Practical Packet Combining for use with Cooperative and Non-Cooperative ARQ Schemes in Wireless Sensor Networks

A Thesis Submitted in Fulfillment of the
Requirements for the Degree of
Doctor of Engineering
(Electronic Engineering)

By

Damien O'Rourke

BEng, MEng, MIEEE

School of Electronic Engineering
Faculty of Engineering and Design
Dublin City University

Research Supervisor
Dr. Conor Brennan
B.A.(mod), Ph.D, MIEEE, MIET

September 2009

Declaration

Acknowledgments

First and foremost I would like to thank my supervisor Dr. Conor Brennan. Conor has been a great source of ideas and inspiration throughout this thesis, not to mention a good friend. Without his guidance, encouragement and patience, this work would have invariably been a lot more difficult. His relentless support when things got tough is something I can't thank him enough for.

I would like also to thank my wife Claire and son Rían both of whom I dedicate this thesis to. I couldn't have done it without either of you. Your patience and love has been a great source of inspiration for me also. I love you both deeply.

I would like to thank all my family, especially my mam and dad who have always been supportive of me in all my endeavours. To all my brothers and sisters also, thank you all for your encouragement and support. I would also like to thank Fred and Therese Croxon both of whom have been a huge support to myself, Claire and Rían. Thank you both from the bottom of my heart for putting up with us in your home.

I would like to thank Szymon Fedor for his friendship and support. I enjoyed the many discussions we had on WSNs and hope we get to do it again someday.

I would like to thank my friends and colleagues at DCU. As always I'm grateful to Prof. Liam Barry for including me in all the optics lab events. Hopefully I'll be able to repay the gesture some day soon. Also, to all the people from the optics lab (both present and past members), Eoin Kennedy, David Molloy (or Holloy as he is known in China), and Donnacha Lowney. I would especially like to thank Paul Maguire who is always ready to go above and beyond whenever asked. I am also grateful to all the staff in the school of electronic engineering who have helped me along the way.

Finally, I would like to thank all my friends outside DCU for always being there.

Dedication

To Claire and Rían - thank you both for your amazing patience and love.

List of Publications

1. D. O'Rourke, and C. Brennan, "On Packet Combining for Wireless Sensor Networks," in *CIICT08*, Beijing, China, Sep. 26th-28th, 2008.

2. D. O'Rourke, and C. Brennan, "A Practical Implementation of an Improved Packet Combining Scheme for Wireless Sensor Networks," *Workshop on Cooperative Communications and Networking: Theory, Practice and Applications*, ICC 2008, May 19th-23rd Beijing, China.

3. D. O'Rourke, S. Fedor, C. Brennan, M. Collier, "Reception Region Characterisation using a 2.4GHz Direct Sequence Spread Spectrum Radio", in *Proc. of the 4th workshop on Embedded networked sensors EmNets '07*, Cork, Ireland, June 25-26, 2007.

4. S. Fedor, D. O'Rourke and M. Collier, "Cross-Layer Routing with Data Delivery Guarantee in Wireless Sensor Networks", in *Proc. of ACM Workshop on Real-World Wireless Sensor Networks*, REALWSN '06 in conjunction with ACM MobiSys , Uppsala, Sweden, June 19, 2006.

Abstract

Practical Packet Combining for use with Cooperative and non-Cooperative ARQ schemes in Wireless Sensor Networks.

Damien O'Rourke, *B.Eng.*, *M.Eng.*, *MIEEE*

Although it is envisaged that advances in technology will follow a "Moores Law" trend for many years to come, one of the aims of Wireless Sensor Networks (WSNs) is to reduce the size of the nodes as much as possible. The issue of limited resources on current devices may therefore not improve much with future designs as a result. There is a pressing need, therefore, for simple, efficient protocols and algorithms that can maximise the use of available resources in an energy efficient manner.

In this thesis an improved packet combining scheme useful on low power, resource-constrained sensor networks is developed. The algorithm is applicable in areas where currently only more complex combining approaches are used. These include cooperative communications and hybrid-ARQ schemes which have been shown to be of major benefit for wireless communications. Using the packet combining scheme developed in this thesis more than an 85% reduction in energy costs are possible over previous, similar approaches. Both simulated and practical experiments are developed in which the algorithm is shown to offer up to approximately 2.5 dB reduction in the required Signal-to-Noise ratio (SNR) for a particular Packet Error Rate (PER). This is a welcome result as complex schemes, such as maximal-ratio combining, are not implementable on many of the resource constrained devices under consideration.

A motivational side study on the transitional region is also carried out in this thesis. This region has been shown to be somewhat of a problem for WSNs. It is characterised by variable packet reception rate caused by a combination of fading and manufacturing variances in the radio receivers. Experiments are carried out to determine whether or not a spread-spectrum architecture has any effect on the size of this region, as has been suggested in previous work. It is shown that, for the particular setup tested, the transitional region still has significant extent even when employing a spread-spectrum architecture. This result further motivates the need for the packet combining scheme developed as it is precisely in zones such as the transitional region that packet combining will be of most benefit.

Contents

Declaration i

Acknowledgement ii

Dedication iii

List of Publications iv

Abstract v

1 Introduction **1**

1.1 Motivation and thesis aims 3

1.2 Thesis Contributions 4

1.3 Thesis Organisation 5

2 Background and Related Work **6**

2.1 Wireless Sensor Networks 6

2.2 The IEEE 802.15.4 Standard 11

2.2.1 The Physical Layer (PHY) 12

2.2.2 The Medium Access Sublayer (MAC) 14

2.3 Fading Effects in Wireless Communications 16

2.4 Error Control in WSNs 18

2.4.1 Forward Error Correction (FEC) 19

2.4.2 Automatic Repeat Request (ARQ) 20

2.4.3 Hybrid ARQ 21

2.4.4 Error Control Coding 23

2.4.4.1 Coding Gain 25

2.4.4.2 Cyclic Codes 26

2.4.4.3 Generating Cyclic Codes 28

2.4.5 Cyclic Redundancy Check 31

2.4.5.1 Error Detection Capabilities 33

2.4.5.2 Probability of undetected error 36

2.4.6 Error correcting capability of the CRC 39

2.5 Diversity 39

2.5.1 Repetition Codes 43

2.6 Multiple-Input, Multiple-Output (MIMO) Systems 44

2.7 Cooperative Communications 47

2.8 Summary 53

3 Reception Region Characterisation 55

3.1 Introduction 56

3.2 Related Work 58

3.3 Transitional Region Extent 59

3.3.1 Experimental 59

3.3.2 Probability of Bit Error 64

3.3.3 Bit Error Rate 66

3.4 Comparison of Results 69

3.5 Conclusions 72

4 Packet Merging 73

4.1 Related Work 74

4.2 Merging Procedure 75

4.3 Practical Setup 76

4.3.1 Search algorithms 79

4.3.2 Packet Selection 80

4.4 Experimental Results 85

4.5 Parity Equivalence 89

4.6 Incremental CRC 91

4.7 Packet Merging Issues 92

4.7.1 False Positives 92

4.7.2 Two Generator Polynomials 94

4.7.3 Choice of second polynomial 96

4.7.4 Hidden Errors and Uncorrectable packets 98

4.8 Summary and Conclusions 100

5 Performance Analysis **101**

5.1 Comparison of ARQ schemes 101

5.1.1 Analytical Model 103

5.1.2 Simulated Setup: One Retransmission 111

5.1.3 Simulated Setup: Two Retransmissions 121

5.2 Energy Measurements 126

5.2.1 Energy Cost of Communications 129

5.2.2 Energy Cost of Merging 133

5.3 Practical Results 139

5.4 Conclusions 145

6 Conclusions and Future Work **147**

Bibliography **153**

Appendix: The Cyclic Redundancy Check (CRC) **163**

List of Tables

2.1 The different available bands in the IEEE 802.15.4-2003 standard [1] 14

3.1 The values obtained for the channel parameters. The reference value d_0 is 1m for the outdoor environment and 2m for the indoor environment. 63

3.2 SNR values obtained using the three curves of figure 3.5. 71

3.3 Transitional Region Parameters obtained using the different graphs. Experimental results are those obtained in section 3.3.1. 72

4.1 The percentage of chosen packets, using either LQI or RSSI, containing the lesser number of errors. 84

4.2 Comparison of the percentage of required iterations for each setup. 88

4.3 Number of hidden errors (e_h) found. 100

5.1 Gains (in dB) for each method used when two transmission attempts are permitted. 120

5.2 Gains in dB for each method used when three transmission attempts are permitted. A packet size of 1080 bits is used. 124

5.3 Gains (in dB) for each method used when three transmission attempts are permitted. A packet size of 320 bits is used. 125

5.4 Transmission power output levels given in the data sheet [2] for a given register value. 129

5.5 Energy Consumption (μJ) for the different parameters measured. Transmission power is set to 0 dBm and payload to 28 bytes. 135

5.6 Comparison of energy consumption and gains for a retransmission, a standard packet merge and iPM. 136

5.7 Subset of transmission power output levels determined practically that were used in the PER experiments. The total number of packets sent for each transmission power is also shown. 141

1 A comparison of five different possible implementations of the CRC algorithm with the time results measured over a 512-byte message (taken from [3]). 173

List of Figures

2.1 Block diagram of a possible wireless sensor. The location finding system and mobiliser are optional in many cases. 7

2.2 The IEEE 802.15.4 channel structure: taken from [1]. 13

2.3 Comparison of probability of bit error for a BPSK system over an AWGN channel and a Rayleigh fading channel. 17

2.4 Plot of the probability of error for a system using a (7,4) Hamming code [4]. . 26

2.5 Plot of the coding gain obtained by using a (7,4) Hamming code for each probability of error. 27

2.6 Comparison of an uncoded and a repetition coded BPSK system over a AWGN with $n = 3$ and $n = 11$ [4] 43

2.7 Model of a cooperative system. 48

3.1 PRR outdoors with each mote separated by a distance of 1m. The nodes were placed on small cardboard boxes about 5cm in height. The transmit power level of each node acting as source was set to -15 dBm. 60

3.2 PRR indoors with each mote separated by a distance of 2m. The transmit power level of each node acting as source was set to -15 dBm. 61

3.3 RSSI outdoors with each mote separated by a distance of 1m. η is approximately 3.6 and σ is approximately 4.1. 61

3.4 RSSI indoors with each mote separated by a distance of 2m. η is approximately 2.6 and σ is approximately 5.6. 62

3.5 Plot of BER given in IEEE 802.15.4 standard [5] compared with the empirical measurements using the CC2420 transceiver. 69

3.6 Plot of BER given in IEEE 802.15.4 standard [5] compared with the empirical measurements using the CC2420 transceiver shifted by constant offsets. 70

4.1 Comparison of the search methods for sPM and iPM. 80

4.2 Plot showing practically obtained error vectors (e_S and e_R) and ambiguity vector (V_A) obtained from experimental setup 3. 81

4.3 Average difference in Hamming weights between the source error vector (e_S) and the relay error vector (e_R) compared with the Hamming weight of the ambiguity vector ($w(V_A)$). 82

4.4 Example of how the Hamming weight of the error vector changes with LQI (taken over a static link with a mean of about 67). 83

4.5 Example of how the Hamming weight of the error vector changes with RSSI (taken over a static link with a mean of about -92 dBm). 83

4.6 Empirical cumulative distribution of $w(V_A)$. 86

4.7 Empirical cumulative distribution of the difference in Hamming weights between m'_S and m'_R for each experiment. 87

4.8 Empirical cumulative distribution of $w(V_A)$ for one set of results. 98

4.9 For each Hamming weight, this graph shows the fraction of ambiguity vectors that contained hidden errors at a particular weight, out of the total at that weight. 99

5.1 Traditional ARQ. 102

5.2 Cooperative ARQ. 102

5.3 Comparison of the exact and approximate functions for the PER of a BPSK system with $L = 1080$ bits in a packet. 104

5.4 Plot of the PER curves for equations 5.3, 5.4, 5.7, 5.9 with $\bar{\gamma} = \sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} = 20$ dB. 106

5.5 SNR gain for both traditional and cooperative system with $\sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} = 20$ dB. 108

5.6 SNR gain for both traditional and cooperative system with $\sigma_{s,d} = \sigma_{r,d}$ and a perfect source-relay channel. 109

5.7 SNR gain for the cooperative system with $\sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} = \{5, 10, 15, 20, 25, 30, 40\}$ dB. 109

5.8 Simplified block diagram of the simulation setup using BPSK. 112

5.9 Comparison of the simulated versus the analytical results for a BPSK system over a slow Rayleigh fading channel with a packet size of 1080 bits. The source-relay channel was assumed to be perfect. 113

5.10 Packet error rate curves for a BPSK system with a packet size of 1080 bits and $w(V_A)_{max} = 10$. 114

5.11 Packet error rate curves for a BPSK system with a packet size of 1080 bits and $w(V_A)_{max} = 15$. 114

5.12 Gains achievable using Packet Merging (PM) over different values of $w(V_A)_{max}$ for a packet size of 1080 bits with a perfect source-relay channel. 115

5.13 The percentage of ambiguity vectors obtained for each SNR with a Hamming weight less than or equal to 15. In this case $\bar{\gamma} = \sigma_{s,d} = \sigma_{r,d}$. 116

5.14 Comparison of two BPSK systems transmitting over a slow Rayleigh fading channel with packet sizes of 320 bits and 1080 bits. The source-relay channel is assumed to be perfect. 118

5.15 Percentage of ambiguity vectors obtained using tradARQC with Hamming weights less than or equal to 15. Both 320-bit and 1080-bit cases are considered. . . . 119

5.16 Packet error rate curves for a BPSK system with a packet size of 320 bits and $w(V_A)_{max} = 15$. 119

5.17 Gains achievable using Packet Merging (PM) over different values of $w(V_A)_{max}$ for a packet size of 320 bits with a perfect source-relay channel. 120

5.18 Comparison of using a traditional and cooperative ARQ scheme with both two and three retransmission attempts, and one and two different relays. The packet size was set to 1080 bits in all cases. 122

5.19 Gains obtained using different combinations of packet merging and majority voting with both traditional and cooperative systems. 123

5.20 Consideration of the actual number of transmissions required when using merging. 124

5.21 a) The power delivered, dissipated or consumed by each component and b) the percentage error of approximating the power dissipated in the node by $V_{cc}i_R$ over different values of v_R. 128

5.22 The current consumed for a transmit event taken at the 8 different power levels specified in the CC2420 datasheet. Only the values of -25 dBm and 0 dBm are annotated. 130

5.23 Current consumed by two separate transmit events on the Tmote Sky. 130

5.24 Current consumption of the Tmote at different transmission power levels. . . . 131

5.25 Current consumed during the reception a packet with a payload of 0 (top graph) and 28 (bottom graph) bytes. 132

5.26 The energy consumption for each extra data byte in the *payload*. 133

5.27 Energy consumed using both a full search algorithm and iPM. 134

5.28 Power levels obtained experimentally for each register value on the CC2420. The values given in the data sheet are plotted for comparison. 140

5.29 Practical results. These results were taken with two nodes only, i.e. tradARQ(C), with $w(V_A)_{max} = 10$ and a source packet size of 216 bits. 142

5.30 Practical results. These results were taken using two nodes only, i.e. tradARQ(C), with $w(V_A)_{max} = 15$ and a source packet size of 216 bits. 143

5.31 Practical results. These results were taken using three nodes, i.e. coopARQ(C), with $w(V_A)_{max} = 10$ and a source packet size of 320 bits. 144

5.32 Practical results. These results were taken using three nodes, i.e. coopARQ(C), with $w(V_A)_{max} = 15$ and a source packet size of 320 bits. 144

1 IEEE 802.15.4 Frame Format. 164

2 Table based CRC code. The table consists of a table of 256 entries and trades-off program space for speed. 165

3 Shift register based CRC code used in TinyOS 2.x (see [6]) 167

4 Calculating the FCS of a message with more than one byte assuming the MSB is the *first* byte of the message. 169

5 Checking the FCS of a message with more than one byte assuming the MSB is the *first* byte of the message. 169

6 Hardware shift register described in IEEE 802.15.4 standard and used by the CC2420 chip. 169

7 Software implementation of the shift register used in the CC2420. The hexadecimal value 0x8408 is 0x1021 viewed backwards. 171

Chapter 1

Introduction

Wireless Sensor Networks (WSNs) offer a new paradigm in communications. The advances, in recent years, of VLSI techniques have made possible very small computational/communication devices which, of themselves, have limited capabilities but on aggregate achieve a great deal. This newly emerging technology has a myriad of possible applications but, as with all new technologies, is laden with design issues. These issues have opened up a whole new playground for researchers. One particular example that has come to the fore is that of energy consumption. WSNs will generally be placed in environments without access to any fixed power sources and where they will be unable to be maintained by humans. They are therefore required to last on the order of years on other forms of power such as batteries, solar cells, vibrations, and thermoelectric effects [7, 8]. This raises issues not only for hardware design but also in the approach taken when designing the protocols and algorithms for these devices. Most of the design strategies for networks such as wireless LANs do not work effectively for WSNs due to the inherently different design constraints.

To help address the issue of energy consumption in WSNs this thesis develops a very simple but practical technique referred to as packet combining[1]. Packet combining can be used when the same data is received from multiple sources or from the same source over different time slots. The former case generally occurs in a promising technique

[1]For more complex systems packet combining is an effective method for achieving higher throughput [9] however in this thesis the emphasis is placed on reduced energy consumption; the two are not mutually exclusive however.

referred to as cooperative communications, which has been shown to be of benefit for WSNs [10] and is a viable alternative to Multiple-Input, Multiple-Output (MIMO) techniques which are unrealisable on current WSNs due to their size, cost and hardware limitations. The latter case can be simply referred to as a traditional Automatic Repeat Request (ARQ) system with packet combining (although a number of variations are possible as will be seen throughout this thesis) [9].

Some of the combining techniques such as maximal-ratio combining [11] or the use of Space Time Block Codes (STBCs) [12] are not viable options for commercially available architectures such as the popular Tmote Sky [13], MicaZ [14], and other similar devices, due to their simplified hardware. In addition, Channel State Information (CSI) at both the transmitter and receiver is generally not available and only partial information can be obtained at the receiver in most cases (in the form of LQI and/or RSSI[2]). It is with these low power, resource-constrained architectures that the current work is based.

Although significant advances are likely to be made in the area of chipset design in the coming years the size of future WSNs are envisaged to become increasingly small in size (in some cases to the order of a piece of dust [15]). The emphasis, then, still remains on less complex protocols despite these advances in CPU speed, available program memory etc. Towards this end a simple packet combining protocol referred to as *improved Packet Merging (iPM)* is developed in this thesis. This protocol is based on previous work in [16], [17] and most recently [18]. Using available metrics such as RSSI/LQI combined with other novel techniques the energy consumption of this protocol is significantly reduced.

As motivation for the need for packet combining, and another contribution of this thesis, an investigation of the *transitional region* was carried out. Several works have re-

[2]Link Quality Indicator (LQI) and Received Signal Strength (RSSI).

ported the existence of a transitional region in which the Packet Reception Rate (PRR) is quite erratic [19–21]. The extent of this region is important as upper layer protocols may disregard its effects (such as link asymmetry) leading to, for example, inefficient routing topologies[3]. The underlying causes of the transitional (or so-called "grey") region have been determined in previous work using both analytical and empirical techniques (see, for example, [22]). Conclusions drawn from this work suggests that the transitional region is caused in part by multi-path fading but also possibly by hardware irregularities (e.g. [23]). It was suggested in [22] that the use of a spread-spectrum architecture may help to reduce the extent of this region. This thesis shows, through practical experimentation, that this region can still have significant extent in certain circumstances, even with the use of a spread-spectrum architecture. The results obtained motivate the need for other techniques (such as packet combining) to reduce its extent.

1.1 Motivation and thesis aims

The main focus of this thesis is to develop an improved packet combining scheme that can be easily implemented on low-power, resource-constrained wireless sensor nodes. The purpose of such a scheme is to reduce the required energy consumption of a wireless node while maintaining the same performance levels or, alternatively, to obtain improved performance at the same energy consumption. As this thesis is concerned with WSNs, and the inherent energy problems they pose, it is the former case that is of interest. The combining scheme developed is implemented in both a cooperative and non-cooperative setup to determine the gains possible; this is done both practically and through simulations. Although the concept of packet combining is not new, very few published papers show practical results of its benefits instead only documenting results based on unrealistic channel models. Also, little or no practical

[3]Efficient routes in one direction may not be efficient in the opposite direction.

results have been published[4] that show the diversity gains obtainable on these resource-constrained devices.

This thesis also documents the results of an experimental study on the transitional region. The experiment was carried out in order to ascertain the need for a packet combining scheme as it was suggested in [22] that a spread-spectrum architecture may reduce the extent of this region. The results show that even with the use a spread-spectrum radio the transitional region still has significant extent in certain setups.

1.2 Thesis Contributions

This work makes the following contributions to the field:

- An improved *practical* implementation of a packet combining scheme based on the Cyclic Redundancy Check has been developed in this thesis.
- This is shown to have a positive effect on the energy consumption of a wireless node as it lowers the number of transmissions required to achieve a given packet error rate
- It is shown that the improved packet combining scheme developed offers as much as an 85% reduction in the number of iterations required to find the correct packet over current implementations. This leads to further significant energy savings.
- The use of a second generator polynomial has been shown to be of benefit in this scheme offering improved performance through the reduction of false-positives.
- A thorough energy analysis has been carried out for a particular type of wireless node.

[4]To the best of this author's knowledge.

- The benefits of packet merging in both cooperative and non-cooperative scenarios have been shown practically and in simulations, for different packet sizes.
- A motivating experimental study is carried out which shows that, under certain circumstances, the transitional region still has significant extent even with the use of a spread-spectrum architecture.

1.3 Thesis Organisation

This chapter has provided a brief introduction to the content and focus of this work. Chapter 2 discusses, in detail, background material and relevant work in areas such as ARQ techniques, cooperative communications and packet combining techniques. Following this, chapter 3 will discuss the motivation for using packet combining by showing that the use of spread spectrum does not always reduce the extent of the transitional region as previously suspected. Chapter 4 develops an improved packet combining scheme that can be implemented on currently available low-power, resource-constrained WSNs. Chapter 5 studies the effects of the combining scheme developed in chapter 4. This is done using a combination of theoretical, simulated and practical results. Chapter 6 looks at future work and, finally, conclusions are drawn on the effectiveness of cooperative communications, ARQ techniques and packet combining in a practical WSN scenario using these resource-constrained devices.

Chapter 2

Background and Related Work

2.1 Wireless Sensor Networks

A Wireless Sensor Network consists of a large number of sensor nodes that can be arranged closely together in or around a specific area of interest. They are required to operate unattended without any fixed power supply for long periods of time (on the order of years) and must therefore consume extremely low amounts of energy in order to enhance longevity. They must be autonomous and adaptive to the environment and production costs need to be kept extremely low as the nodes will ultimately be required to be dispensable, operate in high volumetric densities and be scalable: up to the order of, perhaps, millions of nodes [24]. The dispensability of the nodes may raise environmental issues: if they become widespread then some sort of biodegradability may need to be considered. As the aim is to reduce the size of the nodes to the order of cm^3 this may not be an issue however.

The myriad of possible applications for WSNs include: Environmental monitoring [25], health monitoring [26, 27], tracking and mapping in real environments [28], agriculture [29–32], building automation and control [33], structural-monitoring [34], habitat-monitoring [35, 36], and micro-climate monitoring [37] — to mention but a few. Using WSNs can imply reduced installation costs (no cables are required), rapid reconfiguration of data acquisition procedures [38] and safe deployment in hostile environments [39]. One interesting possible use of WSNs might be to monitor the surfaces of other planets within the solar system [40].

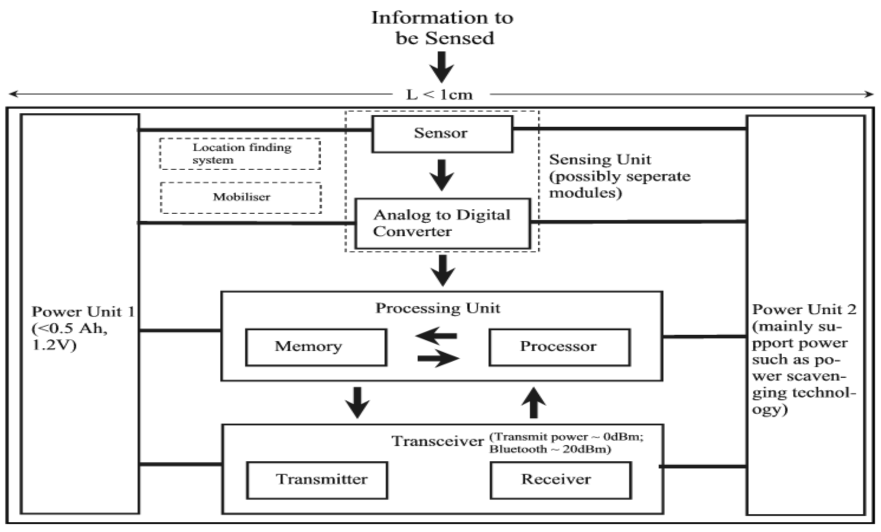

Information to be Sensed
L < 1cm
Sensor
Location finding system
Mobiliser
Sensing Unit (possibly seperate modules)
Analog to Digital Converter
Power Unit 1 (<0.5 Ah, 1.2V)
Processing Unit
Memory
Processor
Power Unit 2 (mainly support power such as power scavenging technology)
Transceiver (Transmit power ~ 0dBm; Bluetooth ~ 20dBm)
Transmitter
Receiver

Figure 2.1: Block diagram of a possible wireless sensor. The location finding system and mobiliser are optional in many cases.

Figure 2.1 shows a block diagram of a possible wireless sensor node. Although it is suggested that each dimension of the node is less than 1cm, currently available nodes are normally much larger. Typically the nodes have a unit for sensing (humidity, temperature, pressure etc.), a unit for processing, a transceiver unit and a power unit; other units are possible such as localisation, mobility, and energy scavenging. In this last case energy is extracted from the environment through phenomena such as vibrations, solar power and heat [7, 41, 42]. For example, the use of solar panels is an option in certain areas: solar insolation ranges from 2.12kWh/m^2/day in certain parts of the world (e.g. Alaska) to about 5kWh/m^2/day (e.g. mainland Australia) and conversion efficiencies of modern solar panels are about 10% [43]. A typical solar cell generates from 80 to 400 kilojoules of solar energy per month which in many cases is more than adequate for WSN applications [32].

As energy scavenging technologies improve, the timescale of years may become a more realistic goal for the lifetime of a node[1]. In the absence of efficient scavenging techniques, however, it is necessary that the node sleeps most of the time. It is shown in [45] that in order to obtain a node lifetime of roughly 2-7 years, an average power consumption of between 10 to 100μW is required when running on a single 1.5V alkaline battery with a leakage current of approximately 30μA and a capacity of 2.6 Ah (Amp-hours). However, it is also noted that the power consumption of currently available radio transceivers is on the order of tens of mWs when in active mode; this renders it impossible to keep the transceiver on all the time. In fact, *idle listening*, where the radio is listening to an idle channel, has been identified as one of the major causes of energy wastage in WSNs [46]. It is noted in [47] that the energy cost of idle-listening for a number of different transceivers is similar to that of receiving and approximately 50% of that of transmitting[2].

The main way of lowering the energy cost of idle-listening is by lowering the *duty-cycle* of the node. An analysis carried out in [48] shows that in order to allow a node to run on one AAA battery for two years the node must remain asleep on average 99.67% of the time it is operational. This low duty-cycle is currently the most popular method for reducing the energy consumption of sensor nodes. Other methods, however, might include (for example): reduction of the amount of data transmitted (by in-network processing, for example) or reduction of frame overhead (by using variable length headers, for example) [48].

It is important also to consider the energy efficiency of the algorithms and protocols used with wireless sensor nodes. The requirements described above (such as scalability and adaptivity) essentially mean that WSNs pose major new challenges for protocol and algorithm design. As the nodes must last on the order of years, unattended, with-

[1]Interestingly, the wireless roadway monitoring system described in [44] can purportedly last up to 10 years on a small battery.

[2]Of course at lower transmit powers the energy consumed receiving a packet may be greater than that for a transmission [2].

out any fixed power source, the current layered protocol design models (such as the Open Systems Interconnect (OSI) model) may not be the best approach [49]. One reason for this is that the topology of sensor networks is generally *ad-hoc* in nature. As noted in [49], an ad-hoc network is a collection of wireless nodes that self-configure to form a network without the aid of any established infrastructure. Communications therefore take place in a peer to peer fashion rather than with a centralised node. The positions of the nodes within the area of interest will generally not be known *a priori*, requiring reconfigurability of the network in the event of node failure or the addition of new nodes. Ad-hoc networks generally require distributed (rather than centralised) networking and control functions with multi-hop routing, and they cannot provide any guarantee in terms of data-rate and delay [49]. Although the layered model offers designers a logical, modular approach to protocol design it may not work well in these ad-hoc scenarios as many of the design issues are strongly inter-connected across layers.

Infrastructure based networks such as cellular networks are generally *single-hop* as each node is within one hop of a wired base-station. This is generally not the case for ad-hoc networks and each node must rely on other nodes to help transport the data back to the sink. In such a *multi-hop* scenario each node not only routes incoming data but is also a source of its own data. Multi-hop routing is used as it decreases the required overall transmission power of a node due to the non-linearly rising nature of the path loss (with respect to distance) and as it also decreases interference in multiple-access networks [50]. However, careful use of multi-hop configurations needs to be considered. In [51] it is shown that multi-hopping only saves energy when increased transmission power, required to overcome the path attenuation, dominates the energy consumption of the hardware which, according to [51], occurs less frequently than is typically believed. This latter point makes sense when one considers a dense deployment of nodes where packet overhead and *overhearing*[3] become significant causing the energy consumption of the hardware to dominate. If the energy of a node is depleted the routing topology must change. This makes the approaches to routing more complex than for infrastructure based networks. An example of a protocol used by WSNs that addresses some of the issues with energy drainage in routing scenarios is the Low-Energy Adaptive Clustering Hierarchy (LEACH) protocol where the nodes are setup into clusters with each *cluster-head* being chosen in a random fashion to help distribute the workload amongst all the nodes [52].

As mentioned earlier traditional protocol design for communication systems has used a layered model with each layer designed and operated independently of the others. This has allowed for a modular approach to network protocol design and has served wired-line networks well leading to, for example, the robust, scalable protocols now used in the Internet. However, the constraints of WSNs, and ad-hoc networks in general, have necessitated changes in this approach and now a *cross layer* approach to designing these protocols has been shown to be needed [49, 53]. As mentioned earlier, many of the design issues are strongly inter-connected across layers and by actively exploiting these inter-dependencies it is possible to achieve performance gains [54].

In [47] it is noted that at the MAC layer energy is conserved by reducing idle listening, protocol overheads and collisions. At the topology control level adaptive duty cycling helps to reduce this energy consumption. At the network layer energy efficiency is achieved by the use of energy aware routing protocols and data-aggregation. Cross-layer design seeks to optimise each of these efforts in a synergistic way and therefore inherently involves violating the layered architecture which defines such things as, limited, static communication interfaces between adjacent layers and no communication between nonadjacent layers.

[3]Overhearing occurs when a node receives a packet destined for some other node.

An example of a cross-layer approach is the use of protocol adaptivity. In this case the protocol attempts to adapt to changes in the current network conditions to compensate for any degradation. These changes may occur on different time-scales depending on the nature of the change. Noting this, and noting that some of these changes may occur at a very fast rate, in [49] it is shown that adaptivity is best applied at local layers first and then, failing any improvement in the condition, at upper layers: with information provided from other lower layers. If adaptation wasn't attempted locally at first then it may be too late, in some cases, to adapt to the change at upper layers. However, as also noted in the cited work it is this integrated approach to adaptive networking that forms the biggest challenge in adaptive protocol design.

Cross-layer design has been considered in other areas also. For example, the wireless medium has offered a new set of modalities that were not available in wired communications. The broadcast capability of the wireless medium allows for nodes to cooperate with one another [55, 56]. This technique, referred to as *cooperative communications* (cf. section 2.7), is inherently a network problem and issues of protocol layering and cross-layer architectures naturally arise that can involve the physical, medium-access control, link, and even network layers [57] in order to obtain maximum gains [54].

2.2 The IEEE 802.15.4 Standard

The large application space for WSNs requires the need for multiple platforms leading to the availability of a number of different designs and architectures. Standardisation is, however, necessary in order to maintain interoperability between different designs. Currently a large number of designers of WSNs are gravitating towards the IEEE 802.15.4 standard [5]. The original IEEE 802.15.4 standard was ratified in 2003 [58] and an updated version published in 2006 [5]. This standard is aimed at Low-Rate Wireless Personal Area Networks (LR-WPANs) which are defined in the standard to be "simple, low-cost communication networks that allow wireless connectivity in applications with limited power and relaxed throughput requirements". The standard deals with two layers of the Open Systems Interconnect (OSI) model: the Physical Layer (PHY) and the Medium Access Control Layer (MAC). Its features include low data rate, low power consumption, low cost, self organisation and flexible topologies [59].

To cater for upper layers not dealt with by IEEE 802.15.4 the de facto ZigBee standard is being developed by the ZigBee Alliance [60]. The ZigBee standard builds on top of the IEEE 802.15.4 MAC and PHY layers and encompasses a complete network stack for WSNs focused on sensor and control networking [61]. The original version of the this standard was ratified in 2004. The ZigBee Alliance is a consortium of leading semiconductor manufacturers, technology providers, Original Equipment Manufacturers (OEMs) and end-users worldwide. The number of members at the time of writing is about 220 [60]. An open source version of the IEEE 802.15.4/ZigBee stack has recently been made available [62].

2.2.1 The Physical Layer (PHY)

The physical layer, or PHY, defines, for example, the frequency band to be used, the data rate, and the modulation scheme to be used. The 2003 version of the standard defines two PHY layers[4] across three bands: the 868/915 MHz Industrial Scientific Medical (ISM) band and the 2.4 GHz ISM band. The 868 MHz band is available in Europe, the 915 MHz in the North America, and the 2.4 GHz band worldwide. The 2006 version of the standard adds different optional modulation and spreading techniques for the 868/915MHz band to allow for a trade-off between complexity and data-rate (the 2.4GHz PHY was not modified in the 2006 version of the standard). For simplicity however, this section discusses mainly the PHY layers specified in the

[4]The 868/915MHz band is classified as one PHY layer when defined with the particular details of the modulation scheme, data-rate etc.

original standard which are still valid in the 2006 version.

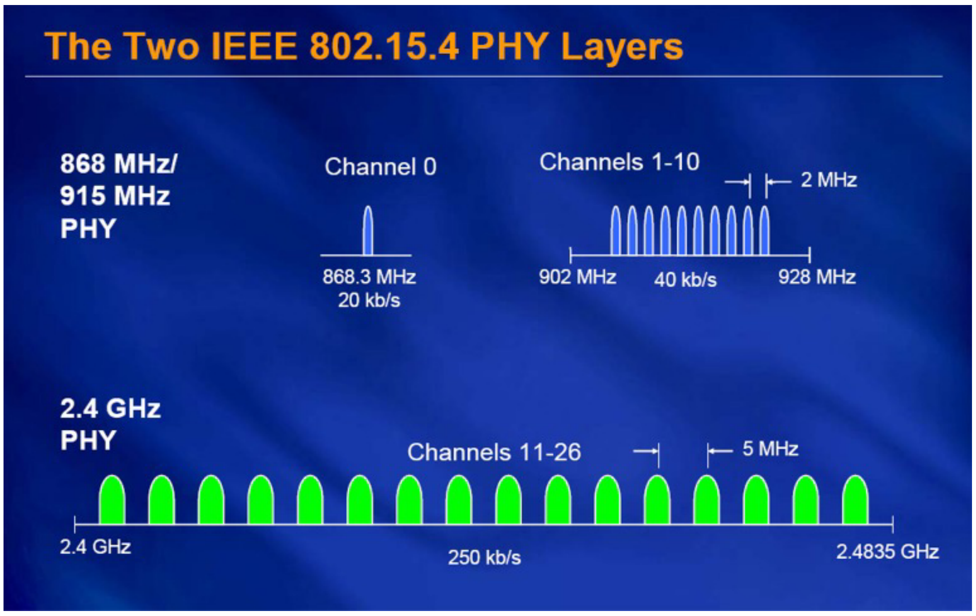


Figure 2.2: The IEEE 802.15.4 channel structure: taken from [1].

