitoring of the PerSpeckz64 via the RS-232 serial link. The functions of these two commands are described as follows:

- ***progme ['parameter'='value']...*** allows new program codes to be transferred from the host computer onto the controller board, which will in turn, program the selected ProSpeckz wirelessly via the bootloader. This command can also be used to program 8-bit or 32-bit network addresses, which would be stored at the ROM address 0x7FC0, onto the ProSpeckz. The parameters for this command are shown in Table 2-2.
- ***results ['parameter'='value']...***is used to provide automated control over the experiments as well as record the statistics generated via the central controller. Using this command, the user is able to specify the number of

iterations for an experiment. The central controller coordinates the start of each execution to ensure that the selected ProSpeckz on the PerSpeckz-64 are all powered at the same time. At the end of each execution, the central controller polls the statistics layer of the ProSpeckz participating in the experiment, and transfers the gathered information to the host computer. The parameters for this command are shown in Table 2-3.

Parameter	**Description / value**
c	The serial port on the host computer used to connect to the Perspeckz-64. E.g. COM1
r	Name of the .rom program file.
p	The poll time, in seconds, used by the central controller to poll all ProSpeckz into boot-loader mode. The default is 10 seconds.
n	The number of ProSpeckz to turn on. ProSpeckz are selected randomly. Range from 1 to 64. Default is 64.
P	The name of a power file, which is a text file, listing the grid addresses of the ProSpeckz to power on.
D	The grid address of a single node to power on. Range from 0 to 63.
a	Program a 32-bit network address onto the ProSpeckz. Typically used with the 'D' parameter
s	Program an 8-bit network address onto the ProSpeckz. The higher 24-bits of the network address are randomly assigned.

Table 2-2: Parameters for 'progme.exe'

Parameter	**Description / value**
c, p, P,n	Same as progme.exe from Table 2-2
r	Number of times the code has to be executed
f	Filename of the log file that will be used to store the captured statistics
t	Duration to run each iteration for
s	Maximum number of ProSpeckz that is involved in the statistics collection
l	Show long address (32 bits) in the captured statistic file if l=1, otherwise, short addresses (8 bits) are used.

Table 2-3: Parameters for 'results.exe'

Using both these commands on the host computer, users can create batch files to run different experiments iteratively without human intervention. For example, the instructions in the batch file in Figure 2-10 assigns 8-bit network addresses to each ProSpeckz in the Perspeckz-64, based on its grid address before it evaluates the performances of two MAC algorithms located in the program files *'BMAC.rom'* and *'SpeckMAC.rom'*. Each algorithm would be executed for 20 iterations, each lasting 20 seconds, with 15 ProSpeckz randomly turned on for every execution. Figure 2-11

shows a snapshot of statistical results collected from the PerSpeckz-64 based on one of the iteration on the experiment performed.

```
FOR /L %%I IN (1,1,10) DO progme c=COM4 D=%%I s=%%I
progme c=COM4 r=BMAC.rom
results c=COM4  n=15 r=20  f=bmac t=20.0 s=15 l=0
progme c=COM4 r=SpeckMAC.rom
results c=COM4  n=15 r=20  f=SpeckMAC t=20.0 s=15 l=0
```

Figure 2-10: An example of the instructions in a batch file

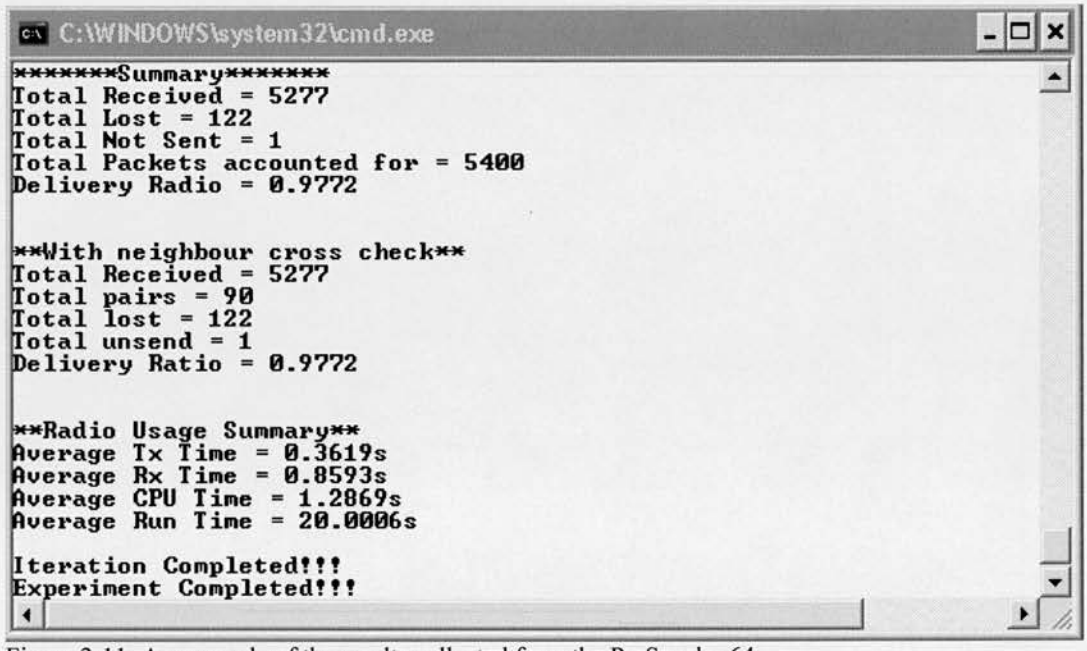


Figure 2-11: An example of the results collected from the PerSpeckz-64

2.6 Summary and Discussions

The main contribution of the work performed in the physical layer was in the development of the Speck prototype, as well as the implementation of a physical test-bed to enable applications and algorithms for Specknets to be evaluated. The ProSpeckz is a Speck prototype constructed using the CC2420 Zigbee radio for communications and a PSoC as the processor. The use of a PSoC instead of a common microprocessor allowed digital and analogue interfaces embedded on the ProSpeckz to be configured via software, thereby reducing the need for additional external interface boards to be constructed for different applications. To demonstrate

the usability of the ProSpeckz, two different demonstrators were constructed and described in Section 2.4. However, in order for the ProSpeckz to be used as a platform for evaluating applications and algorithms for Specknets, a test-bed, the PerSpeckz-64, was developed. Using this test-bed, a user is able to perform experiments on a wireless network of ProSpeckz remotely. This test-bed also allows test scripts to be written so that the experiments can be executed without further human intervention.

2.7 Future work

The ultimate objective is to create a 5mm x 5mm x 5mm Speck, therefore, the future work in this layer will the miniaturisation of the hardware device. This work is currently carried out by other members in the Consortium in Speckled Computing and the circuit board for the first miniaturised Speck, which is constructed using off-the-shelf components, had been fabricated and assembled in May 2006. This Speck prototype, named the 5cube-OTS, is shown in Figure 2-12. The 5cube-OTS was assembled using a radio module from ZarLink [39] and a low-power microprocessor, the MSP430F123 [40], from Texas Instruments. The next stage of the development would be the packaging of the 5cube-OTS to incorporate the miniature ML414 rechargeable batteries from Sanyo [41] with the assembled circuit board in a deployable fashion. A mock-up image for the completed 5cube-OTS is also shown in Figure 2-12

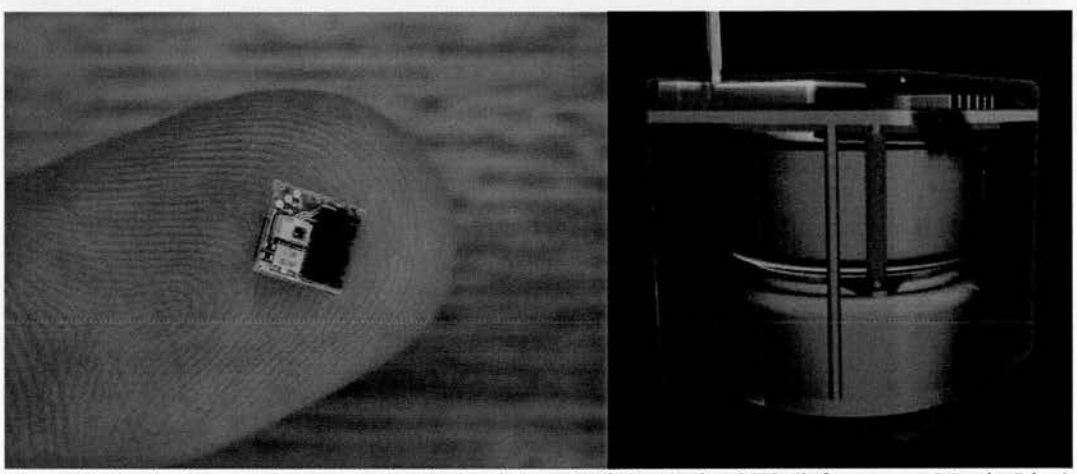


Figure 2-12: A picture of the completed circuit board of the 5cube-OTS (left, source: Martin Ling) and a mock-up image of the final packaged product (right, source: Hugh Leather)

The constraints posed by the limited energy resources on the 5cube-OTS is demonstrated by an experiment carried out, in collaboration with Allan Paterson from the University of St. Andrew's, to determine the estimated life expectancy of a 5cube-OTS operating without any energy-efficient communication protocols. Without energy-efficient protocols, the radio transceiver on the 5cube-OTS would be turned on all the time resulting in a current of 5mA being drawn from the batteries constantly. To mimic this scenario, two fully charged ML414 batteries were placed in parallel and 5mA was drawn from the batteries. This experiment was carried out using a highly precise primary cell test system from Maccor [42] and the result obtained from the test system is shown in the Figure 2-13. From the figure, it could be seen that when a 5mA load was placed onto the battery (at 0.5min), the potential across the batteries dropped drastically from 2.8v to about 2.2v. The voltage then continued to decrease rapidly towards 1.8v. Using the specification for the various components on the 5cube-OTS, the device would stop operating when the voltage supply drops below 1.8v. Thus, with the radio transceiver kept on all the time, the expected operational lifetime of the 5cube-OTS will be only 36.6 seconds long based on the measurements captured. This limited lifetime restricts the number of useful applications in which the 5cube-OTS could be used for. To overcome this restriction, power-efficient communication protocols would have to be developed to extend the lifetime of the Specks by allowing the radio transceiver to be turned off as frequently as possible. Examples of such energy-conserving MAC protocols will be discussed in greater details in the next chapter.

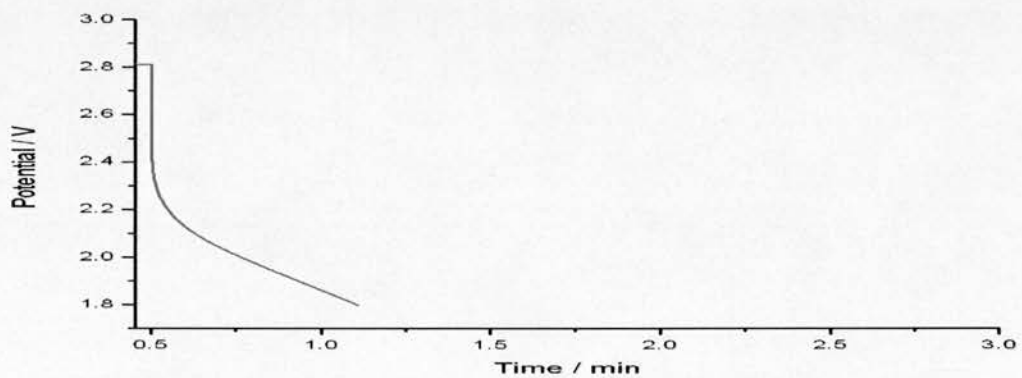


Figure 2-13: The graph showing the voltages of two ML414 batteries (connected in parallel) when a 5mA current is drawn from it (source: Allan Paterson)

Chapter 3

THE MEDIUM ACCESS CONTROL LAYER

"Research, design, implement and evaluate energy-efficient, distributed, asynchronous, random-access MAC algorithms, through the use of mathematical models and physical prototypes, to support Specknet applications with low data transmission requirements without the need for a master node."

3.1 Introduction

Medium Access Control (MAC), as the term implies, controls access to the physical medium. The algorithms and operations of the physical layer only ensure that signals transmitted from a source node can be delivered to the destination nodes through a shared physical medium. However, the physical medium is a limited resource used by a number of nodes, and therefore coordination for efficient access to the medium is important.

3.1.1 Assumptions and limitations

The following requirements were identified regarding applications to be supported by the MAC algorithms running on Specknets:

- **Absence of master nodes:** A single point of coordination, such as data sinks in sensor networks [43] and access points in 802.11 infrastructure networks [44] , are nodes endowed with greater resources compared to the rest of the nodes in the network. These master nodes provide coordination, as well as additional processing and storage capabilities for nodes in the rest of the network. In this thesis, the MAC algorithms were designed to operate in homogeneous networks of Speck nodes.
- **Low data transmission rates:** Wireless communication between Specks is expensive, given the low bandwidth radios used and severely limited energy and storage resources on each Speck. Thus, for this thesis, it was assumed

that Specknet applications would transmit packet infrequently and Specks would not transmit more than one packet per second, with each packet not exceeding a data payload of 32 bytes, excluding the necessary MAC headers.

- **Sporadic data transmission:** Unlike typical sensor networks where sensors periodically transmit sensed data to a sink node, data transmissions in Specknets were assumed to be sporadic. There could be no need for any data transmissions in a Specknet for an extended period of time until the occurrence of an event. For example, in a fire detection system, Specks deployed in a building are placed in an idle state until a possible fire is sensed by one or more Specks. Communications between Specks is then performed to enable some form of distributed computing to illuminate a safe escape route out of the building.

3.1.2 Objective and Scope

The objective for the MAC layer was to design, implement and evaluate MAC algorithms for Specknets with the following attributes:

- **Energy-efficient:** Efficient usage of energy is paramount for MAC algorithms in a Specknet. This can be achieved by reducing redundant access to the medium by switching the radio on only when necessary.
- **Distributed:** The MAC algorithm must be distributed across a network of Specks similar in their processing capabilities, memory space and energy storage.
- **Asynchronous:** The MAC algorithms should be asynchronous and not requiring the clocks on the Specks to be globally synchronised.
- **Random-Access:** Specknets are mobile, ad-hoc networks with requirements for event-driven, aperiodic data transfers. The MAC algorithm should therefore be random-access, rather than based on a pre-determined schedule, allowing the Specks to transmit and receive frames without prior communications with other Specks.

3.1.3 Chapter Overview

The chapter begins with an overview of the MAC layer in the communication protocol stack: background, challenges and performance criteria. A survey of existing MAC algorithms is presented and their suitability for Specknet is analysed. Two novel energy-efficient, distributed, asynchronous and random-access MAC algorithms, SpeckMAC-B and SpeckMAC-D, are proposed and compared against the existing B-MAC [5] algorithm, an energy-efficient MAC designed for sensor networks that would also be suitable for Specknets. Both SpeckMAC algorithms shown improvements over B-MAC in terms of energy efficiency, which was demonstrated using two approaches: based on analytical models of the algorithms and performance measurements on the PerSpeckz-64 test-bed. Furthermore, compared to B-MAC, SpeckMAC-B suffers from less energy wastage caused by overhearing transmissions, and SpeckMAC-D exhibits lower latency in communications and higher rate of success for packet delivery. Both versions of SpeckMAC also conserve more energy when compared to B-MAC when there is interference on the communication channel. Experiments were devised to measure the impact of the three MAC algorithms on the battery lifetime for implementation of a location maintenance algorithm [45] on the ProSpeckz. The results demonstrated that ProSpeckz running both versions of SpeckMAC achieved longer lifetimes for two types of batteries compared to B-MAC. Furthermore, it was also noticed that the drain profile of the battery had a significant impact on the lifetime of a node. This chapter concludes with a summary of results and a discussion on ideas for future work.

3.2 Background

3.2.1 Speck lifetimes without energy-saving MAC algorithms

The design of a MAC for Specknet is challenging given the energy constraints of each Speck. The ProSpeckz, powered by two CR1210 [46] coin cells with its radio receiver switched on all the time runs for less than 40 seconds. Figure 3-1 shows the discharge graph for the batteries in such a setup over four iterations.

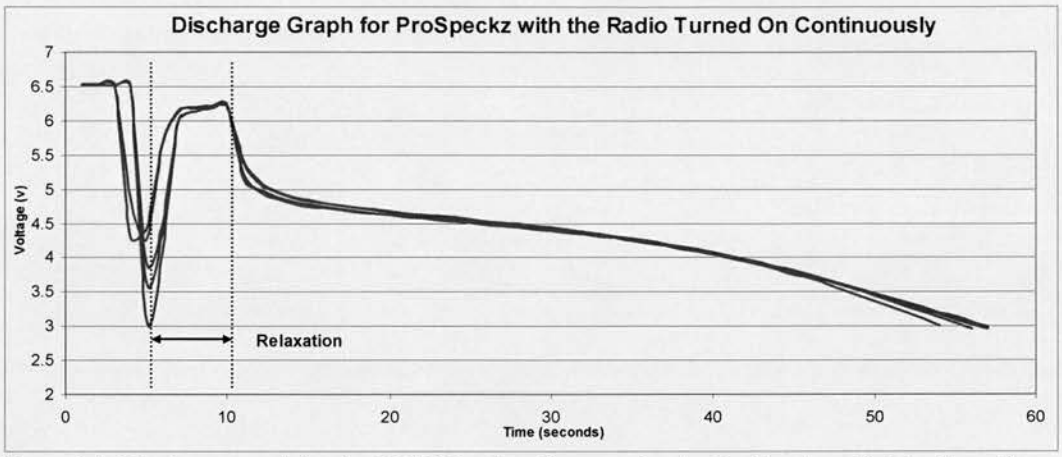


Figure 3-1: Discharge graph for the CR1210 coin cells powering the ProSpeckz which had its radio receiver turned on and the processor in sleep mode

In the first 5 seconds, the battery is drained by the initialisation phase. The battery is relaxed for a period of 5 seconds, during which time, both the radio and the processor allowed to sleep. After the period of relaxation, the radio was turned on while the processor remained in the sleep mode. The results show that the batteries were able to power the ProSpeckz for an average of just 46 seconds before the voltage dropped below the operating voltage of 3 volts. Given that advances in battery technologies do not follow Moore's law and battery capacities are improving at a rate of 2-3% per year [47], highly energy-efficient MAC algorithms would have to be designed for Specks as the energy capacities are unlikely to experience dramatic improvements.

3.2.2 Performance criteria and attributes for Specknet MAC protocols

There are several performance criteria for typical MAC algorithms and some of them are more important in Specknets than others. These criteria and attributes are listed in order of decreasing importance to Specknets;

- **Energy efficiency** is the most important attribute required for any Specknet MAC protocol. On Specks, wireless communication and the processor are the primary sources of power consumption. Given that the MAC protocol directly controls the activities over the communication medium and is the highest layer in the protocol stack that communicates directly with the radio transceiver, MAC protocols with energy-saving features should extend the operating lifetime of the Specks.

- **Robustness** against channel fading is an important consideration as wireless channels are time-varying and prone to interferences, especially if Specks were deployed in noisy environments. Channel fading could cause a link between two Specks to be temporarily broken. Such link failures should not result in unstable behaviour in the MAC algorithms and should allow the Specknet to continue operations.

- **Adaptability** is the ability for a MAC protocol to function under varying densities, topologies, network sizes. Adapting to changes in the network is essential - Specks may dropout of the network over time due to insufficient power or damage; new Specks may join the network; Specks may move from one location to another; an event may trigger a sudden surge in network traffic. These attributes of the network requires a MAC protocol that would gracefully adapt to such changes to the Specknet.

- **Collision avoidance** is one of the primary goals for any MAC protocol. It determines when a node should access the medium and how it should transmit data over it to minimise the chances of interference from other signals transmitted from other nodes. The need for collision avoidance is fairly important as it would increase the success rate for end-to-end packet delivery. This, in turn, would reduce the need for retransmission of data packets which wastes power, increases latency and uses memory.

- **Delay or latency** is the average time spent by a packet from the time it is inserted into the transmission queue of the MAC layer, until the time that the complete message is received and removed from the receiving queue of the destination node. The latency requirement varies based on the application. In applications such as monitoring the behaviour and movement of animals [48], the data can be available in a few hours or even a few days time. However, in a control system, e.g. a search-and-rescue application [49], in which readings from nodes are to be processed and responded to within a hard deadline, the requirement for low latencies becomes very important.

- **Fairness** in networks is exhibited when multiple nodes share a medium without preference for any single node. This attribute may not always apply in Specknets. For example, a Speck that processes critical sensor data may be given higher priority for it to access the medium so that critical information can be distributed with minimal delay.
- **Throughput** is the fraction of the channel capacity that is used for data transmission. This attribute is greatly dependent on other attributes of the MAC protocol such as collision avoidance, delay and control overheads. As radio communications would be expensive in a Specknet, coupled with the assumption of low data traffic requirement, the need for high throughput is not a requirement for Specknets.

As described in the Section 3.2.1, Specks will have extremely short lifetimes if energy-saving MAC is not used; therefore, the consideration in the design of a MAC protocol for Specknet will be energy-conservation. Throughput, on the other hand, is considered to be least important as communication between Specks is assumed to be infrequent.

3.2.3 Existing MAC protocols for supporting multiple access

The MAC protocol enables "Multiple access" of the common wireless medium by the nodes in the network. The following are commonly used, multiple-access techniques: Frequency Division Multiple Access (FDMA) [50], Time Division Multiple Access (TDMA) [51], Code Division Multiple Access (CDMA) [52] and Carrier Sense Multiple Access (CSMA) [53]. These fundamental multiple access techniques be combined to form new hybrid MAC solutions such as Frequency-Division/Code-Divison Multiple Access (FCDMA) [54, 55], TDMA/FDMA in Global System for Mobile Communications (GSM) networks [56], Frequency-Divided Time-Divided Code-Division Multiple Access (F/T/CDMA) [57], Time-Division CDMA (TD-CDMA) [58] , and multi-channel CSMA [59-61].

Figure 3-2 shows four nodes (A, B, C and D) that are within range of each other and share the same radio spectrum. Node A transmits to node B the data frame, (0,1,1,0),

while node C transmits to node D the data frame, (1,0,1,0). This example will be illustrate the sharing of the radio spectrum using the basic multiple-access techniques: FDMA, TDMA, CDMA and CSMA.

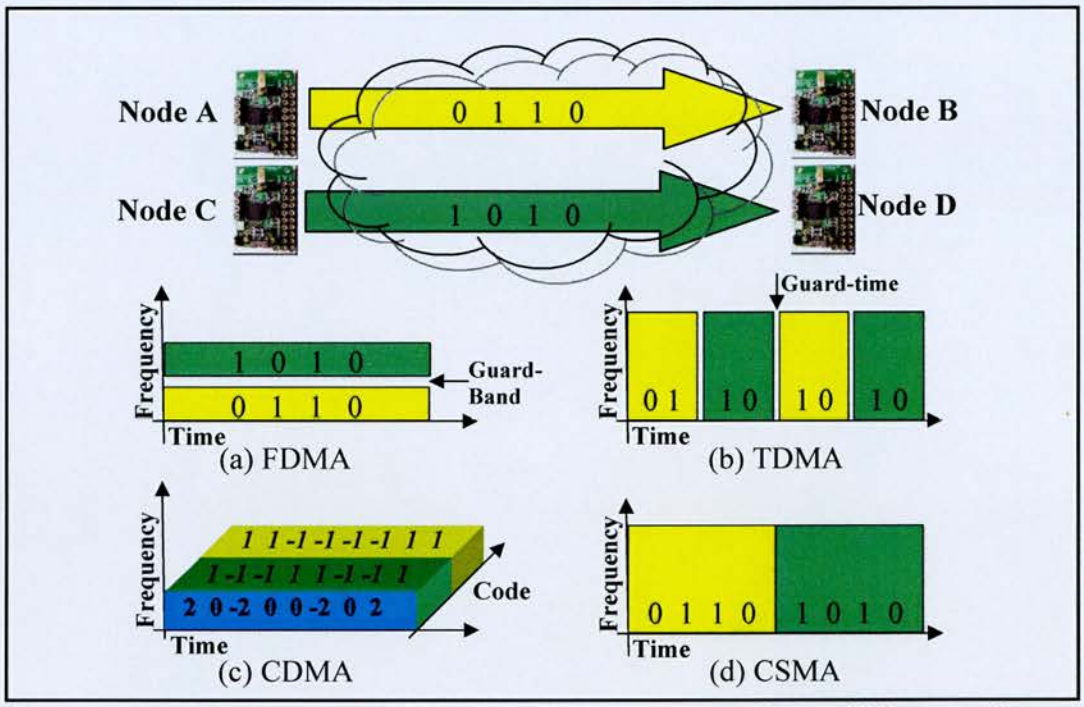


Figure 3-2: An overview on the division of the radio spectrum to allow multiple access for two transmission links based on (a) FDMA, (b) TDMA, (c) CDMA and (d) CSMA.

3.2.3.1 FDMA

FDMA [50] sub-divides the radio spectrum available for communication into channels (Figure 3-2(a)) so that each channel has its own carrier frequency for multi-channel access. To avoid cross-channel interferences and cross-talk, adjacent channels are separated by a guard-band. In the example, the two communication channels were assigned different frequencies, and transmission of the data frames could be carried out concurrently by node A and node C, without interference.

3.2.3.2 TDMA

TDMA [51] divides the temporal space into slots, and nodes are assigned to different slots. To ensure collision-free access to the medium, each node can only transmit in its pre-assigned slots, and each slot is padded with a guard-time to reduce errors due to synchronisation and clock drifts. The larger the guard-time, the less prone is the

communication channel to synchronisation errors. However, this would also decrease the available bandwidth for data transmission, as well as increase the latency. In the example, the spectrum was divided into two time slots and each slot was large enough to transmit exactly two data bits (Figure 3-2(b)). Node A was assigned the first slot and the next slot was assigned to node C. Each node transmits two bits in its pre-assigned slot and transmission through the medium is multiplexed between the two nodes.

3.2.3.3 CDMA

CDMA [52] is a spread-spectrum technique that uses coding of the data stream to enable multiple access of the medium. The code is a vector of digital values, either orthogonal or non-orthogonal. Orthogonal codes are a set of vectors, such that the dot product of any two vectors in the set would result in a zero, e.g. (1,-1) and (-1,-1) are orthogonal vectors as (1,-1).(-1,-1) = (1)(-1) + (-1)(-1). Orthogonal codes are used in synchronous CDMA (SCDMA) systems as the codes are orthogonal only if they were synchronized in time, therefore SCDMA is often used in base-to-mobile node communications in cellular network. To explain the operation of a SCDMA system, using the example in Figure 3-2, the channel between nodes A and B is assigned the code vector, v_1 = (-1,-1), and the channel between nodes C and D is assigned the code vector, v_2 = (1,-1). A binary 'one' in the data frame is coded as v_1 and v_2, while a binary 'zero' is coded as $-v_1$ and $-v_2$, by the transmitters of the node A and node C, respectively. In Figure 3-2(c), node A transmits (1,1,-1,-1,-1,-1,1,1) and node C transmits (1,-1,-1,1,1,-1,-1,1). The resultant signal received by both nodes B and D will be the summation of the two transmitted signals, producing (2,0,-2,0,0,-2,0,2). The data intended for the destination nodes is recovered by the receiver using the dot product between the code vectors and the received signal in parts. If the result of the dot product is positive, the decoded data bit is a binary 'one', otherwise, a binary 'zero' is decoded. For example, the dot product of the received signal at node D will result in ((2,0).(1,-1),(-2,0).(1,-1),(0,-2).(1,-1),(0,2).(1,-1)) = (2,-2,2,-2) , which will be decoded as (1, 0, 1, 0). In an asynchronous system, the use of orthogonal codes will not be possible due to the arbitrary start points of the data streams, therefore

Pseudo-Noise (PN) sequences/codes are used to code the transmission instead. Asynchronous CDMA works on the assumption that the signal strength received from the different transmitters are equal at the receiver. When the PN sequence used by a transmitter is applied to the signal received at the destination node, the amplitude of the signal from that transmitter increases; enabling that signal to be isolated at the receiver using a filter and the transmitted data is recovered. The requirement that the signal strength from the different transmitters be equal at the receiver poses a problem, as the strength of the transmitted signal follows Newton's inverse-square law and the radio strength decreases over distance. Therefore, if all nodes were to transmit at the same power, the nodes closer to the receiver would generally have a stronger signal at the receiver. This problem is known as the near-far problem [62-64] which to-date, given other factors such as channel fading, is still being extensively researched for CDMA-based multi-user/multi-channel radio systems. Furthermore, as the method of dividing the spectrum is non-orthogonal, unlike FDMA, TDMA and SCDMA approaches, asynchronous CDMA suffers from Multiple-Access Interference (MAI) which increases with the number of users and reducing the signal-to-noise ratio of the signals as the number of simultaneous transmission increase.

3.2.3.4 CSMA

CSMA [53] is a asynchronous multiple-access technique that uses a contention-based approach for sharing a single transmission channel. Nodes check the channel for transmissions from other nodes for a period of time prior to transmitting its frame, a process known as "listening". Transmission will only occur if the channel was detected to be free. Using the example in Figure 3-2, assuming that Node A attempts its transmission before Node C, Node A will detect that the channel is free and transmit its data bits. Node C will then detect that the channel is busy and defer its transmission until the channel becomes free again. The activity on the channel is shown in Figure 3-2(d).

There are three variations of CSMA based on the different levels of 'persistence' when a node attempts to transmit a frame. In non-persistent CSMA, if a channel was

sensed to be busy, the node would stop listening to the radio channel and attempt to transmit the packet again after a random back-off period. In 1-persistent CSMA systems, when a busy channel was sensed, the node would continue listening to the channel and would transmit the packet as soon as the channel is free. In *p*-persistence CSMA, if a channel was sensed to be busy, the node will continue listening to the channel and transmit with a probability of *p* once the channel was free. If the node decides not to transmit the frame in the current attempt, it will attempt to transmit the frame again after a random back-off period.

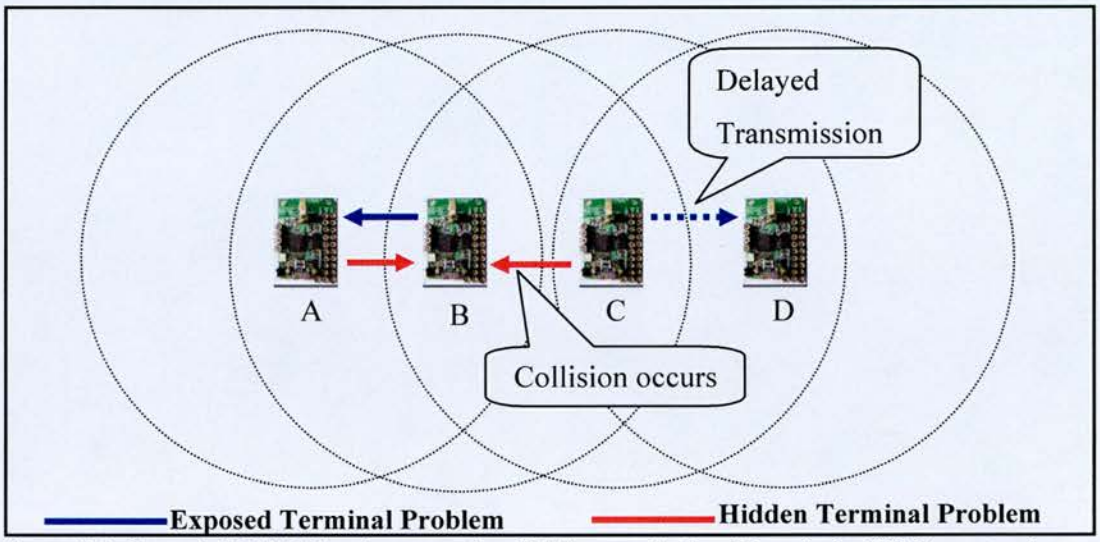


Figure 3-3: Scenario highlighting the exposed and hidden terminal problems in the CSMA protocol

CSMA suffers from two fundamental problems when used in a multi-hop network: the Exposed Terminal problem, and the Hidden Terminal problem [65]. Exposed Terminal problem occurs when a node is prevented from transmitting its frame when in actual fact, it could. For example, Figure 3-3 shows a typical wireless multi-hop network with four nodes where Node C has data to send to node D, while node B was transmitting to Node A. Node C would have sensed the transmission from node B and therefore delay its transmission to node D. This delay is unnecessary as node C could have transmitted its frame without causing any interference to the signal received by node A. The Hidden Terminal problem occurs when two nodes that are not within range of each other send frames concurrently to a common node. With reference to Figure 3-3, node A is transmitting to node B when node C has data to

transmit to node B. As node C is not within the range of node A, it will not be able to detect the signal transmitted by node A. Therefore, Node C will assume that the channel is free and will transmit its frame which causes a collision at node B.

There are a number of extensions to CSMA to provide for better delivery ratio and bandwidth utilisation. CSMA, in its native form, makes use of just carrier sensing to dictate if a node is allowed to send a data frame. However, collision would still occur when two nodes attempt to transmit at the same time on an empty channel as both would sense that the channel was free. Collision would then occur with energy used in transmitting both frames wasted as the receiver may not distinguish either signals. To minimise the energy wastage, CSMA with collision detection CSMA/CD [66] looks out for collisions during transmission. Should it be detected, the transmissions will stop immediately and all the nodes involved in the collision will attempt to retransmit after a random back-off duration. However, CSMA/CD is used in wired systems rather than wireless ones due to the ease in detecting collisions in wired systems such as the Ethernet. CSMA with collision avoidance (CSMA/CA) attempts to minimise the possibility of the collision during data transmission. One approach is the use of short signalling frames such as Request-to-send (RTS), Clear-to-send (CTS) and Acknowledgment (ACK) for performing handshaking to avoid the hidden-terminal problem [13]. When a source node has data to send, it transmits a RTS frame, containing the address of its destination node and the duration for which it requires the channel. The destination node next transmits a CTS frame, which includes the duration information received from the RTS frame, if it is free to receive the transmission from the source node. Other nodes that receive either the RTS or CTS frames will internally mark the channel as being 'reserved' (for example, in 802.11, in the Network Allocation Vector (NAV)) for the duration specified in the received frame. The source node, on receiving the CTS frame, will transmit its data frame, and the destination node, on receiving the data frame will response with an acknowledgement frame. To avoid the hidden terminal problem, a node will defer any transmission to a 'reserved' channel. The channel is 'unreserved' when the NAV on the channel has expired or an acknowledge frame (ACK) is received. This technique of deferring transmission is known as "Virtual Carrier Sensing".

3.2.3.5 Comparison of multiple access protocols for use in Specknets

A common problem faced in the deployment of FDMA and CDMA techniques in wireless mobile ad-hoc networks, such as Specknets, is the complexity involved in the assignment and coordination of the different frequencies and codes used by the nodes in a dynamic system without recourse to a central controller. Codes and frequencies may also have to be reused and reassigned as there are limited channels when one considers a large network consisting of hundreds of nodes. To assign in a distributed fashion a limited set of channels without conflict between neighbouring nodes is an NP complete [67] and distributed dynamic channel allocation schemes [68-70] require additional overheads for nodes to maintain and coordinate channel assignment information over the network. similarly, time slots in TDMA-based systems need to be assigned using either a base-station or some dynamic slot allocation scheme [71, 72]. Synchronous MAC protocols such as TDMA and SCDMA also require accurate synchronization among the nodes in the network in order to function correctly. This can be achieved via coordination provided by a base station or the use of a distributed synchronization algorithm between Specks. However, in both cases, the need for a base-station and the overheads involved in maintaining time synchronization limits the usability of synchronous MAC protocols in Specknets.

To avoid the complexity and additional overheads involved in frequency/time/code assignment and coordination, a single-channel contention-based MAC algorithm, such as CSMA, would be a suitable choice for Specks for both its simplicity and low computational requirements. Although CSMA does not use the available spectrum as fully as multi-channel approaches such as CDMA, it should be adequate for Specknets, given the assumption that high throughput was the least important criteria in the MAC requirements. Furthermore, in this thesis, instead of using signalling frames, an initial random back-off prior to data transmission is used instead as a collision avoidance mechanism as to reduce the communication overhead due to the transmissions of RTS and CTS frames, given that Specks were assumed to both communicate infrequently, and have small frame sizes.

3.2.4 MAC protocols for supporting energy conservation

A number of MAC algorithms have been proposed for energy-limited wireless devices by focussing on conserving energy by reducing the duration of 'idle listening' times. Idle listening occurs when the radio receiver in a node is left turned on and there is no data to be received or transmitted. For nodes that transmit over short ranges such Specks, the energy consumption for powering the transmitter would generally be less than the energy required for powering the receiver. For example, the CC2420 onboard the ProSpeckz requires 18.8mA to power its receiver while powering the transmitter at -25dBm requires just 8.5mA. Therefore, reducing the duration of idle listening will significantly extend the lifetimes of the nodes.

Energy-saving MAC protocols can be either centralized or distributed. In centralised MAC protocols, such as TDMA, a base station provides the coordination required to allow collision-free operation within the network. All the nodes will have to be within the range of the base station in order for centralised protocols to operate and are therefore typically deployed in single-hop networks. For example, in Bluetooth [73], the temporal space is divided into small slots in which nodes are only awake for the times when they are assigned a slot. In the case of Specknets where *all* the nodes in the network have limited energy supply, centralised MAC protocols would not be appropriate. Furthermore, the assumption of the absence of controllers in a Specknet would also prevent the use of centralised algorithms. For all these reasons, centralised MAC algorithms were not considered in this thesis.

Distributed MAC protocols are designed to operate in multi-hop networks. These MAC protocols provide both multiple-access and energy saving through the coordination performed between nodes, rather than through a centralised controller. Decentralised MAC protocols can either be synchronous or asynchronous. Synchronous MAC protocols require all the clocks in the nodes across the network to be synchronised or timing information of the nodes to be shared. Asynchronous MAC protocols, on the other hand, do not require nodes in the network to perform any form of timer or clock updates in order to communicate wirelessly. MAC protocols can also be divided into two classes: scheduled or random-access. These

two classes differ in the way the nodes coordinate their access to the radio channel. Scheduled MAC protocols employ time slots in a fashion similar to TDMA. However, these slots are larger than those employed in centralised schemes, and therefore obviate the need for tight time synchronisation. To avoid the need for a centralised controller, each node has its own schedule of time slots during which to turn its radio receiver on. Nodes are required to transmit their schedule information periodically to enable other ad-hoc nodes to follow their schedule (in the case of a single schedule system), or for other nodes to be aware of their schedules (in the case of a multi-schedule system). In either case, schedule updates would have to occur regularly to support mobility and changes in the ad-hoc network. This requirement could be a significant overhead as mobility increases. Random access MAC protocols, on the other hand, do not require nodes to keep or follow schedules. Instead, nodes compete for access to the channel and the node that gains control of the channel is able to transmit packets.

Table 3-1 lists the distributed energy-efficient MAC algorithms considered in this section. Their approaches are compared, as well as their appropriateness as MAC protocols for Specknets are discussed.

Section	Protocol	Synchronous	Asynchronous	Channel Access
3.2.4.1	S-MAC	Single- Schedule	Multi-Schedule	Schedule-based
3.2.4.2	T-MAC		✓	Schedule-based
3.2.4.3	802.11 DCF/PS	✓		Schedule-based
3.2.4.4	Modified 802.11 DCF/PS		✓	Schedule-based
3.2.4.5	Multi-radio systems		✓	Random-Access
3.2.4.6	B-MAC		✓	Random-Access

Table 3-1: Distributed MAC protocols used in typical wireless networks

3.2.4.1 S-MAC

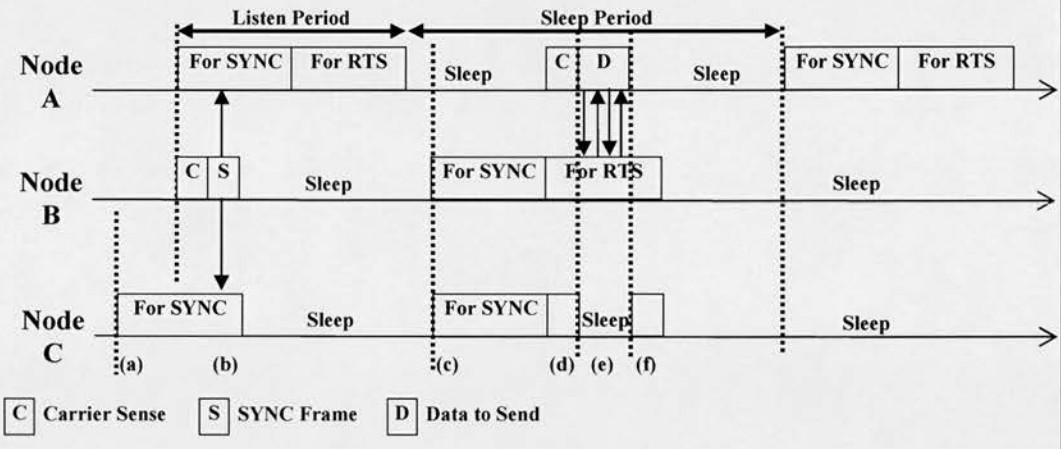
S-MAC [74] trades power off with latency by alternating the radio between sleep and listen modes periodically, as shown in Figure 3-4. Each node also transmits periodically a SYNC frame, containing its address and the time when it sleeps next. This enables those neighbouring nodes which received the SYNC packet to keep track of the sender's schedule, i.e. the times during which the sender of the SYNC packet will be listening. The listen period is divided into 2 parts: one for receiving SYNC frames, and the other for receiving RTS frames from intended transmitters. S-

MAC uses CSMA/CA with message passing (one RTS, one CTS, multiple DATA/ACK) for collision avoidance, together with virtual carrier sense. Avoidance of overhearing avoidance is achieved by turning the radio off when the RTS/CTS frames for other destination nodes are heard.

Each nodes has a schedule to turn its radio on and off and periodically listens for a full cycle (sleep and listen period) to receive SYNC frames from nodes with other schedules. A node determines its schedule during the start-up process by listening to the channel for a period of time. If it hears a schedule (via a SYNC frame sent by some other node) during this time, it will follow the schedule of that node with the initialised node being a "follower". If no SYNC frame was heard, the node assumes the role of "synchroniser" and will send its SYNC packet after a random delay. If two nodes happen to start at the same time and no SYNC frames were received, then both nodes would assume the role of the "synchroniser". In such a case, when a node hears another schedule, it will adopt both schedules, i.e. node will be following two schedules. A node can also just follow one schedule but still keep track of the other node's schedule. In this way, it would still be able to communicate with that node. However, in a multi-schedule scenario, nodes would have to retransmit the data frame several times for broadcast traffic such that at least one transmission of the data frame would occur in the active periods across all the schedules being followed.

S-MAC allowed energy consumption to be reduced by trading off latency and throughput [74]. There are, however, some implications for using the algorithm for Specknets. Firstly, S-MAC operates on the basis of a fixed listen period, which would not adapt well to situations with dynamic traffic loads. Secondly, network changes are not reflected rapidly which might be critical in mobile ad-hoc networks such as Specknets. In Figure 3-4, Node C is not aware of Node A's schedule and will not be able to communicate with Node A, until it receives a SYNC frame from Node A. For example, if SYNC packets are sent by Node A once every 10 seconds and Node C performs schedule updates by listening for a full cycle (assuming a total sleep and listen period of 1 second) of once every 9 seconds, it will take up to a maximum delay of 90 seconds before Node B is aware of Node A's schedule. To reduce this delay, either more SYNC packets would have to be sent, or nodes would

have to perform periodic full cycle schedule listening more frequently. Both these solutions would increase the power consumption of S-MAC. Lastly, the requirements of keeping multiple schedules and the need to perform retransmission for broadcast traffic in multi-schedule scenarios would further reduce the energy-saving capabilities of S-MAC.



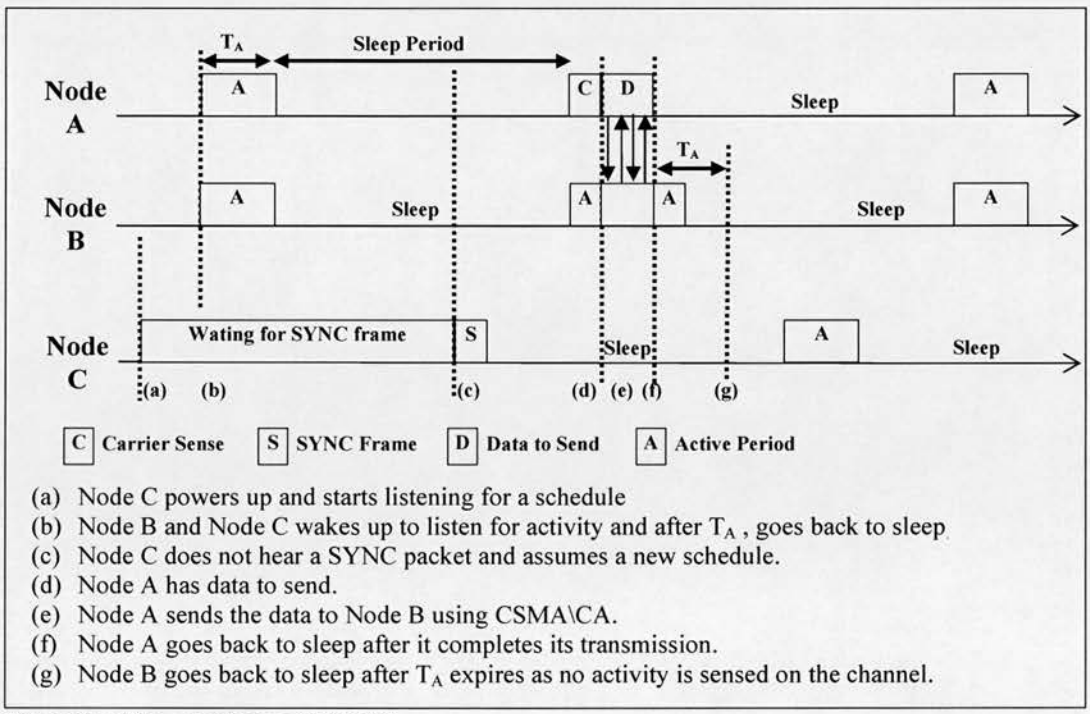
(a) Node C powers up and starts listening for a schedule
(b) Node B sends a SYNC frame. Node A receives it and keeps track of Node B's schedule. Node C receives the SYNC frame and follows the schedule of Node B.
(c) Node A has data to send to Node B.
(d) Node A sends RTS frame to Node B. Node C receives the RTS frame and goes to sleep.
(e) Communication between Node A and Node B continues.
(f) Node C returns to listening to the channel after NAV expires.

Figure 3-4: The operation of S-MAC

3.2.4.2 T-MAC

T-MAC [75] extends the S-MAC protocol by introducing a dynamic active period. The operation, as shown in Figure 3-5, is similar to S-MAC except that instead of a fixed active period, each node periodically wakes up to communicate with its neighbours and sleeps when its active period expires. An active period expires when no activation event has occurred for a time, T_A, and an activation event could be any one of the following: firing of a periodic frame timer, the reception of any data on the radio, the sensing of a busy channel, the end-of-transmission of a node's own data packet or the reception of an ACK frame indicating that the data exchange between two neighbours has ended.

Similar to S-MAC, T-MAC uses virtual clustering for synchronisation with the exception that nodes adopt all the different schedules learnt and only send data during the start of their own active time. Therefore, broadcast packets are only transmitted once. To maintenance of schedules between nodes is achieved using SYNC frames which are retransmitted periodically and nodes will listen for a full sleep/active cycle sporadically.



(a) Node C powers up and starts listening for a schedule
(b) Node B and Node C wakes up to listen for activity and after T_A , goes back to sleep
(c) Node C does not hear a SYNC packet and assumes a new schedule.
(d) Node A has data to send.
(e) Node A sends the data to Node B using CSMA\CA.
(f) Node A goes back to sleep after it completes its transmission.
(g) Node B goes back to sleep after T_A expires as no activity is sensed on the channel.

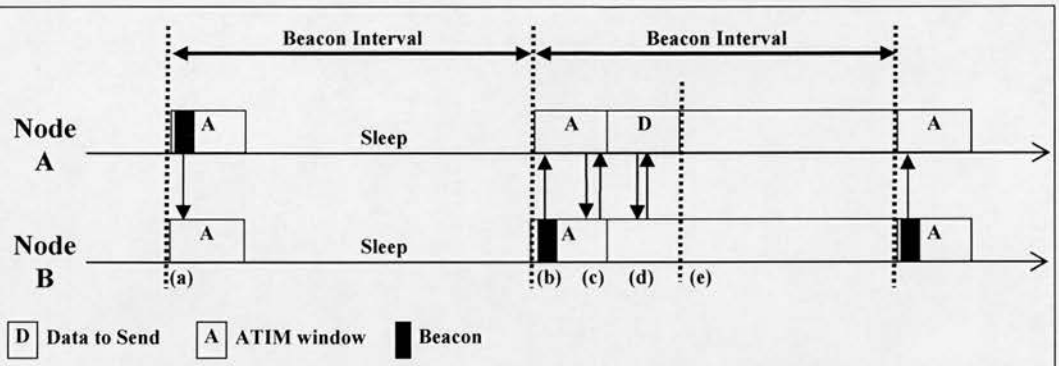
Figure 3-5: The operations of T-MAC

Results [75] have demonstrated that when nodes send data periodically with only one schedule existing in the network, the performance of both S-MAC and T-MAC were comparable in their energy-conservations; however, under variable load, T-MAC uses less energy than S-MAC due to the reduction in idle listening time during the listen/active period. However, these results only consider a single schedule system. In a multi-schedule system, similar to S-MAC, network changes will not be reflected rapidly enough which could be critical in mobile networks such as Specknets. For example, with reference to Figure 3-5, node C will be unaware of the schedule of both nodes A and B until it hears a SYNC frame from either of them when it performs the periodic schedule update. To increase the rate at which nodes detect

schedules, nodes could either send greater number of SYNC frames or perform schedule updates more frequently. Both these approaches will adversely impact energy consumption. Furthermore, unlike S-MAC in which nodes could choose not to adopt schedules, nodes in T-MAC adopt all the schedules, which results in greater energy wastage due to idle listening as the number of schedules in the network increases.

3.2.4.3 802.11 PS/DCF

802.11 PS/DCF [13] assumes that all clocks are synchronised and the network is fully-connected. In its ad-hoc mode, all nodes periodically, based on a beacon interval, wake up for a duration defined by the Ad-hoc Traffic Indication Map (ATIM) window, as shown in Figure 3-6. During the ATIM window, nodes will compete to send a beacon frame using CSMA. On receiving a beacon frame, nodes would stop trying to send a beacon, thus only one node will succeed in sending its beacon for every ATIM window. Nodes with data to send will then contend to send an ATIM frame. The node that had succeeded in transmitting the ATIM frame could then send its buffered packets during the period following the ATIM window. Nodes that were unsuccessful in sending the ATIM frame or with more packets to send will contend for the channel during the next ATIM window. A node which had received an ATIM frame will keep its radio on for the whole duration defined by the beacon interval. As 802.11PS/DCF will only function for fully-connected networks, it will not be suitable for a multi-hop network such as Specknets.



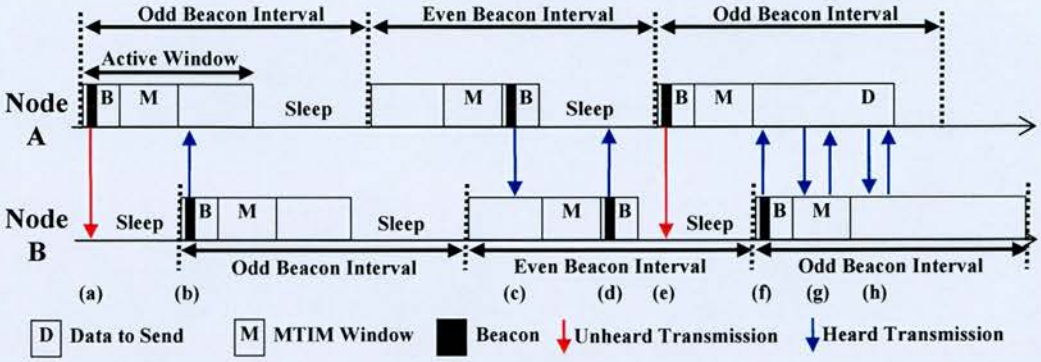
(a) Node A transmits its beacon. Node B receives the beacon and synchronises its clock.
(b) Node B transmits its beacon. Node A receives the beacon and synchronises its clock.
(c) Node A has data to send to Node B and sends an ATIM frame to Node B. Node B replies with an ACK frame.
(d) Node A transmits the data frame to Node B and Node B sends a ACK back to Node A.
(e) Both Node A and B would keep their radios on for the rest of the beacon interval

Figure 3-6: The operation of 802.11PS/DCF

3.2.4.4 Modified 802.11 PS/DCF

The 802.11PS/DCF algorithm has been adapted for multi-hop network without recourse to clock synchronisation by using three different wakeup protocols [76]. The beacon interval is divided into three parts: the beacon window, the Multi-hop Traffic Indication Map (MTIM) and the active window. The nodes use the beacon window to transmit their beacons. Unlike the original 802.11 PS/DCF, each node does not stop attempting to send its beacon even if it hears a beacon from another node. The function of the MTIM window is similar to that of the ATIM window in 802.11 PS/DCF, in which nodes with data to send to a destination node will compete to transmit a MTIM frame during the MTIM window of the destination node. The node which manages to transmit its MTIM frame will then be able to transmit its data packets during the active window. All those nodes that received the MTIM frame will keep their radio receivers on for the entire period of the active window to receive the data packets. Three wakeup protocols have been proposed for 802.11 PS/DCF. The Dominating-Awake-Interval protocol, as shown in Figure 3-7, uses different arrangement for the odd and even active periods to ensure that nodes hear a beacon from its neighbours every two beacon interval, as long as that the following condition is enforced:

$$Active\ Window \geq (Beacon\ Interval) + Beacon\ Window$$

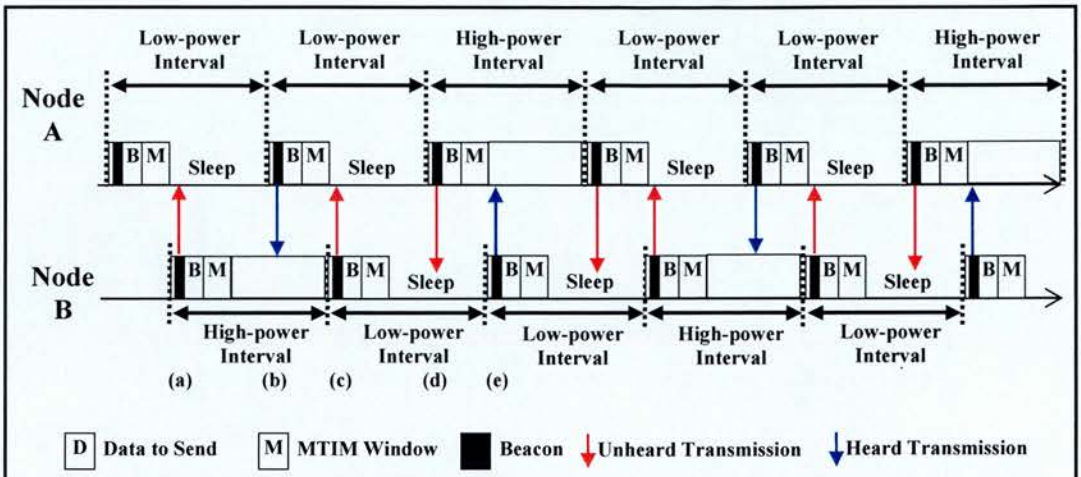


(a) Node A transmits its beacon. Node B cannot receive the beacon from node A as its radio is not turned on. Thus, Node B is still unaware of the presence of Node A.
(b) Node B transmits its beacon. Node A receives the beacon and records the timing information for Node B. Node A is now able to transmit to Node B.
(c) Node A transmits its beacon. Node B receives the beacon and records the timing information for Node A. Node B is now able to transmit to Node A
(d) Node B transmits its beacon. Node A could not receive the beacon as it is in the sleep mode, thus, clock synchronisation is not performed.
(e) Node A transmits its beacon. Node B could not receive the beacon as it is in the sleep mode, thus, clock synchronisation is not performed.
(f) Node B transmits its beacon. Node A receives the beacon and updates the timing information for Node B.
(g) Node A has data to send to Node B. Using the timing information obtained from the last received beacon from Node B, Node A contends and sends a MTIM frame to Node B during the active window of Node B.
(h) Data transmission will then commence.

Figure 3-7: The operation of 802.11PS/DCF using the Dominating-Awake-Interval protocol

The requirement for a large active window requires this wakeup protocol to keep the radio on for more than 50% of the time, which does not provide much energy saving. An alternative approach, the Periodically-Fully-Awake-Interval protocol, as shown in Figure 3-8, improves energy savings by dividing the beacon intervals into low-power ones and high-power ones. During low-power intervals, the radio is turned on just for the beacon window and the MTIM window. During high-power intervals, the radio is left on for the entire duration. This protocol reduces the time that the receiver needs to be turned on; however, it suffers from two drawbacks. Firstly, the radio is on for the entire high-power interval even if there are no data packets to receive. Secondly, unlike the Dominating-Awake-Interval protocol in which nodes discover each other in two beacon intervals, the latency for doing so increases as the ratio between low-power and high-power intervals increase. This restricts the minimum duty-cycle in practice and bounds the amount of mobility that can be supported by

the protocol. The Quorum-based protocol combines the advantages of the Dominating-Awake-Interval and the Periodically-Fully-Awake-Interval protocol. In the protocol, n^2 beacon intervals is arranged into an n*n array. Each node selects a row and a column in the array as its quorum interval and the rest as non-quorum intervals. Nodes turn their radio on during the quorum intervals (2n-1 interval); and during the non-quorum intervals (n^2-2n+1 intervals), the radio is only turned on for the MTIM window and is in sleep mode at other times. This protocol mitigates the disadvantages of the two protocols presented previously however, the nodes still transmit and receive beacons even though there is no data to be sent or received. Maintaining this time coordination which is unnecessary wastes considerable energy. Furthermore, for broadcast traffic, multiple retransmissions would be necessary. From the results presented [76], given a small network neighbour size of only 4 nodes and a light traffic load, this approach still consumes 25% of the power used by a node that was always active. Furthermore, the energy savings are made possible by trading off a large increase in the latency to detect a new neighbour. Such a delay would be unsuitable for Specknets in the degree of mobility that can be support. For these reasons, the modified 802.11 DCF/PS was deemed unsuitable as a MAC algorithm for Specknets.



(a) Node B transmits its beacon. Node A cannot receive the beacon from node B as its radio is not turned on. Thus, Node A is unaware of the presence of Node B.
(b) Node A transmits its beacon. Node B receives the beacon and records the timing information for Node A. Node B is now able to transmit to Node A.
(c) Node B transmits its beacon however; Node A still could not receive the beacon as its radio is not turned on. Thus, Node A is still unaware of the presence of Node B.
(d) Node A transmits its beacon. Node B could not receive the beacon as it was in the sleep mode, thus, clock synchronisation is not be performed.
(e) Node B transmits its beacon. Node A receives the beacon and records the timing information for Node B. Node A is now able to send data packets to Node B.

Figure 3-8: The operation of 802.11PS/DCF using the Periodically-Fully-Awake-Interval protocol

3.2.4.5 Multiple-radio based MAC

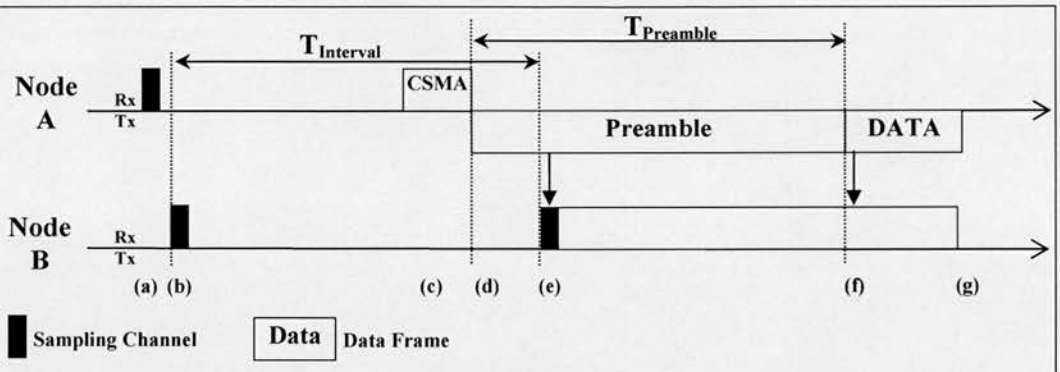
To enable power-saving, Systems [77-80] using two physical radios, one high-powered and the other low-powered, have been proposed. The low-power radio consumes significantly less power but has poorer signal-to-noise properties and lower bandwidths than its high-power counterpart, making it unsuitable for communicating data packets. Instead, the low-power radio is used for communicating wakeup signals. Unlike the low-power radio which is consistently powered all the time, the high-power radio is only turned on when a wakeup signal is received and data communication can commence. The use of two separate radios helps conserves energy at the cost of additional hardware. This space overhead of using two radios precludes this option for the miniature Specks.

3.2.4.6 B-MAC – a Random-Access MAC algorithm for Asynchronous Distributed Wireless Networks

B-MAC [5] is a distributed random-access algorithm for asynchronous wireless sensor networks which conserves energy by using in-channel signalling to wakeup destination nodes. B-MAC is a random access MAC algorithm which does not require nodes to be synchronised to each another and would therefore be appropriate for networks with highly mobile nodes which do not require update of neighbours' schedules. In addition, B-MAC transfers the communication costs to the transmission of data, making it more expensive for a node to transmit a data packet as compared to receiving one. This is appropriate on the premise that Specknets require low data rates and have higher nodal densities (as neighbours per node), and each node will likely transmit less and receive more packets.

B-MAC operates by periodically listening to the channel for activity. The frequency of sampling is determined by a user-selected parameter, $T_{Interval}$. As shown in Figure 3-9, nodes periodically sample the channel periodically and will keep their radio receivers on if the channel is sensed to be busy, and will turn them off after a data packet is received or after a timeout. For data transmission, the source node will send a long preamble lasting the duration of $T_{Preamble}$, before sending the data frame. This preamble is used to wakeup destination nodes during the channel listening phase, for correct operation of the protocol, the following relationship must be enforced:

$$T_{Preamble} > T_{Interval}$$



(a) Node A turns the radio receiver on to sample the channel and turns it off after the sampling duration as no activity was sensed on the channel.
(b) Node B turns on the radio receiver to sample the channel and turns it off after the sampling duration as no activity was sensed on the channel.
(c) Node A has data to send and performs carrier sense (CSMA) to contend for the channel
(d) As no other activity was sensed on the channel, Node A will start transmitting the preamble for a duration determined by $T_{Preamble}$.
(e) Node B turns the radio receiver on to sample the channel and detects activity on the channel. It keeps the radio receiver on.
(f) Node A sends the data frame and the frame is received by Node B as its radio was kept on.
(g) Node A and Node B turns the radio off after transmitting and receiving the data payload respectively. Periodic channel sampling is resumed.

Figure 3-9: The operation of B-MAC

3.2.4.7 Comparison of the suitability of energy-conserving MAC protocols for Specknets

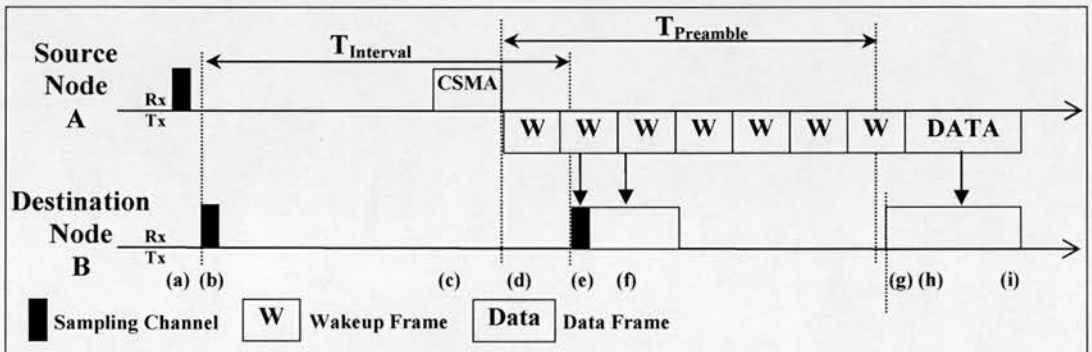
Centralised MAC techniques were not considered as potential MAC for Specknets given the requirement that Specknets must be able to operate without any base-stations or central controller. Distributed synchronous MAC algorithms, such as the single-schedule S-MAC and 802.11 DCF/PS, suffer from the requirement to synchronise the clocks of the nodes in order to communicate effectively. This will result in unnecessary overheads in applications with sporadic data requirements, as clock synchronisation will still be carried out even if there were no data to transmit or receive. Schedule-based asynchronous MAC algorithms, such as the multi-schedule S-MAC, T-MAC and the modified 802.11DCF/PS, rely on the ability of nodes to keep track of the schedule of their neighbouring nodes, i.e., information regarding the times in which neighbouring nodes turn their radio receivers on. This approach suffers from the same problem as synchronous MAC techniques in that

nodes are required to keep track of their neighbours' schedules even when there was no data to be transmitted or received. This overhead in an asynchronous system could be reduced by extending the period in which nodes update their schedules. However, this leads to longer latency before a node can receive and update all its neighbours' schedule information, and limits the mobility supported by the network. Asynchronous random-access MAC protocols, such as B-MAC and multi-radio based systems, eliminate the need for any timing information to be maintained. This reduces the overheads in applications with properties of sporadic data transfers. Therefore, a MAC protocol that is distributed, asynchronous and random-access based would be suitable for Specknets. Multi-radio based approaches were discounted due to the need to accommodate two radio architectures which was unsuitable for the miniature Specks. On the other hand, MAC protocols that used in-channel signalling, such as B-MAC, were more appropriate as no additional radio hardware was required, and no communication overheads was incurred for synchronising clocks or maintaining schedule information. Furthermore, it had been shown that B-MAC outperforms S-MAC [5] in scenarios involving time-varying traffic loads. In summary, of all the MAC protocols considered so far, B-MAC was best suited for a mobile ad-hoc network with low data rates such as Specknets. B-MAC was therefore chosen as the basis for comparison with the family of SpeckMAC protocols proposed in the next section.

3.3 SpeckMAC

SpeckMAC is a distributed random-access asynchronous MAC algorithm developed for Specknets which uses in-channel signalling to control the radio. However, instead of sending long preambles for each data frame as in the case of B-MAC, SpeckMAC uses redundant retransmissions to fulfil the same role. There are two variations of SpeckMAC: SpeckMAC-B (SpeckMAC-Backoff) sends wakeup frames instead of the long preambles, and SpeckMAC-D (SpeckMAC-Data) sends the actual data frame repetitively.

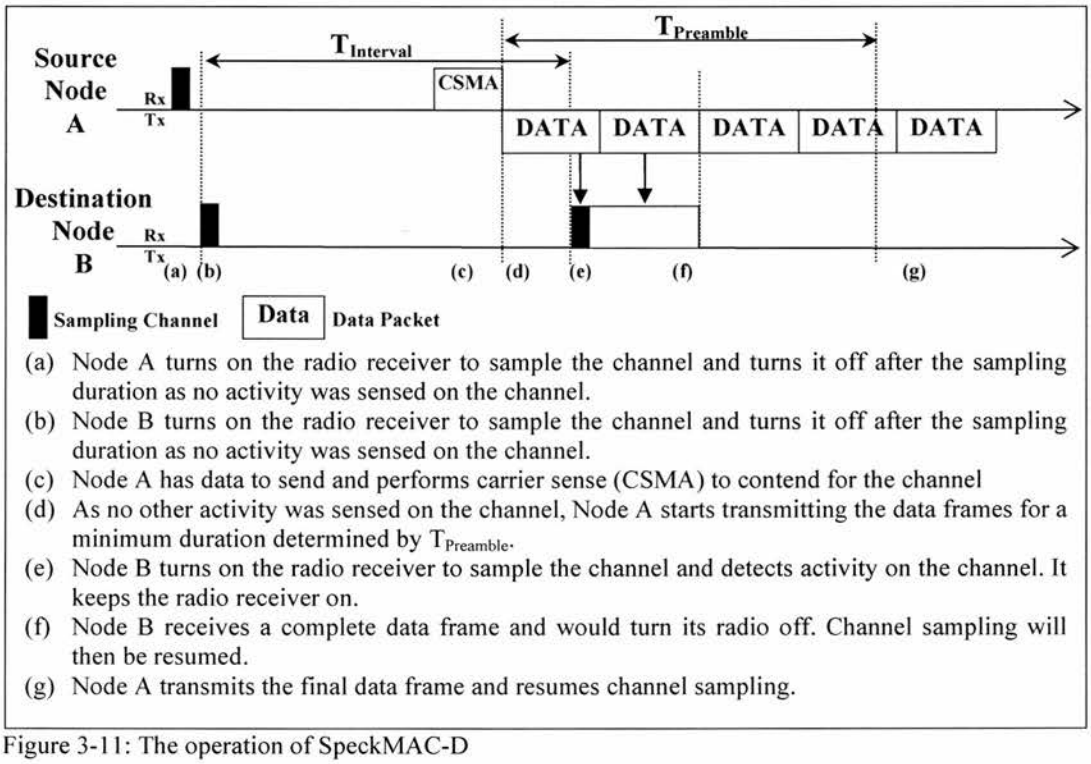
In SpeckMAC-B (Figure 3-10), the source node fills the preamble space with a number of small wakeup frames. Each wakeup frame contains the destination address of the data frame and the timing information as to when the source node expects to transmit the data frames. Similar to the operation performed by B-MAC, every node listens periodically for activity on the channel, as determined by $T_{Interval}$. If activity is sensed, then the node turns on its receiver and waits for the receipt of a frame, either a data or the wakeup one. In the event that a data frame, the radio reverts to the idle state and resumes periodic listening. Should a wakeup frame be received, the node reverts to the idle state and backs off for a duration specified by the received timing information, provided that the destination address specified in the wakeup frame matched its own address or a broadcast address. The node will then turn on its receiver again after the back off duration in anticipation of the data frame. After reception of the data frame, the node resumes periodic channel listening.