Figure 2.2 shows the channel structure for the IEEE 802.15.4 standard. The three separate bands offer three different data rates over a total of 27 possible channels. The 868 MHz band has only 1 channel available (numbered channel 0) with a bandwidth of 600kHz (868.0 - 868.6 MHz) and data rate of 20kbps. The 915 MHz band has 10 channel allocations (numbered 1-10) over a bandwidth of 26MHz (902 MHz to 928 MHz with a 2 MHz spacing between each channel) and a data rate of 40kbps. The 2.4GHz band has 16 channels (numbered 11-26) over a bandwidth of 83.5MHz (2,400 MHz up to 2,483.5 MHz with a 5MHz spacing between each channel). It should be noted that the addition of the optional PHYs in the 2006 revision of the standard increases the number of channels to 3 for the 868MHz band and 30 for the 915MHz band [5]). Table 2.1 (modified from [1]) shows the original (yet still valid) data rates and modulation schemes used in each band.

The 2.4GHz band is used in the experimental section of chapter 4. It specifies the

Channel Number	Center Frequency	Data Rate	Channel Spacing	Modulation Scheme
k = 0	868.3 MHz	20 kbps	-	BPSK
k = 1,2, . . . , 10	906 + 2(k-1) MHz	40 kbps	2 MHz	BPSK
k = 11,12, . . . , 26	2405 + 5(k-11) MHz	250 kbps	5 MHz	O-QPSK

Table 2.1: The different available bands in the IEEE 802.15.4-2003 standard [1]

following modulation scheme: the incoming source bits are grouped into *information* symbols of 4 bits each ($k = 4$). Each symbol is then mapped to one of 16 pseudo-orthogonal chip sequences of length $n = 32$. The 32 chips are then divided into $b = 2$-chip *channel* symbols which are sent to the modulator and mapped to a waveform using O-QPSK modulation with half-sine pulse shaping. This modulation scheme is equivalent to Minimum Shift Keying (MSK) which itself is a form of Coherent Phase Frequency Shift Keying (CPFSK) where the deviation index is equal to 1/2 [63, 64]. Given the values of n and b it can be seen that there are $n/b = 16$ channel symbols for each information symbol. More will be said about this modulation scheme in later chapters.

2.2.2 The Medium Access Sublayer (MAC)

In terms of the OSI model [11], the MAC layer is a sublayer of the Data Link Layer which deals with the problem of achieving reliable communications in a point to point link. The MAC layer itself controls how each node accesses the shared wireless medium. It determines, amongst other things, when the radio should be in sleep-mode, when it should be in active-mode, and the duty-cycle percentage. Ideally, it should maximise the time the radio is in sleep-mode (to save energy) while, at the same time, preserving the highest throughput and minimum latency.

Generally, MAC protocols can either be distributed or centralised. In a distributed protocol, each node of the network makes an independent decision as to whether or not the channel is free to transmit whereas in a centralised protocol these decisions are made by a central controller. Each type has its merits and which one is chosen will depend on the particular application. A distributed MAC protocol offers the following benefits over a centralised approach [65]:

- There is no central point of failure
- There is no central synchronisation (which can be difficult to achieve in practice)
- Security is improved due to redundancy as the overall network cannot be attacked at any one point
- May scale to arbitrarily large networks

The disadvantage of a distributed MAC is that the increased possibility of collisions reduces the ability to guarantee Quality of Service (QoS).

The IEEE 802.15.4 MAC uses a contention based medium access scheme; however, there is an optional method where time slots can be allocated, by a coordinator, to devices with time critical data. In the contention based scheme a technique known as Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) is used[5]. The channel is sensed before transmission to determine whether or not it is free. If the channel is clear then the node can send its information. However, if the channel is not clear then it must wait a random amount of time before checking the channel again. The period of time the node waits is known as the *backoff factor*. A backoff counter is used to determine when the node checks again: when the backoff counter is zero, the channel is deemed clear and the node will attempt to transmit again. If the channel is found to be in use the node resets the counter and waits some more. This continues

[5]Carrier sense is more generally known as Clear Channel Assessment (CCA).

until a clear channel becomes available at which point the node will transmit the signal. CSMA/CA should not be confused with Carrier Sense Multiple Access with Collision *Detection* (CSMA/CD) which is a protocol that decides what to do *after* a collision has occurred.

Although not implemented in the IEEE 802.15.4 standard it is possible to manage collisions in a CSMA/CA system using Request-to-Send (RTS) and Clear-to-Send (CTS) packets. These packets contain the size of the data to be transmitted so other nodes know not to transmit for the duration of this packet. More information on this can be found in [65].

2.3 Fading Effects in Wireless Communications

The wireless channel presents many challenges for communication systems that wired channels do not. The most significant of these is the phenomena of fading whereby the signal arriving at the receiver is a summation of multiple signals traversing different paths. The purpose of this section is to give a brief overview of the effects fading has on a communications link. This will help motivate the need for the combining techniques developed later in the thesis. The details of the how and why of fading will not be dealt with here. However, a good overview can be found in [11, 66], for example.

In an Average White Gaussian Noise (AWGN) channel the probability of error for a Binary Phase Shift Keying (BPSK) scheme is given by

$$Pb = Q(\sqrt{2\gamma_b}) \tag{2.1}$$

where γ_b is the ratio of bit energy E_b to noise power spectral density N_o, and $Q(.)$ is the complementary error function [67]. This probability follows an exponential fall-off. In a flat Rayleigh fading channel using coherent demodulation however it is given by [11]

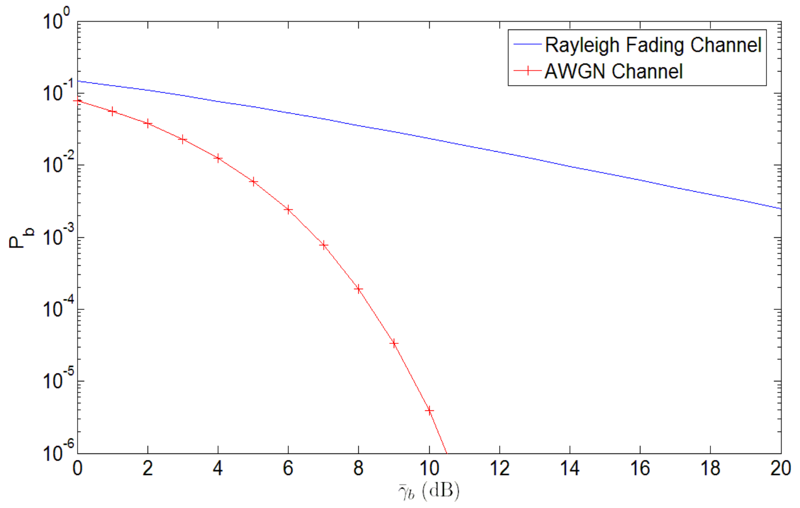


Figure 2.3: Comparison of probability of bit error for a BPSK system over an AWGN channel and a Rayleigh fading channel.

$$Pb = \frac{1}{2}\left[1 - \sqrt{\frac{\bar{\gamma}_b}{1+\bar{\gamma}_b}}\right] \approx \frac{1}{4\bar{\gamma}_b} \tag{2.2}$$

where the approximation holds at high values of $\bar{\gamma}_b$. Similar equations hold for coherent binary Frequency Shift Keying (FSK),

$$Pb = \frac{1}{2}\left[1 - \sqrt{\frac{\bar{\gamma}_b}{2+\bar{\gamma}_b}}\right] \approx \frac{1}{4\bar{\gamma}_b}, \tag{2.3}$$

and for Differential Phase Shift Keying (DPSK),

$$Pb = \frac{1}{2(1+\bar{\gamma}_b)} \approx \frac{1}{2\bar{\gamma}_b}, \tag{2.4}$$

where, again, the approximations holds for large values of values of $\bar{\gamma}_b$. For the DPSK case the assumption is also made that the channel phase is relatively constant over a symbol time [11].

A plot of equations 2.1 and 2.2 can be seen in figure 2.3. For each of the modulation schemes in the fading channel the value of P_b decreases only *inversely* with $\bar{\gamma}_b$. In contrast however, P_b decays *exponentially* in an AWGN. Without taking any extra measures, very large amounts of power would be required in a fading channel in order to obtain reliable communications. Increasing the transmission power leads to decreased battery (and hence node) lifetimes as well as introducing increased levels of interference in a multiple-access scenario; it is therefore not a solution to this problem.

In order to keep the error-rate at a tolerable level then, without exacting too much power, a number of techniques can be used. Three of the most common are error correction/detection codes, repetition coding[6], and/or diversity. Each of these techniques will be looked at in turn in the following sections.

2.4 Error Control in WSNs

Reliable communications over noisy, time-varying channels, inherently requires some form of error control. Even for static channels error control techniques are required due to the effects of noise and interference. There are two main error control techniques: *Forward Error Correction* (FEC) and *Automatic-Repeat-Request* (ARQ) schemes. In the former, an error-correction code is used; in the latter, an error-detection code along with retransmissions, if necessary, is used. This section gives a brief overview of each scheme. Section 2.4.4 then goes into some detail on error control coding of relevance to the work in this thesis. In particular, the packet combining scheme presented in chapter 4 makes extensive use of the Cyclic Redundancy Check (CRC).

A commonly used figure of merit used when discussing different error control schemes (and communication systems in general) is the systems *throughput*. This is defined as the ratio of the average number of packets successfully transmitted in any given time

[6]Although repetition codes are a class of error control codes they follow a much simpler premise than codes such as Hamming codes and are normally used as a form of time diversity. They will therefore be dealt with once diversity has been discussed.

interval divided by the number of attempted transmissions in that interval [11]. For ARQ systems that use an (n, k) error correction code (cf. section 2.4.4) it can be defined as the average number of encoded data packets accepted by the receiver in the time it takes the transmitter to send a single k-bit data packet [9]. While the first definition is concerned with packets the second one is concerned with bits. The type of system under influence will determine which is more appropriate.

2.4.1 Forward Error Correction (FEC)

In an FEC system an error correcting code is used to try to correct any errors in the received data. FECs are required if the channel is simplex as, in that case, there is no way of obtaining the feedback information required to determine the status of the reception. The throughput of an FEC system is constant as there are no retransmissions and is equal to the rate (cf. section 2.4.4) of the code [68]. Two main types of error control codes used in an FEC system are *block codes* and *convolutional codes* [4,9,69]. In [70] it is shown that a type of block code known as a BCH code[7] is found to be 15% more energy-efficient than the best performing convolution codes proposed to date for WSNs. Block codes are also easier to implement on resource-constrained devices and the encoding energy required for them can generally be considered negligible [69]. Section 2.4.4 reviews block codes in more detail.

The ability to reduce the transmit power of a node through the use of error correction techniques means that there will be less errors due to interference in a multi-access network. This not only increases the capacity of the network but also reduces its energy consumption by reducing the number of required retransmissions and levels of overhearing. The particular error control scheme used should be carefully chosen for each situation. Error control codes designed for AWGN channels, for example, do not

[7]Named after its inventors: Bose, Chadhuri and Hocquenghem.

work well for channels that exhibit fading due the bursty nature of the fading channel. Instead, AWGN codes are generally used with interleaving for fading channels [11]. For WSNs in particular, complex error control schemes designed for high-power systems will not work well on low-power, resource-constrained sensor nodes and energy-efficient, low-complexity error control is required.

2.4.2 Automatic Repeat Request (ARQ)

While the number of possible FEC schemes is vast there are only three main types of simple ARQ schemes: stop-and-wait, go-back-N and selective-repeat. For the stop-and-wait scheme the transmitter expects an acknowledgement for each packet sent. If received correctly then a positive acknowledgement (ACK) is sent, otherwise a negative acknowledgement is sent (NACK)[8]. In the former case the transmitter simply transmits the next packet in the queue whereas in the latter case the packet that was incorrectly received is retransmitted. Although, theoretically, the number of retransmissions can be infinite in practice only a finite number is used and the resulting scheme is referred to as *truncated* ARQ.

In the go-back-N scheme packets are continuously transmitted and there is no idle time spent waiting for an acknowledgement. This scheme therefore requires a full-duplex link in order to continue sending packets *and* receive a NACK, if one occurs. Upon receiving a NACK the transmitter goes back to the packet that was received in error and resends it along with all subsequent packets. For the go-back-N scheme a buffer is required at the transmitter to store the transmitted packets in the event of a retransmission request. There are a number of drawbacks to this scheme that make it unsuitable for low-power, resource-constrained sensor nodes. The need for a full-duplex link and a buffer at the transmitter may not be an option in many cases. Even more importantly,

[8]Variations on this theme are, of course, possible such as using a timeout instead of a NACK.

the energy wasted in resending all subsequent packets after the original retransmission, even though many of them may have been correct, would be detrimental to the lifetime of a sensor node.

In the selective-repeat system the packets are also transmitted continuously. However, upon receiving a NACK only the incorrectly received packet is resent. Although this is the most efficient of the three basic ARQ schemes it is also the most complex to implement. For example, buffers must be provided at *both* the transmitter and the receiver; on WSNs this may place a burden on the limited resources. The full duplex requirement may not be available in this case either.

The simplicity of the stop-and-wait scheme makes it very attractive for use on the low-complexity devices used in WSNs. However, it has inherent drawbacks due to the idle time the transmitter spends waiting for an acknowledgement; this generally leads to reduced throughput compared to the other two schemes. If the round-trip delay is negligible however, both stop-and-wait and go-back-N behave identically to selective-repeat in terms of throughput [68]. In WSNs the low transmission powers prohibit long distance communication and therefore limit the round-trip delay leading to similar throughput for all three schemes. For these reasons a stop-and-wait system is implemented in the practical experiments developed in later chapters.

2.4.3 Hybrid ARQ

Using FEC or ARQ alone may not be sufficient for many systems for the following reasons: a) the throughput of an ARQ systems drops quickly as the channel error rate becomes large, b) an ordinary FEC system may have to provide erroneous packets to the user as no retransmissions are possible, and c) as a result of this last point, FEC systems may be quite complex requiring very powerful error-correcting codes in order to achieve good reliability. To overcome these deficiencies, it is possible to combine

FEC and ARQ schemes to provide further improvements. The resulting system is termed a *Hybrid-ARQ* system.

For Hybrid ARQ systems the data is encoded with an error correction code followed by an error detection code. Upon detection of an error an attempt is made to correct it using the error correction code; failing this a retransmission is requested. Provided that an appropriate combination of FEC and ARQ schemes is chosen, Hybrid ARQ schemes offer the potential for better performance than either FEC or ARQ used independently. In a hybrid-ARQ system, the function of the FEC subsystem is to correct more frequently occurring errors. Upon receiving a less frequent uncorrectable error a retransmission is requested by the ARQ system. The incorrect packet is therefore not passed up the stack, improving system reliability. There are two main types of hybrid ARQ schemes. The first is referred to as a *type-I* scheme and the second a *type-II scheme*. The type-I scheme uses an error control code designed for both error detection and error correction. If the corrupt packet cannot be corrected a retransmission of the *same* codeword is requested. This process continues until the packet is either correctly received or correctly decoded. Type-I systems work well for systems in which a constant level of noise and interference is expected. If this is not the case then sending redundant bits on a benign channel is wasteful of energy as they are not needed. For channels producing more errors than the code was designed for, the redundant bits (required for error correction) may also be wasteful as they will be unable to correct the errors. These drawbacks motivate the use of a type-II system.

In a type-II system, the first transmission attempt contains only those parity bits required for error detection. Upon detecting a corrupt packet, instead of retransmitting the same bits again (information and parity), a new block of error correction parity bits, based on the original packet, is sent. If the error correction is not successful then a retransmission is again attempted. This time either more parity bits or the original codeword may be sent. A number of variations on this are possible however[9] (see [68] for more details). The type-II scheme offers the advantage that it is adaptable to changing channel conditions as the extra redundant bits are only sent if required; this is referred to as *incremental redundancy*.

To assist in an understanding of the concepts of FEC, ARQ and Hybrid-ARQ the next section looks at error control codes in more detail. An emphasis is placed upon the *Cyclic Redundancy Check (CRC)* — a type of error detecting code. The reason for focusing on this particular scheme is that, as mentioned, the packet merging algorithm developed in chapter 4 makes extensive use of it. A detailed look at its strengths and weaknesses is therefore provided. Before this however, a brief introduction to linear block codes is given as the CRC is derived from them.

2.4.4 Error Control Coding

Although it is theoretically possible to use random codes in communication systems, the problem in applying them is the complexity of the encoding and decoding operations [69]. To allow for practical implementation it is necessary to define codes with a good deal of structure; block codes are one example that offer such a structure. Consider a message block represented by the binary k-tuple $\mathbf{m} = (m_{k-1}, \ldots, m_1, m_0)$. When applying a block code, the encoder takes as input the message $\mathbf{m}$ and outputs the n-tuple *codeword* $\mathbf{c} = (c_{n-1}, \ldots, c_1, c_0)$ where $n \geq k$. The number of possible k-bit messages is 2^k whereas the number of possible n-bit outputs is 2^n. As the mapping is one-to-one only 2^k outputs are used. The set of these 2^k outputs (of length n) is referred to as an (n, k) block code[10] $\mathbf{C}$. The number of information bits is therefore k, the number of codeword symbols n, and the number of redundant bits $n - k$. The

[9]Such as sending a high-rate error *correction* code to begin with followed by extra parity on subsequent transmissions enabling a reduction in the code rate.

[10]Sometimes written (n, k, d_{min}) where d_{min} is known as the minimum distance of the code.

redundant bits are included to combat the adverse effects of noise.

Linear block codes are a subclass of the class of all block codes. The property of linearity implies that the sum of any two codewords within the set is also a codeword (i.e. the set is closed under addition) and that the all-zero word is also a codeword. Linear block codes are *memoryless* in the sense that the current n-symbol output codeword depends only on the current k-bit input message. These type of codes can be implemented using combinational logic circuits.

The ratio $R_c = k/n$ is known as the *code rate*. To understand the significance of the code rate assume that the rate in which the codeword symbols are transmitted is R_s symbols per second. The rate at which useful bits are transmitted (i.e. the information rate) is then $R_b = R_c R_s = (k/n)R_s$ bits per second. As $k \leq n$ this implies that $R_c \leq 1$. The smaller the value of k for a particular n, the lower the code rate due to the increased redundancy. Generally speaking, a lower rate code has better error correction capabilities than a higher rate code but the trade-off is a reduction in the spectral efficiency of the system.

Two common measures used throughout this thesis are the *Hamming weight* and the *Hamming distance*. The Hamming weight of a codeword **c** is equal to the number of nonzero elements in the codeword and is denoted by $w(\mathbf{c})$. The Hamming distance between two codewords $\mathbf{c}_1$ and $\mathbf{c}_2$ is the number of places in which the codewords differ and is denoted by $d(\mathbf{c}_1, \mathbf{c}_2)$. The error correction/detection capabilities of a linear block code are determined by its *minimum distance* which is the smallest Hamming distance between any two codewords of the code[11]:

$$d_{min} = \min_{\mathbf{c}_i, \mathbf{c}_j \in \mathbf{C}, \mathbf{c}_i \neq \mathbf{c}_j} d(\mathbf{c}_i, \mathbf{c}_j) \tag{2.5}$$

[11]Equivalently, it is the smallest Hamming weight in the code (not including the all-zeros codeword).

where $\mathbf{c}_i$ and $\mathbf{c}_j$ are codewords i and j of the code $\mathbf{C}$ and d is the Hamming distance. The random error *correction* capability, t, of a linear block code is determined by d_{min}:

$$t = \lfloor (d_{min} - 1)/2 \rfloor \tag{2.6}$$

whereas its error *detection* capability is given by $d_{min} - 1$. Both of these capabilities are guaranteed although it is possible to detect/correct other error types. The **Singleton bound** states [4]:

$$d_{min} \leq n - k + 1 \tag{2.7}$$

which ultimately implies that the error detection/correction capabilities of the code are bounded by the number of redundant bits $(n - k)$ used. Examples of the use of linear block codes can be found in any good textbook on error control or communication systems (see, for example [4, 9, 11, 67, 69]).

2.4.4.1 Coding Gain

Given a required probability of error, the *coding gain* of a system in an AWGN channel is the difference between the required signal-to-noise ratio for an uncoded system and a coded system. As an example of coding gains for typical block codes, the (7,4) Hamming code offers a coding gain of about 0.45 dB at an SNR of 10dB [4]. For the slightly more powerful (15,11) Hamming code this increases to about 1.5 dB [9]. Coding gains of 6 to 9 dB are available in standard error correcting systems. This allows for a reduction in the transmitted power by up to a factor of 8 [9]. There is of course a trade-off in bandwidth and decoder complexity to consider so it is not always easy to compare different techniques for reducing the level of SNR required.

The coding gain of a system can be better understood by considering figure 2.4. This

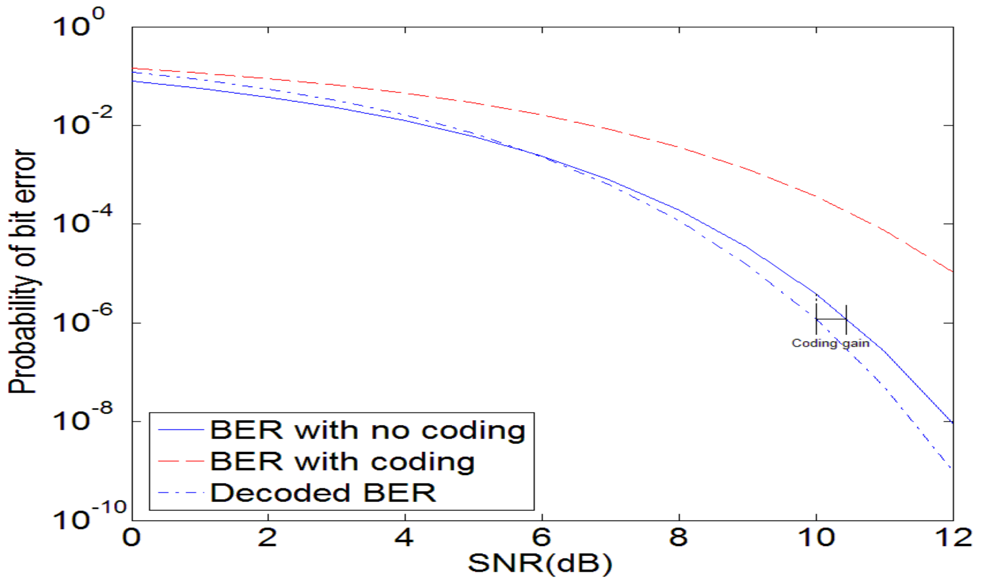


Figure 2.4: Plot of the probability of error for a system using a (7,4) Hamming code [4].

graph shows the probability of error for an uncoded system, a coded system before decoding, and a coded system after decoding. At an SNR of 10dB for the coded system it can be seen that the coding gain of the system using a (7,4) Hamming code is approximately 0.45dB.

It is possible for a code designed for high-SNR channels to have negative coding gain at low SNRs due to the extra redundancy not compensating for the extra cost in spreading the energy over the redundant bits. A plot of the coding gain for different bit error probabilities of the (7,4) Hamming code is shown in figure 2.5. At a probability of error of about 2.8×10^{-3} the coding gain goes to zero. Beyond this the (7,4) hamming code actually gives a negative coding gain.

2.4.4.2 Cyclic Codes

The structure of linear block codes allows for a relatively simple decoding process; further simplifications can be made if the codes are also cyclic. Linear cyclic codes form

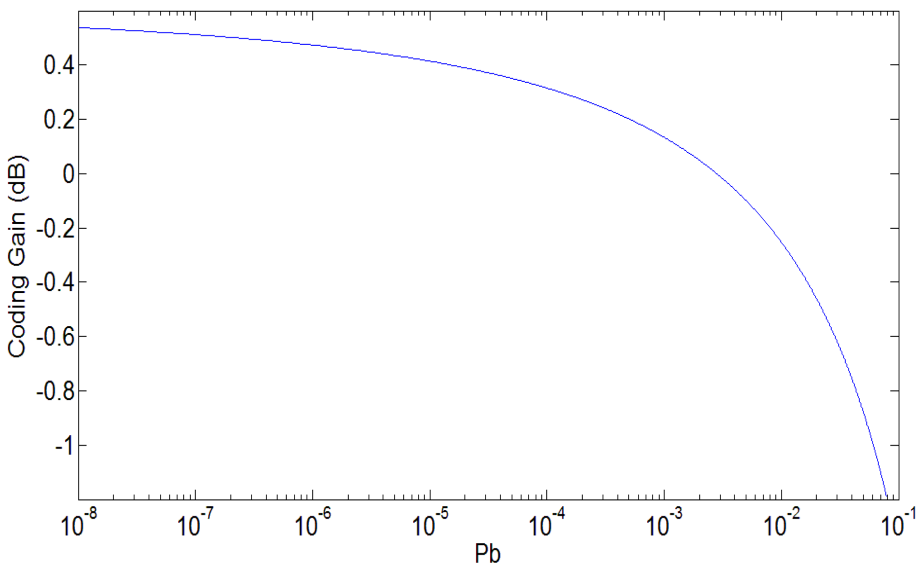


Figure 2.5: Plot of the coding gain obtained by using a (7,4) Hamming code for each probability of error.

a subset of linear block codes with the added property that cyclically shifting a codeword any number of symbol positions, left or right, produces another valid codeword. This implies that cyclic codes possess full cyclic symmetry which makes them easy to implement using simple shift registers and logic circuits. Nonlinear cyclic codes also exist however they have a less useful structure and are used much less in practice than the linear variety [9].

To develop the concepts of cyclic codes it is preferable to use polynomials as an easy method for keeping track of bit positions. The terms *codeword* and *codeword polynomial* can then be used interchangeably due to the one to one mapping between the two. Consider the n-bit codeword $\mathbf{c} = (c_{n-1}, c_{n-2}, \ldots, c_2, c_1, c_0)$. This can be represented in polynomial form as:

$$c(x) = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \ldots + c_2x^2 + c_1x + c_0 \tag{2.8}$$

A cyclic shift to the left by one therefore gives:

$$c^1(x) = c_{n-2}x^{n-1} + c_{n-3}x^{n-2} + \ldots + c_1x^2 + c_0x + c_{n-1} \tag{2.9}$$

which turns out to be equal to the remainder when $xc(x)$ is divided by $x^n + 1$ [4, 69]. The polynomial $c^1(x)$ therefore can be written more compactly as

$$c^1(x) = R_{x^n+1}[xc(x)] \tag{2.10}$$

where $R_{y(x)}[z(x)]$ represents the remainder when $z(x)$ is divided by $y(x)$. In general, a cyclic shift of **c** by m places can be written as:

$$c^m(x) = R_{x^n+1}[x^m c(x)] \tag{2.11}$$

provided that $x^m c(x)$ has degree less than n.

2.4.4.3 Generating Cyclic Codes

A polynomial is called a generator polynomial, $g(x)$, if it can generate every codeword of a particular code. For each (n, k) cyclic code there exists a *unique* generator polynomial from which all the codeword polynomials can be generated; it has the form:

$$g(x) = x^{n-k} + g_{n-k-1}x^{n-k-1} + \ldots + g_2x^2 + g_1x + 1. \tag{2.12}$$

As can be seen, the degree[12] of $g(x)$ is equal to the number of redundant bits in each codeword (i.e. $n-k$) and $g(x)$ is the smallest degree codeword polynomial of an (n, k) code (excluding the all zero codeword polynomial). The coefficients of the highest and

[12]The degree of a polynomial is the value of the exponent of the largest power of x in that polynomial.

lowest order bits of the generator polynomials must be equal to 1.

A binary polynomial of degree $n-1$ or less is a codeword polynomial if and only if it is a multiple of $g(x)$. Every codeword polynomial can therefore be expressed in the following form:

$$\begin{aligned} c(x) &= m(x)g(x) \\ &= (m_{k-1}x^{k-1} + \ldots + m_1 x + m_0)g(x) \end{aligned} \tag{2.13}$$

where the coefficients of $m(x)$ can be considered to be the information bits to be transmitted. The error control properties of the code are determined by the structure of the generator polynomial, as shall be seen.

The codeword generated by (2.13) is generally less useful in practice than one that is in *systematic form*. This is where the most significant k bits of the codeword are simply the information bits to be transmitted, and the least significant $n-k$ bits the parity bits. A systematic code can be generated by using the following procedure:

1. Multiply the message polynomial $m(x)$ by x^{n-k} (this left shifts the message $n-k$ positions),
2. Divide $x^{n-k}m(x)$ by the generator polynomial $g(x)$, obtaining the remainder $r(x)$ of degree $n-k-1$ or less,
3. Add $r(x)$ to $x^{n-k}m(x)$, obtaining the codeword polynomial $c(x)$ of degree $n-1$ or less.

To see that this works note that $x^{n-k}m(x) = q(x)g(x) + r(x)$ for some quotient $q(x)$ and remainder $r(x)$. Addition is carried out modulo-2 on the coefficients and hence addition is the same as subtraction implying: $x^{n-k}m(x) + r(x) = q(x)g(x)$, which is a multiple of the generator polynomial and hence a codeword. The resultant codeword can be written as:

$$\mathbf{c} = (m_{k-1}, \ldots, m_1, m_0, r_{n-k-1}, \ldots, r_1, r_0) \tag{2.14}$$

where it can be seen that the codeword consists of the original message bits $(m_{k-1}, \ldots, m_1, m_0)$ followed by the parity check bits $(r_{n-k-1}, \ldots, r_1, r_0)$. The remainder polynomial $r(x)$ can also be written as

$$r(x) = R_{g(x)}[x^{n-k}m(x)], \tag{2.15}$$

using the notation introduced earlier. The generation of cyclic codes using (2.13) or the systematic encoding procedure produce the same set of codewords but the mapping from message space to codeword space is different for the two procedures. The encoding and decoding procedures used in the latter case are essentially identical, accounting for its popularity over the former.

To decode any block code a *syndrome* is computed. This is an $n - k$ tuple consisting of all zeros if no error has occurred or the error is identical to a non-zero codeword (i.e. an undetected error has occurred). If the received codeword contains detectable errors then the syndrome does not consist of all zeros. If the received codeword contains correctable errors then it is possible to identify the specific error pattern that occurred. For cyclic codes the codeword must be a multiple of the generator polynomial $g(x)$. In this case computation of the syndrome is reduced in complexity as the syndrome will simply be the remainder when the received data is divided by the generator polynomial. The resulting syndrome can then be used for error detection or correction, the difference being whether an attempt is made to map the syndrome to an error pattern that produces it (of which there are 2^k possible solutions). More details on the use of syndromes can be found in, for example [4,69].

To determine whether a polynomial of degree $n - k$ generates an (n, k) code it is sufficient to determine whether it is a factor of $x^n + 1$. This also means that any factor of $x^n + 1$ with degree $n - k$ generates an (n, k) cyclic code. If n is large there may be many factors of $x^n + 1$ of degree $n - k$. However, they don't all produce good cyclic codes and the problem of selecting a good generator polynomial is, in general, a difficult one.

2.4.5 Cyclic Redundancy Check

Error detection is often used after error correction coding takes place in order to detect the presence of any residual errors. In other cases it is used as part of an ARQ scheme in which a retransmission takes place if an error is detected. Error detection schemes are very popular due to their simple implementation and high reliability for a surprisingly small message overhead or redundancy — a simple example being the use of a single parity bit to determine whether an error has occurred. Beyond the single bit parity approach, the Cyclic Redundancy Check (CRC) offers a more powerful alternative with excellent error-detection performance.

The phrase "cyclic redundancy check" is really only a common name used for what is, in fact, a shortened cyclic code (also known as a polynomial code[13]). The shortening process involves taking the subset of codewords, in an (n, k) systematic, cyclic block code with the j high-order *information* bits equal to zero. These high-order bits (i.e. $m_{k-1}, \ldots, m_{k-j+1}, m_{k-j}$) are then deleted from each codeword in the subset and the code that remains is an $(n - j, k - j)$ shortened systematic code (see [4,69] for more details).

A shortened cyclic code has *at least* the same minimum distance as the cyclic code

[13] Due to the fact that the same generator polynomial is used for both the original and shortened codes.

from which it is derived but is, in general, *not* cyclic. As with all codes, the error detection capability depends on the minimum distance of the code. As the CRC is used with variable sized messages however the minimum distance will be determined by the size of the message. Shortening reduces the rate of a code and the resulting code has error detection and correction capabilities *at least* as good as the original code [9, 69].

Since shortened cyclic codes are derived from cyclic codes the codewords are generated using a generator polynomial. For error detection the generator polynomial is generally chosen so that it has the following form:

$$g(x) = (x+1)p(x) \tag{2.16}$$

where $p(x)$ is known as a *primitive* polynomial of degree $n-k-1$ (see [69] for more details). An example of two polynomials of this form are the two standard polynomials used by the ITU-T[14] (formerly CCITT) and ANSI:

$$\begin{aligned}
\text{CRC-ITU:} \quad g(x) &= x^{16} + x^{12} + x^5 + 1 \\
&= (x+1)(x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x + 1) \\
\text{CRC-ANSI:} \quad g(x) &= x^{16} + x^{15} + x^2 + 1 \\
&= (x+1)(x^{15} + x + 1).
\end{aligned} \tag{2.17}$$

Although the above generator polynomials are chosen to create shortened cyclic codes, they of course can be used to obtain the original cyclic code from which the shortened version is derived. The number of high-order deleted information bits, j, from the

[14]Herein referred to simply as ITU.

original cyclic code will depend upon the size of the message ($m(x)$) being sent. The blocklength $n-j$ of a shortened cyclic code is therefore variable in size. The *natural* blocklength n_t of a shortened cyclic code is that of the cyclic code from which the shortened code is derived and is determined by the degree of the primitive polynomial used in the construction of the generator polynomial (i.e, $2^{n-k-1}-1$). For the original cyclic code the natural block length is simply equal to n, i.e. $n = n_t$. For the shortened cyclic code, n will still be used to represent the blocklength of the shortened code with the understanding that it is now variable in size and doesn't necessarily equal n_t. It should be noted also that for a given natural blocklength all codes have the same performance [69]. In practice the code is almost always shortened (in order to accommodate the various possible message sizes) and the codes do not therefore have the same performance [71].

Implementing the CRC is relatively straightforward in both hardware and software. As this thesis emphasises the practical aspects of packet combining, the appendix provides an example and explanation of a CRC implementation in the C programming language — with specific reference to the CC2420 chipset. The methods discussed are easily extended to other languages/platforms. More information regarding the implementations of the CRC can be found in [4].

2.4.5.1 Error Detection Capabilities

There are a few rules which generator polynomials should obey. To help understand these rules note that any received codeword polynomial $c'(x)$ is made up of the original codeword polynomial $c(x)$ and an error polynomial $e(x)$:

$$c'(x) = c(x) + e(x). \tag{2.18}$$

If $c'(x)$ is divisible by $g(x)$ it is accepted as a valid codeword even though it may have errors in it. As $c(x)$ is divisible by $g(x)$ (cf. equation 2.13) then $c'(x)$ will only be divisible by $g(x)$ if $e(x)$ is a codeword as, due to the linearity of the code, the sum of two codewords is another codeword. Therefore the error $e(x)$ will only be detectable if it is *not* divisible by $g(x)$.

The generator polynomial is chosen such that any error patterns required to be detected are not divisible by $g(x)$. For example, the purpose of the factor $(x + 1)$ in (2.17) is to ensure that all *odd*-weight error patterns are detectable. This is possible because no polynomial with an odd number of terms has $x + 1$ as a factor in the modulo-2 system. As the codewords are multiples of $g(x)$ (and hence $x + 1$) they all have even weight and therefore any odd weight errors will be detectable. To see this note that:

$$\begin{aligned} c(x) &= m(x)g(x) \\ &= m(x)(x+1)p(x) \\ \Rightarrow c(1) &= m(1)(1+1)p(1) = 0. \end{aligned} \tag{2.19}$$

As the value of c(1) is zero there must have been an even number of polynomial coefficients equal to one (i.e. the Hamming weight of *all* codewords is even). Therefore, if an error polynomial has an odd Hamming weight it will be detected as it won't be a codeword. This idea is used in section 4.5 to reduce the number of searches required in the packet merging algorithm.

Other considerations taken into account when designing generator polynomials can be found in [72] leading to binary (n, k) CRC codes that are guaranteed to detect the following error patterns:

- All single-bit errors if $deg(g(x)) \geq 1$

- All error patterns with an odd number of errors if the generator polynomial $g(x)$ for the code has an even number of nonzero coefficients (which is the case for all polynomials of form (2.16)).
- All single and double-bit errors provided the length n of the code generated by $g(x)$ is no greater than the exponent e to which it belongs[15].
- All single, double and triple errors if $g(x)$ is of form (2.16) and provided the length n of the code is not greater than e.
- All combinations of $d_{min} - 1$ (or fewer) errors (e.g. $d_{min} - 1 = 3$ for CRC-ITU).

The CRC can also detect many types of error bursts. An error burst of length b is defined in [72] to be any pattern of errors for which the number of bits between the first and last errors, including these, is b. The bits in between may or may not be in error giving a total of 2^{b-2} bursts of length b. The bursts detectable by a binary (n, k) CRC code are (specific values for codes with $n - k = 16$ parity bits, relevant to the CRC-ITU polynomial, are given in brackets) [67, 72]:

- All error bursts of length $n - k$ or less $(n - k = 16)$
- A fraction of the set of error bursts of length *equal* to $n-k+1$ $(n-k+1 = 17)$; the fraction equals $1 - 2^{-(n-k-1)}(99.997\%)$
- A fraction of the set of error bursts of length *greater than* $n - k + 1$; the fraction equals $1 - 2^{-(n-k)}(99.9985\%)$
- Any combination of two burst errors of length two or less provided $g(x)$ is of the form given in (2.16) and also $n \leq e$.