(a) Node A turns on the radio receiver to sample the channel and turns it off after the sampling duration as no activity was sensed on the channel.
(b) Node B turns on the radio receiver to sample the channel and turns it off after the sampling duration as no activity was sensed on the channel.
(c) Node A has data to send and performs carrier sense (CSMA) to contend for the channel
(d) As no other activity was sense on the channel, Node A would start transmitting the wakeup frames for a minimum duration determined by $T_{Preamble}$.
(e) Node B turns on the radio receiver to sample the channel and detects activity on the channel. It keeps the radio receiver on in anticipation of receiving a data or wakeup frame.
(f) Node B receives a complete wakeup frame and would turn it radio off for a duration based on the timing information extracted from the received wakeup frame
(g) Based on the extracted timing information, Node B turns its radio on in anticipation of receiving the data frame.
(h) Node A transmits the data frame.
(i) Node A and Node B turns the radio off after transmitting and receiving the data frame respectively. Periodic channel sampling is resumed.

Figure 3-10: The operation of SpeckMAC-B

SpeckMAC-D (Figure 3-11) operates in a manner similar to SpeckMAC-B. However, instead of filling the preamble space with wakeup frames, SpeckMAC-D transmits repetitions of the data frame. In addition, unlike B-MAC where the preamble is long (tens or hundreds of bytes), each data frame in SpeckMAC-D is padded with a very short preamble (typically less than ten bytes). Nodes listen to the channel periodically at intervals specified by $T_{Interval}$ to sample the channel for activity. Upon detecting channel activity, the node keeps its receiver turned on to enable the reception of the data packet. After the data frame is received successfully, the node places its radio in the idle state for the duration of $T_{Interval}$, before resuming periodic listening.



(a) Node A turns on the radio receiver to sample the channel and turns it off after the sampling duration as no activity was sensed on the channel.
(b) Node B turns on the radio receiver to sample the channel and turns it off after the sampling duration as no activity was sensed on the channel.
(c) Node A has data to send and performs carrier sense (CSMA) to contend for the channel
(d) As no other activity was sensed on the channel, Node A starts transmitting the data frames for a minimum duration determined by $T_{Preamble}$.
(e) Node B turns on the radio receiver to sample the channel and detects activity on the channel. It keeps the radio receiver on.
(f) Node B receives a complete data frame and would turn its radio off. Channel sampling will then be resumed.
(g) Node A transmits the final data frame and resumes channel sampling.

Figure 3-11: The operation of SpeckMAC-D

3.4 Analytical Models for B-MAC and SpeckMAC

This section describes the development of preliminary analytical models for estimating power consumption for the B-MAC, SpeckMAC-B and SpeckMAC-D MAC protocols. These formulae provide comparison and possible scope for improvements in energy efficiency that SpeckMAC-B and SpeckMAC-D could possibly achieve over B-MAC, before considering the implementation of the algorithms on the ProSpeckz platform.

The notations used in the formulae are defined below:

$\mathbf{T}_{\mathbf{Action/Variable}}$: A time variable. e.g. time taken to perform a certain action or function

$\mathbf{E}_{\mathbf{Action/Variable}}$: An energy variable, in milli-watts (as all equations will be normalised to one second), e.g. energy used to perform an action or function

$\mathbf{N}_{\mathbf{Action/Variable}}$: A numeric variable, e.g. the number of times an action is performed or the value of the variable

Table 3-2 and Table 3-3 list the constants and variables, respectively, used in the calculation of the energy consumed by the radio for the different protocols. The values of the constants used in the initial formulae were based on the datasheet for the CC2420 [28] radio chip onboard the ProSpeckz, whilst variables are defined based on typical network scenarios for Specknets.

Term	Description	Unit	Value
T_{Txb}	Time taken to transmit a byte	Second	0.000032
T_{RxTx} T_{IRx}	Time taken for the radio to switch from Receive(Rx)/Idle to Tranmit (Tx) mode	Second	0.000192
T_{IRx}	Time taken for the radio to switch from Idle to Receive mode	Second	0.000192
T_{RSSI}	Time for RSSI to provide a reading	Second	0.000128
T_{Sample}	Time taken from the idle state to sample the channel for activity	Second	0.00032
P_{Rx}	Power consumption of the radio in Receive mode	mW	62.1
P_{Tx}	Power consumption of the radio in Transmit mode	mW	28.1(-25dBm) 57.4(0dBm)
P_{Idle}	Power consumption of the radio in Idle Mode	mW	1.41

Table 3-2: Constants as defined for the CC2420 radio chip

Term	Description	Unit
T_{Tx}	Time taken to transmit a data frame	Sec
T_{Rx}	Time taken to receive a data frame	Sec
$T_{TxWakeup}$	Time taken to transmit a wakeup frame (SpeckMAC-B)	Sec
T_{WGuard}	Wakeup guard time, i.e., extra time that the receiver should be turned on before the anticipated data packet arrives. (SpeckMAC-B)	Sec
$T_{Interval}$	Time between each sampling of the channel during channel listening	Sec
$T_{Preamble}$	The minimum duration for which the preamble should be sent	Sec
T_{Guard}	A guard time to allow for clock skews between nodes	Sec
T_{csma}	Duration for which the channel has to be clear before the node can assume that the channel is free	Sec
N_{Neigh}	Number of neighbours per node	Nodes
N_{Pkts}	Number of packets transmitted per second	Pkt/Sec

Table 3-3: Variables used in the in the analytical models

In addition, to simplify the comparisons, the analytical models were based on the following conditions:

i) All equations were normalised to one second.

ii) All the nodes in the network have similar number of neighbours and transmission requirements.

iii) All the equations and comparisons were be based on the power consumption of the radio only.

iv) All the algorithms would use the following equation for calculating the duration of $T_{Preamble}$:

$$\mathbf{T_{Preamble}} = T_{Interval} + T_{Sample} + T_{Guard} \quad (1)$$

v) To perform channel sampling from an idle state, a node has to turn its radio on and then sample the channel. Thus, the time taken to perform channel sampling is calculated to be:

$$\mathbf{T_{Sample}} = T_{IRx} + T_{RSSI} \quad (2)$$

vi) Before transmission of a packet, the node would turn its receiver on and wait for the Radio Strength Signal Information (RSSI) value to stabilise. It then checks whether the channel is free for the entire duration of T_{csma}. For simplifying the comparison between the different protocols, the channel was always assumed to be clear, and therefore no back-offs were considered during channel contention. For all the protocols, the total time, T_{CSMA}, in which the radio receiver is turned on to perform CSMA is calculated as follows:

$$\mathbf{T_{CSMA}} = (T_{IRx} + T_{RSSI} + Tcsma) * N_{Pkts} \quad (3)$$

vii) Channel noise was not modelled in the equations. It is assumed that the transmitted signals will always be received correctly.

viii) To provide fair comparisons with B-MAC in this analysis, the calculations for B-MAC were based on 'average case' scenarios, while those for the SpeckMAC protocols were based on 'average or worst-case' scenarios.

3.4.1 B-MAC

The equations for calculating the power consumption for the implementation of B–MAC on the CC2420 is shown in this section (adapted from [5]).

The power consumption of the radio in the case of B-MAC, $\boldsymbol{E_{BMAC}}$, is calculated from the durations for which the radio receiver or transmitter is turned on to perform the following operations: CSMA (3), transmission of the data packet (5), reception of a data packet (6), channel listening (7) and idling (8).

$$\mathbf{E_{BMAC}} = E_{CSMA} + E_{BMAC\_Tx} + E_{BMAC\_Rx} + E_{BMAC\_Listen} + E_{BMAC\_Idle} \quad (4)$$

where, ($\mathbf{E_{CSMA}} = T_{CSMA} * P_{Rx}$), ($\mathbf{E_{BMAC\_Tx}} = T_{BMAC\_Tx} * P_{Tx}$), ($\mathbf{E_{BMAC\_Rx}} = T_{BMAC\_Rx} * P_{Rx}$), ($\mathbf{E_{BMAC\_Listen}} = T_{BMAC\_Listen} * P_{Rx}$) and ($\mathbf{E_{BMAC\_Idle}} = T_{BMAC\_Idle} * P_{Idle}$)

Transmitting a data packet involves a node sending a long preamble followed by the data frame. Therefore, the duration, T_{BMAC_Tx}, for which the transmitter will be turned on is calculated as follows;

$$T_{BMAC\_Tx} = (T_{RxTx} + T_{Preamble} + T_{Tx}) * N_{Pkts} \quad (5)$$

Receiving a data packet involves nodes turning it receiver on in the event that channel activity is detected during periodic listening. These nodes will remain turned on until a packet is received and the total time, T_{BMAC_Rx}, for which the receiver is turned on for the reception of a data packet is calculated as follows:

$$T_{BMAC\_Rx} = ((0.5 * T_{Preamble}) + T_{Rx}) * N_{Pkts} * N_{Neigh} \quad (6)$$

For the remainder of the time when the radio is neither receiving nor transmitting data, B-MAC performs periodic channel listening. The duration, T_{BMAC_Listen}, for which the receiver is turned on for channel listening is calculated as follows:

$$T_{BMAC\_Listen} = \lfloor((((1 - T_{BMAC\_Tx}) - T_{BMAC\_Rx}) - T_{CSMA}) / T_{Interval})\rfloor * T_{Sample} \quad (7)$$

Finally, the total time, T_{BMAC_Idle}, for which the radio remains in an idle state, is calculated to be:

$$T_{BMAC\_Idle} = (((1 - T_{BMAC\_Tx}) - T_{BMAC\_Rx}) - T_{CSMA}) - T_{BMAC\_Listen} \quad (8)$$

The equations presented in this section will only be true if the following condition is satisfied:

$$T_{BMAC\_Listen} \geq 0$$

This is explained by the fact that a negative value for T_{BMAC_Listen} implies that the bandwidth of the channel has been exceeded.

3.4.2 SpeckMAC-B

The power consumption of the radio in the case of SpeckMAC-B, $E_{SpeckMACB}$, is calculated from the durations for which the radio receiver or transmitter is turned on to perform the following functions: CSMA (3), transmission of the data packet (10), reception of a data packet (11), channel listening (13) and idling (14).

$$\mathbf{E_{SpeckMACB}} = E_{CSMA} + E_{SpeckMACB\_Tx} + E_{SpeckMACB\_Rx} + E_{SpeckMACB\_Listen} + E_{SpeckMACB\_Idle} \quad (9)$$

where, ($\mathbf{E_{CSMA}} = T_{CSMA} * P_{Rx}$), ($\mathbf{E_{SpeckMACB\_Tx}} = T_{SpeckMACB\_Tx} * P_{Tx}$), ($\mathbf{E_{SpeckMACB\_Rx}} = T_{SpeckMACB\_Rx} * P_{Rx}$), ($\mathbf{E_{SpeckMACB\_Listen}} = T_{SpeckMACB\_Listen} * P_{Rx}$) and ($\mathbf{E_{SpeckMACB\_Idle}} = T_{SpeckMACB\_Idle} * P_{Idle}$)

In order to transmit a data packet, a node will transmit wakeup frames repetitively for a minimum duration of time, $T_{Preamble}$, as shown in Figure 3-10. After sending the last wakeup frame, the node will transmit the data frame. Therefore, the time spent in transmissions is calculated as follows:

$$\mathbf{T_{SpeckMACB\_Tx}} = (T_{RxTx} + (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + T_{Tx}) * N_{Pkts} \quad (10)$$

For the reception of the data packet, in the worst case, channel listening will occur in between wakeup frames. Therefore, the receiver will stay turned on for a maximum duration of time, $2 * T_{TwWakeup}$, in order for the node to receive a complete wakeup frame. The radio will then be placed in the idle state during the back-off period. Based on the information from the received wakeup frame, the radio would be turned on again for a period, T_{WGuard}, before the anticipated arrival of the data frame. This provides allowance for delays due to transmission, propagation and reception of the wakeup frame. The receiver will turn off once the data frame is received. Therefore, the total time that the radio is turned on for receiving data packets is calculated to be:

$$\mathbf{T_{SpeckMACB\_Rx}} = ((2 * T_{TxWakeup}) + T_{IRx} + T_{WGuard} + T_{Rx}) * N_{Pkts} * N_{Neigh} \quad (11)$$

On average, the duration for which the radio would be idling due to the node performing back-off based on the timing information extracted from the received wakeup frames is calculated as follows:

$$\mathbf{T_{SpeckMACB\_Backoff}} = (((\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) - ((2 * T_{TxWakeup}) + T_{IRx} + T_{WGuard})) * N_{Pkts} * N_{Neigh} / 2) \quad (12)$$

For the remainder of time when the radio neither receiving nor transmitting data, SpeckMAC-B will listen to the channel periodically. The duration of time for which the receiver is turned on for this purpose is derived to be:

$$\mathbf{T_{SpeckMACB\_Listen}} = \lfloor((((1 - T_{SpeckMACB\_Tx}) - T_{SpeckMACB\_Rx}) - T_{SpeckMACB\_Backoff}) - T_{CSMA}) / T_{Interval})\rfloor * T_{Sample} \quad (13)$$

Finally, the time that the radio is left in the idle state is therefore:

$$\mathbf{T_{SpeckMACB\_Idle}} = (((1 - T_{SpeckMACB\_Tx}) - T_{SpeckMACB\_Rx}) - T_{CSMA}) - T_{SpeckMACB\_Listen} \quad (14)$$

As worst-case assumptions are being used, the equations presented in this section will only be true if the following conditions are satisfied:

$T_{Preamble} \geq ((2 * T_{TxWakeup}) + T_{IRx} + T_{WGuard}$

$T_{SpeckMACB\_Listen} \geq 0$

The latter condition is explained by the fact that a negative value for $T_{SpeckMACB\_Listen}$ implies that the bandwidth of the channel has been exceeded.

3.4.3 SpeckMAC-D

The power consumption of the radio in the case of SpeckMAC-B, $E_{SpeckMACB}$, is calculated based on the durations for which the radio receiver or transmitter is turned on to perform the following: CSMA (3), transmission of the data packet (16), reception of a data packet (17), channel listening (19) and idling (20).

$$\mathbf{E_{SpeckMACD}} = E_{CSMA} + E_{SpeckMACD\_Tx} + E_{SpeckMACD\_Rx} + E_{SpeckMACD\_Listen} + E_{SpeckMACD\_Idle} \quad (15)$$

where ($\mathbf{E_{CSMA}} = T_{CSMA} * P_{Rx}$), ($\mathbf{E_{SpeckMACD\_Tx}} = T_{SpeckMACD\_Tx} * P_{Tx}$), ($\mathbf{E_{SpeckMACD\_Rx}} = T_{SpeckMACD\_Rx} * P_{Rx}$), ($\mathbf{E_{SpeckMACD\_Listen}} = T_{SpeckMACD\_Listen} * P_{Rx}$) and ($\mathbf{E_{SpeckMACD\_Idle}} = T_{SpeckMACD\_Idle} * PIdle$)

When transmitting a data packet, SpeckMAC_D sends the data frame repetitively for a minimum duration of $T_{Preamble}$, as shown in Figure 3-11. After this, the data frame is transmitted once more. The total time for which the transmitter is turned on is therefore:

$$\mathbf{T_{SpeckMACD\_Tx}} = (T_{RxTx} + ((\lceil T_{Preamble} / T_{Tx} \rceil + 1)) * T_{Tx}) * N_{Pkts} \quad (16)$$

As in the previous section, worst-case conditions have been assumed for fair comparisons with B-MAC, i.e., the channel is sampled between transmissions of two data frames during channel listening. The radio will therefore be turned on at the beginning of a data frame and will remain so until the next data frame is received, i.e., the radio is turned on for a maximum duration of $2 * T_{Rx}$. The total time that the receiver is turned on for the reception of the data packets is calculated as:

$$\mathbf{T_{SpeckMACD\_Rx}} = (2 * T_{Rx}) * N_{Pkts} * N_{Neigh} \quad (17)$$

After receiving a data frame successfully, the node returns its radio to an idle state for a period equivalent to $T_{Preamble}$. The total time spent in this back-off state is expressed as follows:

$$\mathbf{T_{SpeckMACD\_Backoff}} = T_{Preamble} * N_{Pkts} * N_{Neigh} \quad (18)$$

When the node is either receiving or transmitting data packets, it listens to the channel periodically. The total time the receiver it turned on to perform channel sampling is calculated as:

$$\mathbf{T_{SpeckMACD\_Listen}} = \lfloor(((((1 - T_{SpeckMACD\_Tx}) - T_{SpeckMACD\_Rx}) - T_{SpeckMACD\_Backoff}) - T_{CSMA}) / T_{Interval})\rfloor * T_{Sample} \quad (19)$$

Finally, the time for which the radio is in the idle state is calculated as:

$$\mathbf{T_{SpeckMACD\_Idle}} = (((1 - T_{SpeckMACD\_Tx}) - T_{SpeckMACD\_Rx}) - T_{CSMA}) - T_{SpeckMACD\_Listen} \quad (20)$$

The equations presented in this section would be true only if the following condition is satisfied:

$$T_{SpeckMACD\_Listen} \geq 0$$

This is explained by the fact that a negative value for $T_{SpeckMACD\_Listen}$ implies that the bandwidth of the channel has been exceeded.

3.4.4 Comparisons based on the formulae

A preliminary evaluation of the protocols was based on the derived formulae. The formats of the data and wakeup frames are illustrated in Figure 3-12.

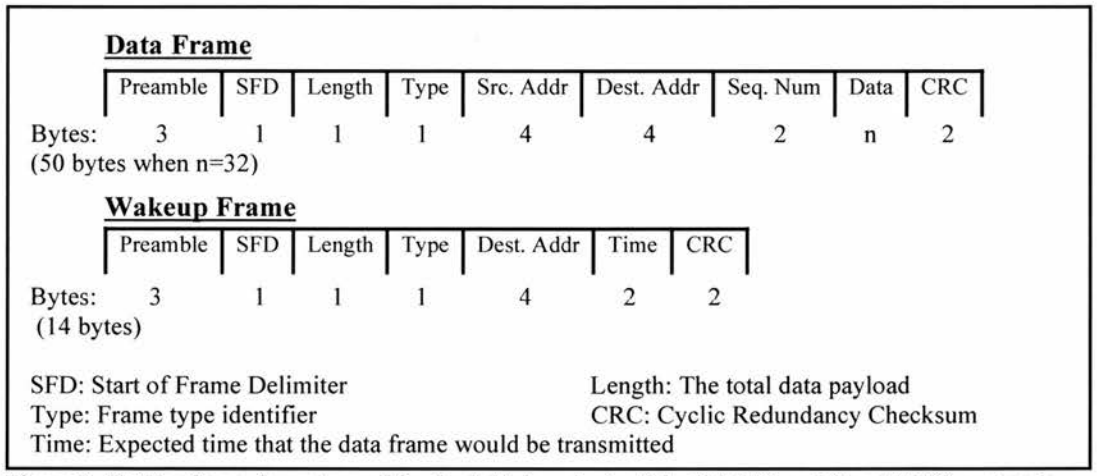


Figure 3-12: The frame formats used for the initial analysis of the B-MAC and SpeckMAC protocols

For the preliminary evaluation, a data payload of 32 bytes was used and the values for the variables T_{Tx}, T_{Rx} and $T_{TxWakeup}$ are evaluated to be:

$T_{Tx} = T_{Rx} = Size\ of\ Data\ Frame * T_{Txb} = 50\ bytes * 0.000032s = 0.0016\ s$

$T_{TxWakeup} = Size\ of\ Wakeup\ Frame * T_{Txb} = 14\ bytes * 0.00032s = 0.000448\ s$

The other parameters in Table 3-3 were assigned the following values:

$T_{WGuard} = 0.001\ s$

$T_{Guard} = 0.00068\ s$

$T_{csma} = 0.001\ s$

Based on the parameters, equations (1) to (20) was solved to determine the optimal value of $T_{Interval}$ for each protocol, given the number of packets expected to be transmitted per second, and the number of one-hop neighbours assumed for each node for any given application.

For example, an application in which each node transmits one packet per second and has eleven neighbours, the power consumed by the radio for the application across the range of $T_{Interval}$ was calculated using equations (4), (9) and (15) and plotted in

Figure 3-13. It was observed that B-MAC performs optimally when $T_{Interval}$ was set to 0.0067s, and the resulting power consumption for the radio was 8.34mW. For the case of SpeckMAC-B and SpeckMAC-D, it was observed the optimal power consumption by the radio was 6.06mW and 5.70mW, respectively, when $T_{Interval}$ was set to 0.0067s. Using these optimal values of $T_{Interval}$ in each case, both variations of SpeckMAC consumed less energy than B-MAC. In fact, across the range of values for $T_{Interval}$, both SpeckMAC protocols had lower power consumptions than B-MAC.

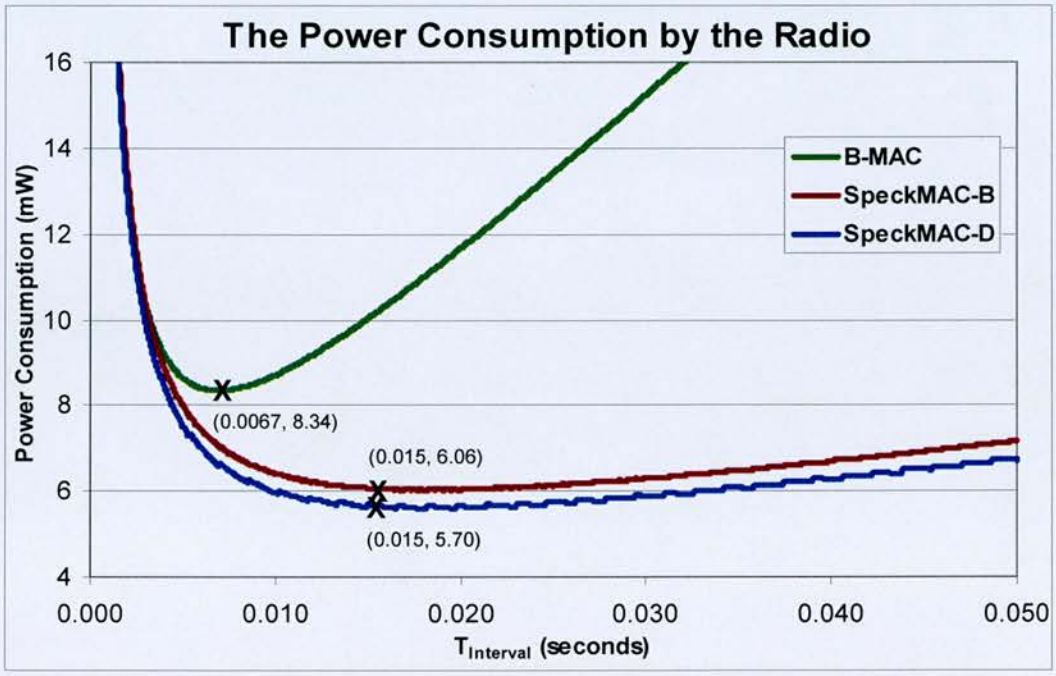


Figure 3-13: The power consumed by the radio across a range of values for $T_{Interval}$ using the three protocols in a scenario in which each node transmitted one packet every second, and had eleven nodes as neighbours

Figure 3-14 shows the power consumption calculated for the protocols for a selection of network load and node densities, with the optimal value of $T_{Interval}$ selected in each case and the payload of 32 bytes per packet. The optimal value of $T_{Interval}$ for each case is determined using the same method as shown in Figure 3-13 and results in the lowest radio power consumption given the number of packets transmitted per second and the number of one-hop neighbours. This preliminary analysis based on analytical models for power consumption, albeit simplified, indicated that both variations of SpeckMAC will consume less energy than B-MAC, for low traffic requirement between 0.001 to 1 packets per second with node densities ranging from 5 to 40

neighbours per node. The next step was to implement all three protocols on the physical ProSpeckz platforms to both confirm and measure the improvements in practice.

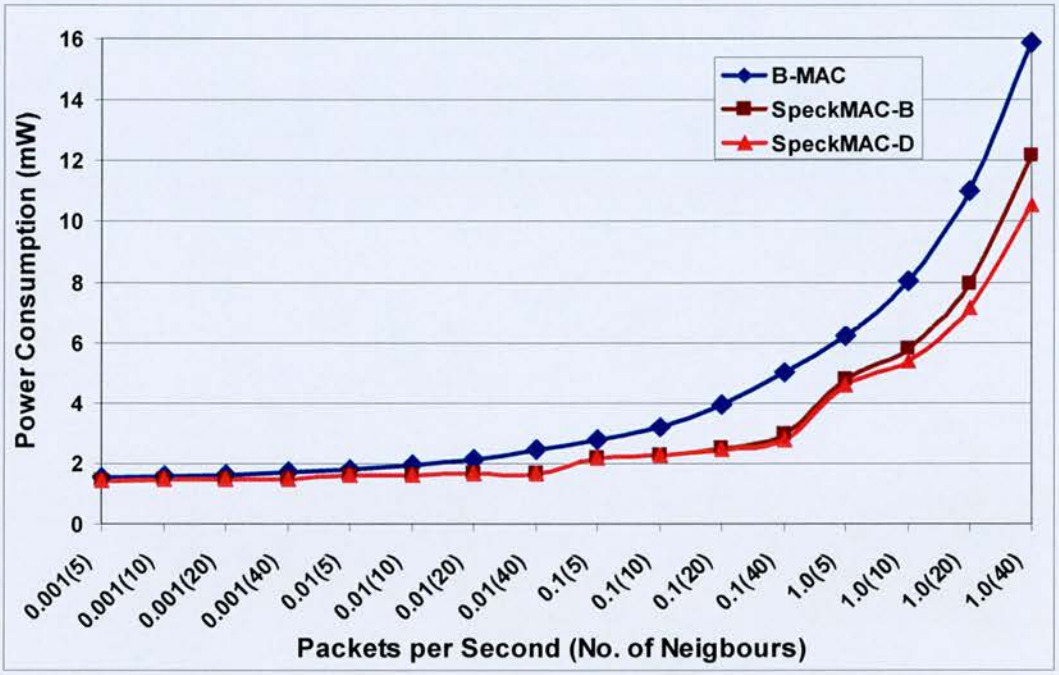


Figure 3-14: The power consumption by the radio running the three protocols under a selection of network loads and node densities, using optimal $T_{Interval}$ in each case

3.5 Implementation on the ProSpeckz Platform

This section describes the details of the implementation, together with the modifications to the algorithms to ensure fair comparisons between the protocols, i.e. both SpeckMAC protocols and B-MAC are required to support the same functionalities.

The default "buffered" transmission mode provided in the CC2420 radio could not be used to implement the SpeckMAC algorithms. This would require the packets to be transferred from the PSoC to the buffer of the CC2420 before packet transmissions, as the processing and communications are handled by two different sub-systems as the ProSpeckz. This causes a gap between successive transmissions of the wakeup and data frames as the content of the frames had to be transferred via the Serial Peripheral Interface (SPI) before transmission. This gap causes SpeckMAC to malfunction as it is possible that a node listens to the channel during this gap and

falsely assume the channel to be free therefore missing the packet. To get around this problem, the initial implementation of the algorithms used the 'Looping' transmission mode as described in Appendix D.1, as this mode allows the frames in the transmit buffer of the CC2420 to be sent repetitively without any gaps in transmission. However, a small gap in the transmission still occurs when switching between the transmission of the preamble (for B-MAC) or the wakeup frame (for SpeckMAC-B) and transmission of the data frame. Although this gap does not cause the algorithms to malfunction, it degrades the performance of B-MAC and SpeckMAC-B. Therefore, the 'Serial' transmission mode of the CC2420 was used to minimise this gap in which the PSoC treats the CC2420 as a wireless serial link, without the need for the frames to be uploaded into the transmit buffer of the CC2420 prior to transmission. In this transmission mode, the CC2420 does not handle packet generation, instead the PSoC was responsible for the generation of the preamble bytes, the start-of-frame delimiter, the data packet and the cyclic redundancy check bytes. The PSoC was therefore able to generate the long preambles required by B-MAC, as well as efficiently retransmit the data or wakeup frames for the SpeckMAC protocols without gaps between retransmissions.

Another modification was the inclusion of backoff timing information in the SpeckMAC-D algorithm, a further development to the algorithm described in the Section 3.3 and in the paper in Appendix D.1. The backoff timing information allows the receiver to determine when the transmitter completes its transmissions. The inclusion of this feature in SpeckMAC-D enabled a fair comparison between SpeckMAC-D and B-MAC. Without this timing information, SpeckMAC-D will not be able to determine the end of transmission which would make it difficult to design a system in which transmissions of acknowledgements are required to be sent immediately after the reception of a data packet.

Furthermore, to ensure better bit synchronisation as well as to conform to the recommendation in the CC2420 datasheet, four preamble bytes were used instead of three. The new data frame formats for the implementation of the three protocols are shown in Figure 3-15.

Data Frame (SpeckMAC-D)

Preamble	SFD	Length	Type	Src. Addr	Dest. Addr	Seq. Num	Data	End Time	CRC
Bytes: 4	1	1	1	4	4	2	n	2	2

(Frame length is 53 bytes when n=32)

Data Frame (B-MAC, SpeckMAC-B)

Preamble	SFD	Length	Type	Src. Addr	Dest. Addr	Seq. Num	Data	CRC
Bytes: x	1	1	1	4	4	2	n	2

(for SpeckMAC-B, x=4 thus frame length is 51 bytes when n=32)
(For B-MAC, x depends on the $T_{Interval}$ selected)

Wakeup Frame

Preamble	SFD	Length	Type	Dest. Addr	Time	CRC
Bytes: 3	1	1	1	4	2	2

(Wakeup frame length is 14 bytes)

SFD: Start of Frame Delimiter
Length: The total data payload
Type: Frame type identifier
CRC: Cyclic Redundancy Checksum
Time: Expected time that the data frame would be transmitted
End Time: Expected time that the transmission would end for the data packet

Figure 3-15: The frame formats used for the implementation of the B-MAC and SpeckMAC protocols

To optimise the algorithms, the transmitter turnaround times, T_{ITx} and T_{RxTx}, were set to 128us instead of 192us (by changing the value of the TX_CTRL register on the CC2420) as 128us was sufficient for the algorithms to function correctly.

CSMA was chosen as the multiple access technique used in all three implementations due to the ease of implementation and for the reasons described in Section 3.2.3.5. A non-persistent CSMA was employed with a constant channel sampling time in order to conserve energy. If the channel was busy, the node will attempt to retransmit the packet after a random-delay, within a fixed backoff window. Furthermore, there was an initial random delay of up to 50ms a packet is transmitted. This prevented all the nodes from transmitting at the same time in response to a received packet. Evaluations of this variation of CSMA (initial-delay, fixed sampling and fixed backoff window) [81] had demonstrated low energy consumption while maintaining the same level of throughput as other versions of CSMA, and therefore was used in the experiments carried out in this thesis. Finally, for channel sampling and the reception of data packets, the activities carried out by nodes when not transmitting a packet are shown in Appendix B.

3.6 Validating the mathematical models

Based on the changes to the algorithm, and following their implementation on the ProSpeckz, the initial equations used for modelling the power consumption (Section 3.4) were modified. These equations are important as they would be used to determine the optimal value for $T_{Interval}$ for a given network of Specks and its traffic environment. The mathematical model had to be verified by comparing the model with measurements recorded from the execution of the algorithms on the PerSpeckz-64. Validations were carried to both ensure that the formulae were correct and to ensure that the algorithms were implemented correctly on the ProSpeckz platform.

3.6.1 Experiment setup for testing and validating the algorithms using the PerSpeckz-64

For the equations and the implementation to be validated, experiments were performed on the PerSpeckz-64 with numbers of nodes, ranging from 2 to 20, randomly chosen to participate in communication within a one-hop network. Each node transmitted a 32 byte data payload once every second. For each scenario, 40 iterations, each lasting 40 seconds were executed, transmitting a total of 1600 packets for each scenario. Statistics on the receiver, transmitter and processor were recorded for the duration of 20 seconds for each execution, and recording commenced 10 seconds after the node sent its first packet, which allowed the network to stabilise before any readings were taken. Finally, a value of 15ms was chosen for $T_{Interval}$ in all the algorithms.

3.6.2 Measurements of T_{ITx}, T_{RxTx}, T_{IRx}, T_{RSSI} and T_{Sample}

Firstly, the transmitter turn-around times, T_{ITx} and T_{RxTx}, were measured using ProSpeckz by using the following codes to trigger an output pin, *Pin 1*, when the transmitter was activated.

Turn on Pin 1
Send command to transmit
Turn off Pin 1

An oscilloscope was used to measure the output of *Pin 1* against the RF output of the CC2420 to determine the turn-around times. In the captured waveform shown in Figure 3-16, T_{ITx} and T_{RxTx} were determined to be 131.7us.

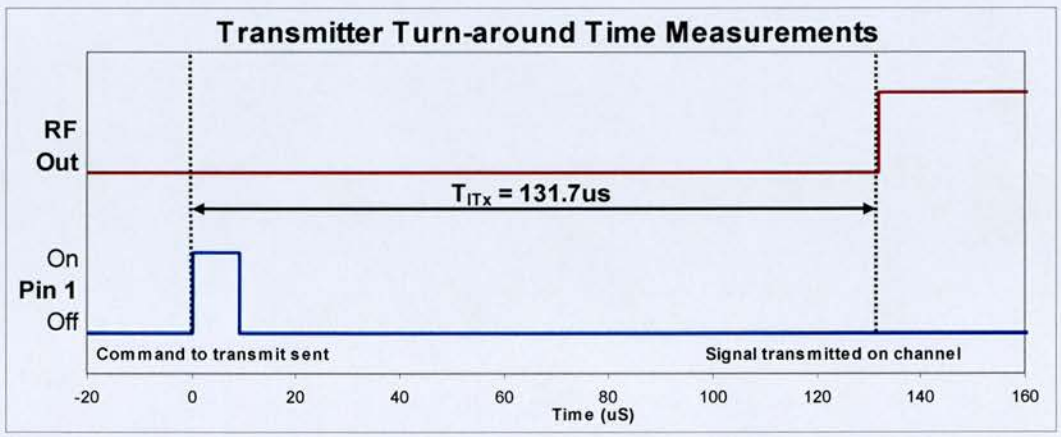


Figure 3-16: Captured waveforms for the measurement of T_{ITx}

Next, the values of T_{IRx} and T_{RSSI} were measured by executing the following pseudo codes on the ProSpeckz.

Send command to turn on radio receiver
Turn on Pin 1 and 2
Wait for PPL to be locked
Turn off Pin 1
Wait for RSSI to be stable
Turn off Pin 2

Measurements from the captured waveform shown in Figure 3-17 determined that the values of T_{IRx} to be 149us, and T_{RSSI} to be 140.6us.

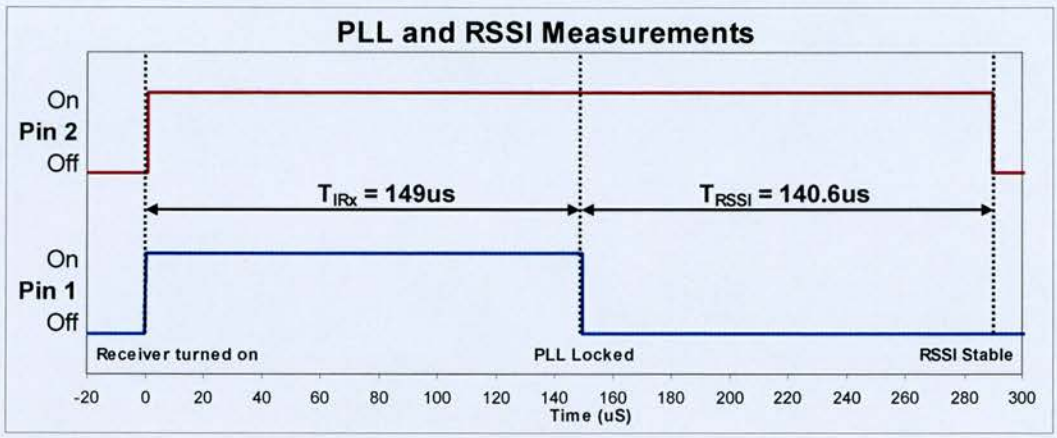


Figure 3-17: Captured waveforms for the measurement of T_{IRx} and T_{RSSI}

Using the values T_{ITx}, T_{RSSI} and T_{IRx} measured, equations presented in Section 3.4 were either validated or modified to fit closer to the actual implementation. In order to measure the radio and CPU usage, three output pins on the ProSpeckz were used to indicate the status of the radio receiver, the radio transmitter and the CPU. Waveforms from these output pins were captured during the execution of the various algorithms using a PC-based oscilloscope with a maximum sampling rate of 20MSamples per second. These captured waveforms were then used to generate timing diagrams enabled the operations of the physical implementation to be compared to the mathematical models. For example, Figure 3-18 shows the timing diagram of a ProSpeckz performing channel sampling, which is similar for B-MAC, SpeckMAC-B and SpeckMAC-D.

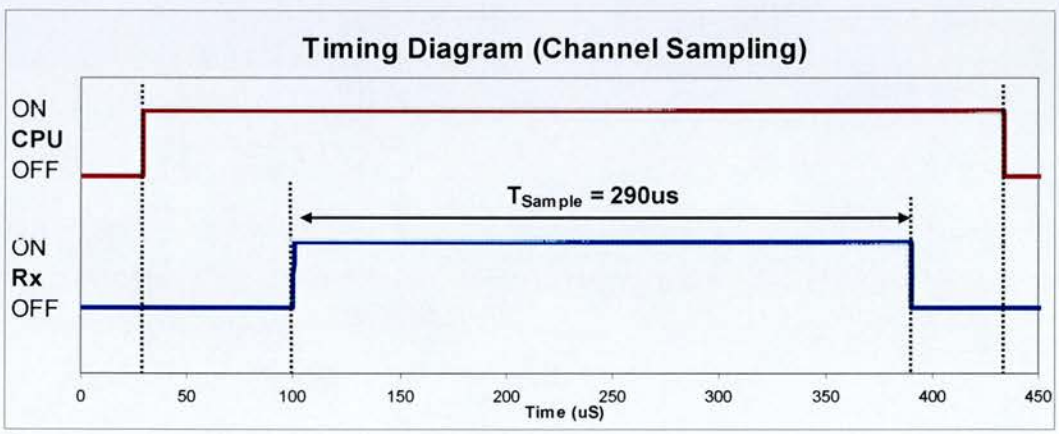


Figure 3-18: Captured waveforms for the measurement of T_{Sample}

The value of T_{Sample} was measured to be 290us and by using the measured values of T_{IRx} and T_{RSSI}, equation (2) was validated as;

$\mathbf{T_{Sample}} = T_{IRx} + T_{RSSI}$ ...from (2)

$= 149us + 140.6us = 289.6us$

$\approx 290us$ (diff: 0.4us, 0.14%) ...from Figure 3-18

Similarly, using the values of T_{ITx}, T_{IRx} and T_{RSSI} measured in the physical implementations, the equations derived in the Section 3.4 and the modified equations were validated by comparing against the measurements for the ProSpeckz platform.

3.6.3 Equations for modelling the radio timings for B-MAC

Figure 3-19 shows the timing diagram captured from the ProSpeckz reflecting the states of the CPU, radio transmitter and receiver when it was transmitting a packet using B-MAC. Using these timing measurements, equation (3) was validated to be:

$\mathbf{T_{CSMA}} = (T_{IRx} + T_{RSSI} + T_{csma}) * N_{pkt}$ ...from (3)

$= 149us + 140.6us + 1ms = 1.2896ms$

$\approx a = 1.299ms$ (diff: 9.4us, 0.72%) ...with ref. to Figure 3-19

Therefore, the timing information derived for T_{CSMA} corresponds to the value measured on the ProSpeckz. In a similar manner, the timing information derived for T_{BMAC_Tx} (5) was compared to physical implementation:

$\mathbf{T_{BMAC\_Tx}} = (T_{RxTx} + T_{Preamble} + T_{Tx}) * N_{Pkts}$ ...from (5)

$= (131.7us + 16ms + 1.632ms) * 1$

$= 17.7637ms$

$\approx b + d$ ... with ref. to Figure 3-19

$= 16.13ms + 1.785ms = 17.915ms$ (diff: 151.3us, 0.84%)

There was an appreciable difference between equation (5) and the measurements from the physical implementation. This inaccuracy can be explained by the fact that the timing calculations described in (5) did not reflect the gap, T_{TxGap}, in the

transmission caused by the processing time needed for the PSoC to switch between sending the preamble and the data frame. In the ProSpeckz implementation, after sending the preamble, the radio is placed in an idle state whilst the PSoC prepares to send the data frame. After which, the radio is set to the transmit mode and the data frame is sent after a duration of time, T_{ITx}. The equation for the duration for which the transmitter is turned on is modified as follows;

$$\mathbf{T_{BMAC\_Tx}} = (T_{RxTx} + T_{Preamble} + T_{ITx} + T_{Tx}) * N_{pkts} \quad (21)$$

The new equation was then validated against the physical measurements:

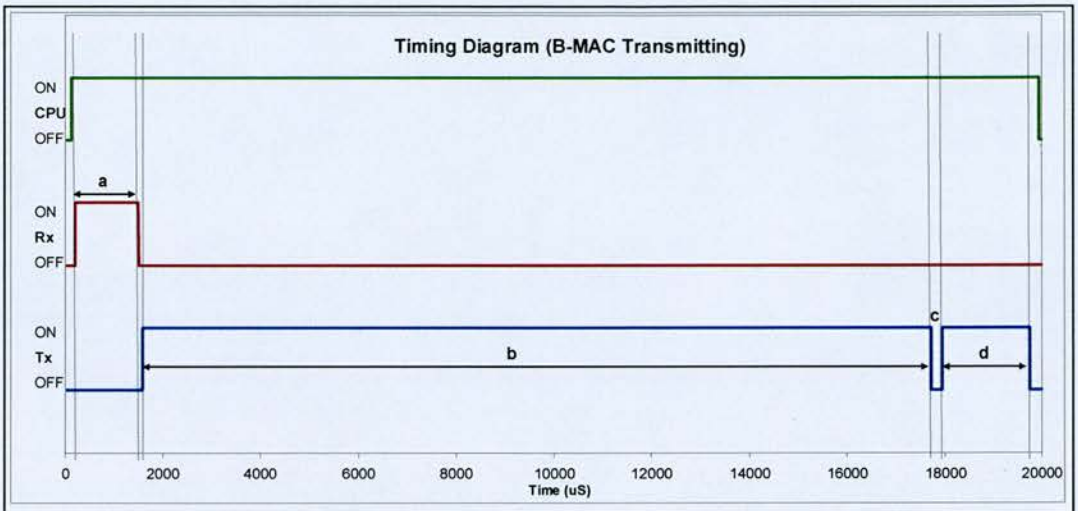
$\mathbf{T_{BMAC\_Tx}}$ = (131.7us + 16ms + 131.7us + 1.632ms) * 1 …from (21)

= 17.8954ms

≈ b + d … with ref. to Figure 3-19

= 16.13ms + 1.785ms = 17.915ms (diff: 19.6us, 0.11%)

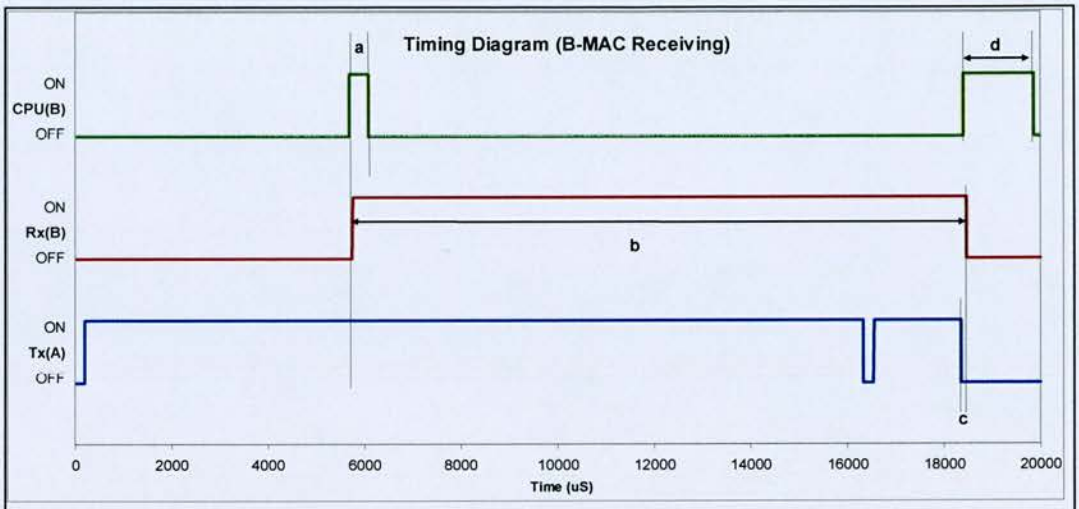
The refined version of the equation maps closer to the measurements and replaced the original one for the calculation of T_{BMAC_Tx}.



a. The receiver turned on to perform CSMA. (T_{CSMA}=1.299ms)
b. The transmitter transmits the preambles that consist of a block of bytes with the value of 0x00. (16.13ms)
c. The PSoC prepares for the transmission of the data frame after the preamble had been transmitted. (T_{TxGap} = 227us)
d. The transmitter transmits the data frame. (1.785ms)

Figure 3-19: Timing diagram captured from a ProSpeckz during data packet transmission using B-MAC

Figure 3-20 shows the timing diagram of a ProSpeckz, *A*, transmitting a packet to another ProSpeckz, *B*, using B-MAC to perform channel sampling. A timer interrupt wakes up the CPU periodically which in turn, turns on the radio receiver. As the channel was detected to be busy, the radio was left on and the CPU returned to sleep to conserve energy. The receiver was left on until a data frame is received. Upon receiving the data frame, the receiver sends an interrupt to the CPU to wake it up. On waking up, the CPU turns off the radio receiver and processes the received data packet. It then goes back to sleep after processing the data packet and periodic channel sampling is resumed. In equation (6), the total duration for which the radio receiver was turned on to receive data packets, as shown in Figure 3-20b, was based on the average-case or best-case scenario. In order to achieve a more accurate mathematical representation based on the average time that the radio receiver is turned on, the worst-case and best-case scenarios were derived using values measured from the physical implementation.



a. CPU woke up to perform channel sampling. As the channel was busy, the radio receiver was kept on.
b. The radio was kept on until the data frame was received. (12.723ms)
c. The processor turned off the radio after the data frame was received. (T_{RxEnd} = 123.3us)
d. The received packet is processed by the CPU.

Figure 3-20: Timing diagram captured from *ProSpeckz A* and *ProSpeckz B* when *B* was receiving a data packet from *A* using B-MAC

Figure 3-21 shows the possible best-case and worst-case scenarios for which a node, *A*, would turn its radio receiver on for receiving a data packet from another node, *B*. In the best case, *A* would sample the channel just before *B* transmits the start of the preamble. *A* would then detect that the channel was not busy and would turn the receiver off. *A* would only detect the preamble from *B* when the next channel sampling is performed. Therefore, the total time that *A* would turn its receiver on for the reception of data packets in the best-case scenario would be:

$$^{Best}T_{BMAC\_Rx} = T_A - T_{Interval} * N_{Pkts} * N_{Neigh} \quad \text{... with ref. to Figure 3-21}$$

$$= (T_{IRx}+T_{RSSI}+T_{Preamble}+T_{TxGap}+T_{ITx}+T_{Tx}+T_{RxEnd} - T_{Interval})*N_{Pkts}*N_{Neigh} \quad (22)$$

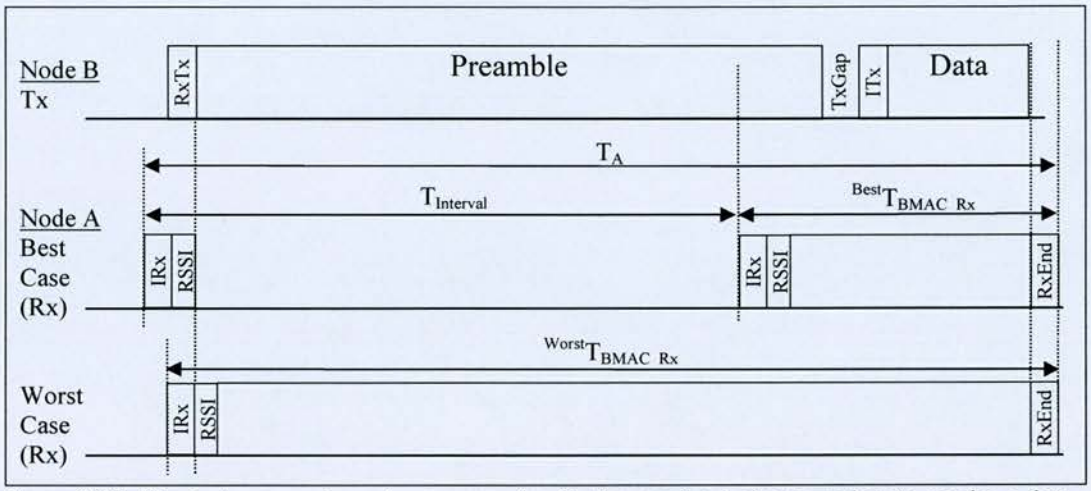


Figure 3-21: The best-case and worst-case scenarios for the receiver being turned on to receive a data packet using B-MAC

In the worst case, *A* would detect channel activity by sampling the start of the preamble sent by *B*. Therefore the receiver of *A* would be left turned on for the remaining duration until *B* completes transmitting the data frame. Therefore, the worst case equation is derived as:

$$^{Worst}\mathbf{T_{BMAC\_Rx}} = (T_{IRx}+T_{Preamble}+T_{TxGap}+T_{ITx}+T_{Tx}+T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (23)$$

The average time that *A* will turn its receiver on for reception of a packet from *B*, $^{Avg}T_{BMAC\_Rx}$, is calculated as the average between the best and worst cases. Therefore,

$$\begin{aligned} ^{Avg}\mathbf{T_{BMAC\_Rx}} &= (^{Worst}T_{BMAC\_Rx}+^{Best}T_{BMAC\_Rx})/2 \\ &= ((T_{Preamble}+T_{TxGap}+T_{ITx}+T_{Tx}+T_{RxEnd}+T_{IRx}) + ((T_{RSSI}- \\ &\quad T_{Interval})/2))*N_{Pkts}*N_{Neigh} \end{aligned} \quad (24)$$

For the time that the node does not transmit data or receive data, it will perform channel sampling at intervals of time, $T_{Interval}$, to check for channel activity. Since the duration that the receiver is turned on for sampling a busy channel was already accounted for by T_{BMAC_RX}, (N_{Pkts} * N_{Neigh}) was subtracted from the number of

samples taken per second. Therefore, the number of times per second that the channel was sampled on an idle channel is calculated as:

$$\mathbf{N_{BMAC\_Samples}} = ((1.0-(T_{BMAC\_Tx}+T_{BMAC\_Rx}))/T_{Interval}) -(N_{Pkts}*N_{Neigh})$$

where the value of T_{BMAC_Rx} is substituted by equations (22), (23) or (24) for the best-, worst- and average-cases, respectively.

The new equation for calculating the total duration that the radio receiver was turned on to perform idle channel sampling is therefore:

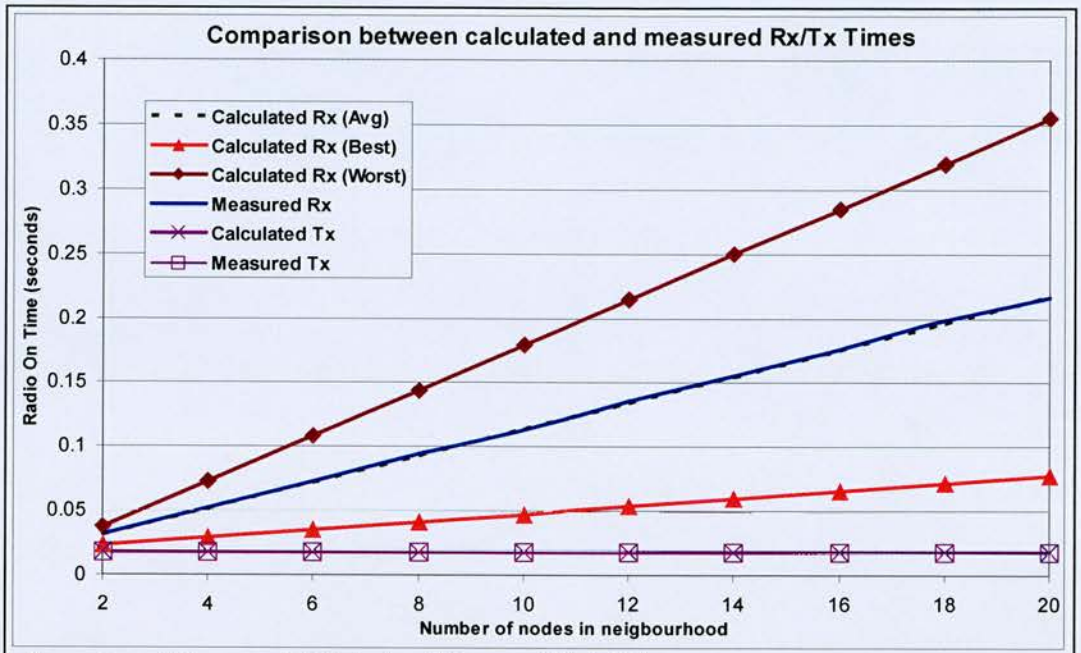
$$\mathbf{T_{BMAC\_Listen}} = N_{BMAC\_Samples}*T_{Sample} \quad (25)$$

Using the equations (3) and (22-25), the total duration that the radio receiver would be turned on when using B-MAC is:

$$\mathbf{T_{BMAC\_Total\_Rx}} = T_{CSMA} + T_{BMAC\_Rx} + T_{BMAC\_Listen} \quad (26)$$

where T_{BMAC_Rx} is substituted by equations (22), (23) or (24) for the best-, worst- and average-cases, respectively.

To validate the equations for calculating the receiver turn-on time (26) and the transmitter turn-on time (21), comparisons between the measured on-times for the receiver/transmitter and the equations were made using the same physical setup describe in Section 3.6.1. The results of the comparison are shown in Figure 3-22. It can be observed that equation (21) maps very closely to the measured transmitter turn-on times, and was on average, deviated by just 0.02% from the measured values. On the other hand, equation (26) also maps reasonably closely to the measured receiver turn-on times with an average difference of 1.24% from the measured values. It can be concluded that the equations presented in this section correspond to the measurements obtained from the physical implementation of the B-MAC protocol on the ProSpeckz.



Percentage difference = [1-(Calculated/Measured)] * 100%

	2	4	6	8	10	12	14	16	18	20
Rx	2.97%	2.53%	1.45%	1.74%	0.34%	1.30%	0.30%	0.46%	1.24%	0.10%
Tx	0.03%	0.03%	0.03%	0.03%	0.01%	0.01%	0.03%	0.03%	0.02%	0.03%

Figure 3-22: Comparisons between the measured and the calculated values for the total duration for which the radio receiver and transmitter were turned on using B-MAC

3.6.4 Equations for modelling the radio timings for SpeckMAC-B

Figure 3-23 shows the timing diagrams of the state of the CPU, radio transmitter and receiver measured on the ProSpeckz when it was transmitting a packet using SpeckMAC-B. Similar to the discussion in the previous section, the original equation (9) for calculating the transmitter turn-on duration did not take into consideration the gap between sending the last wakeup frame and the data frame. This gap was due to the processing time for switching between the transmission of the last wakeup frame and the data frame. This gap caused the radio to switch back into the idle state before the data frame was transmitted. Therefore, the equation for $T_{SpeckMACB\_Tx}$ was modified as follows:

$$\mathbf{T_{SpeckMACB\_Tx}} = (T_{RxTx}+(\lceil T_{Preamble}/T_{TxWakeup} \rceil * T_{TxWakeup})+T_{ITx}+T_{Tx})*N_{Pkts} \quad (27)$$

Based on the measurements from Figure 3-23, equation (27) was validated as:

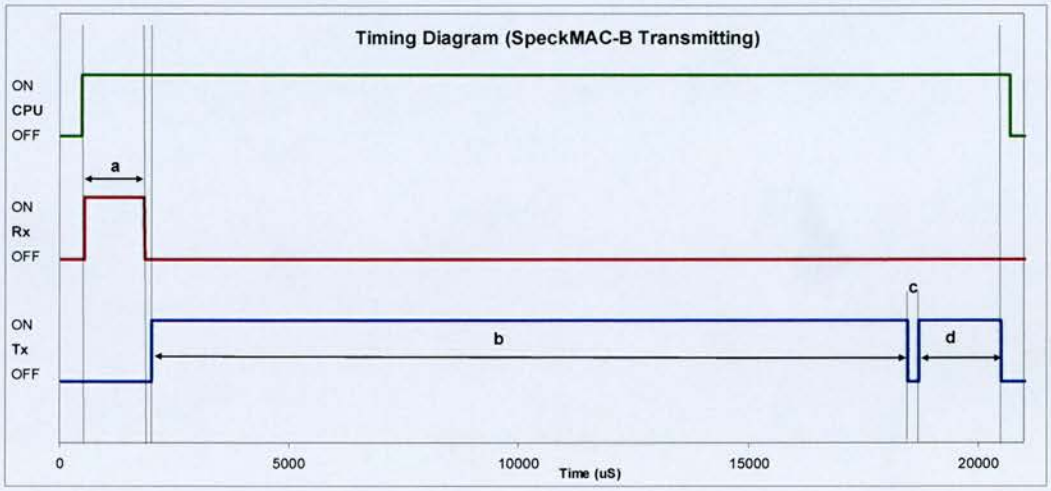
$T_{SpeckMACB\_Tx}$

$= (131.7s + (\lceil 16ms / (480us) \rceil * 480us) + 131.9us + 1.632ms) * 1$ ...from (27)

$= 18.2154ms$

$\approx b + d$ ... from Figure 3-23

$= 16.47ms + 1.785ms = 18.255ms$ (diff: 39.6us, 0.22%)

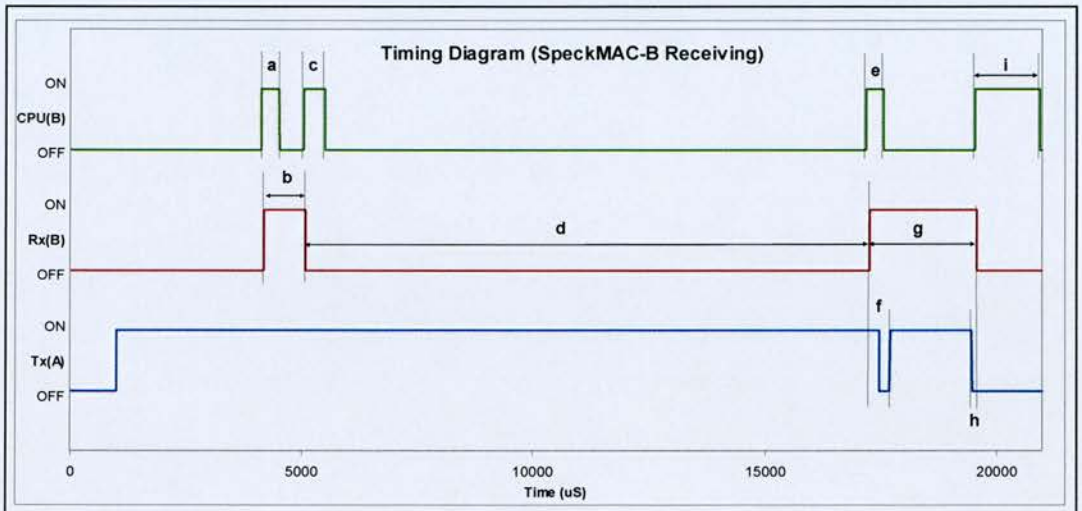


a. The receiver was turned on to perform CSMA. (T_{CSMA}=1.299ms)
b. The transmitter transmits the wakeup frame for a minimum duration specified by $T_{Preamble}$. (16.47ms)
c. The PSoC prepares for the transmission of the data frame after the last wakeup frame had been transmitted. (T_{TxGap} = 227us)
d. The transmitter transmits the data frame. (1.785ms)

Figure 3-23: Timing diagram measured for a ProSpeckz during data packet transmission employing SpeckMAC-B

Figure 3-24 shows the timing diagrams measured from a ProSpeckz, *A*, transmitting a packet to another ProSpeckz, *B*, using SpeckMAC-B. Similar to the case of B-MAC, the CPU on *ProSpeckz B* will turn on the radio receiver to perform channel sampling and detect the transmission by *A*. *B* will then keep the radio receiver on while placing the CPU back into the sleep mode, until a wakeup frame is received by the radio receiver. On receiving the data frame, the receiver will send an interrupt to the CPU to wake it up. On waking up, the CPU turns the radio receiver off and

process the received data packet. It then goes back to sleep after the data packet was processed and periodic channel sampling is resumed. From equation (11), the total duration for which the radio receiver was turned on to receive data packets, as shown in Figure 3-24b and Figure 3-24g, was based on the average-or-worse case scenarios. In order to achieve a more accurate mathematical representation based on the average time that the radio receiver is turned on, the worst-case and best-case scenarios were derived using values measured from the physical implementation.



a. CPU wakes up to perform channel sampling. As the channel is busy, the radio receiver is kept on.
b. The radio is kept on until the wakeup frame is received. (884.76us)
c. The CPU is woken up to process the received wakeup frame. The timing information is extracted from the wakeup frame and a back-off timer is activated. The CPU returns to sleep and is woken up when the back-off timer expires.
d. The radio receiver is in the idle state during the back-off period.
e. The back-off timer expires and the CPU is woken up.
f. A guard-time, as specified in T_{WGuard}, is used to ensure receiver is turned on before the anticipated data frame is transmitted.
g. The receiver is kept turned on until the data frame is received.
h. The CPU turns off the receiver after the data frame is received. (T_{RxEnd} = 123.3us)
i. The received data frame is processed by the CPU.

Figure 3-24: Timing diagram captured from *ProSpeckz A* and *ProSpeckz B* when *B* was receiving a data packet from *A* using SpeckMAC-B

Figure 3-25 shows the best- and worst-case scenarios in which a node, *A*, will turn its radio receiver on for receiving a data packet from another node, *B*. In both cases, node *A* would start checking the channel at exactly the same time as when node *B* is transmitting the last byte of the short preamble preceding every wakeup frame. In the

best case, the radio manages to detect the Start-of-Frame Delimiter (SFD) in time. Node A is then able to receive the current wakeup frame. Therefore, the total duration that the receiver is turned on to receive the wakeup frame is calculated as:

$\mathbf{T_A} = T_{IRx}+(T_{TxWakeup}-(3*T_{Txb}))+T_{RxEnd}$ ... with ref. to Figure 3-25

In the worst case, the receiver in node A is unable to synchronise with the incoming packet from Node B and therefore unable to detect the SFD. The radio would then have to be turned on until the node receives the next wakeup packet. Therefore, the total duration for which the receiver is turned on to receive the wakeup frame is calculated as:

$\mathbf{T_B} = (T_{IRx}+((2*T_{TxWakeup})-(3*T_{Txb}))+T_{RxEnd}$ ... with ref. to Figure 3-25

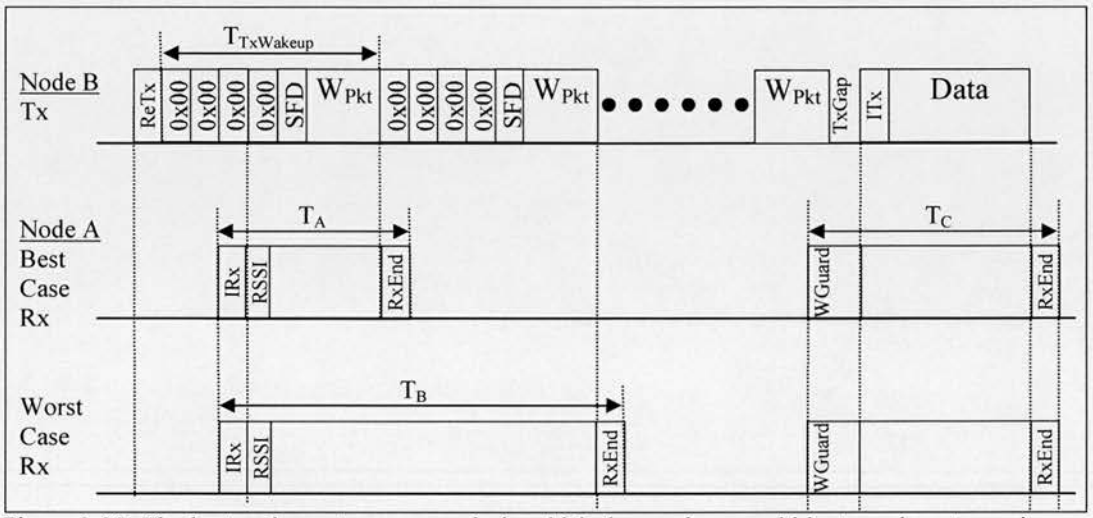


Figure 3-25: The best and worst case scenario in which the receiver would be turned on to receive a data packet using SpeckMAC-B

After the wakeup packet is received, Node A performs a backoff based on the timing information extracted from the wakeup frame. The radio reverts back into the idle state, and is woken up T_{WGuard} before the anticipated arrival of the data frame. The receiver will remain turned on until the reception of the data frame is completed. Therefore, the total duration for which the receiver is turned on to receive the data frame is calculated to be:

$\mathbf{T_C} = T_{WGuard}+T_{ITx}+T_{Tx}+T_{RxEnd}$ ... with ref. to Figure 3-25

Using the equations derived from Figure 3-25 for T_A, T_B and T_C, the total duration that the receiver is turned on to receive data packets using SpeckMAC-B in the best case scenario is calculated to be:

$$^{Best}T_{SpeckMACB\_Rx} = (T_A + T_C) * N_{Pkts} * N_{Neigh} \quad \text{... with ref. to Figure 3-25}$$

$$= (T_{IRx}+(T_{TxWakeup}-(3*T_{Txb}))+T_{RxEnd}+T_{WGuard}+T_{ITx}+T_{Tx}+ T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (28)$$

Similarly, the total duration for which the receiver is turned on to receive data packets using SpeckMAC-B in the best-case scenario is calculated to be:

$$^{Worst}T_{SpeckMACB\_Rx} = (T_B + T_C) * N_{Pkts} * N_{Neigh} \quad \text{... with ref. to Figure 3-25}$$

$$= (T_{IRx}+((2*T_{TxWakeup})-(3*T_{Txb}))+T_{RxEnd}+T_{WGuard}+T_{ITx}+T_{Tx}+T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (29)$$

Therefore, on average, the total duration that the receiver is turned on to receive data packets using SpeckMAC-B is calculated as follows:

$$^{Avg}T_{SpeckMACB\_Rx}$$

$$= (^{Worst}T_{SpeckMACB\_Rx} + ^{Best}T_{SpeckMACB\_Rx})/2$$

$$= ((T_{IRx}+(3*T_{TxWakeup})/2)-(3*T_{Txb})+T_{RxEnd}+T_{WGuard}+T_{ITx}+T_{Tx}+T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (30)$$

In order to calculate the duration for which the receiver was turned on for channel sampling, the time in which the node was 'free' to perform channel listening would have to be determined first. A node is said to be 'free' whenever it is not transmitting a packet or in the process of receiving a packet. A node is said to be in the process of receiving a packet if it had detected channel activity and was awaiting the data packet. Therefore, a node performing back-off after receiving a wakeup frame is also classified as being "in the process of receiving a packet". The total time that a node spends in this state, $T_{SpeckMACB\_Receiving}$, is calculated for the shortest, longest and average cases.

The shortest duration of $T_{SpeckMACB\_Receiving}$ occurs when a node just missed sampling the start of a transmission and will only detect the transmission when it next samples the channel, as shown in Figure 3-26. The shortest duration for $T_{SpeckMACB\_Receiving}$ is calculated to be:

$$\begin{aligned}
&{}^{\mathbf{Shortest}}\mathbf{T}_{\mathbf{SpeckMACB_Receiving}} \\
&= (T_A - T_{Interval}) * N_{Pkts} * N_{Neigh} \qquad \text{... with ref. to Figure 3-26} \\
&= (T_{IRx} + T_{RSSI} + (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + T_{TxGap} + T_{ITx} + T_{Tx} + T_{RxEnd} - T_{Interval}) \\
&\quad * N_{Pkts} * N_{Neigh}
\end{aligned} \tag{31}$$

On the other hand, the longest duration for $T_{SpeckMACB\_Receiving}$ occurs when the node samples the channel at the beginning of the transmission as shown in Figure 3-26. Therefore, the node would be in the process of receiving a packet for the following duration:

$$\begin{aligned}
&{}^{\mathbf{Longest}}\mathbf{T}_{\mathbf{SpeckMACB_Receiving}} \\
&= ((T_{IRx} + \lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + T_{TxGap} + T_{ITx} + T_{Tx} + T_{RxEnd}) * N_{Pkts} * N_{Neigh}
\end{aligned} \tag{32}$$

The average time that a node spends in "the process of receiving a packet", ${}^{Avg}T_{SpeckMACB\_Receiving}$, is calculated to be the arithmetic mean between the longest and shortest case as follows:

$$\begin{aligned}
&{}^{\mathbf{Avg}}\mathbf{T}_{\mathbf{SpeckMACB_Receiving}} \\
&= ({}^{Best}T_{SpeckMACB_Receiving} + {}^{Worst}T_{SpeckMACB_Receiving})/2 \\
&= (T_{IRx} + ((T_{RSSI} - T_{Interval})/2) + (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + T_{TxGap} + T_{ITx} + T_{Tx} \\
&\quad + T_{RxEnd}) * N_{Pkts} * N_{Neigh}
\end{aligned} \tag{33}$$

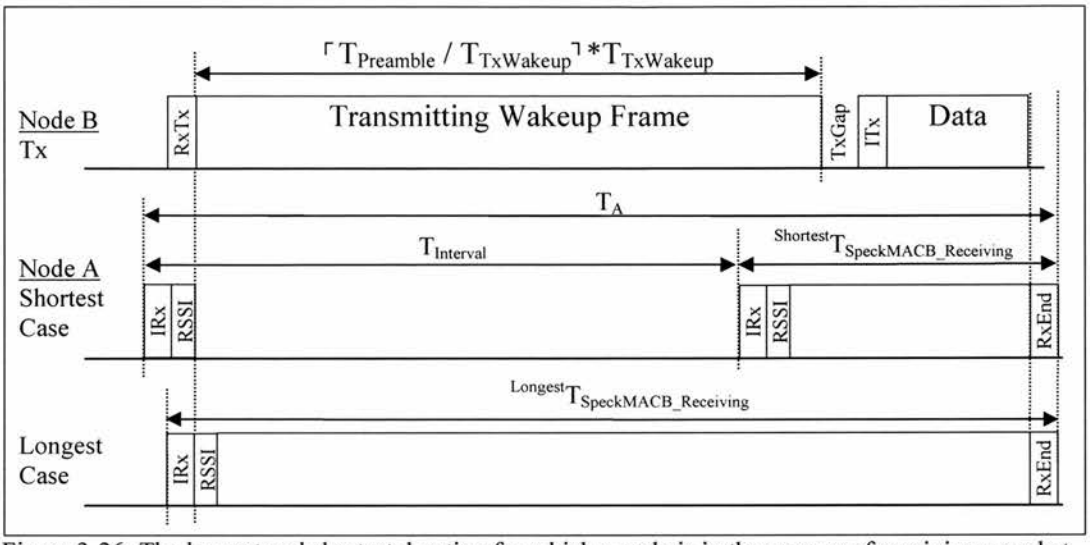
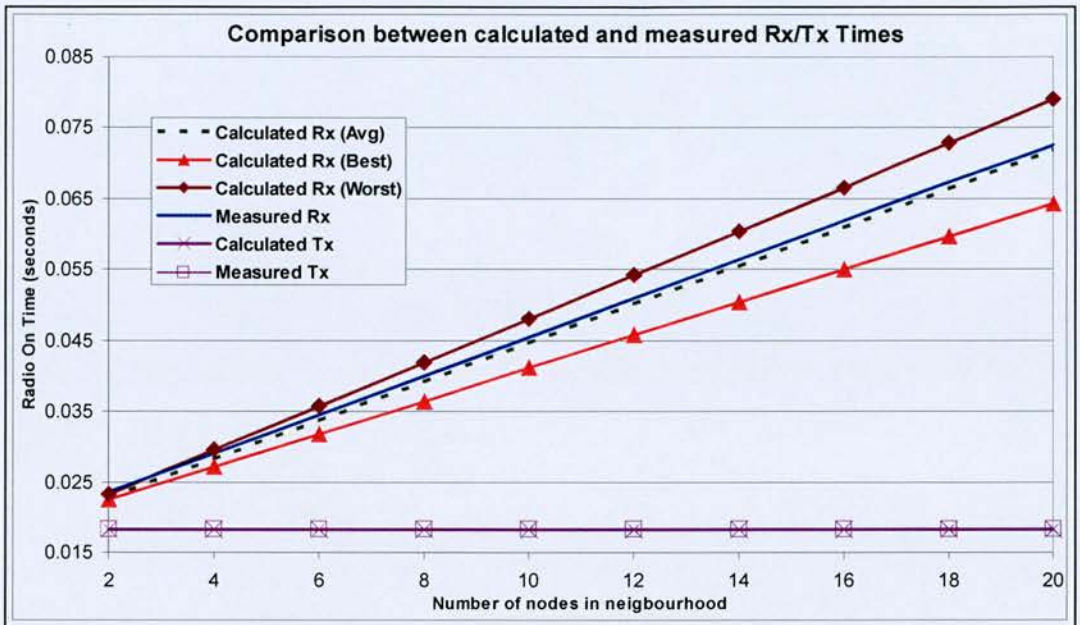


Figure 3-26: The longest and shortest duration for which a node is in the process of receiving a packet using SpeckMAC-B

The physical setup described in Section 3.6.1 was used to validate the equations for calculating the average receiver turn-on time (33) and the transmitter turn-on time (27), by comparing the measured on-times for the receiver/transmitter with the equations as shown in Figure 3-27. It is observed that equation (27) maps very closely to the measured transmitter turn-on times and was on average, across all readings, just 0.08% different from the measured values. Equation (33) also corresponds to the measured receiver turn-on times with an average deviation of about 2.09% from the measured values. Therefore, it is concluded that the equations presented in this section maps closely to the measurements of the physical implementation of the SpeckMAC-B protocol on the ProSpeckz.



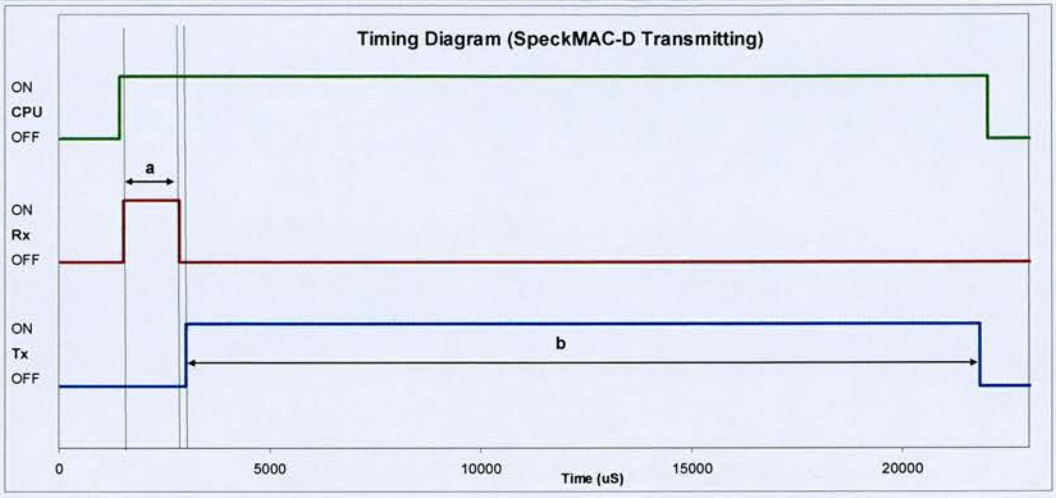
Percentage difference = [1-(Calculated/Measured)] * 100%

	2	4	6	8	10	12	14	16	18	20
Rx	3.27%	2.83%	2.58%	2.21%	2.06%	1.87%	1.69%	1.51%	1.45%	0.98%
Tx	0.08%	0.08%	0.08%	0.08%	0.08%	0.08%	0.08%	0.08%	0.08%	0.08%

Figure 3-27: Comparisons between the measured and the calculated values for the total duration for which the radio receiver and radio transmitter were turned on using SpeckMAC-B

3.6.5 Equations for modelling the radio timings for SpeckMAC-D

Figure 3-28 shows the captured timing diagram for the state of the CPU, radio transmitter and receiver measured on the ProSpeckz when transmitting a packet using SpeckMAC-D. Under SpeckMAC-D, the total duration for which the transmitter is turned on is calculated using equation (16). Unlike the implementation of B-MAC and SpeckMAC-B, equation (16) derived in Section 3.4.3 need not be modified as there was no change in the frame format during the transmission using SpeckMAC-D. Therefore, the transmission gaps present in the other two algorithms are not relevant in the case of SpeckMAC-D as the same data frame format would be used for every retransmission.



b. The receiver was turned on to perform CSMA. (T_{CSMA}=1.314ms)
c. The transmitter transmits the data frame for a minimum duration specified by $T_{Preamble}$. It sends one complete data frame once more after $T_{Preamble}$. (18.81ms)

Figure 3-28: Timing diagram measured on a ProSpeckz for data packet transmission using SpeckMAC-D

To ensure that equation (16) does map closely to the physical implementation, the result calculated using the equation was compared against the timings measured in the physical implementation as follows:

$$\mathbf{T_{SpeckMACD\_Tx}} = (T_{ITx} + ((\lceil T_{Preamble} / T_{Tx} \rceil + 1)) * T_{Tx}) * N_{Pkts} \quad \text{...from (16)}$$

$$= (131.7s + ((\lceil 16ms/1.696ms \rceil + 1) * 1.696ms$$

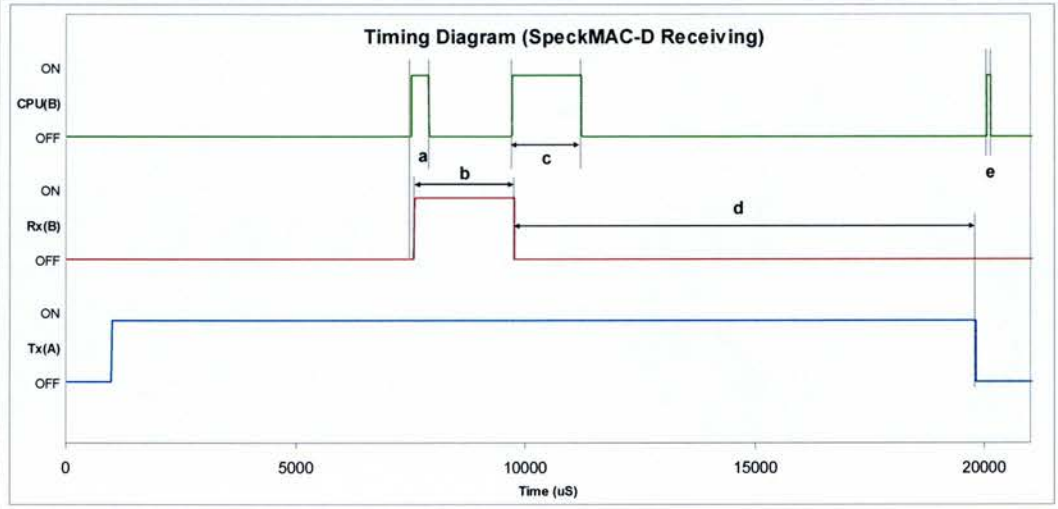
$$= 18.7877ms$$

$$\approx b = 18.81ms \text{ (diff: 22.3us, 0.12\%)} \quad \text{...from Figure 3-28}$$

The results from equation (16) matches closely to the results obtained from the physical platform and therefore, it can be concluded that equation (16) corresponds to the physical implementation.

Figure 3-29 shows the timing diagram measured for a ProSpeckz, *A*, transmitting a packet to another ProSpeckz, *B*, using SpeckMAC-D. The CPU on *ProSpeckz B* will turn on the radio receiver to perform channel sampling and detect the transmission by *A*. *B* will then keep the radio receiver on while placing the CPU back into the sleep mode, until the data frame is received. The CPU is then woken up to process the

received data frame and the backoff information is extracted, The CPU then returns to the sleep mode and periodic channel sampling is resumed after the backoff has been completed. From equation (11), the total duration for which the radio receiver was turned on to receive data packets, as shown in Figure 3-29b was based on the average-or-worse case scenario. In order to achieve a more accurate mathematical representation based on the average time that the radio receiver is turned on, the worst-case and best-case scenarios were derived using values measured from the physical implementation.



a. CPU wakes up to perform channel sampling. As the channel is busy, the radio receiver was kept on.
b. The radio is kept on until the data frame is received. (2.167ms)
c. The CPU is woken up to process the received data frame. The timing information is extracted from the data frame and a back-off timer is activated. The CPU returns to sleep and is woken up when the back-off timer expires.
d. The radio receiver is in the idle state during the back-off period and the node will be permitted to transmit. All outgoing packets will be buffered.
e. The back-off timer expires and the CPU is woken up to reset the timer for channel sampling. If there are buffered packets to be sent, the node will contend for the channel, else, periodic channel sampling will be resumed.

Figure 3-29: Timing diagram captured from *ProSpeckz A* and *ProSpeckz B* when *B* was receiving a data packet from *A* using SpeckMAC-D

Figure 3-30 shows the possible best- and worst-case scenarios in which a node, A, turns on its radio receiver for receiving a data packet from node B. In both cases, node A starts checking the channel at exactly the same time as when node B was transmitting the last byte of the short preamble preceding every transmitted data packet. In the best case, the radio manages to perform synchronisation in time to detect the SFD, in which case, Node A is able to receive the current data packet. The duration that the receiver is turned on for the reception of data transmission, in the best case scenario, is calculated to be:

$$^{Best}\mathbf{T}_{\mathbf{SpeckMACD\_Rx}} = (T_{IRx}+(T_{Tx}-(3*T_{Txb}))+T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (34)$$

In the worst-case scenario, the receiver of node A is unable to synchronise with the incoming packet from Node B using the last byte of the received preamble and is therefore unable to detect the SFD. The receiver continues to leave the receiver turned on until the node receives the next data packet. Therefore, in the worst-case scenario, the equation for calculating the duration for which the receiver is turned on for the reception of data is:

$$^{Worst}\mathbf{T}_{\mathbf{SpeckMACD\_Rx}} = (T_{IRx}+((2*T_{Tx})-(3*T_{Txb}))+T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (35)$$

The average value, $T_{SpeckMACD\_Rx}$, for which the receiver is turned on for receiving the data frames, is calculated as the arithmetic mean between the best- and worst-case scenarios and is calculated as:

$$\mathbf{T}_{\mathbf{SpeckMACD\_Rx}} = (^{Worst}T_{SpeckMACD\_Rx} + {}^{Best}T_{SpeckMACD\_Rx})/2$$

$$= (T_{IRx}+((3*T_{Tx})/2)-(3*T_{Txb})+T_{RxEnd})*N_{Pkts}*N_{Neigh} \quad (36)$$

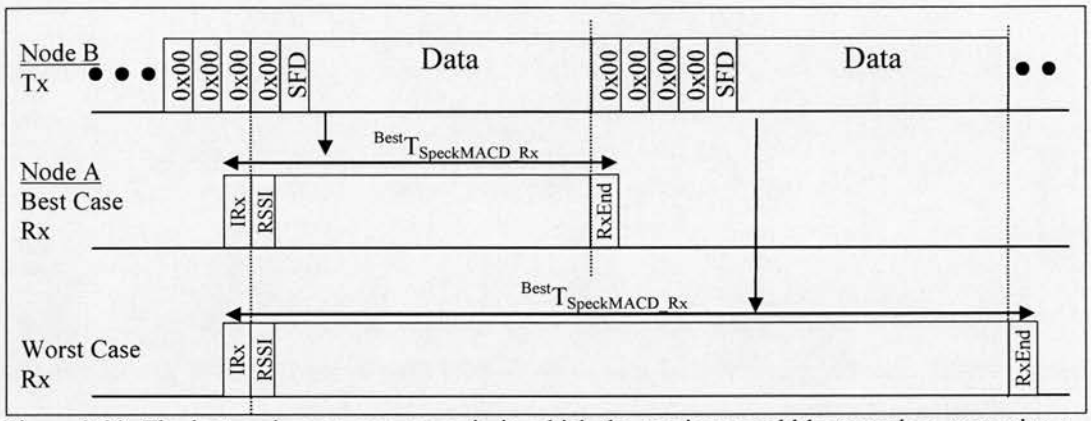


Figure 3-30: The best and worst case scenario in which the receiver would be turned on to receive a data packet using SpeckMAC-D

Just as in the case of SpeckMAC-B, in order to calculate the duration for which the receiver is turned on for channel sampling, the time for which the node is 'free' to perform channel listening has to be determined first. The time spent on receiving a data packet, $T_{SpeckMACD\_Receiving}$ must then be calculated. The longest time for which a node is in the process of receiving a packet, $^{Longest}\mathbf{T_{SpeckMACD\_Receiving}}$, occurs when the node samples the channel at the beginning of the transmission (Figure 3-31) and is calculated as:

$$^{Longest}\mathbf{T_{SpeckMACD\_Receiving}} = ((T_{IRx} + \lceil T_{Preamble}/T_{Tx} \rceil * T_{Tx}) + T_{RxEnd}) * N_{Pkts} * N_{Neigh} \quad (37)$$

On the other hand, the shortest case occurs when the node just missed sampling the start of a transmission as shown in Figure 3-31. It will then sample the channel at $(T_{Interval} - T_{Sample})$ after it has finished its current sample. Consequently, the shortest case for $T_{SpeckMACB\_Receiving}$ is calculated as:

$$^{Best}\mathbf{T_{SpeckMACD\_Receiving}}$$

$$= (((\lceil T_{Preamble} / T_{Tx} \rceil * T_{Tx}) - (T_{Interval} - T_{Sample})) + T_{RxEnd}) * N_{Pkts} * N_{Neigh} \quad (38)$$

The average time that a node would spend in the process of receiving a packet, $T_{SpeckMACB\_Receiving}$, is calculated as the average between the shortest and the longest cases and is derived as:

$$\mathbf{T_{SpeckMACB\_Receiving}}$$

$$= ({}^{Best}T_{SpeckMACB\_Receiving} + {}^{Worst}T_{SpeckMACB\_Receiving})/2$$

$$= (T_{IRx} + ((T_{RSSI} - T_{Interval})/2) + (\lceil T_{Preamble} / T_{Tx} \rceil * T_{Tx}) + T_{RxEnd}) * N_{Pkts} * N_{Neigh} \quad (39)$$

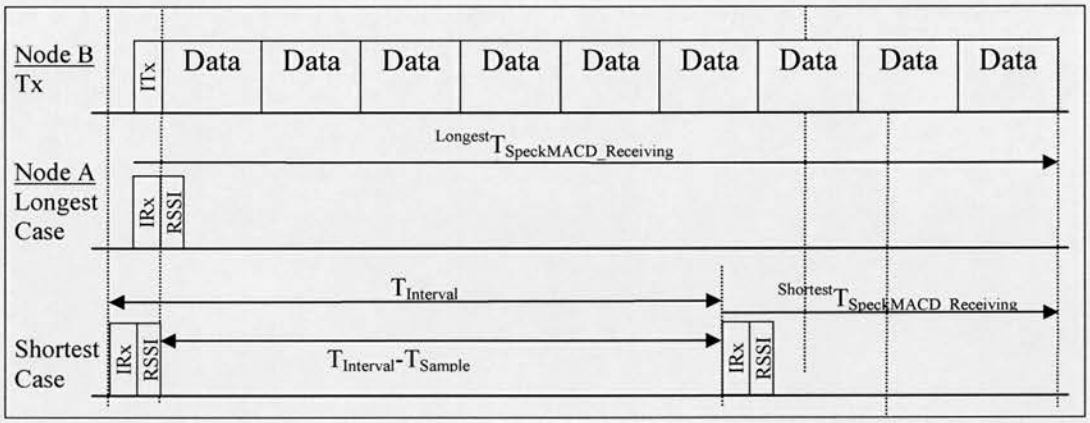
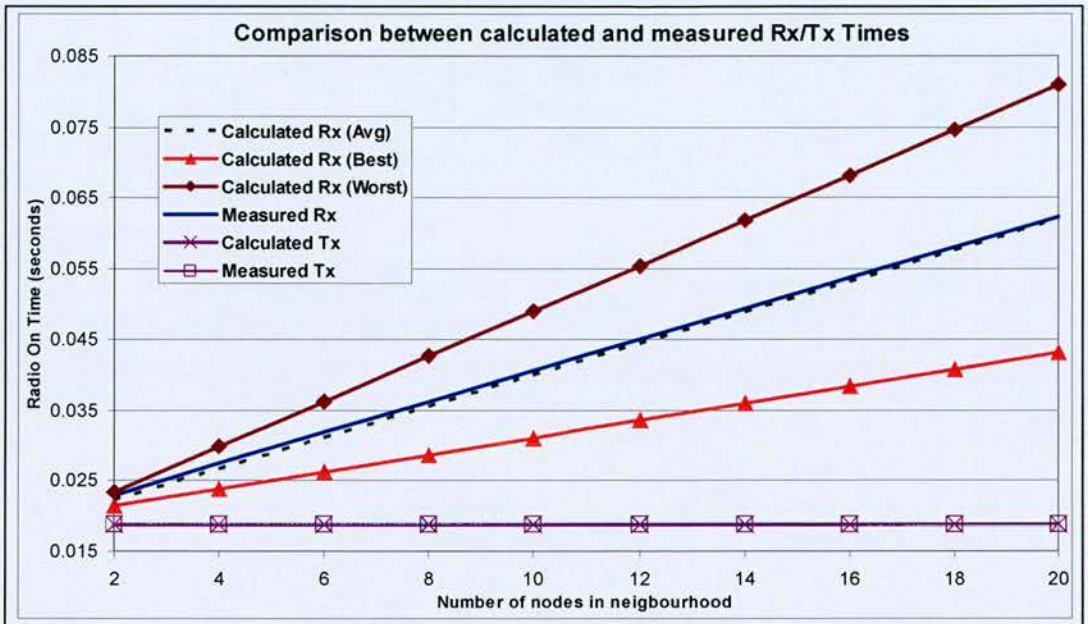


Figure 3-31: The longest and shortest duration in which a node would be in the process of receiving a packet using SpeckMAC-D

The physical setup described in Section 3.6.1 was used to validate the equations for calculating the average turn-on time for the receiver (39) and the transmitter (16), by comparing the measured on-times for the receiver/transmitter with the equations, as shown in Figure 3-32. It is observed that equation (16) maps very closely to the measured transmitter turn-on times and was on average, across all readings, just 0.11% different from the measured values. Furthermore, equation (39) also maps closely to the measured receiver turn-on times with an average deviation of about 1.74% from the measured values. Therefore, it is concluded that the equations presented in this section map closely to the measurements of the physical implementation of the SpeckMAC-D protocol on the ProSpeckz.



Percentage difference = [1-(Calculated/Measured)] * 100%

Nodes	2	4	6	8	10	12	14	16	18	20
Rx	2.91%	2.67%	2.52%	1.89%	1.71%	1.64%	1.32%	1.17%	0.97%	0.61%
Tx	0.11%	0.11%	0.11%	0.11%	0.11%	0.11%	0.11%	0.11%	0.11%	0.11%

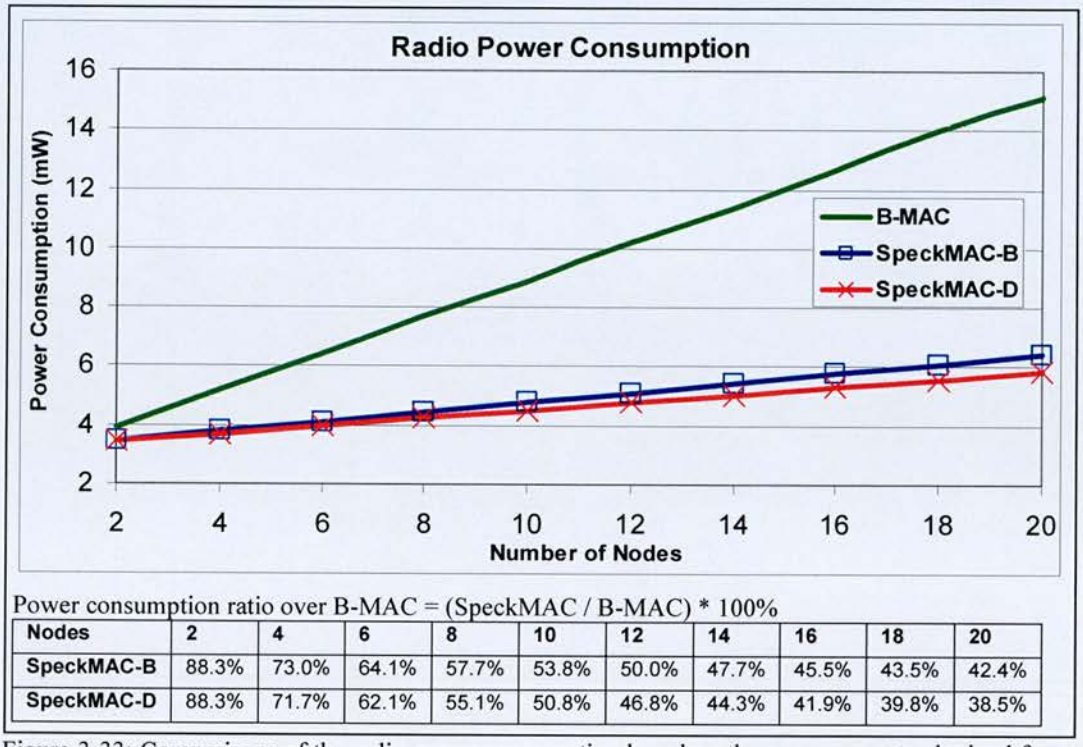
Figure 3-32: Comparison between the measured and the calculated values for the total duration for which the radio receiver and radio transmitter were turned on using SpeckMAC-D

3.7 Analysis of the performances of B-MAC, SpeckMAC-B and SpeckMAC-D

3.7.1 Power consumption for broadcast traffic

In broadcasting, a data packet is transmitted by a node which is meant to be received by all its neighbours. Broadcasting is employed when information or data is required to be disseminated to a large sub-set or all of the nodes in the network, e.g., the flooding of route request packets during route discovery, or the dissemination of sensor information. Figure 3-33 shows comparisons of the power consumption of the radio for B-MAC, SpeckMAC-B and SpeckMAC-D based on actual timing measurements obtained from the experiments outlined in Section 3.6. The timings were multiplied with the power consumption for the radio in the receive mode (62.1mW), the transmit mode (57.4mW) and the idle mode (0.43mW) to obtain the averaged radio power consumption. From the results, it was observed that both

SpeckMAC algorithms consumed significantly less energy than B-MAC, and that the savings in energy consumption increased as the number of nodes was increased. This increase in energy savings was due to the fact that as the number of nodes increases, the number of packets received by each node also rises; therefore, the energy wasted by B-MAC for the reception of redundant preamble bytes would also increase. Furthermore, for a small payload of 32 bytes, it was observed that SpeckMAC-D consumed less energy than SpeckMAC-B.



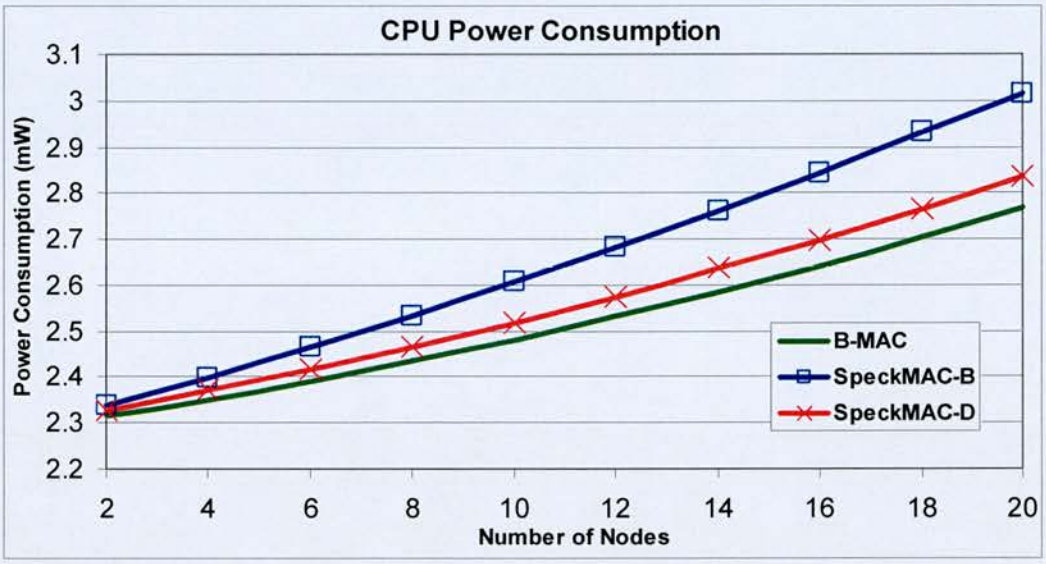
Power consumption ratio over B-MAC = (SpeckMAC / B-MAC) * 100%

Nodes	2	4	6	8	10	12	14	16	18	20
SpeckMAC-B	88.3%	73.0%	64.1%	57.7%	53.8%	50.0%	47.7%	45.5%	43.5%	42.4%
SpeckMAC-D	88.3%	71.7%	62.1%	55.1%	50.8%	46.8%	44.3%	41.9%	39.8%	38.5%

Figure 3-33: Comparisons of the radio power consumption based on the measurements obtained from the physical implementation of the three MAC algorithms

In addition to the timing information of the radio, another measurement obtained during the physical execution of the algorithms was the duration for which the CPU was active. Measurements on the ProSpeckz had revealed that the CPU consumes 1.56mW in the sleep state and 16.5mW in the active state. Based on these measurements, the power consumption of the CPU for the three MAC algorithms was calculated. Figure 3-34 shows that the power consumption of the CPU utilising SpeckMAC-B and SpeckMAC-D was greater than B-MAC. In the case of SpeckMAC-D, the extra CPU processing time was due to resetting the timer for

periodic channel sampling after a packet was received as shown in Figure 3-29e. In the case of SpeckMAC-B, extra CPU processing time is needed to process the wakeup frame as well as to receive the data frame as shown in Figure 3-24c and Figure 3-24e respectively.

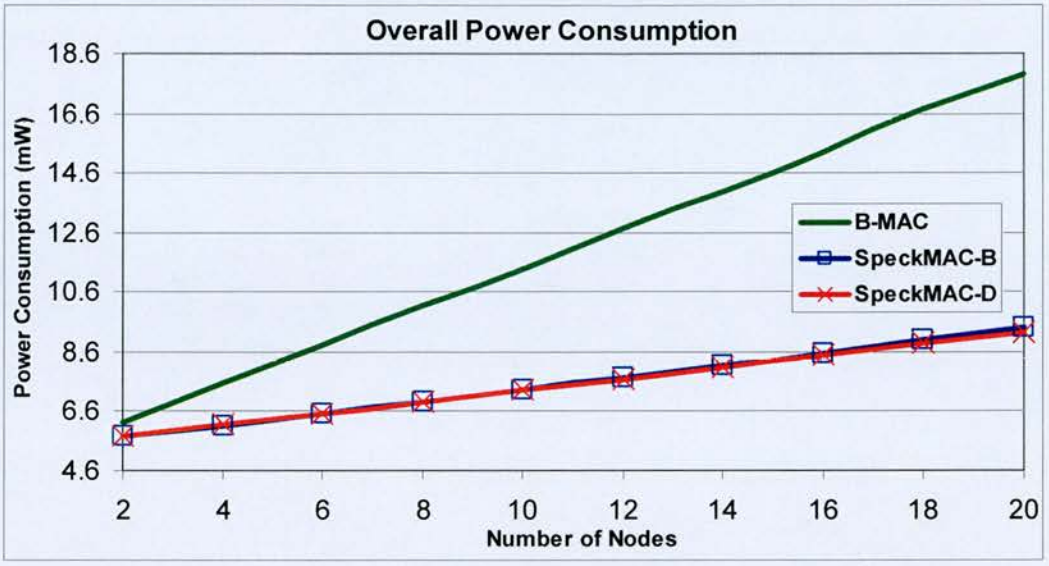


Power consumption ratio over B-MAC = (SpeckMAC / B-MAC) * 100%

Nodes	2	4	6	8	10	12	14	16	18	20
SpeckMAC-B	101%	102%	103%	104%	105%	106%	107%	108%	108%	109%
SpeckMAC-D	101%	101%	101%	101%	101%	102%	102%	102%	102%	102%

Figure 3-34: Comparisons of the CPU power consumption based on the measurements obtained from the physical implementation of the three algorithms

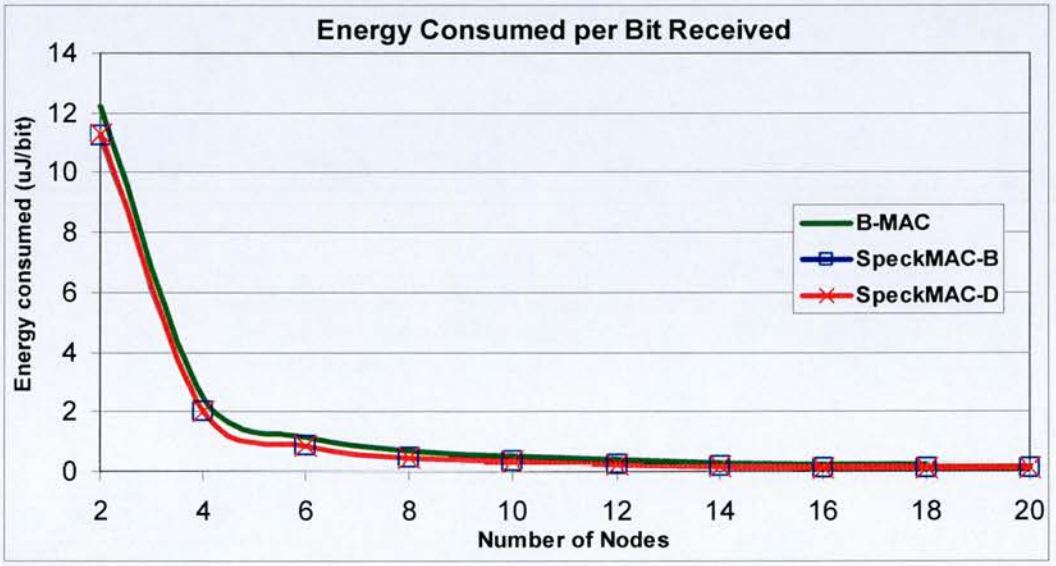
Given the energy savings achieved by both SpeckMAC protocols in the radio outweighs the additional energy consumed by the CPU when compared to B-MAC, both SpeckMAC protocols demonstrated lower overall power consumption (radio + CPU) over B-MAC. In Figure 3-35, with just two nodes, SpeckMAC-B and SpeckMAC-D consumed about 92% of the energy consumed by B-MAC. As the number of nodes is scaled, the savings becomes more significant such that with 20 nodes participating, SpeckMAC-B and SpeckMAC-D used less than 53% of power used by B-MAC. A similar improvement was also observed when comparing the energy consumed per bit successful received between the three MAC algorithms as shown in Figure 3-36.



Power consumption ratio over B-MAC = (SpeckMAC / B-MAC) * 100%

Nodes	2	4	6	8	10	12	14	16	18	20
SpeckMAC-B	92.3%	81.4%	74.1%	68.3%	64.5%	60.7%	58.3%	55.9%	53.7%	52.4%
SpeckMAC-D	92.9%	81.7%	74.2%	68.1%	64.2%	60.3%	57.8%	55.3%	53.0%	51.7%

Figure 3-35: Comparisons of the overall power consumption based on the measurements obtained from the physical implementation of the three algorithms



Energy consumption ratio over B-MAC = (SpeckMAC / B-MAC) * 100%

Nodes	2	4	6	8	10	12	14	16	18
SpeckMAC-B	91.87%	80.94%	73.68%	68.26%	64.24%	60.50%	58.26%	55.74%	53.63%
SpeckMAC-D	92.29%	81.14%	73.49%	67.83%	63.81%	59.86%	57.32%	54.84%	52.55%

Figure 3-36: Comparisons of the energy consumed per bit received based on the measurements obtained from the physical implementation of the three algorithms

3.7.2 Delivery ratio

Figure 3-37 shows the delivery ratio for packets employing the three protocols, based on the experiments in Section 3.6. SpeckMAC-D achieves a consistently higher delivery ratio as the number of nodes scaled when compared to the other two protocols as each data payload is transmitted several times; the inbuilt redundancy, absent from the other two protocols, enable the data to be received when one or more data frames are corrupted by external noise interferences. The example in Figure 3-38 demonstrates this and shows three nodes performing channel sampling at different times, with the occurrence of channel interference that corrupts the data. Both B-MAC and SpeckMAC-B fails to delivery the packet due to corruption in the transmitted data frame. However, under SpeckMAC-D, nodes were able to receive a copy of the data packet, with Node C being the only node affected by the interference.

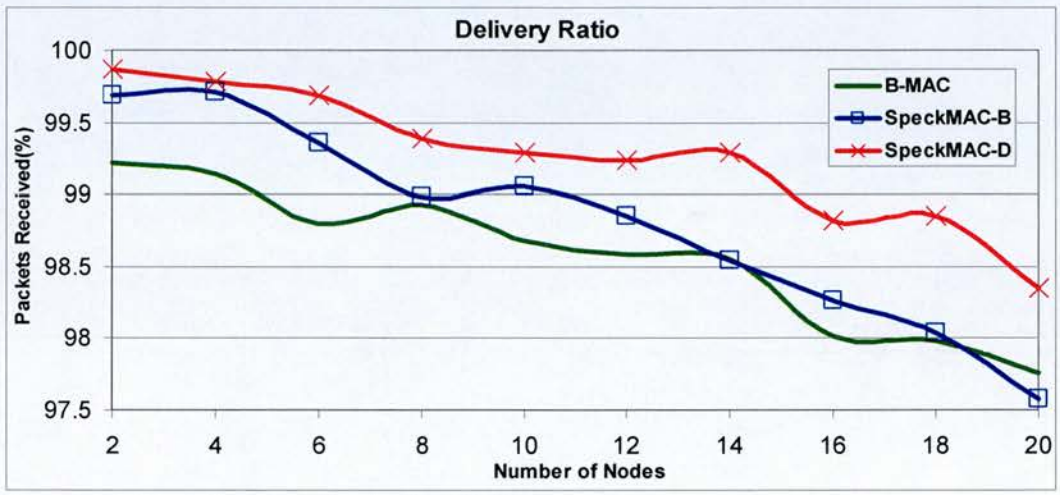


Figure 3-37: The delivery ratio for the three MAC protocols

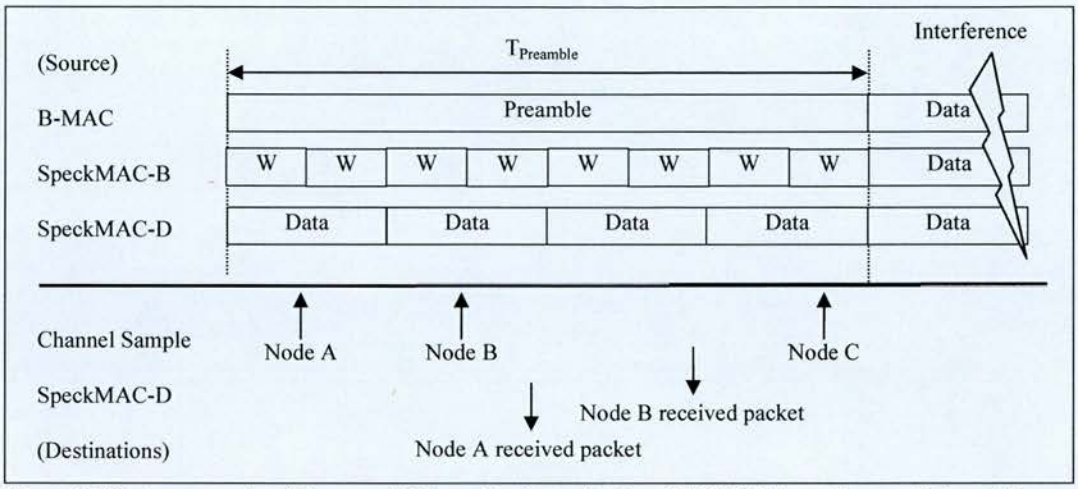


Figure 3-38: An example of the use of data redundancy by SpeckMAC-D to achieve a higher delivery ratio compared to B-MAC and SpeckMAC-B

3.7.3 Power consumption for unicast traffic

Unicast communication occurs when a source node has data to be sent to a pre-determined destination node. In a Specknet, unicast communications is used in situations where a packet has to traverse a pre-determined route, or when a sensor reading from one node is required to be sent to a specific destination node for actuation. In such cases where a data packet was meant for a specific node, energy wastage due to over-hearing becomes a significant factor. Over-hearing occurs when a node receives a packet that was not intended for itself. For power-aware MAC algorithms, frequent over-hearing could result in a significant loss of energy as the node has to wake up from its sleep state, receive and process the packet, after which, determine that the packet is not intended for itself and discard it. An example of overhearing is shown in Figure 3-39. Using B-MAC as an example of the operating MAC algorithm, when node A sends a packet to node B, then node C will also be woken up as it detects activity on the channel while performing channel listening.

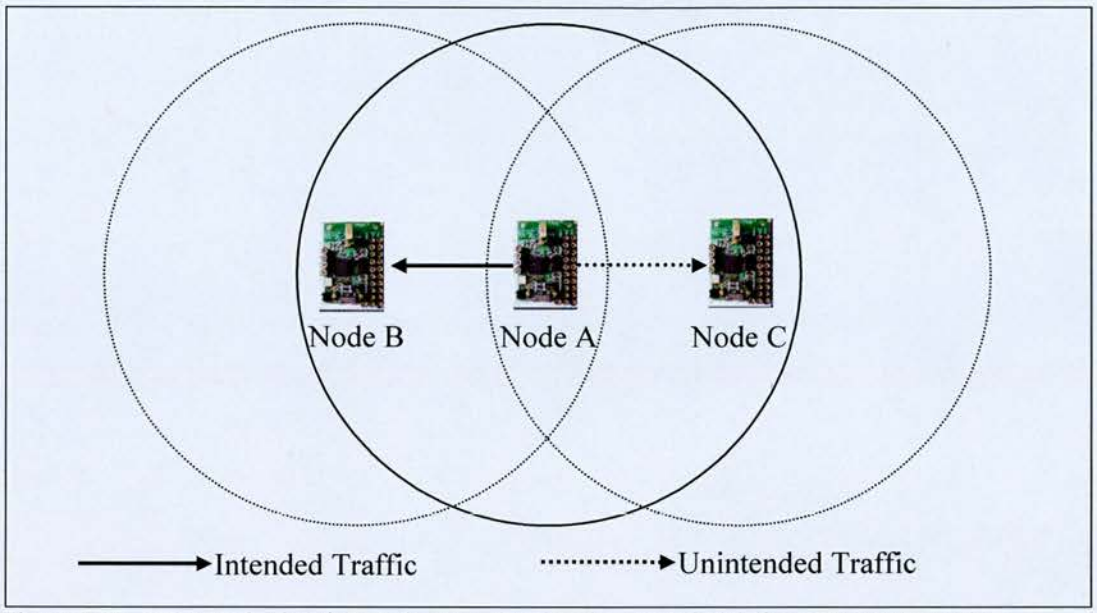


Figure 3-39: An example showing Node C overhearing the transmission from Node A to Node B

Random access based MAC algorithms suffer from overhearing because schedules are not maintained at the MAC layer to determine whether the node was anticipating data. Therefore, it is not possible for the nodes to determine if it should turn on its radio based on an incoming data packet or wakeup signal that is yet to be received. Scheduling protocols could be incorporated into the higher layers, such as the networking or the application layer. However, this was not considered here, as the emphasis is to compare the energy wastage between the three MAC algorithms at the link-level.

In all three cases, a node always wakes up when it senses activity during periodic listening. In both B-MAC and SpeckMAC-D, the node can only determine whether it is the destination for the data packet after it has received the packet header containing the destination address. In the case of the ProSpeckz, this can only be done after receiving the entire data packet. Therefore, in both the cases of B-MAC and SpeckMAC-D, there should be no difference in the reception of a unicast or a broadcast packet as both cases, the radio will have to be turned on, and can only be turned off after the data packet is received. SpeckMAC-B, on the other hand, consumes less power due to the use of the wakeup packet, which enables nodes to determine if it has to backoff and wakeup to receive the data payload at a later time.

This is done by checking the destination address of the wakeup packet. As long as the length of the data packet is longer than the length of the wakeup packet, then SpeckMAC-B will consume less power than SpeckMAC-D in a unicast situation.

To demonstrate the effects of overhearing and the power consumption in the three MAC algorithms, an experiment was conducted to emulate an environment in which ten nodes, all within radio range of one another, were required to transfer 32 bytes of sensor readings to a common destination node once every second. For each of the MAC algorithms, the experiment was executed across the value of $T_{Interval}$ ranging from 10ms to 30ms, in steps of 5ms. The experiment was performed on the PerSpeckz-64 with 10 nodes randomly chosen during each iteration. For each $T_{Interval}$ selected, 40 iterations were performed and the results averaged. The process was repeated, however, broadcast packets were sent in this instance. The total time for which the radio receiver node was turned on is shown in Figure 3-40. From the results, in the cases of B-MAC and SpeckMAC-D, the receiver was turned on for exactly the same duration for both broadcast and unicast packets as anticipated. However, as the duration for data reception is independent of $T_{Interval}$, nodes utilising SpeckMAC-D did not turn on their receivers any longer as $T_{Interval}$ was increased. On the other hand, the cost of overhearing using B-MAC is largely dependent upon chosen $T_{Interval}$ as there was no useful information in the preamble, therefore a node could make a decision only after it has received the data portion of the packet. Unlike B-MAC and SpeckMAC-D, SpeckMAC-B allows the receiver to determine if it has to turn on its receiver for the data packet based on the information in the wakeup packet. In Figure 3-40, for nodes sending broadcast packets, SpeckMAC-B consumed more power than SpeckMAC-D as its receiver was turned on for a longer duration. However, when unicast packets were sent, nodes employing SpeckMAC-B turned their receiver less frequently, as these nodes were able to distinguish if one of them using the wakeup frames.

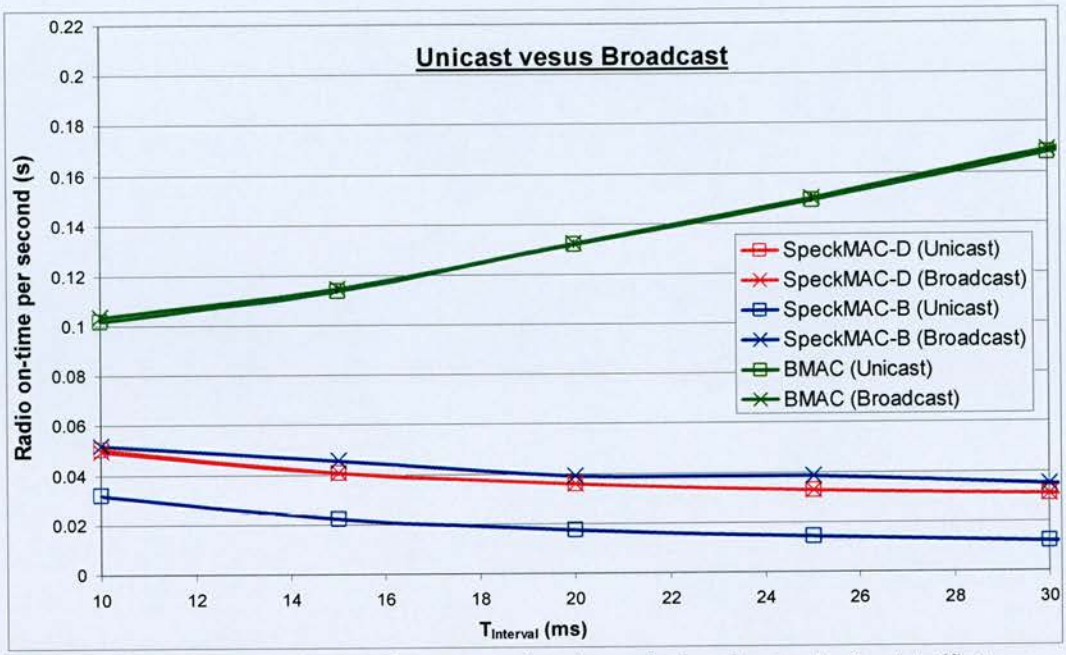


Figure 3-40: Comparisons of the receiver turn-on duration under broadcast and unicast traffic types

3.7.4 Energy wastages due to channel jamming

As the B-MAC and SpeckMAC protocols operate on channel listening, the assumption is that a busy channel signifies that some source nodes have data to send. However, this might not always be true as other activities on the channel, such as interferences and noise, could cause nodes using the protocols to assume a busy channel. In order to enable the nodes to overcome sporadic interferences, a timeout was used so that a node turns off the radio receiver and return to periodic channel sampling if the data frame was not received within a certain time period after the receiver was turned on. Therefore, a node with shorter timeout duration will waste less energy when the channel is jammed. However, the timeout duration must also be long enough to ensure that the MAC protocols can still operate correctly. Therefore, in the worst-case scenario, the timeout duration for the three algorithms must be longer than the time it takes a node to receive a data packet. The worst-case would occur when a node samples the channel at the beginning of the transmitted preamble.

In the case of B-MAC, the timeout duration, $T_{BMAC\_Timeout}$, must satisfy the following condition:

$$\mathbf{T_{BMAC\_Timeout}} > (T_{IRx}+T_{Preamble}+T_{TxGap}+T_{ITx}+T_{Tx}+T_{RxEnd}) = (T_{Preamble} + T_{Tx} + 631us) \quad (40)$$

Likewise, the timeout duration for SpeckMAC-B, $T_{SpeckMACB\_Timeout}$, must satisfy the following in order to ensure that the receiver was turned on for a sufficient duration of time in order to receive a data frame:

$$\mathbf{T_{SpeckMACB\_Timeout}} > (T_{WGuard} + T_{ITx} + T_{Tx} + T_{RxEnd}) = 690us + T_{Tx} \quad (41)$$

Furthermore, the timeout value for SpeckMAC-B must be longer than the duration needed to receive the wakeup frames in the worst case. The worst case would occur when the node samples the channel at the start of a wakeup frame. It would then have to keep the receiver on until the next wakeup frame is received completely. Therefore, $T_{SpeckMACB\_Timeout}$ must also satisfy the following condition:

$$\mathbf{T_{SpeckMACB\_Timeout}} > T_{ITx}+(2*T_{TxWakeup}) + T_{RxEnd} = 1215us \quad (42)$$

Since the minimum number of bytes transmitted per packet, including the preamble, is 19 bytes for SpeckMAC-B (therefore, $T_{Tx} \geq 19 * T_{Txb} = 608us$), equation (42) will always be true if equation (41) was satisfied based on the implementation on the ProSpeckz.

Fianlly, the worst-case for receiving a packet using SpeckMAC-D would occur when a node samples the channel at the start of a transmitted data frame. Therefore, the receiver has to be turned on until after the next data frame has been received. The

timeout duration, $T_{SpeckMACD\_Timeout}$, for SpeckMAC-D must satisfy the following condition:

$$\mathbf{T_{SpeckMACD\_Timeout}} > T_{ITx} + (2*T_{Tx}) + T_{RxEnd}$$
$$= 255us + (2*T_{Tx}) \quad (43)$$

Given that the minimum number of bytes transmitted per packet, including the preamble, is 21 bytes for SpeckMAC-D (therefore $T_{Tx} \geq 21 * T_{Txb} = 672us$) and assuming that $T_{Preamble} > (2*T_{Tx})$, the following inequalities will always be true for the ProSpeckz implementations:

$$\mathbf{T_{BMAC\_Timeout} > T_{SpeckMACD\_Timeout} > T_{SpeckMACB\_Timeout}}$$

A small timeout value would exert a significant effect on the energy wasted in the situations when the channel is jammed by some external devices or interferences. To demonstrate the importance of a small timeout value, an experiment was carried out using the ProSpeckz. Using a value of 15ms for $T_{Interval}$ and a 32-byte data payload, equations (40), (41) and (43) presented the following minimum timeout durations;

$\mathbf{T_{BMAC\_Timeout}} > 18.263ms$ (rounded up to 19ms)

$\mathbf{T_{SpeckMACB\_Timeout}} > 2.322ms$ (rounded up to 3ms)

$\mathbf{T_{SpeckMACD\_Timeout}} > 3.647ms$ (rounded up to 4ms)

Using the rounded up values, two scenarios were tested: one with the transmission channel being jammed by a ProSpeckz consistently transmitting values from a CRC lookup table, and the other without. Each scenario was executed for 30 iterations, each lasting 60 seconds, and statistics for the radio receiver was recorded between the 10^{th} and the 50^{th} second as shown in Figure 3-41. When free from interference and the channel was clear, nodes using any of the protocols turned the receiver on for 2.029% of the time. However, when the channel was jammed, the ProSpeckz utilising B-MAC turned its radio on for more than 56% of the time. This was significantly longer than ProSpeckz utilising SpeckMAC-B or SpeckMAC-D, which turned on their radios for about 19% and 23% of the time, respectively. SpeckMAC-

B was also able to turn on its receiver for a shorter duration than SpeckMAC-D when the channel was jammed, as a shorter timeout value was used for SpeckMAC-B.

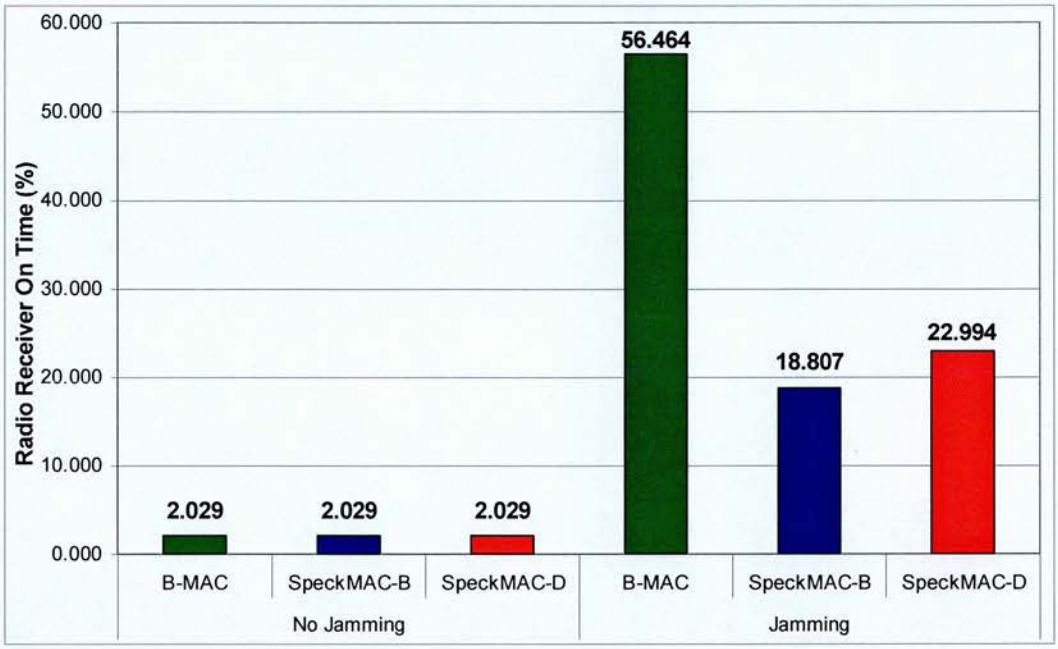


Figure 3-41: Impact of jamming on the receiver turn-on duration for the different MAC protocols

3.7.5 One-hop latency

The one-hop latency is defined as the time between a packet being received into the receive buffer at the destination node after it was placed into the transmission buffer at the source node. Therefore, a simple generic equation for calculating the one-hop latency between any two wireless nodes, A and B, can be summarised as:

$$\mathbf{T_{Latency}} = A_{ProcessPacket} + A_{Transmit} + T_{Propagation} + B_{ProcessPacket} \quad (44)$$

where, $A_{ProcessPacket}$ is the time taken by the MAC layer to format the packet in the transmit buffer and the time for the packet to be transferred to the radio's buffer (in the case of a dual sub-system setup like the ProSpeckz). $A_{Transmit}$ is the time taken to transmit the packet and $T_{Propagataion}$ is the time for the first transmitted signal from A to reach the receiver node B. $B_{ProcessPacket}$ is the time taken for the receiving node, B, to transfer the packet from the radio buffer, extract the data payload from the packet and place the data payload into the receive buffer so that it could be accessed by the upper layers.

Using the generic equation (44), an equation for calculating the average one-hop latency for B-MAC, L_{BMAC}, is derived as follows:

$$\mathbf{L_{BMAC}} = T_{BMAC\_TXProcess} + T_{CSMA} + T_{RxTx} + T_{Preamble} + T_{Tx} + T_{Propagation} + T_{BMAC\_RXProcess} \quad (45)$$

At the source node, when an application intends to send a packet, $T_{BMAC\_TXProcess}$ is the duration taken for the processing and formatting of the packet, as well as transferring the data bytes into the radio transmit buffer. T_{CMSA} is the time required for assessing that the channel is clear, after which, T_{RxTx} is the time to turn the radio from the receiver mode into the transmitter mode. A long preamble with duration of $T_{Preamble}$ would then be sent, followed by the data packet. The transmitted signal propagates to the destination receiver after a duration of $T_{Propagation}$, and on receiving the whole packet, the destination node checks the packet for errors (CRC checks),

processes it (buffering, formatting) and stores the packet in the receive buffer. Similarly, the average latency for SpeckMAC-B, $L_{SpeckMACB}$, is derived as follows:

$$\mathbf{L_{SpeckMACB}} = T_{SpeckMACB\_TXProcess} + T_{CSMA} + T_{RxTx} + (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + T_{Tx} + T_{Propagation} + T_{SpeckMACB\_RXProcess} \quad (46)$$

where, $T_{SpeckMACB\_TXProcess}$ and $T_{SpeckMACB\_RXProcess}$ are the times needed to process and format the transmitted and received packets respectively.

The latency calculations for SpeckMAC-D differs from B-MAC and SpeckMAC-B, as unlike these two algorithms, nodes utilising SpeckMAC-D need not wait until the end of transmission to receive the data frame. Assuming that on average, receiving nodes would sample the channel at a time, $T_{Interval}/2$, after the start of a packet transmission, the the average latency for SpeckMAC-D, $L_{SpeckMACD}$, is calculated as:

$$\mathbf{L_{SpeckMACD}} = T_{SpeckMACD\_TXProcess} + T_{CSMA} + T_{RxTx} + T_{Interval}/2 + T_{Tx} + T_{Propagation} + T_{SpeckMACD\_RXProcess} \quad (47)$$

Having derived the equations for estimating the latency for the three algorithms, the equations were further simplified by assuming that $T_{SpeckMACB\_TXProcess}$, $T_{SpeckMACD\_TXProcess}$, $T_{BMAC\_TXProcess}$, $T_{SpeckMACB\_RXProcess}$, $T_{SpeckMACD\_RXProcess}$, $T_{BMAC\_RXProcess}$, T_{CSMA}, T_{RxTx}, T_{Tx} and $T_{Propagation}$ as constants. The equations simplifies to:

$$\mathbf{L_{BMAC}} = T_{Preamble} + C_{BMAC}$$

$$\mathbf{L_{SpeckMACB}} = (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + C_{SpeckMACB}$$

$$\mathbf{L_{SpeckMACD}} = (T_{Interval}/2) + C_{SpeckMACD} \quad (48)$$

where, C_{BMAC}, $C_{SpeckMACB}$ and $C_{SpeckMACD}$ are constants for B-MAC, SpeckMAC-B and SpeckMAC-D, respectively.

From these simplified equations (48), it can be seen that latency should increase with the rise in $T_{Interval}$, given a fixed data packet length and the assumption that no collisions occurred. Furthermore, since $T_{Preamble} <= (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup})$, and $T_{Preamble} > T_{Interval}$, provided that the constants for the equations are equal, it could be assumed that:

$$\mathbf{L_{SpeckMACB}} \geq \mathbf{L_{BMAC}} > \mathbf{L_{SpeckMAC\text{-}D}} \qquad (49)$$

To demonstrate the latency assumption (49), an experiment was carried out using two ProSpeckz. The source ProSpeckz toggles the state of an I/O pin when a packet was placed into the transmission buffer, and the destination ProSpeckz would toggle the state of an I/O pin when the same packet was received and placed into the receive buffer. The I/O pins of the two ProSpeckz were monitored and the timing interval between the two signals was recorded to measure the latency. The packet format used for the experiment is shown in Figure 3-15 with each packet containing a data payload of 32 bytes. For each algorithm, values of $T_{Interval}$ ranging from 15ms to 85ms (in intervals of 10ms) were used and for each scenario, 200 iterations were performed. The measurements from the experiment were averaged across the iterations for each scenario and the result is shown in Figure 3-42. The following linear equations were derived by mapping the results:

$$\mathbf{L_{BMAC}} = (0.9993 * T_{Interval}) + 0.0078$$

$$\mathbf{L_{SpeckMACB}} = (1.0005 * T_{Interval}) + 0.0083$$

$$\mathbf{L_{SpeckMACD}} = (0.5251 * T_{Interval}) + 0.0064 \qquad (50)$$

Equations (50) demonstrated that for both B-MAC and SpeckMAC-B, an increase in the latency was directly proportional to the rise in $T_{Interval}$, as expected. Furthermore, it can be seen that B-MAC will have a slightly shorter latency than SpeckMAC-B. This corresponded to the previous statement that $T_{Preamble} \leq (\lceil T_{Preamble} / T_{TxWakeup} \rceil *$

$T_{TxWakeup}$), and the results support the assumption (49) that $L_{BMAC} \leq L_{SpeckMACB}$. On the other hand, the one-hop latency measured when SpeckMAC-D was utilised increased with a constant of proportionality 0.5. This was because unlike B-MAC and SpeckMAC-B, which were directly proportional to $T_{Interval}$, nodes utilising SpeckMAC-D were able to receive the data packet at any time during the preamble interval. Therefore, the results observed from the mapped equations shown in (50) also supported the assumption (49) that $L_{SpeckMACD} < L_{BMAC}$. In conclusion, SpeckMAC-D outperformed SpeckMAC-B and B-MAC in terms of one-hop latency.

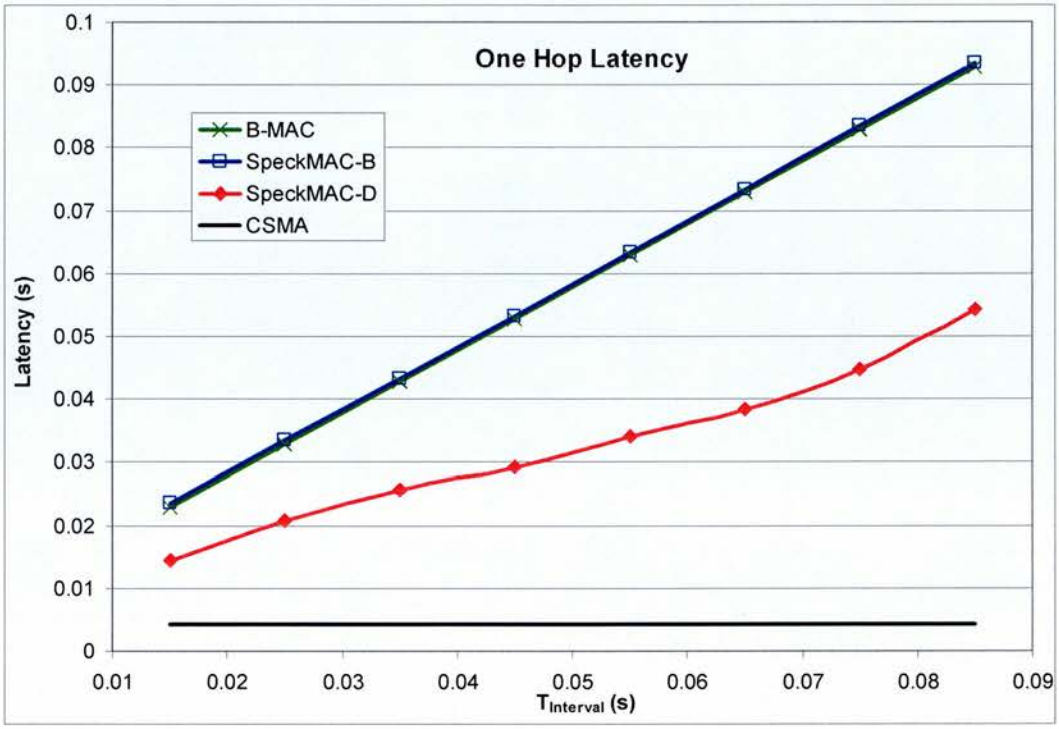


Figure 3-42: The one-hop latency measured on the ProSpeckz under the three MAC protocols

3.7.6 Multi-hop latency

In the previous section, the latency performances for the three MAC protocols were presented for a single hop. In this section, the performance of multiple-hop latency is explored. In multi-hop wireless networks such as Specknets, the radio range of each node provides coverage only across a small proportion of the entire network. Packets are relayed from one node to another to support network connectivity across the entire network. An example of a multi-hop network is shown in Figure 3-43 in which the packet sent by Node A is relayed by Node B and Node C before reaching Node D. In most cases, the equation for calculating multi-hop latency can be approximated by:

$$n\text{-hop latency} = (\text{one-hop latency}) * n \tag{51}$$

where n is the number of hops to be traversed by the packet.

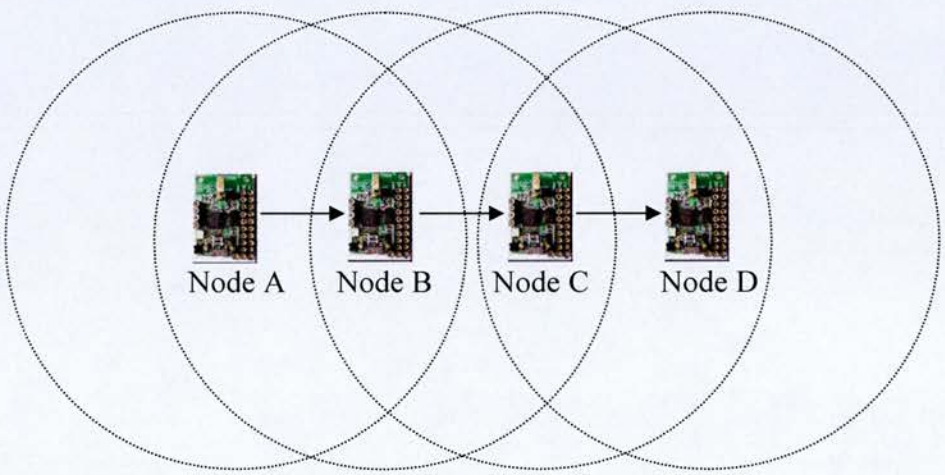


Figure 3-43: An example showing a multi-hop network where nodes B and C relay the packets sent by node A to node D.

Equation (51) was validated by measuring the latency for the three MAC protocols in a multi-hop situation depicted in Figure 3-43. Four ProSpeckz devices were positioned such that the coverage for each node mimics the radio coverage. For the experiment, Node A transmitted a packet using the frame formats shown in Figure 3-15 with a data payload of 32 bytes. On placing the packet in its transmission buffer, Node A would toggle the state of its I/O pin. On receiving the data packet, nodes B and C retransmits the received packet and Node D then toggle the state of its I/O pin when the packet is received into its receiver buffer. The latency across this three-hop network was obtained by measuring the timing difference between signals of the I/O pins from Node A and Node D. For each algorithm, the values of $T_{Interval}$ ranged from 15ms to 85ms (in intervals of 10ms), and for each scenario, 200 iterations were performed. The measurements from the experiment were averaged across the iterations for each scenario, and the results are shown in Figure 3-44.