[15]A polynomial is said to belong to an exponent e if e is the smallest positive integer such that $g(x)$ divides $x^e + 1$.

2.4.5.2 Probability of undetected error

On a Binary Symmetric Channel (BSC) the probability of undetected error for a particular CRC code of blocklength n is determined by the *weight distribution* of the code. The weight distribution of a block code is the number of codewords of a particular weight, over all possible weights, and can be denoted by $[A_0^{(n)}, A_1^{(n)}, \ldots, A_n^{(n)}]$ where (n) represents the fact that it is a function of the blocklength n. The probability of undetected error P_{ue} is given by the following [73]:

$$P_{ue}(p,n) = \sum_{i=d_{min}(n)}^{n} A_i^{(n)} p^i (1-p)^{n-i} \tag{2.20}$$

where p is the probability of bit error, $A_i^{(n)}$ is the number of codewords of length n and weight i, and $d_{min}(n)$ is the minimum distance of the code at blocklength n. It can be seen that $d_{min}(n)$ has a large influence on the value of P_{ue}. The dependence of $d_{min}(n)$ on the blocklength n means that different packet sizes will produce different levels of undetected errors[16]. It is therefore of interest to maximise this value over all blocklengths of interest. According to [73], however, there are no CRC codes with 16-bit redundancy (denoted by CRC-16) that achieve a maximum minimal distance for all values of n. Codes with maximal d_{min} at short blocklengths do not have maximal d_{min} at large blocklengths and vice-versa — hence the reason for multiple CRC-16 standards.

The minimum distance of the CRC-ITU and CRC-ANSI codes is 4 for $2 \leq k < 2^{n-k-1} - (n-k)$ (typical in WSNs with packet sizes on the order of 300 bits) and 2 for $k \geq 2^{n-k-1} - (n-k)$ [74]. For other generator polynomials the minimum distance will not remain constant over such a large range of k — a fact that may need to be

[16]The optimal packet size for data communication in energy constrained WSNs is studied in [70] using energy-efficiency as the chosen optimisation metric.

taken into consideration when determining optimal packet sizes for WSNs. CRC codes in general will have a minimum distance between 2 and the weight of the generator polynomial [74].

An exact calculation of P_{ue} in (2.20) requires knowledge of the weight distribution of the code which is, in general, not known. It is necessary, therefore, to resort to the use of a bound instead. One common bound that holds for certain (n, k) linear block codes is (see [69] for more details)

$$P_{ue} \leq 2^{-(n-k)}[1 - (1-p)^n]. \tag{2.21}$$

A similar, weaker, bound that only a few known codes have been proven to satisfy [75] is:

$$P_{ue} \leq 2^{-(n-k)}, \tag{2.22}$$

which for a CRC-16 code is about 1.53×10^{-5}. An important point to note is that, as k is increased in size, the undetected error probability of a CRC code used for error detection on a BSC with crossover probability $0 \leq p \leq 1/2$ approaches that given by (2.22) [74]. For small values of k, P_{ue} can sometimes move beyond this bound as p increases, although this latter point is not true for every code. Larger packet sizes will result in a probability of undetected error close to (2.22) regardless of the value of p.

Codes that satisfy the bound in equation (2.22) are referred to as *good* error-detecting codes. Some classes of codes that satisfy this bound are distance-4 Hamming codes, double-error-correcting, and some triple-error-correcting primitive BCH codes of natural length [75]. If P_{ue} is an increasing function of $p \in [0, 1/2]$ then the code is referred to as a *proper* error-detecting code. The fact that a certain code of natural length satisfies this bound this does not necessarily imply that shortened versions of this code do. In fact, whether or not the shortened code satisfies this bound will depend on the degree of shortening which in turn depends on the length of the message being sent across the channel. For the CRC-ITU polynomial for example, provided that the codeword length satisfies $n \geq 2^8$ this bound is found to hold for all values of channel error probabilities (see fig. 1, [75]). For the weaker polynomial studied in the cited work ($g(x) = x^{16} + x^{14} + x + 1$) this number is increased to about $n \geq 2^{10}$. This particular code is equivalent to the CRC-ANSI code generated by $g_{ANSI}(x) = x^{16} + x^{15} + x^2 + 1$ in that their weight distributions are identical; this can be seen because $g(x) = x^{16} g_{ANSI}(x^{-1})$ [74].

In [76] it is noted that the performance of most standard polynomials (including the CRC-ITU polynomial) is unsatisfactory when compared with other possible polynomials over sizes from $18 \leq n \leq 1024$ (the values over which the study was undertaken). Although good and proper polynomials with the best minimum distance over certain intervals were studied in [73] the weight distribution of the minimum weight words of those particular polynomials is larger than that of the best polynomials described in [76] and hence they perform worse at low channel error probabilities. The reason for this can be seen by noting that equation (2.20) is approximated by

$$P_{ue}(p, n) \approx A^{(n)}_{d_{min}} p^{d_{min}} (1-p)^{n-(d_{min})} \tag{2.23}$$

for sufficiently small values of p.

As a point to note, the *guaranteed* error detection capability of a code is generally much inferior to the codes *actual* performance. The former stipulates that the code can detect *all* errors with a weight less than d_{min}. However, it doesn't take into account all detectable errors with a weight *greater* than d_{min} of which there are quite a few [77].

2.4.6 Error correcting capability of the CRC

It is interesting to note that if a code of any type can detect all double-bit errors, then it can, in principle, correct all single-bit errors. Consider the single-bit error polynomial $e(x) = x^i$ where i is the position of the error. If bit j in the polynomial is complemented the resulting error will be a polynomial of the form $x^i + x^j$ for all $i \neq j$ and 0 for $i = j$. The former case is always detectable by a double error detecting code and the latter case gives the desired codeword. A similar result holds for codes that can detect $2t$ errors. In this case all t-bit errors can be corrected [72]. In particular the CRC-ITU can detect all single, double and triple-bit errors and is therefore a single error-correcting code by the same reasoning. One problem with this technique, however, is the fact that a packet already containing $2t$ errors will have an extra error introduced at certain points during the iteration of the correction algorithm leading to the possibility of false-positives; the larger the value of $2t$ the less likely this is however.

Having looked in some detail at the ideas behind error detection/correction and, in particular, the CRC it is clear that very powerful error correction codes would be required to mitigate the effects of fading which can introduce large error bursts when the channel goes into a deep fade. Error correction is therefore not used alone in a fading channel. To assist in reliable communications a technique known as diversity can be used in conjunction with error correction/detection codes; this is discussed next.

2.5 Diversity

One of the most powerful techniques used to combat the deleterious effects of fading is *diversity*. Signals traversing statistically independent paths to the destination have a low-probability of experiencing deep fades simultaneously [11]. By sending the same data over independent paths and combining the individual signals at the receiver, it is possible to significantly reduce the effects of fading.

Any method that can produce statistically independent channels to the destination can provide diversity. For example, *frequency diversity* sends the same information at different frequencies ensuring that the frequency separation is larger than the *coherence bandwidth* of the channel; *time diversity* sends the same information over different time-slots ensuring that the time separation is larger than the *coherence time* of the channel; *spatial diversity* makes use of multiple antennas at either the transmitter, receiver, or both to create independent channels separated in space. The method for realising independent channels in this last case is to ensure that the antenna spacing is larger than some multiple of the wavelength. The actual spacing depends on the particular antenna used and the type of channel. In [11] an antenna spacing of $.38\lambda$ is shown to be required in a uniform scattering environment using isotropic transmit and receive antennas. However, for non-ideal setups a larger spacing is required. If there is an insufficient spacing between antennas, a lack of scattering, or a significant line-of-sight (LOS) component between source and destination, the individual transmission paths can be correlated with one another reducing the effects of diversity [78].

The *diversity order* of a system is the number of independent fading channels between the source and destination. As the diversity order approaches infinity, the fading channel reduces to an AWGN channel. This is intuitively satisfying: as the number of channels grows the chances that all of them are simultaneously in a deep fade goes to zero; hence, fading is effectively eliminated leaving only the AWGN to contend with.

Whichever diversity technique is used, the result is multiple independently faded signals at the receiver: which are then combined in some fashion. Coherent combining of the signals at the receiver provides an *array gain* which is the average increase in the SNR over that which would have been obtained with just one channel. Array gains are also possible in non-fading channels. On the other hand, the independent signal paths introduce a *diversity gain* which is the change in the slope of the error probability curve due to the use of diversity techniques. The concept of gain was met in a similar manner in section 2.4.4.1 where the coding gain of a system was introduced. Sometimes the term *SNR gain* is used when the reason for the gain is not of specific interest or is obvious from the context.

In general, combining is a linear operation and requires the use of *co-phasing*. Co-phasing removes the channel induced phase shift of each branch by use of an appropriate complex weighting factor applied at the receiver. The resulting signals entering the combiner will therefore be in-phase and add up constructively. The SNR at the output of the combiner is a function of the number of diversity paths, the fading distribution of each path and the particular techniques used for combining [11]. Some combining techniques include *Selection combining*, *Threshold combining*, *Equal gain combining* and *Maximal-ratio combining*.

With Maximal-Ratio Combining (MRC) the incoming signal on branch i is weighted by a factor $a_i e^{-j\theta_i}$ where the real-value a_i is the gain factor and θ_i is the phase of the incoming signal. The value of a_i on each branch is chosen to maximise the combined SNR. The parameter θ_i needs to be estimated using Channel State Information (CSI). The average SNR at the output of the combiner increases in a linear fashion with the number of diversity branches and is given by [11, 79]

$$\gamma_c = \sum_{i=1}^{M} \gamma_i. \tag{2.24}$$

where γ_c is the SNR at the output of the combiner, γ_i that of the i^{th} branch, and M is the number of diversity branches.

Selection Combining is somewhat less effective than MRC; however, it doesn't require knowledge of the channel phases. In this case the combiner outputs the signal on the branch with the highest SNR. Technically "combining" is somewhat of a misnomer in this case as the other signals are discarded altogether. The selection can be done pre-detection or post-detection and normally requires only one receiver, which is an advantage of this particular technique. No co-phasing is required making this scheme more practically implementable. The packet merging scheme developed in chapter 4 uses a form of post-detection selection combining to choose the best packet for correction.

For pre-detection selection combining over M independent and identically distributed Rayleigh fading channels, the average SNR at the output of the combiner ($\bar{\gamma}_c$) is given by [11]

$$\bar{\gamma}_c = \bar{\gamma} \sum_{i=1}^{M} \frac{1}{i} \tag{2.25}$$

where equal average SNR values on each branch i (i.e. $\bar{\gamma}_i = \bar{\gamma}$) is assumed. Selection combining therefore offers diminishing returns in terms of the SNR gain for increasing M. More details on any of the combining schemes mentioned can be found in [11, 63, 79], for example.

Spread-spectrum can also be used as a form of diversity. It is not usually used for diversity alone, however, since it requires significantly more bandwidth than other diversity techniques [11]. More specifically, spread-spectrum techniques are ineffective when the coherence bandwidth of the channel is larger than the spreading bandwidth or, equivalently, where there is relatively small delay spread in the channel [80]. In chapter 3 experiments are described in which the effects of using a spread-spectrum radio on the transitional region of a channel (which is caused in part by fading) are examined. The transitional region is shown to still have significant extent as will be seen.

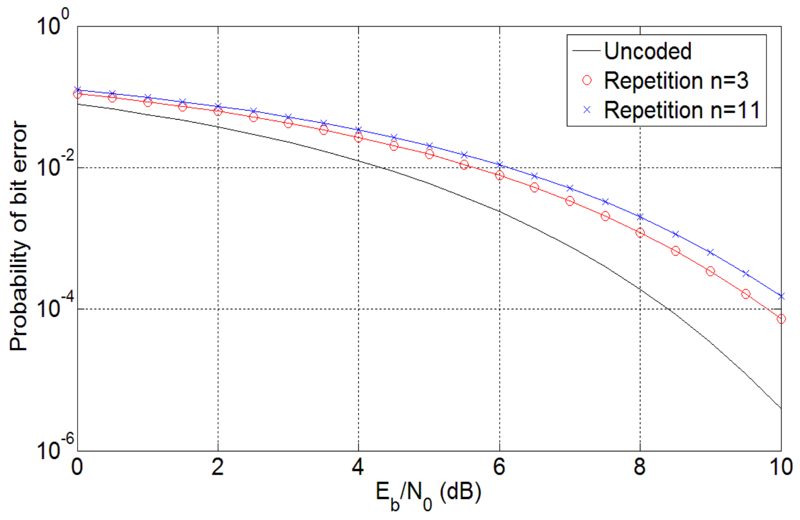


Figure 2.6: Comparison of an uncoded and a repetition coded BPSK system over a AWGN with $n = 3$ and $n = 11$ [4]

Following on from diversity techniques, the next section examines the use of repetition codes. These codes can be used with diversity in order to mitigate the effects of fading in an efficient and simple manner. Repetition codes can be considered a form of error correction codes but are dealt with in the next section due to their simplicity and inherent value when complementing diversity techniques.

2.5.1 Repetition Codes

Repetition codes are a class of error correction codes where each symbol is repeated n times for some value n ($n \in \mathbb{N}_0$). The rate R of such a code is $1/n$. A majority voting scheme is then used at the receiver in order to estimate the value of each symbol. The value of n is normally odd although even values are possible at the expense of possible decoder failures (when exactly half of the repeated symbols are in error). Assuming odd n however, the guaranteed error correction capability is given by $t = (n - 1)/2$.

Repetition codes are very easy to implement and are therefore used in situations where the cost and complexity of coding is a primary concern [81]. However, using a simple majority voting scheme, repetition codes are of no use over an AWGN channel offering *zero* coding gain with optimal soft-decision and *negative* coding gain with hard decision [82]. An example of the latter case is shown in figure 2.6. It can be seen that the performance of the system actually decreases with increasing n. The reason for this is because the same energy per bit used for the uncoded system has to be spread over the bits in the coded system increasing the probability that the demodulator will make an error for any coded bit. This, of course, is the case for any error correction scheme. However, for repetition coding the error correction capability of the majority decoder is not enough to compensate for the spreading of the bit energy. A similar situation was noted in section 2.4.4.1 where the Hamming code produced a negative coding gain at low SNRs.

It may seem fruitless therefore to use repetition coding at all. However, in fading channels a different result is observed [83]. The use of repetition codes over a nonselective Rayleigh fading channel is considered in [84]. It is shown that for modulation schemes such as Phase Shift Keying (PSK), Differential Phase Shift Keying (DPSK) and Frequency Shift Keying (FSK) [66] repetition codes actually improve the performance when the coded-bit energy is sufficiently high [84]. This improvement is determined by the number of repetitions and an optimal number exists that minimises the BER. For binary modulations the probability of error approaches an exponential drop-off versus SNR for increasing n — a satisfying result [85].

2.6 Multiple-Input, Multiple-Output (MIMO) Systems

Spatial diversity was introduced in section 2.5 and involves the use of multiple antennas at either the transmitter, receiver, or both to create independent channels separated in space. It owes its popularity to the fact that, unlike time and frequency diversity, it does not reduce the spectral efficiency of a system. Spatial diversity is classified as being either *transmit diversity* or *receive diversity* depending on whether there are multiple antennas at the transmitter or receiver, respectively.

With transmit diversity the transmission power must be divided equally among each of the individual antennas. In order to obtain an array gain using transmit diversity channel knowledge is required at the transmitter so that the signals are coherently combined when they arrive at the receiver. The array gain obtained is reduced by a factor of M relative to that of receive diversity, where M is the number of antennas used. This is because the power is divided among each of the M antennas. However, transmit diversity can also provide an array gain using transmit beamforming [86]. Beamforming involves placing the antennas sufficiently close together so that their fading is highly correlated. This of course reduces the level of diversity and a trade-off is therefore necessary (see [87] for more details). Transmit diversity is more desirable in systems where resources such as space, power, and processing capability are more readily available at the transmitter than the receiver. With receive diversity no increase in transmission power or bandwidth is required and channel knowledge is easily obtained relative to that of transmit diversity.

A Multiple-Input, Multiple-Output (MIMO) system uses multiple antennas at the transmitter and/or receiver in order to leverage spatial diversity gains. The idea was first introduced by Foschini and Gans [88] and, independently, by Telatar [89]. An overview of MIMO can be found in [90] and of its energy-efficiency in [91]. Different combinations are possible for MIMO systems and although the acronym MIMO generally encompasses all combinations they are often categorised more specifically. Systems using only one transmit and receive antenna can be viewed as Single-Input, Single-Output (SISO) systems; systems having multiple antennas at the transmitter and only a single antenna at the receiver are referred to as Multiple-Input, Single-Output (MISO) systems; systems having multiple antennas at the receiver and a single antenna at the transmitter are referred to as Single-Input, Multiple-Output (SIMO) systems.

Ordinarily the attractiveness of MIMO systems is the potential capacity gains using *spatial multiplexing* with no additional power or bandwidth expenditure. This is achieved by transmitting independent data streams over each antenna (as opposed to the same data in the case of diversity). The streams are separated at the receiver by means of each paths *spatial signature*. Using this technique, the maximum data rate over a fading channel at which error free transmission is possible is proportional to the smaller of the number of transmit or receive antennas (assuming all possible paths are independent of one another) [78]. For low power devices however the emphasis is on the diversity gain as opposed to the multiplexing gain.

In MIMO systems it is possible to use Space Time Block Codes (STBCs) such as the Alamouti scheme [80]. Space-time coding is a type of coding across both space and time which was first introduced by Tarokh *et al.* as a novel means of providing transmit diversity using multiple antennas on the transmitter [92]; a brief review can be found in [12]. The coding in space is obtained by using multiple antennas at the transmitter. Using space-time coding, diversity gains can be obtained without requiring channel knowledge at the transmitter [93].

The Alamouti scheme is defined for a 2-antenna system and achieves a diversity order of 2 *without* knowledge of the channel at the transmitter. The disadvantage is that the array gain of 2 obtainable by an MRC system is reduced to 1 in this case [11]. As noted in [86] the Alamouti scheme is more efficient than a simple repetition code as it spreads the information onto two orthogonal dimensions and therefore utilises the available degrees of freedom more efficiently.

In [91] the total energy consumption per bit is determined for a MIMO and SISO system (using the Alamouti scheme for the former) in a flat Rayleigh fading channel. It is shown that, for the simple BPSK system studied, with a transmitter-receiver separation of less than 62m the SISO system required less energy per transmitted bit. This is due to the more complex circuitry required by the MIMO system in the receiver. By optimising the constellation size, it is then shown that large energy savings can be afforded by both the MISO and SISO case and that the threshold distance (i.e. the point where MIMO becomes more energy-efficient than SISO) can be reduced significantly (down to approximately 3.8m for the particular setup described). A similar set of results is shown to hold for the cooperative MISO or SIMO cases except that the threshold distance is somewhat larger (approximately 15m). On small mobile units or wireless sensor nodes the use of multiple antennas may not be feasible due to insufficient mounting space. It is in these situations that the cooperative scenario becomes more interesting. This is discussed next.

2.7 Cooperative Communications

The concept of MIMO communications has been shown to offer large improvements for wireless systems in fading channels. However, the small size of the nodes in WSNs means that they may not have sufficient space to carry multiple antennas. A natural adaptation of MIMO systems to WSNs therefore is a promising technique known as *cooperative communications*. In cooperative systems each particular node only has one antenna, unlike MIMO systems which have multiple send and/or receive antennas at each node. However, neighbouring nodes can act as the extra antennae in the cooperative scheme in order to retransmit the information sent by the original node, if required. The destination then combines the information received from the source node and its cooperating neighbour. An obvious disadvantage of cooperative systems over MIMO systems is the imperfect channel between cooperating nodes. In a MISO system, for example, an exact replica of the signal is sent from each antenna; in cooperative systems the original signal and its estimate are sent. Results from MIMO

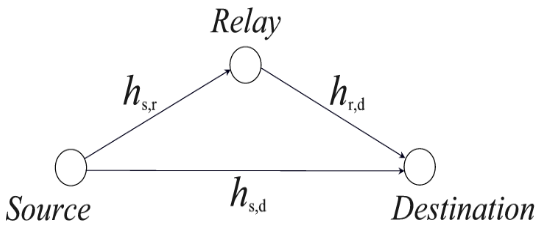


Figure 2.7: Model of a cooperative system.

systems are therefore sometimes used as bounds on the performance of cooperative systems [94].

Figure 2.7 depicts a simple three node cooperative system: a Source (S), a Relay (R) and a Destination (D). In general, the relay will also have its own data to send so a more appropriate name might be the cooperating node. However, the terms relay and cooperating node will be used interchangeably (even in the cases where the relay acts solely to assist the source node) as the distinction is not of relevance in this thesis. The channel coefficients $h_{s,r}$ (from source to relay), $h_{s,d}$ (from source to destination) and $h_{r,d}$ (from relay to destination) represent the complex fading gains over each channel. Generally in this scenario two-phases of operation are required. In the first phase the source transmits to the relay and the destination overhears a possibly weak version of the signal. In the second phase the relay resends the information received from the source if a) the destination incorrectly received the original signal from the source and b) the relay decoded correctly[17]. When the source-relay channel is better than the relay-destination channel (i.e. $|h_{s,r}| > |h_{r,d}|$) the system emulates more closely a transmit diversity scheme. When the source-relay channel is worse than the relay-destination channel (i.e. $|h_{s,r}| < |h_{r,d}|$) the system emulates more closely a receiver diversity scheme. In a cooperative system each node has its own information to send also which is not the case in the MIMO setup.

[17]This is, of course, only one example and a number of variations are possible.

Another commonly used term found in the literature is *cooperative diversity*. A clarification of the terms cooperative communications and cooperative diversity is noted in [57]. Cooperative communications is an interaction between distributed nodes to jointly transmit information in wireless environments. Cooperative diversity is effectively the result of this interaction which leverages the spatial diversity available among the distributed radios. These two terms are often used interchangeably along with another: *cooperative relaying* which makes reference to the fact that this scheme is an extension of a basic relaying technique.

A good overview of cooperative communications is given in [10]. The idea, however, was first introduced in [55] and [56] where it is shown that cooperation not only improves the capacity for the cooperating nodes but also improves the robustness of the system in terms of achievable rates. Significant results are developed in [94] and it is noted that the broadcast nature of wireless communications is the key property allowing nodes to cooperate: in theory, it is possible for transmitted signals to be received by any number of terminals. This fact was actually taken into account in early work by both Van der Muelen [95] and, Cover and Elgamal [96] which paved the way for the cooperative techniques now being developed.

In [55] it is shown that when the source-destination and relay-destination are of similar quality, but are less than that of the source-relay channel, cooperation is of benefit. However, if the quality of the source-relay channel begins to degrade, then the achievable rate region[18] approaches that of no cooperation. Stated another way: more power is applied to cooperation when the source-relay channel is favourable and vice-versa [10]. When the quality of the source-destination and relay-destination channels is different, cooperation improves the achievable rate region; however, it is the user that has the channel with the highest fade that benefits most. From the point of view

[18]The achievable rate region for distributed sources of information (i.e. multiple senders) is the set of all achievable rates.

of WSNs, these achievable rate regions translate into lower energy consumption due to improved outage probabilities.

An impractical assumption in the theoretical model of [55] is that no *orthogonality* constraint is imposed on the system. Also known as the *half-duplex* constraint, this implies that the terminals cannot transmit and receive simultaneously on the same channel; this constraint introduces a loss of spectral efficiency when compared with MIMO systems using multiple onboard antennas. The *full duplex* assumption of [55] is not possible with current radio technology [94]. The pioneering work carried out in [94] developed and analysed low-complexity cooperative diversity protocols that not only removed the full duplex assumption but also required only partial CSI at the receiver. Furthermore, delay-constrained scenarios were considered and performance was characterised by outage probabilities. It is noted that the potential impact of the protocols developed become less substantial when other forms of diversity are available.

In [97] a number of possible cooperative schemes are discussed based on the different types of forwarding strategies: *detect-and-forward*, *amplify-and-forward* and *decode-and-reencode*. In the detect-and-forward scheme, the cooperating node fully demodulates and decodes the incoming signal. It then retransmits the signal using the same encoding as the source node. For the amplify-and-forward scheme the cooperating node acts as an analog repeater and amplifies and retransmits the incoming signal without demodulating it; in this case the noise also gets amplified. Decode-and-Reencode is a form of *coded cooperation* which was developed in [98] and uses incremental redundancy to achieve improved performance.

The protocols can then be further categorised into the type of protocol: *fixed protocols*, *adaptive protocols* and *feedback protocols*: giving a total of nine possible protocols. For fixed protocols the relay *always* forwards a processed version of its received message. The adaptive protocols use some form of threshold rule at the relay to decide whether or not to forward. With feedback protocols the relay only forwards if explicitly requested by the destination. In chapters 4 and 5 the experimental setups makes use of both feedback and the CRC to determine whether or not the relay or source will forward the packet.

The ideas in [97] were inspired by [94] where three protocols are described: fixed relaying, selection relaying and incremental relaying. The term fixed is used in the same sense as in fixed protocols, i.e. the relay always forwards a processed version of its received message. Selection relaying uses the measured SNR between cooperating nodes to determine whether to use cooperation or not. If the measured value of $|h_{s,r}|^2$ (i.e. from source to relay) is less than a certain threshold the source repeats its transmission in the second phase using repetition coding or some more powerful technique. If the measured value of $|h_{s,r}|^2$ is larger than the given threshold the relay forwards the estimate of the source message obtained in the first phase. Incremental relaying uses limited feedback from the destination to improve upon the spectral efficiency of the fixed and selection relaying scheme. The fixed relay, decode-and-forward strategy does not provide diversity. Requiring the relay to fully decode the source message every time essentially reduces the performance to that of direct transmission between the source and relay link. Selection decode-and-forward is shown to be identical to fixed amplify-and-forward (in terms of diversity gains) at high SNRs. Incremental amplify-and-forward achieves full second order diversity.

Considering the low-power, resource constrained nodes under consideration in this thesis[19] a number of these protocols are not practical: the most obvious being the amplify-and-forward protocol. Moreover, amplify-and-forward does not provide full order diversity in the low-SNR regime as the majority of the signal at the relay is noise which then gets amplified and sent on to the destination [99]. However, the advantage

[19]Recall that, in many cases, the size of future nodes will be reduced in size leading to similar constraints found with today's architectures, despite advances in processor and peripheral architectures.

of noise-reduction by decoding at the relay can be offset by fading effects which tend to limit the effects of decode-and-forward [57]. Decode-and-Reencode is possible although for most currently available mote-class devices the reencoding would have to be performed in software when implemented practically. This of course may not be the case with future devices however. In [100] it is shown that a form of the decode-and-reencode protocol referred to as *coded cooperation* offers a 1-2dB advantage over amplify-and-forward techniques. Error propagation is reduced by the use of a CRC also.

In terms of the type of forwarding strategy, currently the most obvious choice for low power, resource-constrained WSNs is the adaptive decode-and-forward protocol, using the CRC to determine whether or not to forward at the relay. An example of such a protocol is the Simple Adaptive Decode-and-Forward protocol developed in [101] and is the one most closely followed in section 4.3. However, in chapter 5 a more advanced feedback protocol is used in which timeouts, at both the source and relay, overcome the problem of missed packets; a similar technique was used in [102].

Other protocols/codes are possible in addition to those described above. A *network coding* based protocol for cooperative communications is described in [103]. In this case, both the source and relay combine their own messages with the received relay and source messages, respectively. This is done by performing a "network coding" operation which, in the cited work, is a simple modulo-2 sum. One possible protocol consists of two phases in which there are a total of four time slots. In phase one, the transmitter and relay broadcast their respective messages X_s and X_r (in separate time slots). In phase two, both the source and relay then send the modulo-2 sum of the source and relay messages to the destination (i.e., $X_{s\oplus r} = X_s \oplus X_r$). A simple exclusive-OR operation is then performed, if necessary, to recover the message (e.g. $X_r \oplus X_{s\oplus r} = X_s$). It is shown that network-coding based protocols are suitable when the source-relay channel is unreliable. When the source-relay channel is good, however, it is better not to use network coding. Network coding is also shown not to be suitable for fixed protocols; more details can be found in [103].

Distributed space-time codes may also be possible [104] but inherently have many challenges. For example, the model in figure 2.7 shows only one cooperating node (i.e. the relay). However, in practice, the actual number of cooperating nodes is essentially random. It depends not only upon the state of the channel at any particular time but also on the amount of neighbours available to the source (which may change due to node mobility, link failures etc.). This issue, coupled with the imperfect estimates of the information at the cooperating node, make it very difficult to create distributed space-time codes from those already being used in MIMO systems [105].

2.8 Summary

This chapter has given an overview of Wireless Sensor Networks and some of the challenges they pose. It has mainly focused on techniques for improving the link reliability in both AWGN and fading channels. These techniques include error control coding, repetition coding and diversity. It was noted that both FEC and simple ARQ schemes have certain drawbacks that make them unsuitable to be used alone on WSNs. Using a combination of the two, however, it is possible to develop schemes that offer benefits for WSNs. Inherent in the implementation of some of the techniques discussed is the use of combining. The complexity of some of the combining schemes available may not be suitable for resource-constrained devices and there is a need to develop more simplistic, yet efficient, techniques; this is the study of later chapters. These combining techniques can be used with a number of cooperative schemes presented in this chapter. The different schemes are summarised here for clarity:

- Amplify-and-forward Non-regenerative forwarding procedure in which the signal is received and amplified at the relay with no decoding.

- Decode-and-forward Regenerative forwarding procedure in which the signal is received, demodulated, decoded and reencoded using same encoding procedure.
- Decode-and-Renencode Also referred to as coded-cooperation, a regenerative forwarding procedure in which the signal is received, demodulated, decoded and reencoded using a different scheme.
- Network coding

These protocols can be used with the following protocols

- Fixed - The relay *always* forwards a processed version of its received message. When used with decode-and-forward this does not achieve full order diversity.
- Adaptive (Selection) - The adaptive protocols use some form of threshold rule at the relay to decide whether or not to forward. For resource-constrained nodes a similar technique is to use the result of the CRC to determine whether or not to forward.
- Feedback (Incremental) - With feedback protocols the relay only forwards if explicitly requested by the destination.

Before developing the combining schemes to use with these protocols, the following chapter looks at the effect of using a spread-spectrum radio on the reception region around a typical WSN. The aim of doing this is to determine the extent of the transitional region (an area around WSNs in which combining techniques would be needed) when using spread-spectrum, a readily available form of diversity on IEEE 802.15.4 compliant devices.

Chapter 3

Reception Region Characterisation

Experimental studies on Wireless Sensor Networks (WSNs) have revealed the existence of three distinct reception regions [19–21]. These regions can be classified as connected, transitional, and disconnected. Their location and size can have a significant impact on the performance of communication protocols [106]. The underlying causes of the transitional (or so-called "grey") region have been determined in previous work using both analytical and experimental techniques [19–21, 106]. It has been shown that the transitional region is caused in part by multi-path fading. Techniques described in the previous chapter, such as diversity, can possibly be used to reduce the transitional regions extent as a result.

This chapter represents a preliminary study carried out to motivate the need for the packet combining techniques developed in later chapters. The IEEE 802.15.4 standard makes use of a spread-spectrum technique which can also be used as a form of diversity (as noted in chapter 2). It was suggested in [22] that the use of spread-spectrum may reduce the extent of the transitional region. Although this region was shown to have significant extent in previous work (such as [22] and [20]) the radios used in determining this did not employ any mechanisms with the ability to combat fading, such as spread-spectrum. This left the question open as to the extent of the transitional region when using a radio of this type. A significant reduction in its extent may reduce the need for other techniques.

The results of this chapter show, however, that even with the use of a spread-spectrum radio, for the static setup considered (no mobile nodes were used)[1], the transitional region still has significant extent — motivating the need for other techniques such as packet combining. Many WSN deployments will be of a static nature (e.g. [35], [37] and [30]), justifying the need for such a setup. However, it is of course entirely possible that spread-spectrum techniques used in more dynamic environments will have more of an impact on the extent of the transitional region. This was beyond the scope of this preliminary study.

3.1 Introduction

The reception region around a wireless communications device is significantly more complex than the simple connected and disconnected regions assumed in some simulator studies (see, for example, [107]). Several works have reported the existence of a third "transitional" region in which the Packet Reception Rate (PRR) is quite erratic. The extent of this region is important as upper layer protocols may disregard its effects leading to, for example, inefficient routing topologies. This can be understood due to the asymmetrical nature of the links in this region in which the forward link may be more reliable than the reverse link, or vice-versa. Link asymmetries are due to manufacturing variances in the receivers used which may have slightly different sensitivities and noise floors. The issues with this can arise in routing algorithms such as TinyAODV [108] where routes are identified using link qualities in the reverse direction [61].

One commonly used definition of the transitional region is the area around a wireless device in which the Packet Reception Rate (PRR) ranges from approximately 90% down to about 10%. Consideration of this region is critical for forwarding strategies in WSNs as it determines how much energy has to be spent for successful packet de-

[1]The channel itself was not completely static due to the movement of people and objects in the surrounding environment. These are second order effects however and are ignored.

livery [109]. Knowledge of its extent and underlying causes is therefore a necessity. In [22] a thorough evaluation of the transitional region is carried out using analytical and experimental techniques. The practical experiments described make use of a radio architecture that employs Non-Return to Zero (NRZ) encoding and Non-Coherent Frequency Shift Keying (NCFSK) [66]. The results obtained show that the transitional region is caused, in part, by multi-path fading. A suggestion is made in the cited work that other architectures, employing techniques that combat multi-path fading, may be successful in reducing the width of the transitional region.

In this chapter an attempt is made to determine the extent of the transitional region when employing a spread-spectrum based radio. Towards this end an experimental setup is described in which Packet Reception Rate (PRR) measurements are used to determine the extent of the transitional region. An analytical calculation is then carried out using the probability of bit error equation given in the IEEE 802.15.4 standard [5]. This calculation is then repeated using the experimentally evaluated BER curves obtained from two different environments. The results are then compared with those of [22].

The remainder of this chapter is organised as follows. In section 3.2 previous related work in this area is briefly discussed. Section 3.3.1 discusses the experimental setup used to determine the Packet Reception Rate (PRR) from which the transitional region is determined. Section 3.3.2 then uses the theoretical probability of bit error equation from the IEEE 802.15.4 standard to help determine the transitional region analytically. Section 3.3.3 experimentally evaluates the BER curves for comparison with the results of section 3.3.2. The overall results are then analysed in section 3.4 and finally conclusions are drawn in section 3.5.

3.2 Related Work

In [22] a model of the transitional region is introduced. This model makes use of the log-normal shadowing path loss model as described in [110]:

$$P_r(d) = P_t - PL(d_0) - 10\,\eta \log_{10}\left(\frac{d}{d_0}\right) + \mathcal{N}(0,\sigma) \tag{3.1}$$

where $P_r(d)$ is the received power at distance d, P_t is the transmit power, $PL(d_0)$ is the power loss at a reference distance d_0, η is the decay factor and $\mathcal{N}(0,\sigma)$ is the shadowing component with zero mean and standard deviation σ. For non-static setups (e.g. with nodes that are mobile) the value $\mathcal{N}(0,\sigma)$ is a random *process* that is a function of time. However, in the static environment considered in this chapter it is simply a random variable. From this equation expressions can be derived to calculate the extent of the transitional region[2] [106]:

$$\begin{aligned} d_s &= 10^{\frac{P_n+\gamma_U-P_t+PL(d_0)+2\sigma}{-10\eta}} \\ d_e &= 10^{\frac{P_n+\gamma_L-P_t+PL(d_0)-2\sigma}{-10\eta}} \end{aligned} \tag{3.2}$$

where d_s and d_e are the transitional region's start and end points, respectively, γ_U and γ_L represent the upper and lower SNRs above which the PRR is greater than 90% and 10%, respectively, and P_n represents the noise floor of the system. In order to compare different experimental setups another expression can be derived referred to as the *transitional region coefficient*. This is the ratio of the lengths of the transitional and connected regions:

[2]The reference distance d_0 is set to 1m in this case. For distances greater than this d_s and d_e need to be multiplied by the reference distance d_0.

$$\begin{aligned}\Gamma &= \frac{d_e - d_s}{d_s} \\ &= 10^{\frac{(\gamma_U - \gamma_L) + 4\sigma}{10\eta}} - 1\end{aligned} \tag{3.3}$$

The next section makes a determination of the size of the transitional region using a spread-spectrum radio. It does this in three ways: experimentally, using a theoretical equation for the probability of bit error, and finally using experimentally derived BER curves. The experimental setup described makes use of the Received Signal Strength Indicator (RSSI) [2] to determine the extent of the transitional region. The accuracy of the RSSI value as an estimator of true signal power is important if correct values are to be observed. The results obtained in [111] suggest that RSSI is a good estimator of the quality of a link. This motivated its use for the experiments in this chapter. However, some discrepancies were found as will be seen.