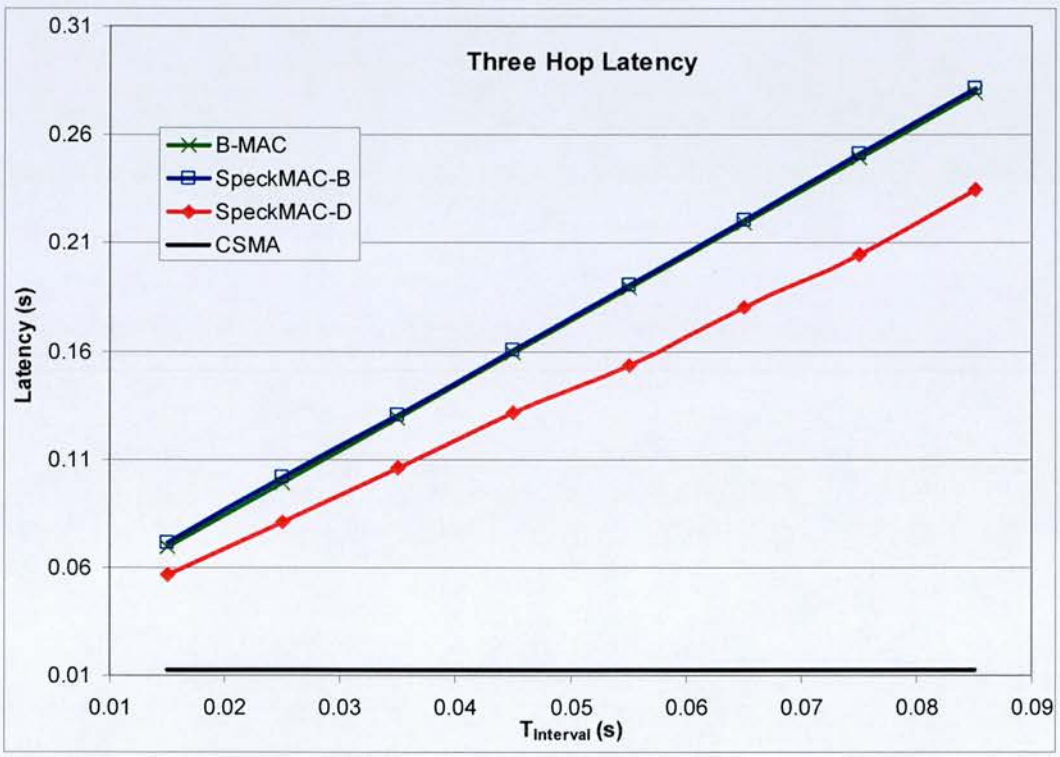


Figure 3-44: The three-hop latency measurements from the ProSpeckz running the different MAC protocols

In order to compare the results obtained from the one-hop and the three-hop scenarios, the results shown in Figure 3-44 were mapped into linear equations as follows:

$$^{3hop}\mathbf{L_{BMAC}} = (2.9982 * T_{Interval}) + 0.0243$$

$$^{3hop}\mathbf{L_{SpeckMACB}} = (3.0016 * T_{Interval}) + 0.0257$$

$$^{3hop}\mathbf{L_{SpeckMACD}} = (2.5051 * T_{Interval}) + 0.0182$$

(52)

where, $^{3hop}L_{BMAC}$, $^{3hop}L_{SpeckMACB}$ and $^{3hop}L_{SpeckMACD}$ are the 3-hop latency measurements for B-MAC, SpeckMAC-B and SpeckMAC-D respectively. Equation (51) was validated by substituting results in (50) into (51) and comparing against (52) as follows:

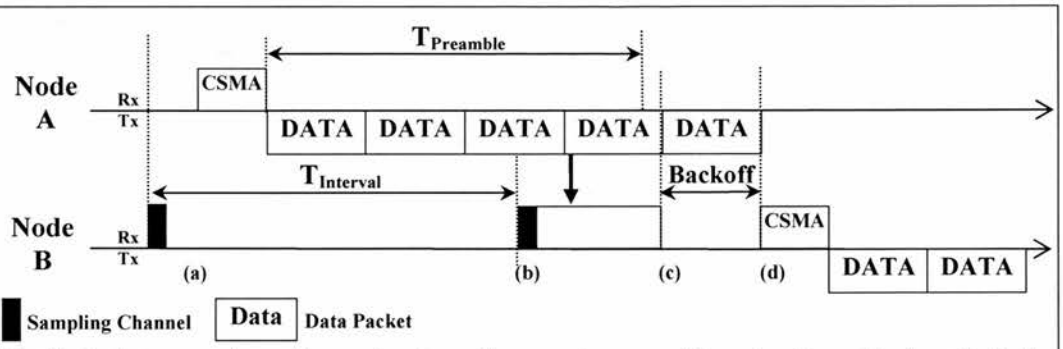
$$3 * L_{BMAC} = (2.9979 * T_{Interval}) + 0.0234 \approx {}^{3hop}L_{BMAC}$$

$$3 * L_{SpeckMACB} = (3.0015 * T_{Interval}) + 0.0249 \approx {}^{3hop}L_{SpeckMACB}$$

$$3 * L_{SpeckMACD} = (1.5753 * T_{Interval}) + 0.0192 \neq {}^{3hop}L_{SpeckMACD}$$

(53)

It can be seen in (53) that the latency results for ProSpeckz utilising the B-MAC and SpeckMAC-B algorithms follow equation (51). In contrast, this was not the case for SpeckMAC-D. This is explained using Figure 3-45 which shows the interaction between a transmitting node, Node A, and a relaying node, Node B using SpeckMAC-D.



(a) Node A turns on the radio receiver to perform carrier sense. Since the channel is free, Node A begins transmitting the data frames.
(b) Node B performs channel sampling and sense a busy channel. Thus, it keeps the radio receiver on in anticipation of an incoming data frame.
(c) Node B completes receiving the data frame. However, it could not relay the packet immediately as Node A is still transmitting. Instead, it backoffs using the timing information extracted from the data frame.
(d) Node B completes the backoff process and is now able to contend for the channel for transmission. It performs carrier sense and since the channel is free, it begins relaying the received data packet.

Figure 3-45: The interactions between Node A, a transmitting node and Node B, a relaying node using SpeckMAC-D.

With reference to Figure 3-45, even though Node B had received the data packet from Node A, it is not be able to relay the packet immediately. This was because Node B had to backoff from the channel after receiving the data frame. Using the backoff timing information embedded in the received data frame, Node B turns off its radio until Node A finishes its transmission as shown in Figure 3-45(c). For a *n*-hop network, the latency for SpeckMAC-D, ${}^{n\text{Hop}}L_{\text{SpeckMACD}}$, can be approximated as follows:

$$
\begin{aligned}
{}^{n\text{Hop}}L_{\text{SpeckMACD}} &= \text{Latency for first (n-1) hops} + \text{Latency for last hop} \\
&= (n\text{-}1)\,(T_{\text{Interval}} + C_{\text{SpeckMACD}}) + (T_{\text{Interval}}/2 + C_{\text{SpeckMACD}})
\end{aligned}
\tag{54}
$$

where $C_{\text{SpeckMACD}}$ is the constant mentioned introduced in Section 3.7.5.

Assuming a value of 0.0064 for $C_{SpeckMACD}$ based on the latency measurement obtained from the one-hop scenario for SpeckMAC-D (50), equation (54) is validated by comparing it with the results in (52):

$$^{3Hop}L_{SpeckMACD} = (3\text{-}1)\,(T_{Interval} + 0.0064) + (T_{Interval}/2 + 0.0064) \quad \text{...from (54)}$$

$$= (2.5 * T_{Interval}) + 0.0192$$

$$\approx (2.5051 * T_{Interval}) + 0.0182 \quad \text{...from (52)}$$

Therefore, equation (54) maps to the latency performance achieved by the ProSpeckz. Using the equation (51) for B-MAC and SpeckMAC-B, as well as equation (54) for SpeckMAC-D, the following inequality is derived:

$$^{n\text{Hop}}L_{SpeckMACB} \geq {}^{n\text{Hop}}L_{BMAC} > {}^{n\text{Hop}}L_{SpeckMAC\text{-}D} \tag{55}$$

where $^{n\text{Hop}}L_{SpeckMACB}$, $^{n\text{Hop}}L_{BMAC}$ and $^{n\text{Hop}}L_{SpeckMAC\text{-}D}$ are the latencies for B-MAC, SpeckMAC-B and SpeckMAC-D, respectively. In conclusion, in a multi-hop scenario, the experiments demonstrated that SpeckMAC-D outperformed B-MAC in terms of latency.

3.8 Battery lifetime experiments for an application using broadcasting traffic

In order to compare the battery lifetime performances realistically, B-MAC, SpeckMAC-B and SpeckMAC-D were implemented on the ProSpeckz prototypes for a logical location maintenance algorithm (see Appendix D.5) that was designed for Specknets. In this algorithm, each node sends a location-update packet periodically to inform its one-hop neighbours about its estimated logical location, together with the list of its neighbours. On receiving the update packet, a node compares the list of neighbours of the sending node with its own list of neighbours, and recalculates the estimate of its logical location. As reported in the paper, this algorithm performed optimally in a dense Specknet when each node has eleven or more neighbours. Therefore, for the experiments in this section, twelve nodes were used to form a single hop network.

In the implementation of the experiment, in order to conserve energy, the processor onboard the ProSpeckz was placed in the sleep mode whenever the ProSpeckz was neither processing, transmitting nor receiving a packet. The structure of the data payload for the transmission of the location update is shown in Figure 3-46. Each node transmits one location update packet per second using a transmission power of 0dBm.

Data Payload

Estimated X Coordinate	Estimated Y Coordinate	Hash list of neighbours' addresses
Bytes: 4	4	24

(Total payload size: 32bytes)

Figure 3-46: The data payload format used for the location maintenance algorithm

The performance of the protocols depends on the choice of $T_{Interval}$. Based on the parameters from the physical implementation, and the equations derived in Sections 3.4 and 3.6, the optimal $T_{Interval}$ for each algorithm could be determined, given the expected number of packets to be transmitted per second and the expected one-hop neighbours a node will have for a given application. For this experiment, the calculated power consumption of the radio across the range of $T_{Interval}$ is shown in Figure 3-47. In the figure, B-MAC performs optimally when the value of $T_{Interval}$ is 7.5ms and the resultant power consumption would be 8.6mW. Similarly, the values of $T_{Interval}$ selected for SpeckMAC-B and SpeckMAC-D were 16.7ms and 15.9ms, respectively. The expected ratio of power consumption achieved for the radio by the nodes running the two variations of SpeckMAC over B-MAC in the given scenario was calculated to be:

$$RATIO_{B\text{-}MAC/SpeckMAC\text{-}B} = 8.6mW / 5.47mW = 1.572$$

$$RATIO_{B\text{-}MAC/SpeckMAC\text{-}D} = 8.6mW / 5.09mW = 1.69 \quad (56)$$

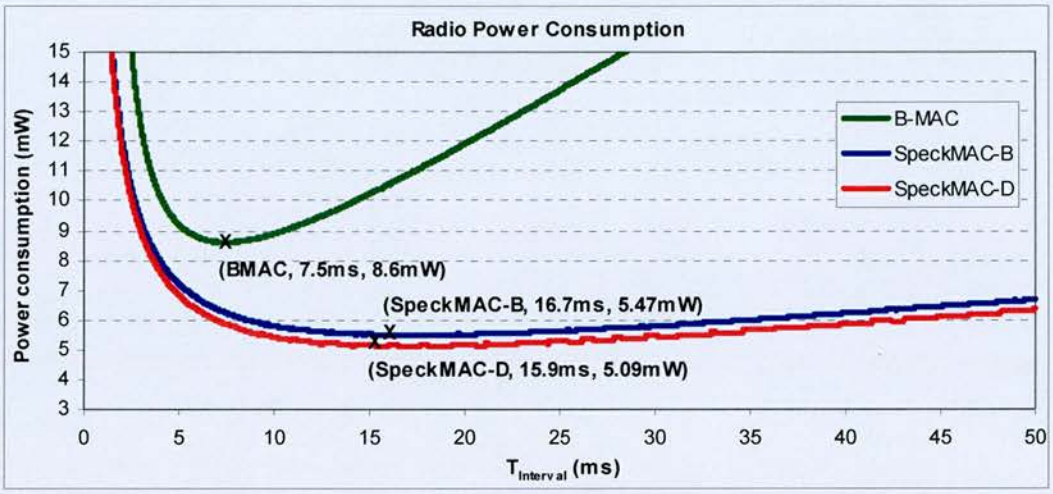


Figure 3-47: The radio power consumption of the three protocols transmitting one packet per second with eleven neighbours across a range of $T_{Interval}$

In the experiment, a battery-powered ProSpeckz was placed with eleven other mains-powered ProSpeckz to form a single-hop networked cluster. The cluster was tested to ensure that bi-directional links were established between the battery-powered ProSpeckz and the mains-powered ones. This ensured that the battery-powered ProSpeckz had eleven neighbours as required.

Four such clusters were formed, with each cluster operating a different MAC protocol and at a different non-interfering frequency. The four MAC protocols used in this experiment were B-MAC, SpeckMAC-B, SpeckMAC-D and CSMA. The ProSpeckz running CSMA had its radio turned on all the time. All the ProSpeckz communicated with a transmission power of 0dBm and sent location-update packets every second. The optimal values for $T_{Interval}$ for the three protocols marked in Figure 3-47 were used in the experiment.

Two types of batteries were employed for evaluating the lifetime of the nodes: 12x14mm Polymer Li-ion cells (3.7v, 40mAh@40mA), and CR1220 Lithium coin cells (3v, 40mAh@0.1mA). The voltage levels of the batteries were sampled every millisecond by a 12-bit oscilloscope calibrated to provide an accuracy greater than 99%. For each type of battery, the experiments were repeated four times to allow each MAC protocol to be rotated between the different ProSpeckz, and across the set of frequencies. For each execution, the battery-powered ProSpeckz were turned on at

exactly the same time and placed at close proximity to each other. This mitigated any effects due to variations in room temperature (which was kept between 20-25 degrees Centigrade), and noise interferences between them during the experiment.

For the experiments using the Polymer Li-ion batteries, new ones were used for each run. The batteries were first fully discharged before being recharged in parallel so as to ensure all batteries would have a similar charge. Based on the data recorded from the 12-bit digital storage oscilloscope, the discharge graph for the Polymer Li-ion batteries over the four runs is shown in Figure 3-48. The discharge graphs for the Polymer Li-ion battery were plotted using the minimum voltage measured every ten thousand samples (ten second resolution). For this analysis, the life of a ProSpeckz was assumed to have ended once the battery voltage falls below three volts.

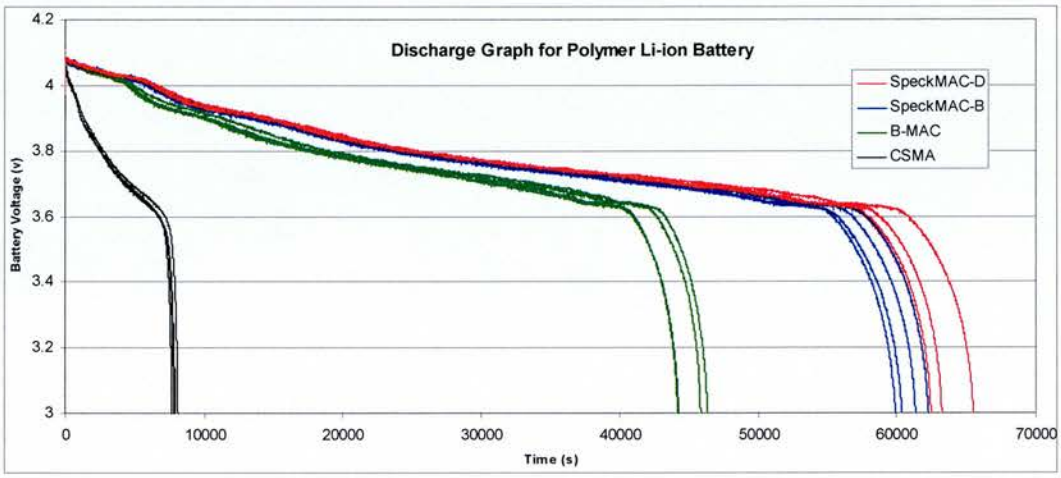


Figure 3-48: The discharge graph for the Polymer Li-ion batteries over the four iterations

Over the four runs, the average lifetimes of the ProSpeckz running the different protocols are shown in Table 3-4. The results demonstrated that the energy-saving capabilities of B-MAC, SpeckMAC-B and SpeckMAC-D enabled the Polymer Li-ion batteries to power the ProSpeckz for 473%, 675% and 705%, respectively, longer than when no energy-saving MAC was used. Furthermore, ProSpeckz utilising SpeckMAC-B and SpeckMAC-D managed to operate for periods of 35% and 40% longer, respectively, than B-MAC. This improvement in lifetime was less than the ratios calculated in equation (56) because the energy consumed by the CPU and other 'real life' parameters such as noise interferences, battery characteristics and

temperature fluctuation had not been modelled by the equations. In conclusion, for the logical location maintenance algorithm, both versions of SpeckMAC outperformed B-MAC in terms of life expectancy when Polymer Li-ion batteries were used to power the ProSpeckz.

	CSMA	**B-MAC**	**SpeckMAC-B**	**SpeckMAC-D**
Lifetime	7858s (2.18hrs)	45088s (12.52hrs)	60938s (16.93hrs)	63310s (17.59hrs)
(σ)	187.68s	1101.34s	1051.42s	1435.15s
(C_V)	2.39%	2.44%	1.73%	2.27%
Improvement over CSMA		473.81%	675.53%	705.72%
Improvement over B-MAC			35.15%	40.42%

Table 3-4: The average lifetime of the ProSpeckz powered by the Polymer Li-ion batteries (Cv = Coefficient of Variation)

In the next set of experiments, the Polymer Li-ion battery was replaced by two CR1220 button cells for powering each ProSpeckz. This mimicked scenarios where smaller batteries are used to power the miniature specks. Given their sizes, these batteries would not be able to sustain high current drains and therefore, the effects of duty-cycling would have a significant impact on the lifetime of the batteries. This set of experiments compared the performance of the MAC protocols using the same logical location maintenance algorithm and setup as in the previous set of experiments.

The discharge graphs for the CR1210 coin cells used in the experiment are shown in Figure 3-49 in which the minimum voltage measured every thousand samples (one second resolution) were plotted. Unlike the Polymer Li-ion batteries, the coin cells reached a deep discharge depth during the initialisation of the radio and the PSoC. A relaxation period of 5 seconds enables the battery to recover and return to its nominal voltage before the location algorithm was executed. During the recovery period, both the radio and the PSoC on the ProSpeckz were placed in the sleep mode.

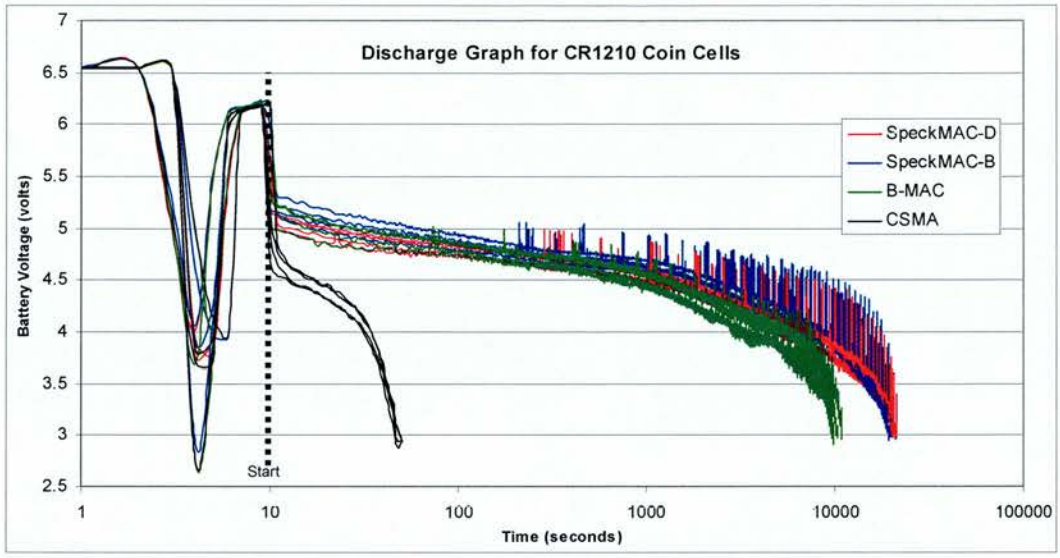


Figure 3-49: The discharge graph for the CR1210 coin cell batteries over the four iterations

The lifetime of the ProSpeckz for the four protocols was averaged over the four runs and is shown in Table 3-5. A slightly bigger variation between the results was observed over the different runs as compared to the experiments with the Polymer Li-ion batteries. This could be explained by the fluctuations in the battery capacities of the coin cells. Unlike experiments with the Polymer Li-ions batteries in which all the batteries used in a particular run were completely discharged and re-charged in parallel to ensure that the energy capacities of the batteries were as close as possible, the same approach could not be used for the coin cells as they were not rechargeable. Furthermore, due to the smaller capacities of coin cells, a small difference in capacities would results in greater variation in lifetime measurements.

	CSMA	**B-MAC**	**SpeckMAC-B**	**SpeckMAC-D**
Lifetime	38s (0.01hrs)	10173s (2.83hrs)	20039s (5.57hrs)	20725s (5.76hrs)
(σ)	1.26s	571.05s	801.49s	410.17s
(C_V)	3.33%	5.61%	4.00%	1.98%
Improvement over CSMA		26848%	52982%	54803%
Improvement over B-MAC			96.98%	103.73%

Table 3-5: The average lifetime of the ProSpeckz powered by the CR1210 coin cell batteries
(Cv = Coefficient of Variation)

The results from the experiment demonstrated that the button cells deplete rapidly when a current that is beyond its nominal load was drawn constantly. With the radio on all the time, the button cell powered the ProSpeckz for an average of 41 seconds before falling below the operating voltage of three volts. Using B-MAC, SpeckMAC-B and SpeckMAC-D, the cells managed to achieve lifetimes that were 26848%, 52982% and 54803%, respectively, longer than the ProSpeckz without any energy-saving MAC. This was a significant improvement that demonstrated the importance of energy-saving MAC algorithms in Specknets and other energy-constrained devices.

Some interesting results were also observed when using the different types of batteries when comparing the lifetimes achieved by the two SpeckMAC protocols and the B-MAC protocol. When the CR1210 coin cell batteries were used, ProSpeckz employing SpeckMAC-B and Speck-MAC-D displayed a 97% and 103.7% increase in lifetime over B-MAC, respectively. These improvements were significantly greater than when the ProSpeckz were powered by the Polymer Li-ion batteries, which had shown improvements of 35.1% and 40.4% over B-MAC for SpeckMAC-B and SpeckMAC-D, respectively. These differences in level of improvements were due to the abilities of the different battery types to sustain the current draws required by the ProSpeckz. To demonstrate this, two ProSpeckz were used with one powered by the CR1210 batteries and the other powered by the Polymer Li-ion battery. Both ProSpeckz communicated using B-MAC. The voltages across the batteries were recorded using a digital storage oscilloscope and the results are shown in Figure 3-50. Whenever the radio and/or the processor were turned on, the drain causes the voltage of the cells to fall to a certain discharge depth (the stressed phase). When the current drain was removed, the cell was able to recover some of its lost voltage (the recovery phase). The results obtained from the storage scope showed that the voltage of the Polymer Li-ion batteries dropped by less then 2% when the ProSpeckz was sampling the channel, receiving a packet or transmitting a packet. However, in the case of the CR1210 batteries, the voltage across the batteries dropped by more than 10% when transmitting or receiving a packet, and by almost 8% when sampling the channel. Furthermore, it could be seen that the

CR1210 coin cells required a longer recovery time, as compared to the Polymer Li-ion batteries, to return to a stable nominal voltage.

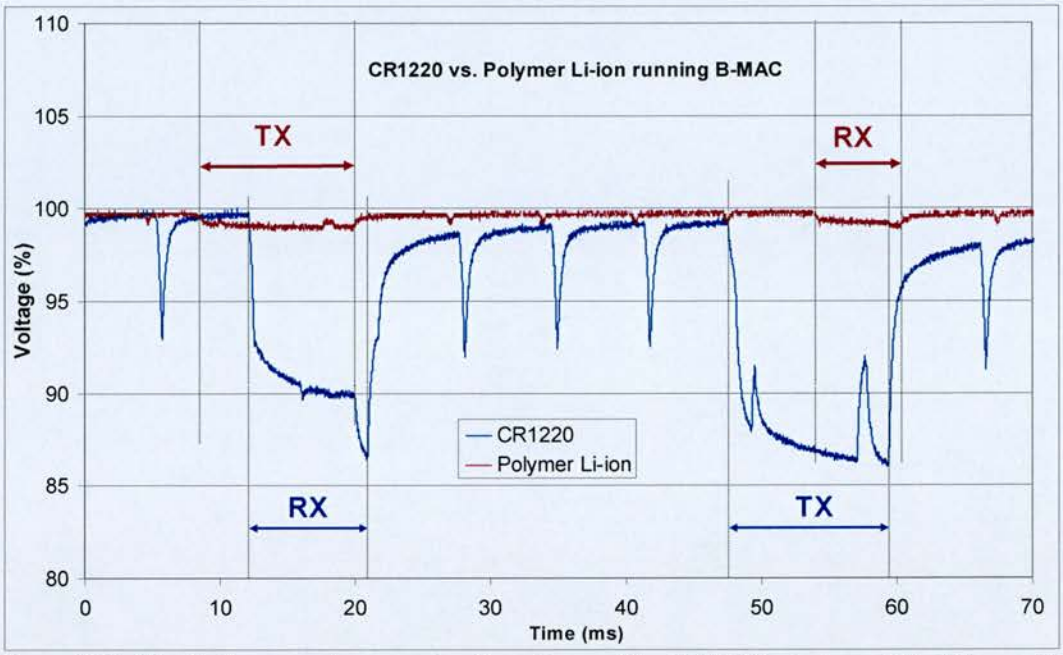


Figure 3-50: The discharge graph comparing the voltages of the CR1210 batteries and the Polymer Li-ion battery that were used to power the ProSpeckz utilising B-MAC

As the CR1210 coin cells were more sensitive to the current drains as demonstrated in Figure 3-50, a small difference in the receiver turn-on times would have a greater impact on the lifetime of the nodes. This was demonstrated by another example, where two ProSpeckz were powered by the CR1210 coin cell batteries, with one ProSpeckz employing B-MAC and the other ProSpeckz utilising SpeckMAC-D. As the SpeckMAC-D ones would generally turn on their receivers for a shorter duration than the ProSpeckz utilising B-MAC, less time would be required for batteries on the ProSpeckz using SpeckMAC-D to recover from the stressed phase. Figure 3-51 shows both ProSpeckz beginning their recovery phase at 30ms. However, the batteries on the SpeckMAC-D ProSpeckz were able to return to its nominal voltage before the next channel sampling. Nodes utilising B-MAC, however, needed a much longer recovery time before its batteries returned to its nominal voltage, as the batteries suffered a longer stressed phase due to the longer receiver on-time.

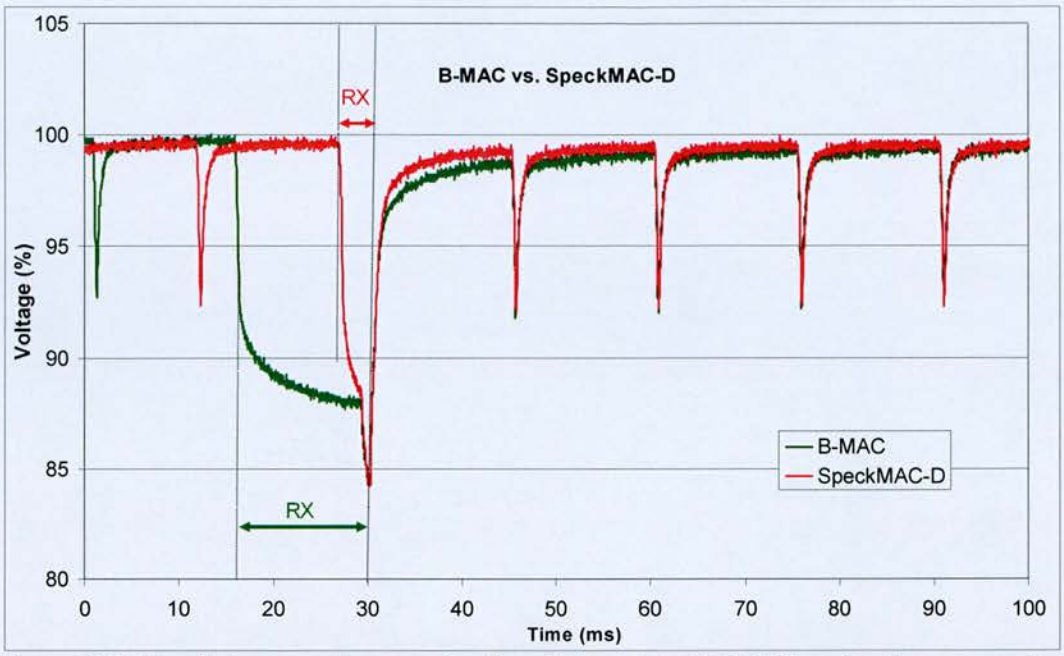


Figure 3-51: The discharge graph comparing the voltages of the CR1210 batteries that were used to power the ProSpeckz utilising B-MAC and SpeckMAC-D

In conclusion, the experiments using the two type of batteries demonstrated that both variations of SpeckMAC outperform B-MAC in terms of life expectancy, and this improvement was more significant as the battery's ability to sustain the discharge load from the radio decreases.

3.9 Summary and Discussion

This chapter described the design, implementation and evaluation of a collection of novel, distributed, asynchronous and random-access MAC algorithms, SpeckMAC, designed to enable energy-savings in Specknets, given the assumption that wireless communications in Specknets have relatively small packet sizes and low bandwidth requirements. The two variations of SpeckMAC; SpeckMAC-B and SpeckMAC-D, replaced the preamble bytes transmitted by B-MAC with wakeup frames and redundant data frames respectively. Using simple mathematical models, the initial feasibility analysis demonstrated the promise of the SpeckMAC algorithms to achieve higher energy-efficiency than B-MAC.

A more realistic comparison between the performances of the B-MAC, SpeckMAC-B and SpeckMAC-D, was achieved by implementing them on the ProSpeckz and the PerSpeckz-64 hardware platforms. The results from the hardware experiments are summarised:

i) in terms of energy-efficiency under broadcast traffic conditions: SpeckMAC-D demonstrated the best energy-efficiency, with B-MAC demonstrating the worst

ii) in terms of energy-efficiency under unicast traffic conditions: SpeckMAC-B demonstrated the best energy-efficiency, with B-MAC demonstrating the worst

iii) in terms of delivery ratio: SpeckMAC-D obtained the highest delivery ratio, while both SpeckMAC-B and B-MAC demonstrated similar, but lower, delivery ratios

iv) in terms of energy wastage when the radio channel is jammed: SpeckMAC-B demonstrated the possibility of wasting the least amount of energy, whereas B-MAC demonstrated the biggest energy loss when the channel was jammed.

v) In terms of latency: SpeckMAC-D achieved the shortest latency, while SpeckMAC-B achieved a slightly longer latency than B-MAC

In summary, the two variations of SpeckMAC demonstrated better energy-efficiency than B-MAC. Furthermore, SpeckMAC-D demonstrated a higher delivery ratio and shorter latency than B-MAC. To obtain a realistic measure on the improvement in battery lifetime, the MAC protocols were used to support a logical location maintenance algorithm running on battery-powered ProSpeckz. Two types of battery cells with different drain profiles were used, and the results from the experiments demonstrated that the use of energy-efficient MAC algorithms would significantly prolong the lifetime of the ProSpeckz. Furthermore, ProSpeckz utilising SpeckMAC-B and SpeckMAC-D had achieved a longer lifetime than ones utilising B-MAC, and this increase in lifetime was more significant in the case of the coin cell batteries.

3.10 Future work

In this chapter, it was shown that the drain profile of the batteries has a significant impact on lifetime of the nodes; however, the exact relationship between the two was not investigated in detail and is left for future work. The analytical models presented in this chapter can also be improved to take into consideration collisions, as well as the CPU energy usage. An algorithm for dynamic $T_{Interval}$ could also be devised to reduce the energy required for channel listening when there is minimal activity in the network. These can be done by profiling the data usages and network neighbourhood sizes to predict possible traffic. In this case, a distributed algorithm will also be needed for coordinating the value of $T_{Interval}$ between nodes in the network such that all nodes in a participating network have the same $T_{Interval}$ value. These algorithms and the complex implementation were not investigated in this thesis and are left for future work. Another improvement that can be made is the use of some handshaking mechanism to improve the delivery ratio of the SpeckMAC algorithms. However, the overheads needed to support handshaking might have a detrimental effect on the energy efficiency. This cost-benefit relationship will be explored in the future.

Several algorithms had been developed by other researchers for improving the energy-efficiency of B-MAC. These algorithms may also be ported to SpeckMAC and would be investigated in the future. Two examples are briefly described.

- Zebra-MAC (Z-MAC) [82] is a hybrid MAC protocol that is built on-top of B-MAC and combine the strengths of CSMA and TDMA. Nodes are synchronised and TDMA slots are assigned to nodes (the nodes will "own" the slot) in a distributed fashion when the network is deployed. However, unlike TDMA, nodes can transmit at any time using CSMA but will be given priority to transmit in the slots they own. Using this approach, Z-MAC is able to improve the energy-efficiency, as compared to B-MAC, when traffic increases in the network due to the contention avoidance achieved by using a TDMA approach. However, evaluations presented were based on a static network and thus, the design of Z-MAC may have to be extended in order to support a multi-hop mobile ad-hoc network like Specknets.

- Uncertainty-driven B-MAC (UBMAC) [83] is an example of integrating a time synchronisation algorithm with B-MAC. In this case, the Rate Adaptive Time Synchronization (RATS) protocol was used to provide a synchronisation mechanism between nodes that allows estimations of the uncertainty between the clocks of the sender and receiver to be determined. As such, the sender is able to estimate the approximate period in which an intended receiver is likely to sample the channel. Thus, the length of the preamble transmitted can be shortened to cover just sufficiently the uncertainty and wakeup the intended receiver. This improves the energy-efficiency of the transmitter by decreasing the number of preamble bits transmitted. However, evaluations presented were based on a single-hop network with just three static nodes and thus, further investigation would have to be carried out to determine the scalability of this time synchronisation protocol to support the requirements of a Specknet.

It is also possible to further improve the energy efficiency of the Specks by performing some clustering protocols between the routing algorithms (at the network layer) and the SpeckMAC algorithms (at the MAC layer). These clustering approaches allow the Specks to be placed into sleep states more frequently by coordinating the responsibility of forwarding packets between nodes in a cluster. For example, if the geographical location of the Specks could be determined, Geography-informed Energy conservation (GAF) [84] protocol could be deployed to divide the network into clusters of square grids. Specks in each grid would take turns to listen and sleep such that, at any time, there is always one active Speck in each grid to forward packets for the routing protocol. In the absence of geographical location information, local connectivity measurements can also be used, such as in the case of the ASCENT [85] protocol, to determine which Specks are to be active. For dense networks such as a Specknet, not all Specks need to be active to support the routing algorithms due to the overlapping coverage of the Specks. In such a scenario, the SPAN [86] protocol can be used to determine the status of the nodes, either active or sleeping, in a distributed and randomized manner to actively maintain a backbone such that the connectivity or capacity of the network is not diminished. These

complementary techniques are performed above the SpeckMAC algorithms to further enhance the energy-efficiency of the Specks and will be explored in the future.

Some ideas presented in Appendix C may also be improved and evaluated as possible energy-efficient MAC protocols for Specknet in the future.

Lastly, in this chapter, the performance of the SpeckMAC protocols was compared with the B-MAC protocol. It would be interesting to evaluate the performance of SpeckMAC with other wireless sensor network MAC protocols developed after, or at the same time as SpeckMAC, including:

- X-MAC [87] is a algorithm that uses a sequence of wakeup frames containing short preambles and the destination address to "wakeup" the intended destination node. This approach is similar to SpeckMAC-B, except that source nodes will listen for a short duration after transmitting each wakeup frame for an acknowledgement from the destination node. After receiving the acknowledgement, the source node would transmit the data frame. This reduces the number of redundant wakeup frames transmitted for unicast traffic.
- Scheduled-Channel Polling MAC (SCP-MAC) [88] combines the use of scheduling and random access techniques. Similar to B-MAC, all nodes would periodically wakeup to sample the channel for activity and a node with data to send would transmit a busy signal to wakeup neighbouring nodes. However, in SCP-MAC, all nodes are synchronised to a common schedule and will perform channel sampling at the same time. Thus, to transmit a packet, only a short busy signal needs to be transmitted. To decrease the possibility of a collision, a two-phase contention using CSMA is performed, once before the busy signal is transmitted, and once again before the data frame is transmitted.
- Spatial correlation-based collaborative MAC (CC-MAC) [89] makes use of the spatial correlation between the location of the sensor nodes and the location of the events-of-interest in selecting the appropriate nodes to participate in the transmission of the data. CC-MAC is made up of two components - an event MAC (E-MAC) and a network MAC (N-MAC). The Event MAC (E-MAC) is used to filter and aggregate sensor node measurements using the estimated

distance between the location of the event and the sensor nodes thus reducing the traffic generated. N-MAC is then used to allow the data filtered-out by E-MAC to be forwarded with higher precedence to the data sink over non-filtered data of a concurrent event.

Chapter 4

THE NETWORK LAYER

"Investigate the possibility of employing common MANET routing algorithms, such as Dynamic Source Routing (DSR) and Ad-hoc On-Demand Vector Routing (AODV), on Specknets running SpeckMAC as the MAC protocol in order to support energy-efficient peer-to-peer communications in a wireless multi-hop ad-hoc network."

4.1 Introduction

MAC algorithms enable nodes in the same neighbourhood to coordinate the access to the common wireless medium; however, routing algorithms at the network layer would be needed to enable nodes to transmit data packets to nodes other than their immediate neighbours. This requires the use of intermediate nodes to relay the packets, and the task of selecting these intermediate nodes should be handled by the routing algorithm in the network layer. The operations of any routing algorithm can usually be divided into three parts: route discovery, route recovery and route maintenance. Route discovery finds routes between any two nodes, and once found, are stored in a cache or a route table. Route recovery is used whenever a discovered or cached route has been broken. The routing algorithm would either perform route recovery to mend the route or it could perform route discovery again. Route maintenance ensures that the routes are consistent to some routing performance metrics and ensures some bounded quality of service for the routes.

Routing algorithms can be classified into the following two categories: proactive and reactive. Reactive routing algorithms finds a route 'on-demand' when initiated by the source or destination nodes, whereas in the case of proactive routing algorithms, routes in the network would be consistently discovered and maintained in advance, even though not all routes are required at any moment in time. In both approaches, supporting routing algorithms in a resource-constrained Specknet poses challenges when compared to traditional wireless Mobile Ad-hoc Networks (MANET) due to the

power and memory limitations. These difficulties would be further compounded given that energy-efficient MAC protocols that were designed for sensor networks and Specknets usually duty-cycle the radio to minimise the power consumption based on the assumption of low data bandwidth requirements and tolerance to long latency. However, these assumptions would be critical to routing protocols as their functionality, especially in mobile ad hoc networks, depends heavily on the bandwidth available to perform the required routing tasks, as well as short latencies to enable discovered routes to remain valid. As SpeckMAC belongs to the class of MAC protocols that trades bandwidth and latency performance in return for better energy efficiency, it would be interesting to investigate the possibility of supporting the traffic requirements of MANET routing algorithms with SpeckMAC as the underlying energy-efficient MAC algorithm.

4.1.1 Assumptions and limitations

The results and analysis presented in this chapter were based on simulations carried out using the Qualnet network simulator [90]. The models of the simulated nodes were based on the attributes of the ProSpeckz; however, only the radio was modelled and not the processor. Therefore, all computational tasks performed by the nodes in the simulation were assumed to compute instantaneously. Furthermore, as there were no suitable short-range channel model available in the Qualnet distribution, (and the modelling of such a channel model would be beyond the scope of this thesis) the range of the radios was simulated to be 250m instead of the actual 1-2 meters that would be supported by the Specks. Additionally, only two environments were considered: (i) all nodes are static, (ii) all nodes are static except for the server node. These two simulated environments do not cover the whole spectrum of density, scale and mobility patterns which are possible, although they are representative of the experiments for analysing the feasibility of SpeckMAC supporting MANET algorithms in both static and mobile scenarios.

4.1.2 Objective and Scope

The objective of this chapter was to demonstrate, through the use of software simulations, that SpeckMAC protocols supports common peer-to-peer MANET routing algorithms, such as DSR and AODV, in a more energy efficient manner than MAC algorithms that keeps the radio on constantly. It also provides some insight into the performance of the routing algorithms using radios that communicate with low data-rates of 250kbps (instead of the commonly simulated 1-11Mbps rates supported by radios used in 802.11 networks) before the implementation of either DSR or AODV on the physical devices with low bandwidth, such as the ProSpeckz.

4.1.3 Chapter Overview

This chapter begins with a survey of the categories of routing protocols commonly used in wireless mobile ad-hoc networks and a comparison between appropriate techniques for routing in Specknets to support peer-to-peer communications. Following this, the implementation of the Specknet algorithms on the Qualnet simulator would be discussed. The validation of the models in Qualnet is presented by comparing the output of the simulator with the operation of the ProSpeckz implementation. Further verification is presented by comparing the results obtained from the simulators with the results obtained from the physical experiments and mathematical calculations shown in Figure 3-27 and Figure 3-32. Two reactive routing protocols, DSR and AODV, were simulated in Qualnet and the results for both static and mobile environments were analysed. In both cases, SpeckMAC was able to support the reactive routing algorithms with a higher energy-efficiency than MAC algorithms that kept the radio on consistently. A novel hybrid version of SpeckMAC is proposed which enables SpeckMAC-B and SpeckMAC-D to be used simultaneously in a Specknet. This hybrid approach is an attempt to combine the advantages of the two SpeckMAC protocols by using the knowledge at the higher layers of the communication stack to select the appropriate version of the SpeckMAC protocol to be used for transmitting each packet. A simple combination is proposed where SpeckMAC-B and SpeckMAC-D would be used for unicast and

broadcast packets respectively. The simulation results demonstrated that when a data payload was larger than 64 bytes, SpeckMAC-H managed to achieve higher energy-efficiency than the singular approaches. Finally, this chapter concludes with ideas for future research in the networking layer.

4.2 Background

Routing algorithms for wireless mobile ad-hoc networks had been an active area of research in the recent years given the greater availability and increased usage of portable devices such as PDAs, laptops and mobile phones. Routing algorithms could normally be classified into the following types: proactive, reactive, hybrid, data-centric and location-based. In this section, example algorithms of each type are discussed and their feasibility for Specknets is explored.

4.2.1 Proactive (table-driven) routing algorithms

4.2.1.1 Destination-Sequenced Distance-Vector (DSDV)

In DSDV [91], every node holds a routing table which contains information about all possible destination nodes in the network and the number of hops it takes for a packet to reach the node. This algorithm is a variation of the classical Bellman-Ford routing mechanism [92]. The routing table is updated regularly with each node periodically broadcasting its routing table entries to its neighbours. This update could be a full dump (whole table is transmitted) or a partial dump (only entries that changed is transmitted). Since the table is always updated, routes to any destination node are always maintained regardless of whether such a route is ever used results in redundant traffic. Furthermore, this algorithm may not be suitable for networks where link changes are frequent, e.g., due to high mobility or duty-cycling as in the case of Specks. This is because route changes may occur much faster than the propagation of route table updates across the network. Specknets are also meant to be highly dense networks; thus, the routing table used in this algorithm could be significantly large (O(n) per node where n is the total number of nodes in the network). The only advantage of using this algorithm is the low latency required to

send a data packet as all routes are constantly maintained and only table lookups are required to find a route to the destination node.

4.2.1.2 Cluster-head Gateway Switch Routing (CGSR)

CGSR [93] differs from DSDV in the way that the network is organised. Instead of treating the network as 'flat', CGSR organises it into clusters of multi-hop networks with several heuristic routing schemes. Having cluster-heads controlling a group of ad-hoc nodes in this 'hierarchical' fashion provides a framework for code separation (among clusters), routing, channel access and bandwidth allocation. Using DSDV as the underlying routing scheme, each node now stores a 'cluster member table' which stores the destination cluster-head for each node in the network. To route a data packet, the node would first route to its cluster-head which in turn would route the packet to one of the gateways towards the destination node's cluster-head. A gateway is any node that has access to more than one cluster-heads. This routing algorithm works well in the case of highly dense networks with low mobility. However, it is less well suited for Specknets due to the requirement to elect cluster-heads. In a Specknet, Specks have limited power thus those elected as cluster-heads would deplete their energy resources quite quickly in order to route packets to and from its cluster members. Therefore, frequent cluster-head re-elections would be necessary to replace cluster-heads with low remaining energy resources. The need for frequent cluster-head re-elections would create additional overheads to the network.

4.2.1.3 Other proactive routing algorithms

Other commonly known proactive routing algorithms include Wireless Routing Protocol (WRP) [94] and Optimised Link-State Routing (OLSR) [95]. WRP is a table-based protocol that belongs to the general class of Path-Finding Algorithms (PFA) [96, 97], defined as the set of distributed shortest-path algorithms that calculate the paths using information regarding the length and second-to-last hop (the predecessor) of the shortest path to destination. OLSR is an optimisation over the classical link state protocol, tailored for mobile ad hoc networks. The key idea of OLSR is the use of multi-point relay (MPR) [98] nodes to flood the network in an

efficient way by reducing duplicate packets in the same region. The protocol also selects bi-directional links for the purpose of routing, so that the problem of packet transfer over uni-directional links is avoided.

4.2.2 Reactive (on-demand) routing algorithms

4.2.2.1 Ad-hoc On-Demand Vector Routing (AODV)

AODV [99] is a reactive routing protocol which discovers routes only when there is a need to route a packet to some destination nodes. Any nodes transmitting a packet to some destination node that is more than one hop away would first check if it has a route in its route table to that destination. If it does not, then route discovery is performed. The node would broadcast a route request packet (RREQ) with a unique 'broadcast and node' identifier. Any node receiving the RREQ will remember which node sent the first unique RREQ and stores each RREQ as an entry into a RREQ table. Any duplications of the RREQ are discarded. RREQ is relayed through the network until it reaches the destination or some intermediate node with a route to the destination node. A route reply packet (RREP) would then be sent back using the reverse route via the RREQ tables of the intermediate nodes. The advantage of AODV is that, being an 'on-demand' algorithm, no redundant routes would be stored or discovered, thus lowering network traffic and memory requirements in the nodes. However, there is an additional memory requirement to store the RREQ table at each node. The RREQ table would be larger on those nodes that have critical links to many nodes in a hotspot.

4.2.2.2 Dynamic Source Routing (DSR)

DSR [100] is a reactive routing algorithms similar to AODV which broadcast a route request packet (RRQ) whenever there is a need to route a packet to some destination node. There is a route record in each RRQ which stores the list of nodes the RRQ packet has transversed. Any nodes that had received the RRQ will check if it is in the RRQ's route record. If it does not exist in the record, then the node will append its node identifier into the RRQ table and relay the message to all its neighbours. Nodes, when receiving a RRQ with its node identifier in the RRQ record would discard the

RRQ packet as it would have received the packet before. On reaching a destination route or an intermediate node with the destination in its route cache, a route reply packet (RRP) is sent by the intermediate or destination node to the source node using the RRQ's route record. The method in which DSR discovers a route is similar to that of AODV, except that instead of using RREQ tables in each node during route discovery, a RRQ record is stored in the route request packet. This eliminates the need for an additional table on each node. However, the memory requirements for RRP still exist especially in large networks. As the RRP could grow potentially huge, large packet buffers may still be needed at each node. Unlike AODV, where the source node does not have the knowledge of the complete route to destination and just knows which node to forward the message to, source nodes in DSR stores these complete route information in its cache route.

4.2.2.3 Other reactive routing algorithms

Other commonly used reactive routing protocols include Temporally Ordered Routing Algorithm (TORA) [101] and Associativity Based Routing (ABR) [102]. TORA is a highly adaptive loop-free distributed routing algorithm based on the concept of link reversal. TORA is designed to react efficiently to topological changes and to deal with network partitions. However, TORA assumes all nodes have synchronised clocks, and is therefore unsuitable for Specknets. ABR is a routing protocol that is free from loops, deadlocks and packet duplicates. It defines a routing metric for route selection known as the degree of association stability. The key idea is the use of the longevity of routes, instead of the shortest length of the routes, as the main selection criterion. Therefore, longer-lived routes are preferred over shorter-lived ones. Longevity of a node can easily be estimated by the remaining power, signal strength and node speed.

4.2.3 Hybrid (Proactive and Reactive) routing algorithms

Hybrid routing algorithms such as the Zone Routing Protocol (ZRP) [103] are designed to capitalise on the advantages of proactive and reactive routing by using both of them simultaneously in the network. Using ZRP, each node maintains an *n*-

hop table, where n is the radius of the zone, where paths between the nodes and their neighbours within n hops are stored. To route within this zone, the node uses a proactive approach such as DSDV. Thus, the routing table is maintained consistently by using limited-hop broadcast. For other destination nodes, border-casting is used when reactively looking for a route. Nodes that are exactly n-hop away from the source are known as the border nodes and are responsible for relaying messages out of the zone. Reactive routing is then used to find a route to the destination node. The use of both proactive and reactive approaches enables ZRP to limit the latency for close destinations and eliminate the need to maintain huge routing tables for more distant ones. Memory used for proactive routing can usually be limited by the radius of the zone. In the case of Specks, density information of the targeted application can be utilised to estimate the radius of the zone. Out of the zone, broadcasting of route request packets would have to be used to enable reactive routing. Query control schemes [104, 105] has also been incorporated into ZRP to limit the rebroadcasts required in the network to minimise the routing overheads.

4.2.4 Data-centric routing algorithms

Directed diffusion [106] is a data-centric routing algorithm which is typical of ones that have been adapted for sensor networks. This approach is very similar to routing algorithms [107-109] based on ant-colony optimisation [110]. Data generated by sensor nodes is tagged as attribute name-value pairs. A task in a 'sink' node requests data by sending interest for the named data in the form of an interval and duration field. The interval field relates to the frequency at which sensor information should be transmitted back to the sink, and the duration field is the length of time for which the interest is valid. A task requiring some sensor data would periodically broadcast an interest message for the data, with the initial interest message having a much longer interval attribute. Each node maintains an interest cache and each item in it corresponds to a distinct interest received from some nodes. It also contains a gradient towards the node that transmits the interest to it where the node may or may not be the originator of the interest. Gradients are reinforced by decreasing the interval of an interest, thus increasing the data rate on reinforced links. Likewise,

negative reinforcement would be carried out either by not reinforcing a gradient, whereby it would leave the system once its duration expires, or by forcing a higher interval on the interest. As routing tables are looked up by interests instead of node addresses, direct diffusion would suit routing in sensor networks where the sensed data is the source for all interactions, and where data items can easily be quantified and identified.

4.2.5 Location–based routing algorithms

Location-based routing algorithms make use of position information of the nodes in a MANET to assist in routing. Examples of such routing algorithms include the Global Positioning Systems (GPS) Zone Routing Protocol (GZRP) [111], Greedy Perimeter Stateless Routing (GPSR) [112], Distance Routing Effect Algorithm for Mobility (DREAM) [113], Location-Aided Routing (LAR) [114] and Fisheye State Routing (FSR) [115].

GZRP is an extension to ZRP described in Section 4.2.3 which uses the position information of the destination to optimise the route query of the inter-zone protocol in order to reduce flooding overheads. GPSR uses greedy forwarding when decisions are informed by location information of the router's immediate neighbours and the destination node. The advantage of GPSR is that no routing tables are required; the stateless nature of the algorithm is attractive as the memory requirement is quite low. DREAM is a type of directional flooding scheme that uses position information of the destination node to calculate the expected region of the destination and only the direction towards that region is flooded with data. LAR uses an estimation of the position of the destination node to increase the efficiency of the discovery procedure. Only a subset of nodes is queried during the discovery phase, specifically those in the so-called 'request zone' near the estimated position. LAR uses standard flooding as a last resort when no estimations are available. FSR looks to maintaining topological information at each node, as traditional link state routing generates significant overheads in maintaining the topology information at each node. In FSR, the update message does not contain information about all nodes. Instead, it includes more information about closer nodes, and less information about further nodes in its update

message. Therefore, the information about the near neighbours of each node is more accurate, and this falls away with distance from the node. As a result, the route gets optimal as it converges towards its destination.

4.2.6 Comparison of routing protocols for Specknets

The drawback of employing location-based routing algorithms in Specknets are principally twofold: given the small size of Specks, the location is required to be resolved to a fine granularity (order of a few centimetres for small networks), which is difficult to achieve in resource-constrained Specks; and, the requirement to avoid infrastructure for positioning such as beacons and GPS. Several algorithms [116-118] have been suggested to determine relative or logical location, usually using triangulation or graph prediction, but these approaches are computationally expensive and requires significant communication overheads. Furthermore, location-based routing algorithms impose a memory premium to store location information about the destination and neighbouring nodes. The data-centric routing algorithms are appropriate for sensor networks; however, for re-programmable computational networks such as Specknets, the quantification and identification of data items or interest would relatively complex due to the sheer number of data item types and the flexibility which is required. Proactive routing algorithms are also unsuitable for Specknets as these protocols do not scale easily as routes between any two nodes in the network have to be maintained constantly. This would impose unnecessary communication and storage overheads for maintaining routes that may never be used. Hybrid routing protocols may be more suitable for Specknets as they reduce the overheads for maintaining and storing redundant route information by providing a compromise between the reactive and proactive approaches. However, additional complexity and algorithms would be required by hybrid routing protocols to determine the appropriate routing approach to be used based under different network conditions. In contrast, reactive routing algorithms are apt for routing in Specknets. Routes will only be created when needed, thus keeping redundant route information to a minimum. However, the underlying MAC algorithm must be able to support the variable traffic requirements of reactive routing protocols for route discovery,

maintenance and repair. In this chapter, commonly used reactive MANET protocols, such as DSR and AODV, were used to determine whether the SpeckMAC algorithms are indeed able to support reactive routing algorithms in an energy efficient manner by trading savings in energy against latency.

4.3 Implementation and validation of the network models

The MANET reactive routing protocols and the SpeckMAC algorithms were modelled in the Qualnet network simulator. Such an approach allows a number of different network scenarios, both static and dynamic, to be explored in the shortest time compared to physical implementation on the ProSpeckz. The network simulator allows one to explore different combination of parameters, especially for mobile scenarios, in a repeatable and scalable fashion. However, the limitations of using Qualnet should be recognised. Firstly, accurate short-range radio channel model were not available in the Qualnet distribution. It was therefore not possible for typical operations of Specknets to be simulated. Instead, longer communication ranges were used in the simulations and the simulation area was scaled accordingly to extrapolate a reasonable estimate of the performance of the SpeckMAC algorithms in supporting reactive MANET protocols in Specknets. IPv4 [119] was used as the transport layer protocol as it was the only addressing protocol available on Qualnet.

The physical layer of the original Qualnet distribution did not model the idle state of the radio, as these were assumed to be either in the transmission or the receiving modes. The physical layer of the simulator had to be modified accordingly to allow the state of the radio to be idle. Other relevant parameters, such as the idle/receive/transmit turn-around times, were included in the modified radio model to reflect the operations of the radio in the ProSpeckz hardware platform.

The status and timing information of the radio of the ProSpeckz was used as the benchmark to validate the fidelity of the Qualnet models against the operations of SpeckMAC on the ProSpeckz. In order to capture the operations of the ProSpeckz, two output pins were tapped to reflect the status of the radio and were recorded on a digital storage scope. A trace file in the simulator captured the different states and

timings of the simulated radio for comparisons. Figure 4-1 shows diagrams of the status and timing information captured from from the ProSpeckz and the simulator. Node B transmits a packet to Node A using SpeckMAC-B. With reference to the figure, the following sequence of events was observed:

(a) In both environments, Node B turns on the radio to perform carrier sense for CSMA.

(b) Node A in the ProSpeckz implementation (real environment) performs channel sampling and receives the wakeup frame packet from Node B before returning to sleep.

(c) Node A in the simulated environment performs channel sampling and receives the wakeup frame packet from Node B, before returning to sleep.

(d) In both environments, Node A turns on its radio in anticipation of the data frame based on the timing information received in the wakeup frame.

(e) In both environments, Node B transmits the data frame, which is received by Node A, after which, both nodes resumes periodic channel sampling.

In the case of SpeckMAC-B, the timing diagrams confirm that the simulated operations of the radio at the physical layer corresponded very closely to the operation of the ProSpeckz. Figure 4-2 shows diagrams of the status and timing information captured for the SpeckMAC-D algorithm for the same scenario as in Figure 4-1. With reference to the figure, the following sequence of events was observed:

(a) In both environments, Node B turns on the radio to perform carrier sense for CSMA.

(b) Node A in the ProSpeckz implementation (real environment) performs channel sampling and detects a busy channel. The radio receiver therefore remains turned on.

(c) Node A in the real environment received the data frame transmitted by node B and resumes periodic channel sampling after performing a back-off based on the timing information embedded in the received data frame.

(d) Node A in the simulated environment performs channel sampling and detects a busy channel. The radio receiver therefore remains turned on.

(e) Node A in the simulated environment receives the data frame transmitted by node B and resumes periodic channel sampling after performing a back-off based on the timing information embedded in the received data frame.

(f) In both environments, Node B completes all re-transmissions of the data frame and resumes periodic channel sampling.

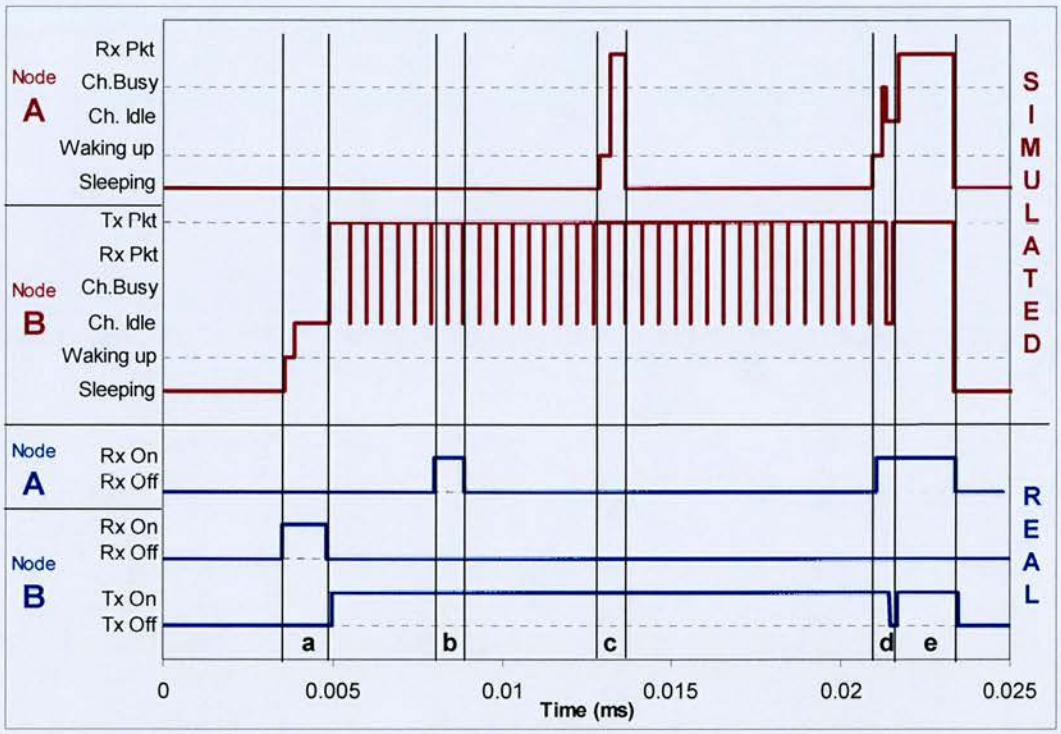


Figure 4-1: A comparison of the output of the simulator with the operations performed by the ProSpeckz for SpeckMAC-B

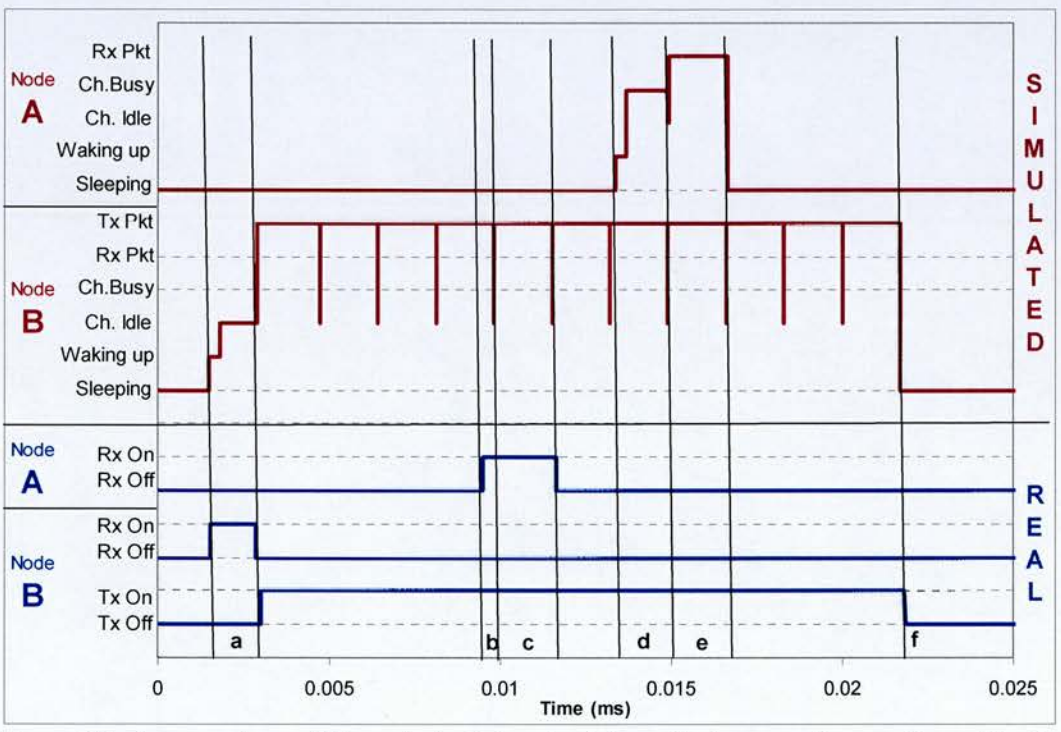


Figure 4-2: A comparison of the output of the simulator with the operations performed by the ProSpeckz for SpeckMAC-D

To further verify the output of the simulator, the experimental scenarios described in Section 3.6.1 were modelled in the simulator and the simulation result were compared against the results obtained from the experimental results, as well as with the results calculated using the mathematical models presented in Section 3.6. The comparison is shown in Figure 4-3 and Figure 4-4, for SpeckMAC-B and SpeckMAC-D respectively. The simulated turn-on times for the receiver and transmitter matched closely to the results obtained from the physical experiments and with the mathematical models. This increased the confidence that the simulator models the radio and MAC operations of the ProSpeckz accurately.

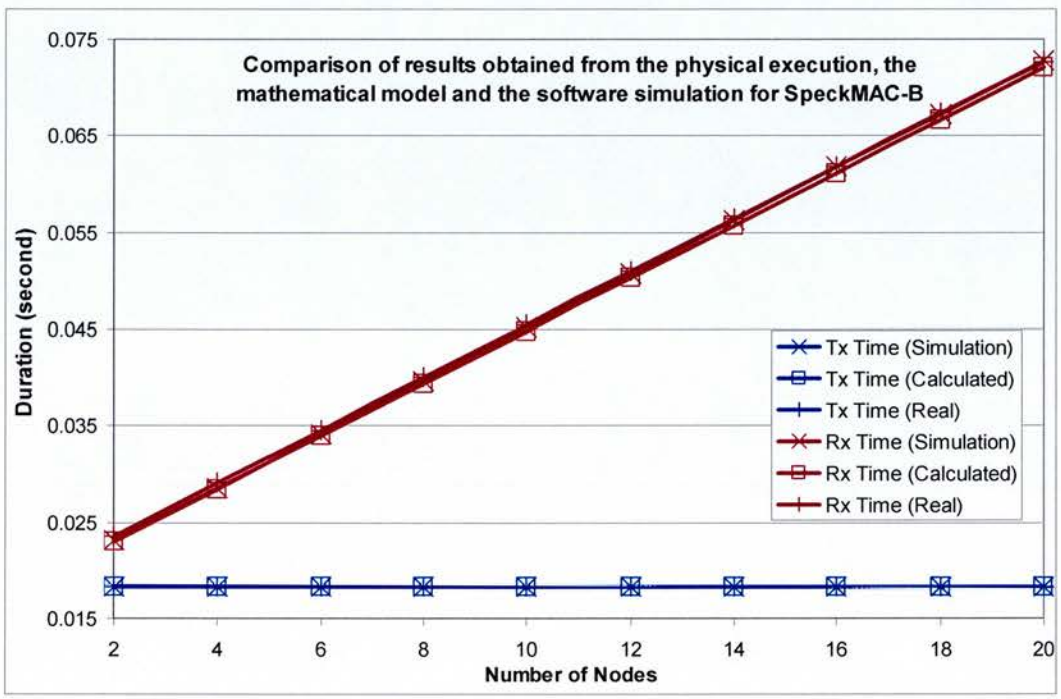


Figure 4-3: The comparison of the results obtained from the simulation, the mathematical model and the actual results measured on the ProSpeckz when nodes were utilising SpeckMAC-B

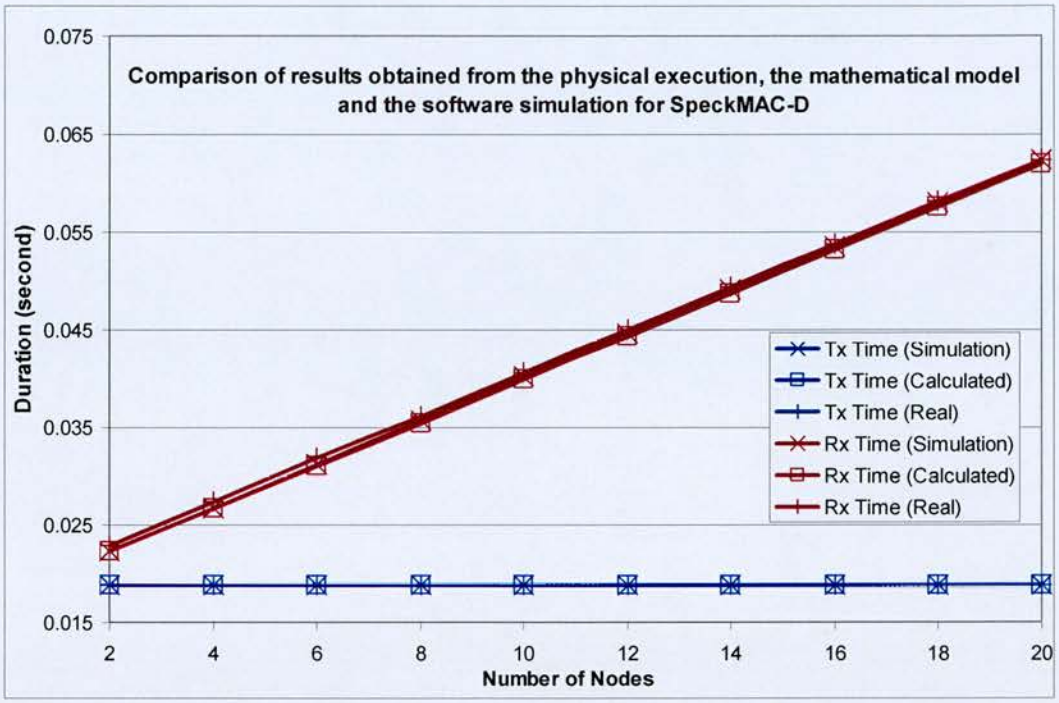


Figure 4-4: The comparison of the results obtained from the simulation, the mathematical model and the actual results measured on the ProSpeckz when nodes were utilising SpeckMAC-B

4.4 Performance of DSR and AODV running on SpeckMAC

Once the Qualnet models were verified, the simulator was used to measure and compare the performance of reactive MANET routing algorithms for the SpeckMAC target. CSMA was used as the benchmark for comparison, as a non-energy saving MAC algorithm, with SpeckMAC. The DSR and AODV routing algorithms were simulated in a static and a mobile scenario. 121 nodes were placed in an evenly distributed 11 x 11 grid in a simulated area measuring 1000 x 1000m and each node with a radio range of 250m. The transmission rate supported by the radio was 250kbps, which was data-rate supported by the ProSpeckz. Each scenario was simulated for 120 seconds. Constant bit-rate (CBR) server and clients were used to receive and transmit CBR packets, respectively, with all the CBR clients starting to transmit packets after 10 simulated seconds at a rate of one packet per second. The CBR clients would stop transmission after 110 simulated seconds. For all SpeckMAC protocols, a $T_{Interval}$ of 15ms was selected. The power consumption for

each node was simulated based on the parameters in Table 3-2, with nodes transmitting at 0dBm.