3.3 Transitional Region Extent

The radio architecture used in the experiments described herein was the CC2420 from Chipcon [2]. The CC2420 operates at a frequency of 2.4GHz and uses Direct Sequence Spread Spectrum (DSSS) making it an obvious choice for testing the assumption of [22] that a device with spread-spectrum capabilities will also reduce the extent of the transitional region.

3.3.1 Experimental

In order to determine the PRR, twenty Tmote Sky motes [13] were evenly spaced in a linear setup for both an indoor (corridor) and outdoor (open field) environment[3]. These

[3]The spacing between each node was set to 2m and 1m for the indoor and outdoor environments, respectively.

motes are equipped with an MSP430 microcontroller [112] and a Chipcon CC2420 transceiver [2] — the latter being a spread-spectrum device, as required. The transmit power was set to -15 dBm and each node sent 6000 packets in turn. The remaining nodes (all in receive mode) determined the number of packets correctly received from the current transmitter along with the average RSSI over the 6000 packets sent.

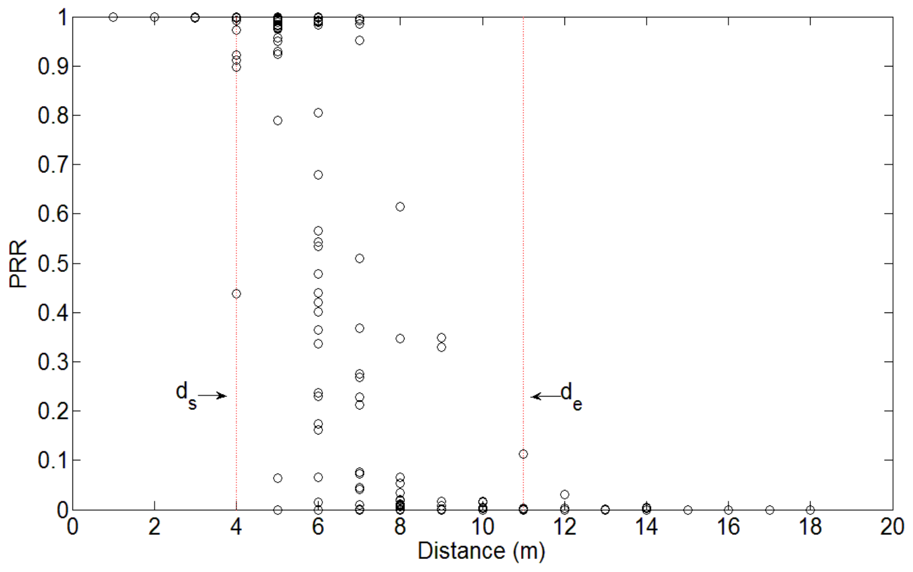


Figure 3.1: PRR outdoors with each mote separated by a distance of 1m. The nodes were placed on small cardboard boxes about 5cm in height. The transmit power level of each node acting as source was set to -15 dBm.

The indoor environment was a long corridor; the outdoor, an open field. A similar setup was described in [22] where the aisle of a building and a football field were used for both the indoor and outdoor environments, respectively. Figure 3.1 shows a plot of the PRR for the outdoor environment with the transitional region marked by the vertical lines labelled d_s and d_e. Figure 3.3 shows the corresponding signal strength decay over distance. Similar plots were obtained for the indoor environment (cf. figures 3.2 and 3.4).

It should be understood that what is being measured is the individual PRR and average

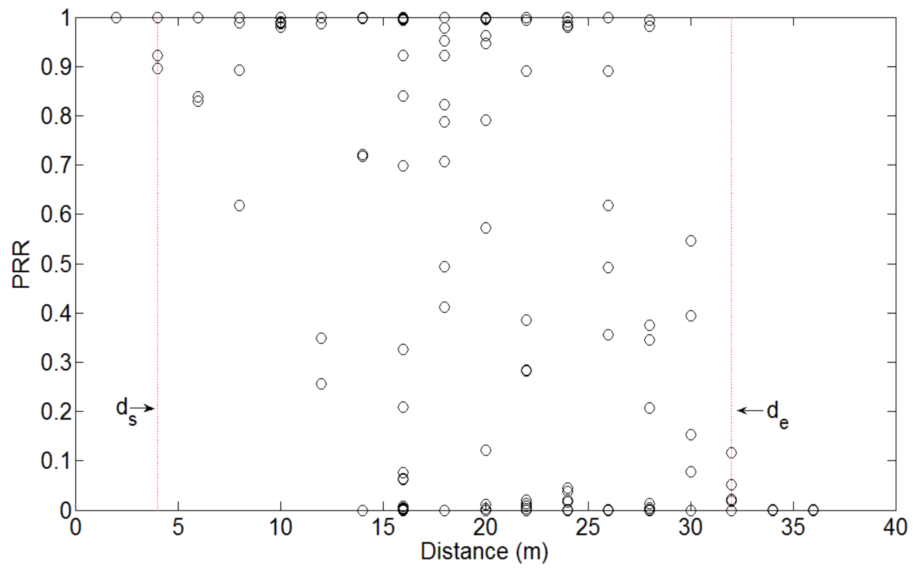


Figure 3.2: PRR indoors with each mote separated by a distance of 2m. The transmit power level of each node acting as source was set to -15 dBm.

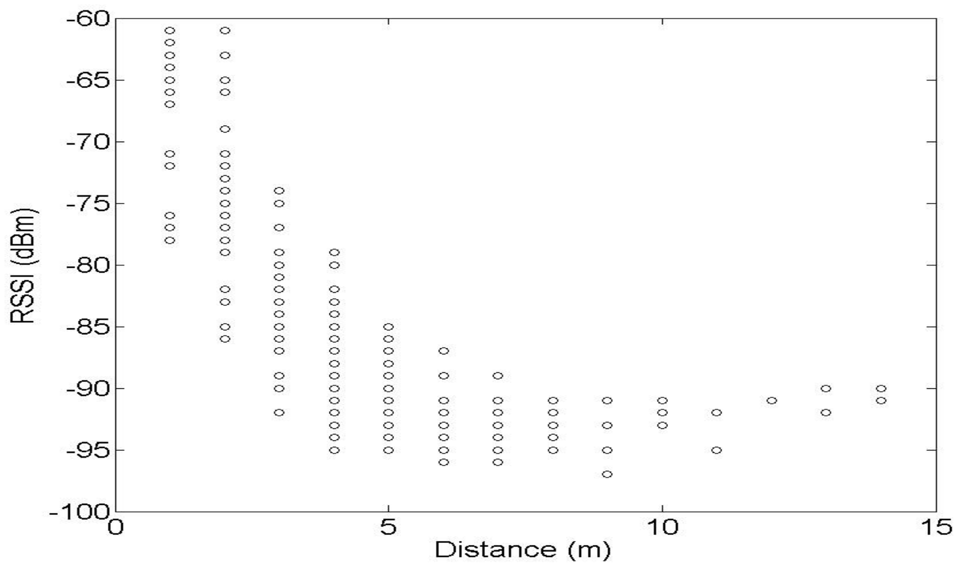


Figure 3.3: RSSI outdoors with each mote separated by a distance of 1m. η is approximately 3.6 and σ is approximately 4.1.

RSSI (over each 6000 packet set) on each receiver placed at increasing distances from the source node (in a linear setup). For ease of exposition let the set of all nodes at

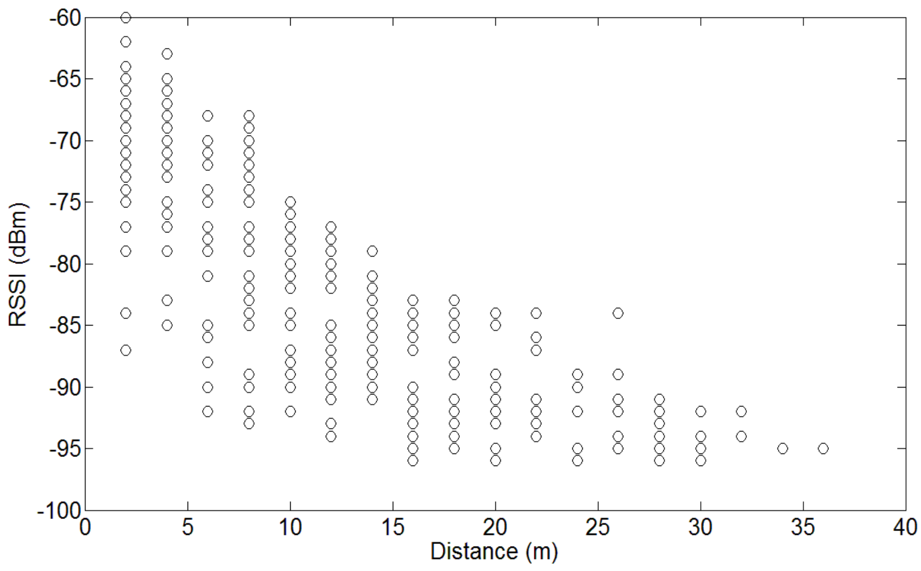


Figure 3.4: RSSI indoors with each mote separated by a distance of 2m. η is approximately 2.6 and σ is approximately 5.6.

distances $m \times i$ from the source node n_0 (where $m = 2$m for the indoor case and $m = 1$m for the outdoor case) be represented by n_i, where $i \in \{1, 2, ..., 19\}$. In an ideal scenario (i.e. free space) each node in the set n_i would experience the same value of SNR (due to path loss alone) and the level of SNR experienced would be larger than that of the set n_{i+1}. In a multi-path environment this is not true however. In general the levels of SNR experienced by each member of the set n_i would be different and a member of the set n_i may experience a smaller SNR than a member of the set n_{i+1} (the SNR for each individual node would remain relatively constant throughout the experiment, however, due to the static environment). The reason for multiple nodes in the set n_i is due to the particular setup used. Each of the twenty nodes took it in turn to act as source. So, considering the set n_1, for example, for each node acting as source, except for the two end cases, two nodes are added to this set: one on each side of the current source node; the two end cases add one extra node each. Figures 3.1, 3.2, 3.3 and 3.4 show all PRR and average RSSI values obtained from the nodes in each set n_i

plotted against the distance $m \times i$.

From figure 3.2 the values of d_s and d_e for the indoor environment can be seen to be 4m and 32m, respectively, resulting in $\Gamma = 7$. Comparing these with the results from the indoor experiments in [22] ($d_s = 6.9$m, $d_e = 25.9$m and $\Gamma = 2.8$) it can be seen that the transitional region still has significant extent for the spread-spectrum setup.

The outdoor environment is quite different where it can be seen that $d_s = 4$m, $d_e = 11$m and $\Gamma = 1.8$. Again comparing these with the outdoor setup in [22] ($d_s = 3.2$m, $d_e = 8.6$m and $\Gamma = 1.7$) it can be seen that the transitional regions are very similar in this case. The reason for this similarity is possibly due to the two outdoor environments being more alike than the two indoor environments. The outdoor environments contained no walls, furniture and other items close to the nodes. This was not the case with the indoor environment and there are many more factors to consider. However, the fact that the transitional region is a lot larger for the indoor environment is caused by a smaller decay factor (η) and a larger standard deviation σ (cf. equation 3.1).

The larger decay factor outdoors is likely to be caused by absorption of the signal energy in the moisture contained in the soil and grass. The larger decay factor was the reason for using a separation distance of 1m in the outdoor case. Had the nodes been separated by 2m very few of them would have received any packets at all. The larger standard deviation for the indoor environment is likely to be caused by increased levels of multi-path — there are more obstacles indoors creating more paths for the signal to take.

Environment	η	σ	$PL(d_0)$
Outdoors	3.6 (4.7)	4.1 (4.6)	49.7 dB
Indoors	2.6 (3.0)	5.6 (3.8)	50.7 dB

Table 3.1: The values obtained for the channel parameters. The reference value d_0 is 1m for the outdoor environment and 2m for the indoor environment.

Using the results plotted in figures 3.3 and 3.4 the channel parameters were determined and are shown in table 3.1. The values of eta were calculated using least squares fitting and the $sigma$ values by taking the mean of the standard deviation values obtained over multiple distances. The values taken from [22] are included in brackets for reference and it can be seen that they are quite close, although the decay factor for the setup being described in this chapter is smaller in both cases. For the indoor environment σ is a lot larger also which adds to the length of the transitional region. While it is not possible to draw concrete conclusions as to why these differences occurred (e.g. as a result of differences in the environments or due to the different radios) it is possible to say that the transitional region still has significant extent in similar environments even when using a spread-spectrum radio. It is precisely in these situations that packet combining would be required.

3.3.2 Probability of Bit Error

This section looks at calculating the transitional region using the probability of bit error p_e. In [22] the PRR is shown to be determined by p_e as follows:

$$\text{PRR} = (1 - p_e)^{8f} \tag{3.4}$$

where p_e is the probability of bit error and f is the frame length of the packet (in bytes). The value for p_e was taken from the IEEE 802.15.4 standard [5]:

$$p_e = \frac{8}{15}\frac{1}{16}\sum_{k=2}^{16}(-1)^k \binom{16}{k} e^{20\cdot\text{SINR}\cdot(\frac{1}{k}-1)}. \tag{3.5}$$

where SINR represents the instantaneous Signal-to-Interference plus Noise Ratio.

Rearranging equation (3.4) as

$$p_e = 1 - \text{PRR}^{\frac{1}{8f}} \tag{3.6}$$

enables the values for γ_U and γ_L to be determined. This is achieved by using equation 3.6 to calculate the values of p_e for a PRR of 90% and 10% (which marks the beginning and end of the transitional region, respectively, as discussed in section 3.2). The plot of equation 3.5 can then used to determine the values of γ_U and γ_L which result in the p_e values obtained.

For each experimental setup a frame size (f) of 39 bytes was used, leading to $\gamma_U = -.3$ dB and $\gamma_L = -2.3$ dB. The channel parameters were obtained from the results of the previous section (cf. table 3.1). Inserting γ_U, γ_L and the appropriate channel parameters into equations 3.2 and 3.3, along with $P_t = -15$ dBm and $P_n = -97$ dBm (see equation 3.9), the extent of the transitional region can be calculated. The values obtained are $d_s = 12.2$m, $d_e = 105.7$m and $\Gamma = 7.7$ for the indoor environment with $d_s = 4.8$m, $d_e = 15.4$m and $\Gamma = 2.2$ for the outdoor environment. It can be seen that the values for Γ agree quite closely for both environments (recall it was 7 for the indoor and 1.8 for the outdoor environment). However, for the indoor environment the values of d_s and d_e disagree quite strongly with those obtained experimentally (4m and 32m, respectively). The same is true for the outdoor environment although to a lesser degree (4m and 11m, respectively).

The discrepancy was initially thought to be due to the simplified model for the probability of bit error used (eq. (3.5)). The modulation scheme used by the IEEE 802.15.4 (and hence the CC2420) is Offset Quadrature Phase Shift Keying (O-QPSK) with half sine pulse shaping which is the same as Minimum Shift Keying (MSK) [64]. O-QPSK has the same theoretical bit error performance as Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK) assuming coherent detection [66]. The same is true for MSK detected using a matched filter to recover each of the quadrature components independently [66]. Equation 3.5 is clearly quite different to those for BPSK and QPSK (see [66]). This is because MSK can also be viewed as a special form of Continuous-Phase Frequency Shift Keying (CPFSK) where the deviation index is exactly equal to $\frac{1}{2}$ [63]. However, the error probability for CPFSK also depends on whether the detector is coherent or non-coherent and whether it uses symbol-by-symbol detection or sequence estimation [11]. Another possible discrepancy lies in whether or not hard-decisions are made at chip level. In the CC2420 data sheet [2], for example, it is specified that *soft* decisions are made at chip level. This fact is not taken into account in the derivation of 3.5. It can be seen, therefore, that there are a number of simplifying assumptions in the derivation of 3.5 which may not lead to the correct probability of bit error equation for all possible chipsets. For all of these reasons an attempt was made to determine the BER experimentally.

3.3.3 Bit Error Rate

To determine the BER a pair of Tmote Sky motes were used. It is possible to configure the CC2420 transceiver to run in a number of different modes: one of which is an *unbuffered serial mode*. In this mode the transmitter sends a synchronisation sequence to the receiver followed by a constant stream of data bits at 250kbps. For the setup described, upon receiving a Start of Frame Delimiter (SFD) the CC2420 forwarded the decoded bits directly to the MSP430 microcontroller (without buffering). The microcontroller then compared them to reference data and updated the BER for each corresponding RSSI. The RSSI value itself was read directly from the appropriate CC2420 register.

The MSP430 runs at a clock rate of 8MHz[4], while the CC2420 transceiver at a rate of 16MHz; this proved problematic for the setup described. The CC2420 transmits

[4]The clock frequency is determined by the presence or absence of the “rosc” resistor attached to pin 25 of the microcontroller - see [13].

at a data rate of 250kbps however due to the lower clock frequency the MSP430 was not fast enough to process the incoming bits received by the CC2420. To overcome this problem a frequency divider circuit was placed between the transceiver and the microcontroller. This allowed the MSP430 to collect every 32nd bit in the sequence without having to change the transmission rate (which is not actually possible with this particular transceiver). The divider itself was used to divide the synchronisation clock which occurs on the FIFOP pin of the CC2420.

With the frequency divider circuit in place, a random sequence of one million bits was generated at the beginning of each test. These bits were then transmitted to the receiver a number of times. The RSSI was sampled at the receiving node once every 32 bit periods and a histogram of the number of bits received at a particular RSSI obtained. Experiments were run at two different locations (in a corridor and a spacious laboratory) and about 10 million bits were collected for each RSSI value. An estimation of the noise floor was also obtained by sampling the RSSI register with no motes transmitting. This enabled a rough approximation of the SNR using the following equation:

$$\text{SNR} \approx 10\log_{10}\left(\frac{\text{RSSI} - P_n}{P_n}\right) \tag{3.7}$$

where RSSI and P_n represent the received signal strength and noise floor respectively, both of which are in Watts. Equation 3.7 is only an estimation as it neglects interference from other sources that may be included in the measured RSSI value.

The noise floor can be determined analytically using the equation:

$$P_n = FkT_0B \tag{3.8}$$

where F is the noise figure of the radio, T_0 is the ambient temperature and B is the noise equivalent bandwidth. The value of F for the CC2420 radio is about 11/12 dB and the bandwidth of the channel is approximately 3MHz [5] giving an approximate value for the noise floor as:

$$\begin{aligned} P_n(dB) &= 12 - 198.6 + 10\log(300) + 10\log(3 \times 10^6) \\ &\approx -97\text{dBm}. \end{aligned} \tag{3.9}$$

This value corresponded quite closely with the measured values which were between -94 and -98 dBm.

The results of the BER tests can be seen in figure 3.5. Included is a plot of equation 3.5 for comparison. The initial tests were performed in the corridor and bits were sent at 0 dBm so the nodes had to be placed about 20m apart in order to capture any errors. It is observed that the results obtained are considerably different from the analytical values given in the standard. A second test was performed in the laboratory where two nodes were placed on a table with no obstacles in the vicinity. The transmission power was set to -25 dBm and errors were observed at very close proximity (around 1m). As can be seen from figure 3.5 the BER curve obtained in this case is closer to the theoretical estimation although a large discrepancy still exists.

The differences in both cases may have been due to a number of factors. Firstly, the theoretical formula given in [5] may not be suitable for the CC2420 as mentioned. Also, the transceiver does not estimate the RSSI value for every bit but calculates an average over eight symbol periods. Even though the environment was relatively static the values read from the CC2420 register varied by about 1 or 2 dBm.

Finally the approximation of the SNR using equation 3.7 is almost certain to have an influence on the BER. All of these factors combined contribute to a difference between

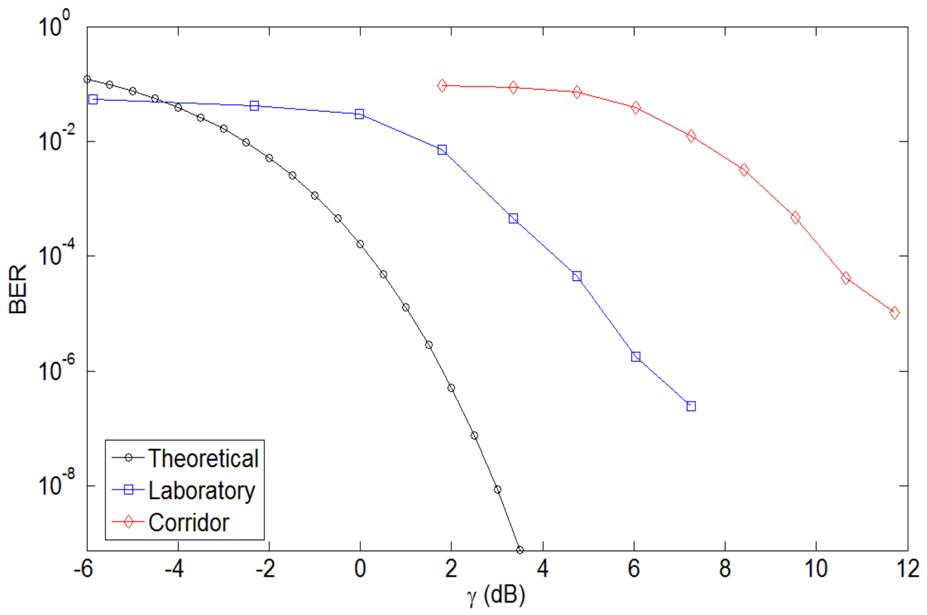


Figure 3.5: Plot of BER given in IEEE 802.15.4 standard [5] compared with the empirical measurements using the CC2420 transceiver.

results obtained using the p_e and the BER plots. As a side point, the results obtained would suggest that the RSSI may not be a very good estimator of the PRR because, as was observed, for the same RSSI value in two different environments the BER was quite different. This would seem to contradict the results in [111] however it is possible that under reduced levels of interference RSSI may in fact be quite a good estimator of the link quality.

3.4 Comparison of Results

For the experiments carried out in section 3.3.1 a frame size (f) of 39 bytes (28 bytes payload and 11 bytes header) was used. Using this value in equation 3.6 and the BER curves, the values of γ_U and γ_L can be determined. Table 3.2 shows these results along with those obtained for the theoretical BER equation in section 3.3.2. Interestingly, the difference between γ_U and γ_L is approximately the same in each case (also shown in

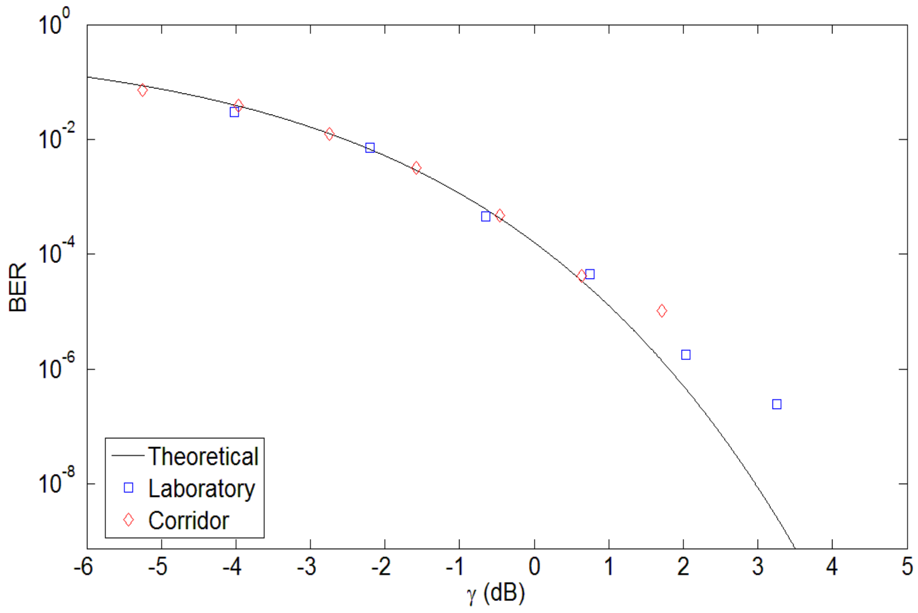


Figure 3.6: Plot of BER given in IEEE 802.15.4 standard [5] compared with the empirical measurements using the CC2420 transceiver shifted by constant offsets.

the table). Considering this and inspecting figure 3.5, it is clear that each of the curves have a very similar shape — only differing by a reasonably constant SNR offset. This is shown more clearly in figure 3.6 where the BER curves from the laboratory and corridor experiments are both plotted with an offset of 4 dB and 10 dB, respectively. The shape of all three curves can be seen to be quite similar. Each of the three curves would lead to very similar values for γ_U and γ_L in this case (i.e. approximately $-.3$ dB and -2.3 dB, respectively).

One possible explanation for this is again due to equation 3.7 which uses the RSSI to calculate the SNR. Although it is not stated in the CC2420 data sheet how the RSSI is determined, the value measured would consist of the power of the received signal plus noise plus interference. In the corridor there may have been increased levels of wireless interference possibly due to close-proximity to an access point. The SNR calculated would therefore have been an over-estimation of the actual SNR which would explain the curve moving towards higher SNR values in figure 3.5.

Plot Used	γ_U (dB)	γ_L (dB)	Γ(indoor)	Γ(outdoor)
Theoretical	-.3	-2.3	7.7	2.2
Laboratory	3.57	1.77	7.5	2.2
Corridor	9.7	7.7	7.7	2.2

Table 3.2: SNR values obtained using the three curves of figure 3.5.

Table 3.3 shows the overall results obtained for the transitional region. The values of Γ determined using the three BER curves of figure 3.5 show similar results in each environment. This is easily explained by referring to equation 3.3 and noting that the difference between γ_U and γ_L is approximately the same for each curve. As can be seen for both the indoor and outdoor environments the values of Γ agree quite well with the experimentally obtained values.

In some cases a discrepancy occurs, however, when comparing the calculated values of d_s and d_e with the experimental results. For example, the values obtained using p_e are in strong disagreement with the experimental results for the indoor environment and, as mentioned, to a lesser degree for the outdoor environment. This is to be expected however as only one of the curves should correspond to the true BER curve. It can be noticed that, for the indoor environment, the values obtained using the corridor BER agree quite closely with the experimental values. In the outdoor case the values obtained using the laboratory BER curve more closely agree, however. Again, this is possibly due to the fact that equation 3.7 is only an approximation. As RSSI values were used for both the analytical and experimental results it would seem plausible that similar environments would produce similar results. An interesting observation is to note that if the curve of figure 3.3 is shifted down by approximately 4 dB (the same difference between the theoretical and laboratory curves of figure 3.5) and the value of PL(d$_0$) recalculated as $49.7\text{dBm} + 4\text{dB} = 53.7\text{dBm}$, then the transitional region is best approximated in this case by the theoretical curve of figure 3.5 giving values of

$d_s = 3.6$, $d_e = 11.7$ and $\Gamma = 2.2$. This result is interesting because it suggests that if the level of interference can be ascertained for a particular environment, then γ_U and γ_L derived from the theoretical BER can be used to obtain the transitional region using RSSI values; this assumption would require more experimental evidence of course.

	Indoors			Outdoors		
	$d_s(m)$	$d_e(m)$	Γ	$d_s(m)$	$d_e(m)$	Γ
Theoretical	12.2	105.7	7.7	4.8	15.4	2.2
Laboratory	8.6	73.7	7.5	3.7	11.9	2.2
Corridor	5.0	43.6	7.7	2.5	8.1	2.2
Experimental Results	4.0	32.0	7.0	4.0	11.0	1.8
From [22]	6.9	25.9	2.8	3.2	8.6	1.7

Table 3.3: Transitional Region Parameters obtained using the different graphs. Experimental results are those obtained in section 3.3.1.

3.5 Conclusions

In this chapter preliminary work has been carried out in order to study the effects of using a spread-spectrum device on the transitional region around a wireless sensor node. It has been suggested in previous work that due to the transitional region being caused in part by multi-path, techniques used to combat multi-path may also be effective in reducing the extent of this region.

In the absence of access to non-spread spectrum devices, against which to compare performance within the same controlled environment, it is not possible for us to definitively answer the question of whether spread spectrum systems can reduce the size of the transitional region, it is clear that the region still has significant extent, however. This motivates the use of other methods to overcome the reduced packet reception rates. One such method is packet combining which will be discussed in the next chapter.

Chapter 4

Packet Merging

A number of methods were described in chapter 2 for counteracting the effects of fading in the wireless channel. One of the most popular techniques was shown to be diversity where the same data is sent over a number of independent fading channels and combined at the receiver. Many of the combining techniques used, such as maximal-ratio combining or the use of Space Time Block Codes (STBCs) [11], are not viable options on commercially available architectures such as the popular Tmote Sky [13], MicaZ [14], and other similar devices, due to their simplified hardware. In addition, Channel State Information (CSI) at both the transmitter and receiver is not available and only partial information may be available at the receiver (in the form of LQI and RSSI[1]).

In this chapter an improvement to a recently proposed *packet combining* procedure based upon the CRC is developed. Upon receiving two packets, sent over independent channels, a large error burst may appear in one; however, it is unlikely that the same situation will have occurred for its pair. By making an intelligent choice as to which packet the burst is most likely to be in, and choosing the other packet for correction, the decoding time can be reduced significantly. A practical implementation is provided with real results that show not only can this decision be made correctly on the majority of occasions but, once the correct packet is chosen, an intelligent search operation can be performed further reducing overhead. The particular design described here

[1]Link Quality Indicator (LQI) and Received Signal Strength Indicator (RSSI).

makes the technique developed useful in both burst error channels and non-burst error channels.

Although the experimental results in this chapter focus on a system consisting of multiple cooperating nodes (to create the independent channels) the procedure developed also works very effectively for situations where each packet originates from the same source. This situation is studied in more detail in chapter 5 along with an energy analysis of the system.

4.1 Related Work

Packet combining was first described by Sindhu [16] using forward error correction in bursty channels. An incremental redundancy combining technique called *code combining* was then developed by Chase [113] and was designed with very noisy channels in mind. Code combining concatenates the received packets to form codewords from increasingly longer and lower-rate codes. In [114] a scheme is described whereby the transmitter sends a packet to L different nodes within a cluster. Code combining is then used at the cluster head in order to correct any errors that may have arisen.

In [17] an ARQ scheme with packet combining is presented that is suited to the resource-constrained devices under consideration in this thesis. The particular scheme, referred to as Extended ARQ (EARQ), has a throughput for smaller packet sizes that is sufficiently tight to an upper bound for type-II ARQ schemes (up to a very high BER). The combining method in EARQ uses a modulo-2 sum of the received packets in order to assist in error correction (cf. section 4.2). In [18] an interesting protocol called Simple Packet Combining (SPaC) is described that makes use of this combining procedure.

The SPaC protocol is designed specifically for low-rate, low duty cycle sensor networks and has been shown to offer a lot of promise for these type of devices. When a node receives two or more corrupt packets, a packet combining procedure is attempted. Either a *packet merging* or *packet decoding* operation is performed: the latter can be thought of as a form of coded cooperation [100] when used in a cooperative sense or of type-II hybrid ARQ when used in a non-cooperative sense [9]. The packet decoding procedure uses a systematic, invertible block code and sends a parity packet if the first (non-parity) packet is received with errors. This allows for the use of redundancy only if required and is hence a form of incremental redundancy.

The packet merging part of the SPaC protocol is the same as the combining procedure presented in [17]; the name change is required to distinguish it from packet decoding. Packet merging is shown to be somewhat inferior to packet decoding: the reason being claimed to be the fact that packet merging is a repetition code. Packet merging is, in fact, not a repetition code as it makes use of the frame check sequence (used for error detection in a cyclic-redundancy check) to correct the packets. A repetition code uses a majority voting scheme and it is impossible to perform a decoding operation based on only two packets — exactly what packet merging does. However, with an improved packet merging scheme the SPaC protocol shows even greater potential as an error control technique for WSNs. The rest of this thesis develops a significantly improved packet merging protocol which can then be used to greatly enhance protocols such as SPaC or, if simplicity is a major requirement, to be used as a standalone device.

4.2 Merging Procedure

For packet merging, the destination accepts two packets of the same type and creates a merged error mask which will be referred to as an “Ambiguity Vector” (V_A):

$$m'_S \oplus m'_R = e_S \oplus e_R = V_A. \tag{4.1}$$

Here m'_S and m'_R are the two received, corrupt data vectors from the Source (S) and Relay (R) respectively, and e_S, e_R are, respectively, the error vectors corresponding to these packets. The ambiguity vector V_A shows in which bit positions the two packets differ. It will contain a 1 in the positions where an error has occurred in one or other of the data packets, and a 0 where either no error occurred or where one occurred in the same position in both packets. This latter case is not correctable using packet merging and is referred to as a *hidden error* (cf. section 4.7.4).

Once V_A is obtained, it is then used with one of the packets, m_{ch} (i.e. the chosen packet m'_S or m'_R), and the Frame Check Sequence (FCS) to try to obtain the original packet. This is done by modifying m_{ch} in each bit position where there is a potential error (as indicated by V_A), and checking whether the resulting packet matches the FCS.

In the SPaC protocol the chosen packet, m_{ch}, is randomly selected and an exhaustive search is used where every possible error candidate is tested. This leads to the following possibilities: (i) A unique error pattern yielding the correct FCS is obtained in which case a success is declared, (ii) Multiple candidate error patterns yield the correct FCS in which case a failure is declared, and (iii) none of the candidate patterns match the original error. This last event is only possible in the case of hidden errors. The purpose of testing every possible error candidate is to identify the second scenario, in which case it is impossible to tell which candidate is the correct one. As will be seen, a reduced search is possible leading to significant reduction in energy consumption or improved throughput performance.

4.3 Practical Setup

In order to gain a deeper understanding of the merging process a practical setup was devised consisting of a three node architecture: a Source (S), a Relay (R) and a Destination (D). The particular devices used were the popular Tmote Sky motes [13] which incorporate a 2.4GHz spread-spectrum radio compliant with the IEEE 802.15.4 standard [5]. Three different experiments were undertaken at three separate transmit power levels: -25 dBm, -15 dBm and 0 dBm (referred to as Ex1, Ex2 and Ex3 respectively). The main aim of each setup was to produce packets with large numbers of errors in order to test packet merging in these scenarios. To this end, the nodes were setup such that the interuser channel (i.e. from S to R) was good, but the uplink channels (i.e. from S to D, and R to D) were bordering on the sensitivity levels of the receiver.

A number of different environments were tested for the experimental setup. The chosen environment was a cluttered utility room as it was found to give the best results (i.e. the largest packet error rates). This room contained metal shelves, a number of toolboxes, a washing machine, two bicycles and other similar items. The source and relay nodes were placed close to one another within this room and were not moved throughout the entire experiment. The placement of the receiving node was varied for each transmission power: as the power levels were increased it was necessary to increase the separation distance between the receiving node and the source-relay pair. This was done in order to maintain a situation where the receiver operated at the fringe of its radio range (from both source and relay) and involved using node separations from 1 to 30 meters. At the smaller distances the communication would have been dominated by a large LOS component while at large separations (involving the receiver being placed in separate rooms), there would have been no LOS but increased levels of multi-path (due to changes in the surrounding environment).

The number of packets sent for each experiment was 500,000. The data sent was test data and was therefore available to the receiver for comparative purposes. This allowed, for example, an explicit evaluation of e_S and e_R which normally would not be possible. The size of the transmitted packets was dependent upon whether the source or relay was transmitting. As the entire source packet (header + payload +

FCS) was retransmitted by the relay, the relay's packet was necessarily 13 bytes longer than the transmitters (relay's header + entire source packet + relay's FCS). For these experiments, therefore, the source packet contained 27 bytes (216 bits) in total whereas the relay's packet contained 40 bytes (320 bits). The combining operation was carried out over the full 27 bytes of the source packet (and the 27 byte payload of the relay packet). The experiments described in chapter 5 further considers *source* packet sizes of 40 bytes (320 bits). Although the source header was included in the combining operation (for completeness), whether or not this is possible, or necessary, will depend upon the particular network topology, the protocols used, and the particular layer of the protocol stack combining is attempted.

The protocol used was essentially the simple Adaptive Decode-and-Forward protocol (simple AdDF) described in [101] (with some slight modifications): the source node broadcast its message (m_S) to R and D in phase one. The relay having decoded each received message (m_R), only forwarded it in phase two if the CRC check passed. Upon reception, the destination carried out another CRC check on the received versions of m_S and m_R. If either passed then no further action was taken; however, if both failed then the error correction algorithm was only applied if the following was true:

$$(m'_S \oplus m'_R \neq 0) \text{ AND } (\text{FCS}_S \oplus \text{FCS}_R = 0), \tag{4.2}$$

where FCS_S and FCS_R represent the frame check sequence of the source and relay messages respectively. In the basic scheme described in this section[2] FCS_S = FCS_R and will therefore be referred to simply as FCS. Corrections were not attempted either if the Hamming weight (denoted by $w(.)$) of the ambiguity vector was greater than 10; i.e. $w(V_A)_{max} = 10$. The size of $w(V_A)_{max}$ is influenced by the computational cost of the merging operation and the percentage of hidden errors that might occur. The value

[2]Section 4.7.2 enhances the protocol by using two FCS polynomials as will be seen.

of 10 was chosen in this case as the experimental results showed a similar number of packets (approximately 2.2%) containing hidden errors for each $w(V_A)$ ranging from 1 through 10. For larger values of $w(V_A)_{max}$ the number of hidden errors began to increase slightly up to a value of about 15 - beyond which there was a significant increase. Also, as the computational complexities of the algorithms developed in this thesis are less than that used in SPaC (where $w(V_A)_{max} = 6$) it is possible to allow for a larger value of $w(V_A)_{max}$. In some circumstances values of $w(V_A)_{max}$ larger than 10 may be possible; this will be discussed further in chapter 5.

4.3.1 Search algorithms

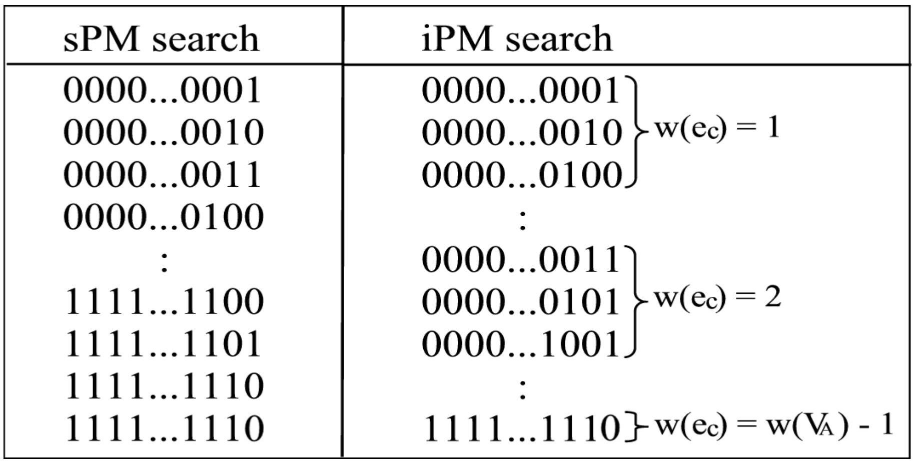
Two algorithms were developed for the above scheme: the second resulting from the first. The first algorithm, which will be referred to as Simple Packet Merging (sPM), chooses m_{ch}, *a priori*, and adheres to that choice throughout the experiment. As there are a total of $2^{w(V_A)} - 2$ possible error candidates (see [18]) the algorithm runs through each possible candidate (e_c) from 1 to $2^{w(V_A)} - 2$ in unit increments (referred to as an "incremental" search; cf. figure 4.1) modifying m_{ch} and determining whether the following is true:

$$\text{CRC}(m_{ch} \oplus e_c) \oplus \text{FCS} = 0, \tag{4.3}$$

where CRC(x) calculates the FCS for the value x. Once equation 4.3 is satisfied the algorithm returns the value found[3].

It makes more sense, however, to step through the error candidates in a more structured manner. The second algorithm, which will be referred to as Improved Packet Merging (iPM), attempts to do this. Instead of an incremental search, as was done for sPM, it first tests all error candidates with a Hamming weight of one, then all those with a

[3]The justification for this will be provided throughout the rest of this chapter.



sPM search	iPM search
0000...0001	0000...0001
0000...0010	0000...0010 } $w(e_c) = 1$
0000...0011	0000...0100
0000...0100	:
:	0000...0011
1111...1100	0000...0101 } $w(e_c) = 2$
1111...1101	0000...1001
1111...1110	:
1111...1110	1111...1110 } $w(e_c) = w(V_A) - 1$

Figure 4.1: Comparison of the search methods for sPM and iPM.

Hamming weight of two and so on until either a packet is found which produces the correct FCS or all candidates have been exhausted (cf. figure 4.1). The motivation for using this particular search method (which will be referred to as an "improved" search) can be seen from the graph of figure 4.2 (showing actual data). While this represents an extreme example, it shows that it is possible for a large error burst to be contained in one packet only. It can be seen from the figure that m'_R contains many more errors than m'_S. Although $w(e_R)$ has a value of 27, $w(e_S)$ is only 3 — a significant difference. Therefore, by choosing the packet with the smaller number of bits and using the improved search method a significant energy saving can be made.

4.3.2 Packet Selection

Figure 4.2 clearly shows that in some cases one of the packets may contain many more errors than its pair. While this is an extreme case, the experimental results showed that, on average, the errors were not equally distributed between the two packets. To see this, consider figure 4.3 (obtained from the results of experiment 3). This figure consists of a plot of the average difference in Hamming weights between the source and relay error

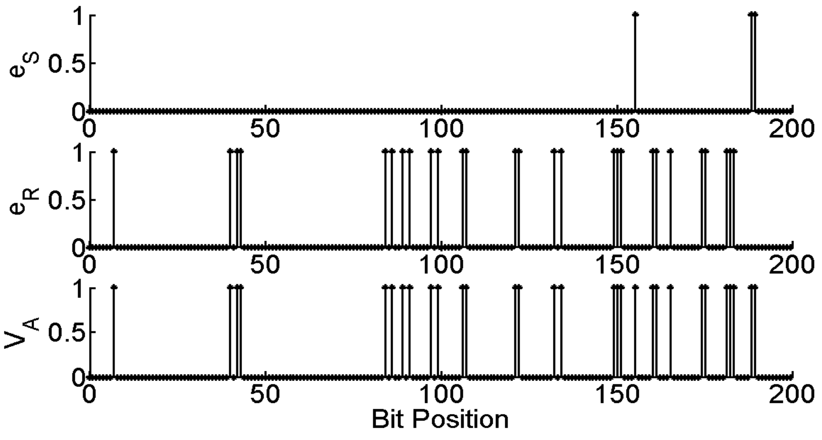


Figure 4.2: Plot showing practically obtained error vectors (e_S and e_R) and ambiguity vector (V_A) obtained from experimental setup 3.

vectors (E$[|w(e_S) - w(e_R)|]$)[4] against the Hamming weight of the ambiguity vector ($w(V_A)$). Had an equal number of errors been contained in each packet, for each value of $w(V_A)$, then E$[|w(e_S) - w(e_R)|]$) would have remained at zero and the graph would simply have been a horizontal line. If, however, all the errors had been contained in only one of the packets then the graph would have appeared with a slope of 1. As can be seen, the graph in figure 4.3 has a slope of approximately 0.5 implying that, on average, one packet contained 3 times as many errors as its pair. The reason for this is likely due to the independent channels the data were sent over[5] implying that a deep fade corrupting one packet is unlikely to occur for its pair. Similar graphs were seen for the other setups.

The second innovation of iPM attempts to correctly choose the packet with the least number of errors and use it as m_{ch} (cf. equation 4.3). The capabilities of the physical devices being used will obviously have an influence on this selection policy. Although the current setup was carried out on the Tmote Sky mote it can easily be extended to

[4]For notational clarity E[.] is used to represent the average value. It therefore does not represent the expected value in this context.

[5]Due to the orthogonality constraint, these channels are separated in both time and space.

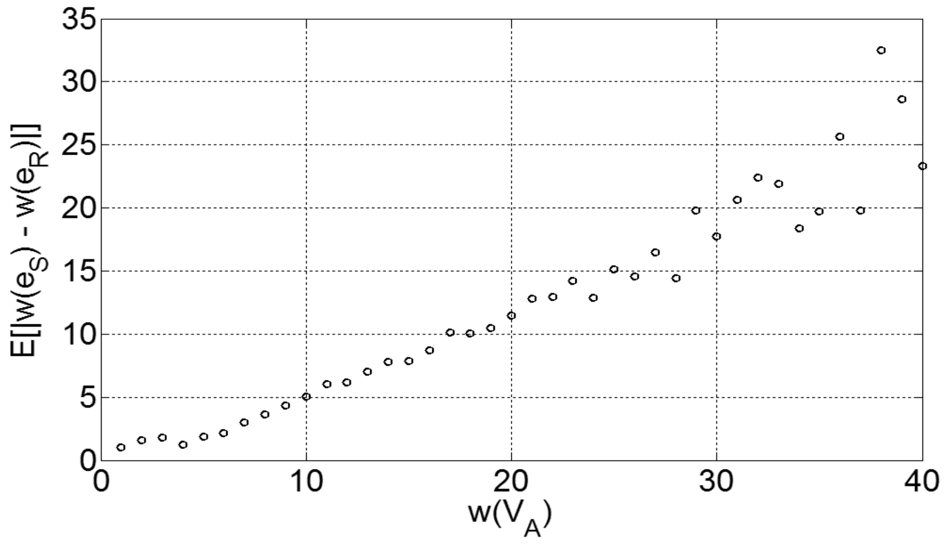


Figure 4.3: Average difference in Hamming weights between the source error vector (e_S) and the relay error vector (e_R) compared with the Hamming weight of the ambiguity vector ($w(V_A)$).

many other similar devices (such as those supported by the TinyOS operating system [115]).

With the Tmote Sky mote (and others of its class) there are two obvious methods for choosing the best packet: the Received Signal Strength Indicator (RSSI) or the correlation value (CORR)[6]. Both of these values are returned by the CC2420 (the transceiver used on the Tmote) for each incoming packet and are averages over the first eight symbol periods after the Start of Frame Delimiter (SFD).

The use of one of these metrics to determine the best packet is similar to the idea of selection combining, where the signal with the largest instantaneous power is chosen for processing. In general, the main drawback of selection combining is that the SNR on each branch has to be monitored separately leading to either an increase in hardware complexity or reduced spectral efficiency. In the cooperative relaying situation the

[6]Although not the same, CORR will henceforth be referred to as LQI as it is more common; the former can be used as an estimation of the latter.

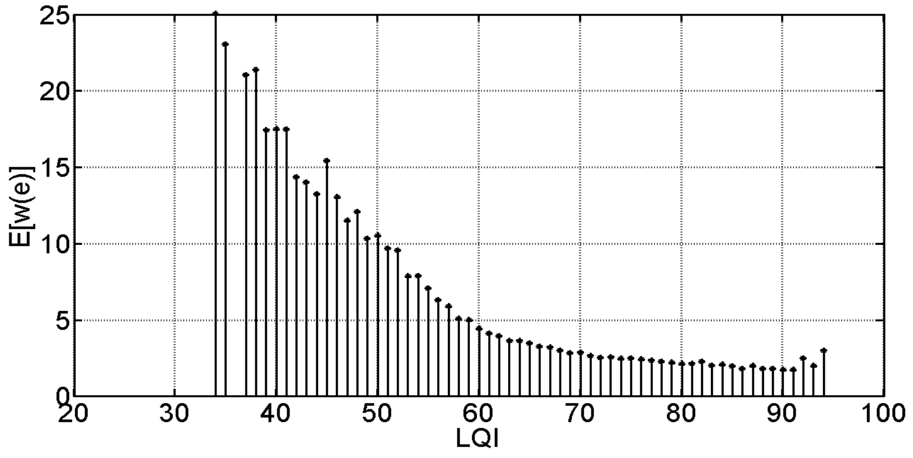


Figure 4.4: Example of how the Hamming weight of the error vector changes with LQI (taken over a static link with a mean of about 67).

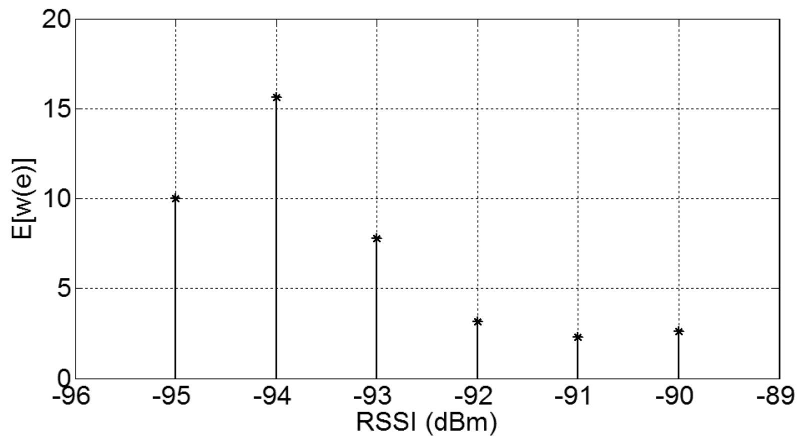


Figure 4.5: Example of how the Hamming weight of the error vector changes with RSSI (taken over a static link with a mean of about -92 dBm).

relay channel has to comply with the orthogonality constraint, and this is therefore not an issue. Of course, the advantage of the scheme described in this thesis is that it doesn't discard the other "branch" (as selection combining does) but makes use of it to create the ambiguity vector.

Figure 4.4 (resp. 4.5) shows the relationship between the Hamming weight of an error vector and its LQI (resp. RSSI) value; both of these graphs were taken from the experimental results. At first it would appear that the LQI is a much better packet selection metric than the RSSI; however, these are plotted against the average values of the Hamming weight of the error vectors (E[w(e)]) and the distributions are in fact quite different. The RSSI value was quite consistent throughout the entire experiment with a mean of about -92 dBm and a standard deviation of only 0.63 dB;

It can be seen that the decrease in the number of errors as the LQI increases, on average, is quite consistent; however, the RSSI value doesn't produce such fine grained results. Interestingly, a number of RSSI values above -90 dBm (not shown) were obtained where the number of errors jumped dramatically. This apparent contradiction is likely due to in-band interference increasing the RSSI value and causing a weak link to appear to be of good quality; similar effects were noted in chapter 3.

Considering these results, whether or not to use RSSI for the selection of the appropriate packet will, in general, be determined by the particular environment. If noise is the main problem with the system then RSSI driven diversity could be used (perhaps in conjunction with CORR). However, in environments with large levels of co-channel interference the RSSI can overestimate the SNR and may need to be avoided. However, if the level of interference is constant over the course of two packets then this will not be a problem as it is a relative, and not absolute, value that is of interest for this particular scheme.

	Ex1 (%)	Ex2 (%)	Ex3 (%)	Average (%)
LQI	61	73	79	71
RSSI	53	69	81	68

Table 4.1: The percentage of chosen packets, using either LQI or RSSI, containing the lesser number of errors.

Over the course of quite a number of experiments (including others not described in this thesis), the LQI did in fact prove to be a slightly better choice for determining m_{ch}, but not by much. An example of this can be seen in table 4.1 where, averaging over the three experiments, on 71% of occasions the correct packet was chosen (i.e. that with the smallest Hamming weight). This can be compared to the RSSI value (also shown) which averaged at about a 68% success rate — still quite accurate. The closeness of these results implies that in some cases the RSSI may prove a better metric, and indeed this can be seen to be the case for experiment 3. One possible way of choosing between the use of RSSI and LQI in real-time may be to initially alternate between both metrics for each successive merging operation. The number of iterations required compared with a complete search can then be determined for each. The metric leading to the least number of iterations on average would then be chosen.

For the experiments described in this chapter it was simply decided to use the LQI for packet selection due to its higher success rate in general. Future chip designs may provide a means of sampling the RSSI/LQI for each symbol of the incoming packet. It is very likely that this would significantly improve the selection policy as the current algorithm relies on one sample only (an average RSSI value taken over the first 8 symbol periods of the packet) to provide accurate information about the rest of the packet.

4.4 Experimental Results

The two algorithms described in section 4.3.1 (sPM and iPM) were used in each experiment. Recall that the search algorithm used by the first was the simple incremental search. The selection criterion was to choose either m'_S or m'_R at the beginning of each experiment and adhere to this choice throughout. Both options were evaluated for each experiment and will be referred to as Case 1 and Case 2, respectively. For iPM the LQI was used to choose, in real time, the best packet to use as m_{ch}. The search algorithm used was the improved one described in section 4.3.1. In summary:

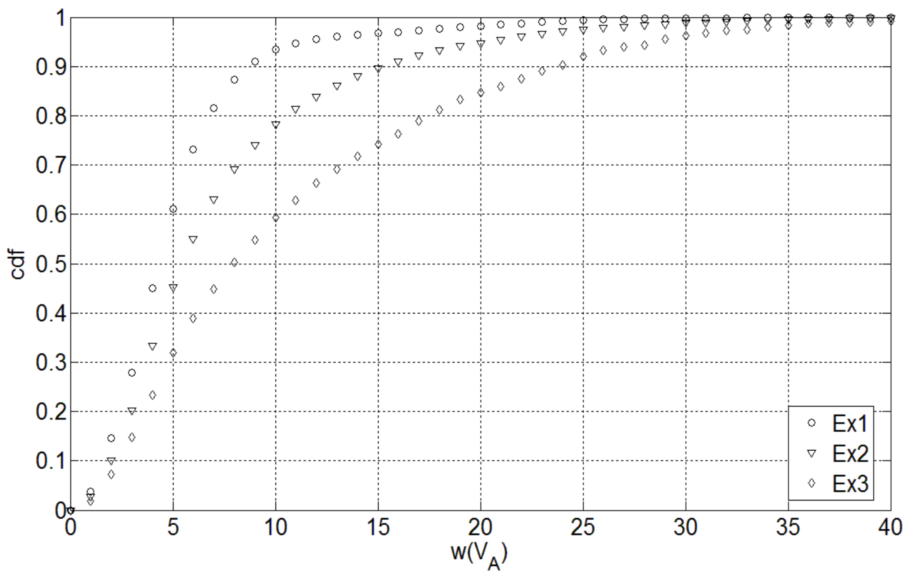


Figure 4.6: Empirical cumulative distribution of $w(V_A)$.

1. sPM
   (a) Selection Criteria:
   - Case 1: Always choose m'_S
   - Case 2: Always choose m'_R

   (b) Search Method: Incremental
2. iPM:
   (a) Selection Criteria:
   - If $\mathrm{M}_{m'_R} > \mathrm{M}_{m'_S}$, choose m'_R; else choose m'_S

   (b) Search Method: Improved

where $\mathrm{M}_{m'_x}$, $x \in \{S, R\}$, is the chosen metric RSSI, LQI or some other available quantity.

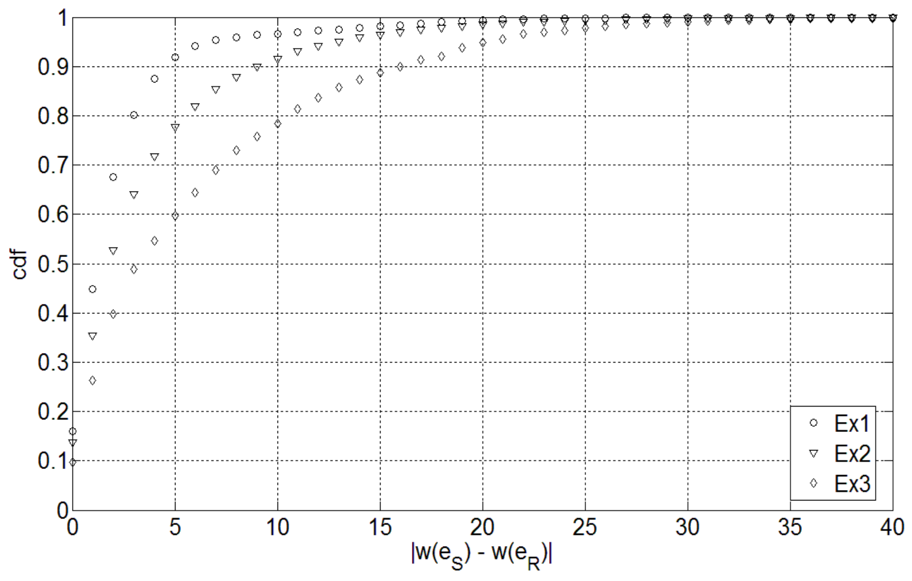


Figure 4.7: Empirical cumulative distribution of the difference in Hamming weights between m'_S and m'_R for each experiment.

Figure 4.6 shows the empirical cumulative distribution (marked as cdf for short) of the Hamming weight of the ambiguity vectors received. It can be seen that there is a clear increase in reception of larger weights from experiments 1 to 3 possibly due to the increased multipath component (cf. section 4.3). In the first and second experiments, respectively, 93% and 78% of the ambiguity vectors had a Hamming weight less than $w(V_A)_{max} = 10$, while for the third case this was reduced to 59%.

Consider now figure 4.7. It can be seen that the difference between the error vectors gets increasingly large for each experiment. For the third experiment, 40% of the vectors are separated by a difference greater than 5, whereas this value is only 22% and 8% for experiments 2 and 1, respectively. By using iPM, it should be possible to cope more efficiently with the larger $w(V_A)$.

Table 4.2 shows a comparison of the results between the sPM and iPM for the three different setups. These values represent the number of iterations the algorithms had to go through before finding the correct packet (expressed as a percentage of the total

	sPM (%)			iPM (%)	Difference (%)
	Case 1	Case 2	Average		
Ex1	57	44	51	47	4
Ex2	57	46	52	36	16
Ex3	42	65	54	33	21

Table 4.2: Comparison of the percentage of required iterations for each setup.

possible number of iterations). For experiment 1, it can seen that sPM, in fact, performed slightly better than iPM when constantly choosing m'_R (case 2). The problem in this case is most likely the fact that the LQI only achieved a 61% success rate (cf. table 4.1) which itself was most likely due to the very similar channels from S to D and R to D.

In a realistic scenario, however, it may not be possible to choose, *a priori*, which is always going to be the best packet and a real time evaluation is needed; iPM does exactly this and it is seen that, even with only a 61% correct-choice success-rate, it comes closer to the better of the two cases for sPM. On the other hand, experiments 2 and 3 exemplify the benefits of iPM even more so due to the increase in $|w(e_S) - w(e_R)|$. For experiment 2, iPM improves upon the best sPM case (which there is no way of guaranteeing in practice) by 10% and for experiment 3 by 9%. In the latter case, a further improvement would have been made had the RSSI been used in choosing the optimal packet (cf. table 4.1).

Due to the unrealistic selection procedure for sPM, a more reasonable comparison might be to compare iPM with the average of the two cases for each experiment. This is also shown in table 4.2 and it can be seen that there is a definite improvement in each case. As much as a 21% improvement in the number of iterations required can be seen. When compared to the scheme used in SPaC (which requires a full search) the reduction in the number of iterations is found, from table 4.2, to be between about

50-70%. Later experiments showed iPM requiring, in many cases, *less than* 30% of the total possible number of iterations.

4.5 Parity Equivalence

The computational complexity of both sPM and iPM can be improved further by noting that, for generator polynomials with an even Hamming weight the parity of the message will be the same as that of the FCS. Using this fact, the number of required iterations, and hence the search time, can be reduced in half. The reason both message and FCS have the same parity can be understood as follows.

In section 2.4.4.3 the procedure for calculating a cyclic code in systematic form was described. The first two steps of this procedure are used for generating the FCS also. As a result, the FCS is simply a repeated modulo-2 sum of shifted versions of the generator polynomial with the message. It can therefore be said that

$$w(ps_{i+1}) = w(ps_i) + w(G) - 2n, \tag{4.4}$$

where ps_i is the partial sum at the i^{th} stage of the FCS calculation and n is the number of ones in the same bit position for both ps_i and G. As $2n$ will always be even and assuming $w(G)$ is even, then $w(ps_{i+1})$ will be even if $w(ps_i)$ is even, and odd if $w(ps_i)$ is odd. The parity of the FCS will therefore always equal that of the message for an even parity generator polynomial. To view this another way recall from chapter 2 that standard generator polynomials used for error detection have the following form:

$$g(x) = (x+1)p(x), \tag{4.5}$$

where p(x) is known as a primitive polynomial. The hamming weight of the message vector will always equal that of the FCS as

$$\begin{aligned} c(x) &= x^r m(x) + r(x) = q(x)g(x) \\ \Rightarrow c(1) &= q(1)(1+1)p(1) = 0. \end{aligned} \tag{4.6}$$

where $c(x)$ is the codeword polynomial (i.e. the message concatenated with the FCS), $m(x)$ is the message polynomial and $r(x)$ is the remainder polynomial (i.e. the FCS). The codeword polynomial, therefore, always contains an even number of terms, as has already been noted in section 2.4.5. Both the message and FCS polynomial can now be seen to have the same FCS as follows:

$$\begin{aligned} 1^r m(1) + r(1) &= c(1) \\ \Rightarrow 1^r m(1) + r(1) &= 0 \\ \Rightarrow m(1) &= r(1) \end{aligned} \tag{4.7}$$

which is effectively the same as saying that the parity of the message is equal to that of the FCS. Error candidates that produce messages with the opposite parity can therefore be eliminated.

To test the effect of implementing this technique in iPM the experiments were run again with this method in place. The reduction in the number of iterations over a full search was found now to increase to approximately 85%. This is intuitively satisfying as without the parity equivalence the value was approximately 70%. A further reduction of $(100 - 70)/2 = 15\%$ thus leads to the value of 85% seen in the experiments.

4.6 Incremental CRC

Despite iPM's significant improvements over a complete search it is still desired to reduce the computational complexity of the packet merging algorithm further. An incremental CRC technique is described in [18]. This technique is outlined in this section as it is essential for the efficient operation of the merging procedure.

As the set of bits that change between each candidate packet is small (at most $w(V_A)_{max}$) it is inefficient to recalculate the FCS over the entire packet each time a new bit position is modified. It is possible to use the properties of modular arithmetic to reduce this calculation to one operation per changed bit. To see how this works note that the calculation of the FCS for each candidate error pattern is as follows:

$$r_c(x) = x^r(m_{ch}(x) + e_c(x)) \bmod G(x) \tag{4.8}$$

where $m_{ch}(x)$ is the packet chosen for correction and $e_c(x)$ is the candidate error. This can be split up into the following:

$$r_c(x) = x^r m_{ch}(x) \bmod G(x) + x^r e_c(x) \bmod G(x) \tag{4.9}$$

where the first term on the right hand side of the equation is fixed throughout the entire search and the second term is variable. The error candidate e_c can be written as the sum of single nonzero coefficient error polynomials. Along with the fact that the modulus operation is linear:

$$\begin{aligned} x^r e_c(x) \bmod G(x) &= \sum_{i=0}^{L} b_i x^r E_i(x) \bmod G(x) \\ &= \sum_{i=0}^{L} b_i x^{(r+i)} \bmod G(x) \end{aligned} \tag{4.10}$$

where $E_i(x) = x^i$ and b_i is a 1 if there is an error in position i and a zero otherwise. A precomputed lookup table can then be used to store the appropriate values: $T[i] = x^{(r+i)} \bmod G(x)$. This technique offers a significant reduction in computation time for the packet merging scheme.

4.7 Packet Merging Issues

As with all error correction schemes there is a limit to the amount of errors that can be corrected. This section looks at the concepts of *false-positives* and *hidden errors*. While a method is described for dealing with false positives, the concept of hidden errors is only briefly touched upon here but is more thoroughly dealt with in [17] and [18].

4.7.1 False Positives

Undetected errors were discussed in section 2.4.5.2. They occur when the extent of the errors introduced by the channel are such that the received packet is equal to a valid codeword different from the original. This is an inherent problem with any error correction scheme as has been shown. The packet merging scheme described in this work makes use of the CRC in a different way than it is normally used. It is possible that the search for the appropriate error candidate produces a valid codeword different from the original. Rather than viewing this as an undetected error it is more accurate to refer to it as a false-positive, although the idea is the same. An understanding of the extent of the occurrence of false-positives using iPM is important. In SPaC all possible error candidates are considered, even after a candidate has been discovered that is a valid codeword. The only reason for doing this is to detect the presence of false-positives. This consumes a significant amount of energy over the lifetime of a node however.

In iPM the algorithm makes the assumption that the first packet matching the FCS is the correct one. It is therefore of interest to determine the proportion of false-positives that iPM produces. In each of the experiments described earlier the percentage of correctly decoded packets after merging was on average about 99.94%. The other 0.06% uncorrectable packets were due to false positives. This may seem like a rather large value however two points should be noted. Firstly, it was obtained under adverse conditions. Secondly, the probability of an actual merging occurring in the first place was not taken into account. When considered against the total number of transmissions sent the fraction of false-positives was, in fact, only 2.8×10^{-5}. Again, this value is representative of a worst case scenario and is on the order of the upper bound given by equation 2.22 (i.e. 2^{-16}). To use iPM in its basic form there is hence a trade-off between an 85% reduction in the number of iterations required over a brute-force merging operation and a small probability of producing an unrecognised false-positive[7].

The two main techniques used in iPM play an important role in keeping the number of false-positives at a minimum. The effect of the selection policy means that the packet with the least number of errors is chosen on average. This, coupled with the improved search algorithm, minimises the number of extra errors introduced into the packet in the early stages of the search. With the search method used in sPM (and possibly SPaC), the number of bit changes jumps between large and small values, increasing

[7]As is a problem with the CRC in general, the number of false-positives will tend to increase with packet size and this should be kept in mind during system design.

the likelihood of false-positives in the early stages of the algorithm.

4.7.2 Two Generator Polynomials

The number of false-positives is relatively small and for most cases will not be an issue. However, in more critical situations it might be desired to reduce them further. Although not available in current hardware designs (of the type considered in this thesis), future designs may provide an option for sampling the RSSI/LQI value for each symbol of the packet (as opposed to relying only on the sample taken at the beginning). It is very likely that this will not only increase the likelihood of choosing the packet with the least number of errors but also, if the bits are then arranged in order of decreasing confidence, reduce the number of false-positives and increase the search efficiency. Until this option becomes available however, another very effective method is to use a different FCS for the original packet and its retransmitted version. This is achieved using a second generator polynomial for calculating the retransmitted packet's FCS. The use of a second generator polynomial requires a negligible increase in memory space and complexity.

Considering now the use of another generator polynomial the first question to ask is this: can false-positives still occur when using two generator polynomials? To answer this question consider a message polynomial $m(x)$ of degree k. The relationship between the FCS (or $r(x)$), $m(x)$ and $g(x)$ is given by

$$\begin{aligned} x^{n-k}m(x) &= q_1(x)g_1(x) + r_1(x) \\ x^{n-k}m(x) &= q_2(x)g_2(x) + r_2(x) \end{aligned}$$

depending on whether a polynomial $g_1(x)$ or $g_2(x)$ is used. As before, a false-positive occurs in either case when another message $m'(x) \neq m(x)$ produces the same remainder:

$$\begin{aligned} x^{n-k}m'(x) &= q_3(x)g_1(x) + r_1(x) \\ x^{n-k}m'(x) &= q_4(x)g_2(x) + r_2(x). \end{aligned}$$

Combining these equations therefore gives the following:

$$\begin{aligned} x^{n-k}(m(x) + m'(x)) &= q_5(x)g_1(x) \\ x^{n-k}(m(x) + m'(x)) &= q_6(x)g_2(x). \end{aligned} \tag{4.11}$$

The set of polynomials $x^{n-k}m_i(x)$ that produce the same remainder for a given generator polynomial is known as a *congruence-class* (denoted here by $[r_g(x)]$). Equation 4.11 states that the sum of two polynomials from the same congruence-class produce a polynomial from $[r_g(x) = 0]$. Determining whether or not a false-positive can still occur given *two* FCS's is then the same as asking whether any of the polynomials from $[r_{g_1}(x) = 0]$ equals any from $[r_{g_2}(x) = 0]$. As each of these congruence-classes is a multiple of $g_1(x)$ and $g_2(x)$ respectively, a value will appear in both sets only if the following equality is satisfied (cf. equation 2.13):

$$a(x)g_1(x) = b(x)g_2(x) \tag{4.12}$$

where $a(x)$ and $b(x)$ are polynomials of degree k. This will be the case if either $a(x)$ or $b(x)$ is a multiple of $g_2(x)$ or $g_1(x)$ respectively. It is possible to find polynomials of degree $2k - n$ which satisfy this relationship and false positives are therefore still possible. However, due to the need for the candidate packets to satisfy both FCS values the probability of false-positives occurring is significantly reduced. To show that this is indeed the case an experiment was setup to determine the number of errors obtained in practice. Before discussing the results, however, the particular choice of second polynomial used is important.

4.7.3 Choice of second polynomial

While it might be tempting to choose a standard polynomial (such as CRC-ANSI) as the choice of second polynomial in general they do not produce the best[8] polynomials. In fact, they perform rather poorly when compared to the best polynomials [76]. Better polynomials with 16-bit redundancy were found in [73], for example. In [76], Baicheva *et al.* investigate *all* polynomials of degree 16 that are suitable to be used as generator polynomials of a cyclic code. It is from this source that a second polynomial was chosen for use with the packet merging scheme.

Noting the fact that a given generator polynomial does not produce codes that perform equally well over all packet sizes, it was decided to consider a source packet size of 320 bits — a typical size for mote class devices [115]. The polynomial chosen for the packet merging scheme was therefore 0x1A801 or

$$g_2(x) = x^{16} + x^{15} + x^{13} + x^{11} + 1. \tag{4.13}$$

This polynomial was shown in [76] to be best for $310 \le n \le 325$ with a minimum distance of 4 from $105 \le n \le 683$. For different applications it may be necessary to choose a different polynomial. However, it would be straight forward to set up a table of polynomials and switch between each of these polynomials for different

[8]"Best" in this sense means with the lowest probability of undetected error.

size packets. The receiver would have a copy of this table and use the length of the incoming packet to determine which second polynomial was used.

To determine the effectiveness of using two generator polynomials a similar experiment to that described in section 4.3 was carried out. This time, however, the aggregate number of packets over the course of 12 different experiments was used, leading to a much larger number than used previously. For each experiment the receiver was placed in varying positions both within the same room and in adjacent rooms. In each case the receiver was placed on the fringe of its radio range (from both source and relay) in order to introduce as many errors as possible. For half of the tests it was placed on a record player which was set to a revolution speed of $33\frac{1}{3}$rpm. The purpose of doing this was to introduce as much multipath as possible. More will be said about the effectiveness of this setup in chapter 5.

Over the course of the 12 experiments approximately 5 million packets were sent. Of these, 216,000 combining operations were attempted. To give an example of the number of errors in each packet, the empirical cumulative distribution of the Hamming weight of the ambiguity vector for one of the experiments is shown in figure 4.8. It can be seen that almost 90% of the obtained ambiguity vectors had a weight greater than 4, 70% greater than 5, 50% greater than 6, and over 40% larger than 7 — similar results were obtained for the other experiments. Recall that the minimum distance of the codes generated by each of the polynomials is 4. This meant that almost 90% of the combining operations had the potential to produce a false-positives, the probability increasing with larger values of $w(V_A)$.

As in the previous set of experiments, the fraction of false-positives obtained was approximately 2.8×10^{-5}. However, this was without considering use of the second generator polynomial. Considering this, the number of false positives was reduced to 0. While this does not explicitly state what the probability of a false-positive occur-

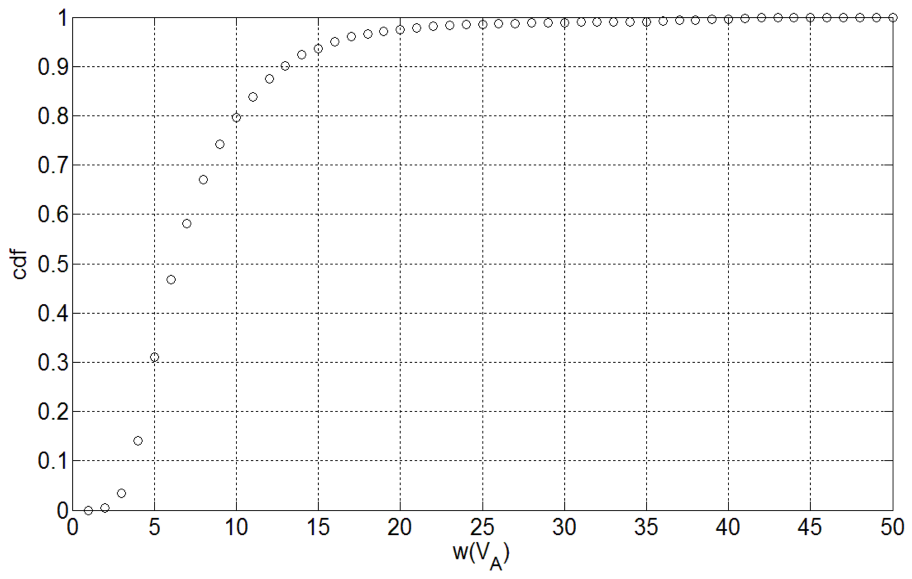


Figure 4.8: Empirical cumulative distribution of $w(V_A)$ for one set of results.

ring is using a second generator polynomial (it is still non-zero as described above), it shows that it is less than one in 5 million. It is therefore *at least* an order of magnitude better than the upper bound given in equation 2.22 which is an entirely satisfying result.