4.4.1 The Static Scenario

Figure 4-5 depicts the static network scenario simulated in Qualnet. Three CBR clients and a CBR server were place in the corners and the arrows indicate the data flow connections between them. The effects of different numbers of routes in the network was studied for one (A), two (B,C) and three (A,B,C) connections. The performance under different traffic loads were investigated for packet sizes from 16 to 256 bytes, in power of two. For each packet size, the scenario was simulated 80 times and the simulation results averaged.

The performance metrics studied by the simulation were the first packet arrival time, the end-to-end latency, the peak queue size, the packet delivery ratio and the energy consumption.

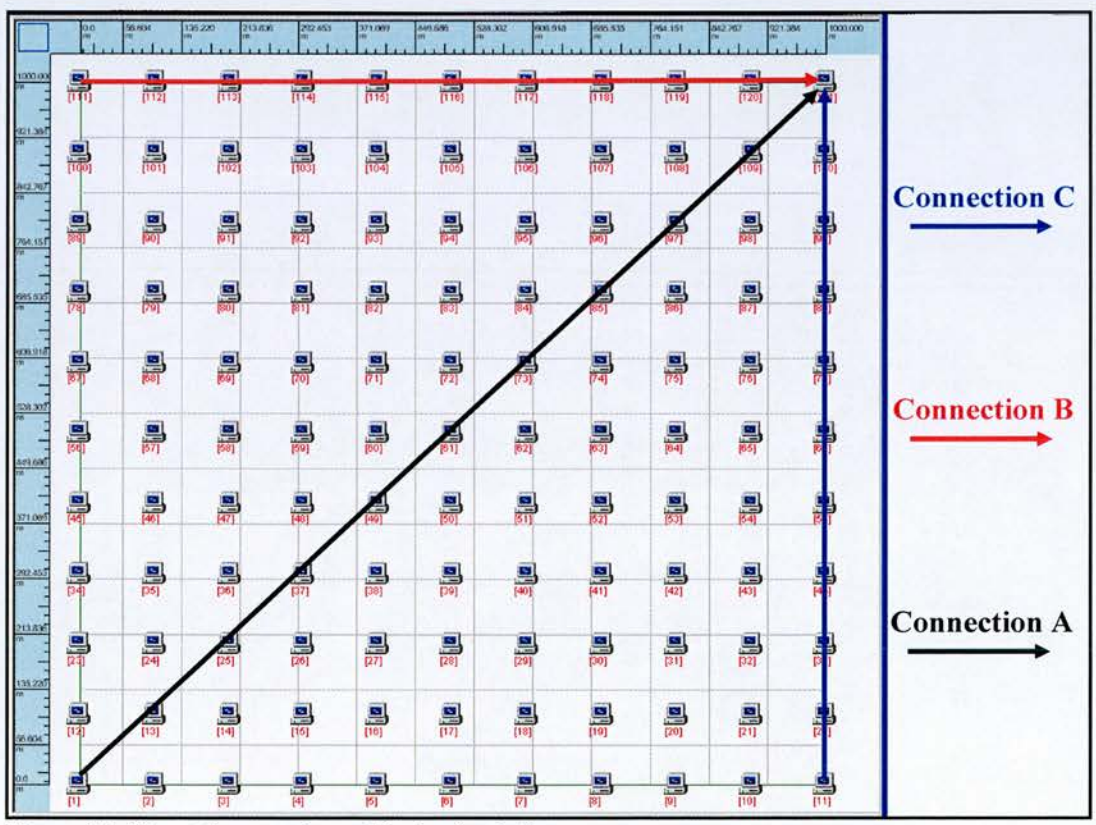


Figure 4-5: The static scenario used in the simulation

4.4.1.1 First packet arrival time

The first packet arrival time measurements, shown in Figure 4-6, indicate when the CBR server first receives a packet from a CBR client, which is a measure of the inherent delay in route discovery. CSMA has the lowest first packet arrival time, due to the radio being turned on all the time and therefore does not incur any delays due to the duty-cycle of the radio. SpeckMAC-D demonstrated an earlier first packet arrival time than SpeckMAC-B for each of the routing algorithm simulated. This conformed to the analysis presented in Chapter 3 as nodes using SpeckMAC-D exhibited lower latency than SpeckMAC-B, resulting in a shorter route discovery time. Furthermore, for each of the MAC algorithms, nodes running the DSR routing algorithm received the first packet earlier than AODV.

4.4.1.2 End-to-end delay

The end-to-end delay is the time elapsed between a CBR packet being placed in the transmitter buffer of a CBR client, and it being placed in the buffer of the receive buffer of the CBR server. Therefore, the end-to-end measurements shown in Figure 4-7 include delays due to multi-hop latencies. For each of the routing algorithm simulated, CSMA exhibited the shortest end-to-end delays and nodes using SpeckMAC-D outperformed nodes running SpeckMAC-B, as described in Chapter 3. The graphs also indicate that, for each MAC algorithm, the DSR routing algorithm exhibited shorter end-to-end delays compared to AODV.

4.4.1.3 Peak Queue Length

The queue length is the number of storage bytes in a unified transmitter/receiver buffer. The peak queue length for each simulation is the maximum number of bytes used by any node across the network at any time. Results in Figure 4-8 were obtained by averaging the peak queue length results across all the iterations for each simulation scenario. For each routing algorithm, CSMA cases have the smallest peak queue length, whereas SpeckMAC-B cases have the largest peak queue length. For each MAC algorithm, cases with AODV routing algorithm generally have a smaller peak queue length than cases with DSR routing algorithm for small CBR packet sizes.

However, for larger packet sizes, DSR cases gradually demonstrated shorter peak queue length than those utilising AODV. Finally, in the worst case, the size of the buffer was up to 4000 bytes for supporting AODV using SpeckMAC-B.

4.4.1.4 Delivery ratio

Delivery ratio is defined as the percentage of CBR packets received by the CBR server over the total number of packets sent by the CBR clients. Figure 4-9 shows the averaged delivery ratio across the three CBR connections. Across all packet sizes and routing algorithms, CSMA exhibited the highest delivery ratio. This if explained by the fact that nodes utilising CSMA have the largest bandwidth as the radio receiver are always turned on, and there is no additional transmission overheads for waking up neighbouring nodes. SpeckMAC-D demonstrated higher delivery ratios than SpeckMAC-B, which conforms to the analysis in Section 3.7.2 as SpeckMAC-D transmits each data packet multiple times which provides data redundancy and better robustness against channel noise. Across all MAC algorithms, DSR demonstrated higher delivery ratios than AODV for the cases of either one or two connections in the network. However, for larger number of connections, AODV achieve higher delivery ratios than DSR, as demonstrated in the scenario with three connections.

4.4.1.5 Energy Consumption

Figure 4-10 shows the percentage of energy consumed by nodes utilising the SpeckMAC algorithms over the energy consumed by nodes utilising CSMA. Nodes utilising either SpeckMAC algorithm managed to significantly reduce the amount of energy consumed. When three connections were used with packet sizes of 256 bytes, nodes utilising the SpeckMAC protocols managed to consume at most only 15% of the energy consumed by nodes utilising CSMA. Furthermore, for each routing algotihm simulated, SpeckMAC-B nodes generally consumed less energy than SpeckMAC-D nodes as SpeckMAC-D suffers from more energy wastage under uni-cast traffic conditions as mentioned in section 3.7.3, and that the energy wasted increases with the increase in packet size. The SpeckMAC-B nodes did not consume

as much energy as SpeckMAC-D nodes as wakeup frames were used decide if the radio should be turned on to receive the data frame.

4.4.1.6 Energy Efficiency

In order to compare the energy efficiency of the SpeckMAC algorithms, the delivery ratio and energy consumption measurements have to be considered, not in isolation, but together, as the average energy consumed per node for each CBR byte successfully received by the CDR server as shown in Figure 4-11. It can be seen that nodes using the SpeckMAC algorithms were considerably more energy-efficient than CSMA, which demonstrated the ability of the energy-efficient SpeckMAC protocols to support reactive MANET routing algorithms at the networking layer. Figure 4-12 shows the percentage of energy consumed per byte which is a clearer comparison of improvements in energy efficiencies for SpeckMAC and the CSMA protocols. SpeckMAC nodes consumed at most 16% of the energy consumed per byte compared to CSMA ones. Furthermore, for each routing algorithm simulated, SpeckMAC-D nodes were more energy-efficient than SpeckMAC-B ones for smaller sized packets as greater number of retransmissions of the data frames lead to better delivery ratios. This ability to transmit packets with greater success outweighs the energy saved by SpeckMAC-B nodes when the packet sizes are small. However, this is reversed as the packet sizes increases SpeckMAC-D nodes due to overhearing.

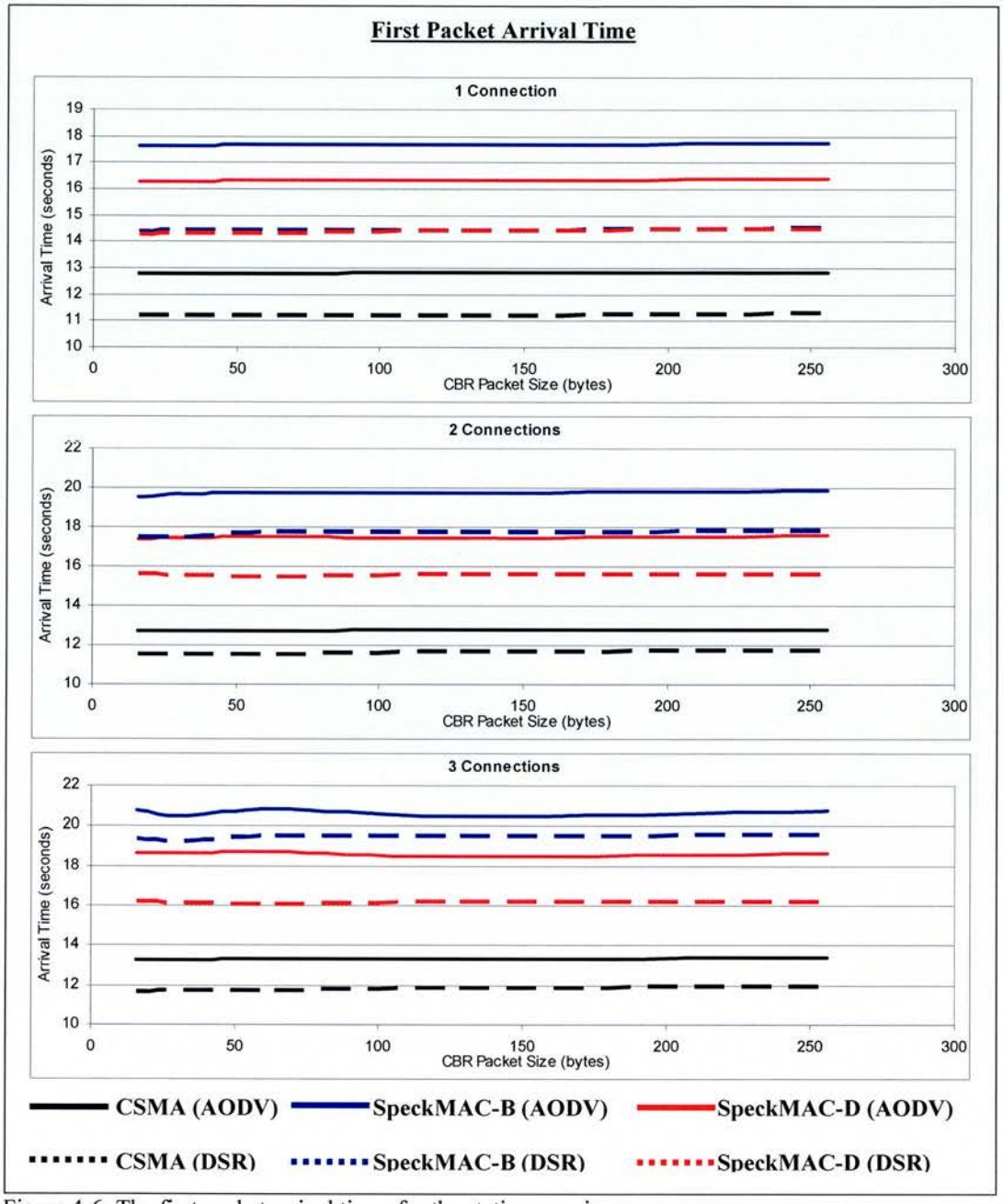


Figure 4-6: The first packet arrival times for the static scenario

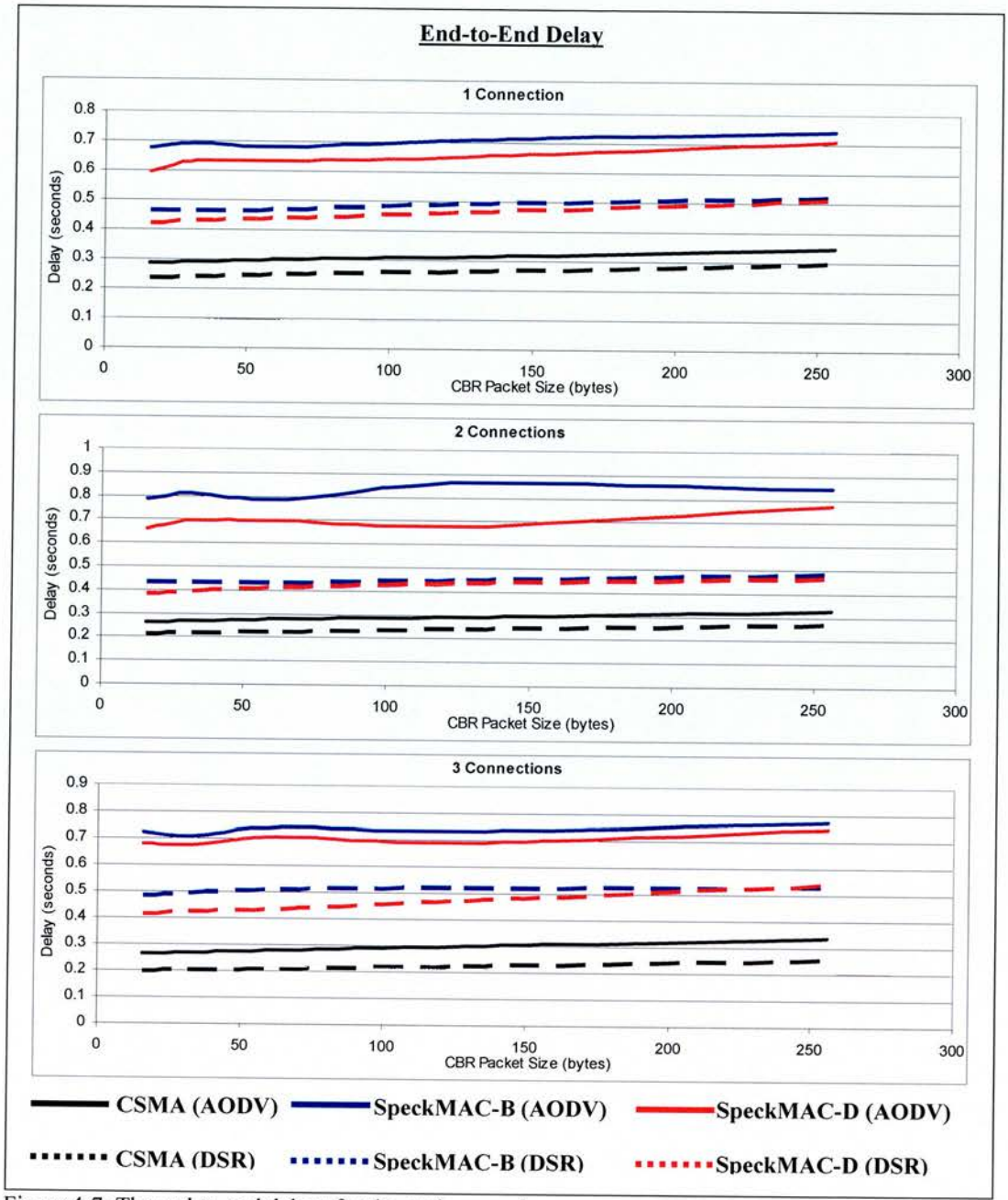


Figure 4-7: The end-to-end delays for the static scenario

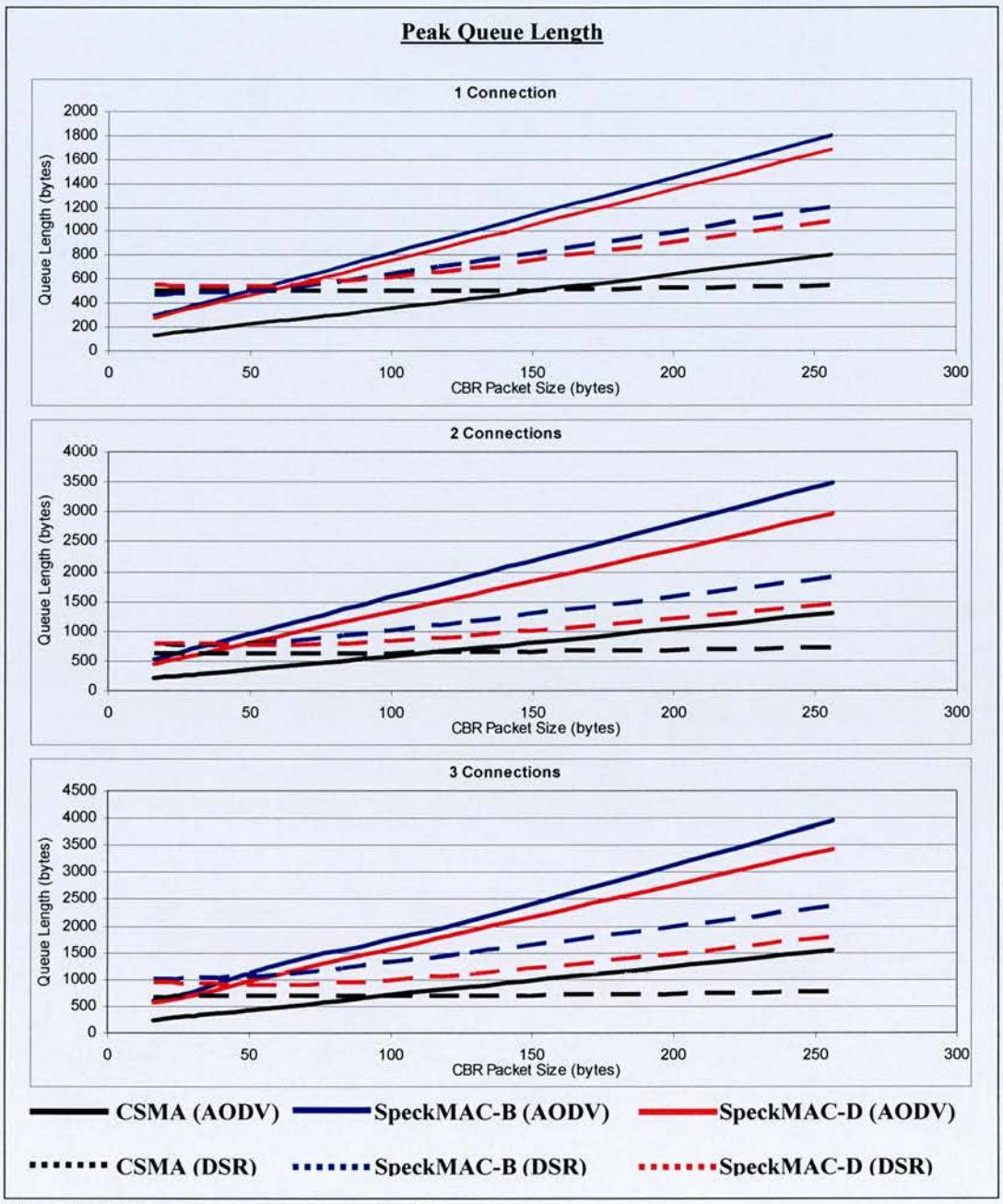


Figure 4-8: The peak queue length for the static scenario

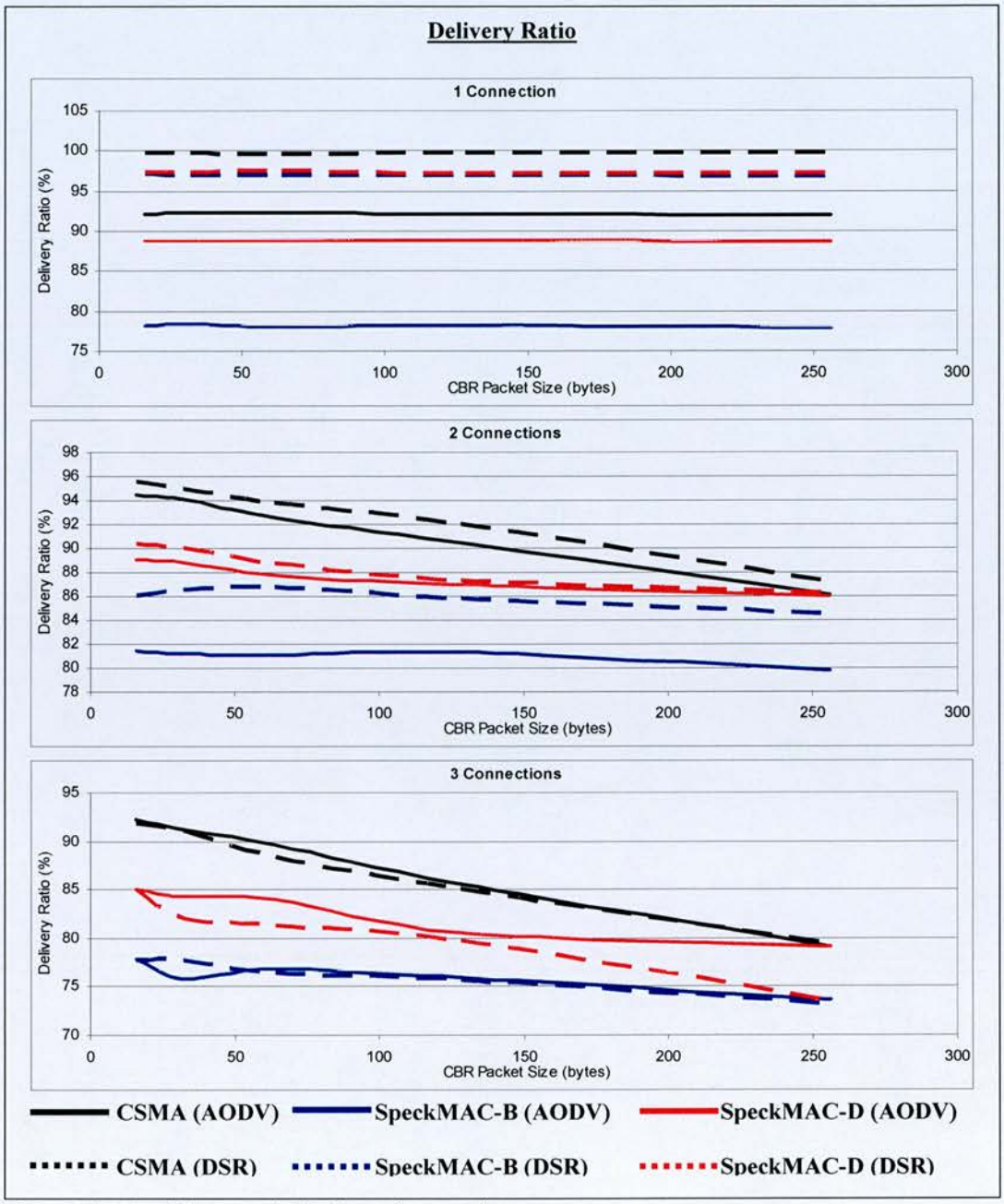


Figure 4-9: The delivery ratio for the static scenario

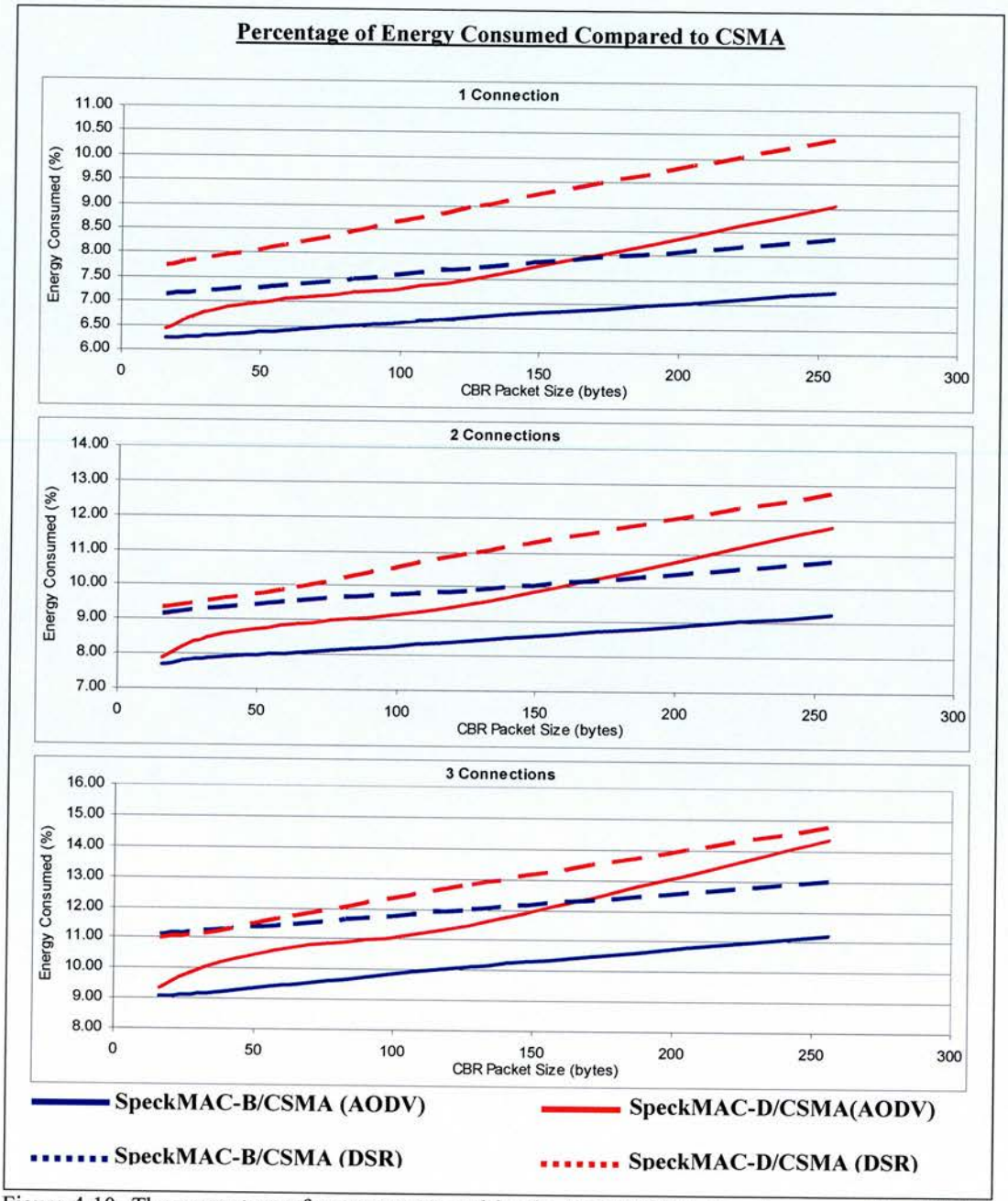


Figure 4-10: The percentage of energy consumed by the SpeckMAC protocols compared to CSMA for the static scenario

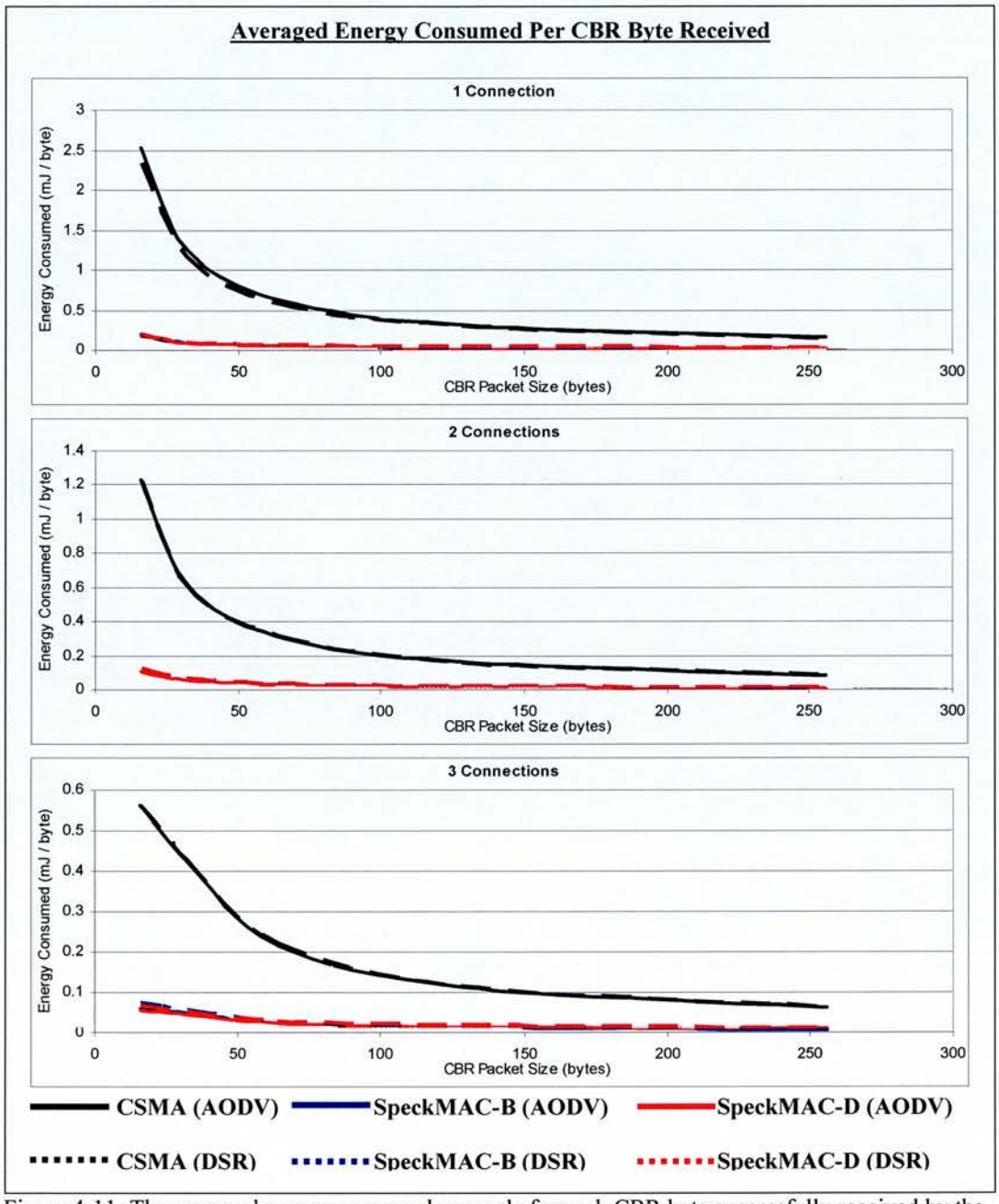


Figure 4-11: The averaged energy consumed per node for each CBR byte successfully received by the CBR server for the static scenario

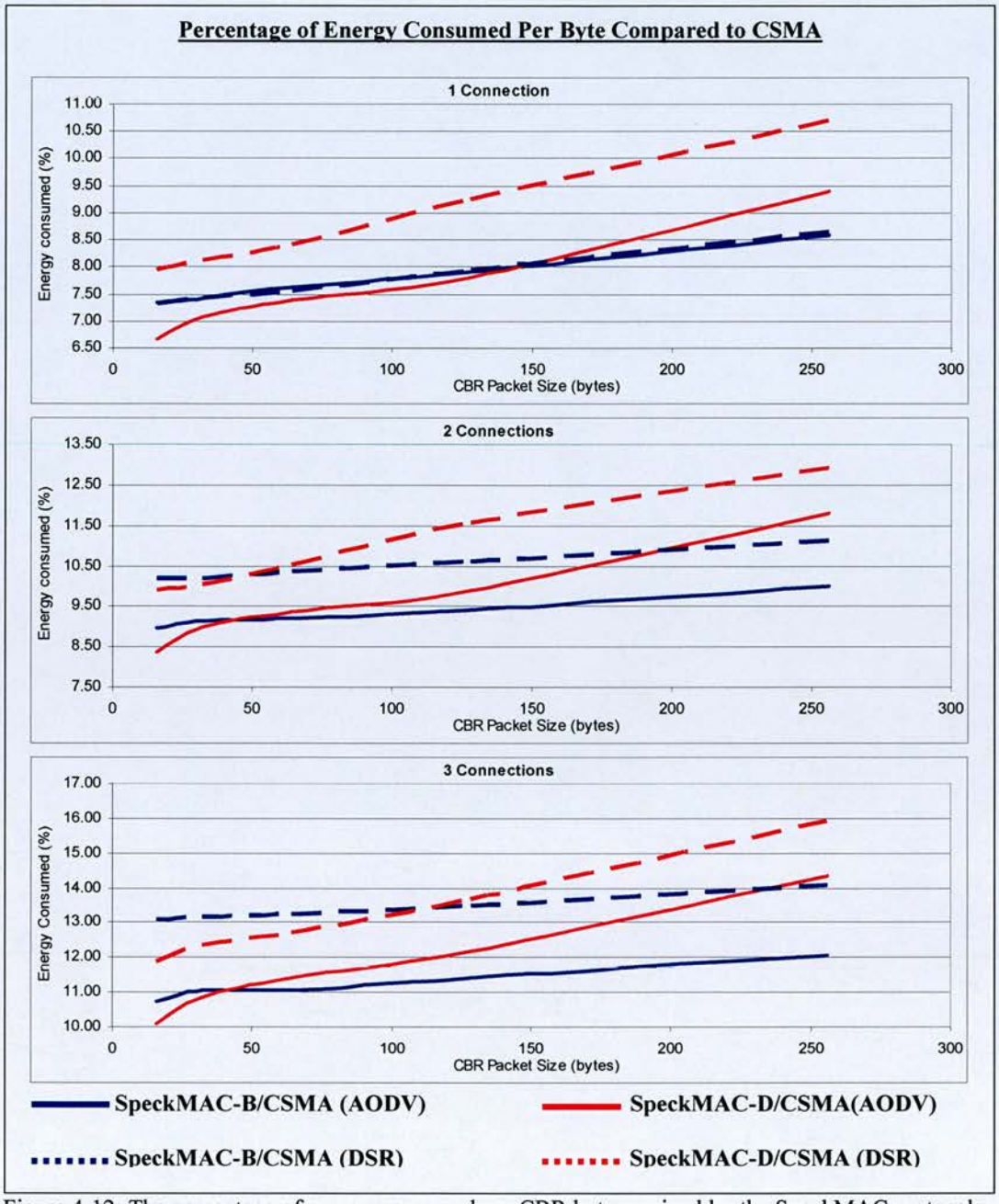


Figure 4-12: The percentage of energy consumed per CBR byte received by the SpeckMAC protocols compared to CSMA for the static scenario

4.4.2 The Mobile Scenario

Figure 4-13 depicts the mobile network simulation scenario for comparing the performances of the SpeckMAC protocols

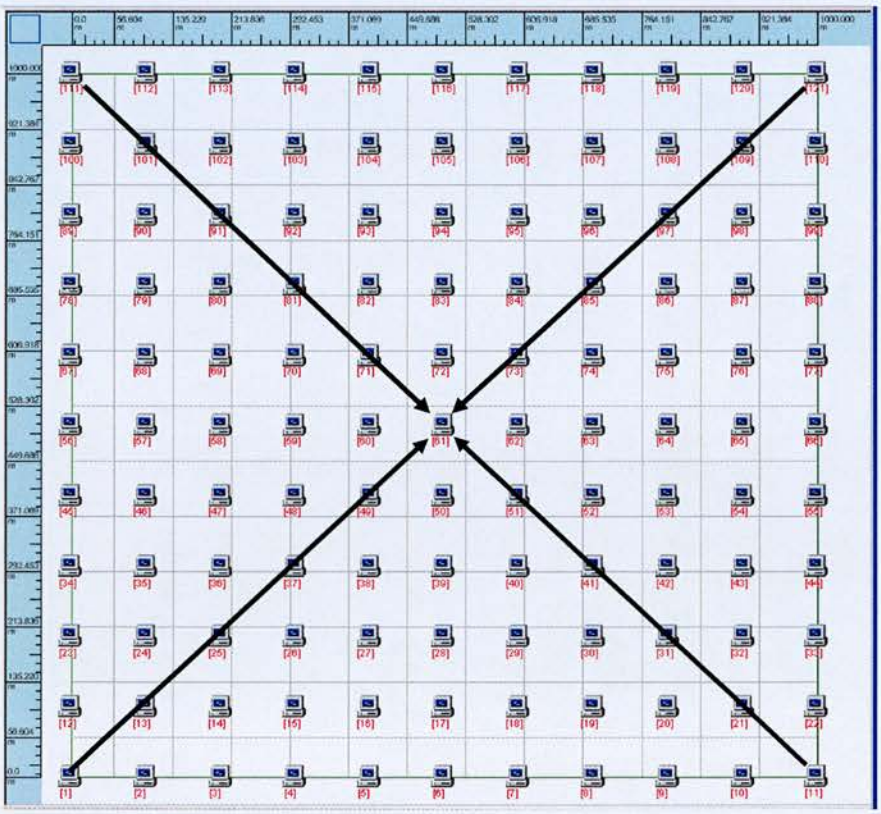


Figure 4-13: The mobile scenario used in the simulation

Four connections were simulated by four CBR clients placed in the four corners and one CBR server in the centre of the network. Arrows in Figure 4-13 indicate data communications transmitted as 64-byte sized packets from the CBR clients to the server. All the nodes in the network were static except for the mobile CBR server. This mimicked a scenario of a mobile server roaming in a network of fixed Specks with data being transferred from some of the Specks to the server. The movement model of the CBR server is as follows: moves towards a randomly chosen waypoint at a random speed, where it stops for one simulated second, and so on. For each set of simulations, the server was limited to settings of maximum speeds ranging from 2m/s to 10m/s in steps of 2m/s. For each setting of maximum speed, the experimental

scenario was simulated 150 times, and the simulation results were averaged across all iterations.

As in Section 4.4.1, the main performance indicators for the simulations of the SpeckMAC protocols were latency, packet queue length, delivery ratio and energy consumption. Results in Figure 4-14 shows that CSMA nodes exhibit the earliest first packet arrival times for a range of maximum speeds, as observed in the static case in Section 4.4.1.For the AODV routing algorithm, both SpeckMAC algorithms showed similar first packet arrival times. However, in the case of the DSR routing algorithm, SpeckMAC-D nodes demonstrated an earlier first packet arrival time the nodes SpeckMAC-B.

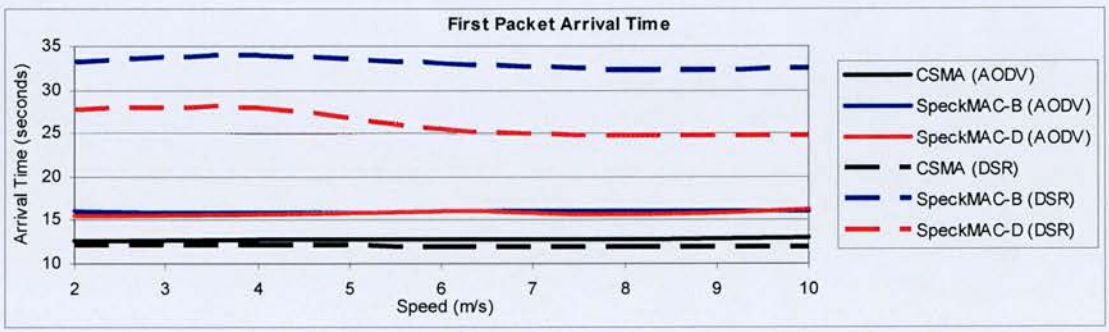


Figure 4-14: The first packet arrival times for the mobile scenario

For the end-to-end delay measurements, Figure 4-15 shows the results obtained from the simulation. CSMA nodes achieved the lowest end-to-end delays, regardless of the routing algorithm used, as the radios on these nodes were not duty-cycled. When similar routing algorithms were used, nodes utilising either SpeckMAC algorithms demonstrated similar end-to-end delay performances. It was observed that AODV nodes had much lower end-to-end delays than DSR ones, irrespective of the SpeckMAC protocols used.

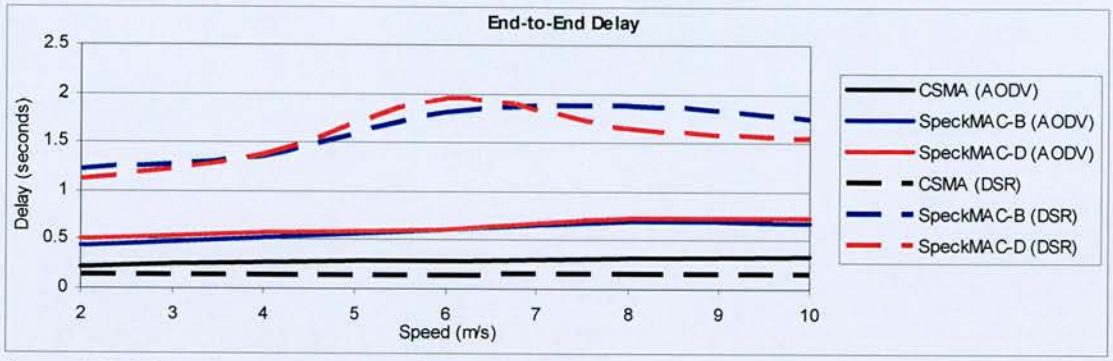


Figure 4-15: The end-to-end delays for the mobile scenario

The requirement for peak queue lengths, as shown in Figure 4-16, demonstrated longer ones for DSR nodes (16Kbytes in the worst case) compared to AODV ones (0.6Kbytes in the worst case). These results were quite different from the results obtained from the simulations based on the static scenarios where nodes running DSR generally used less memory for its queue. Furthermore, SpeckMAC-B nodes used more than twice as much memory for its queues than SpeckMAC-D ones.

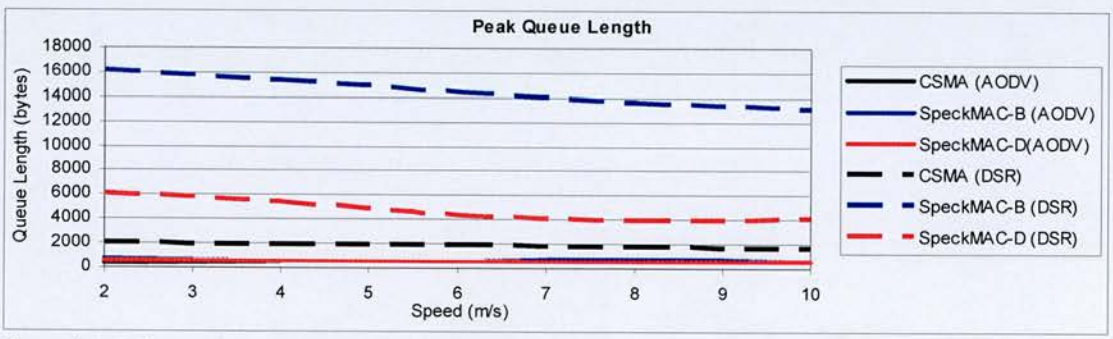


Figure 4-16: The peak queue length for the mobile scenario

In Figure 4-17, CSMA nodes exhibited the highest delivery ratio regardless of the routing protocols. SpeckMAC-D nodes demonstrated a higher delivery ratio than SpeckMAC-B ones for a given routing protocol, which is consistent with the results in the static scenario. However, unlike the static case, AODV demonstrated higher delivery ratio than DSR for the same MAC protocols.

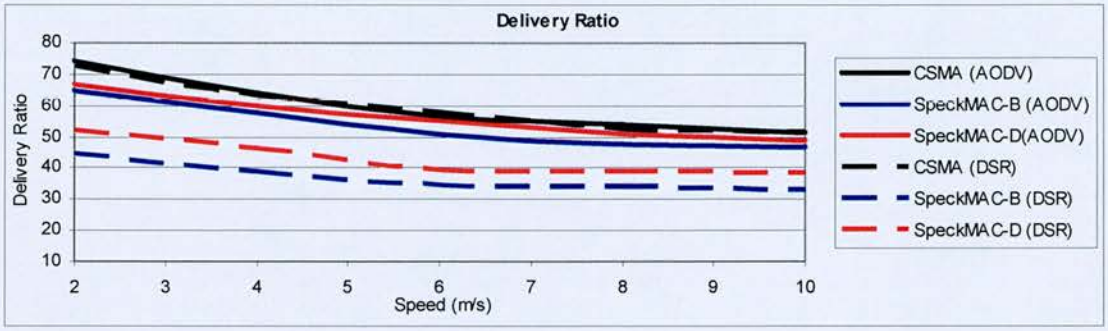


Figure 4-17: The delivery ratio for the mobile scenario

Comparisons of the SpeckMAC protocols in terms of percentage of total energy consumed by each node relative CSMA (Figure 4-18) yielded contrasting results: for AODV, SpeckMAC-D nodes consumed more energy than SpeckMAC-B whereas the reverse was true for DSR. In the worst case, SpeckMAC nodes, on average, consumed less than 16% energy of that consumed by CSMA nodes.

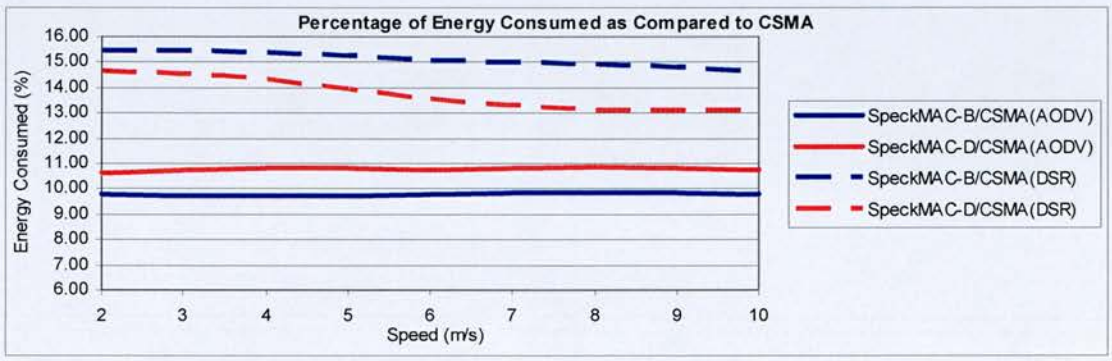


Figure 4-18: The percentage of energy consumed by the SpeckMAC protocols as compared to CSMA for the mobile scenario

From the results presented in Figure 4-19, the SpeckMAC protocols consumed at most 23% of the energy consumed per byte received by the CBR server as compared to CSMA. This result support the conclusion that for energy conscious mobile applications, the SpeckMAC protocols offer an energy-efficient MAC layer for at least two representative reactive MANET routing algorithms.

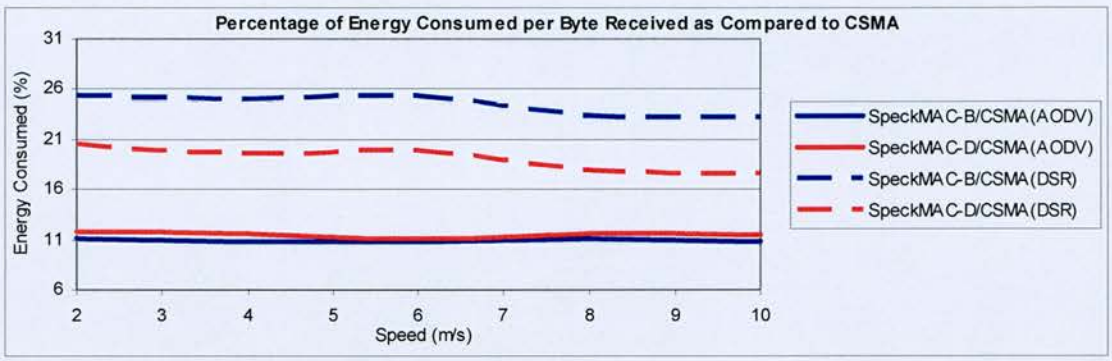


Figure 4-19: The percentage of energy consumed per CBR byte received by the SpeckMAC protocols as compared to CSMA for the mobile scenario

4.5 SpeckMAC-H: A Hybrid MAC Algorithm

Section 3.7 presented performance analysis using the ProSpeckz prototypes for the two SpeckMAC algorithms, demonstrating their respective strength and weaknesses for a range of metrics. SpeckMAC-D, when compared to SpeckMAC-B, is able to achieve higher delivery ratios due to redundant data frame transmissions, however, suffers from greater wastage of energy due to overhearing when unicast packets were transmitted as SpeckMAC-B was able to reduce overhearing using information extracted from the wakeup frames. SpeckMAC-H (SpeckMAC-Hybrid) is an attempt to combine the strengths of both SpeckMAC protocols in order to achieve better energy efficiency. This hybrid MAC protocol allows the transmitting node to decide locally the choice of algorithm for transmitting a data packet, without any prior coordination or handshaking with the receiver node. The latter uses the 'type' field in the frame header (Figure 3-15) to differentiate between the frame types. A simple heuristic for deciding the version of SpeckMAC could be based simply on the type of data packet to be transmitted: SpeckMAC-D for broadcast packets and SpeckMAC-B for unicast packet. Using the heuristic, SpeckMAC-H should achieve a higher packet delivery ratio than SpeckMAC-B whilst consuming less energy then SpeckMAC-D.

All three SpeckMAC protocols were next compared based on the scenario described in Section 4.4.1 for possible improvements in energy efficiency using SpeckMAC-H over the exclusive use of either protocol. Figure 4-20 shows the delivery ratio for different scenarios achieved by the three MAC protocols for the different CBR packet sizes. SpeckMAC-H had, as expected, achieved better results than

SpeckMAC-B across the range of parameters for the scenarios. This is the effect of the redundant retransmissions used for broadcasting routing packets, which results in more reliable route discovery, and thereby improving the delivery ratio.

Figure 4-21 shows the energy consumed by the radios for the different CBR packet sizes and scenarios. SpeckMAC-H nodes had, as expected, consumed less energy than SpeckMAC-D ones across all the scenarios and parameters for the respective routing algorithms. This is the effect of using the wakeup frames to minimise the energy consumed due to overhearing.

Figure 4-22 illustrates the improvements in energy-efficiency due to the hybrid SpeckMAC-H protocol, to the singular ones. It records the percentage of energy consumed per CBR byte received by the SpeckMAC-H nodes compared to the SpeckMAC-B and SpeckMAC-D nodes. SpeckMAC-H nodes were, in most cases, more energy-efficient than SpeckMAC-B ones. This is because SpeckMAC-H consumed similar amount of energy as SpeckMAC-B, while achieving higher delivery ratio due to redundant retransmissions of the data frames. SpeckMAC-H outperformed SpeckMAC-D as for larger packet sizes due to the increase of overhearing cost for the latter. However, for smaller packet sizes, the higher delivery ratio of SpeckMAC-D offsets the energy savings achieved by SpeckMAC-H.

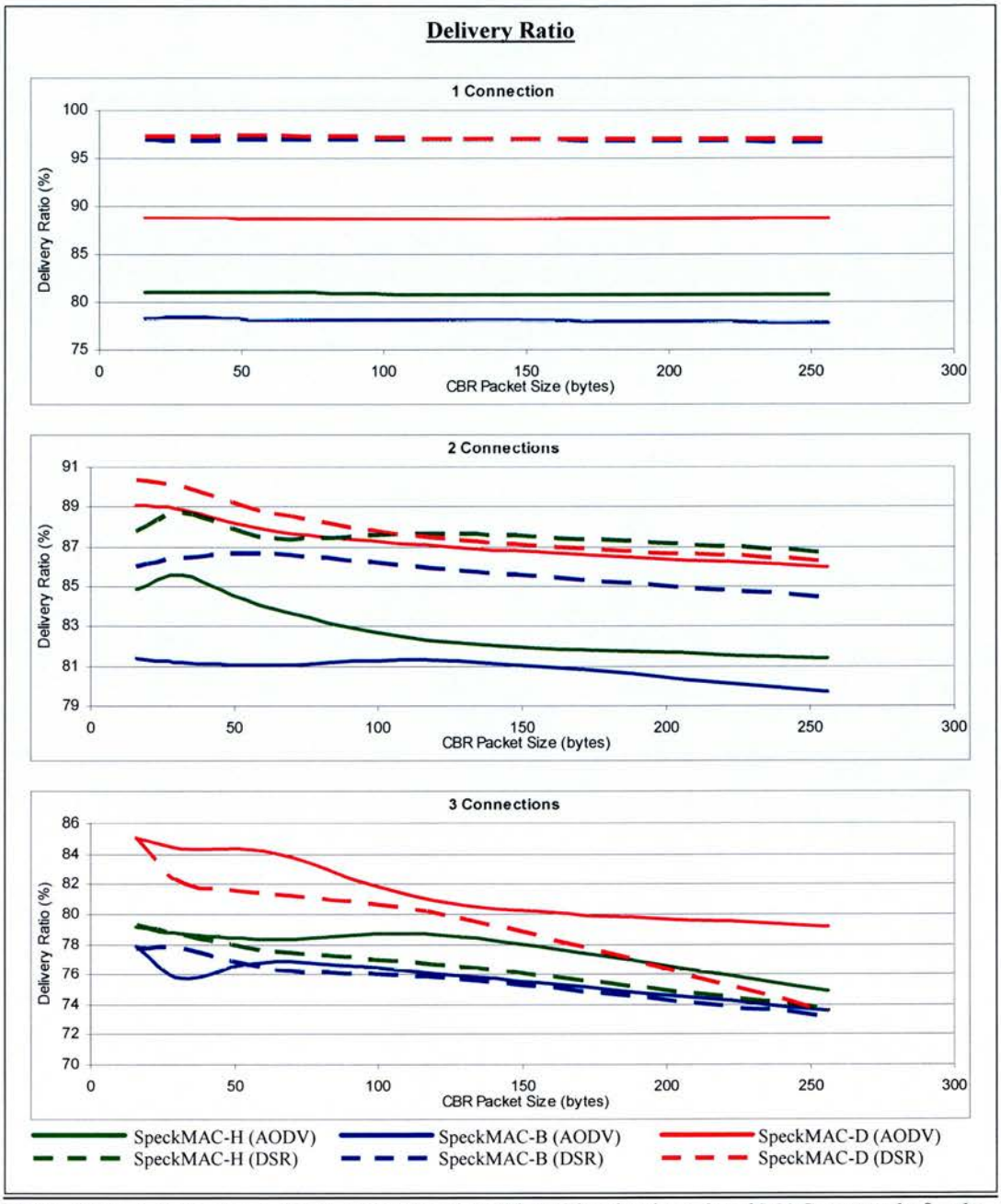


Figure 4-20: A comparison of the delivery ratio achieved by the three SpeckMAC protocols for the static scenario

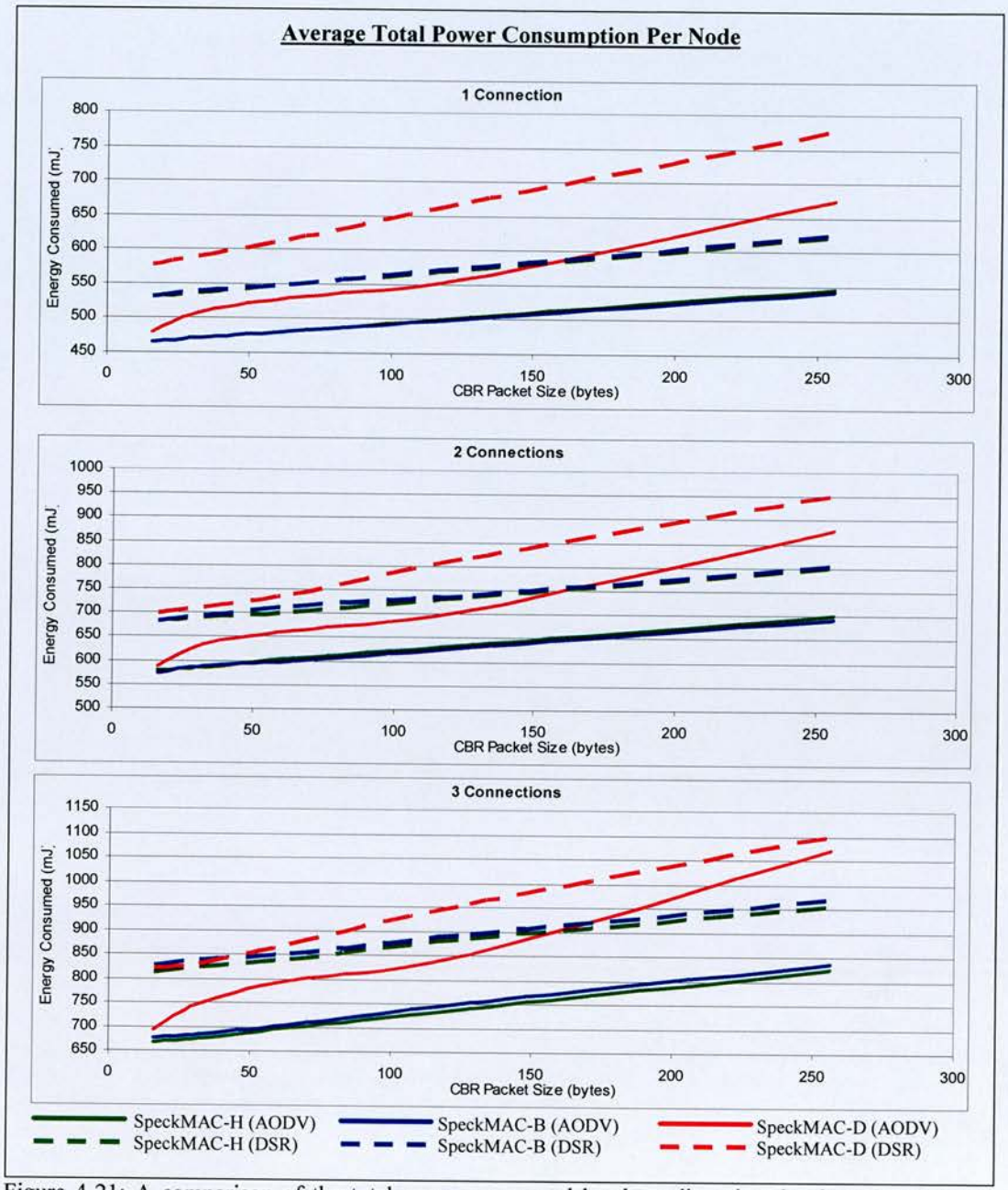


Figure 4-21: A comparison of the total energy consumed by the radio using the three SpeckMAC protocols for the static scenario

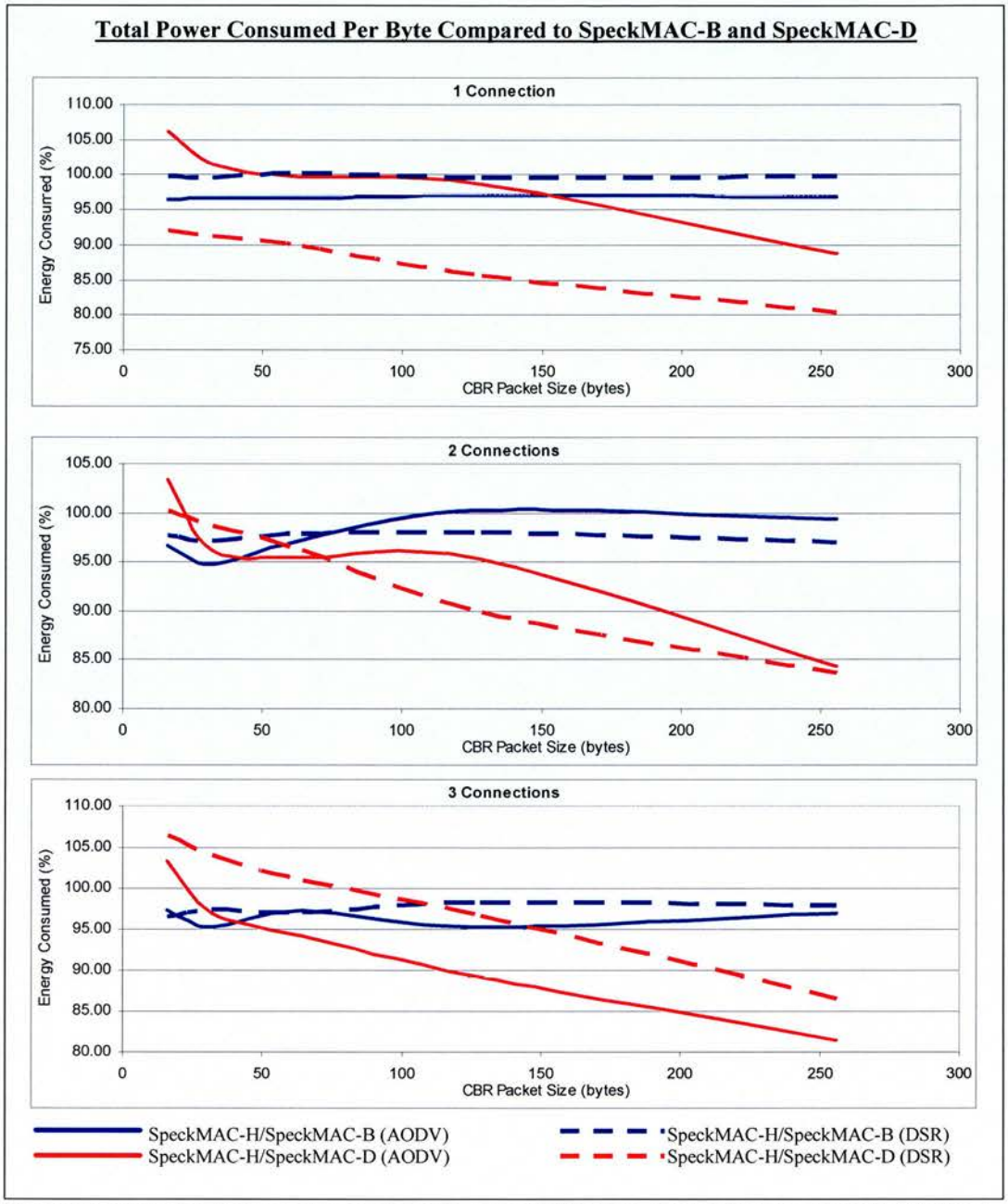


Figure 4-22: The percentage of energy consumed per CBR byte received by the SpeckMAC-B and SpeckMAC-D over SpeckMAC-H for the static scenario

4.6 Summary and Discussions

The work in this chapter set out to answer this question: do the SpeckMAC algorithms provide energy-efficiency support to MANET routing protocols for Specknets? To answer this question, the SpeckMAC protocols were modelled in the Qualnet simulator and a selection of reactive MANET routing protocols were evaluated. To validate the models, the outputs from the simulator were correlated with the results presented in Chapter 3 from the PerSpeckz-64 and the analytical models.

Using AODV and DSR as the routing algorithms, it was concluded from the simulation results that all versions of the SpeckMAC protocol were able to support the communication requirements of AODV and DSR whilst conserving energy for both static and mobile networks. A hybrid version of SpeckMAC, SpeckMAC-H, was also presented to allow nodes to transmit using either SpeckMAC-B or SpeckMAC-D based on decisions made utilising some higher layer information. From the simulation results, it was demonstrated that, in a majority of the simulated scenarios, SpeckMAC-H managed to achieve better energy-efficiency than SpeckMAC-B and SpeckMAC-D when the appropriate SpeckMAC algorithm was selected based on the packet type to be transmitted.

In conclusion, the results presented in the Chapter 3 and the simulations results presented in this chapter demonstrated the ability of the SpeckMAC algorithms to allow nodes to conserve energy when there is little or no data requirements, e.g. in the case of the periodic data transmission in normal sensor network, and at the same time support the sporadic peer-to-peer communications required for Speckled Computing.

4.7 Future Work

The results presented in this chapter only demonstrate the possibilities of using DSR and AODV with SpeckMAC as the underlying MAC algorithm to support energy-efficient peer-to-peer communications in Specknets. However, more exhaustive experiments based on carefully planned simulation scenarios have to be performed to evaluate, in greater details, the performance of the various algorithms and protocols.

Further improvements to the Qualnet simulator are also needed to ensure more accurate simulation results. One such improvement is the design of an accurate short range channel model for the radio. This allows the simulation results to relate more closely to the short communication ranges of the Specks. Furthermore, battery models should also be implemented into the Qualnet simulator so that lifetimes of the nodes could be accurately modelled. Some work had been done in this aspect by others [120, 121], however, as the batteries modelled were based on larger cells with higher capacities, new battery models will be needed to simulate the characteristics of the batteries with extremely low capacities, such as the coin cells used in the experiments described in Section 3.8.

Further research will also have to be undertaken in the design of SpeckMAC-H such that the decision to use either SpeckMAC-B or SpeckMAC-D can be made optimally. The algorithm proposed in this thesis made the decision based on the packet type; however, other attributes (such as the nodal speed, packet size, etc) and statistical data (such as profiling of the radio traffic) could also be considered.

The emphasis of this chapter had been focused on energy efficiency as Specks would have very limited power available. However, another important limitation is the small amount of memory available on each Speck. As memory storage is required to store routing information, the number of routes that could be supported would be limited by the amount of memory available. Thus, it will be useful to create a distributed memory model among Specks such that routing tables could be distributed amongst a cluster of Specks. One idea is to divide the Specks into zones, also known as SpeckZones, such that intra-zone routing table could be distributed

amongst the Specks within each zone. More information about this approach can be obtained from Appendix D.3.

Lastly, it will be interesting to evaluate the performance of AODV and DSR on the ProSpeckz prototypes using SpeckMAC algorithms and B-MAC as the MAC protocol based on some real-life application.

Chapter 5

CONCLUSION

"In a nutshell..."

5.1 Summary of the thesis

The key objective of this thesis was to design, implement and evaluate energy-efficient communication MAC protocols for Specknets using the motivation of replacing the long preambles transmitted by B-MAC with either the wakeup-frames (SpeckMAC-B) or the retransmissions of the data frame (SpeckMAC-D). It has been successfully demonstrated that both SpeckMAC algorithms managed to achieve higher energy-efficiency than B-MAC based on results obtained from the analytical models as well as the experiments performed on the physical test-bed. It was also shown that the increased energy-efficiency was realised without sacrificing other performance attributes, such as the transmission latency and delivery ratios, when compared against B-MAC: SpeckMAC-D had achieve higher delivery ratios and lower latencies than B-MAC whereas SpeckMAC-B had achieved lower energy wastage due to overhearing.

The main contributions of this thesis were presented in Chapter 3 and are summarized as follow:

- the proposed use of wakeup frames in place of the long preambles transmitted by B-MAC (SpeckMAC-B)
- the proposed use of data frame retransmissions in place of the long preambles transmitted by B-MAC (SpeckMAC-D)
- the derivation of analytical models for SpeckMAC-B, SpeckMAC-D and B-MAC

- the implementation of SpeckMAC-B, SpeckMAC-D and B-MAC on the ProSpeckz platform
- the evaluations, using analytical models and physical experiments, between the performance achieved by B-MAC, SpeckMAC-B and SpeckMAC-D in terms of energy used to support broadcast traffic, energy used to support unicast traffic, delivery ratios, energy wasted in a jammed/noisy communication channel, latency in a single-hop and multi-hop latency. It was shown that :
    - SpeckMAC-B achieved higher energy-efficiency than B-MAC, and this improvement is more significant under unicast traffic conditions.
    - SpeckMAC-D achieved higher energy-efficiency , higher packet delivery success rate and lower end-to-end latencies than B-MAC,
- the actual measurement of the lifetimes achieved by ProSpeckz performing a location maintenance algorithm using B-MAC, SpeckMAC-B and SpeckMAC-D based on two different types of batteries. The Polymer Li-ion on the ProSpeckz running SpeckMAC-B and SpeckMAC-D managed to extend the lifetimes when compared to the B-MAC ones by 35% and 40% respectively. The CR1210 coin cells on the ProSpeckz running SpeckMAC-B and SpeckMAC-D managed to extend the lifetimes when compared to the B-MAC ones by 96% and 103% respectively.
- the description on the impact that drain profiles of the Polymer Li-ion cells and the coin cells have on the lifetime of the Specks

Other contributions of this thesis (presented in Chapter 2 and Chapter 4) are summarized as follow:

- ***(As described in Chapter 2)***
- the design and implementation of ProSpeckz, the first Speck prototype, using commercially off-the-shelf components
- the implementations of demonstrators using the ProSpeckz platform

- the design and implementation of PerSpeckz-64, a test-bed to provide functionalities for evaluating potential Specknet algorithms and protocols in a controlled and systematic manner

  (As described in Chapter 4)

- the implementation of the SpeckMAC-B and SpeckMAC-D protocols onto the Qualnet simulator
- the verification of the simulator using results obtained from the ProSpeckz implementation
- the evaluation of a proposed cross-layer algorithm, SpeckMAC-H, which switches between the use of wake-up frames and data frame retransmissions based on the decision made on the higher layers of the communication stack
- the performance evaluations demonstrating that all variations of the SpeckMAC algorithms were able to support the DSR and AODV network protocols in an energy-efficient manner

BIBLIOGRAPHY

[1] *Consortium in Speckled Computing,* website: www.specknet.org (Last accessed: March 2007)

[2] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Emerging Challenges: Mobile Networking for Smart Dust," *Journal of Communications and Networks*, vol. 2, pp. 188-196, 2000.

[3] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next century challenges: mobile networking for "Smart Dust"," *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 271-278, 1999.

[4] B. Warneke, M. Last, B. Liebowitz, and K. S. J. Pister, "Smart Dust: communicating with a cubic-millimeter computer," *Computer*, vol. 34, pp. 44-51, 2001.

[5] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 95-107, 2004.

[6] J. D. Day and H. Zimmermann, "The OSI reference model," *Proceedings of the IEEE*, vol. 71, pp. 1334-1340, 1983.

[7] J. Rabaey, J. Ammer, J. L. da Silva Jr, and D. Patel, "PicoRadio: Ad-hoc wireless networking of ubiquitous low-energysensor/monitor nodes," *VLSI, 2000. Proceedings. IEEE Computer Society Workshop on*, pp. 9-12, 2000.

[8] J. M. Rabaey, M. J. Ammer, J. L. da Silva Jr, D. Patel, and S. Roundy, "PicoRadio supports ad hoc ultra-low power wireless networking," *IEEE Computer Journal*, vol. 33, pp. 42-48, 2000.

[9] D. D. Patel, "Energy in Ad-Hoc Networking for the PicoRadio (Masters Thesis)," Department of Electrical Engineering and Computer Sciences, University of California, 1999.

[10] J. L. Da Silva Jr, J. Shamberger, M. J. Ammer, C. Guo, S. Li, R. Shah, T. Tuan, M. Sheets, J. M. Rabaey, and B. Nikolic, "Design methodology for PicoRadio networks," *Proc. Design Automation and Test in Europe*, pp. 314–323.

[11] J. A. Gutierrez, "IEEE Standard for Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)," IEEE, 2003.

[12] S. I. G. Bluetooth, "Specifications of the Bluetooth System (Core)," *Version*, vol. 1, pp. 2003.31-39.

[13] LAN MAN Standards Committee of the IEEE Computer Society, "IEEE Std 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specification," *IEEE*, 1999 (reaffirmed 2003).

[14] J. Foerster, E. Green, S. Somayazulu, and D. Leeper, "Ultra-Wideband Technology for Short-or Medium-Range Wireless Communications," *Intel Technology Journal*, vol. 2, pp. 1-11, 2001.

[15] G. R. Aiello and G. D. Rogerson, "Ultra-wideband wireless systems," *Microwave Magazine, IEEE*, vol. 4, pp. 36-47, 2003.

[16] Federal Communications Commission, "Revision of Part 15 of the Commission's Rules Regarding Ultra-Wideband Transmission Systems," *First Report and Order (FCC 02-48)*, Feb 2002.

[17] T. M. Siep, I. C. Gifford, R. C. Braley, and R. F. Heile, "Paving the way for personal area network standards: an overview of the IEEE P 802. 15 Working Group for Wireless Personal Area Networks," *IEEE Personal Communications*, vol. 7, pp. 37-43, 2000.

[18] T. G. Zimmerman, "Personal Area Networks (PAN): Near-Field Intra-Body Communication (PhD Thesis)," Massachusetts Institute of Technology, 1995.

[19] J. Karaoguz, "High-rate wireless personal area networks," *Communications Magazine, IEEE*, vol. 39, pp. 96-102, 2001.

[20] *International Telecommunication Union,* website: http://www.itu.int/home/index.html (Last accessed: March 2007)

[21] *European Telecommunications Standards Institute,* website: http://www.etsi.org/ (Last accessed: March 2007)

[22] *Federal Communications Commission,* website: http://www.fcc.gov/ (Last accessed: March 2007)

[23] International Telecommunications Union, "International Table of Frequency Allocations," *Radio Regulations*, vol. 1, pp. RR5.138,RR5.150, 2004.

[24] Crossbow Technology Inc, "Wireless Microsensor Mote: MICA2DOT," *Document Part Number: 6020-0043-04 Rev D*, Feb 2005.

[25] O.Kasten and M.Langheinrich, "First Experiences with Bluetooth in the Smart-Its Distributed Sensor Network," *Workshop on Ubiquitous Computing and Communications*, Sept. 2001.

[26] R. Kling, R. Adler, J. Huang, V. Hummel, and L. Nachman, "Intel Mote: using bluetooth in sensor networks," *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 318-318, 2004.

[27] L. Nachman, R. Kling, R. Adler, J. Huang, and V. Hummel, "The Intel® Mote platform: a Bluetooth-based sensor network for industrial monitoring," *Proceedings of the 4th international symposium on Information processing in sensor networks*, 2005.

[28] A. S. Chipcon, "CC2420 2.4 GHz IEEE 802.15. 4/ZigBee-ready RF Transceiver," *Chipcon AS, Oslo, Norway*, 2004.

[29] Phycom Taiwan Ltd, "Multilayer Ceramic Antenna for Bluetooth and WLAN IEEE 802.11b (2.45GHz ISM BAND) Datasheet."

[30] Cypress MicroSystems, "PSoC™ Mixed Signal Array: Technical Reference Manual (TRM)," *PSoC TRM, Version 1.22*, 2005.

[31] J. Polastre, R. Szewczyk, and D. Culler, "Telos: enabling ultra-low power wireless research," *Fourth International Symposium on Information Processing in Sensor Networks*, pp. 364-369, 2005.

[32] Texas Instruments Incorporated, "MSP430C11x1, MSP430F11x1A Mixed Signal Microcontroller," *SLAS241H*, Sept 2004.

[33] R. B. Smith, "SPOTWorld and the Sun SPOT," *Proceedings of the 6th international conference on Information processing in sensor networks*, pp. 565-566, 2007.

[34] C. T. Inc, "MicaZ Datasheet."

[35] C. Park, J. Liu, and P. H. Chou, "Eco: an ultra-compact low-power wireless sensor node for real-time motion monitoring," *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pp. 398-403, 2005.

[36] N. Madabhushi, "KMote-Design and Implementation of a Low Cost, Low Power Hardware Platform for Wireless Sensor Networks," *Master of Technology Dissertation, IIT, Kanpur*, 2007.

[37] I. Moteiv, "Tmote Sky datasheet," 2006.

[38] D. Lymberopoulos and A. Savvides, "XYZ: a motion-enabled, power aware sensor node platform for distributed sensor network applications," *Proceedings of the 4th international symposium on Information processing in sensor networks*, 2005.

[39] *Zarlink Semiconductor Inc,* website: http://www.zarlink.com/ (Last accessed: 2007)

[40] Texas Instruments Incorporated, "MSP430x12x Mixed Signal Microcontroller," *SLAS312C*, Sept 2004.

[41] SANYO Electric Co. Ltd, "Cell Type ML414 Specifications," 2006.

[42] *MACCOR Inc,* website: www.maccor.com (Last accessed: 2007)

[43] I. F. Akyildiz, S. Weilian, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, pp. 102-114, 2002.

[44] B. P. Crow, I. Widjaja, L. G. Kim, and P. T. Sakai, "IEEE 802.11 Wireless Local Area Networks," *Communications Magazine, IEEE*, vol. 35, pp. 116-126, 1997.

[45] R. McNally, K. J. Wong, and D. K. Arvind, "A distributed algorithm for logical location estimation in speckled computing," *Wireless Communications and Networking Conference, 2005 IEEE*, vol. 3, 2005.

[46] GPI International Ltd, "GPCR1210 Data Sheet," *rev. 0300/765*, 2005.

[47] J. M. Rabaey, "Silicon Platforms for the Next Generation Wireless Systems-What Role Does Reconfigurable Hardware Play," *Proceedings of the The Roadmap to Reconfigurable Computing, 10th International Workshop on Field-Programmable Logic and Applications*, pp. 277-285, 2000.

[48] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet," *Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*, pp. 96-107, 2002.

[49] R. R. Murphy, "Rescue robotics for homeland security," *Communications of the ACM*, vol. 47, pp. 66-68, 2004.

[50] V. H. MacDonald, "AMPS: The Cellular Concept," *The Bell System Technical Journal*, vol. 58, pp. 5–41, 1979.

[51] E. I. Standard, "Cellular System Dual-Mode Mobile Station—Base Station Compatibility Standard (IS-54)," *Electronic Industries Association EIA , May*, 1990.

[52] A. J. Viterbi, *CDMA: principles of spread spectrum communication*: Addison Wesley Longman Publishing Co., Inc. Redwood City, CA, USA, 1995.

[53] L. Kleinrock and F. Tobagi, "Packet Switching in Radio Channels: Part I--Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics," *IEEE Transactions on Communications*, vol. 23, pp. 1400-1416, 1975.

[54] T. Eng and L. B. Milstein, "Comparison of hybrid FDMA/CDMA systems in frequency selectiveRayleigh fading," *IEEE Journal on Selected Areas in Communications*, vol. 12, pp. 938-951, 1994.

[55] J. R. Foerster and L. B. Milstein, "Analysis of hybrid, coherent FDMA/CDMA systems in Ricean multipathfading," *IEEE Transactions on Communications*, vol. 45, pp. 15-18, 1997.

[56] M. Mouly and M. B. Pautet, *The GSM System for Mobile Communications*: Telecom Publishing, 1992.

[57] K. Kammerlander, "Benefits of combined TDMA/CDMA operation for third generationmobile radio systems," *IEEE 4th International Symposium on Spread Spectrum Techniques and Applications*, vol. 2, 1996.

[58] M. Haardt, A. Klein, R. Koehn, S. Oestreich, M. Purat, V. Sommer, and T. Ulrich, "The TD-CDMA based UTRA TDD mode," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1375-1385, 2000.

[59] A. Nasipuri, J. Zhuang, and S. R. Das, "A multichannel CSMA MAC protocol for multihop wireless networks," *IEEE Wireless Communications and Networking Conference*, pp. 1402-1406, 1999.

[60] N. Jain, S. R. Das, and A. Nasipuri, "A multichannel CSMA MAC protocol with receiver-based channelselection for multihop wireless networks," *Proc. of the 10th International Conference on Computer Communications and Networks*, pp. 432-439, 2001.

[61] A. Nasipuri and S. R. Das, "Multichannel CSMA with signal power-based channel selection formultihop wireless networks," *Vehicular Technology Conference, 2000. IEEE VTS-Fall VTC 2000. 52nd*, vol. 1, 2000.

[62] A. Muqattash, M. Krunz, and W. E. Ryan, "Solving the Near Far Problem in CDMA-based Ad Hoc Networks," *Ad Hoc Networks Journal*, vol. 1, pp. 435–53, 2003.

[63] R. Lupas and S. Verdu, "Near-far resistance of multiuser detectors in asynchronous channels," *IEEE Transactions on Communications*, vol. 38, pp. 496-508, 1990.

[64] A. Muqattash and M. Krunz, "CDMA-based MAC protocol for wireless ad hoc networks," *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pp. 153-164, 2003.

[65] F. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part II--The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution," *IEEE Transactions on Communications*, vol. 23, pp. 1417-1433, 1975.

[66] S. S. Lam, "A Carrier Sense Multiple Access Protocol for Local Networks," *Technical Report: CS-TR-113, University of Texas at Austin*, 1979.

[67] I. Holyer, "The NP-completeness of edge-colouring," *SIAM Journal in Computing*, vol. 10, pp. 718-720, 1981.

[68] L. Hu, "Distributed code assignments for CDMA packet radio networks," *Proc. of the INFOCOM'91*, pp. 1500-1509, 1991.

[69] I. Katzela and M. Naghshineh, "Channel assignment schemes for cellular mobile telecommunicationsystems: a comprehensive survey," *IEEE Personal Communications*, vol. 3, pp. 10-31, 1996.

[70] S. L. Wu, C. Y. Lin, Y. C. Tseng, and J. P. Sheu, "A New Multi-Channel MAC Protocol with On-Demand Channel Assignment for Multi-Hop Mobile Ad Hoc Networks," *International Symposium on Parallel Architectures, Algorithms, and Networks, I-SPAN*, pp. 232–237, 2000.

[71] T. Herman and S. Tixeuil, "A Distributed TDMA Slot Assignment Algorithm for Wireless Sensor Networks," *Proceedings of the First Workshop on Algorithmic Aspects of Wireless Sensor Networks (AlgoSensors' 2004)*, pp. 45–58.

[72] S. Ramanathan, "A unified framework and algorithm for channel assignment in wireless networks," *Wireless Networks*, vol. 5, pp. 81-94, 1999.

[73] S. I. G. Bluetooth, "Specification of the Bluetooth System–Version 1.1," *Core Specification* vol. 1, 2001.

[74] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, 2002.

[75] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," *Proc. of the first international conference on Embedded networked sensor systems*, pp. 171-180, 2003.

[76] Y. C. Tseng, C. S. Hsu, and T. Y. Hsieh, "Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks," *Proc. of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 1, 2002.

[77] C. S. Raghavendra and S. Singh, "PAMAS: Power Aware Multi-Access Protocol with Signaling for Ad Hoc Networks," *ACM Computer Communication Review (July 1998)*, pp. 5–26, 1998.

[78] C. Guo, L. C. Zhong, and J. M. Rabaey, "Low Power Distributed MAC for Ad Hoc Sensor Radio Networks," *Global Telecommunications Conference, 2001*, vol. 5, pp. 2944-2948, 2001.

[79] X. Yang and N. H. Vaidya, "A wakeup scheme for sensor networks: achieving balance between energy saving and end-to-end delay," *10th IEEE Real-Time and Embedded Technology and Applications Symposium*, pp. 19-26, 2004.

[80] L. Gu and J. A. Stankovic, "Radio-Triggered Wake-Up for Wireless Sensor Networks," *Real-Time Systems*, vol. 29, pp. 157-182, 2005.

[81] A. Woo and D. E. Culler, "A transmission control scheme for media access in sensor networks," *Proceedings of the 7th annual international conference on Mobile computing and networking*, pp. 221-235, 2001.

[82] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: a hybrid MAC for wireless sensor networks," *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pp. 90-101, 2005.

[83] S. Ganeriwal, D. Ganesan, H. Shim, V. Tsiatsis, and M. B. Srivastava, "Estimating clock uncertainty for efficient duty-cycling in sensor networks," *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pp. 130-141, 2005.

[84] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for Ad Hoc routing," *Proceedings of the 7th annual international conference on Mobile computing and networking*, pp. 70-84, 2001.

[85] A. Cerpa and D. Estrin, "ASCENT: adaptive self-configuring sensor networks topologies," *Mobile Computing, IEEE Transactions on*, vol. 3, pp. 272-285, 2004.

[86] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Wireless Networks*, vol. 8, pp. 481-494, 2002.

[87] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," *Proceedings of the 4th international conference on Embedded networked sensor systems*, pp. 307-320, 2006.

[88] W. Ye, F. Silva, and J. Heidemann, "Ultra-low duty cycle MAC with scheduled channel polling," *Proceedings of the 4th international conference on Embedded networked sensor systems*, pp. 321-334, 2006.

[89] M. C. Vuran and I. F. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, pp. 316-329, 2006.

[90] "QualNet," in http://www.scalablenetworks.com, 2006.

[91] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," *Proceedings of the conference on Communications architectures, protocols and applications*, pp. 234-244, 1994.

[92] D. Bertsekas and R. Gallager, *Data Networks*: Prentice-Hall, Inc., 1987.

[93] C. C. Chiang, H. K. Wu, W. Liu, and M. Gerla, "Routing in clustered multihop, mobile wireless networks with fading channel," *Proceedings of IEEE SICON*, vol. 97, pp. 197-211, 1997.

[94] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," *Mobile Networks and Applications*, vol. 1, pp. 183-197, 1996.

[95] T. Clausen and P. Jacquet, "RFC3626: Optimized Link State Routing Protocol (OLSR)," *Internet RFCs*, 2003.

[96] P. A. Humblet, "Another adaptive distributed shortest path algorithm," *IEEE Transactions on Communications*, vol. 39, pp. 995-1003, 1991.

[97] C. Cheng, R. Riley, S. P. R. Kumar, and J. J. Garcia-Luna-Aceves, "A loop-free extended Bellman-Ford routing protocol without bouncing effect," *ACM SIGCOMM Computer Communication Review*, vol. 19, pp. 224-236, 1989.

[98] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks," *35th Annual Hawaii International Conference on System Sciences (HICSS'2001)*, 2001.

[99] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing," presented at Second IEEE Workshop on Mobile Computer Systems and Applications, 1999.

[100] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, vol. 353, pp. 153-181, 1996.

[101] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," presented at IEEE INFOCOM, 1997.

[102] C. K. Toh, "Associativity-Based Routing for Ad Hoc Mobile Networks," *Wireless Personal Communications*, vol. 4, pp. 103-139, 1997.

[103] Z. J. Haas and M. R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks," *Internet Draft-Mobile Ad hoc Networking (MANET) Working Group of the Internet Engineering Task Force (IETF), March*, 2000.

[104] Z. J. Haas and M. R. Pearlman, "The performance of query control schemes for the zone routingprotocol," *IEEE/ACM Transactions on Networking*, vol. 9, pp. 427-438, 2001.

[105] M. R. Pearlman and Z. J. Haas, "Determining the optimal configuration for the zone routing protocol," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1395-1414, 1999.

[106] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 56-67, 2000.

[107] M. Gunes, U. Sorges, and I. Bouazizi, "ARA-the ant-colony based routing algorithm for MANETs," *Proc. of the International Conference on Parallel Processing Workshops*, pp. 79-85, 2002.

[108] G. Di Caro, F. Ducatelle, and L. M. Gambardella, "AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks," *Proceedings of Parallel Problem Solving from Nature (PPSN VIII)*, vol. 3242, pp. 461–470.

[109] F. Ducatelle, G. Di Caro, and L. M. Gambardella, "Ant agents for hybrid multipath routing in mobile ad hoc networks," *Proc. of the second Wireless On-demand Network Systems and Services Conference*, pp. 44-53, 2005.

[110] M. Dorigo and G. Di Caro, "The ant colony optimization meta-heuristic," *Mcgraw-Hill'S Advanced Topics In Computer Science Series*, pp. 11-32, 1999.

[111] A. Boukerche and S. Rogers, "GPS query optimization in mobile and wireless ad hoc networking," *Proceedings of the 6th IEEE Symposium on Computers and Communications*, pp. 198–203, 2001.

[112] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 243-254, 2000.

[113] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A distance routing effect algorithm for mobility (DREAM)," *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 76-84, 1998.

[114] Y. B. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in mobile ad hoc networks," *Wireless Networks*, vol. 6, pp. 307-321, 2000.

[115] G. Pei, M. Gerla, and T. W. Chen, "Fisheye state routing: a routing scheme for ad hoc wirelessnetworks," *IEEE International Conference on Communications*, vol. 1, 2000.

[116] J. Albowicz, A. Chen, and L. Zhang, "Recursive position estimation in sensor networks," *IEEE Int. Conf. on Network Protocols*, pp. 35–41, 2001.

[117] N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'Dea, "Relative location estimation in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2137-2148, 2003.

[118] C. Savarese, J. Rabaey, and K. Langendoen, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks," *USENIX Technical Annual Conference*, pp. 317–328, 2002.

[119] J. Postel, "RFC 791: Internet Protocol," *Information Science Institute, University of Southern California, CA, Sept*, 1981.

[120] B. Vasu, M. Varshney, R. Rengaswamy, M. Marina, A. Dixit, P. Aghera, M. Srivastava, and R. Bagrodia, "SQualNet: a scalable simulation framework for sensor networks," *Proceedings of the 3rd international conference on Embedded networked sensor systems*, pp. 322-322, 2005.

[121] S. Park, A. Savvides, and M. B. Srivastava, "Battery capacity measurement and analysis using lithium coin cellbattery," *International Symposium on Low Power Electronics and Design*, pp. 382-387, 2001.

Appendix A. Impact of the Shape of Communication

Due to the requirement for line-of-sight, optical systems are usually unidirectional. One such example is the infrared systems as defined by the IrDA. Transmitters emit infrared light over a certain degree of coverage, and the receiver would likewise be able to detect infrared signals within a certain angle of view. Therefore, the area of coverage for infrared communications looks like a pie or section. An example of an off-the-shelf IrDA compliant infrared transceiver is the ZiLOG ZHX1820.

Unlike optical systems, radio communications is usually omni-directional. An example of such a radio system is described by the Zigbee standards and an example of a Zigbee transceiver is the Chipcon CC2420 chipset. Radio waves are usually transmitted using tuned antennas, and these antennas allow the area of coverage for radio systems to be, in an ideal case, approximately circular.

To analyse the impact that these coverage shapes have on networking, a simulation program was designed on the ns-2 network simulator to compare omni-directional radio communication and unidirectional infrared systems with the following parameters:

- Since the point of interest is the shape of the area of coverage, it was assumed that nodes were relatively small and would therefore not obstruct the line-of-sight between any two nodes
- The simulation would be carried out in a two-dimensional plane with a size of 300mm x 300mm.
- To be fair, the radio was given a circular range of 40mm, whereas infrared is given a range of 80mm with a 90 degree coverage angle. A simple calculation showed that both radio and infrared have the same area of coverage:

  Radio area of coverage = $40^2 * \pi = 5026mm^2$

  Infrared area of coverage = $(90/360) * 80^2 * \pi = 5026mm^2$

- Simulations were performed with a range of 30 to 120 nodes, in intervals of 20 nodes.

- In each simulation set, nodes moved with a limited maximum speed. The maximum speed simulated range from 0mm/s to 3mm/s, in intervals of 0.5mm/s, with nodes moving in a random way-point walk, with a pause rate of 1 second. This means that from a rest position, each node will move at a randomly-selected speed to a random destination; once it reaches the destination, it would pause for 1 second, before selecting another random destination and move towards it at a new random speed.
- Each scenario was simulated 20 times and the results were averaged. The simulated time was 30 seconds.

Two interesting attributes were measured in the simulations:

- **Number of link changes:** This is the number of changes in the direct link communication between any two nodes in the network. In other words, it is the number of times changes in one-hop neighbour occurs.
- **Number of route changes:** This is the number of changes in the shortest route between any node pairs in the network.

Figure A.1 shows the number of link changes over the various simulation scenarios. When the network is static, nodes do not have mobility, therefore no link changes would ever occur for all densities. As the speed of the nodes increase, the number of link changes would occur more frequently as expected due to the higher number of connections being made and broken. The number of link changes that occurred over infrared links (in red) is almost three times greater than that of the radio links, even though theoretically both have the same area of communication coverage.

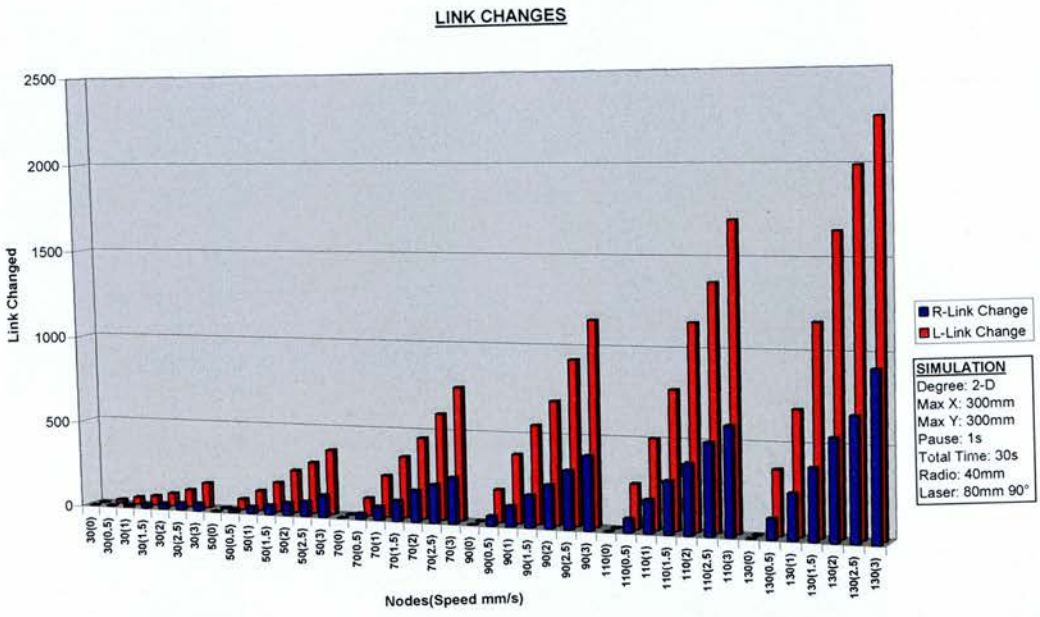


Figure A.1: The number of link changes that occurred during the simulation over various network density and nodal speed

The number of link changes that occur in a network has a huge impact on the performance of any network routing algorithm. An example of such an impact was demonstrated using routes calculated by the shortest path algorithm, and counting the number of route changes that occurs as shown in Figure A.2.

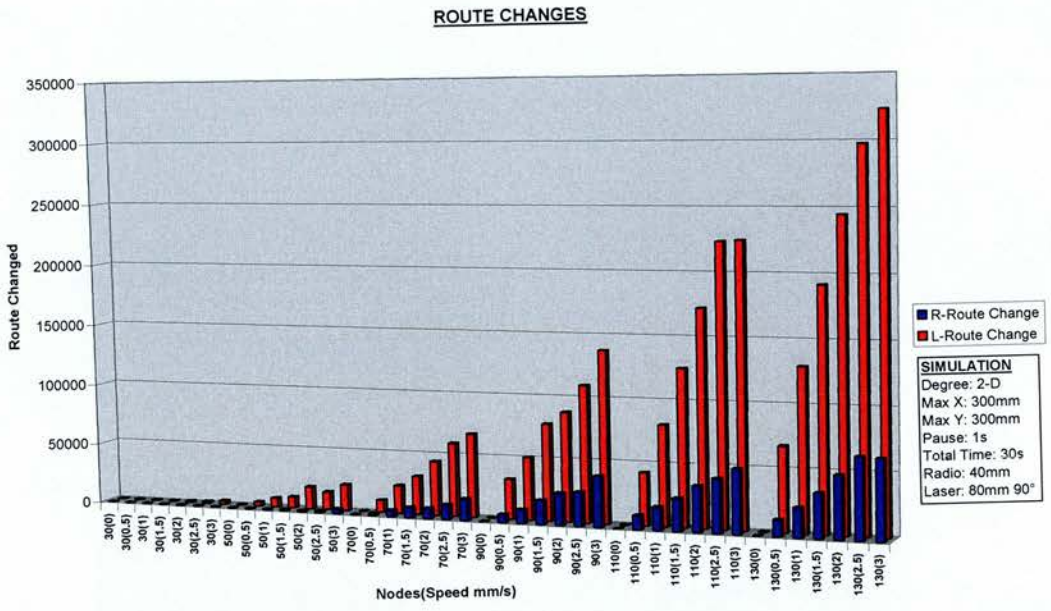


Figure A.2: The number of route changes that occurred during the simulation over various network density and nodal speed

It can be observed that the circular area of coverage provided by radio communications is preferred over the pie-shaped area of coverage provided by infrared for a constant area of coverage. Furthermore, the experiment had ignored the fact that infrared, or any optical communications, cannot pass through opaque objects, which requires a receiver that is correctly aligned to the receiving node and does not have symmetrical links. The inclusion of these facts into the simulation makes unidirectional optical-based communications a less likely candidate for Specknets. Therefore, based on these simulation results, radio communication was selected as the preferred wireless medium for the Speck prototypes.

Appendix B. Flowcharts for B-MAC and SpeckMAC

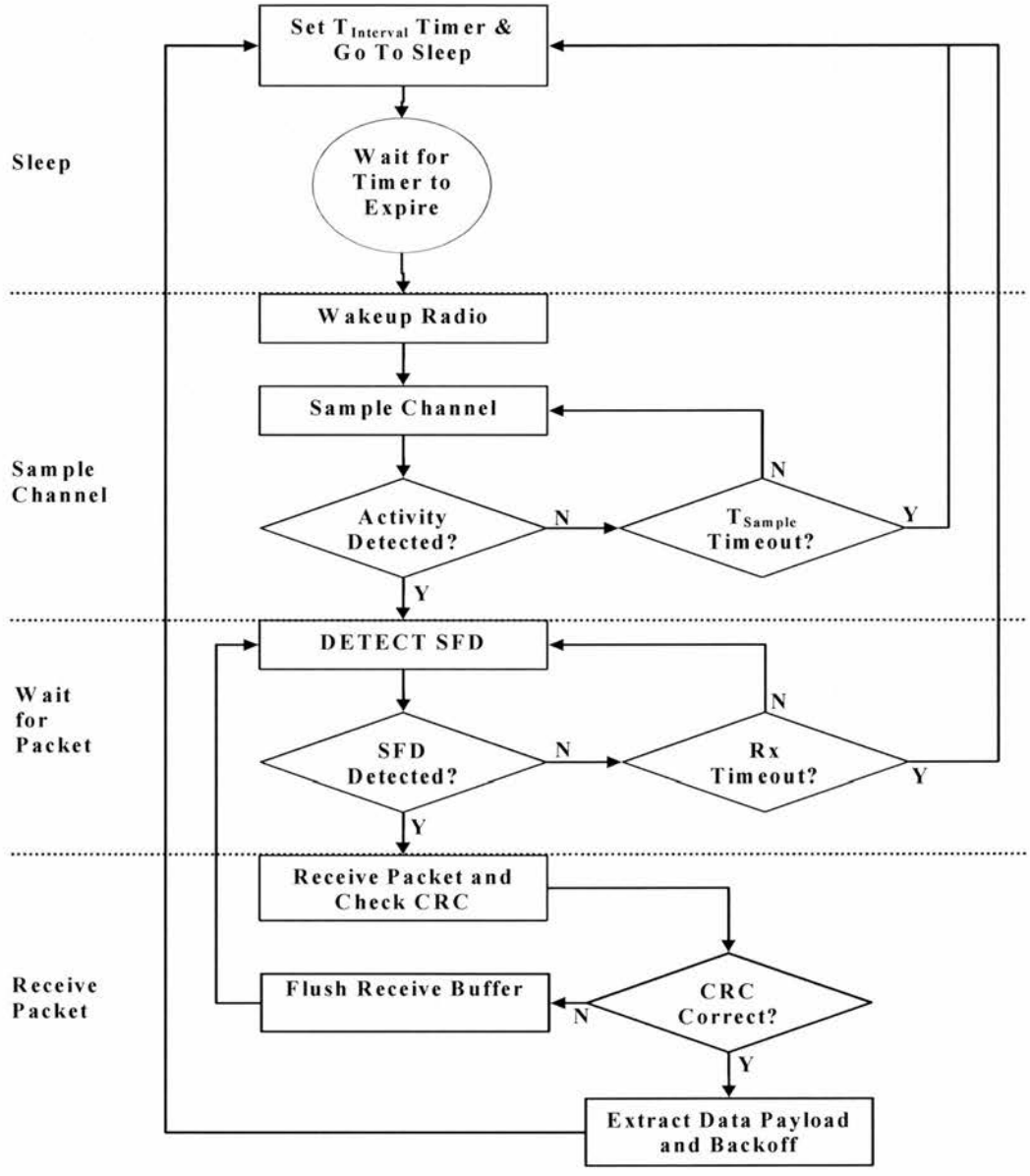


Figure B.1: Flowchart for channel sampling and the reception of data packets using B-MAC

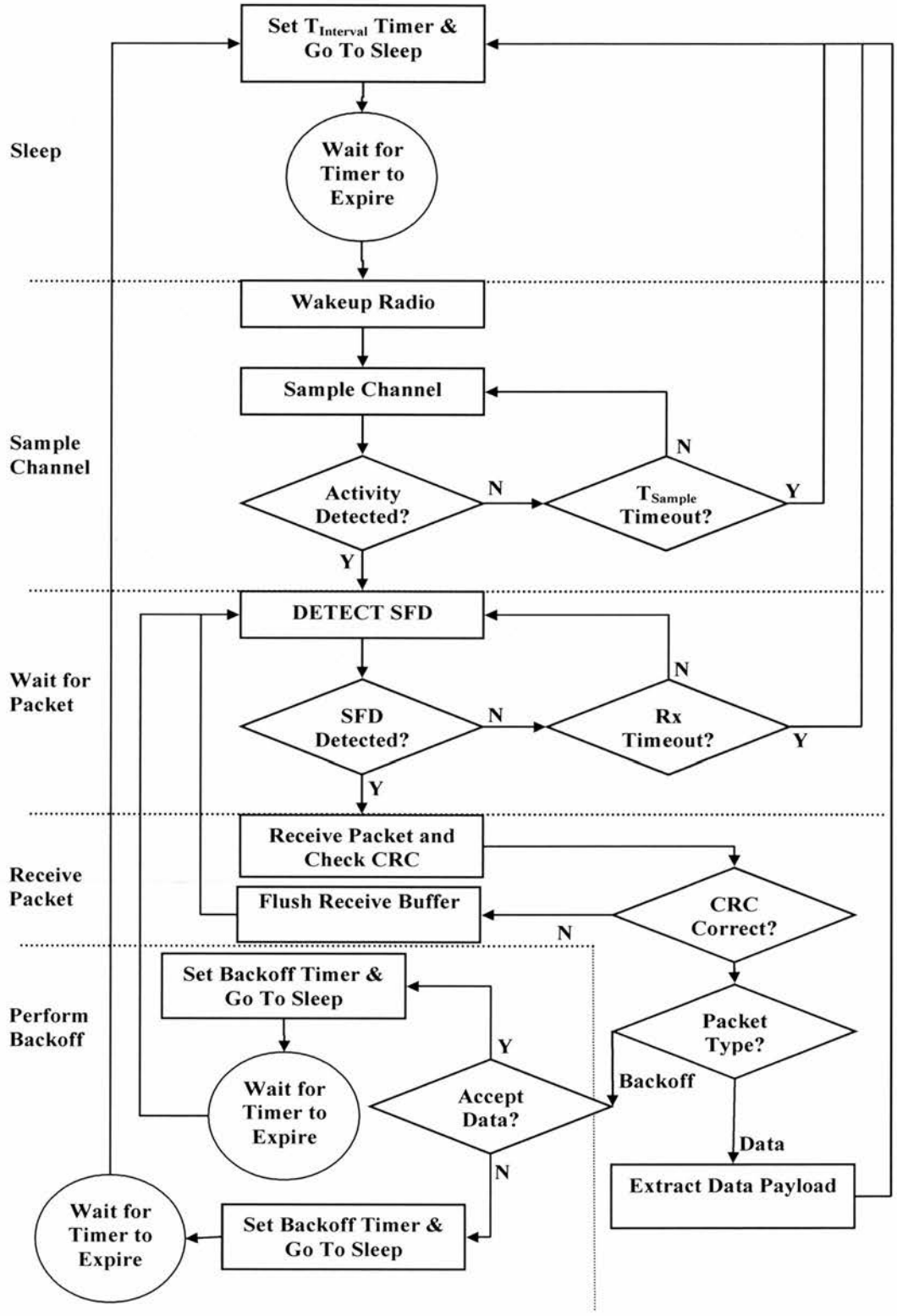


Figure B.2: Flowchart for channel sampling and the reception of data packets using SpeckMAC-B

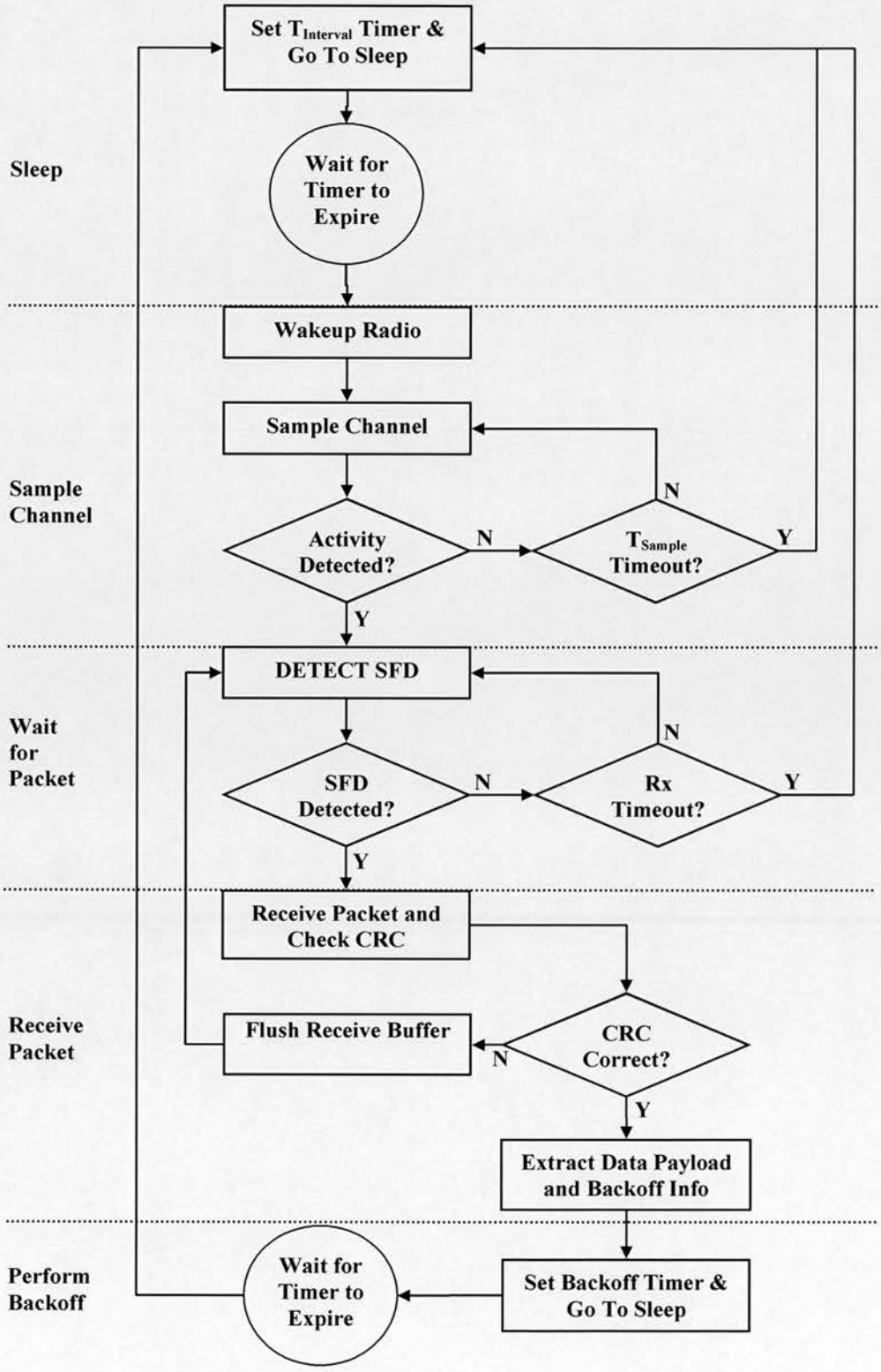


Figure B.3: Flowchart for channel sampling and the reception of data packets using SpeckMAC-D

Appendix C. Some of the initial MAC ideas for Specknet

Several initial ideas for energy-efficient MAC algorithms where tried out and implemented on the ProSpeckz prior to the development of SpeckMAC. Two of such ideas would be briefly mentioned here; however, further developments for these ideas were not continued for this thesis as the initial implementations of the algorithms demonstrated less than satisfactory results. In spite of that, these ideas could still be useful in other scenarios, therefore, further development and modification to the ideas would be left for future work.

C.1 simMAC: A simple distributed random-access power-aware MAC algorithm

To overcome the need for transmitting SYNC frames in S-MAC and T-MAC, a simple MAC protocol, simMAC, was designed that allowed Specks to communicate using low duty-cycle rates and in an unsynchronised manner. This simple algorithm relied on performing redundant retransmissions to remove the need for any synchronisation within the network while allowing nodes to duty-cycle its receiver by only turning it on periodically. Without the need for synchronisation and centralized control, simMAC allowed Specknets to be scalable and for Specks to communicate in a peer-to-peer fashion in the face of node mobility. Furthermore, unlike scheduled MAC algorithms, transmission of any radio signals was only carried out when a node had at least one packet to send as no schedule information needs to be maintained.

Figure C.1 shows the operation of simMAC. As the name suggest, the operation of the algorithm was extremely simple consisting of two phases, the sleep phase and the active phase. All nodes would periodically turn on their receivers for the duration of T_{RxOn} (the active phase) and then turn it off for T_{RxOff} (the sleep phase). Thus, the duty-cycle of the radio receiver, when there is no packets to be transmitted, could be calculated as follows:

Duty-cycle period, $T_{DC\_Period} = T_{RxOn} + T_{RxOff}$

Receiver Duty-cycle, $DC_{simMAC} = T_{Rx\_On}/T_{DC\_Period}$

For a source node to send a packet to a destination node, the source would have to continue retransmitting the packet for at least the duration of T_{DC_Period}. In order to ensure that at lease one of the retransmissions would be received by the destination node, the interval between each retransmission, T_{ReTx}, must satisfy the following rule;

$$T_{ReTx} < T_{RxOn} - (T_{TxPkt} + 2*T_{MCD})$$

where T_{MCD} is the maximum clock skew between the embedded oscillator of any two nodes and T_{TxPkt} is the time needed to retransmission one packet. Furthermore, a node that is in the process of receiving a packet, i.e. the start-of-frame delimiter had been detected, would always complete receiving the packet before turning its receiver off.

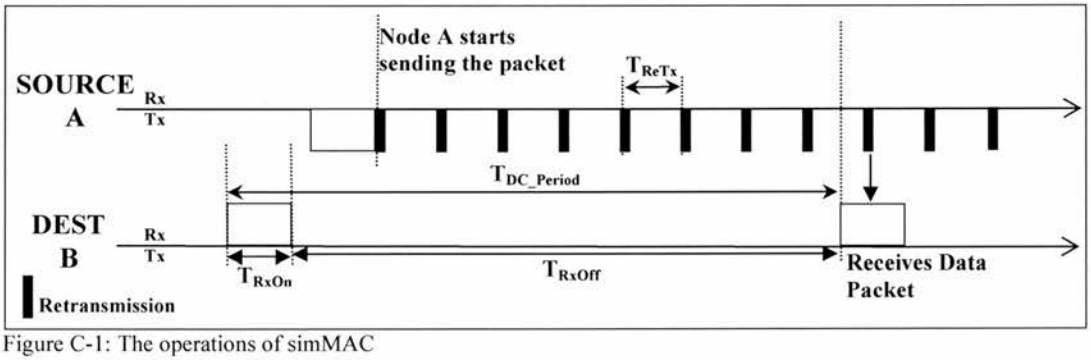


Figure C-1: The operations of simMAC

The overhead required for retransmission is a function of T_{RxOn} and T_{DC_Period}, and the number of times a packet has to be transmitted, N_{simMAC_Tx}, is calculated as follows;

$$N_{simMAC\_Tx} = T_{DC\_Period}/T_{RxOn}$$

It is then easy to see that the any node using simMAC with the same parameters would always be able to receive at least one retransmission given that both nodes are within range of each other for at least $2*T_{Dc\_Period}$.

In order to avoid collision and reduce the overheads associated with performing CSMA, nodes would only be able to begin transmission of a new data packet immediately after an active phase, provided that no packets were received during the

active phase. Thus, within a one-hop network neighbourhood, there would only be one node that is in the process of retransmitting the data packets at any one time.

Further work on simMAC was halted due to two problems. Firstly, like S-MAC, simMAC required that all nodes have a common and fixed duty-cycle. While this is alright if data traffic requirements are uniformed across the whole network and is constant over time, this would not be ideal in networks such as Specknets where data transmissions were assumed to be sporadic. Furthermore, the delivery performance for the algorithm was also unsatisfactory when simMAC was implemented onto the ProSpeckz. This conclusion was derived from an experiment carried out using the PerSpeckz-64 test-bed where different network sizes consisting of 2, 4, 6, 8, 10 and 15 nodes were selected to participate in communications within a one-hop neighbourhood. Each node would transmit a 32 byte packet periodically, ranging from 0.5, 1.0, 1.5 and 2.0 seconds per second. For each scenario, 50 iterations were executed and the results were averaged and shown in Figure C.2.

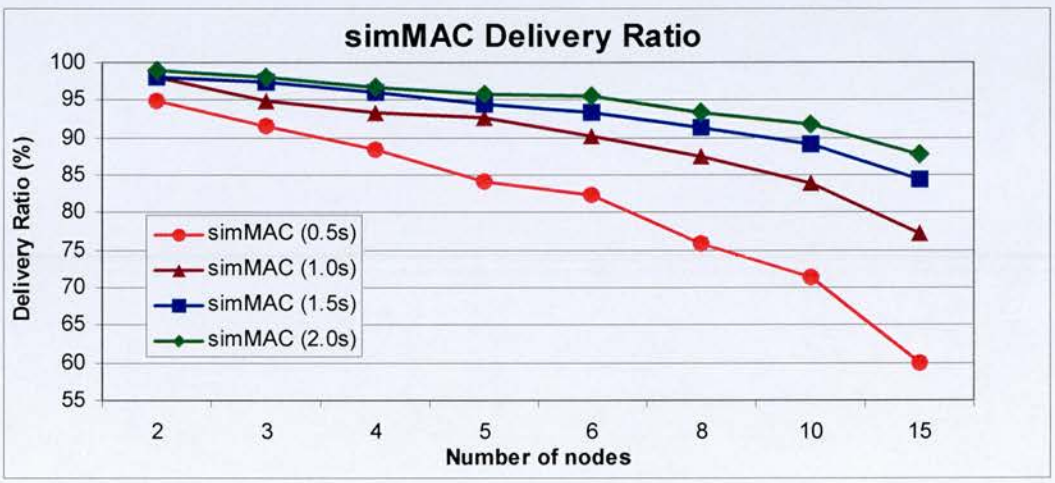


Figure C-2: Delivery ratio of simMAC over a range of neighbourhood sizes and traffic load

C.2 MAC using ambient noise

Instead of using two separate radio architectures as described in section 3.2.4.5 to enable energy-conservation and remove the need for a base-station to provide coordination in synchronised MAC protocols, a possible alternative solution is the use of ambient noise to synchronise communications in Specknet. Ambient noise,

such as those generated from the power supply lines, the flickering of a bulb, ambient sound and vibrations could also be a possible signal source for coordinating communication between Specks. Since Specks are expected to have very short ranges, all neighbours of a Speck should also experience similar ambient noise. Thus, a trial was carried out to experiment with the possibility of utilising the 50/60Hz signal noise generated by the power supply lines using the ProSpeckz. A simple noise detector was designed using the reconfigurable digital and analogue blocks on the ProSpeckz as shown in Figure C.3. A short wire was connected to the pin "Port_0_7" of the ProSpeckz to detect any possible noise. This input was passed through a programmable gain amplifier which amplified the received signal. This amplified signal was then passed through a 4^{th} order Butterworth low-pass filter with a corner frequency of 70Hz. The output of the filter was then connected to pin "Port_0_5". A sample output of this pin is shown in Figure C.4(a).

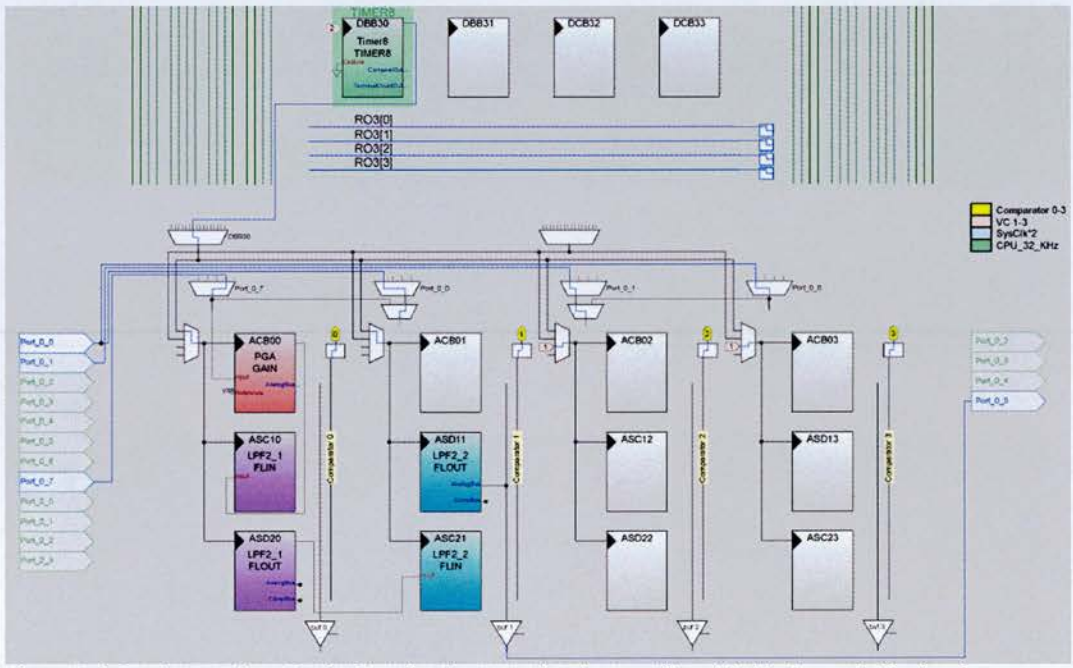


Figure C-3: Implementation of a simple noise detector using the reconfigurable blocks on the ProSpeckz

The easiest method to enable power-savings using of the signal from the noise detector was to turn off the radio receivers upon detecting a negative-going transition in the noise signal and turning them on during a positive-going transition as shown in

Figure C.4(b). The period in which the radio was turned on is the active period while the period in which the radio is turned off is the sleep period. All communications could only occur during the active period. The processor could also go to sleep during the sleep period and can be woken-up by a positive-going transition on the noise signal. Due to the nature of the power line noise, this allows the radio to sleep for about 40% of the time.

To enable lower duty cycles, a slightly more complex method could used as shown in Figure C.4(c). This method makes use of a timer to delay the start of the active period. The duration of the active period would also be controlled using the timer instead of using the negative-going transition of the noise signal.

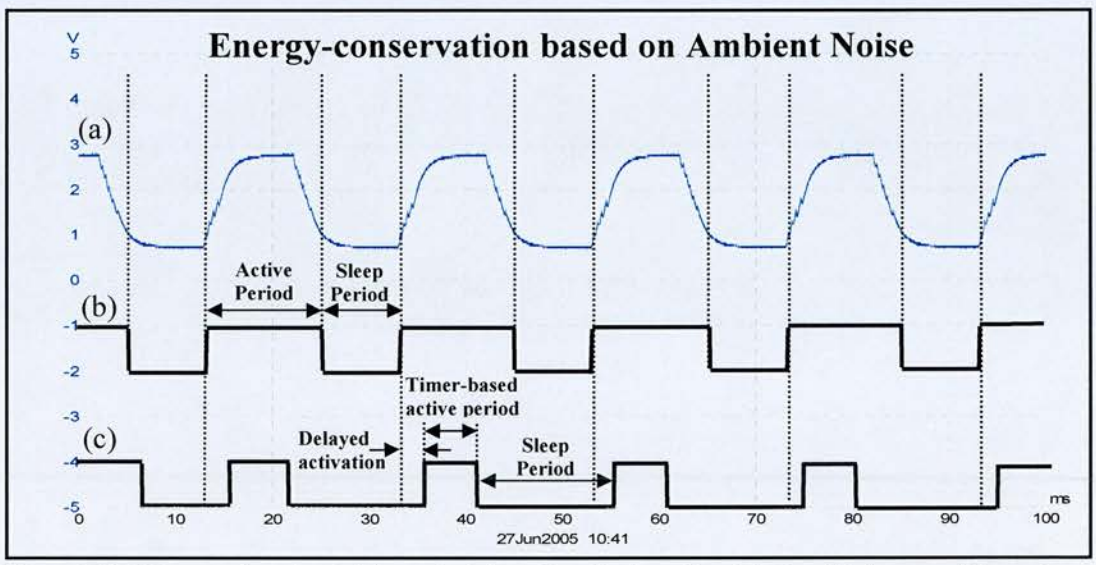


Figure C-4: Diagram showing (a) the output from the simple noise detector, (b) the use of the noise signal for 50% duty-cycle of the radio (c) the use of the noise signal for less then 50% duty-cycle of the radio

Using the method shown in Figure C.4(b), two ProSpeckz were placed on an AC power source as shown in Figure C.5. One of the ProSpeckz would change the color of its LED and transmit one 32 byte frame with the LED colour information every second. The receiving ProSpeckz would then change its LED colour based on the received packet. It was observed that when both ProSpeckz were placed directly above an AC power source, the LED colours of the ProSpeckz were coordinated; however, when the ProSpeckz were moved just 10cm away, no packets were

received at all at the destination ProSpeckz. Furthermore, it was also discovered that the simple implementation was highly sensitive to external interferences; for example, when a human hand was placed near the ProSpeckz, the ProSpeckz would turn on continuously until the hand was remove. Thus, it was concluded that even though it might be possible to coordinate active and sleep intervals of Specks using ambient noise, more complex and carefully designed circuitry for the noise detector would be necessary to improve its sensitive but at the same time, the circuitry must also be able to filter unwanted external interferences. The design of such a noise detector would be left for future work due to its possible complexity.

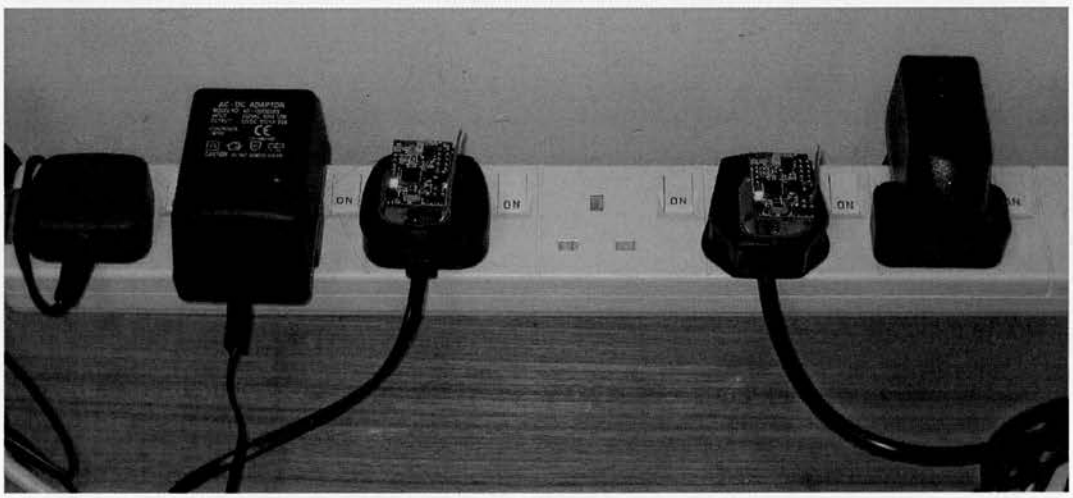


Figure C-5: Experiment carried out to access the possibility of using ambient noise from the power lines to coordinate data transmission between two ProSpeckz

Appendix D. Papers published

As mentioned in the declaration, some of the work presented in this thesis as well as some of the contents of this dissertation had been published in the following peer-reviewed conferences. These papers are attached to the end of this thesis in the following order:

D.1 **SpeckMAC: Low-power Decentralised MAC protocols for Low Data Rate Transmission in Specknets**
K.J Wong, D.K Arvind, in Proceedings of The Second International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality (ACM SIGMOBILE), pp. 71-78, Italy, May 2006
(Acceptance Rate ~17%)

D.2 **Perspeckz-64: Physical Test-bed for Performance Evaluation of MAC and Networking Algorithms for Specknets**
K.J Wong, D.K Arvind, in Proceedings of The Second International Workshop on Multi-hop Ad Hoc Networks: from Theory to Reality (ACM SIGMOBILE), pp. 122-124, Italy, May 2006

D.3 **Specknets: New Challenges for Wireless Communication Protocols**
K.J Wong, D.K Arvind, in Proceedings of The International Conference on Information Technology and Applications (IEEE), vol. 2, pp. 728-733, Australia, July 2005
(Acceptance Rate ~25%)

D.4 **Specknet-based Responsive Environments**
K.J Wong, D.K Arvind, , N. Sharwood-Smith, A. Smith, in Proceedings of The International Symposium on Consumer Electronics (IEEE), pp. 334-338, Macau, June 2005

D.5 **A Distributed Algorithm for Logical Location Estimation in Speckled Computing**
R.McNally, K.J Wong, D.K Arvind, in Proceedings of The Wireless Communications & Networking Conference 2005 (IEEE), U.S.A, March 2005
(Acceptance Rate ~35%)

D.6 **Speckled Computing: Disruptive Technology for Networked Information Appliances**
D.K. Arvind, K.J.Wong, in Proceedings of The International Symposium on Consumer Electronics (IEEE), pp. 219-223, U.K, September 2004

SpeckMAC: Low-power Decentralised MAC Protocols for Low Data Rate Transmissions in Specknets

Kai-Juan Wong, D.K.Arvind
Research Consortium in Speckled Computing
School of Informatics, University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ, United Kingdom
k.j.wong@sms.ed.ac.uk | dka@inf.ed.ac.uk

ABSTRACT

This paper introduces SpeckMAC, a novel low-power distributed, unsynchronised, random-access MAC protocol for a wireless mobile ad-hoc network of miniature specks called specknet. Two variations of the SpeckMAC protocol were compared theoretically with the well-known B-MAC protocol. All three MAC protocols were implemented on a larger speck prototype called the ProSpeckz for a logical location maintenance algorithm for two types of batteries with differing current drain profiles. ProSpeckz utilizing SpeckMAC-B and SpeckMAC-D showed improvements in lifetime over those using the B-MAC protocol by 27.4% and 38.0%, respectively when Polymer Li-ion batteries were used; and by 83.5% and 117.9%, respectively, when CR1220 button cells were employed.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communications; C.2.2 [**Network Protocols**]

General Terms

Performance, Design, Measurement, Experimentation

Keywords

Speckled Computing, Specknet, Wireless Sensor Networks, Wireless Mobile Ad-hoc Networks, Low-Power MAC protocols

1. INTRODUCTION

Specks are intended to combine sensing, processing, wireless networking and energy source (battery) in a single package designed to be contained in a volume approximately 5x5x5 mm in dimension. A network of specks, called a specknet, is intended to be mobile and assumed to be unreliable. Computing with specks, or Speckled Computing [1], connects the virtual world of computers with the physical world of sensory data such as temperature, pressure, acceleration and position.

REALMAN'06, May 26, 2006, Florence, Italy.

Wireless communications in specknets bring about several challenges to protocol designers due to the characteristics and properties of the network [2]. The radio range of each speck will be small (a few centimetres) and specknet will be an instance of a dense, wireless, mobile, ad-hoc network. Furthermore, due to the minute footprint of each speck, energy and memory resources are extremely limited onboard each speck, thus lightweight power-aware communication protocols are necessary.

This paper introduces a new low power distributed unsynchronised random-access MAC protocol called SpeckMAC to support low data rates in wireless mobile ad-hoc networks such as specknets. Theoretical comparisons have been made with the well-known B-MAC protocol [3], which are confirmed by experimental results on a larger speck prototype, the ProSpeckz [4], for a logical location maintenance algorithm [5] for two types of batteries which differ in their ability to sustain high current drains. The results demonstrate that the SpeckMAC protocols outperform the B-MAC protocol both theoretically and experimentally in terms of energy savings.

Both the B-MAC and SpeckMAC protocols are link-level MAC algorithms and other higher level issues such as scheduling, acknowledgement, routing and handshaking (CTS-RTS) were not considered in this paper, but should be easily implemented if needed by the application.

In the rest of this paper: Section 2 describes the existing power-aware MAC protocols with a detailed description of B-MAC; Section 3 describes the two variations of the SpeckMAC protocol which differs in the nature of the redundant transmissions in lieu of the long preamble in the B-MAC protocol; Section 4 introduces the equations for calculating the energy consumption for the protocols under comparison; Section 5 gives an outline of the physical implementation; Sections 6 and 7 provide results of the theoretical and experimental comparisons with conclusions drawn in Section 8.

2. EXISTING MAC PROTOCOLS

MAC protocols for low-power wireless networks can be broadly classified as either centralised or decentralised. In the first category, a base station or cluster head provides collision-free operation within the network or cluster. Time Division Multiplexing Access (TDMA) [6] is an example of centralized MAC protocols. In TDMA-based MAC protocols such as Bluetooth [7], spatial space is divided into small slots in which nodes are only awake for the times when they are assigned a slot. This requires accurate time synchronisation among the nodes in

the network. Furthermore, the base station responsible for coordination consumes more power than the other nodes. Therefore, in the case of specknets where *all* the nodes in the network have limited energy supply, centralised MAC protocols would not be appropriate.

Decentralised MAC protocols designed to address these deficiencies can be divided into two classes, scheduled or random-access. Scheduled MAC protocols, such as S-MAC [8] and T-MAC [9], employ time slots in a fashion similar to TDMA. However, these slots are relatively larger than those used by centralised schemes, which obviates the need for tight time synchronisation. To avoid the need for a centralised controller, each node keeps a schedule of the time slots in which it would be awake. Nodes are then required to transmit its schedule information periodically to allow other ad-hoc nodes to follow its schedule (in the case of a single schedule system), or for other nodes to be aware of its schedule (in the case of a multi-schedule system). In either case, schedule updates would have to occur regularly to support mobility in nodes, which could be a significant overhead as mobility increases.

Random-access MAC protocols allow nodes to access the channel "on-demand" without recourse to any schedule or synchronisation, and are therefore suitable for networks which are both mobile and ad-hoc. PAMAS [10] uses a separate radio channel for waking up the destination nodes during data transmission. This requires additional hardware, and would be inappropriate for specks where both space and energy are at a premium. B-MAC [3], on the other hand, uses in-channel signalling to wakeup destination nodes. The following section describes the properties of B-MAC in greater detail.

2.1 B-MAC

B-MAC is a random access MAC protocol that does not require nodes to be synchronised to one another. B-MAC is therefore appropriate in networks with highly mobile nodes as updates of neighbour's schedules are not required. Furthermore, B-MAC transfers the communication costs to the transmission of data, making it more expensive for a node to transmit a data packet as compared to receiving one. This is appropriate on the premise that sensor networks require low data rates and have higher nodal densities, and each node would tend to transmit less and receive more packets.

B-MAC operates by periodically listening to the channel for activity. The frequency of the sampling is determined by a user-selected $T_{Interval}$, as shown in Figure 1. Nodes will turn on their receivers if the channel is sensed to be busy and would turn them off after a data packet is received or after a timeout. For data transmission, the source node will send a long preamble lasting the duration of $T_{Preamble}$, before sending the data packet. This preamble is used to wakeup destination nodes during the channel listening phase, and therefore for the correct operation of the protocol, the following must be enforced:

$$T_{Preamble} > T_{Interval}$$

Of all the MAC protocols considered so far, B-MAC would best suit a mobile ad-hoc network with low data rates such as specknets, and as it has been shown to outperform S-MAC [3], B-MAC was chosen as the basis for comparison with the SpeckMAC protocols introduced in this paper.

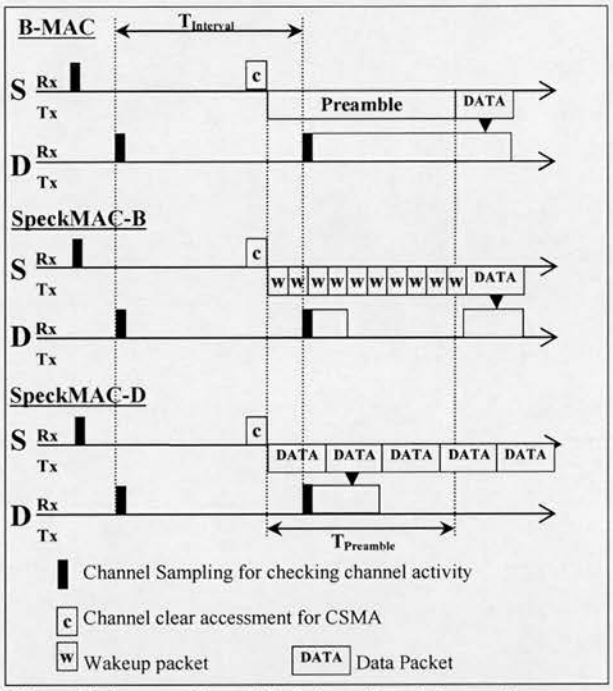


Figure 1. An overview of the interactions between the source node, S and the destination node, D for the three different MAC protocols

3. SpeckMAC

SpeckMAC is a distributed random-access MAC protocol based on in-channel signalling. However, instead of sending long preambles for each data packet as in the case of B-MAC, SpeckMAC uses redundant retransmissions to fulfil the role of the long preamble in B-MAC. There are two variations of SpeckMAC: SpeckMAC-B sends wakeup packets in place of the preambles, and SpeckMAC-D sends the actual data packet repetitively. Figure 1 shows the differences in the operations of the three MAC protocols through the interaction of a source node, S, sending a data packet to a destination node D.

For transmission of data packets using SpeckMAC-B (SpeckMAC-Backoff), the source node fills the preamble space occupied by B-MAC with a number of small wakeup packets. Each wakeup packet contains information about the destination address of the data packet and also timing information about when the source node is expected to send the data packets. Every node will periodically, as determined by $T_{Interval}$, listen to the channel for activity (similar to B-MAC). If activity is sensed, the node will turn on its receiver and wait for the reception of a packet, which could be either a data or a wakeup packet. If a data packet is received, the radio will revert into the idle state and periodic listening resumes. Should a wakeup packet be received, the node will revert to the idle state and backs-off for a duration specified by the received timing information, provided that the destination address specified in the wakeup packet matches its own address or a broadcast address. The node will then turn on its receiver again after the back-off duration in anticipation of the data packet. After reception of the data packet, the node will resume periodic channel listening.

SpeckMAC-Data (SpeckMAC-D) operates in a manner very similar to SpeckMAC-B. However, instead of filling the preamble space with wakeup packets, SpeckMAC-D fills it with repetitions of the data packet. Additionally, unlike B-MAC, each of these data packets will only be padded with a short preamble of 3 bytes. For the reception of data packets, nodes will listen to the channel periodically at intervals specified by $T_{Interval}$, and if activity is detected on the channel, the node will turn on its receiver to enable the reception of the data packet. After the data packet is successfully received, the node will place its radio into idle state for the duration of $T_{Interval}$, before resuming periodic channel listening.

4. THE CALCULATION OF ENERGY CONSUMPTION

Table 1 and Table 2 show the constants and the variables, respectively, for the calculation of the energy consumption by the radio for the different protocols based on the CC2420 specification [11].