4.7.4 Hidden Errors and Uncorrectable packets

As mentioned previously, the presence of hidden errors is an issue for packet merging. These occur when the same bit is in error in both packets used for combining (i.e. m'_S and m'_R) and therefore cancel each other out. Hidden errors are not correctable by packet merging in its basic form and in fact cause the algorithm to run through all possible iterations before determining an inability to decode. This situation is clearly wasteful of resources and some method of at least detecting their presence is desirable.

Figure 4.9 shows the percentage of ambiguity vectors with hidden errors for a particular weight obtained in experiment 3 of section 4.3. For $w(V_A) \leq 10$, the number of

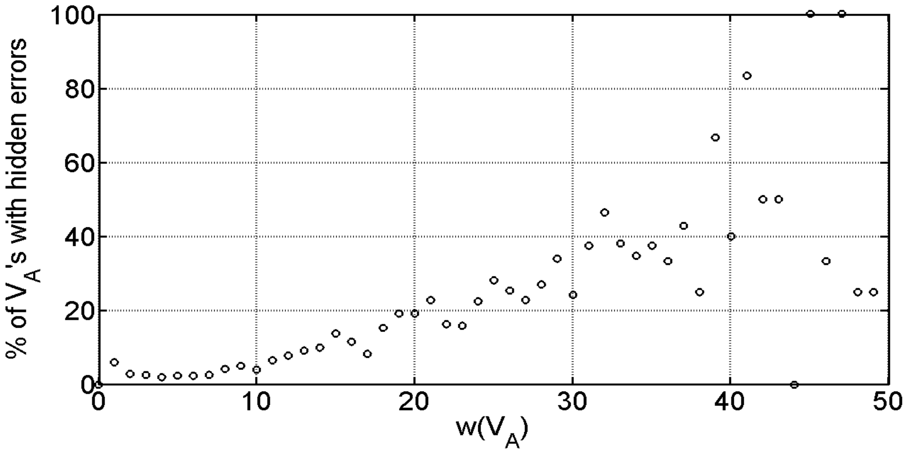


Figure 4.9: For each Hamming weight, this graph shows the fraction of ambiguity vectors that contained hidden errors at a particular weight, out of the total at that weight.

hidden errors that occur is, on average, about 2%. For $w(V_A) \leq 15$ this increases to an average of about 3.9%. It can be seen therefore that for this particular scenario, 10 seems to be a good choice for $w(V_A)_{max}$ although with further improvements to the merging operation a larger value may also be feasible. The greatly improved efficiency of iPM permits larger values of $w(V_A)_{max}$ than would be possible with a brute-force search method; this is a welcome improvement as the hidden error problem and false positive problem (cf. section 4.7.1) were shown to be less of an issue in choosing this value [18]. It will be shown in chapter 5 that the SNR gains possible with packet merging are a function of the value of $w(V_A)_{max}$ with larger values leading to a larger gain.

Table 4.3 shows the average number of hidden errors (e_h) found as a percentage of a) the number of ambiguity vectors with $w(V_A) \leq 10$, and b) the total number of ambiguity vectors. For all three experiments the number of hidden errors with $w(V_A) \leq 10$ was only about 2.2% as was expected from figure 4.9.

	Ex1	Ex2	Ex3
e_h (% of $w(V_A) \leq 10$)	2.1	2.3	2.2
e_h (% of total)	2.8	5.5	9.8

Table 4.3: Number of hidden errors (e_h) found.

4.8 Summary and Conclusions

An improved type of packet merging that is practical for use on simple, low-power wireless sensor nodes has been developed in this chapter. Two algorithms were implemented both of which use a pair of packets received over independent channels for error correction. The first algorithm, referred to as Simple Packet Merging (sPM), chooses one channel, *a priori*, and always uses the packets from that channel in the error correction process. It also implements a simplistic search procedure for the correct packet by running through all possible error candidates in an incremental fashion. The second algorithm, referred to as Improved Packet Merging (iPM), not only intelligently chooses, in real time, which packet has less errors, but also does a more efficient search for the correct packet. It was shown that iPM improves upon the average number of iterations required by sPM by up to 21%. This value was then reduced further by using the parity equivalence technique. Experiments showed that using the better of the two algorithms, iPM, a reduction of more than 85% in the required number of iterations compared with a brute-force search is obtainable. The number of false positives occurring was shown to be quite small when using iPM in its basic form. More critical situations, however, can practically reduce this value to 0 by using a second generator polynomial if so desired.

The next chapter takes a look at the effects of using packet merging in a cooperative and non-cooperative ARQ scenario. It is compared with maximal-ratio combining and is shown to offer quite a significant SNR gain despite its simplicity. The methods used to show this are a combination of simulation techniques and practical experiments.

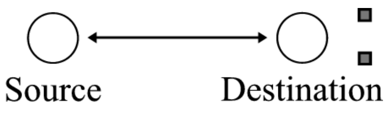
Chapter 5

Performance Analysis

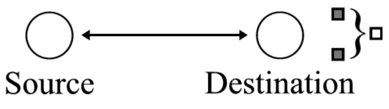
This chapter studies the performance of the improved Packet Merging (iPM) scheme developed in chapter 4. It does this in two ways: the first is through the development of simulations which are used to compare packet merging with an analytical derivation of a Maximal-Ratio Combining (MRC) scheme. The second is through the implementation of practical experiments to show the gains obtainable in a real environment, and to allow an energy audit to be carried out.

5.1 Comparison of ARQ schemes

For the following analysis it is informative to consider four simple cases. The first case is referred to as *traditional ARQ (tradARQ)*. In this case the source node repeats the information when the original packet received at the destination is incorrect. If the second packet is correctly received a success is declared, if not a failure is declared. The second case is referred to as *traditional ARQ with combining (tradARQC)*. In this case the source node again resends the data, if an error occurs. However, upon receiving an erroneous packet on the second attempt, rather than declaring a failure, a combining (either MRC or packet merging) operation takes place. If the combining operation fails only then is a failure declared. The third and fourth cases are analogous to the first and second, respectively, and are referred to as *cooperative ARQ (coopARQ)* and *cooperative ARQ with combining (coopARQC)*. For these cases, rather than the source retransmitting the data the relay is requested to do so, provided it has correctly decoded the packet from the source. For the case when the relay has not correctly decoded the packet it is the source that retransmits. Each of these situations is depicted in figures 5.1 and 5.2.

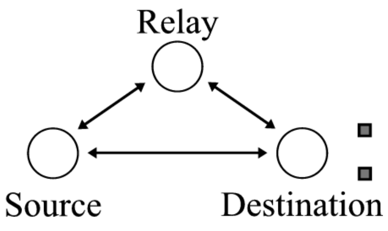


(a) tradARQ

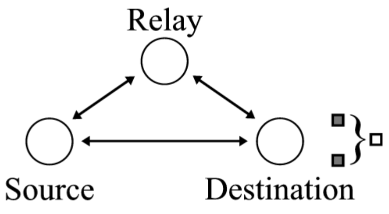


(b) tradARQC

Figure 5.1: Traditional ARQ.



(a) coopARQ



(b) coopARQC

Figure 5.2: Cooperative ARQ.

To help simplify the analysis a total of two transmissions are allowed for the initial study: the original transmission and one retransmission. In section 5.1.3 the simulations are used to extend the total number of possible transmissions per packet to three. Packet merging is most effective once two packets have been received as majority voting is not possible in this case. If more retransmissions are required majority voting then becomes a valid option. However, packet merging will still be required after the first retransmission regardless of the number of subsequent retransmissions. This will be made more clear in section 5.1.3.

As has been noted previously, the type of combining generally assumed is Maximal-Ratio Combining (MRC) which, unfortunately, is normally not possible on resource-constrained devices. As a benchmark, however, the performance of an MRC system is also considered. It should be stressed that MRC is only being used to help gauge the performance of packet merging. The latter is not meant as a replacement for MRC but as a substitute when more powerful combining techniques, such as MRC, are not feasible.

5.1.1 Analytical Model

In [116] an interesting analytical analysis is carried out where the Packet Error Rates (PERs) obtainable using an MRC system are presented for each of the four cases described earlier. The analysis presented is of relevance for this thesis as it permits a performance rating of iPM in comparison with MRC. The modulation scheme considered in this work is Binary Phase Shift Keying (BPSK) and a packet size of 1080 bits is used. The channel assumed is a slowly varying Rayleigh fading channel and the SNR values quoted for a given link are thus average values, with the instantaneous value for any given packet varying accordingly. Due to the slow fading nature of the channel the instantaneous SNR is assumed to remain constant over the course of two transmissions.

In order to simplify the analysis, the Packet Error Rate (PER) for the system under consideration is approximated by [117]:

$$\mathrm{PER}(\gamma) = \begin{cases} 1 & \text{if} \quad 0 < \gamma < \gamma_t \\ \alpha \exp(-g\gamma) & \text{if} \quad \gamma \geq \gamma_t \end{cases} \tag{5.1}$$

where γ is the instantaneous received SNR and γ_t is a pre-defined SNR threshold. A comparison of this approximation with the exact PER, given by

$$1 - (1 - \mathrm{BER}(\gamma))^L, \tag{5.2}$$

is shown in figure 5.3. The values (α, g, γ_t) used for the particular BPSK scheme with

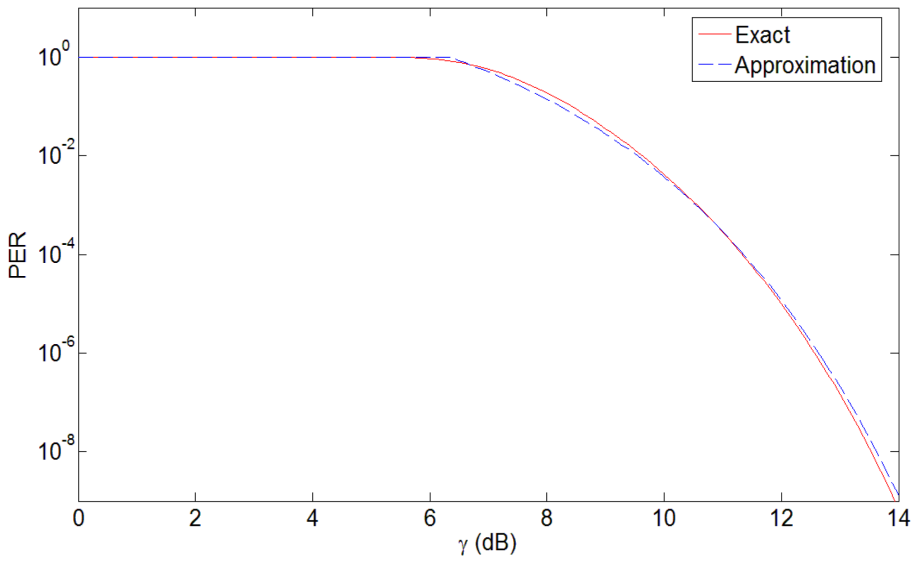


Figure 5.3: Comparison of the exact and approximate functions for the PER of a BPSK system with $L = 1080$ bits in a packet.

a packet size of 1080 bits are (67.7328, 0.9819, 6.3281 dB) which were determined using a least squares fitting method with the threshold γ_t set such that α exp$(-g\gamma_t) = 1$ [117].

Expected values of PERs ($\bar{P}^e$) are derived for each of the four cases (tradARQ, tradARQC, coopARQ and coopARQC) and are repeated here for reference (see [116] for more details). In each case $\sigma_{i,j}$ is the average SNR between nodes i and j (i.e. $\bar{\gamma}_{i,j} = \sigma_{i,j}$) with $i, j \in \{s, r, d\}$ which represents the source, relay and destination channels respectively[1].

The following are the four equations of interest in this chapter:

1). Traditional ARQ:

$$\bar{P}_1^e(\sigma_{s,d}) = 1 - \frac{2\sigma_{s,d}g}{1 + 2\sigma_{s,d}g}\text{exp}\left(-\frac{\gamma_t}{\sigma_{s,d}}\right) \tag{5.3}$$

[1]The subscripts in $\bar{\gamma}_{i,j}$ will be dropped when $\bar{\gamma}$ refers to multiple quantities.

2). Traditional ARQ with Maximal Ratio Combining:

$$\begin{aligned}\bar{P}_2^e(\sigma_{s,d}) = 1 - &\frac{2\sigma_{s,d}g}{1+2\sigma_{s,d}g}\text{exp}\left(-\frac{\gamma_t}{2\sigma_{s,d}}\right)\\ &- \frac{\sigma_{s,d}g}{(1+2\sigma_{s,d}g)(1+3\sigma_{s,d}g)}\text{exp}\left(-\frac{\gamma_t}{\sigma_{s,d}}\right)\text{exp}(-g\gamma_t)\end{aligned} \tag{5.4}$$

In the cooperative cases it is necessary to consider whether or not the relay received the source packet correctly. So:

3). Cooperative ARQ (when the relay has correctly received the source packet):

$$\bar{P}_{3-r}^e(\sigma_{s,d},\sigma_{r,d}) = \left(1 - \frac{2\sigma_{s,d}g}{1+\sigma_{s,d}g}\text{exp}\left(-\frac{\gamma_t}{\sigma_{s,d}}\right)\right)\left(1 - \frac{\sigma_{r,d}g}{1+\sigma_{r,d}g}\text{exp}\left(-\frac{\gamma_t}{\sigma_{r,d}}\right)\right) \tag{5.5}$$

and the probability that the relay correctly receives the source packet:

$$P(\sigma_{s,r}) = \frac{\sigma_{s,r}g}{1+\sigma_{s,r}g}\text{exp}\left(-\frac{\gamma_t}{\sigma_{s,r}}\right) \tag{5.6}$$

which gives an overall average PER of

$$\bar{P}_3^e(\sigma_{s,d},\sigma_{r,d},\sigma_{s,r}) = (1-P(\sigma_{s,r}))\bar{P}_1^e(\sigma_{s,d}) + P(\sigma_{s,r})\bar{P}_{3-r}^e(\sigma_{s,d},\sigma_{r,d}) \tag{5.7}$$

4). Cooperative ARQ with Maximal Ratio Combining (when the relay has correctly received the source packet):

$$\bar{P}^e_{4-r}(\sigma_{s,d},\sigma_{r,d}) = 1 - \exp\left(-\frac{\gamma_t}{\sigma_{s,d}}\right) + \frac{1}{(1+2\sigma_{s,d}g)(1+\sigma_{s,d}g)}\exp\left(-\frac{\gamma_t}{\sigma_{s,d}}\right) \\ - \frac{\sigma^2_{s,d}g}{(1+\sigma_{s,d}g)(\sigma_{s,d}-\sigma_{r,d})}\left(\exp\left(-\frac{\gamma_t}{\sigma_{s,d}}\right) - \exp\left(-\frac{\gamma_t}{\sigma_{r,d}}\right)\right) \tag{5.8}$$

along with the probability that the relay correctly receives the packet (eq. (5.6)) the overall average PER is given by:

$$\bar{P}^e_4(\sigma_{s,d},\sigma_{r,d},\sigma_{s,r}) = (1-P(\sigma_{s,r}))\bar{P}^e_2(\sigma_{s,d}) + P(\sigma_{s,r})\bar{P}^e_{4-r}(\sigma_{s,d},\sigma_{r,d}) \tag{5.9}$$

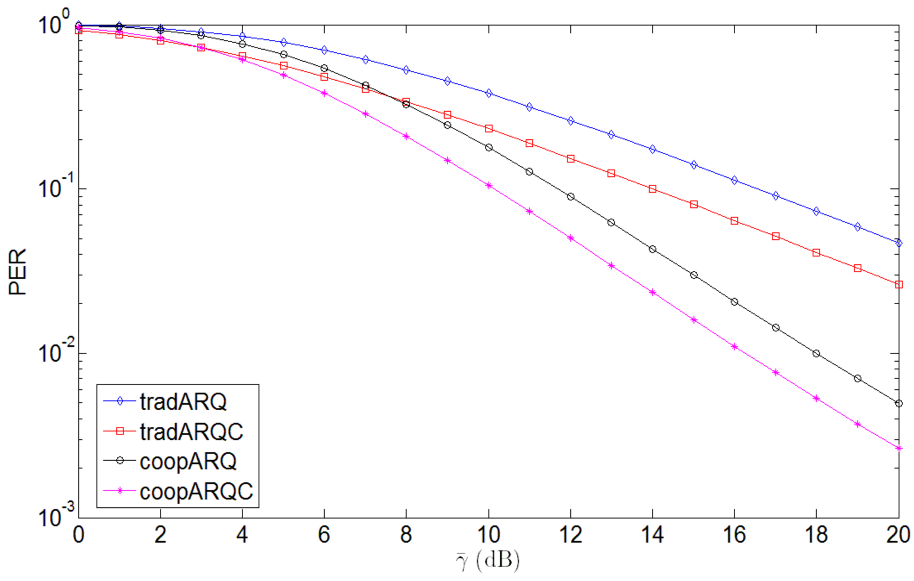


Figure 5.4: Plot of the PER curves for equations 5.3, 5.4, 5.7, 5.9 with $\bar{\gamma} = \sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} = 20$ dB.

The PER equations 5.3, 5.4, 5.7, 5.9 imply that a packet is only considered to be in error if, after two transmission attempts (and a combining operation in the case of equations 5.4 and 5.9), the receiver still has not obtained a correct packet. This particular way of defining the PER is used throughout rest of this chapter (including the situation where two retransmissions are allowed). It should be stressed, however, that packet merging is not limited to the case of retransmissions in the usual sense. In a multi-hop network the same packet may be received by a node multiple times due to *overhearing*. In these situations packet merging can be used to take advantage of a situation that is normally considered undesirable in multi-hop networks.

A plot of equations 5.3, 5.4, 5.7, 5.9 for the symmetric channel case $\bar{\gamma} = \sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} = 20$ dB can be seen in figure 5.4. There are two different types of gains observable in this figure, each of which is with respect to the traditional ARQ scheme (tradARQ). Firstly, enhancing traditional ARQ with MRC produces an SNR gain due to combining. Secondly, using a diversity scheme without combining (i.e. coopARQ) produces an SNR gain due to diversity (again, with respect to traditional ARQ). This latter case is analogous to selection combining where the signal with the largest SNR is chosen and the other signals disregarded. In this thesis, however, “combining” implies that use is made of the signals (or packets) on the other channels to assist in packet recovery. When combining is used in the cooperative scenario (i.e. cooperative ARQ with combining), using the information from the other channels leads to yet a further gain in addition to the diversity gain.

Figure 5.5 shows a plot of the SNR gains *due to combining alone* for both the traditional case and the cooperative case. This means that the “Traditional” gain is the difference in SNRs required for a particular PER between tradARQ and tradARQC. For the “Cooperative” case it is the difference between coopARQ and coopARQC. The figure therefore does not take diversity effects into account (i.e. the SNR gain between coopARQ and tradARQ) as this is not what is being studied in this thesis. Whilst diversity gains are very important, and constitute the majority of the SNR gain in the cooperative system, they are discussed elsewhere (e.g. [57, 97]). The SNR gain due to combining (hereafter referred to as the SNR gain, or simply, gain) for the traditional system is a lot larger than that of the cooperative system over most of the PER

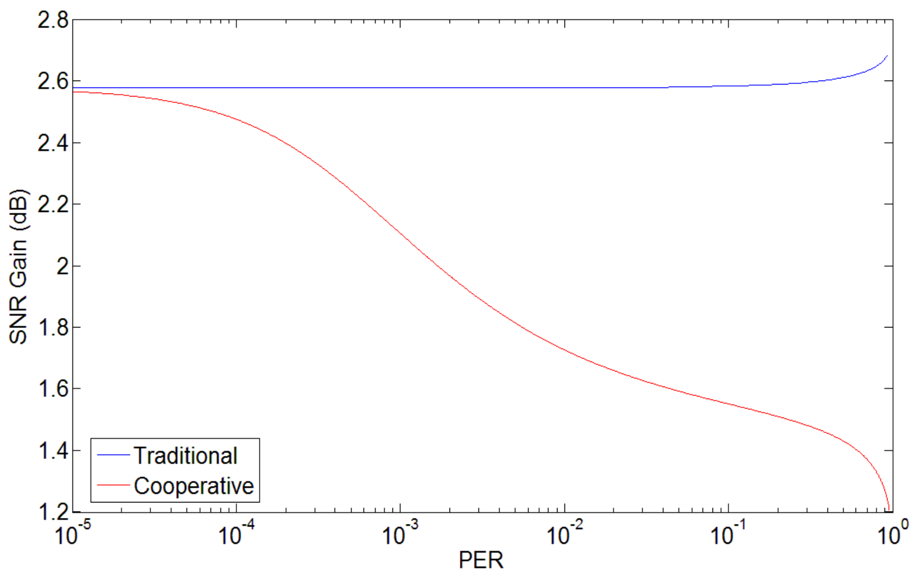


Figure 5.5: SNR gain for both traditional and cooperative system with $\sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} =$ 20 dB.

range. However, for a non-perfect source-relay channel (such as that investigated in figure 5.5), the gain of the cooperative ARQ system approaches that of the traditional ARQ system in the very high SNR regions (i.e. as the PER tends to zero).

Consider now figure 5.6. In this case $\sigma_{s,d} = \sigma_{r,d}$ as before but now the source-relay channel is perfect. It can be seen that the SNR gain for the cooperative system never approaches that of the traditional system. The maximum gain for the cooperative case is only about 1.5 dB whereas for the traditional case it is 2.58 dB.

Three main questions arise from figures 5.5 and 5.6:

1. Why does the gain of the cooperative system with the non-perfect source-relay channel approach that of the traditional system as the PER tends to zero?
2. Why is the gain of the traditional system (and the cooperative system for perfect source-relay channel) largely invariant over all values of PER?
3. Why is the gain of the cooperative system smaller than that of the traditional

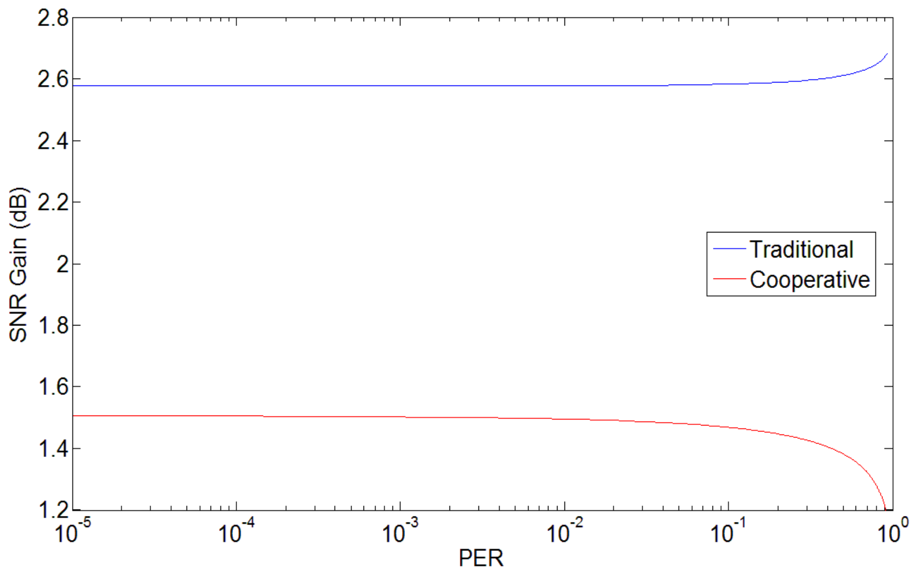


Figure 5.6: SNR gain for both traditional and cooperative system with $\sigma_{s,d} = \sigma_{r,d}$ and a perfect source-relay channel.

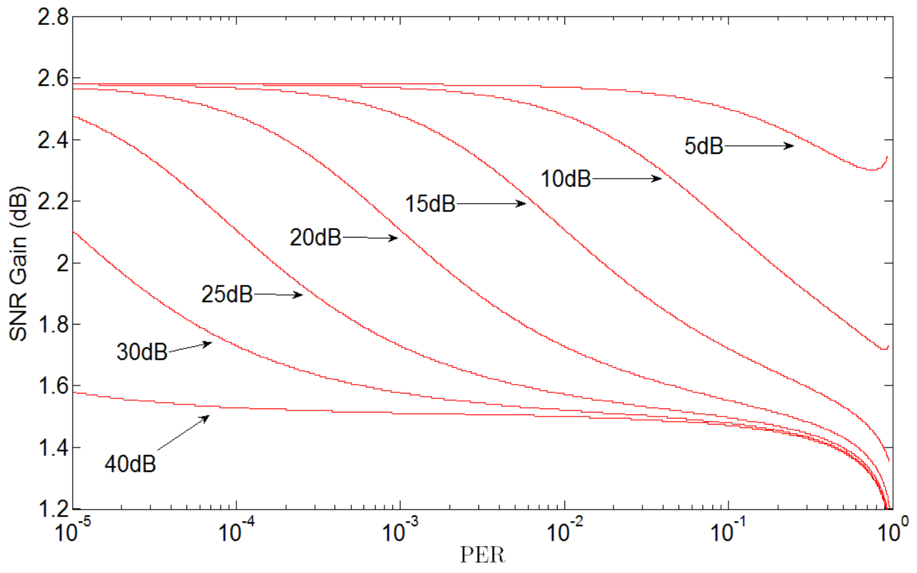


Figure 5.7: SNR gain for the cooperative system with $\sigma_{s,d} = \sigma_{r,d}$ and $\sigma_{s,r} = \{5, 10, 15, 20, 25, 30, 40\}$ dB.

system?

To answer the first question the following can be noted. When the source-relay channel isn't perfect there is a non-zero probability that the relay will decode the packet incorrectly. For the cooperative ARQ systems this means that, in the case where the relay incorrectly decodes, the source will be required to retransmit the packet and hence diversity will not be leveraged for this particular transmission. In the limiting case where the source-relay channel is so bad that no packet can ever be received, the system reverts back to a traditional ARQ system. This can be seen from equations 5.7 and 5.9 by setting $P(\sigma_{s,r}) = 0$ giving $\bar{P}_3^e(\sigma_{s,d}, \sigma_{r,d}, \sigma_{s,r}) = \bar{P}_1^e(\sigma_{s,d})$ and $\bar{P}_4^e(\sigma_{s,d}, \sigma_{r,d}, \sigma_{s,r}) = \bar{P}_2^e(\sigma_{s,d})$, respectively.

When there is no channel available between the source and relay therefore, the cooperative systems revert back to their traditional counterparts and the combining gains will be equal for both systems (i.e. 2.58 dB for the analytical model under consideration). This doesn't really address the question of why the cooperative gain moves towards that of a traditional gain as the PER tends to zero. Given a specific value of $\sigma_{s,r}$, the PER tending to zero is equivalent to $\sigma_{s,d} = \sigma_{r,d}$ tending towards infinity. When $\sigma_{s,d} = \sigma_{r,d} < \sigma_{s,r}$ the cooperative system is the dominant one. However, when $\sigma_{s,d} = \sigma_{r,d} > \sigma_{s,r}$ it is more likely that the two packets will come from the source than the relay and the traditional system is the dominant one. Figure 5.7 helps clarify this point by showing a plot of the gains obtained for 7 different values of $\sigma_{s,r}$. The slope of each curve begins to increase dramatically once $\sigma_{s,d} = \sigma_{r,d} > \sigma_{s,r}$ (this can be seen in conjunction with figure 5.4). When $\sigma_{s,r} = 5$ dB the system has effectively reverted back to a tradARQ system. Also, only when $\sigma_{s,r}$ reaches about 40 dB does the plot of the gain behave similar to that of a perfect source-relay channel.

An answer to the second question is provided in [116] where it is shown that for large values of SNR $\sigma\bar{P}_1^e(\sigma)$ reduces to

$$\left(\gamma_t + \frac{1}{2g}\right)$$

and $\sigma\bar{P}_2^e(\sigma)$ reduces to

$$\left(\frac{\gamma_t}{2}+\frac{1}{2g}-\frac{\exp(-g\gamma_t)}{6g}\right).$$

The SNR gain is related to the ratio of $\bar{P}_1^e(\sigma)/\bar{P}_2^e(\sigma)$ which is constant above a certain value of SNR. A similar situation holds for the cooperative scenario (see [116] for more details).

The third question can be answered in a similar manner to the second. As mentioned, the SNR gains are related to the ratios of $\bar{P}_1^e(\sigma)/\bar{P}_2^e(\sigma)$ and $\bar{P}_3^e(\sigma)/\bar{P}_4^e(\sigma)$. Through calculations it can be shown, for the case at hand, that these result in values of approximately 1.81 and 1.36, respectively. This in itself shows that the cooperative gain will be smaller. More significantly however, the SNR gain is also *inversely* proportional to the slope of the curves in figure 5.4, which is larger for the cooperative system. In the limiting case where both curves are horizontal (i.e. PER doesn't depend on SNR), for example, the SNR gain is infinite even though the ratio of PER values is not.

5.1.2 Simulated Setup: One Retransmission

In order to analyse packet merging and compare it with the analytical equations for MRC just described, a simulation was devised. The purpose of using a simulation was not only for comparative purposes but also because it offered a more controlled environment in which to study packet merging (e.g. in a slow Rayleigh fading with no interference).

The simulation consisted of a three node architecture: a Source (S), a Relay (R) and a Destination (D). The channel model used was slow Rayleigh flat fading with Additive White Gaussian Noise (AWGN). The channel gain remained constant during the time to transmit two consecutive packets. A simplified block diagram of the simulated

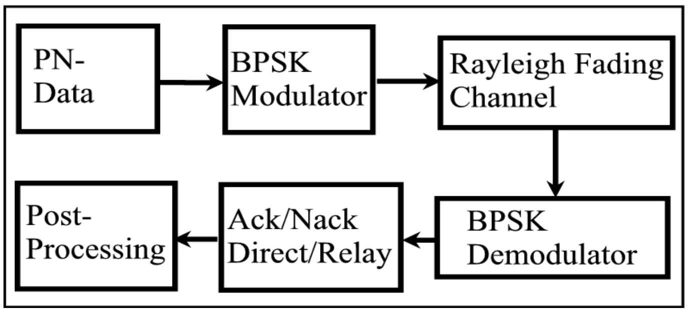


Figure 5.8: Simplified block diagram of the simulation setup using BPSK.

environment is shown in figure 5.8. A Pseudorandom Noise (PN) sequence was used to create a data packet which was then modulated using BPSK[2]. The modulated signal was transmitted over the channel. The received signal was then demodulated and the accuracy of the packet determined by comparing it with the original data. If the packet was received correctly no further action was taken. However, if the packet was received incorrectly then the data was retransmitted on the same channel in one case (traditional ARQ), and on an independent fading channel in the second case (cooperative ARQ). The simulation was mainly concerned with analysing the gains obtainable using packet merging and did not attempt to pick the best packet or perform a structured search as was done for iPM — these techniques were added to simplify the practical implementation of iPM making it faster and more energy efficient. It simply determined if a combining was possible by determining whether $w(V_A) \leq w(V_A)_{max}$ and whether a hidden error occurred. It also did not take into account false-positives but, as shown in section 4.7.2, these can be reduced effectively to zero.

The simulator was validated by comparing its output to the PERs given by equations 5.3 and 5.7. This was achieved by setting the packet size to 1080 bits and transmitting 100,000 packets for each value of average SNR. As with the analytical model $\sigma_{s,d} = \sigma_{r,d}$, both of which were varied from 0-20 dB. The source-relay channel was assumed

[2]BPSK was used instead of O-QPSK with half sine pulse shaping [5] to facilitate comparison with the analytical model.

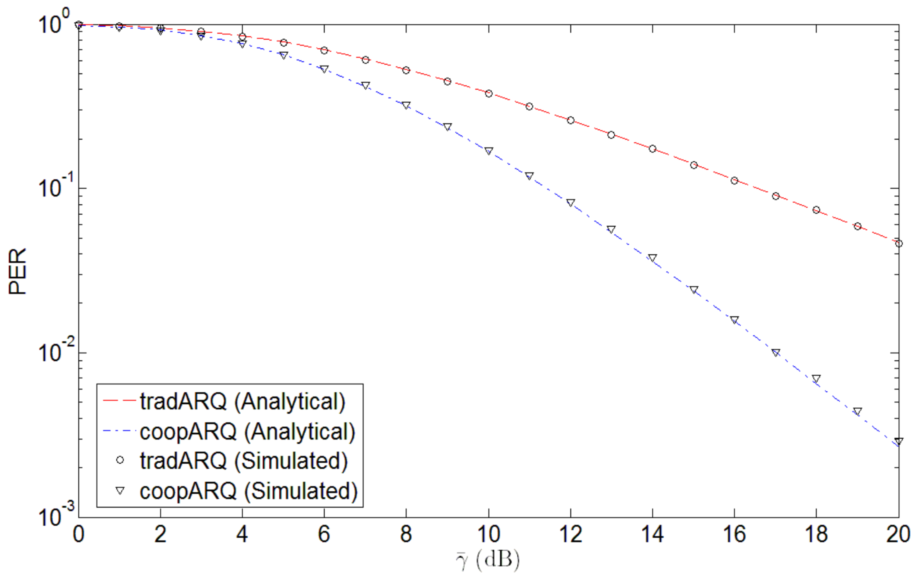


Figure 5.9: Comparison of the simulated versus the analytical results for a BPSK system over a slow Rayleigh fading channel with a packet size of 1080 bits. The source-relay channel was assumed to be perfect.

to be perfect in order to reduce the complexity of the simulation. A comparison was carried out using only the tradARQ case and coopARQ case as the simulator was not designed to implement maximal-ratio combining. Figure 5.9 shows the results of the tests where it can be seen that the simulation curves agree closely with those of the analytical model.

Having validated the simulation, packet merging was then implemented in order to ascertain the performance gains achievable. The gains are determined not only by $\sigma_{s,d}, \sigma_{r,d}$ and $\sigma_{s,r}$ but also by the size of $w(V_A)_{max}$. Larger values of $w(V_A)_{max}$ offer increasingly large gains (as it permits the correction of a larger amount of errors); however, the computational complexity also increases. The gains obtainable for $w(V_A)_{max} = 10$ were determined using the results shown in figure 5.10. As with the curves obtained from the analytical model (figure 5.4), traditional ARQ achieves a larger combining gain than cooperative ARQ. However, the overall gain, including

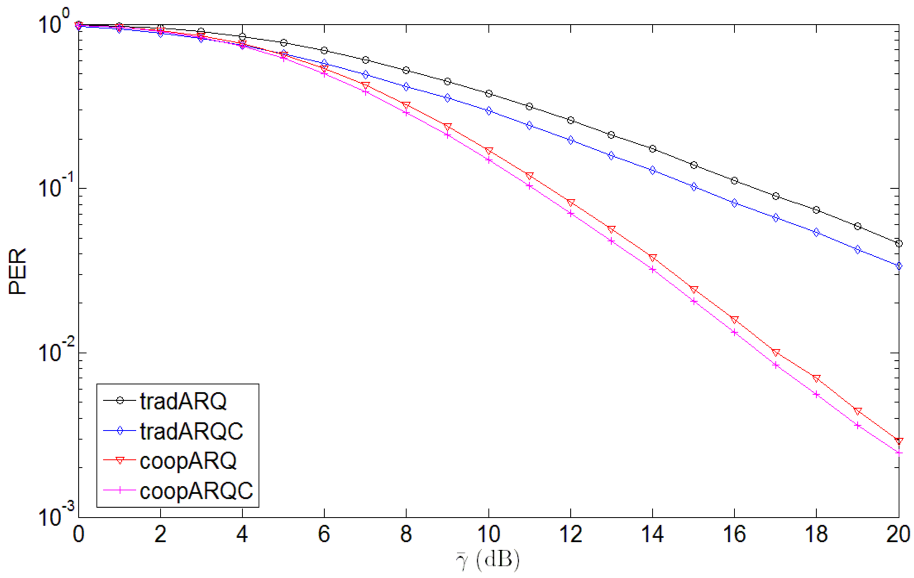


Figure 5.10: Packet error rate curves for a BPSK system with a packet size of 1080 bits and $w(V_A)_{max} = 10$.