Table 1. Constants defined by the CC2420 radio chip

Term	Description	Unit	Value
T_{Txb}	Time to transmit a byte	Sec	0.000032
T_{RxTx}	Time for radio to switch from Rx/Idle to Tx modes	Sec	0.000192
T_{IRx}	Time for radio to switch from idle to Rx mode	Sec	0.000192
T_{RSSI}	Time for RSSI to provide a reading	Sec	0.000128
T_{Sample}	Time to sample the channel for activity from idle state ($= T_{IRx} + T_{RSSI}$)	Sec	0.00032
P_{Rx}	Power required for radio to receive	mW	62.1
P_{Tx}	Power required for radio to transmit	mW	28.1(-25dBm) 57.4(0dBm)
P_{Idle}	Idle Power	mW	1.41

Table 2. Variables used in the equations

Term	Description	Unit
T_{Tx}	Time to transmit a packet	Sec
T_{Rx}	Time to receive a packet	Sec
$T_{TxWakeup}$	Time to transmit a wakeup packet (Used by SpeckMAC-B)	Sec
T_{WGuard}	Wakeup guard time; extra time that the receiver should be turned on before the anticipated data packet arrives. (Used by SpeckMAC-B)	Sec
$T_{Interval}$	Time between each sampling of the channel during channel listening	Sec
$T_{Preamble}$	Duration that the preamble has to be sent for	Sec
T_{Guard}	A guard time to provide allowance for slight clock skews between nodes	Sec
T_{csma}	Duration a channel has to be clear before node assumes that the channel is free	Sec
N_{Neigh}	Number of neighbours per node	Nodes
N_{Pkts}	Number of packets sent per second	Pkt/Sec

The following equations will be used to compare the protocols:

$$\mathbf{T_{Preamble}} = T_{Interval} + T_{Sample} + T_{Guard} \quad (1)$$

Furthermore, the following conditions are used to simplify computation and comparisons:

i) All equations are normalised to one second.

ii) Before the transmission of a packet, the node will turn on its receiver and wait for the RSSI (Radio Strength Signal Information) to be stable. It then checks if the channel is free for the whole duration of T_{CSMA}. For calculations used in the following sections, the channel is always assumed to be clear, and therefore no collision back-off is taken into consideration. For all protocols, the total time, T_{CSMA}, and the energy consumed for CSMA is as follows:

$$\mathbf{T_{CSMA}} = (T_{IRx} + T_{RSSI} + T_{csma}) * N_{Pkts}$$

$$\mathbf{E_{CSMA}} = T_{CSMA} * P_{Rx} \quad (2)$$

iii) To provide fair comparisons with B-MAC, the calculations for B-MAC are based on the "average case", while those for SpeckMAC is based on the "average or worst case".

4.1 B-MAC

The equations for calculating the energy consumption for the implementation of B–MAC on the CC2420 is shown in this section (with reference from [3]).

The total energy, $\boldsymbol{E_{BMAC}}$, consumed by B-MAC is the total energy needed for CSMA (2), transmission (4), reception (5), channel listening (6) and idling (7) of the radio.

$$\mathbf{E_{BMAC}} = E_{CSMA} + E_{BMAC\_Tx} + E_{BMAC\_Rx} + E_{BMAC\_Listen} + E_{BMAC\_Idle} \quad (3)$$

To transmit a data packet, a node will send a long preamble followed by the data packet. Thus, the duration, $\boldsymbol{T_{BMAC\_Tx}}$, for which the transmitter will be turned on, and the energy consumption is as follows;

$$\mathbf{T_{BMAC\_Tx}} = (T_{RxTx} + T_{Preamble} + T_{Tx}) * N_{Pkts}$$

$$\mathbf{E_{BMAC\_Tx}} = T_{BMAC\_Tx} * P_{Tx} \quad (4)$$

For the reception of a data packet, the nodes will turn on their receiver on hearing activity on the channel during channel listening. The nodes will remain on until a packet is received. The total time, $\boldsymbol{T_{BMAC\_Rx}}$, that the receiver is turned on for and the energy consumed is as follows:

$$\mathbf{T_{BMAC\_Rx}} = ((0.5 * T_{Preamble}) + T_{Rx}) * N_{Pkts} * N_{Neigh}$$

$$\mathbf{E_{BMAC\_Rx}} = T_{BMAC\_Rx} * P_{Rx} \quad (5)$$

For the rest of the time that the radio is not receiving or transmitting data, B-MAC performs periodic channel listening.

The duration for which the receiver is turned on for channel listening and the energy consumption is as follows:

$$\mathbf{T_{BMAC\_Listen}} = \lfloor((((1 - T_{BMAC\_Tx}) - T_{BMAC\_Rx}) - T_{CSMA}) / T_{Interval})\rfloor * T_{sample}$$

$$\mathbf{E_{BMAC\_Listen}} = T_{BMAC\_Listen} * P_{Rx} \quad (6)$$

The time for which the radio is idling and the energy consumption is therefore:

$$\mathbf{T_{BMAC\_Idle}} = (((1 - T_{BMAC\_Tx}) - T_{BMAC\_Rx}) - T_{CSMA}) - T_{BMAC\_Listen}$$

$$\mathbf{E_{BMAC\_Idle}} = T_{BMAC\_Idle} * P_{Idle} \quad (7)$$

The equations presented in this section are true if, and only if, the following condition is satisfied:

$$T_{BMAC\_Listen} \geq 0$$

This is because if T_{BMAC_Listen} is negative, it would imply that the bandwidth of the channel has been exceeded.

4.2 SpeckMAC-B

The total energy, $E_{SpeckMACB}$, consumed by SpeckMAC-B is the total energy needed for CMSA (2), transmission (9), reception (10), channel listening (12) and idling (13) of the radio.

$$\mathbf{E_{SpeckMACB}} = E_{CSMA} + E_{SpeckMACB\_Tx} + E_{SpeckMACB\_Rx} + E_{SpeckMACB\_Listen} + E_{SpeckMACB\_Idle} \quad (8)$$

To transmit a data packet, a node would send wakeup packets repetitively for a minimum duration of $T_{Preamble}$ as shown in Figure 1. After sending the last wakeup packet, the node would then transmit the data packet. Therefore, the time spent in transmission and the energy consumption is as follows:

$$\mathbf{T_{SpeckMACB\_Tx}} = (T_{RxTx} + (\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) + T_{Tx}) * N_{Pkts}$$

$$\mathbf{E_{SpeckMACB\_Tx}} = T_{SpeckMACB\_Tx} * P_{Tx} \quad (9)$$

For the reception of the data packet, in the worst case, channel listening would occur in between wakeup packets. Therefore the receiver would have to stay on for a duration of $2 * T_{TwWakeup}$ in order to receive a complete wakeup packet. The radio would then be placed in the idle state during the back-off duration. The radio would be turned on again for a period of, T_{WGuard}, before the anticipated arrival of the data packet based on the information from the received wakeup packet. This provides allowances for transmission, propagation and reception delays of the wakeup packet. The receiver is turned off once the packet is received.

$$\mathbf{T_{SpeckMACB\_Rx}} = ((2 * T_{TxWakeup}) + T_{IRx} + T_{WGuard} + T_{Rx}) * N_{Pkts} * N_{Neigh}$$

$$\mathbf{E_{SpeckMACB\_Rx}} = T_{SpeckMACB\_Rx} * P_{Rx} \quad (10)$$

Thus, on average, the duration that the radio will be idle due to back-off is calculated as follows:

$$\mathbf{T_{SpeckMACB\_Backoff}} = (((\lceil T_{Preamble} / T_{TxWakeup} \rceil * T_{TxWakeup}) - ((2 * T_{TxWakeup}) + T_{IRx} + T_{WGuard})) * N_{Pkts} * N_{Neigh} / 2) \quad (11)$$

For the rest of the time when the radio is not receiving or transmitting data, SpeckMAC-B listens to the channel periodically. The duration that the receiver is turned on for this purpose and the energy consumed is given as follows:

$$\mathbf{T_{SpeckMACB\_Listen}} = \lfloor(((((1 - T_{SpeckMACB\_Tx}) - T_{SpeckMACB\_Rx}) - T_{SpeckMACB\_Backoff}) - T_{CSMA}) / T_{Interval})\rfloor * T_{Sample}$$

$$\mathbf{E_{SpeckMACB\_Listen}} = T_{SpeckMACB\_Listen} * P_{Rx} \quad (12)$$

The time that the radio is left in the idle state and the energy consumed is therefore:

$$\mathbf{T_{SpeckMACB\_Idle}} = (((1 - T_{SpeckMACB\_Tx}) - T_{SpeckMACB\_Rx}) - T_{CSMA}) - T_{SpeckMACB\_Listen}$$

$$\mathbf{E_{SpeckMACB\_Idle}} = T_{SpeckMACB\_Idle} * P_{Idle} \quad (13)$$

As worst cases assumptions are being used, the equations presented in this section are true if, and only if, the following conditions are satisfied:

$$T_{Preamble} \geq ((2 * T_{TxWakeup}) + T_{IRx} + T_{WGuard})$$

$$T_{SpeckMACB\_Listen} \geq 0$$

4.3 SpeckMAC-D

The total energy, $E_{SpeckMACD}$, consumed by SpeckMAC-D is the total energy needed for CSMA (2), transmission (15), reception (16), channel listening (18) and idling (19) of the radio.

$$\mathbf{E_{SpeckMACD}} = E_{CSMA} + E_{SpeckMACD\text{-}Tx} + E_{SpeckMACD\_Rx} + E_{SpeckMACD\_Listen} + E_{SpeckMACD\_Idle} \quad (14)$$

For transmission of a data packet, SpeckMAC_D would send the data packet repetitively for a minimum duration of $T_{Preamble}$, as shown in Figure 1. After that, it would send the data packet once more. Therefore, the total time for which the transmitter is turned on and the energy consumption is as follows:

$$\mathbf{T_{SpeckMACD\_Tx}} = (T_{RxTx} + ((\lceil T_{Preamble} / T_{Tx} \rceil + 1)) * T_{Tx}) * N_{Pkts}$$

$$\mathbf{E_{SpeckMACD\_Tx}} = T_{SpeckMACD\_Tx} * P_{Tx} \quad (15)$$

As in the previous section, worst case assumptions have been made to provide fair comparisons with B-MAC, i.e., the channel is sampled during channel listening between the transmissions of two data packets. Therefore, the radio will be turned on at the beginning of a data packet and remain so until the next data packet is received, i.e., the radio is turned on for a duration of $2 * T_{Rx}$.

The total time that the receiver is turned on for the reception of data, and the total energy consumption is as follows:

$$\mathbf{T_{SpeckMACD\_Rx}} = (2 * T_{Rx}) * N_{Pkts} * N_{Neigh}$$

$$\mathbf{E_{SpeckMACD\_Rx}} = T_{SpeckMACD\_Rx} * P_{Rx} \quad (16)$$

After every successful packet reception, the node will keep its radio idle for a period equivalent to $T_{Preamble}$. The total time spent in this back-off state is as follows:

$$\mathbf{T_{SpeckMACD\_Backoff}} = T_{Preamble} * N_{Pkts} * N_{Neigh} \quad (17)$$

For the rest of the time when the node is not in the process of receiving or transmitting data packets, it listens to the channel periodically. The total on-time for the receiver and the energy consumption for channel listening is as follows:

$$\mathbf{T_{SpeckMACD\_Listen}} = \lfloor(((((1 - T_{SpeckMACD\_Tx}) - T_{SpeckMACD\_Rx}) - T_{SpeckMACD\_Backoff}) - T_{CSMA}) / T_{Interval})\rfloor * T_{Sample}$$

$$\mathbf{T_{SpeckMACD\_Listen}} = T_{SpeckMACD\_Listen} * P_{Rx} \quad (18)$$

The time that the radio is left in the idle state and the energy consumed is therefore:

$$\mathbf{T_{SpeckMACD\_Idle}} = (((1 - T_{SpeckMACD\_Tx}) - T_{SpeckMACD\_Rx}) - T_{CSMA}) - T_{SpeckMACD\_Listen}$$

$$\mathbf{E_{SpeckMACD\_Idle}} = T_{SpeckMACD\_Idle} * P_{Idle} \quad (19)$$

The equations presented in this section are true if, and only if, the following condition is satisfied:

$$T_{Guard} > T_{Sample}$$

$$T_{SpeckMACD\_Listen} \geq 0$$

5. IMPLEMENTATION ON PHYSICAL PROTOTYPES

In order to evaluate the differences in performance between B-MAC and SpeckMAC, all the protocols were implemented on a physical prototype, the ProSpeckz [4]. This version of the ProSpeckz (ProSpeckz-IIk as shown in Figure 2) measures 32mm x 22mm in size and consists of a CC2420 [11] radio chip, an onboard antenna and a CY8C29666 [12] Programmable System-on-Chip (PSoC) with 32Kbytes of Flash memory and 2Kbytes of Random-Access Memory.

To support the various transmission requirements of the protocols on the ProSpeckz, the "TXFIFO looping" transmission mode of the CC2420 is used instead of the normal "Buffered mode" during the transmission of data. In the "TXFIFO looping" mode, the CC2420 will transmit raw data bytes by looping through the FIFO buffer. This allows a set of data bytes to be transmitted repetitively. In this mode, the CC2420 does not handle packet generation, thus, the PSoC would be responsible for writing the preamble bytes, the start-of-frame delimiter, the data packet and the cyclic redundancy check bytes into the FIFO. This allows the PSoC to generate the long preambles required by BMAC, as well as to efficiently retransmit the data or wakeup packets for the SpeckMAC protocols. In the implementation of the protocols, the following packet structure was used:

```
#define MAX_PKT_LEN 32
typedef struct{
   unsigned char preamble[3]; //preamble
   unsigned char SOD; //start of frame delimiter
   unsigned char dataLength; //length packet – (preamble+SOD)
   unsigned char pktType; //the packet type
   unsigned long srcAddr; //source address
   unsigned long destAddr; //destination address
   unsigned int seqNo; //Sequence number
   unsigned char data[MAX_PKT_LEN]; //the data to be sent
   unsigned int crc; //cyclic redundancy check value
}Packet; //maximum packet length is 50 bytes
```

The wakeup packet structure for SpeckMAC-B is as follows:

```
typedef struct{
   unsigned char preamble[3]; //preamble
   unsigned char SOD; //start of frame delimiter
   unsigned char dataLength; //length packet – (preamble+SOD)
   unsigned char pktType; //the packet type
   unsigned long destAddr; //destination address
   unsigned int boTimer; //backoff timer value till data is sent
   unsigned int crc; //cyclic redundancy check value
}WakeUpPktType; //wakeup packet length = 14bytes
```

Using these data structures, T_{Tx}, T_{Rx} and $T_{TxWakeup}$ can be evaluated to be:

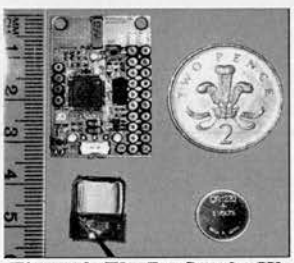


Figure 2. The ProSpeckz IIk with the Polymer Li-ion battery and the CR1220 button cell

$T_{Tx} = T_{Rx} = 50 * T_{Txb} = 0.0016$ seconds

$T_{TxWakeup} = 14 * T_{Txb} = 0.000448$ seconds

The other parameters, as described in Table 2 were implemented with the following values:

$T_{WGuard} = 0.001$ seconds

$T_{Guard} = 0.00068$ seconds

$T_{csma} = 0.001$ seconds

6. THEORETICAL COMPARISONS BETWEEN THE PROTOCOLS

The performance of the protocols depends upon the choice of $T_{Interval}$. Using the parameters obtained from the physical implementation, the equations in (1) to (19) can be solved to determine the optimal $T_{Interval}$ for each protocol, given the expected number of packets to be transmitted per second, and the expected one-hop neighbours a node would have for a certain application.

For example, for an application where each node transmits one packet per second and each node has eleven neighbours, the power consumed by the radio for the application across the range

of $T_{Interval}$ is shown in Figure 3. From the figure, it can be seen that for B-MAC to perform optimally, $T_{Interval}$ would have to be set to 0.0067s and the resultant energy consumption would be 8.34mW. Thus, when $T_{Interval}$ is set to 0.015s for both SpeckMAC-B and SpeckMAC-D, it can be seen that the radio consumes 6.06mW and 5.70mW respectively. The ratio of energy savings achieved by the nodes running the two variations of SpeckMAC over B-MAC under the given scenario can then be calculated as follows:

$$R_{B\text{-}MAC/SpeckMAC\text{-}B} = 8.34/6.06 = 1.376$$

$$R_{B\text{-}MAC/SpeckMAC\text{-}D} = 8.34/5.70 = 1.463 \quad (20)$$

Furthermore, it can be seen that across all values of $T_{Interval}$, both variations of SpeckMAC outperform B-MAC in terms of energy consumption.

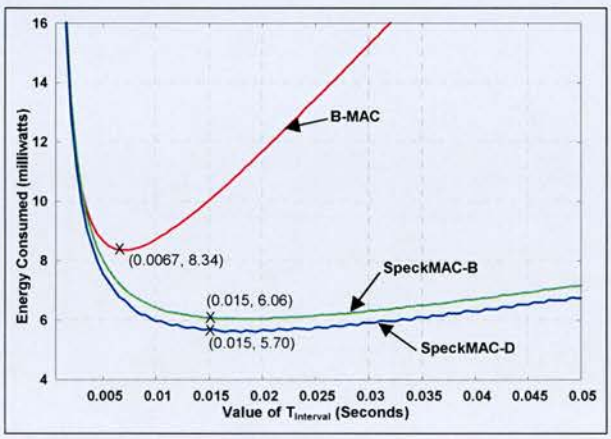


Figure 3. Power consumption of the various protocols transmitting one packet per second with eleven neighbours across a range of $T_{Interval}$

Figure 4 shows the calculated power consumption for the protocols under different network load and densities, using the optimal $T_{Interval}$ in each case. It can be seen that across the whole range of the loads and density shown, SpeckMAC would consume less energy than B-MAC.

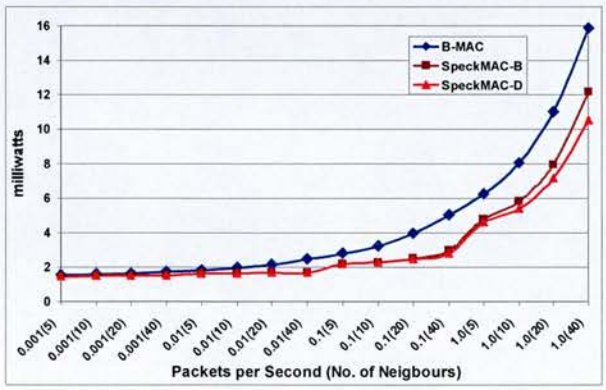


Figure 4. Power consumption of the various protocols under various network loads and densities using optimal $T_{Interval}$

7. COMPARISONS OF THE PROTOCOLS BASED ON EXPERIMENTAL RESULTS FOR THE LOCATION MAINTENANCE ALGORITHM

In order to compare the performances in practice, the three protocols were implemented on the ProSpeckz prototypes for a logical location maintenance algorithm [5]. In this algorithm, each node sends a location-update packet periodically to inform its neighbours about its estimated logical location, together with the list of its neighbours. Upon receiving such a packet, a node compares the list of neighbours of the sending node with its own list of neighbours, and recalculates the estimate of its logical location. As reported in the paper [5], this algorithm performs optimally when there are eleven or more neighbours. Thus, the neighbourhood size used in the experiments was also set to eleven.

In the implementation of the experiment, the processor was sent to sleep whenever it was not processing, transmitting or receiving a packet, in order to conserve energy. For the transmission of the location update, the data component of the packet would then contain the following structure:

```
typedef struct{
        double myX;         //estimate of X coordinate
        double myY;         //estimate of Y coordinate
        unsigned char neighList[24]; //hash list of neigbours
}Loc_Info;
```

For the experiment, a battery-powered ProSpeckz was placed with eleven other mains-powered ProSpeckz to form a networked cluster and was tested to ensure that the bi-directional links were established between the battery-powered ProSpeckz and the mains-powered ones. This ensures that the battery-powered ProSpeckz always has eleven neighbours.

Four such clusters were formed, with each cluster running a different MAC protocol and at a different non-interfering frequency. The four MAC protocols used in this experiment were B-MAC, SpeckMAC-B, SpeckMAC-D and "No MAC". The ProSpeckz running on "No MAC" would have its radio turned on all the time. All the ProSpeckz were communicating at maximum radio strength and each node would send location-update packets every second. The optimal $T_{Interval}$ for the various protocols can be seen in Figure 3. These optimal values were used in this experiment.

Two types of batteries were employed: 12x14mm Polymer Li-ion cells (3.7v, 40mAh@40mA) [13], and CR1220 Lithium coin cells (3v, 40mAh@0.1mA) [14].

The voltage levels of the batteries were sampled every millisecond by a 12-bit scope calibrated to provide an accuracy greater than 99%. For each type of battery, the experiments were run four times to allow each MAC protocol to be rotated among the different ProSpeckz, and across the set of different frequencies. For each run, the four battery-powered ProSpeckz are powered at exactly the same time and are placed in close proximity to each other. This reduces any effects of variations in room temperature (which was kept between 20-25 degrees centigrade) and noise interferences between them during the experiment.

For the experiments using the Polymer Li-ion batteries, new batteries were used for each run. The batteries were fully discharged before they were recharged in parallel. The discharge graph for the Polymer Li-Ion battery for one of the runs is shown in Figure 5. The discharge graphs for the Polymer Li-Ion battery were plotted using the minimum voltage measured every ten thousand samples (ten second resolution). For the evaluation, the life of a ProSpeckz is assumed to have ended once the battery voltage reaches below three volts.

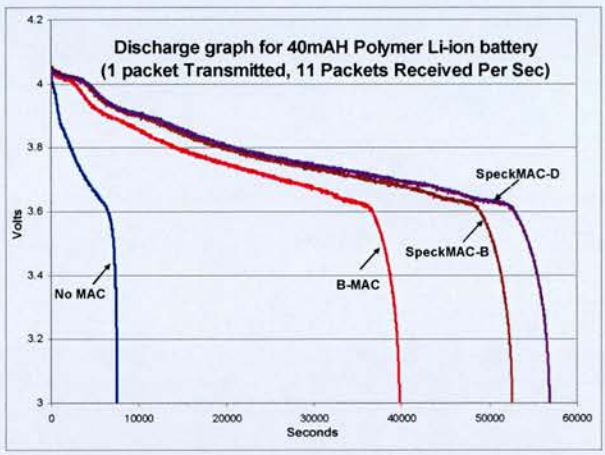


Figure 5. A discharge graph for the Polmer Li-ion battery

Over the four runs, the lifetime of the ProSpeckz running the different protocols were recorded. These lifetimes were averaged and shown in Table 3.

Table 3. The average lifetime for the ProSpeckz running on the Lithium Polymer batteries (C_V = Coefficient of Variation)

	No MAC	B-MAC	SpeckMAC-Backoff	SpeckMAC-Data
Lifetime	7875s	41263s	52568s	56943s
(σ)	288s	1424s	933s	1415s
(C_V)	3.66%	3.45%	1.77%	2.48%
Improvement over No MAC		5.24	6.68	7.23
Improvement over B-MAC			1.274	1.380
Calculated improvement over B-MAC (20)			1.376	1.463

It can be seen from Table 3 that the lifetime achieved by SpeckMAC-B and SpeckMAC-D, as measured experimentally, differs by less than 10% from the ratios calculated in Equation (20). This supports the calculations derived in the earlier sections as the difference can be accounted for by collisions, CPU energy consumption and other "real-life" parameters such as noise interference, battery characteristics and temperature fluctuation which were not modelled by the equations. It can therefore be concluded that for the logical location detection algorithm, SpeckMAC outperforms B-MAC in terms of life expectancy in both the theoretical calculation and the physical implementation.

In the next set of experiments, the Polymer Li-ion battery was replaced by two CR1220 button cells for powering each ProSpeckz. This mimics scenarios where smaller batteries are used to power miniature sensor nodes or specks. Due to their sizes, these batteries often do not have high enough sustain currents and the effects of duty-cycling could have a significant impact on the lifetime of the batteries. This experiment compares the performance of the MAC protocols under the same scenario as before, using the same location algorithm and setup as in the previous set of experiments.

The discharge graph for one of the runs is shown in Figure 6. The discharge graphs for the coin cells are plotted by using the minimum voltage measured every thousand samples (one second resolution). Unlike the case where the Polymer Li-ion batteries were used, the coin cells in the ProSpeckz fall to a deep discharge depth during the initialisation of the radio and the PSoC. A relaxation period of 5 seconds was therefore used to enable the battery to recover before the location algorithm is executed. During the recovery period, both the radio and the PSoC on the ProSpeckz were placed in a sleep mode.

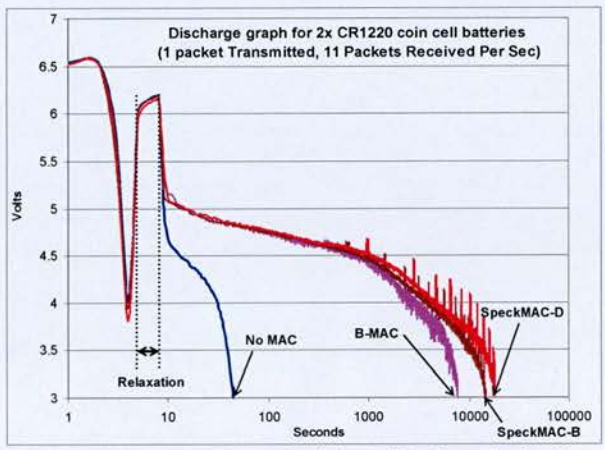


Figure 6. A discharge graph, in logarithmic scale, for the CR1220 Lithium coin cells

Over the four runs, the lifetime of the ProSpeckz were averaged and shown in Table 4. One of the reasons for the slightly higher variations between the lifetimes over different runs as compared to the Polymer Li-ion batteries could be due to the fluctuations in the battery capacity of the coin cells. Unlike the experiments with the Polymer Li-ions batteries where all the batteries used in a particular run were completely discharged and recharged in parallel to keep the capacities of the batteries as close as possible, the same approach could not be used for the coin cells as they were not rechargeable.

Table 4. The average lifetime for the ProSpeckz running on the Lithium CR1220 coin cells (C_V = Coefficient of Variation)

	No MAC	B-MAC	SpeckMAC-Backoff	SpeckMAC-Data
Lifetime	38.8s	8124.3s	14908.8s	17701.5s
(σ)	2.3s	519.8s	776.0s	685.4s
(C_V)	5.88%	6.40%	5.20%	3.87%
Improvement over No MAC		209.66	384.74	456.81
Improvement over B-MAC			1.835	2.179
Improvement over B-MAC (Polymer Li-ion, Table 3)			1.274	1.380

The results from the experiment demonstrate that the button cells deplete rapidly when a current beyond its nominal load was drawn constantly. With the radio on all the time, the button cell powered the ProSpeckz for an average of just 38.8 seconds before falling below the operating voltage. With SpeckMAC-D, the cells managed to power the ProSpeckz for over four hours, which is over 450 times longer than having the radio turned on all the time.

The most interesting results came from the improvement obtained by running SpeckMAC on the button cells. Compared to B-MAC, ProSpeckz running SpeckMAC-B and SpeckMAC-D displayed an 83.5% and 117.9% increase in lifetime over B-MAC, respectively. This improvement is significantly larger than when the ProSpeckz was powered by the Polymer Li-ion batteries, which had shown improvements of 27.4% and 38.0% over B-MAC for SpeckMAC-B and SpeckMAC-D, respectively. This is due to the fact that SpeckMAC turns the receiver on continuously for a shorter period of time when compared to B-MAC as shown in equations (5), (10) and (16). This allows SpeckMAC to minimise the discharge depth of the button cell during reception of a data packet as well as allowing it more time for recovery. This effect is not obvious in the case where the ProSpeckz was powered by the Polymer Li-ion batteries due to its ability to sustain the current draws from the radio and thus limiting the discharge depth.

From the set of experiments using the button cells, it can be concluded that for both the experiment scenarios, both variations of SpeckMAC outperform B-MAC in terms of life expectancy, and this improvement is more significant as the battery's ability to sustain the discharge load from the radio decreases.

Lastly, for all experiments, the battery-powered nodes successfully received over 95% of all the packets sent by their neighbours, regardless of the MAC protocol used.

8. CONCLUSIONS

In this paper, a low power distributed unsynchronised random-access MAC protocol, SpeckMAC, has been introduced to support low data rates in a wireless mobile ad-hoc network. The two variations, SpeckMAC-B and SpeckMAC-D, outperform B-MAC in terms of power consumption theoretically, which was confirmed experimentally. This was demonstrated by running a logical location maintenance protocol on the ProSpeckz platform. The SpeckMAC-B and SpeckMAC-D MAC protocols outperform the B-MAC by 27.4% and 38.0%, respectively when Polymer Li-ion batteries are used; and by 83.5% and 117.9%, respectively, when CR1220 button cells were employed.

9. ACKNOWLEDGEMENTS

The authors wish to acknowledge support from the Scottish Higher Education Funding Council and the UK Engineering and Physical Sciences Research Council. We would like to thank members of the Research Consortium in Speckled Computing, in particular, the Speckled Computing group at the University of Edinburgh, especially Hugh Leather and Matthew Barnes, for helpful discussions, and Allan Paterson, Department of Chemistry, University of St. Andrews, for the data on battery characterisations.

10. REFERENCES

[1] D K Arvind, "Speckled Computing", in Proc. Nanotech2005, vol 3, pp 351-354, ISBN 0-9767985-2-2, Anaheim CA, USA, May 2005.

[2] K.J Wong, D.K Arvind, "Specknets: New Challenges for Wireless Communication Protocols", in Proceedings of The IEEE International Conference on Information Technology and Applications, vol. 2, pp. 728-733, Australia , July 2005

[3] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks", In Proceedings of The Second ACM SenSys, Nov. 2004.

[4] D.K Arvind, K.J Wong, "Speckled Computing: Disruptive Technology for Networked Information Appliances", in Proceedings of The IEEE International Symposium on Consumer Electronics, (U.K), pp. 334-338, September 2004

[5] R.McNally, K.J Wong, D.K Arvind, "A Distributed Algorithm for Logical Location Estimation in Speckled Computing", in Proceedings of The IEEE Wireless Communications & Networking Conference 2005, U.S.A, pp. 1854-1859, March 2005

[6] P. Havinga and G. Smit, "Energy-efficient TDMA medium access control protocol scheduling", Proc. Asian International Mobile Computing Conference (AMOC 2000), pp. 1-9, November 2000

[7] www..bluetooth.com, "Specifications of the Bluetooth System – Core" , vol.1, v1.1

[8] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks", Proc. 21st Conference of the IEEE Computer and Communications Societies (INFOCOM), volume 3, June 2002

[9] T. van Dam, K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks", Proc. ACM SenSys'03, pp. 171-180, Nov 2003

[10] S. Singh and C. Raghavendra, "PAMAS: Power aware multi-access protocol with signalling for ad hoc networks", ACM SIGCOMM Computer Communication Review, pp. 28(3):5-26, July 1998

[11] Chipcon AS, "SmartRF CC2420 PRELIMINARY Datasheet", rev. 1.2, February 2004

[12] Cypress MicroSystems, "PSoC™ Mixed Signal Array: Technical Reference Manual (TRM)", PSoC TRM, Version 1.22, 2005

[13] Great Power-Battery Technology, Stongwill Ltd, "051213 Polymer Li-ion Data Sheet", 2005

[14] GPI International Ltd, "GPCR1220 Data Sheet", rev. 0300/765, 2005

Perspeckz-64: Physical Test-bed for Performance Evaluation of MAC and Networking Algorithms for Specknets

Kai-Juan Wong, D.K. Arvind
Research Consortium in Speckled Computing
School of Informatics, University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ, United Kingdom
k.j.wong@sms.ed.ac.uk | dka@inf.ed.ac.uk

ABSTRACT
This paper describes a physical test-bed for evaluating the performance of MAC and networking algorithms targeted at a network of specks called the specknet. The Perspeckz-64 is a workbench available to researchers to conduct experiments remotely on an ensemble of up to 64 speck prototypes called the ProSpeckz. The firmware on the ProSpeckz can be programmed wirelessly to effect different network connectivity and traffic generators, and the platform has been instrumented to collect statistical information for individual nodes as well as power consumption data for the entire network.

Categories and Subject Descriptors
B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids; C.2.1 [**Network Architecture and Design**]: Wireless communications

General Terms
Measurement, Experimentation

Keywords
Speckled Computing, Specknet, Wireless Sensor Networks, Wireless Ad-hoc Networks, Wireless Test-bed

1. INTRODUCTION
Computer simulations of networking algorithms are by definition based on models which are abstractions of the physical world. The fidelity of these models can be validated by comparing the simulations with the results of deploying the algorithms on the target physical platforms. The results of the comparisons can be used to refine the models which will nevertheless not capture all the nuances and intricacies of the physical implementation. The advantage of the simulation approach is the ease in exploring scalability issues and parametric variations.

This paper describes a physical test-bed – Perspeckz-64 - for experimenting with MAC and networking algorithms on a wireless network of up to 64 speck prototype nodes called ProSpeckz. The test-bed is intended as a resource which can be accessed remotely by the research community and has the following features:

i) The size of the physical network can be varied between 1 and 64 nodes

ii) The connectivity of the network can be changed by altering the radio ranges on the individual ProSpeckz

iii) New network scenarios and algorithms can be programmed wirelessly

iv) Statistics, such as radio and processor usage information, can be collected systematically for the MAC and networking algorithms

v) The power consumed by the network can be accurately measured and stored via a PC-based oscilloscope.

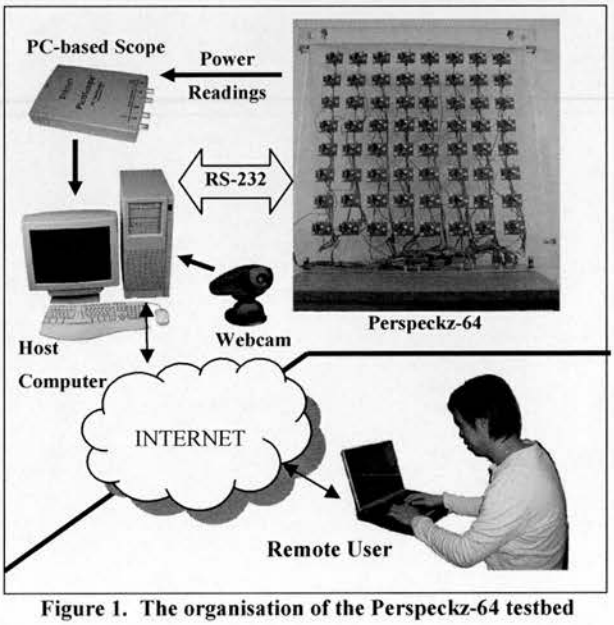


Figure 1. The organisation of the Perspeckz-64 testbed

REALMAN'06, May 26, 2006, Florence, Italy.
ACM 1-59593-360-3/06/0005.

2. THE PROSPECKZ

The Perspeckz-64 testbed is designed as a platform for developing MAC and networking algorithms for the 5X5X5mm Speck [1] currently under fabrication. The ProSpeckz [2], as shown in Figure 3, is a physically larger prototype of the miniature Speck and is intended for implementing and testing Speck applications, protocols and algorithms in the absence of the Specks. The ProSpeckz has been constructed using off-the-shelf components, including a 2.4 GHz radio with an embedded antenna, for wireless communications, and a Programmable System-on-Chip which provides the processing capability and the digital/analogue reconfigurable interfaces to external sensors. Each ProSpeckz has limited storage capabilities: 2 Kilobytes of Random-Access Memory (RAM) and 32 Kilobytes of FLASH memory.

The firmware on the ProSpeckz is shown in Figure 2. The traffic generator can be programmed by the user to mimic different network traffic scenarios, such as a constant/variable packet generator. The network and MAC layers contain the algorithms and protocols under study. The statistics layer has a book-keeping role of noting the number of packets sent and received and tracking the radio and processor usage. A boot-loader enables modifications of the MAC and networking algorithms to be programmed wirelessly onto the ProSpeckz. Furthermore, it also allows the traffic generator and the statistics layer to be modified. The boot-loader occupies the highest 4 Kilobyte of the FLASH memory (actual usage is just 2628 bytes) and 84 bytes of RAM. As the boot-loader has its own MAC/networking layer, it is not dependent upon the MAC and networking algorithms under study.

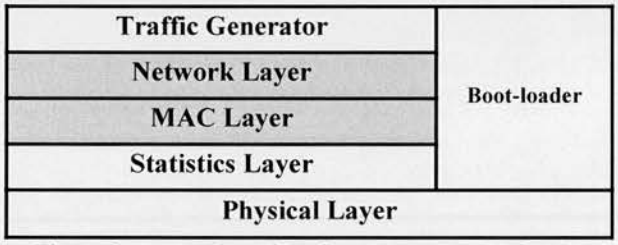


Figure 2. An overview of the firmware on the ProSpeckz

3. PERSPECKZ-64

The Perspeckz-64, as shown in Figure 3, is an array of 64 ProSpeckz nodes placed in an 8 x 8 grid with a 10cm separation between adjacent points on the grid. Two different power supplies are used to isolate the control circuitry of the Perspeckz-64 from that of the ProSpeckz which enables the current consumption of the ProSpeckz to be monitored accurately using a PC-based oscilloscope. Each ProSpeckz is attached to a relay board and can be powered by toggling the relay that is controlled by a central controller which is capable of addressing each relay board individually using a multiplexing board. In addition to controlling the power supply, the central controller is also responsible for the wireless programming of all the ProSpeckz on the testbed as well as any other ProSpeckz nodes deployed in the vicinity. In addition, the central controller interfaces to a remotely accessible host computer through the use of a RS-232 serial link thereby allowing researchers to design, program and download MAC and networking algorithms onto the physical devices remotely. Finally, the central controller polls each ProSpeckz for statistical information at the end of each experiment and transfers them onto a log file on the host computer. This log file can then be accessed remotely to determine the performance of the algorithms.

Two DOS-based commands are provided on the host computer to allow reprogramming, controlling and monitoring of the ProSpeckz onboard the PerSpeckz64 via the RS-232 serial link. These two commands and their parameters are as follows:

progme ['parameter'='value']...

Table 1. Parameters for progme.exe

Parameter	Description / value
c	The serial port on the host computer used to connect to the Perspeckz-64. E.g. COM1
r	Name of the .rom program file.
p	The poll time, in seconds, used by the central controller to poll all ProSpeckz into boot-loader mode. The default is 10 seconds.
n	The number of ProSpeckz to turn on. ProSpeckz are selected randomly. Range from 1 to 64. Default is 64.
P	The name of a power file, which is a text file, listing the grid addresses of the ProSpeckz to power on.
D	The grid address of a single node to power on. Range from 0 to 63.
a	Programs a 32-bit network address onto the ProSpeckz. Typically used with the 'D' parameter.
s	Programs an 8-bit network address onto the ProSpeckz. The higher 24-bits of the network address are randomly assigned.

results ['parameter'='value']...

Table 2. Parameters for results.exe

Parameter	Description / value
c, p, P,n	Same as progme.exe from Table 1.
r	Number of times the code has to be executed.
f	Filename of the log file that will be used to store the captured statistics.
t	Duration to run each iteration for.
s	Maximum number of ProSpeckz that is involved in the statistics collection.
l	Show long address (32 bits) in the captured statistic file if l=1, otherwise, short addresses (8 bits) are used.

Using both these commands, users can create batch files to run different experiments iteratively without human intervention. For example, the following instructions in a DOS batch file would assign an 8-bit network address to each ProSpeckz in the

Perspeckz-64 based on its grid address before it evaluates the performances of two MAC algorithms located in the program files BMAC.rom and SpeckMAC.rom. Each algorithm would be executed for 20 iterations lasting 100 seconds each with 15 ProSpeckz randomly turned on for every execution.

FOR /L %%I IN (0,1,63) DO progme c=COM4 D=%%I s=%%I
progme c=COM4 r=BMAC.rom
results c=COM4 n=15 r=20 f=bmac t=100.0 s=15 l=0
progme c=COM4 r=SpeckMAC.rom
results c=COM4 n=15 r=20 f=SpeckMAC t=100.0 s=15 l=0

4. THE DEMONSTRATION

As part of the demonstration, the Perspeckz-64 located in Edinburgh, Scotland will be exercised remotely from Florence using the remote desktop sharing functionality of Microsoft NetMeeting. Different MAC algorithms will be modified and downloaded remotely onto the ProSpeckz64. These algorithms will then be tested under different network scenarios and the results from the execution will be recorded onto a log file on the remote PC. This log file can then be accessed to show the network connectivity (number of neighbours per ProSpeckz), the packet delivery statistics (number of packet received and sent), the usage of the radio (total time that radio is transmitting, receiving and idle) and the CPU time.

5. FUTURE WORK

At the moment, the PerSpeckz64 can only be accessed via desktop sharing utilities such as Microsoft NetMeeting. In the future, a web-based interface will provide greater flexibility for accessing the Perspeckz-64 via the internet without the need for additional software at the remote site.

6. ACKNOWLEDGEMENTS

The authors wish to acknowledge support from the Scottish Higher Education Funding Council via the Strategic Research Development Grant, and the UK Engineering and Physical Sciences Research Council via the Basic Technology Research Programme. We would also like to thank Matthew Barnes and Tom Feist for their assistance in the fabrication of the PCB boards and the assembly of the Perspeckz-64.

7. REFERENCES

[1] D K Arvind, "Speckled Computing", in Proc. Nanotech2005, vol 3, pp 351-354, ISBN 0-9767985-2-2, Anaheim CA, USA, May 2005.

[2] D.K Arvind, K.J Wong, "Speckled Computing: Disruptive Technology for Networked Information Appliances", in Proceedings of IEEE ISCE'04, pp. 334-338, U.K, September 2004.

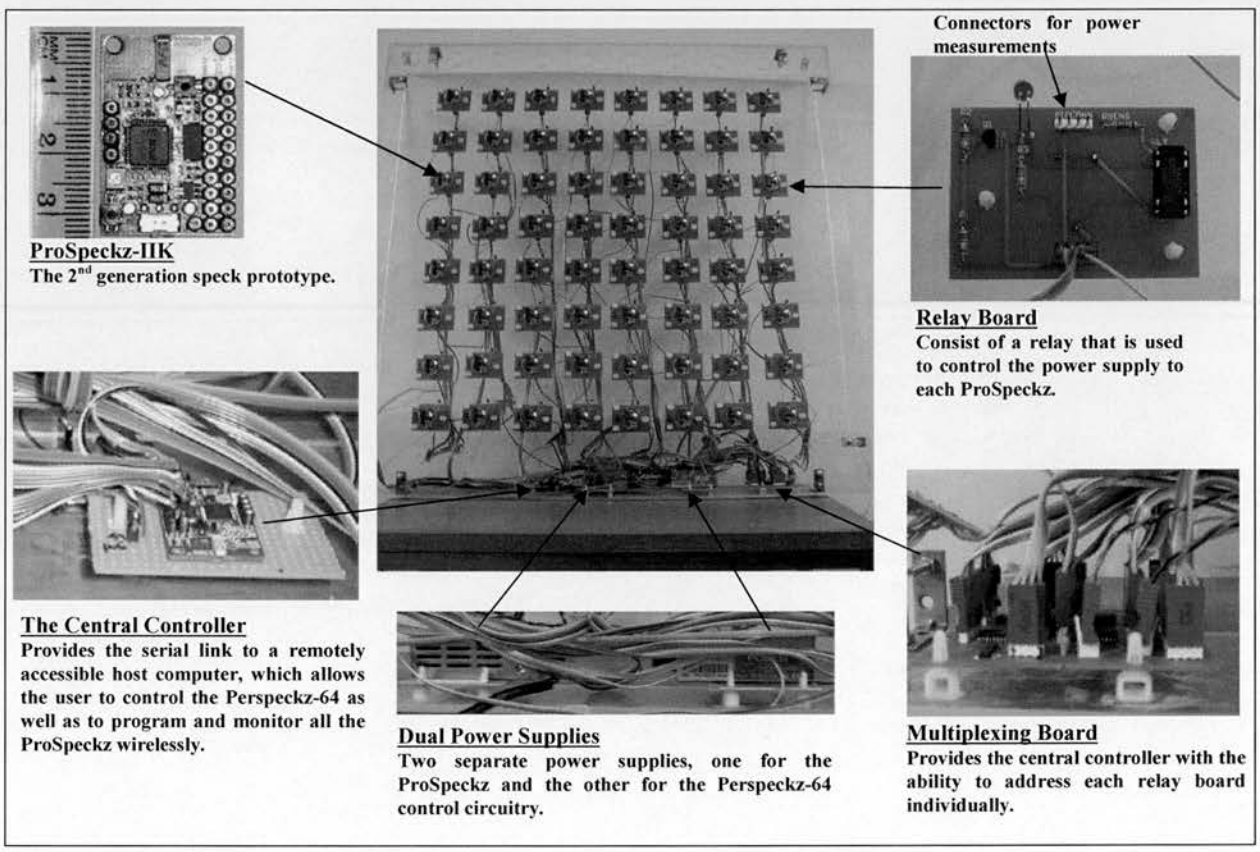


Figure 3. The Perspeckz-64 and its constituent components

Specknets: New Challenges for Wireless Communication Protocols

K.J Wong and D.K Arvind
Research Consortium in Speckled Computing
School of Informatics, University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ, Scotland, U.K.
k.j.wong@sms.ed.ac.uk | dka@inf.ed.ac.uk

Abstract

Speckled Computing [1] is an emerging technology in which data will be sensed and processed in small (around 5X5 sq. millimeter) semiconductor grains called Specks. A dense and non-static wireless network of thousands of these specks – called a Specknet - will collaborate to extract information from the data. Speckled Computing demands requirements of wireless communications in excess of typical mobile ad-hoc networks (MANET) and sensor networks. This paper presents new challenges for the design of communication protocols for specknets, in which each speck is modest in terms of energy, storage and computational resources

1. Introduction

A Speck integrates three capabilities: sensing, processing and wireless networking in a small (around 5X5 sq. millimeter) semiconductor grain. Specks are intended to be autonomous, each with its own renewable energy source, and can be mobile if needed. Thousands of Specks will collaborate as programmable computational networks called Specknets. Computing with Specknets, or Speckled Computing, will enable linkages between the physical and digital worlds with a finer degree of spatial resolution than hitherto possible. Indeed, Specknets are intended to be platforms for truly ubiquitous or pervasive computing applications.

This paper first highlights the differences between specknets and traditional sensor networks. Next, the design issues for the three layers in the communication protocol - the physical, medium access control (MAC) and network layers - are discussed. Finally, preliminary experimental results and ideas for future work are presented.

2. How is a Specknet different from typical sensor networks?

We next highlight the main differences between specknets and traditional sensor networks from the point of view of networking and communication.

a) Data-centric/Program-centric networks - The main aim of any sensor network is to sense data and transmit it back to a central hub or sink node, where it is processed and the extracted information is stored and acted upon. In contrast, a specknet does away with this centralised processing hub; each programmable speck is endowed with the capability to extract information locally in collaboration with its neighbours and act upon the information via embedded actuators in each speck. Specknets disseminate computational tasks as well as information within the network. Specknets are therefore program-centric, whereas sensor networks are typically data-centric.

b) Range of communication - Sensor networks are usually deployed over a wide area where each node is capable of transmitting to distances ranging from a few meters to a few kilometers. In the case of specknets, each speck is designed to transmit within a range of tens of centimeters. Specknets are in essence highly dense networks with a deployment area of a few square meters whereas a sensor network is a sparse network covering areas of up to tens of square meters or even a few square kilometers. The difference in the communication model is due to different costs involved in communication. In a typical sensor network, the energy cost for data transmission is higher than for data reception, whereas in a specknet the reverse is true as the ranges are relatively much smaller.

c) Mobility Model – Communication protocols designed for specknets have to assume that all specks are non-static by default. The reason for this is

twofold: specks are intended to address a new class of applications in pervasive and ubiquitous computing where mobility is an inherent feature; and given the scale of specks, even small movements could affect the communication between specks. In contrast, sensor networks are mostly treated as static networks.

d) Data transfer model – In sensor networks, nodes are classified as source nodes and sink nodes. Source nodes are ones that sense the environment periodically and the sensed data is routed to the sink nodes. In a specknet by contrast, a peer-to-peer model is used with no distinction between source and sink nodes as each Speck can dynamically be tasked to carry out different operations at various times as dictated by needs. Data transfers in a specknet are aperiodic as communication needs are highly dependent on the processing that has to be carried out.

3. The Physical layer

Specks only communicate wirelessly (radio and free-space optics are the options being considered), without cumbrous wires, to provide ease and flexibility of deployment. However, wireless communications present massive challenges given speck's extremely small form factor.

Radio communication systems can be currently classified broadly as either narrowband or wideband systems. Narrowband communication depends on sinusoidal waves transmitted continuously at some known frequency (the carrier frequency), and the data to be transmitted is modulated onto this carrier frequency. Wideband radio, on the other hand, does not require a constant carrier frequency; but instead data is transmitted by sending "impulses" of radio frequency signals that occupy very high bandwidth. Ultra Wide Band (UWB) [2] operating at a radio frequency band of between 3.1GHz and 10GHz is more suited for specks due to the simpler transceiver circuitry (although the complexity is transferred to the signal processing circuitry), possible lower power consumption and the ability to communicate using a smaller antenna.

Free-space optical communication uses light pulses to transmit data. Theoretically, the use of light as the communication medium provides for higher data rates, as light has a shorter wavelength than radio. The disadvantage however is that the medium is susceptible to obstruction: whereas radio can communicate through obstructions such as walls, optical signals would be reflected, diffused or adsorbed.

3.1. Preliminary experiment

To analyse the impact that the different shapes of areas of communication coverage provided by different mediums could have on networking, a simulation program was designed on Ns-2 [3] to compare omni-directional radio [4] communications with unidirectional infrared systems [5] in a network covering 300mm x 300mm with nodes moving using a waypoint walk. To be fair, the radio is given a range of 40mm whereas infrared has a range of 80mm with a 90 degree coverage angle. Thus, both media have equal coverage area of 5026mm^2. The simulated time is 30 seconds.

Two interesting attributes were measured in the simulations. The *number of link changes* is the number of changes in the direct link (one-hop) communication between any two nodes in the network and the *number of route changes* is the number of changes in the routes whenever a discovered shortest route between any node pairs in the network is broken.

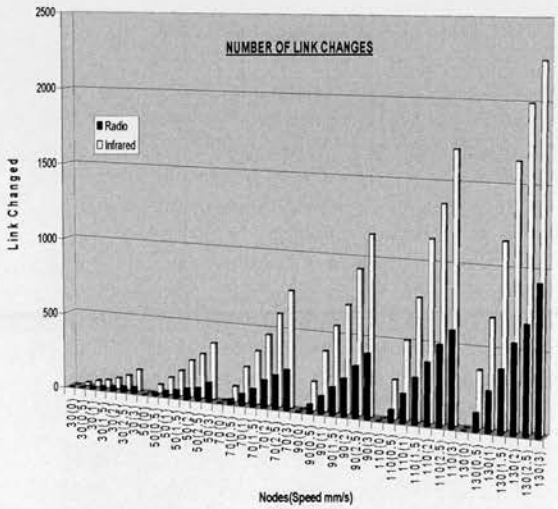


Figure 1. The number of link changes over different network densities and nodal speed

Figure 1 shows the number of link changes over the different simulation scenarios. When the network is static, no link changes would ever occur for all densities. As the speed of the nodes increases, the number of link changes would occur more frequently as expected due to the higher number of connections and disconnections. It can be observed, that the number of link changes that occur over infrared links is almost three times greater than in the case of radio links, although both cover the same area of communication.

The number of link changes that occurs in a network has a huge impact on the performance of any network routing algorithm. An example of such an impact can be demonstrated using routes calculated by the shortest path algorithm and looking at the number of route changes that occur, as shown in Figure 2.

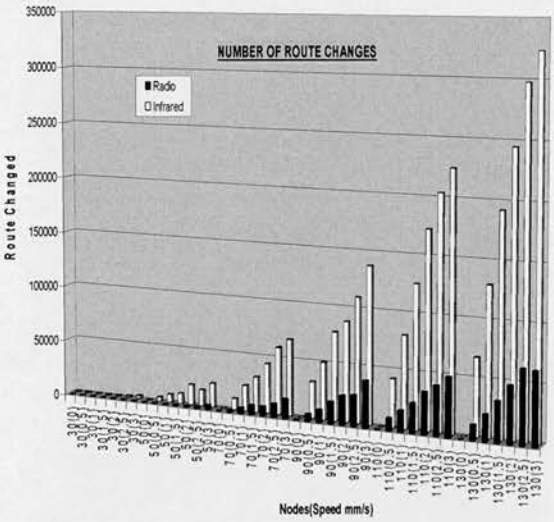


Figure 2. The number of route changes over different network densities and nodal speeds

The simulation results demonstrate that infrared systems would incur more route changes than radio systems, resulting in greater overheads in the network layer as routes are often invalidated quickly. Therefore, omni-directional coverage provided by radio communications is preferred over infrared.

3.2. Future work on the physical layer

Radio seems more suited for inter-speck communication based on the network connectivity properties. In future experiments, other attributes such as power consumption would be investigated for the two media.

4. The MAC layer

The Medium Access Control (MAC) layer manages access to the physical communication channel. It is responsible for ensuring that a given quality of service is maintained for throughput, reliability, and robustness. Given that the specks will have limited energy storage, the most important function of the MAC layer is to maintain communication between specks for as long as possible.

One way of achieving this goal is by trading power consumption with data latency by using duty-cycling of the communication channel. Several power-aware MAC schemes had been designed for MANET and sensor networks using such an approach. TDMA-based [6] protocols divides time into small slots and nodes are assigned slots in which they can communicate. These protocols require individual nodes to keep track of their neighbours' slot assignment in order to communicate in a peer-to-peer fashion, and therefore require significant memory resources. The complexity of assigning slots between nodes without base-stations or cluster-heads would also render TDMA-based protocols unsuitable for specks.

Other MAC schemes, such as PRAMAS [7], use a separate radio channel to achieve power savings. However, this requires additional hardware circuitry. Given the limited space available on each speck, such an approach would not be feasible. MAC protocols, such as S-MAC [8], have been created for sensor networks and require "loose" synchronisation between nodes. Just as in TDMA-based protocols, time is divided into large frames and each frame is divided into active and sleeping parts. Nodes can only communicate during their active parts. In order for S-MAC to operate optimally, nodes in the network would have to be either synchronised to one schedule so that all nodes wake up at the same time, or nodes would have to keep track of their neighbours' schedule. The complexity of synchronisation, either global or local within clusters, not only creates overheads (via the SYNC packets that are used in S-MAC, or the need to keep track of its neighbours' schedule) but also makes the algorithm unsuitable for dense networks in which the nodes are also mobile, as in the case of specknets.

The ideal MAC protocol for specknets should allow each speck to duty-cycle its radio transceiver and still communicate with others specks without the need for any synchronisation or coordination. The ability to communicate without synchronisation would also allow specknets to be scalable and for such a protocol to function effectively with mobile specks.

4.1. Preliminary experiment

For specks to communicate effectively using low duty-cycle rates and in an unsynchronised manner, a simple MAC – SimMAC - was designed based on the principle of using redundant retransmissions to remove the need for any synchronisation within the network. SimMAC therefore allows specknets to be

scaled and communicate in a peer-to-peer fashion in the face of node mobility.

Figure 3 shows the consumption of current in milliamperes of running SimMAC on a speck prototype – the ProSpeckz [1]. Each speck in the specknet would turn on its receiver at a fixed duty cycle, D_C, and would retransmit intended messages x times at intervals of I_C using the following formula:

$$D_C = P_R / P_{DC}$$
$$I_C < P_R - (M_{CD} * 2)$$
$$x = P_{DC} / I_C$$

where P_R is the user-defined period when the receiver is turned on, P_{DC} is the user-defined duty cycle period and M_{CD} is the maximum clock skew of the embedded oscillator on the specks.

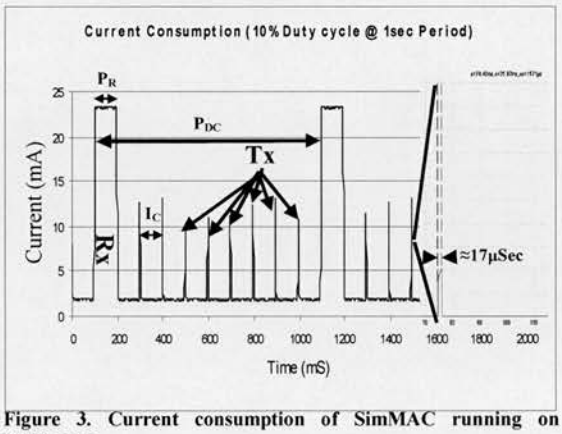


Figure 3. Current consumption of SimMAC running on ProSpeckz

The overhead required for retransmission is a function of P_R and P_{DC}, which would be reduced with a longer P_R and P_{DC}. Figure 4 shows the duty-cycling rate of the radio transceiver as compared to the intended duty-cycling rate, D_C. It is observed that when P_R is four seconds, the overhead induced by retransmission using SimMAC is negligible when the intended duty-cycle of the receiver is more than 5%.

4.2. Future work on the MAC layer

In the current version of SimMAC all specks are required to operate at a fixed duty cycling rate. However, this would present significant overheads when there is little or no traffic in the network. Therefore the ability for each speck to dynamically change its duty cycling rate based on local traffic requirements would be explored in the future. Different specks can then operate at different duty-cycling rates by keeping track of the values of P_R and P_{DC} of their neighbours.

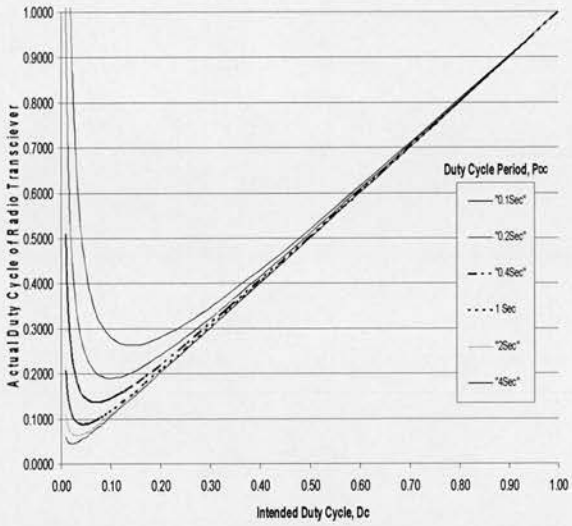


Figure 4. Comparison of actual versus intended duty-cycling using SimMAC

5. Network layer considerations and constraints

Given their small sizes, specks will be space constrained and only 2 Kilobyte of random-access memory (RAM) has been allocated for networking. This includes the memory space needed to buffer incoming/outgoing messages (the interface queues), the routing tables, and the one-hop neighbour list. This memory constraint and the properties of a specknet, as mentioned in Section 2, render many existing MANET and sensor network routing protocols unsuitable for specks. These algorithms can be grouped into the following categories:

a) Proactive (table-driven) algorithms – These algorithms [9] constantly maintain routing table(s) on each node by storing the route to any destination node in the network. The need to maintain these tables means that these algorithms are constantly exchanging data even though some routes would never be used. This overhead and the need to maintain huge network tables make such algorithms unsuitable for specknets, especially given their very limited memory.

b) Reactive (on-demand) algorithms – These algorithms [9] would discover routes only when one is needed. Even though these algorithms do not suffer from overheads caused by maintaining redundant routes, the memory required to perform route discovery is still quite considerable as the ones which are discovered need to be stored either in a cache or in the route reply packets.

c) Hybrid (proactive/reactive) algorithms – An example of a hybrid algorithm is ZRP [10] which aims to combine the advantages of both proactive and reactive routing. Each node would use proactive routing for routing packets within a certain hop distance and revert to reactive routing for destinations further away. However, as the route cache used for performing reactive routing is still limited by the memory available on each node, this algorithm would be an unsuitable choice for specknets.

d) Location-based algorithms – As the name implies, location-based algorithms [11] make use of location information to assist in route discovery. The primary problem in the case of specknets is the requirement to determine location information. This is difficult given the small size of specks, low memory and power resources, and the lack of positioning systems such as GPS. Although algorithms using triangulation or graph prediction could possibly determine relative location or logical location of specks, these would be computationally expensive and result in increases to the network overheads significantly. Furthermore, there is a need to keep location information about the destination and neighbouring nodes for this type of routing algorithms, which further increases the memory requirement for each node.

d) Data-centric algorithms – Directed diffusion [12] is a data-centric algorithm which is typically used in sensor networks. Data generated by sensor nodes is named by attribute-value pairs and routes are established by positive and negative reinforcement. Direct diffusion would suit routing in sensor networks where the sensed data is the source for all interaction and the data items can be easily quantified and identified as routing tables are looked up by interests instead of node addresses. However, in the case of highly reprogrammable computational networks such as specknets, the quantification and identification of data items or interest may not be straightforward or even possible due to the sheer numbers of data item types and the flexibility required. Also, the latency and the reliance on broadcasting would create huge initial overheads that may not be supported by the limited memory on each speck.

5.1. Preliminary experiment

In order to demonstrate the importance of memory resource requirement on routing algorithms, a simulation was designed in Ns-2 to determine the amount of memory required for the interface queues to buffer incoming packets, using DSR [9] as an example routing algorithm.

The simulation scenario is based on a network of 100 wireless mobile nodes in an area of 200 x 200mm. Nodes would pause for 1 second on reaching a random destination before selecting a new destination using a random waypoint walk. The maximum speed that the nodes will travel at is 1 millimeter per second. The nodes will communicate using the radio with a range of 40mm and a data rate of 2Mbps. At any time, there could be a maximum of 50 pairs of nodes communicating with each other in a peer-to-peer fashion with packet sizes of 32 bytes. Figure 5 shows the length of the interface queue for each node required for different transmission rates for a simulated time of 60 seconds.

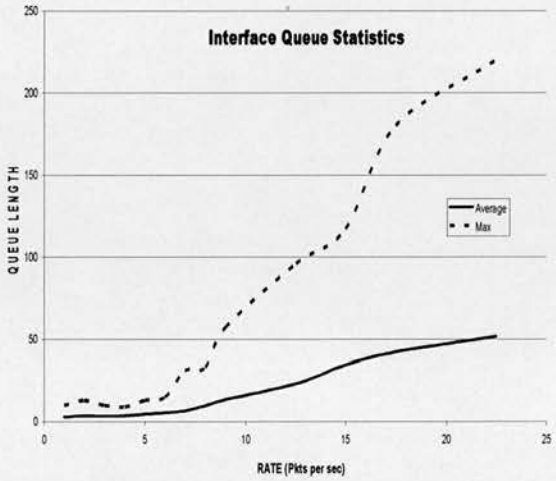


Figure 5. Simulation results showing the average and maximum interface queue length required

It was observed that in order to provide optimal performance for DSR, the memory needed for interface queues to buffer data packet could be quite large compared to the memory that is available in the specks. One point to note is that at certain hotspots in the network, there would be a requirement for a larger interface queue to sustain optimal performance as shown in the maximum queue size used by the simulation in Fig 5. Given that only 2 Kilobytes of memory is available on each speck, this could pose a possible problem when implementing DSR on specks. The same problem may occur for other "on-demand" routing algorithms, as all of them require global flooding to discover a route to a destination node. The simulation demonstrates the memory constraints that have to be considered when designing the network layer protocol for specknets.

5.2. Future work on the network layer

A novel protocol - Speckzone Based Routing Protocol (SBaRP) - is currently being designed to enable specks to route data under stringent memory requirements. SBaRP is a hybrid routing algorithm that utilises the advantages of both proactive and reactive protocols to enable routing in specknets. It is similar to ZRP, but unlike ZRP where each node has its own zone defined and maintained based on hop distances, SBaRP defines Speckzones by maintaining the number of nodes within each zone to a certain bound. Thus each Speckzone is shared by a collection of Specks where each Speck can only belong to one Speckzone. Routing within the Speckzone is done proactively whereas routing to Specks in other Speckzones would be done reactively.

As the number of Specks within each Speckzone is bounded, the routing table for proactive routing is also bounded and hence the memory space needed to store the proactive routing table can be defined. For communications between each SpeckZone, the routes which are reactively discovered are stored in distributed routing tables maintained by all Specks within the Speckzone.

SBaRP is currently being simulated on Ns-2 and is being implemented on the ProSpeckz. Further experiments and simulations would then be carried out to determine the performance of utilising the distributed memory model for the routing tables as proposed in SBaRP.

6. Conclusions

Specknets is currently in the early stages of its development, but a number of new challenges have been identified for communication and networking. These include the requirement for the wireless medium to be designed for specks with minute footprint at the physical layer, the need for an unsynchronised power-aware MAC layer and the ability for data routing to be carried out by the network layer in face of both extremely limited memory space and power onboard each speck. These constraints should present the research community with new challenges to develop novel designs and protocols which is unprecedented in the networking requirements for sensor networks and MANET.

7. References

[1] D.K. Arvind, K.J.Wong, "Speckled Computing: Disruptive Technology for Networked Information Appliances", Proc. IEEE International Symposium on Consumer Electronics (ISCE'04), September 2004, pp. 219-223

[2] Barrett, T.W., "History of UltraWideBand (UWB) Radar & Communications: Pioneers and Innovators", Proc. Progress in Electromagnetics Symposium 2000 (PIERS2000), Cambridge, MA, July 2000

[3] Ns-2 simulation, www.isi.edu/nsnam/

[4] IEEE Standard for Information Technology, IEEE Std 802.15.4-2003, 2003

[5] The Infrared Data Association, www.IrDA.org

[6] P. Havinga and G. Smit, "Energy-efficient TDMA medium access control protocol scheduling", Proc. Asian International Mobile Computing Conference (AMOC 2000), November 2000, pp. 1-9

[7] S. Singh and C. Raghavendra, "PAMAS: Power aware multi-access protocol with signalling for ad hoc networks", ACM SIGCOMM Computer Communication Review, July 1998, pp. 28(3):5-26

[8] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks", Proc. 21st Conference of the IEEE Computer and Communications Societies (INFOCOM), volume 3, June 2002, pp. 1567-1576

[9] E. Royer, C.K Toh, "A Review of Current Routing Protocols for Mobile Ad-Hoc Networks", IEEE Personal Communications, April 1999, pp. 46-55

[10] Z.J. Haas and M.R. Pearlman, "The Zone Routing Protocol (ZRP) for Ad Hoc Networks", Internet Draft, draft-ietf-manet-zone-zrp-02.txt, June 1999

[11] Y.C Tseng, C.S Hsu, "Location-Aware Routing and Applications of Mobile Ad Hoc Networks,", The Handbook of Ad Hoc Wireless Networks, CRC Press, 2003, Chpt. 18

[12] C. Intanagonwiwat, R. Gocinda, D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks", in Proc. Of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCOM 2000), August 2000

Specknet-based Responsive Environments

K. J. Wong, D. K. Arvind, N. Sharwood-Smith, A. Smith

Abstract — *Two responsive environments were implemented using prototype speck technology called Prospeckz: a behaviourally-responsive suite of furniture, and an emotionally-responsive Mood Cloud. The paper describes their implementation and their deployment at an arts exhibition in Edinburgh in December 2004. The results of a survey of the visitors to the exhibition are analysed and light is cast on issues which may be significant towards the acceptance of responsive environments in people's everyday lives.*

Index Terms — Pervasive Computing, Ubiquitous Computing, Responsive Environments, Speckled Computing.

I. Introduction

Specks are intended to be minute (about 5x5mm) semiconductor devices that will be capable of sensing the environment, processing the sensed data and communicating wirelessly with other specks to extract information in a collaborative manner. The wireless network of specks is called a Specknet, and the distributed processing of information on the programmable network is termed as Speckled Computing [1]. Everyday artefacts, the surroundings, and the people, can be "speckled", to realise networked environments of people, artefacts and surroundings which are responsive. Given their small dimensions and the autonomous nature of specks, one can reach sensing and computation, with a finer degree of granularity, to parts that could not be reached before in unobtrusive ways. Whereas it is technically feasible to realise responsive environments [4][5][6], it is still unclear how people would react to inhabiting such spaces. The influence of new technology in "Smart Homes" was investigated by a study sponsored by the Joseph Rowntree Foundation as to how people normally used their homes and worked out how different technologies could make everyday tasks easier [2]. In this paper we describe the fruits of collaboration between computer scientists at University of Edinburgh and designers at the Edinburgh College Art, which resulted in two environments being realised using speck prototypes called Prospeckz [3]. They were both presented at an exhibition in Edinburgh in December 2004, and the results of the preliminary survey of the visitors' views on the idea of responsive environment using speck technology are summarised in this paper.

This work was supported by the Scottish Higher Education Funding Council through a Strategic Research Development Grant.

K. J. Wong is with the Institute for Computing Systems Architecture, School of Informatics, University of Edinburgh, Mayfield Road, Edinburgh EH9 3JZ, Scotland, on leave from the Nanyang Technological University, Singapore. (email: k.j.wong@sms.ed.ac.uk)

D. K. Arvind is with the Institute for Computing Systems Architecture, School of Informatics, University of Edinburgh, Mayfield Road, Edinburgh EH9 3JZ, Scotland. (email: dka@inf.ed.ac.uk)

N. Sharwood-Smith and A. Smith were with the Edinburgh College of Art, Lauriston Place, Edinburgh, EH3 9DF, Scotland, when this work was undertaken. (email: n_ssmith@hotmail.com | andrewsmithdesign@yahoo.co.uk)

II. The Implementation

This section describes the implementation of two fully operational responsive environments using the current Speck prototypes, ProSpeckz (shown in Fig. 1).

The ProSpeckz were constructed using commercial, off-the-shelf components to allow rapid development of specknet-based applications. These components included an 8-bit micro-controller with 256 bytes of RAM/16 kilobytes of ROM, a Programmable System on Chip (PSoC), a 2.4GHz radio transceiver and an embedded antenna.

The Prospeckz is approximately forty times larger than the miniature specks, but despite the larger footprint it provides an ideal platform to design applications for Speckled Computing, and the experience will in turn inform the design of the miniature specks under development.

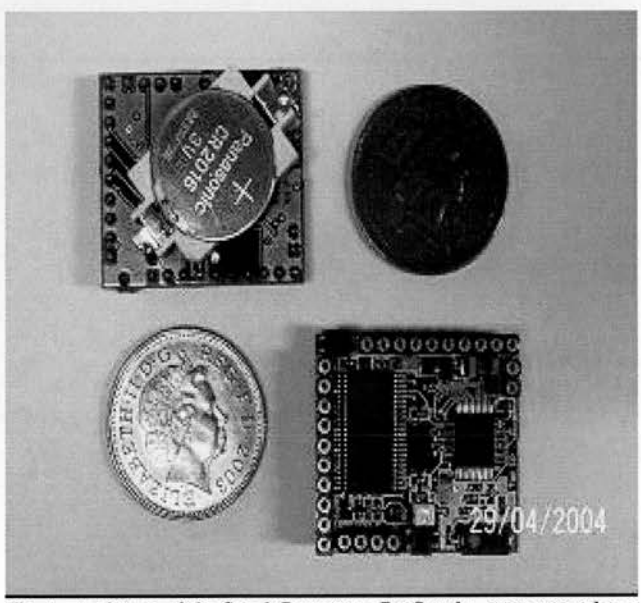


Fig. 1. A picture of the Speck Prototype, ProSpeckz, as compared to a British two pence coin

A) Smart Furniture – A Behaviourally Responsive Environment

Fig. 2. The behaviourally responsive environment displayed at an exhibition held at the Edinburgh College of Art in December 2004

Normally passive artefacts such as furniture and appliances such as the radio are incorporated with Prospeckz devices to enable them to interact with and respond to the users. The scenario of a living room shown in Fig. 2, which consists of the following furniture with enhanced functions:

(i) A reading table – when the user removes a book from the reading table, the lamp on the table or other remote lights would automatically turn on.

(ii) A smart-throw – when the user sits on the smart throw, which could be placed on a sofa or a chair, then selectable electrical appliances, such as the TV, lamps, video-recorder, would be remotely turned on.

(iii) A radio table – when the user removes the remote-control from the table, the radio embedded within the table automatically turns on. As the user moves away from the table with the remote control, the volume of the radio automatically increases.

Fig. 3 shows a simplified schematic for the smart furniture and the interaction between the different parts. Simple contact switches are used on the reading table and smart-throw to allow the ProSpeckz to detect the presence of a book and person respectively. Appliances, such as the reading lamp, television, and radio are controlled by relay switches activated by a remote ProSpeckz. This allows the user to change easily the appliances that are controlled via the reading table or smart-throw.

The radio embedded in the table is turned on directly using a normally-closed push switch that detects when the remote control is removed from the table. The remote control system for the radio is implemented using two ProSpeckz devices: one is interfaced to the radio which changes the volume as well as the radio station, and a remote ProSpeckz allows the user to remotely control the radio by sending wireless commands to the ProSpeckz local to the radio. In addition, the remote ProSpeckz broadcasts a beacon every second which allows the local ProSpeckz to gauge the distance of the remote ProSpeckz based on the received signal strength information of the beacon, and modifies the volume of the radio accordingly.

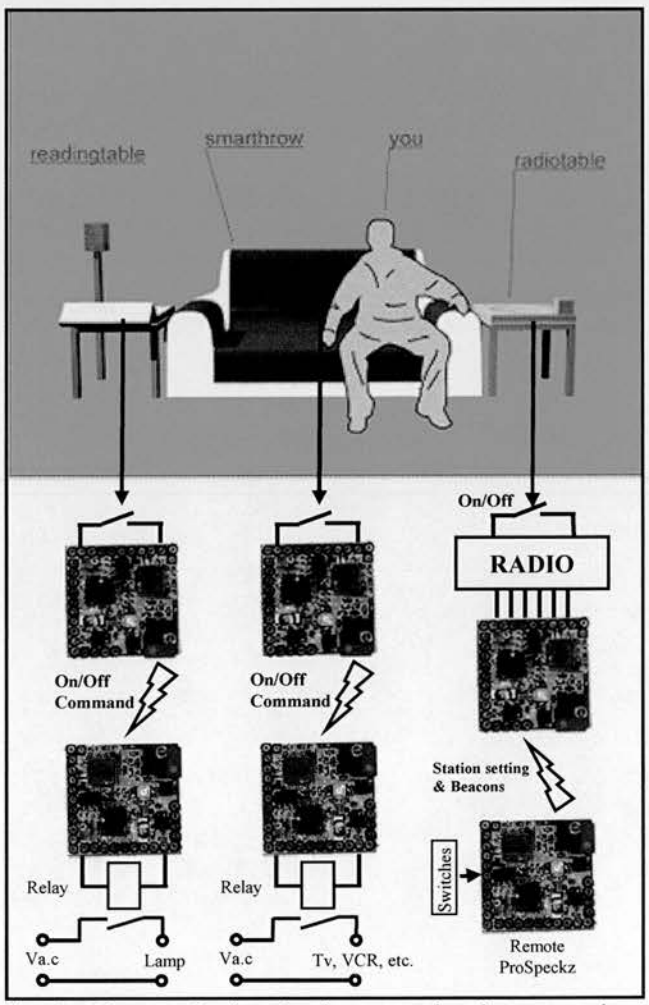


Fig. 3. Diagram showing the interconnection between various components of the behaviourally responsive environment

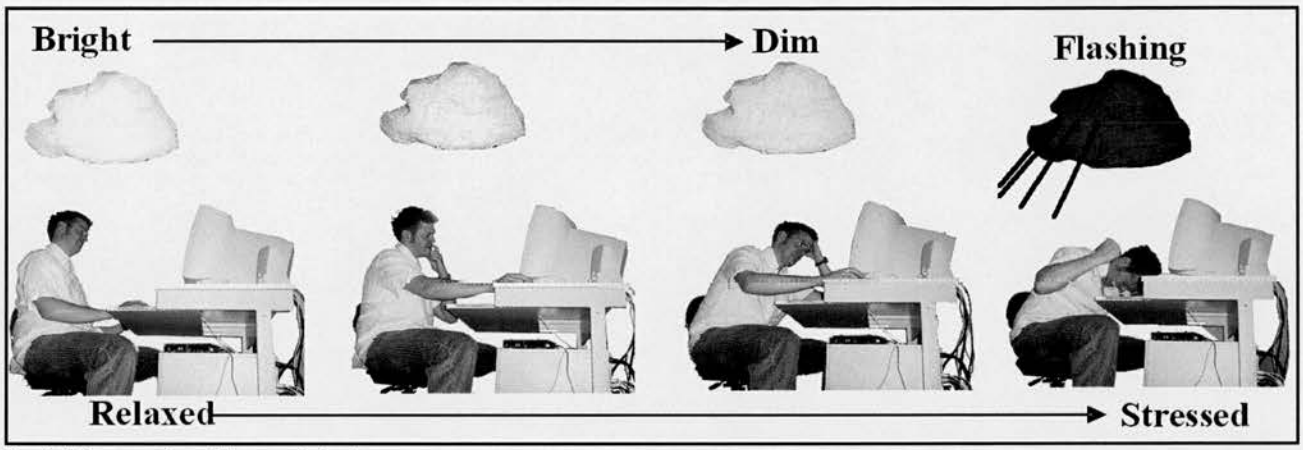


Fig. 4. The operation of the mood cloud

B) The Mood Cloud – An Emotionally Responsive Environment

The concept behind an emotionally responsive environment is to allow the immediate environment surrounding a human to react based on his or her mood. In this particular scenario, a ProSpeckz in the computer keyboard monitors the heartbeat of the user, and the lighting in a remote mood cloud is changed relative to the stress level based on the user's heartbeat. As shown in Fig. 4, as the stress level of the user increases, the light in the mood cloud dims and at a prefixed maximum stress level, the mood cloud starts to flash suggesting a break away from the computer. It could also be used as part of a feedback system to reduce the stress by playing soothing music.

The implementation of this emotionally responsive environment is shown in Fig. 5. Two copper pads on the palm rest of the keyboard are connected to an inexpensive heartbeat monitor. These monitors are normally tied across the body of the user to determine the heartbeat when exercising. As these monitors are used to detect the occurrence of a heartbeat instead of providing an accurate electrocardiograph (ECG), there is no third lead or "right leg drive" to reduce the common-mode noise. Instead, a simple peak detector is used to determine the presence of a pulse. The output of this detector is sampled by a ProSpeckz that monitors the changes in the rate of the user's heartbeat. For different thresholds, a command packet is sent to the remote ProSpeckz to change the level of lighting in the mood cloud provided by three white lamps and a flashing strobe. These lights are powered directly from the mains and are turned on using relays activated by the remote ProSpeckz.

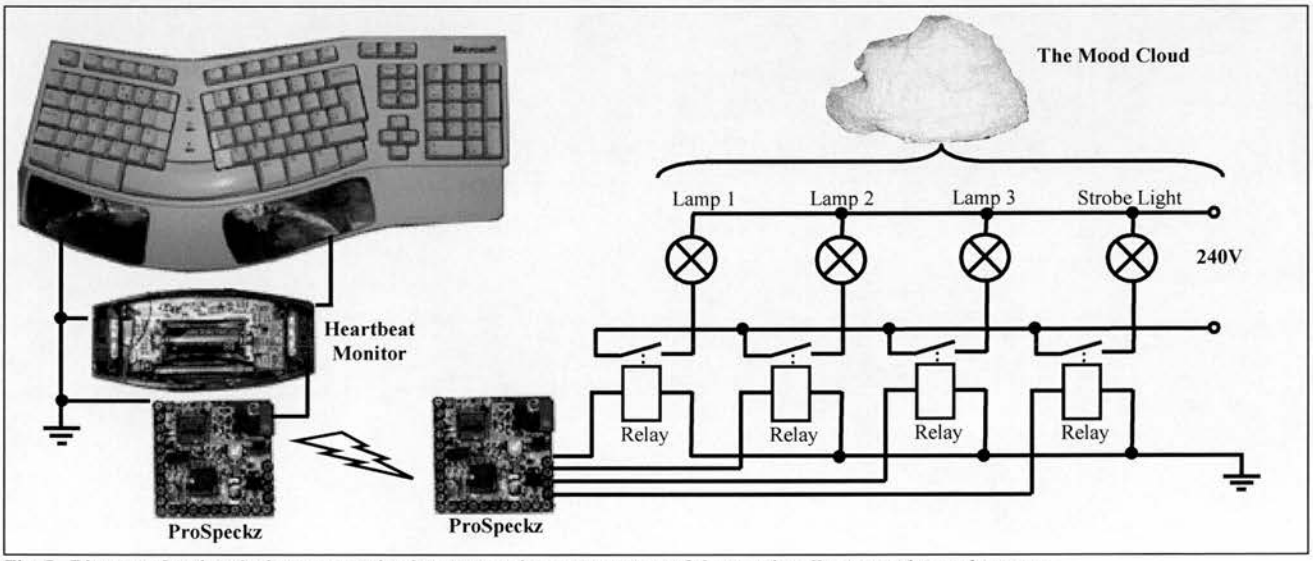


Fig. 5. Diagram showing the interconnection between various components of the emotionally responsive environment

III. Survey Results

The responsive environments were displayed as part of an exhibition at the Edinburgh College of Art in the United Kingdom, between 9th and 15th December, 2004. Visitors were asked to fill in a survey form to gauge their reaction to the implemented responsive environments. Visitors to an art exhibition (shown in Figure 6) were more likely to reflect the general mix in the population, when compared to visitors to a, say, science exhibition, who are likely to be more technically savvy.

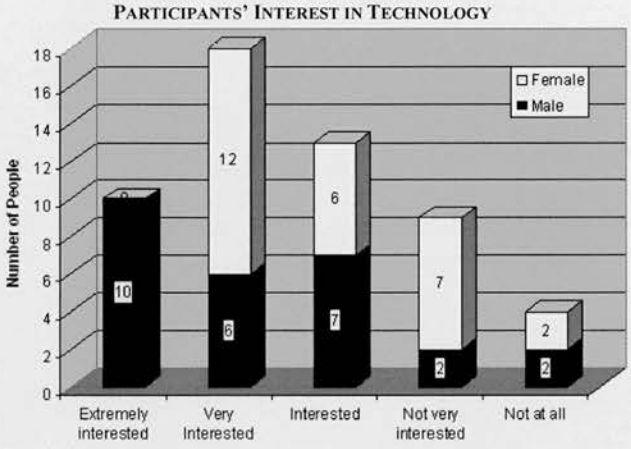


Fig. 6. Statistics on the participants' interest in technology

Fifty-four people in total participated in the survey over the week-long period of the exhibition and the demographics of the survey population , which was self-selected, is shown in Figure 7.

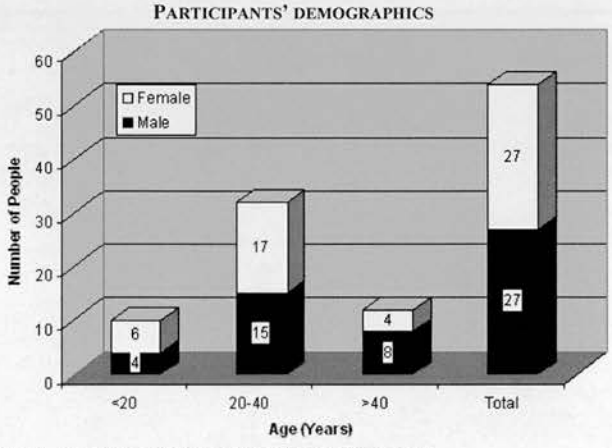


Fig. 7. Age group distribution of the participants

Two of the more important questions in the survey were:

(i) Do you think that the concept behind the exhibit would enhance your way of life?

(ii) Would you want the exhibits to be implemented in your home or your office?

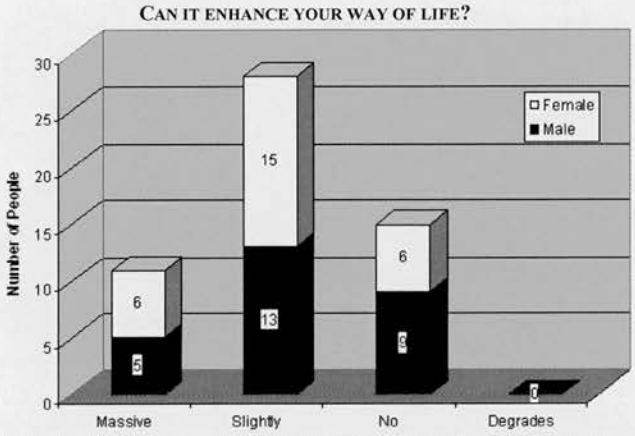


Fig. 8. Statistics on participants' perception of the responsive environments

Question (i) probed the perception of the visitors on the likely impact of responsive environments on their quality of life. As seen in Figure 8, 20% of the survey participants suggested that this technology would massively enhance their way of life, while another 52% reckoned that their way of life would be enhanced slightly. Only 28% of the participants indicated that there would be no enhancement, and most importantly, none of the participants thought that responsive environments would degrade their current quality of life.

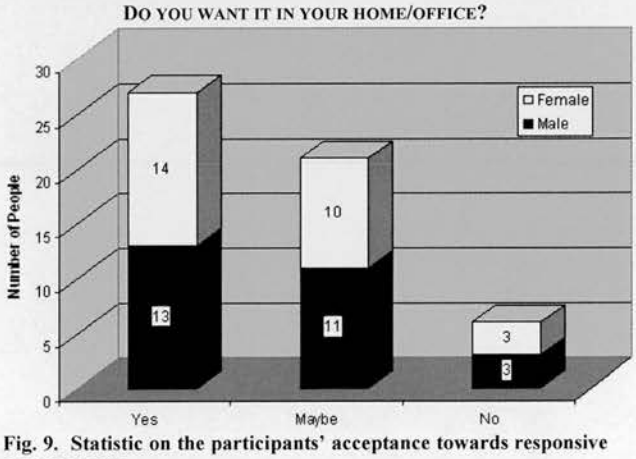


Fig. 9. Statistic on the participants' acceptance towards responsive environments

Question (ii) probed the consumers' acceptance of responsive environments into their everyday life and the survey results are shown in Figure 9. While half the survey participants were open to implementing responsive environments in their homes or offices, 39% were unsure. This uncertainty may be attributed to a number of reasons, including the consumers' reluctance to change if they were unsure of the benefits, as well as doubts about the reliability of such environments. However, only 11% of the participants were certain that they would not want

responsive environments in their homes or offices.

One interesting point that had emerged from the results of the survey was that the opinions were shared almost equally across gender lines, even though a higher proportion of the male participants were more interested in technology in general.

IV. Conclusions

Speckled Computing is an emerging technology which is anticipated to be a platform for pervasive and ubiquitous computing in the future and likely to have a significant impact on our daily lives. This paper has presented two examples of responsive environments which were implemented using prototype speck technology and deployed in an arts exhibition. The results of a preliminary survey have been presented to gauge views in the early development stages of the technology. The results of the survey, albeit limited, demonstrates that people will be amenable to the inclusion of responsive environments in their homes and offices if they perceive improvements to their way of life.

Acknowledgments

The Research Consortium for Speckled Computing has been funded by a Strategic Research Development Grant from the Scottish Higher Education Funding Council (SHEFC) since October 2003. The authors wish to thank SHEFC for their support, The Edinburgh College of Art for their co-operation, and the members of the Consortium, especially the Speckled Computing Group at Edinburgh, for helpful discussions.

References

[1] D.K. Arvind "Speckled Computing", Invited Paper, Proc. Nanotech 2005, Vol. 3, pp 351-54, ISBN: 0-9767985-2-2, May 2005.

[2] David Gann, James Barlow, Tim Venables, "Digital Future: Making homes smarter", Chartered Institute of Housing and the Joseph Rowntree Foundation, ISBN: 1 900396 149, 1999.

[3] D.K. Arvind, K.J.Wong, "Speckled Computing: Disruptive Technology for Networked Information Appliances", Proc. IEEE International Symposium on Consumer Electronics (ISCE'04), pp. 219-223, September 2004

[4] D. Estrin, A. Kerpa, "ASCENT: Adaptive Self-Configuring Sensor Networks Topologies", In IEEE Transactions on Mobile Computing, Special Issue on Mission-Oriented Sensor Networks. Vol. 3, No. 3, July-September 2004.

[5] M. Laibowitz, J. Paradiso, "The UbER-Badge: A Versatile Platform at the Juncture between Wearable and Social Computing", Advances in Pervasive Computing, Eds. A. Ferscha et al, Oesterreichische Computer Gesselschaft, 2004, pp263-8.

[6] J. Paradiso, "Wearable Wireless Sensing for Interactive Media", Invited Paper, In Proc. of the First International Workshop on Wearable and Implantable Body Sensor Networks, Imperial College, London, April 6-7 2004. http://vip.doc.ic.ac.uk/bsn_2004/home/

Kai Juan Wong became a member of IEEE in 2001. He graduated from Nanyang Technological University (NTU), Singapore, with a B.A.Sc. (Class I) in 2001 and was awarded the Compaq Gold Medal for graduating first in general proficiency throughout his course of study. Upon graduation, he joined NTU as a Senior Tutor. He is currently on study leave, pursuing his PhD at the University of Edinburgh, U.K since 2003. His research interests include wireless communications in mobile ad-hoc networks, sensor networks and embedded systems.

D. K. Arvind is a Reader in the School of Informatics at the University of Edinburgh, and the Director of the Research Consortium in Speckled Computing, a multidisciplinary grouping of computer scientists, electronic engineers, electrochemists and physicists drawn from several universities. The Consortium has received funding from the Scottish Higher Education Funding Council since 2003, and from the UK Engineering and Physical Sciences Research Council since 2005. His research interests include the design, analysis and integration of micro-scale networked embedded systems which combine sensing, processing and wireless communication capabilities.

Nick Sharwood-Smith (BSc (Hons), MDes) was born in Edinburgh on the 11th Aug 1981, educated at George Watsons College, then studied Industrial Design at Napier University and exhibited work at the New Designers show in London'03, followed by study at the Edinburgh College of Art to gain a MDes degree in Product and Furniture design, for which he was awarded funding. He is continuing to work with the Speckled Computing Consortium on novel design ideas whilst pursuing other design work.

Andrew Smith is a young designer with a quirky imagination and a passion for his work. Andrew designs furniture and products, which have an emotional meaning to the people who use them. He produces highly evolved designs of intelligence and beauty, which invigorate the user, and the world of design. In collaboration with the Research Consortium in Speckled Computing, Andrew is continuing to design objects which have an emotional relationship with the user incorporated into the process of interaction.

A Distributed Algorithm for Logical Location Estimation in Speckled Computing

R. McNally, K.J Wong, D.K Arvind
Institute for Computing Systems Architecture
School of Informatics, University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ, Scotland.
r.mcnally@sms.ed.ac.uk | k.j.wong@sms.ed.ac.uk | dka@inf.ed.ac.uk

Abstract — *Speckled Computing [1] is an emerging technology in which data will be sensed in minute (eventually one cubic millimetre) semiconductor grains called Specks. Information will be extracted in situ from each Speck and will be exchanged and processed in a collaborative fashion in a wireless network of thousands of Specks, called a Specknet. Specks are not assumed to be static and therefore estimating and maintaining the logical location of the mobile Specks in a network would be essential for a number of Speckled Computing and sensor network applications. A novel lightweight distributed algorithm is introduced in this paper for this purpose and simulation results are presented to determine the goodness of the algorithm for different parameters. The algorithm was also successfully ported to a hardware prototype of the Speck called the ProSpeckz. The problems and issues of porting the algorithm onto such a resource-constrained hardware platform will also be discussed. Finally, the paper concludes with plans for future work to be carried out to improve the algorithm.*

Index Terms — Logical Location Estimation Algorithm, Speckled Computing, Intelligent Sensor Networks

I. Introduction

A Speck is intended to integrate sensing, processing and wireless networking capabilities in a minute semiconductor grain, which will ultimately be the size of a pinhead. Specks will be autonomous with their own energy source and will be assumed to be mobile. Thousands of Specks are designed to collaborate as a programmable computational network called a Specknet. Specknets are intended to be a generic platform for ubiquitous computing, wherein data is sensed processed and information is extracted *in situ* in a collaborative fashion. Some Specknet applications require the estimation and maintenance of the location of the location of each Speck node. Sensor data may have little or no value if it is not informed by the location where it was sensed; indeed, the location of each node in the network might very well be the data to be sensed.

Computing the location of nodes in such a dynamic wireless network is not without difficulties. Specks are typically feeble in terms of computation and storage, communication between nodes will be unreliable and relatively expensive, and the network can be of any size and density. A good solution to this problem should require minimal computation and storage, be robust against unreliable communication, and be fully distributed across the network. This paper presents such an algorithm which meets these criteria for estimating the location of each Speck locally, which is based on two-hop neighbour information, in the face of movement and importantly does not use received signal strength as an indication of range between Specks.

In the rest of this paper, Section II outlines existing algorithms for location discovery and their unsuitability for Specknets; Section III describes the proposed algorithm; Section IV describes the simulator and the metrics chosen to evaluate the goodness of the algorithm; Section V details the simulation scenarios and discusses the results; Section VI describes the implementation of the algorithm on a hardware prototype of a Speck called the ProSpeckz and the problems involved in porting to such a platform with plans for future work and conclusions outlined in Sections VII and VIII, respectively.

II. Related Work on Location Discovery

Existing work in this area has understandably focused on location discovery, and one of the ways of maintaining location data is to repeatedly run a location discovery algorithm. However, most methods for location discovery have drawbacks that would preclude their use for Speckled Computing.

For instance, the Lighthouse location System [2], proposed for use with Berkeley's Smart Dust, requires a base station equipped with rotating beacon lights and a clear line of sight to every node in the network. This approach is clearly unsuitable for applications in which the Specknet must be unobtrusive.

Doherty et al [3] describes another approach that also requires a base station. This approach coalesces the connectivity data of the entire network into the base station, where it is processed and the resultant location data is then dispersed back into the network. Such an approach also suffers from the lack of scalability. As the network size increases, the network latency and energy requirement of collecting and disseminating such a large data set, along with the computation time, would prove inefficient.

0-7803-8966-2/05/$20.00 (C) 2005 IEEE

The system proposed by Bulusu et al [4] does not require a central base station but does require an infrastructure of beacon nodes to be placed with known and fixed positions. A node's position can then be estimated based on which of the beacon nodes can be contacted by radio. However, the ad-hoc nature of SpeckNet disallows the use of such an infrastructure.

Other algorithms [5], [6] rely on Received Signal Strength Information (RSSI) in order to estimate the location. However, RSSI is notoriously unreliable indicator of physical distances, as they are affected by multi-path radio propagations and interferences. Specknets aspires to be a truly generic technology, with the ability to be deployed and to adapt in any environment without the need for additional calibration.

III. THE PROPOSED ALGORITHM

The proposed algorithm presented here is a simple distributed algorithm that enables each node to estimate its own logical position (or logical location). A logical position is defined as a coordinate in two/three-dimensions that is relative to the logical positions of the other nodes. Thus, a set of logical positions would give the layout of the nodes in a network without the physical effects of rotation and translation. To simplify the notations, all locations or positions referred from this point onwards in this paper would be logical coordinates instead of physical coordinates.

The basic premise of the proposed algorithm is the use of inclusive and exclusive distance constraints on a node's position. By using the estimated location of the nodes within a two-hop distance, a node is able to estimate its location by including and excluding areas in which it could possibly be. If node *A* can communicate with nodes *B* and *C*, then *A* must be located within a distance *r* of their positions, for a given radio range *r*. This forms the inclusive distance constraint. On the other hand, exclusive distance constraints on a node's position can be inferred from the nodes that cannot be contacted. In Figure 1, if *A* cannot communicate directly with *D*, then it must be located at a distance further than *r* from its position. Thus, by applying the inclusive distance constraints with *B* and *C* and the exclusive distance constraint with D, the set of possible positions for *A* is restricted to the shaded segment.

The algorithm can thus pithily be summarised as "move towards your neighbours, move away from your neighbour's neighbours who are not your neighbours". A more comprehensive description is as follows:

a. Each node maintains a neighbour list of its one hop neighbours' identifiers (id) and positions. The list degrades over time and is updated when the node receives a location information packet.
b. Each node will periodically broadcast a location information packet that consists of its id and position, along with the contents of its neighbour list.
c. Upon receiving such a packet, the receiving node will:
   - Update its neighbour list with the sender's details
   - For every entry in the neighbour list, satisfy the inclusive distance constraint on the node's position.
   - For every node detailed in the received message that is not in the neighbour list, satisfy the exclusive distance constraint.
d. Satisfying an inclusive constraint implies moving the node's computed position closer to the position of the other node involved in the constraint. For instance, if node A receives a broadcast from node B that has a maximum range of 10 units, but A's and B's computed positions are such that the distance between them is 16 units, then A will estimate its position closer to B's. To fully satisfy the constraint would require A to be moved 6 units closer to B, but since A and B are equal partners in the constraint, A should only move 3 units closer. The remaining distance should be moved by B the next time B receives a broadcast from A.
e. Satisfying an exclusive constraint is almost identical to the inclusive case, where instead of pulling nodes closer, they are pushed apart.

Computing an exact solution to satisfy each node's set of constraints requires complex calculations. However, the proposed algorithm here uses a disarmingly simple iterative approach. Each node will satisfy each of its constraints in turn by moving its computed estimated position, and over time the network as a whole will converge to a steady and valid state after several iterations even though each node begins with an erroneous estimation of its own location.

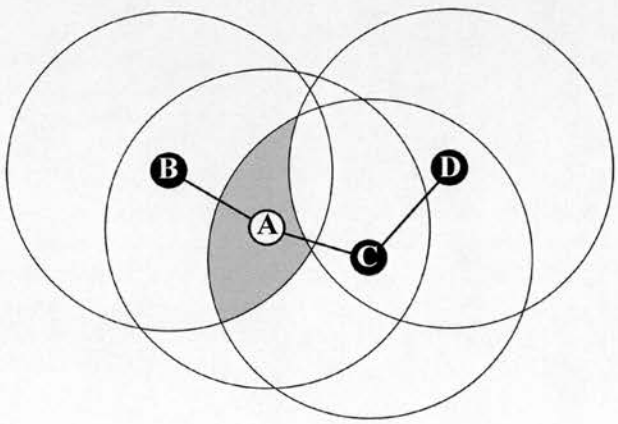


Figure 1. The use of inclusive and exclusive constraints to estimate the possible position of node A

0-7803-8966-2/05/$20.00 (C) 2005 IEEE

IV. THE SIMULATOR

The performance of the algorithm was simulated on a Java-based stimulator. The assumption was made that radio propagation is spherical, with a known and uniform range. The movement model used is one of random waypoints. Initially, the nodes are scattered over a unit square. Each node chooses a random destination point and moves towards it at a defined maximum speed. Upon reaching the destination, a new waypoint is chosen, and the operation continues. The challenge is to maintain each node's location data using minimal resources.

The performance metrics measured on the simulator are:

a. ***Raw error*** - The distance between where a node actually is and where it thinks it is. This is a measure of the error in the actual physical location.

b. ***Aligned error*** - The distance between where a node actually is and where it thinks it is, but disregarding the effects of any network-wide rotation and translation. This is used to measure the internal accuracy of the network layout. This is the measure of the error in the logical location.

c. ***Trajectory-based routing (TBR) statistics*** - One possible use of the location data is to aid in packet routing decisions. For example, if node *A* wishes to send a packet to some distant node *B*, with a known location, then it can use the location data to decide which of its immediate neighbours the packet should be routed through. There are three different variations of TBR implemented on the simulator:

- Maximum Distance - The packet is routed to the node that makes the most progress towards node *B*.
- Minimum Distance - The packet is routed to the node that is closest to node A, while still making some progress towards node B.
- Closest Path - The packet is routed to the node that lies closest to the line between node *A* and node *B*.

The performance of TBR using the estimated logical location can then be quantified in two ways:

- In comparison with perfect location information: For every decision in the routing algorithm, it is run once based on the source node's neighbour list, and again based on perfect information of the network. If the two choices agreed, then the correct decision was made; if they disagreed, then the routing decision was incorrect.

- In comparison with the actual progress that each decision makes: For every decision in the routing algorithm, a node chooses a neighbour that it believes will be most likely to make progress towards the destination based on the calculated logical positions. If the chosen neighbour was physically closer to the destination, then the choice was correct, or incorrect, otherwise.

V. SIMULATION SCENARIO AND RESULTS

All the simulations were run on a 1 unit square area with the radio range of each node being set to 0.2 units. Each set of results would be based on scenarios generated from densities ranging from 10 to 200 nodes per unit square (in 10 node steps) and speeds between 0.0 to 0.2 units per message transmitted (in 0.01 unit steps). Each scenario was executed 20 times and the results were averaged. One point to note is that the speed is defined in the results as the unit distance travelled per message transmitted (unit per message) and is a normalisation of transmission rate (message per second) and velocity (unit per second).

Two sets of simulation results are next discussed. Firstly, the aligned error caused by the algorithm is shown in Figure 3. The maximum error that is possible in the worst-case scenario is approximately 0.35 units due to the geographical constraints of the simulations. The worst performance occurs when the density is 10 nodes per unit square. Moving at speeds of about 0.2 units per message, the algorithm in this case would reach the maximum error. As the density increases, the performance of the algorithm improves. The best possible results are presented when densities are above 90 nodes per unit square. The error rates are almost linear to the speed. From the results, one can surmise that the location algorithm presented in this paper would perform optimally in networks with nodes having approximately 11 neighbours (Radio Coverage area * density = $(3.14*0.2^2)*(90/1) = 11$). Therefore the algorithm should perform well for Speckled Computing applications that require highly dense, mobile networks.

Logical location information can also be used for location based network routing algorithms such as those presented in [7], [8], [9]. The simulation results in Figure 4 are based on TBR, which demonstrates the utility of the proposed algorithm for making routing decisions. Each node will forward messages towards the destination, and if the node that it forwards to is physically closer to the destination then the decision is deemed to be correct, or wrong, otherwise. For each simulation run, the number of correct decisions made is then divided by the total number of decisions made to give the probability of a correct decision.

The simulation results presented in Figure 4 demonstrates that in a static network, the proposed algorithm works perfectly by making the correct decisions all of the time. As the speed increases, the performance of the algorithm degrades: this decrease is initially gradual – at speeds of up to 0.06 unit per message, correct decisions are still made 95% of the time, for a density greater than 110 nodes per unit square; above this

0-7803-8966-2/05/$20.00 (C) 2005 IEEE

speed, the performance drops much faster. The drop stabilises at a speed of 0.16 units per message, when 60%-55% of correct decisions are made most of the time. One point to note is that even at speeds of 0.2 units per message, the estimated logical position information still provides the routing algorithm with some assistance, and outperforms a random guess (probability of 0.5).

VI. Implementation on the ProSpeckz (Speck Prototype)

The feasibility of the algorithm was investigated on a hardware Speck prototype called the ProSpeckz (Programmable Specks over Zigbee Radio) which the size of a quarter credit card. Figure 2 shows a picture of the ProSpeckz and the layers within the prototype. ProSpeckz contains a Programmable Systems-on-chip (PSoC), which is an 8-bit microcontroller with 16Kbytes of ROM and 256 bytes of RAM. The PSoC also allows the user to reconfigure hardware analogue circuits such as ADCs, DACs, amplifiers and filters on-board the ProSpeckz under software control. This allows the ProSpeckz to be extremely flexible and easy to interface with existing electrical infrastructure. The ProSpeckz is fitted with a 2.4GHz radio for communication at a data rate of 250kbps, with an adjustable range between 20 centimetres and 20 meters, using an on-board antenna.

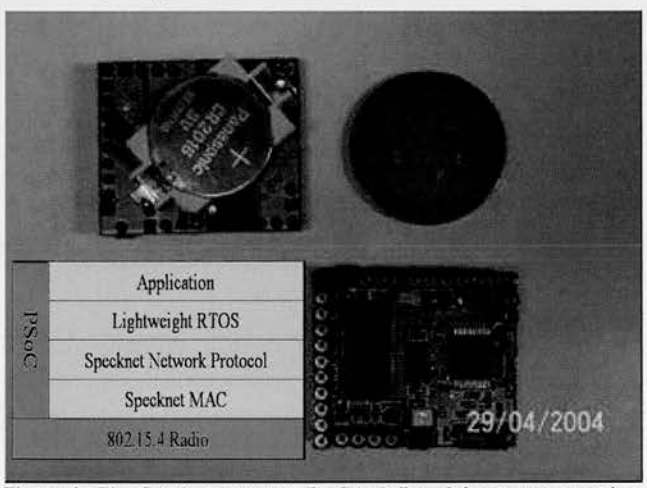


Figure 2. The Speck prototype "ProSpeckz" and its system overview diagram

Porting the algorithm to ProSpeckz presents a number of challenges. Real-world phenomena such as radio interferences and multi-path propagations will be present which are often unpredictable. Furthermore, a memory limitation of only 256 bytes in the PSoC limits the amount of local information that can be stored for location discovery and radio communication. The limited memory also restricts the complexity of the location estimation algorithm that can be implemented.

Despite all the above-mentioned constraints, the location estimation algorithm presented has been successfully implemented on the ProSpeckz. However, several problems have emerged during the development. These problems and the solutions implemented will be discussed briefly.

a. ***Asymmetrical Inclusions*** –This occurs when a node A is able to hear a distant node C, which is theoretically out of its range. But, Node C cannot hear A. Thus, when A receives a packet from C, A will pull its location towards C, which is an erroneous decision. To overcome this phenomenon, an additional constraint is placed such that before A pulls itself towards C, it will check whether it is on the list of neighbours detailed by the location information packet sent by C. This enforces symmetrical link constraints on the algorithm.

b. ***Somnambulism/New Nodes*** – It is possible for a node to have to switch itself off for a while due to lack of power. After it has scavenged enough energy from the environment, through the use of a photovoltaic source or some similar renewable energy device, it will be able to resume its activities. This scenario presents an interesting challenge for the algorithm. The node may have moved during its sleep cycle, and so its knowledge of its own location will have become out-of-date. This problem will also occur when a new node joins the network. If the node then broadcasts its location information, as detailed in the algorithm, it will very likely lead to an increase in the error for the neighbours as its stale location estimate will adversely affect the neighbours' own estimates about their current positions. Instead, an alternative approach has been implemented such that the node would clear its location data and neighbour table when it awakens/starts. It will then listen to the broadcast of the other nodes and only resume broadcasting either after it can make a sufficiently good initial guess of where it is, based on the transmissions of its new neighbours, or after a timeout.

c. ***Rotation and Translation Errors*** – It was observed both in the simulator and in the hardware implementation that a global rotation and translation occurred. This caused the logical locations to change even though they were still logically correct at any time. This phenomenon can be overcome in two possible ways. Firstly, beacon nodes with fixed positions could be placed in the network. This will place some weight on the logical positions. For example, if 3 beacons were placed at 3 different physical corners of the network, rotation and translation errors would cease to occur. The second method solves the problem without the need for beacons. The solution is to make nodes "selfish" once their estimates are stable for some time. "Selfish" nodes will not recalculate their positions as long as there are no changes to its 1-hop neighbour list.

d. **Erroneous data packets and Malicious nodes** – It is possible that nodes receive erroneous data from other nodes, either due to poor radio connectivity or via

0-7803-8966-2/05/$20.00 (C) 2005 IEEE

malicious nodes. To limit the damage caused by such data, a constraint is placed on the algorithm that it would discard any location information that would cause it to shift drastically. This is not foolproof and other methods for detecting erroneous data packets and malicious node will be the subject of future research.

VII. Future Work

6.1 Modelling and investigating the effects of realistic radio models in a large scale network

The simulator will be developed to investigate the effects of realistic radio propagation models on a large Specknet. The radio model implemented currently in the simulator is idealistic in assuming a spherical maximum extent, which is simply not the case in the real world, as demonstrated in the physical implementation on the ProSpeckz.

The addition of unreliable radio communication and radio-opaque barriers in the network will be detrimental, but it is anticipated that some simple modifications to the algorithm can be made to mitigate the degradation. Currently, if two nodes cannot communicate, the algorithm makes the simple assumption that this is because they are too distant. Relaxing this assumption should compensate somewhat for imperfect radio communication and would be investigated in the future.

6.2 Evaluation of the algorithm for location discovery

In the results reported in this paper, the simulations have been carried out to evaluate the location maintenance capability of the algorithm. The nodes start life with correct location information. The challenge addressed here is to maintain that information in the face of movement. Future work will investigate how this algorithm would also perform location discovery.

6.3 Movement models

The waypoint movement model implemented in the simulator was chosen due to its simplicity of implementation and fairly pathological nature. It is unlikely, however, that this movement model will be encountered in the real world. It would therefore be useful to determine the performance of the algorithm in more realistic situations. For example:

- Having chunks of the network moving around as a unit, to simulate the effects of nodes being attached to some rigid body that then has some degree of motion, such as an articulate body
- Nodes in a turbulent fluid
- Networks in a mixture of static and mobile nodes
- Nodes and networks that have constrained movement.

The algorithm presented in this paper operates on a two-dimensional plane but need not be limited to one. Adding a third dimension would be straightforward, and it would be interesting to see how the algorithm performs in a virtual 3D space.

6.4 Use of sensors to enhance location estimations

Most location algorithms focus on the radio as the tool for distance measurements, but there are a number of other sensors that lend themselves naturally to this area. One such is the accelerometer, which can also be used to dynamically adjust the transmission rates of the packets used for location discovery. As discussed in section V, as the velocity of the nodes increases, the transmission rates of the location information packets would also have to be increased proportionately to maintain a level of quality in the location estimates. Conversely, as the velocity of the nodes decrease, the transmission rates could be lowered to both save power and decrease the network overheads.

VIII. Conclusions

A novel lightweight distributed logical location approximation algorithm has been described. Simulation results have been presented to qualify the goodness of algorithm against different parameters. The efficacy of the algorithm was tested on a hardware prototype of the Specknet. Although the algorithm has been developed and demonstrated in the context of Specknets, it is sufficiently general for use in traditional sensor networks, vehicular tracking, and robotics, without the need to use expensive Global Positioning System (GPS) devices in applications where logical location estimates would be sufficient.

IX. Acknowledgements

The Scottish Higher Education Funding Council as part of the Strategic Research Development Grant funds the Research Consortium for Speckled Computing. The authors wish to acknowledge SHEFC for their support, and thank the members of the Consortium, especially the Speckled Computing Group at Edinburgh, for helpful discussions.

References

[1] D.K. Arvind, K.J.Wong, "Speckled Computing: Disruptive Technology for Networked Information Appliances", Proc. IEEE International Symposium on Consumer Electronics (ISCE'04), pp. 219-223, September 2004

[2] Kay Romer, "The Lighthouse Location System for Smart Dust," *Proc. MobiSys'03*, pp. 15-30, May 2003.

[3] L. Doherty, K. Pister, and L. El Ghaoui, "Convex Position Estimation in Wireless Sensor Networks," Proc. IEEE Infocom 2001, April 2001.

[4] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less Low Cost Outdoor Localization For Very Small Devices," IEEE Personal Communications, Vol. 7, No. 5, October 2000.

[5] D. P. Robinson & I. W. Marshall , "An Iterative Approach to Locating Simple Devices in an ad-hoc Network ," Proc. London Communications Symposium 2002, London, September 2002.

[6] J.Hightower, G.Borriello and R.Want, "SpotON: An Indoor 3D Location Sensing Technology Based on RF Signal Strength," UW CSE Technical Report #2000-02-02, February 2000.

[7] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," Proc. ACM/IEEE MobiCom 2000, August 2000.
[8] Y. Ko and N. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," Proc. ACM/IEEE MobiCom'98, October 1998.
[9] S. Basagni, I. Chlamtac, V.R. Syrotiuk, and B.A. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," Proc ACM/IEEE MobiCom'98, pp. 76-84, October 1998.

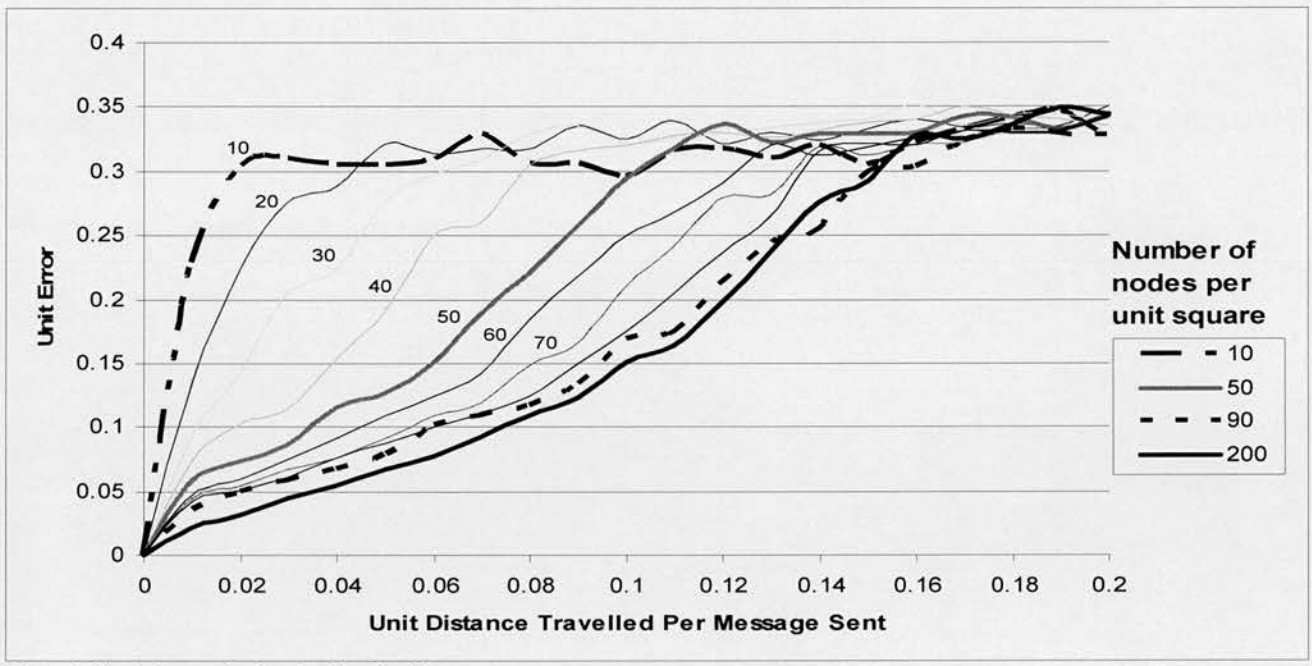


Figure 3. Simulation results showing the aligned error

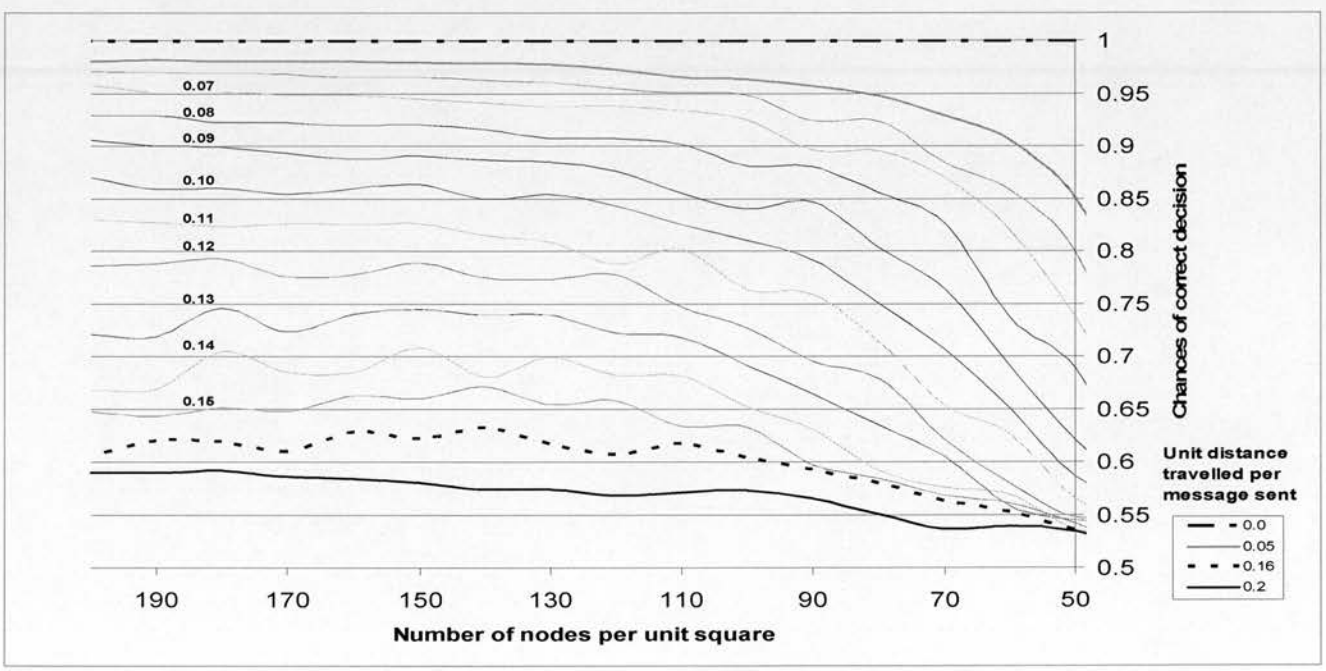


Figure 4. Simulation results showing the percentage of correct decisions made by Trajectory-Based Routing using the estimated logical locations

0-7803-8966-2/05/$20.00 (C) 2005 IEEE

Speckled Computing: Disruptive Technology for Networked Information Appliances

D.K Arvind and K.J Wong

Institute for Computing Systems Architecture

School of Informatics, University of Edinburgh

Mayfield Road, Edinburgh EH9 3JZ, SCOTLAND.

dka@inf.ed.ac.uk | k.j.wong@sms.ed.ac.uk

Abstract — *Speckled Computing is an emerging technology in which data will be sensed in minute (ultimately around one cubic millimetre) semiconductor grains called Specks. Information will be extracted, exchanged and processed in a collaborative fashion in a wireless network of thousands of Specks, called a Specknet. The impact of Speckled Computing on consumer electronics, especially future information appliances, will be disruptive. Objects and the surrounding environment when treated with Specks, or "Speckled", will be endowed with sensing, processing and wireless networking capabilities. This would effectively "smarten" everyday objects and surroundings post hoc, transforming them into networked information appliances. This paper introduces the concepts of Specks, Specknets and Speckled Computing, and outlines the challenges to be overcome to realise this technology. A prototype for Specks called ProSpeckz (Programmable Specks over Zigbee Radio) which is currently used as a rapid development platform for Speckled Computing is described. ProSpeckz is also intended as an enabler for integrating the technology of Speckled Computing into consumer electronics applications and some illustrative examples are described in this paper.*

Index Terms — Networked Information Appliances, Smart Home applications and protocols, Speckled Computing, Wireless computational networks

I. Introduction

A Speck is designed to integrate sensing, processing and wireless networking capabilities in a minute (ultimately one cubic millimetre) semiconductor grain. Specks are intended to be autonomous, each with a renewable energy source, and can be mobile if needed. Thousands of Specks, scattered or sprayed on any person or surface, will collaborate as programmable computational networks called Specknets. Computing with Specknets, or Speckled computing, will enable linkages between the material and digital worlds with a finer degree of spatial resolution than hitherto possible. Specknets are intended to be a generic technology for ubiquitous computing, where data is sensed, processed and information extracted in situ in a collaborative fashion.

Speckled Computing is the culmination of a greater trend in consumer electronics. As the once separate worlds of computing and wireless communications collide, a new class of information appliances will emerge. Where once these appliances stood proud; the PDA bulging in the pocket, or the mobile phone nestling in one's palm, the post-modern equivalent might not be explicit after all. Rather, data sensing and information processing capabilities will fragment and disappear into everyday objects and the living environment. At present there are sharp dislocations in information processing capability; the computer on a desk, the PDA/laptop, mobile phone, smart cards and smart appliances. In our vision of Speckled Computing, the sensing and processing of information will be highly diffused; the person, the artefacts and the surrounding space become at the same time computational resources and interfaces to those resources. Surfaces, walls, floors, ceilings, articles, and clothes, when sprayed with Specks (or "Speckled"), will be invested with a "computational aura" and sensitised post hoc as props for rich interactions with the computational resources. Given their minute sizes, Specks will have the ability to bring sensing and computation to places hitherto unreachable and wireless communication will enable Specknets to be the first (last) millimetre of the world-wide web. We predict that the impact of Speckled Computing on consumer electronics will be truly disruptive.

II. Specks and Specknets

Fig. 1 gives a system-level overview of the individual Speck and the Specknet. It is intended that Specks will be programmable, with the Specknet operating as a fine-grained distributed computation network, employing a lightweight and power-conscious communication protocol. The means of wireless communication will be a combination of optics (for example, laser and infrared) and radio, each with its strengths and weaknesses. Whereas optical communication is unidirectional, the radio communication is omni-directional; although optical communications consume several orders of magnitude less energy than radio, they are susceptible to breaks in communication due to occlusion or movement.

Specknet presents unique networking problems that will demand novel solutions. This is networking at its extreme and some of the key features of the Specknet will be decentralised control and adaptability. Adaptability is manifest in several ways, such as: dynamic routing to account for loss of connectivity either due to communication failures or the expiry of Specks; a power-sensitive Medium Access Control (MAC) layer which is aware of power depletions in a Speck and adapts the computation and communication accordingly; a combination of redundant processing and Speck bypassing within a group of Specks.

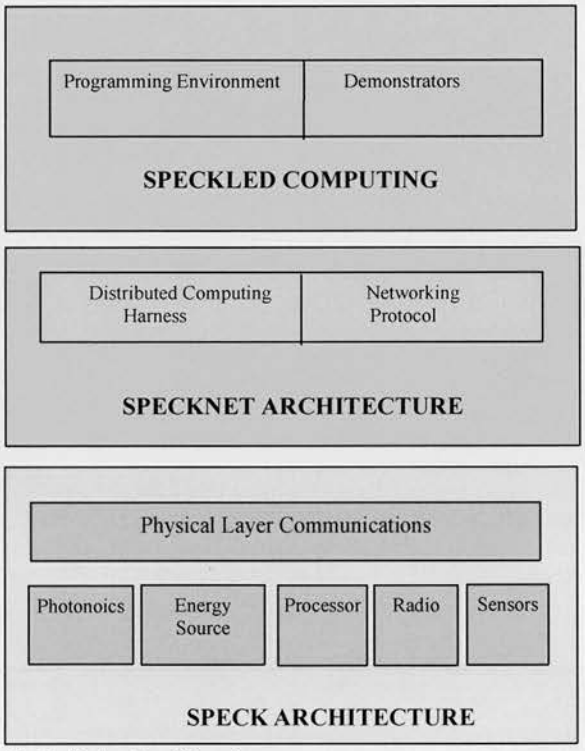


Fig. 1. System-level Overview

The Speck, though modest in terms of processing and storage resources, can be powerful as part of a collective system when harnessed as a Specknet. Specks would process their own sensor data and report only results and summaries externally. Limited individual processing power implies that Specks would need to organise the required processing collectively within a Specknet. Thus, a new model of distributed computation is being developed which will take into account some specific attributes of Specknets, such as unreliability of communication, a higher than normal failure rate of Specks due to harsh operating environment and very large volume manufacturing.

III. ProSpeckz – A Speck Prototype

ProSpeckz ("Programmable Specks over Zigbee Radio") is a prototype (Fig. 2) to enable both the rapid development of Specks and provide academia and industry with a platform to develop new applications for this emerging technology.

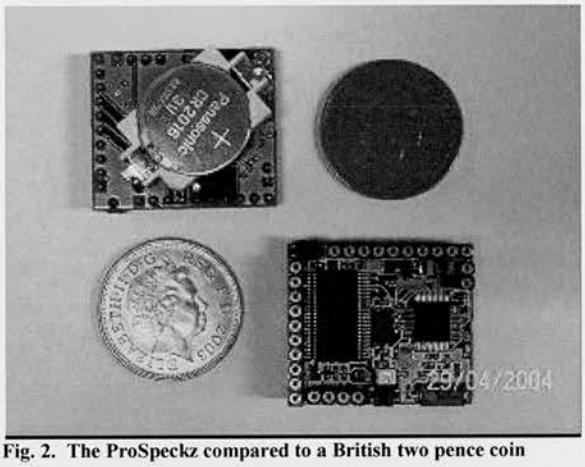


Fig. 2. The ProSpeckz compared to a British two pence coin

ProSpeckz provides the researcher with a means to determine the constraints and parameters in the real world which will inform the design of the semiconductor Specks and validate the simulation models that would be developed for Specks and Specknets in the future. It can also be incorporated, albeit with a form factor larger than the proposed Specks, into prototypes of future products for functional verification of the design.

ProSpeckz uniquely combines the following components into a versatile development platform:

a. An 802.15.4 compliant [1] radio chipset provides wireless communications up to data rates of 250kbps over 16 channels.
b. 2.4GHz matched antenna and filter circuitries allow software adjustable ranges from 30 centimetres to over 20 meters.
c. A Programmable System-on-Chip [2] (PSoC) enables ProSpeckz to provide software reconfigurable analogue circuitries to external interfaces and components. The PSoC is also the processing core of the ProSpeckz providing an 8-bit micro-controller with 16Kbytes of FLASH and 256bytes of RAM.

The systems overview of the ProSpeckz, which is constantly in development to reflect the actual semiconductor Specks, is shown in Fig. 3. On the hardware layers, the 802.15.4 radio is used to provide for physical wireless communications while the analogue re-configurability of the PSoC allows for the easy integration of sensors and actuators on the ProSpeckz. On the firmware layers, a power aware Medium Access Control (MAC) layer is used to provide efficient use of the radio channel via duty-cycling. It also manages the channel allocation to provide contention free access whenever possible.

A novel lightweight Specknet network protocol handles the routing between Specks using a unicast or multicast approach. This provides the higher layers the capability to wirelessly transmit data across the Specknet easily.

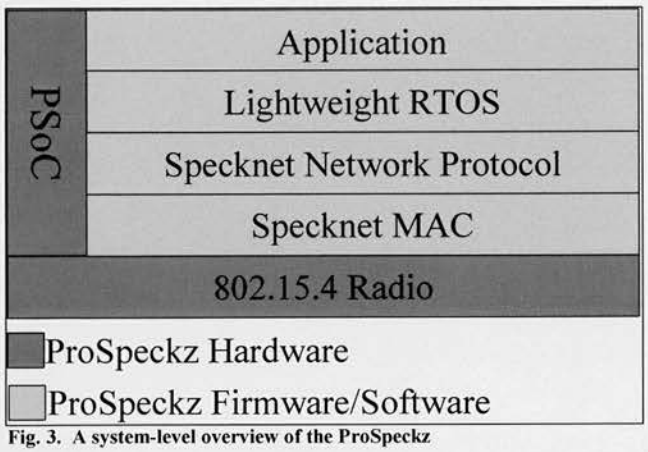


Fig. 3. A system-level overview of the ProSpeckz

A lightweight real-time operating system provides device drivers for the sensors and actuators connected to the board as well as allowing the scheduling of tasks, events and commands. A virtual machine could be implemented alongside the operating system to ensure interoperability for application developers as well as allowing for the easy development of simulation tools for the ProSpeckz. The virtual machine is intended to provide isolation between the hardware layers and the application program such that programs can run with least intervention between different versions of ProSpeckz.

Finally, the power consumption for the ProSpeckz in different modes is displayed in Table 1.

ProSpeckz Operational Modes	Consumption
Radio and PSoC in Sleep State	1mW
PSoC Running, Radio in Sleep State	17mW
PSoC, Radio Oscillator and State Machine Running	19mW
Data Transmission at:	
0 dBm	80mW
-5 dBm	68mW
-10 dBm	57mW
-15 dBm	53mW
-25 dBm	48mW
Data Reception	88mW

Table 1. Power consumption of the ProSpeckz in different modes

IV. Deployment of ProSpeckz in Consumer Electronics

The re-configurability of the ProSpeckz enables it to be interfaced easily to existing consumer electronics appliances such as televisions, hi-fi sets, and video recorders. The analogue circuitry on the ProSpeckz can be easily changed under software control to allow one hardware implementation to be used across multiple application domains. For example, using the PSoC Design Developer from Cypress Microsystems [2], the applications designer could easily reconfigure the 16 interface pins provided by the ProSpeckz to provide the following components:

a. Analogue Components
Programmable gain amplifiers, programmable low-pass or band-pass filters, switched capacitor blocks, 9-bit digital-to-analogue converters, among others.

b. Digital Components
Pulse width modulators, 8 to 32 bits counters/timers, hardware cyclic redundancy check (CRC) units, 8/16/32 bits pseudo random generators, to name a few.

c. Communications Components
Universal asynchronous receiver/transmitter (UART), serial peripheral interfaces (SPI), I^2C, infrared controls (IrDA), for instance.

ProSpeckz is an ideal platform to investigate the impact of Speckled Computing in the "Smart Home" [3] and the "Smart Space" applications [4]. By connecting sensors such as heat, pressure, motion and light detectors onto the ProSpeckz, a network of them can monitor the environment in a house and react under program control to any changes. The ability to reprogram ProSpeckz "over-the-air" enables upgrades to the ProSpeckz operating system and application programs without the need to physically connect. It also allows the network to be reprogrammed instantaneously and effortlessly thus providing the ability to upgrade pervasively when new sensors and actuators are added to the network.

Finally, ProSpeckz is an ideal platform to explore the integration of Speckled Computing technology into toys. ProSpeckz, when connected to an appropriate actuator such as a motor, can be used as a mobile platform for distributed game algorithms in robot soccer [5] and seek-and-destroy games. ProSpeckz can also be placed into toys on a smart playing mat to provide an interactive narrative of story, rhymes, songs and factual information when particular toys are in close proximity or the focus of attention. These linkages between the physical and digital worlds will provide a rich source of interaction during the child's educational development.

V. Demonstrators of ProSpeckz in Action

To further demonstrate the wide applicability of the ProSpeckz, several simple applications had been built and will be discussed briefly in this section.

a. A typical sensor network application

ProSpeckz can easily be used to demonstrate possible sensor network applications. By attaching a temperature sensor on each ProSpeckz, the ProSpeckz can be reprogrammed to function as fire detectors as shown in Fig. 4. Multiple ProSpeckz are then placed around a building and they will communicate with each other wirelessly forming a distributed wireless fire alarm network. Unlike conventional wired fire alarm systems, the proposed fire system is extremely intelligent. Instead of sounding an alarm in some far away centralized fire panel when a fire is detected, the ProSpeckz will lead the people in the building away from the fire by

forming a trail of running lights using the LED on each ProSpeckz as a guide. A simple distributed program onboard each ProSpeckz enables them to coordinate in a distributed matter in order to display the trail. Furthermore, unlike conventional wired system where the fire would potentially burn away the wires, ProSpeckz communicate wirelessly and the system would still be functional even if some of the ProSpeckz are destroyed by the fire.

Fig. 4. Demonstrating a distributed intelligent wireless fire alarm system

b. A test bed for developing distributed algorithms for Speckled Computing

As discussed earlier, one of the main usages of the ProSpeckz is to assist the development of algorithms for Speckled Computing. An example of such algorithms is a distributed algorithm for logical location estimation. This algorithm allows each node in a wireless network to estimate its own logical position based on its two-hop neighbour information. A logical position is defined as a coordinate in a two or three dimensional plane that is relative to the logical positions of the other nodes. Thus, a set of logical locations would give the layout of the nodes without the physical effects of rotation and translation. Logical location information is extremely important in most sensor network applications as the location information about where the data is sensed is deemed almost, if not, as valuable as the sensed data itself.

The algorithm has been designed and simulated on a Java software platform and is seen to be functional. However, there is a need to fully test out the algorithm in the physical world. This is where the ProSpeckz plays an important part. Each ProSpeckz is interfaced with a liquid crystal display (LCD) module (as shown in Fig. 5) which allows real-time readout of essential data on each node like the estimated logical location and the number of neighbours each node can communicate directly with. By moving the ProSpeckz about we can now determine, via the readouts on the LCD, the feasibility of the algorithm in real-life physical implementation and derive ways of improving it so as to achieve more accurate and stable estimations quickly.

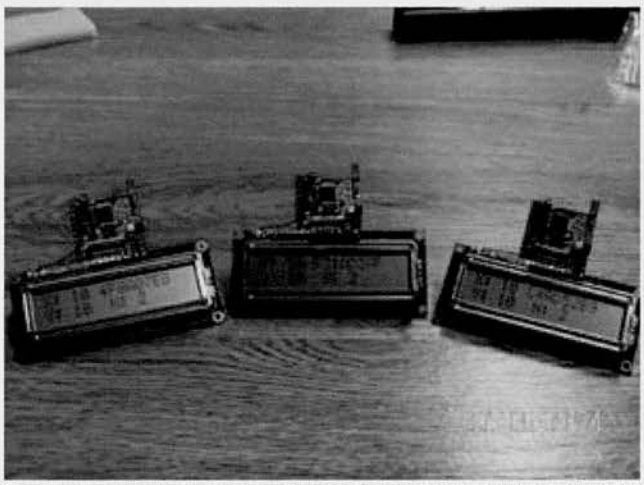


Fig. 5. Demonstrating ProSpeckz to test, develop and experiment with distributed Speckled computing algorithms

c. A platform to discover and develop novel consumer applications

The ability to easily interface to sensors and actuators commonly used in consumer electronic appliances allows the ProSpeckz to be utilized effectively in the search for a novel consumer application. For example, in Fig. 6, a speaker and an infrared range detector is connected to the ProSpeckz. Based on the measured distance between the ProSpeckz and an object (for example, a hand), a different note is played on the speaker. One could imagine the use of this as a "keyboard-less" musical keyboard where a song can be played by the movement of the hand or body. By using several of the ProSpeckz that are placed in strategic locations, it is also possible to emulate other musical instruments, for example, a string-less harp (a harp with virtual strings).

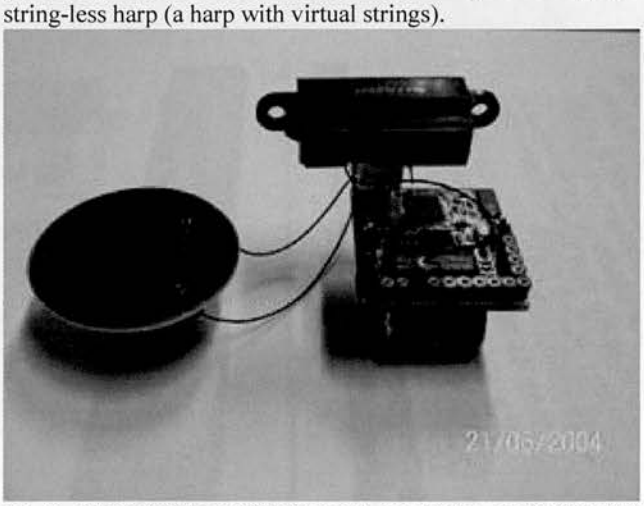


Fig. 6. Demonstrating ProSpeckz being used in the search for novel consumer applications, for example, a "keyboard-less" keyboard

d. A generic technology for mobile toys and robotics

A possible area of exploration for Specks is the use of actuators such as motors to allow mobility of the Specks. This could be made possible on the minute Specks via microelectromechanical system (MEMS) actuator arrays as used in the design of a walking silicon micro-robot [6]. However, to investigate the algorithms and applications in the absence of physical Specks, a ProSpeckz can be used. As shown in Fig. 7, a ProSpeckz is interfaced with a miniature car. This allows the programmer to develop applications in which multiple of these cars have to cooperate in order to achieve a certain function. Examples for such applications include mobile network routers, robotics games and mobile sensor networks.

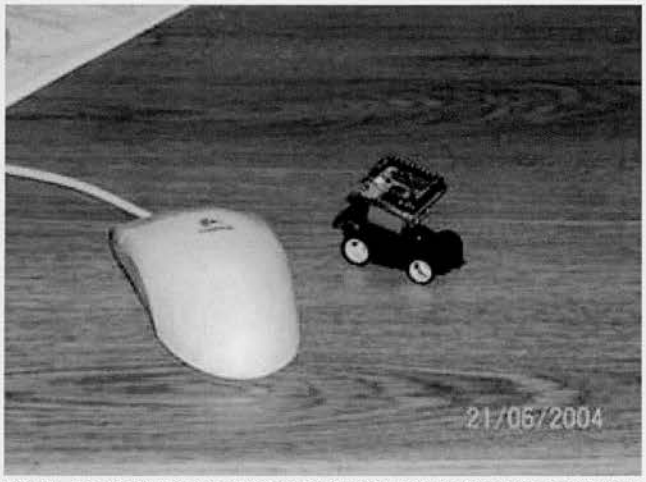


Fig. 7. Demonstrator in which a ProSpeckz is interfaced to a miniature toy car to enable it to move in a coordinated fashion via wireless communications with other ProSpeckz

VI. CONCLUSIONS

Speckled technology can be seen as a new age of computing and its introduction to consumer electronics will be extremely disruptive, both in the ways in which new applications are built as well as in the ways which true pervasive and ubiquitous implementations are made possible. A Speck prototype, ProSpeckz, is designed to serve a dual purpose. It is intended to inform the design of the semiconductor Specks by providing a flexible platform to experiment with the firmware, networking protocols, distributed computation harness and programming environment. Secondly, it will help prospective users of Specks to develop applications which could be ported to the semiconductor Specks with minimal effort. This will allow users to specify the requirements and determine the list of attributes for the minute Specks which are essential in the early design stages of this technology.

VII. ACKNOWLEDGEMENTS

The Research Consortium for Speckled Computing is funded by the Scottish Higher Education Funding Council as part of the Strategic Research Development Grant. The authors wish to acknowledge SHEFC for their support, and thank the members of the Consortium, especially the Speckled Computing Group at Edinburgh, for helpful discussions.

REFERENCES

[1] IEEE Standard for Information Technology, IEEE Std 802.15.4-2003, 2003

[2] Cypress MicroSystems, "PSoC Mixed Signal Array Final Data Sheet for CY8C27643, Silicon Revision A", November 2003

[3] http://www.smartthinking.ukideas.com/Smart%20Homes.html

[4] Robert J. Orr and Gregory D. Abowd, "The Smart Floor: A Mechanism for Natural User Identification and Tracking", GVU Technical Report GIT-GVU-00-02, January 2000

[5] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, It-suki Noda, and Eiichi Osawa, "Robocup: The robot world cup initiative", In Proceedings of the IJCAI-95 Workshop on Entertainment and AI/Life, pages 19–24, 1995

[6] Ebefors, T., Mattsson, J., Kalvesten, E., and Stemme, G., "A Walking Silicon Micro-Robot," In Proceedings of the 10th International Conference on Solid-State Sensors and Actuators, pages 1202-1205, 1999