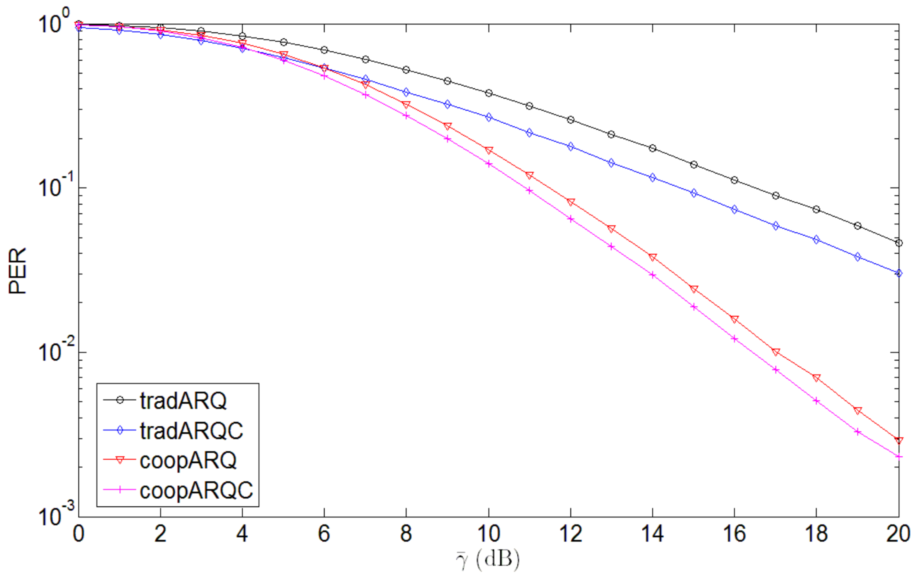


Figure 5.11: Packet error rate curves for a BPSK system with a packet size of 1080 bits and $w(V_A)_{max} = 15$.

diversity, is much larger in the latter case. The gain improvement obtained when using $w(V_A)_{max} = 15$ is evident from figure 5.11. Although the computational complexity of iPM (and hence the practical feasibility of this method) is increased somewhat in this case, with optimised code (or hardware implementations) it may become more feasible. This will be studied in more detail in section 5.2 when the energy consumption of a node implementing these techniques is analysed.

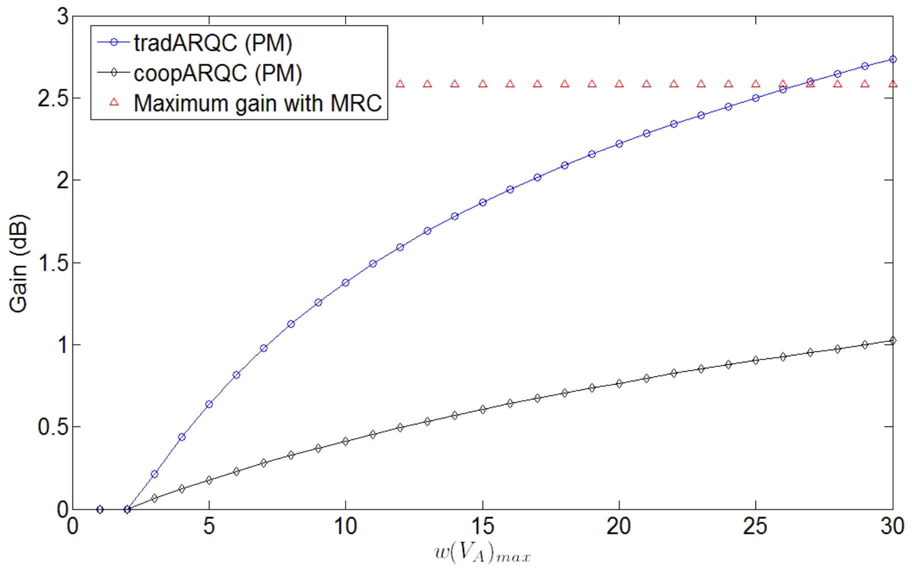


Figure 5.12: Gains achievable using Packet Merging (PM) over different values of $w(V_A)_{max}$ for a packet size of 1080 bits with a perfect source-relay channel.

Due to the assumption of a perfect source-relay channel, the gains for both the traditional and cooperative ARQ systems are constant over the majority of SNR values. The same situation was noted for the analytical model in figure 5.6. The effects of permitting different values of $w(V_A)_{max}$ on this gain is shown in figure 5.12. For traditional ARQ the gains obtained for $w(V_A)_{max} = 10$ and $w(V_A)_{max} = 15$ are 1.39 dB and 1.86 dB respectively. In the cooperative case they are .42 dB and .61 dB with the perfect source-relay channel. Of course, if diversity is taken into account the overall gains of the cooperative system would be those shown in figure 5.12 *plus* a gain of approximately 0.3 dB to 6.5 dB due to diversity — higher values of SNR leading to larger gains.

The assumption of a perfect source-relay channel means that cooperation is always available to be leveraged (where necessary) and that the system never resorts back to traditional ARQ. The combining gain of the traditional system is always larger than the cooperative system for a perfect source-relay channel (cf. figures 5.6 and 5.12). If an imperfect channel is assumed the combining gain of the cooperative case would be larger than .42 dB (for the same value of $w(V_A)_{max}$) in the regions of mid-SNR tending towards that of the traditional ARQ schemes in the limit of high SNR (cf. figure 5.5) as the system reverts back to traditional ARQ on occasion. This compensates somewhat for the loss of diversity due to the imperfect source-relay channel.

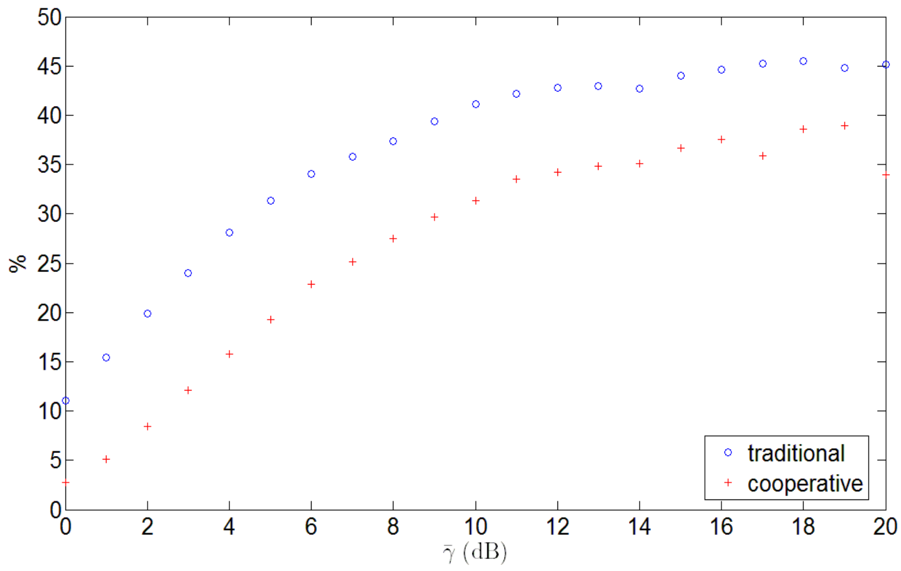


Figure 5.13: The percentage of ambiguity vectors obtained for each SNR with a Hamming weight less than or equal to 15. In this case $\bar{\gamma} = \sigma_{s,d} = \sigma_{r,d}$.

The larger cooperative gain obtainable for the imperfect source-relay channel can be understood further by considering figure 5.13. This is a plot of the percentage of ambiguity vectors for each SNR with a Hamming weight less than or equal to 15. Both the traditional case and the cooperative case are shown. It can be seen that, for each value of SNR, the cooperative situation produces, on average, 9.5% fewer ambiguity vectors in this range than the traditional case. Now consider the method used to calculate the PER in each case. In tradARQ and coopARQ a packet is only declared in error if *both* the first and the second transmissions are in error. In tradARQC and coopARQC a packet is only declared in error if both the first and second transmissions are in error *and* the packet merging operation fails. If packet merging always fails then the PER of each system reverts back to that of the tradARQ and coopARQ cases, respectively. With a lower percentage of ambiguity vectors less than or equal to $w(V_A)_{max}$ in the cooperative case than in the traditional case this means the gain will also be less as there will be fewer occasions when packet merging is a viable option. However, as the source-relay channel degrades from perfection, an increasing number of retransmitted packets come from the source node creating more opportunities for a successful merging operation.

It is interesting to note from figure 5.12 that the gain of the traditional packet merging system actually exceeds that of the corresponding MRC system for very large values of $w(V_A)_{max}$. In reality, however, it would not be feasible to process such a large number of errors. For more realistic levels of $w(V_A)_{max}$ the gains are not as large as MRC, which is to be expected. The packet merging scheme is not meant as a replacement of MRC but is to be used where the hardware resources required for techniques such as MRC are not available.

It will be highly beneficial to use iPM to enhance protocols such as SPaC. In these cases iPM will not be used as a stand-alone device but in conjunction with other forms of combining such as packet decoding. As packet merging has already been implemented alongside packet decoding [18] it is relatively straight forward to retrofit the existing architecture with iPM. The achievable gains will then be larger and the system more energy efficient (cf. section 5.2). To keep things in perspective however, recall that the coding gain obtainable by the (7,4) Hamming error correction code is approximately

.45 dB at an SNR of 10 dB (cf. figure 2.5)[3]. The (7,4) Hamming code requires 3 redundant bits for every 4 information bits. For a packet size of 300-bits this is an extra 225-bits of redundancy which adds significantly to the cost of a transmission and reception (cf. section 5.2).

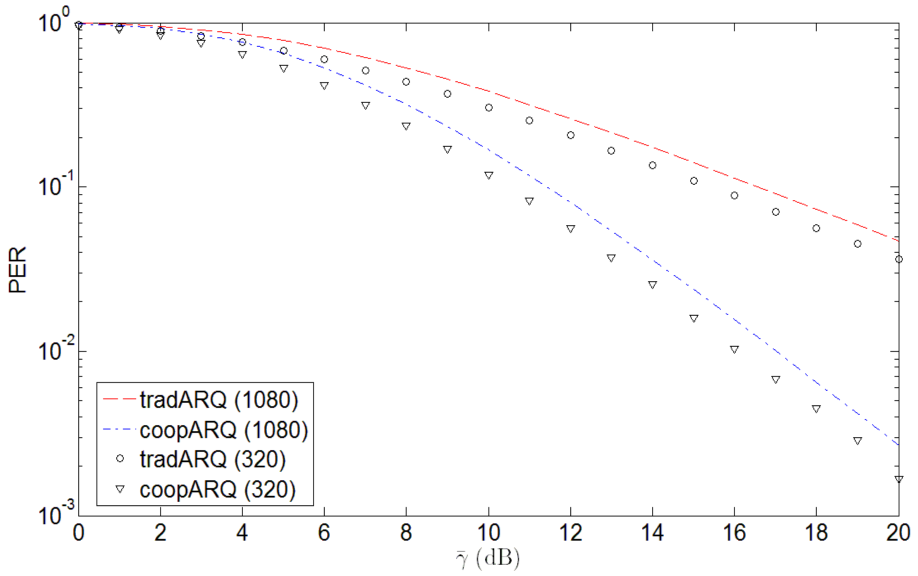


Figure 5.14: Comparison of two BPSK systems transmitting over a slow Rayleigh fading channel with packet sizes of 320 bits and 1080 bits. The source-relay channel is assumed to be perfect.

A more commonly used packet size for WSNs is about 320 bits [115]. The simulation was run again, with this packet size, for both tradARQ and coopARQ (i.e. without any combining). Figure 5.14 shows the resulting plots compared with those of a tradARQ and coopARQ system using a packet size of 1080 bits. The system with the packet size of 320-bits outperforms that with the larger packet size of 1080-bits. This is to be expected as the probability of obtaining one or more bits in error in a packet increases with packet size (cf. equation 5.2). The percentage of ambiguity vectors that have Hamming weights less than or equal to $w(V_A)_{max}$ therefore increases for smaller

[3]Of course this coding gain is calculated in terms of Bit Error Rate (BER) and not Packet Error Rate (PER) but it is still informative to keep it in mind.

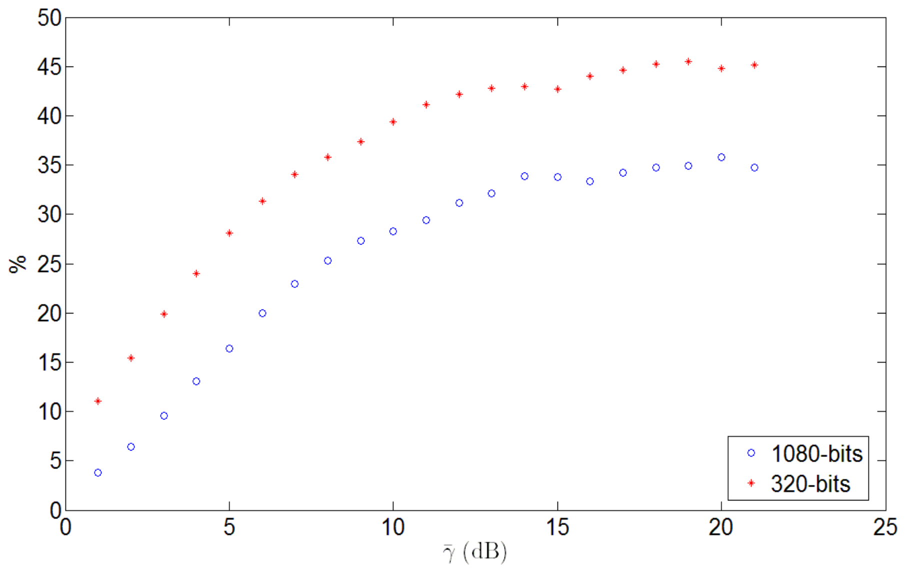


Figure 5.15: Percentage of ambiguity vectors obtained using tradARQC with Hamming weights less than or equal to 15. Both 320-bit and 1080-bit cases considered.

packet sizes (as shown in figure 5.15 for the traditional ARQ case).

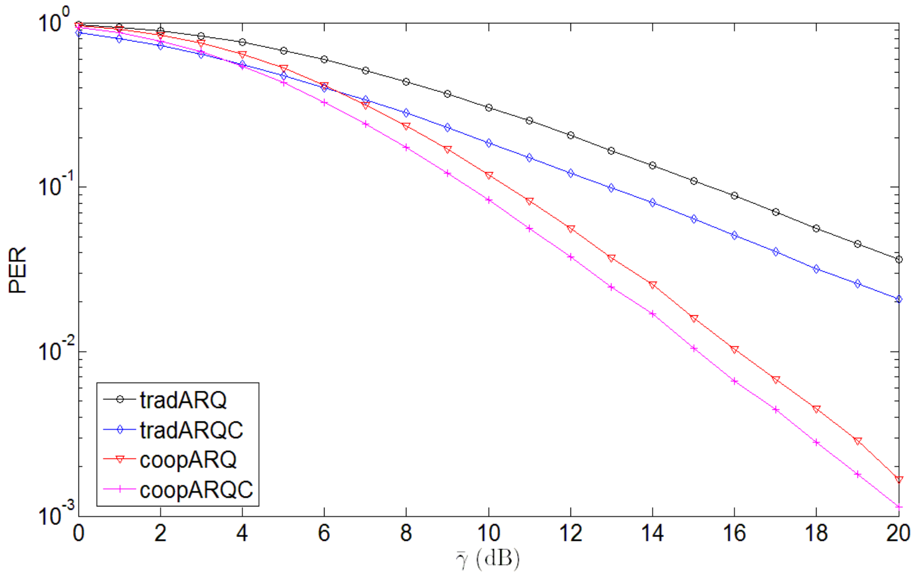


Figure 5.16: Packet error rate curves for a BPSK system with a packet size of 320 bits and $w(V_A)_{max} = 15$.

Figure 5.16 shows the PERs obtained for tradARQ, tradARQC, coopARQ, coopARQC

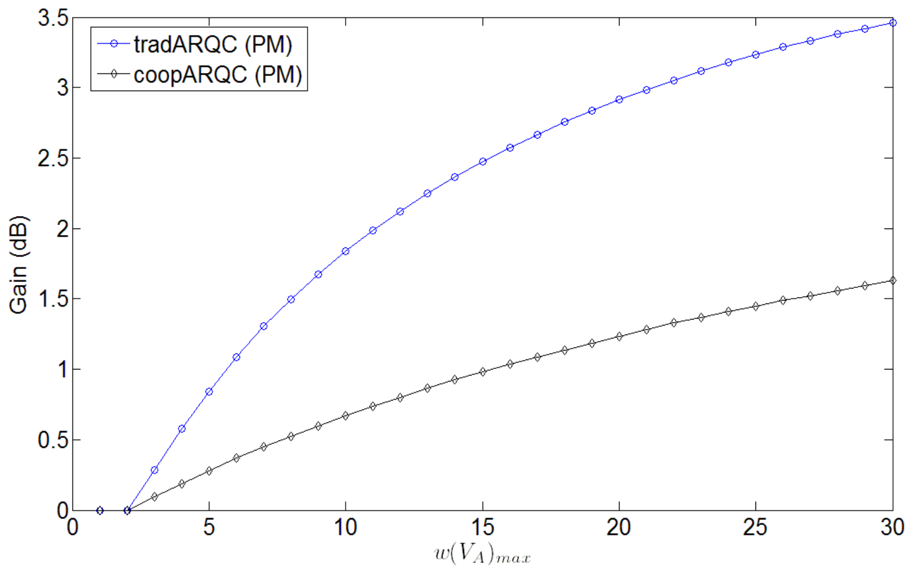


Figure 5.17: Gains achievable using Packet Merging (PM) over different values of $w(V_A)_{max}$ for a packet size of 320 bits with a perfect source-relay channel.

for a value of $w(V_A)_{max} = 15$. Interestingly, combining works better for the smaller and more commonly used packet size [115], [118], [2]. This can be seen when plotting the gains obtained over all values of $w(V_A)_{max}$ (figure 5.17). For the traditional case the gains at $w(V_A)_{max} = 10$ and $w(V_A)_{max} = 15$ are 1.84 dB and 2.47 dB respectively. This can be compared with the gains of 1.39 dB and 1.86 dB for the system with a packet size of 1080-bits. In the cooperative case the gains are .67 dB and .98 dB with a perfect source-relay channel. This can be compared with the gains of .42 dB and .61 dB for the cooperative case using a packet size of 1080-bits and a perfect source-relay channel. A summary of the gains obtainable in each case is provided in table 5.1.

Packet Size	1080 bits		320 bits	
$w(V_A)_{max}$	10	15	10	15
Traditional ARQ	1.39	1.86	1.84	2.47
Cooperative ARQ	0.42	0.61	0.67	0.98

Table 5.1: Gains (in dB) for each method used when two transmission attempts are permitted.

5.1.3 Simulated Setup: Two Retransmissions

Packet merging is not restricted to the case of a single retransmission. However, when more than one retransmission is permitted majority voting now becomes possible; for the single retransmission case it is not. Regardless of the number of retransmissions, however, the results of the previous section are still valid and show that merging is preferable once the first retransmission has been received. This section considers the effects of packet merging when two retransmissions are permitted.

The simulation was modified to permit two retransmissions. Again, a slow Rayleigh fading channel was considered where it was assumed that the channel gains remained the same over the course of three transmissions. As majority voting is now an option the following strategies are possible in both traditional "t" and cooperative "c" forms:

1. Retransmit the packet a maximum of three times and if all three packets are in error declare a failure (tARQ/cARQ).
2. If the third attempt fails perform a majority vote on the three packets available (tMaj/cMaj).
3. Perform a packet merging on the second attempt and a majority vote on the third attempt (tMerMaj/cMerMaj).
4. Perform a packet merging on the second attempt and a packet merging on the third attempt (tMerMer/cMerMer).
5. Perform a packet merging on the second attempt and, initially, a majority vote on the third attempt. If the majority vote fails then a merging operation is attempted (tMerMajMer/cMerMajMer).

Consider now figure 5.18 which shows the PER of a traditional/cooperative simple (i.e. without combining) scheme using both one and two retransmission attempts. For

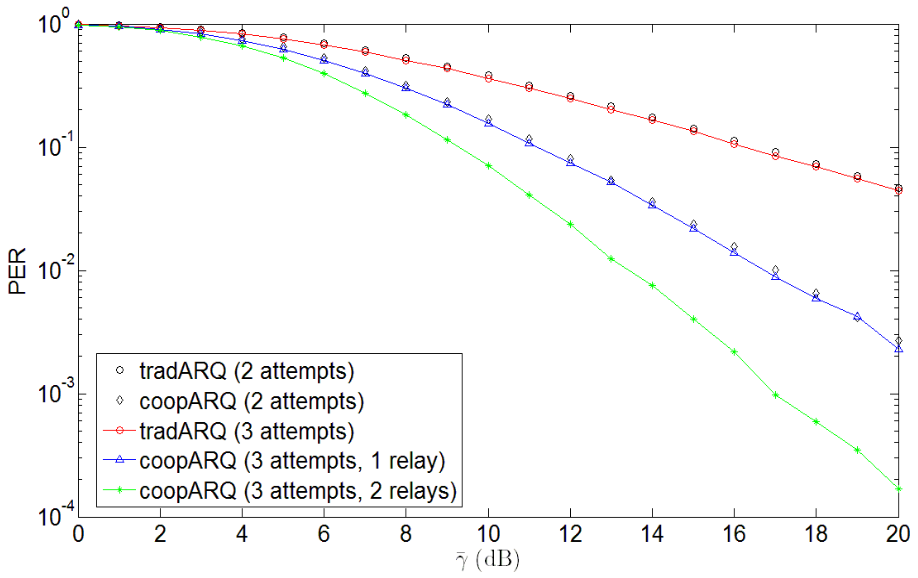


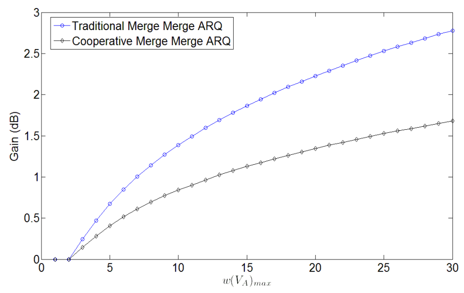
Figure 5.18: Comparison of using a traditional and cooperative ARQ scheme with both two and three retransmission attempts, and one and two different relays. The packet size was set to 1080 bits in all cases.

the cooperative case the use of both one and two relays is also considered: the first retransmission is sent from one of the relays and the second from either the same relay or a different one. It is interesting to note that for both the traditional and one relay cooperative ARQ systems, increasing the number of permitted transmissions by one does not improve the PER significantly (an SNR gain of approximately .31 dB is achieved). In the cooperative case, however, if the second retransmission attempt is sent from a different relay (i.e. over another independent channel) then the increased diversity gain provides a large improvement to the system.

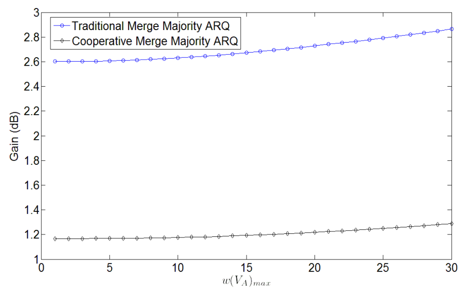
Each of the ten transmission/combining strategies described above was implemented in turn, with a packet size of 1080 bits and one relay used in the cooperative system. For tMaj the gain obtained relative to tARQ was approximately 2.60 dB[4]. This fact highlights the implications of using SNR gain as the sole metric for system performance. When MRC is used with two retransmissions the gain was shown to be 2.58

[4]This value is independent of $w(V_A)_{max}$ of course.

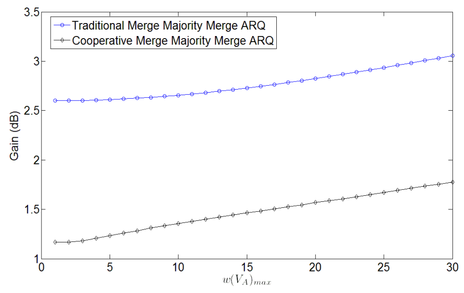
dB for the traditional system. This, of course, does not mean that a simple majority voting scheme is superior to MRC: the number of transmissions required for the latter is significantly less than that for the former. Figures 5.19(a), 5.19(b) and 5.19(c) show plots of the gains versus $w(V_A)_{max}$ for each of the remaining methods considered (i.e. other than tMaj/cMaj). Table 5.2 summarises all results for $w(V_A)_{max} = 10$ and $w(V_A)_{max} = 15$.



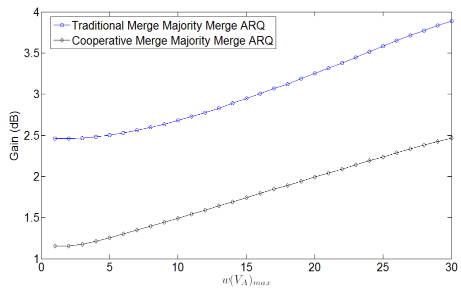
(a) Merge Merge (1080 bits)



(b) Merge Majority (1080 bits)



(c) Merge Majority Merge (1080 bits)



(d) Merge Majority Merge (320 bits)

Figure 5.19: Gains obtained using different combinations of packet merging and majority voting with both traditional and cooperative systems.

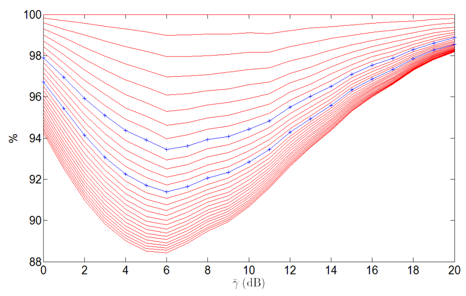
Performing a merging operation alone on both the second and third attempts respectively does not offer the best gains. The problem with this particular approach is that packet merging is dependent on $w(V_A)_{max}$. This is not the case for a majority vote which can be applied regardless of the number of errors in the packets. The highest gains are achievable using tMerMajMer/cMerMajMer as is to be expected.

Consider now a packet size of 320 bits. Table 5.3 summarises the results for each

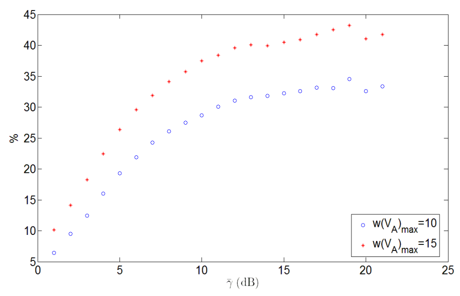
Method	tMerMer	cMerMer		tMaj	cMaj	
Relays	-	1	2	-	1	2
$w(V_A)_{max} = 10$	1.37	0.84	0.71	2.60	1.17	0.78
$w(V_A)_{max} = 15$	1.86	1.13	0.99	2.60	1.17	0.78

Method	tMerMaj	cMerMaj		tMerMajMer	cMerMajMer	
Relays	-	1	2	-	1	2
$w(V_A)_{max} = 10$	2.63	1.18	0.85	2.66	1.36	1.01
$w(V_A)_{max} = 15$	2.67	1.19	0.90	2.73	1.47	1.19

Table 5.2: Gains in dB for each method used when three transmission attempts are permitted. A packet size of 1080 bits is used.



(a) Percentage of total packets transmitted



(b) Reduction in the required number of third transmissions

Figure 5.20: Consideration of the actual number of transmissions required when using merging.

method. In this case packet merging offers quite a large improvement. The gain of tMaj over tARQ is now only 2.46 dB and that of tMerMajMer, 2.95 dB. The gain as a function of $w(V_A)_{max}$ is plotted in figure 5.19(d). When compared with figure 5.19(c) it can be seen that merging is a much more effective strategy for smaller packet sizes.

For each packet size, however, it would again appear that very large gains are obtainable using *only* a majority vote. This issue was noted earlier and the "gain" metric used needs to be qualified. Section 5.1.2 clearly shows the advantages of using packet merging once an initial retransmission has been received. These advantages hold regardless of the number of retransmissions. When permitting more than one retransmission however, it is necessary to consider the amount of transmissions that were actually required to move a packet from source to destination. Figure 5.20(a) shows the percentage of transmissions required when using merging on the second attempt compared with that without using merging and going straight to the third attempt. Each curve of the plot represents a different value of $w(V_A)_{max}$, from one to 30. The cases for $w(V_A)_{max} = 10$ and $w(V_A)_{max} = 15$ are highlighted in blue where it can be seen that, at $\gamma = 6$, only 93% ($w(V_A)_{max} = 10$) and 91% ($w(V_A)_{max} = 15$) of transmissions are required, respectively. These curves were computed by calculating the total number of transmissions required to move 100,000 packets from source to destination in both cases, given a maximum retransmission count of two.

Method	tMerMer	cMerMer		tMaj	cMaj	
Relays	-	1	2	-	1	2
$w(V_A)_{max} = 10$	1.89	1.16	1.08	2.46	1.15	0.81
$w(V_A)_{max} = 15$	2.56	1.59	1.50	2.46	1.15	0.81
Method	tMerMaj	cMerMaj		tMerMajMer	cMerMajMer	
Relays	-	1	2	-	1	2
$w(V_A)_{max} = 10$	2.57	1.20	0.96	2.68	1.49	1.27
$w(V_A)_{max} = 15$	2.72	1.28	1.05	2.95	1.74	1.58

Table 5.3: Gains (in dB) for each method used when three transmission attempts are permitted. A packet size of 320 bits is used.

Of course figure 5.20(a) masks the effects of merging somewhat as it takes into account *all* packets transmitted. For example, if merging was a perfect combining mechanism and had the ability to correct every erroneous packet pair on the second transmission attempt then the reduction in the amount of transmissions required for both situations would only be 33%, assuming each packet was always received incorrectly. This is because two transmissions are required when using merging on the second attempt whereas three are required otherwise. For the situation in which a third transmission is never required in either case, due to a very good channel, the reduction in the number of transmissions as a result of merging is zero. This explains why the curves in figure 5.20(a) move towards 100% in regions of high SNR. To obtain a clearer understanding of the performance of packet merging in this situation it is therefore more suitable to look at the *reduction* in the required number of third transmissions. A plot of these values is provided in figure 5.20(b). This graph is very similar to that of figure 5.15, which is to be expected as merging is still as effective on the second attempt as has been shown in the section 5.1.2.

5.2 Energy Measurements

This thesis has developed an improved and more efficient packet merging scheme than techniques currently applied. Using iPM the overall energy consumption of the network can be reduced, not only through a decreased number of retransmissions but also through a reduction in the number of collisions and failed transmission attempts (in the case of a carrier sense protocol — see chapter 2). To help determine the performance of iPM this section studies the energy consumption of a typical mote type architecture and discusses the energy cost of performing a merging operation. This is then compared with communication costs. A study such as the one carried in this section is not representative of all devices and scenarios of course. However, it is still possible to draw some general conclusions from the results obtained.

Although the mote used in the study was the Tmote sky [13] the measurement techniques developed are quite generic and are easily extended to other devices. A regulated 3V power supply was used in place of the battery pack to help reduce error[5]. A resistor (Z_R) was then placed across the ground line of the node in order to allow measurements of the current consumed by the node. A clean trigger was then taken from one of the I/O pins on the mote. This allowed for accurate timing measurements

[5]Absolute accuracy is not a major requirement in a study such as this. However, precautions were taken where possible.

also.

It was necessary that the resistance used be small enough so as not to interfere with the operation of the node itself but large enough to allow the resistor's voltage drop to be easily detectable. Some initial experiments with $5 - 10\Omega$ resistors showed them to be too large leading, at times, to unpredictable results (such as malfunctioning LEDs). However, a precision 1Ω resistor was later found to work remarkably well as it allowed for easily detectable voltage drops while the node operated unaffected.

The resistor was placed in series with the node; while it was quite feasible to connect it to the node's positive supply terminal this would have required two scope probes for the measurement. To simplify matters, the resistor was connected to the node's ground terminal thereby requiring only one probe. The current (i_R) through both resistor and node was determined by measuring the voltage (v_R) across Z_R. Using Kirchoff's voltage law, it follows that:

$$\begin{aligned} V_{cc} &= i_R(Z_R + Z_N) \\ &= i_R(1 + Z_N) \\ &\approx i_R Z_N \end{aligned} \tag{5.10}$$

where Z_N is the impedance of the node. The approximation in the last line of this equation makes the assumption that $Z_N \gg 1\Omega$ which is required in practice due to the low power consumption of the nodes. The power dissipated by the node can therefore be approximated by:

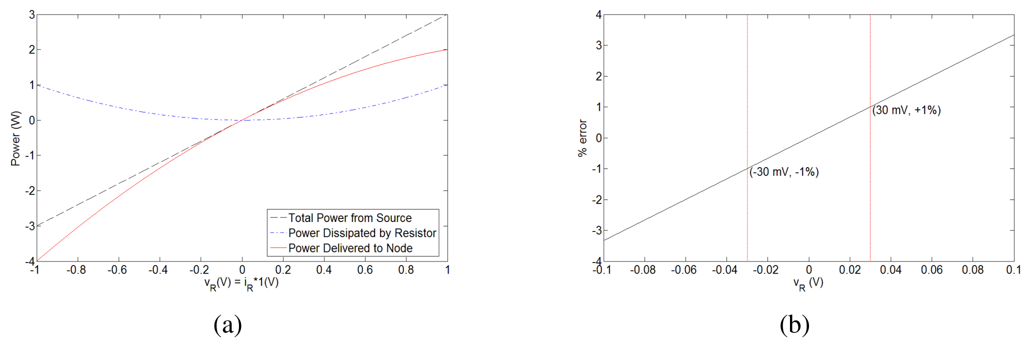


Figure 5.21: a) The power delivered, dissipated or consumed by each component and b) the percentage error of approximating the power dissipated in the node by $V_{cc}i_R$ over different values of v_R.

$$\begin{aligned} P_N &= V_{cc}i_R \\ &\approx i_R^2 Z_N \end{aligned} \tag{5.11}$$

It is interesting to examine the approximation used in equation 5.10 in more detail. The assumption is stating that almost the entire supply falls across the node itself. The power consumed by the node is then $P_N = i_R^2 Z_N = v_N i_R \approx V_{cc}i_R$, where v_N is the voltage drop across the node. The percentage error can therefore be written as follows:

$$\begin{aligned} P_{error} &= 100\frac{V_{cc}i_R - i_R^2 Z_N}{V_{cc}i_R}\% \\ &= 100\frac{V_{cc} - i_R Z_N}{V_{cc}}\% \\ &= 100\frac{v_R}{V_{cc}}\% \end{aligned} \tag{5.12}$$

which shows that the larger the current drawn by the node (and hence the larger the value of v_R) the larger the percentage error will be.

Figure 5.21(a) shows a plot of the power delivered by a 3V supply (as was used in the measurements) against different values of v_R. Also shown in this figure is the power dissipated by Z_R and consumed by Z_N. As $Z_R = 1\Omega$ the values of current and voltage will be the same. As $|v_R|$ increases above about 30mV the approximations described above become less accurate. This can be seen more clearly in figure 5.21(b) where P_{error} is plotted against v_R. Provided $|v_R|$ is kept below about 30mV, P_{error} will be less than 1% which is an acceptable approximation for the measurements at hand. More accurate measurements would require knowledge of v_N also.

5.2.1 Energy Cost of Communications

The energy consumed by both a transmission and a reception over different payload sizes was obtained for the node under test[6]. Arguably, this information could have been calculated using values obtained from the data sheet for the CC2420 transceiver; however, a number of points should be noted. The energy cost of a packet merging could not be obtained in this way and this was the main reason for setting up the experiment; extending the measurements to transmit and receive events was trivial. Also, this method allows for an accurate evalutation of the *system* energy consumption and not just that of a single component. For example, the energy cost of I/O between the microcontroller and the radio can be seen from figure 5.22 between .4ms and 1.9ms (this figure will be explained in more detail shortly). The graphs therefore illustrate nicely exactly what is happening during a transmission/reception, as well as providing accurate information regarding timing and energy consumption.

`PA_LEVEL`	3	7	11	15	19	23	27	31
Output Power (dBm)	-25	-15	-10	-7	-5	-3	-1	0

Table 5.4: Transmission power output levels given in the data sheet [2] for a given register value.

[6]Evaluating the energy cost of a reception is interesting as, in much work in this area, it is considered far less than that of a transmission and is often ignored (see, for example, [114]). This section shows that this is not true for the particular device under test.

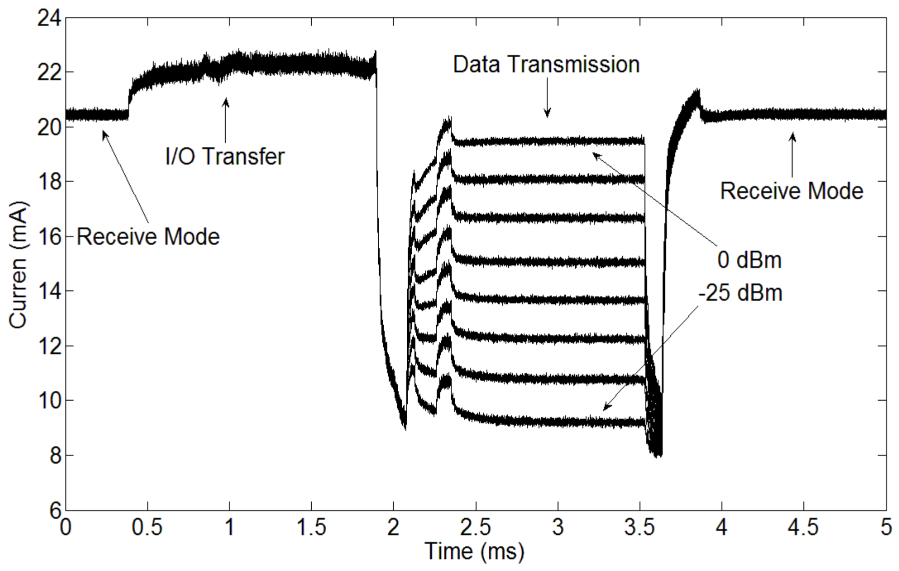


Figure 5.22: The current consumed for a transmit event taken at the 8 different power levels specified in the CC2420 datasheet. Only the values of -25 dBm and 0 dBm are annotated.

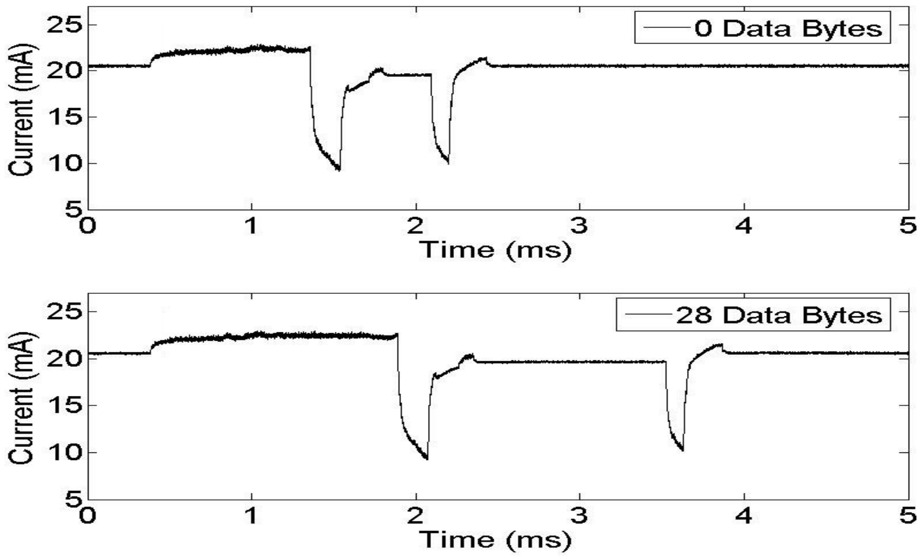


Figure 5.23: Current consumed by two separate transmit events on the Tmote Sky.

For the first measurement, a total of 200 voltage waveforms were collected for each transmission using the maximum TinyOS payload (28 bytes). The waveforms were then averaged to reduce noise. This was done over the eight different power levels

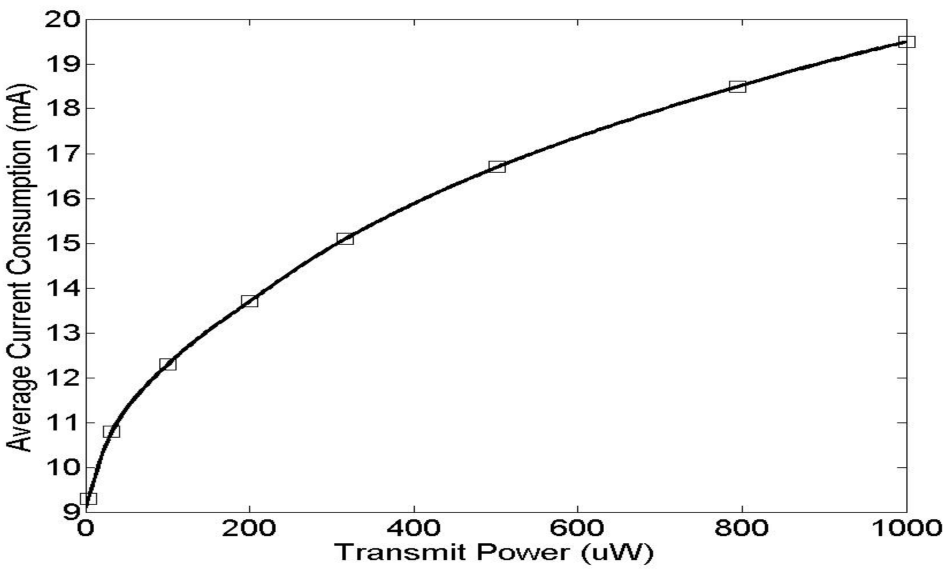


Figure 5.24: Current consumption of the Tmote at different transmission power levels.

specified in the CC2420 datasheet [2]. Table 5.4 shows each power level along with the control register value, `PA_LEVEL`, used to set that level (see section 5.3 for more details). Each packet contained a header of 11 bytes so there was always at least this number of bytes being transmitted/received. The resulting waveforms were then plotted, one on top of the other, as shown in figure 5.22. The following events can be discerned from the plot: idle listening[7] from 0ms to 0.4ms, moving the data from the microcontroller to the radio (i.e. I/O) from 0.4ms to 1.9ms, the transmission event itself from 1.9ms to 3.7ms, and idle listening from 3.7ms to the end of the waveform (the radio switches back to receive mode after the transmission event has completed). The experiment was carried out a second time keeping the transmission power at its maximum level (0 dBm) but varying the payload size from 0 to 28 bytes. Figure 5.23 shows two of the resulting voltage waveforms for illustrative purposes (0 and 28 byte payloads only).

A plot of the current consumption versus transmit power was then evaluated as shown

[7]The radio is in receive mode but not receiving any packets.

in figure 5.24 (the idle listening portions of each waveform were not taken into account). From this last graph it can be seen that if the curve is extrapolated to higher transmission powers, the current consumption will eventually exceed that of a reception (which is approximately 22mA). However, this will never be possible on the CC2420 as its maximum transmission power is 0 dBm. It is clear therefore that a reception constitutes a significant portion of the communications energy cost, for this particular device, and cannot be ignored.

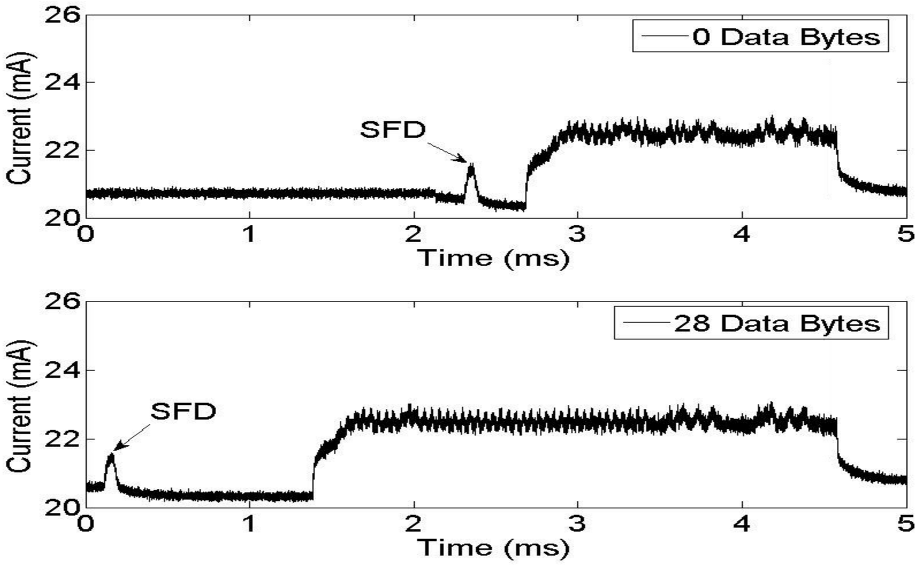


Figure 5.25: Current consumed during the reception a packet with a payload of 0 (top graph) and 28 (bottom graph) bytes.

Similar experiments were carried out for a receive event — the resulting current plot shown in figure 5.25. The Start of Frame Delimiter (SFD) [2] is effectively a pin on the CC2420 transceiver that is set upon detection of a correct preamble[8]. The coupled energy from the SFD pin can clearly be seen in the figure. The waveform to the left of the SFD is due to idle listening and is not part of the current receive event.

A comparison of the energy costs for a transmission and a reception is shown in figure

[8]These are synchronisation bytes at the beginning of each packet.

5.26. The energy consumed for each byte transmitted (E_{txb}) is approximately 3.2μJ at a transmission power of 0 dBm. For each byte received, however, the energy cost (E_{rxb}) is 5.1μJ. This is approximately 400nJ per bit and 638nJ per bit, respectively. Of course for higher transmission powers the cost of a transmission will eventually exceed that of the reception. In some applications it may be necessary to transmit at higher transmission powers [119]. For example in [120] the NRF905 radio [118] is used at a power level of 10 dBm consuming about 693μJ for a 32 byte packet. This is nearly 22μJ per byte, which is about 2.5 times that of a reception on the NRF905 radio.

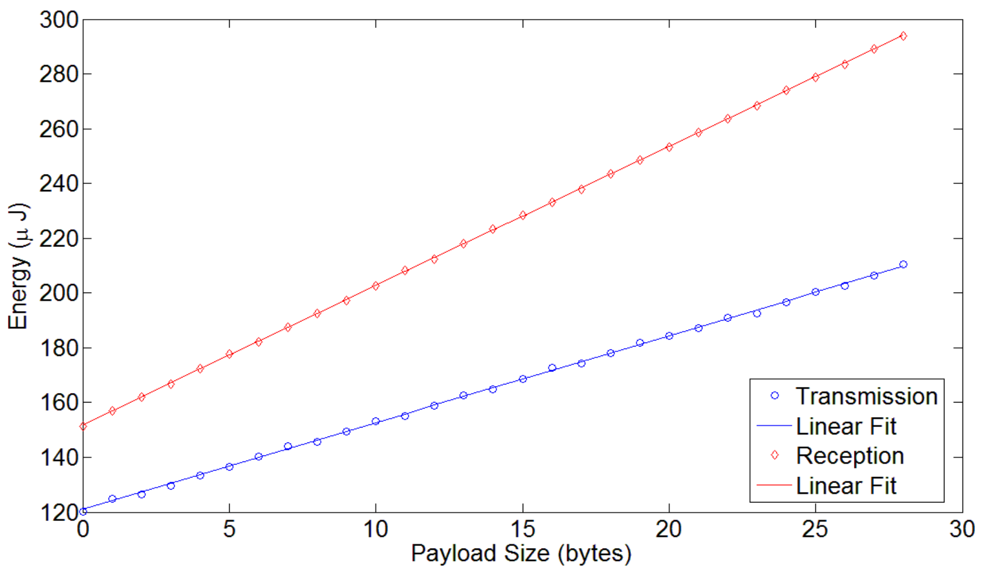


Figure 5.26: The energy consumption for each extra data byte in the *payload*.

5.2.2 Energy Cost of Merging

Energy measurements were carried out while the node was running the merging operation on a standard full search algorithm and iPM. Although the methods described throughout this thesis were implemented to make the search as efficient as possible, no attempts were made at optimising the source code itself. The results obtained from these measurements are therefore by no means the best obtainable. They still offer an idea of the energy savings possible using iPM however.

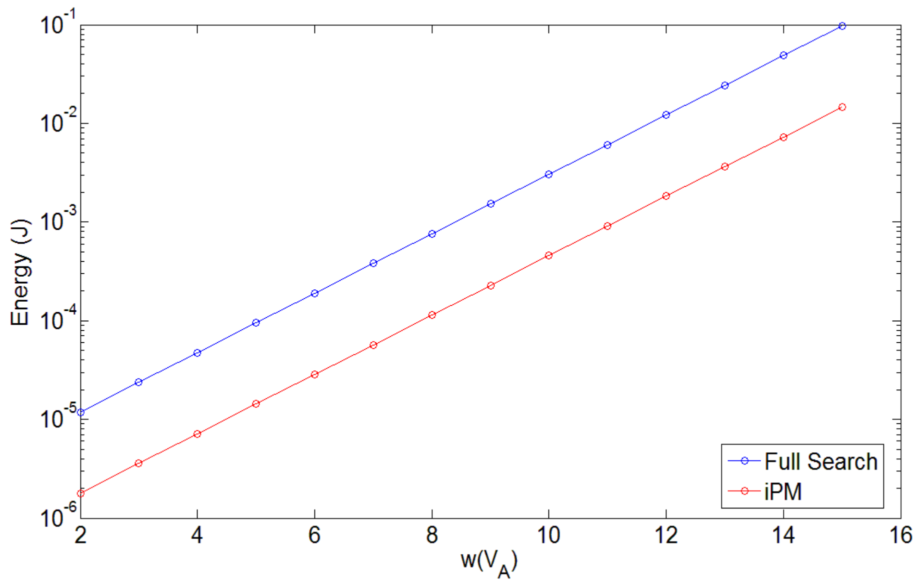


Figure 5.27: Energy consumed using both a full search algorithm and iPM.

Figure 5.27 shows an extrapolation of the energy measurements carried out for each Hamming weight. This graph was produced as follows: the nodes were instructed to perform a full packet merging operation on ambiguity vectors with Hamming weights between 2 and 10 inclusive. An average of 200 voltage waveforms, for each value of Hamming weight, was then collected. The time from the beginning point of the algorithm to the final point was easily identified by the use of an I/O pin on the mote. The energy was then calculated by performing a numeric integration over the product of the supply voltage (3V) and the current waveform obtained. The energy required per iteration of iPM (E_{it}) was quite consistent over all tests and was determined to be, on average, 2.97μJ with a standard deviation of about 1.7×10^{-7}J. The average value was then used to create the graph of figure 5.27 as it allowed extrapolation up to Hamming weights higher than 10 (15 in this case). The energy consumption for iPM was simply taken to be 15% of that of a standard search as, on average, the reduction in the number of iterations afforded by iPM was 85%, as shown in chapter 4.

Using these results an examination of the benefits of iPM can be obtained as follows. Consider a simple case in which two packets are required to be sent and up to two transmissions can be attempted for each packet before it is considered to be in error. Assuming tradARQ initially, the first node transmits a packet and the energy consumed is E_{tx} J. This is received by the second node and the energy consumed is E_{rx} J. An acknowledgement is sent consuming E_{ackt} J. The acknowledgement is then received by the transmitter consuming E_{ackr} J. The total energy for one transmission is therefore:

$$E_{\text{tr}} = E_{tx} + E_{rx} + E_{ackt} + E_{ackr} \text{ J.} \tag{5.13}$$

Each of these values can be determined from figure 5.26. Assuming a full payload (i.e. 28 bytes) transmitting at 0 dBm the value of E_{rx} is 294μJ and that of E_{tx}, 210μJ. Assuming a two-byte acknowledgement the value of E_{ackt} would be 127μJ and that of E_{ackr}, 162μJ. The total energy for one transmission (E_{tr}) is therefore 793μJ; this does not include the energy cost of idle listening. As can be seen from the upper graph in figure 5.25, due to the large current consumption (the portion before the SFD, for example) idle listening can potentially consume a significant amount of energy if time in this mode is not minimised. The different energy measurements are summarised in table 5.5.

E_{txb}	E_{rxb}	E_{tx}	E_{rx}
3.2	5.1	210	294

E_{it}	E_{ackt}	E_{ackr}	E_{tr}
2.97	127	162	793

Table 5.5: Energy Consumption (μJ) for the different parameters measured. Transmission power is set to 0 dBm and payload to 28 bytes.

Assume now that two packets have already been transmitted and that both are incorrect. The decision to be made is whether to ask for a second retransmission or to perform a merging operation. Figures 5.18 and 5.11 show that the reduction in the required SNR for a given PER is about .31 dB when sending three packets and about 1.84 dB when using a merging operation (for a packet size of 320-bits and $w(V_A)_{max} = 10$). These reductions are both with respect to a tradARQ system using only one retransmission. Assume for the moment, however, that the PERs are equivalent in both cases. The question then becomes: which operation consumes less energy?

If the receiver decides to ask for a retransmission it must first send a NACK which consumes E_{ackt} J of energy. This is received by the transmitter, a process that consumes E_{ackr} J of energy. The transmitter then retransmits the packet which involves the consumption of E_{tx} J of energy. Finally, this is then received by the receiver consuming E_{rx} J of energy. The total energy consumed by an extra transmission is therefore the same as equation 5.13, i.e. 793μJ.

If the receiver decides to perform a merging operation it must choose an appropriate value of $w(V_A)_{max}$. When $w(V_A)_{max}$ is set to 15 then the energy for a packet merging operation is 97.3mJ for a full search and 14.6mJ using iPM. It is clear therefore that with the unoptimised code considered in this case, setting $w(V_A)_{max}$ to 15 is not cost effective, in terms of energy. However, if $w(V_A)_{max}$ is set to 10 and iPM is used the energy consumption in this case is only 456μJ. Using the standard brute-force search $w(V_A)_{max}$ would have to be set to a value of about 8, although even then it still consumes far more energy than iPM at 760μJ.

	Retransmission	Standard merge	iPM	iPM
$w(V_A)_{max}$	-	8	8	10
Energy Consumed	793μJ	760μJ	114μJ	$456\mu J$
Gain achieved	.31 dB	1.5 dB	1.5 dB	1.84 dB

Table 5.6: Comparison of energy consumption and gains for a retransmission, a standard packet merge and iPM.

Further motivation to use packet merging can be obtained if the significant difference in the SNR gains between a retransmission and a merging operation is taken into account. For a reduction in the amount of energy required (i.e. from 793μJ for a retransmission compared with 456μJ to perform a merging using iPM) and an extra gain of (1.84 - .31 = 1.53) dB is also obtained. With regard to the different packet merging approaches it is clear that iPM outperforms a traditional brute-force approach and should therefore be used. If the value of $w(V_A)_{max}$ is set to 8 for the brute-force approach the energy cost is approximately 760μJ with a gain of 1.5 dB (cf. figure 5.17). Using iPM with $w(V_A)_{max} = 10$ the energy is *reduced* to 456μJ and gain *increased* to 1.84 dB. However, it is also possible to use iPM with the *same* value of $w(V_A)_{max}$ possible with a standard merge (i.e. 8). In this case the gain is the same (1.5 dB) however the energy consumption is only 15$\%$ of 760μJ, i.e. 114μJ. For the cooperative scenario the extra reception required by the cooperating node makes merging even more advantageous, assuming a cooperative system is already in place. Table 5.6 summarises these results for the tradARQ case.

As mentioned earlier, the code used when obtaining these measurements was far from optimised. Code optimisation would likely offer significant improvements and hardware implementations even more so. It is not possible to say exactly how much improvement is possible using a hardware implementation of the algorithm but in some cases hardware implementations of algorithms have been shown to offer up to 10,000 times a reduction in energy costs over software [121]. This is highly dependent on the nature of the algorithm and whether or not it can be easily implemented in hardware. However, iPM is a simple iterative technique and might easily benefit in this respect. This would represent very interesting future work on these algorithms.

To motivate the idea of code optimisation it was noted in [18] that the cost of a merging operation with $w(V_A)_{max} = 6$ is roughly only 7$\%$ of that of a transmission with a 29 byte payload and 2% with a 128 byte payload. These measurements were carried out on a CC1000 radio operating at a frequency of 433MHz and a power level of -15 dBm. The system therefore used less power than the CC2420 at a transmission power of 0 dBm [122], [123] and the latter can therefore be used as a loose reference. Using figures 5.26 and 5.27, therefore, the cost of a transmission for a 29 byte payload would be 297.4μJ. The energy cost of a merging operation using the values quoted in [18] would therefore be approximately 21μJ. The measurements shown in figure 5.27 show the energy cost for the same operation at almost an order of magnitude larger (i.e. 192μJ at $w(V_A)_{max} = 6$).

If idle listening is now taken into account the situation changes somewhat. Each time the radio finishes transmitting it must wait a certain period of time for a response. Assuming time of flight from transmitter to receiver is negligible the length of time the receiver must wait after the NACK is sent is equal to the time spent receiving the NACK at the transmitter. For CSMA based networks there is also the likelihood of not being able to transmit immediately. A random backoff timer is then used adding another $E_{rb}\mu$ J of energy, the value of which increases the more transmissions that are occurring in the network. This means that a larger value of $w(V_A)_{max}$ now becomes feasible with iPM and the current non-optimised code. Coupled with the fact that the reduction in PER by retransmission is very small (cf. figure 5.18) iPM is certainly a more beneficial option.

To motivate the situation further, in multiple access networks interference and collisions are a major issue in many cases. By choosing to combine rather than retransmit increased levels of interference and collisions can be avoided. The less channel use taken up by retransmitted packets the more the network is free to transmit new data and the less probable collisions will be. This in itself represents interesting future work as it will save energy and increase throughput.

5.3 Practical Results

An experiment was setup up in order to determine the performance of iPM in practice. Each of the four situations described in section 5.1 were considered (i.e. tradARQ, tradARQC, coopARQ, coopARQC) and the value of $w(V_A)_{max}$ set to values of 10 and 15, as was emphasised for the simulations. A number of issues existed in trying to obtain PER curves in the practical experiments. Most significantly, it was very difficult to obtain an accurate estimation of the received SNR, as noted in chapter 3. Instead, the PER was plotted against the transmission power rather than $\bar{\gamma}$.

In order to emulate the slow Rayleigh fading channel assumed in the simulations and analytical model the receiver was placed on a record player which was set to revolve at 33 1/3 rpm. The transmit node was placed about 10m away and remained at this distance for each different transmit power level used. The large distance between the transmitter and the center of the record player meant that even though the receiver was moving relative to the transmitter the affect on the average received SNR was minimal. The antennas on the Tmote sky nodes are omnidirectional which allowed the node to be rotated in this manner. To compensate for the periodic nature of the rotation a random timer was used on the transmitter for each transmission.

The choice of transmit power levels to use invited some challenges. On the CC2420 transceiver only a small number of transmission power levels are permissible — according to the data sheet, only 8 distinct values are possible [2]. The information, however, is rather ambiguous in this regard as the control register used to set the transmission power level accepts values from 0 to 31 (5-bits). Table 5.4 shows the transmission power output levels provided by the CC2420 data sheet [2]. The value `PA_LEVEL` represents the lower 5-bits of the register and not its full 16-bit contents.

It was not clear from the data sheet whether or not setting the register to other values would have any effect on the output power. To help clarify this point a simple experi-

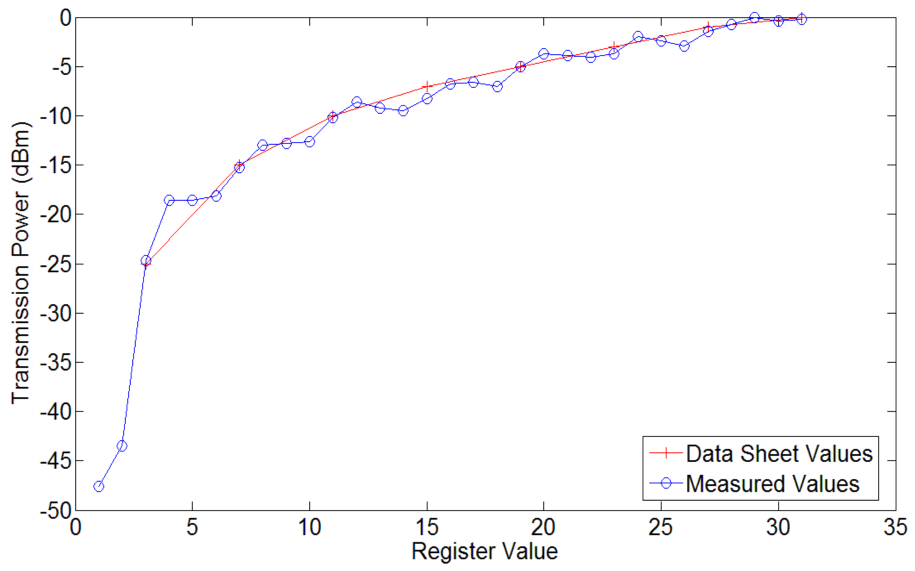


Figure 5.28: Power levels obtained experimentally for each register value on the CC2420. The values given in the data sheet are plotted for comparison.

ment was set up where a Tmote sky was placed beside a spectrum analyser fitted with a 2.4GHz antenna. The mote was then set to transmit 5000 packets in quick succession with the control register value set to 0. The spectrum analyser returned the peak average output power over the 5000 packets. The register value on the mote was then incremented by 1 and the procedure repeated. This process was continued for each register value up to 31. The entire experiment was automated and the setup was left unattended until completion so as to avoid any interference.

A plot of the results is shown in figure 5.28. Each value is shifted by a constant amount to account for path loss and other factors (the largest value obtained in the experiment was set equal to 0 dBm and the remaining values shifted accordingly). It can be clearly seen that the transmission power levels specified in the data sheet are not the only ones possible. It would also appear, however, that not all register values produce changes in the transmission power level. A `PA_LEVEL` value of 1 results in an output power of approximately -48 dBm[9]. A register value of 2 then increases this level by about 4 dB. With `PA_LEVEL` set to 3 a value of -24.73 dBm is obtained which is very close to the value provided in the data sheet (-25 dBm). Although a `PA_LEVEL` of 4 increases the transmission power by approximately 6 dB again, register values of 5 and 6 appear to offer no increase at all. A similar trend is seen for other values of `PA_LEVEL`. As a result of the ambiguity in the data sheet and because the PER experiments required a more fine-grained control of the output power, the values obtained in this experiment were used instead of those in the data sheet.

`PA_LEVEL`	3	4	7	8	11	12	16
Output Power (dBm)	-24.73	-18.55	-15.27	-13.01	-10.22	-8.63	-6.8
Total no. of packets sent ($\times 10^6$)	1	1.6	2.2	4.6	6.4	8.0	8.5

Table 5.7: Subset of transmission power output levels determined practically that were used in the PER experiments. The total number of packets sent for each transmission power is also shown.

In the simulations the PER was obtained over SNR values ranging from 0 to 20 dB, in 1 dB increments. This was not possible for the practical experiments. In order to obtain a sufficient number of incorrect packets it was necessary to set the receiver node on the fringe of its radio range for the lowest transmission power used. This allowed incorrect packets to be obtained at higher power levels with a reasonable number of transmissions. Resolution was the main issue in choosing the lowest power to use. There is almost a 20 dB increase in output power when `PA_LEVEL` is increased from 2 to 3. This is equal to the *full* SNR range used in the simulations. The difference in transmission powers was smaller at higher transmit powers. However, the highest transmission power available (0 dBm) was used for the acknowledgements to ensure they were received correctly. If the nodes were set up in such a way as to obtain errors in this power range then the acknowledgements would also contain errors. A

[9]When set to 0 the output power takes on its maximum value and is therefore not shown.

tradeoff was therefore required as to which power levels to use. The values chosen are shown in table 5.7 and offered the best trade-off between fine-grained control and correct acknowledgements. This table also shows the total number of packets sent, over multiple trials, for each transmission power.

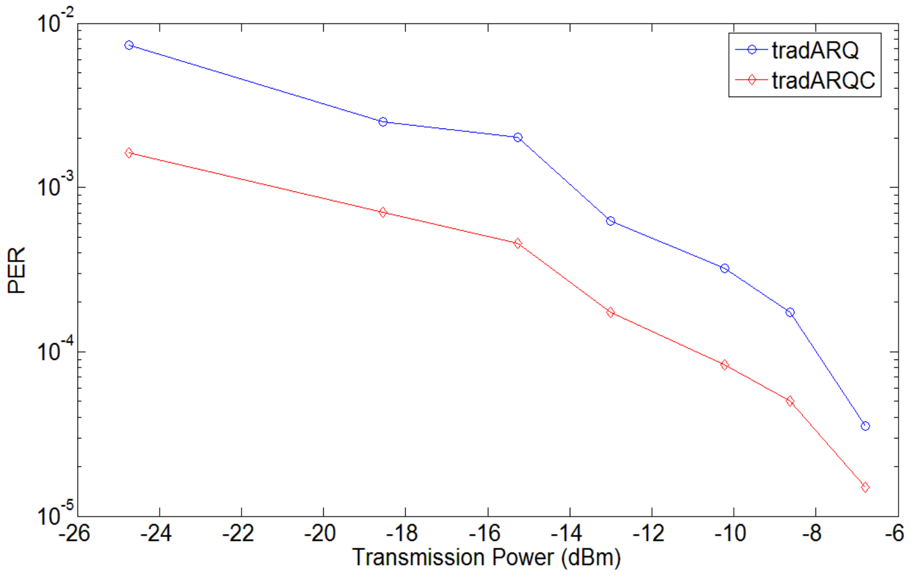


Figure 5.29: Practical results. These results were taken with two nodes only, i.e. tradARQ(C), with $w(V_A)_{max} = 10$ and a source packet size of 216 bits.

The results for the traditional ARQ case (using only a source and destination node) with $w(V_A)_{max} = 10$ and $w(V_A)_{max} = 15$ are shown in figures 5.29 and 5.30 respectively. These results were obtained using a source packet of 216 bits in length. As is generally the case, the practical results proved a lot more difficult to obtain than the simulated values. The main issue was the extremely large number of packets required at high transmission powers to produce meaningful PER values (cf. table 5.7). This meant that the experiment had to be run over the course of a number of days and required several changes of batteries during this time. Although the nodes were setup carefully unavoidable changes occurred within the surrounding environment over this period.

Another issue with the practical setup was that the limited number of usable transmis-

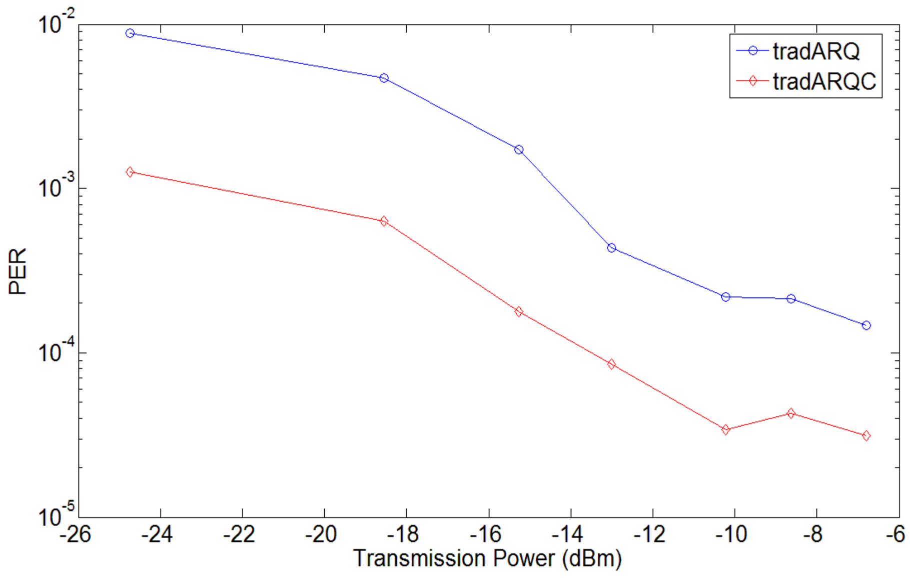


Figure 5.30: Practical results. These results were taken using two nodes only, i.e. tradARQ(C), with $w(V_A)_{max} = 15$ and a source packet size of 216 bits.

sion powers meant it was difficult to obtain smooth curves. This made estimating the SNR gains rather error prone. To help quantify the increase in effectiveness for larger values of $w(V_A)_{max}$, rather than using SNR gain therefore, it is easier to consider the average ratio of the PER values at each transmission power. For $w(V_A)_{max} = 10$ the mean ratio of the PER values is approximately 3.7 whereas for $w(V_A)_{max} = 15$ it is 6.5, clearly showing the benefits of using larger values of $w(V_A)_{max}$. These experiments were carried out using only one generator polynomial for the CRC. This inevitably introduced false-positives into the results. For the larger size of $w(V_A)_{max} = 15$ the percentage of false-positives obtained was on the order of about 1.2%. Naturally, most of these occurred at the lower transmission powers with very few, if any, occurring at upper transmission powers. It should be kept in mind, however, that in a well designed network the average SNRs at the receiving node will, more often than not, be closer to those produced by the higher transmission power levels in this experiment. Regardless, using two generator polynomials would effectively eliminate the false-positive problem, as was shown in section 4.7.2.

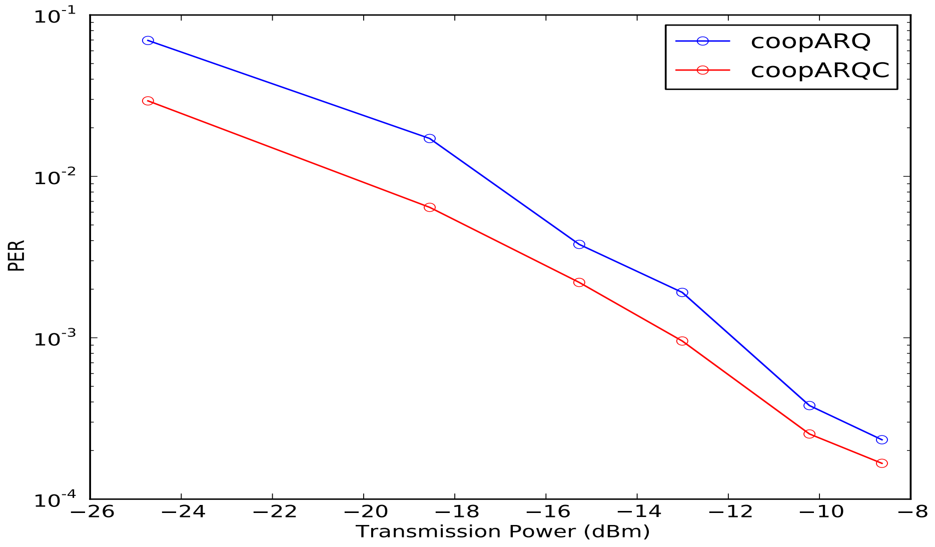


Figure 5.31: Practical results. These results were taken using three nodes, i.e. coopARQ(C), with $w(V_A)_{max} = 10$ and a source packet size of 320 bits.

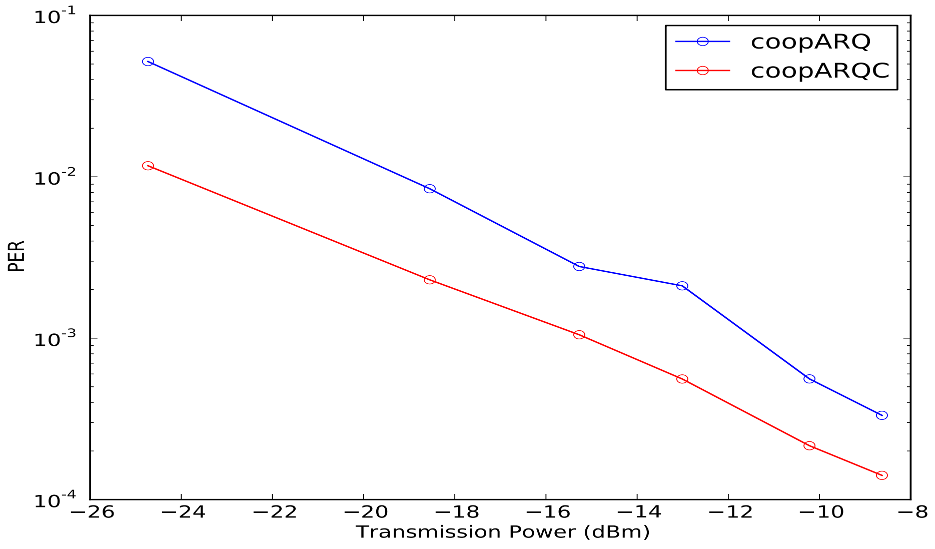


Figure 5.32: Practical results. These results were taken using three nodes, i.e. coopARQ(C), with $w(V_A)_{max} = 15$ and a source packet size of 320 bits.

The results for the cooperative ARQ case (using a source, relay and destination node) with $w(V_A)_{max} = 10$ and $w(V_A)_{max} = 15$ are shown in figures 5.31 and 5.32 respectively[10]. These results were taken with the source packet set to 320 bits in length. Again the ratio of PERs for each transmission power was used as a metric of success. For the first case, packet combining reduced the PER by an average factor of 1.94. For the second case packet combining reduced it by an average factor of 3.2. The lower relative values to the traditional case are to be expected due, to some extent, to the longer packet size. More significant, however, is that the setup emulated a cooperative system with a very good source-relay channel. This type of system was shown to produce a smaller combining gain as diversity becomes more significant. The values produced are still quite significant and clearly show the benefits of using packet combining in both cooperative and non-cooperative scenarios.

5.4 Conclusions

An analysis of the performance of iPM (and packet merging in general) has been carried out in this chapter. By employing analytical equations for maximal-ratio combining, the significance of the SNR gains obtainable when using packet merging were determined. The gains were seen to be quite large considering the relative simplicity of the technique. Maximal-ratio combining is a powerful and complex scheme that is widely used in wireless communication systems [11]. However, it requires special hardware and knowledge of the channel for maximum effectiveness. This hardware can consume significant amounts of energy [91] and is therefore not implemented on typical low power WSN devices.

The iPM technique can be used when MRC is not available. While not intending to be a replacement for MRC it does offer quite significant gains considering its simplicity. It will be most effective when used in conjunction with other forms of combining such

[10]Due to the difficulty in obtaining these results the highest power level was omitted as this did not significantly effect the results.

as the “decoding” method described in [18]. However, if simplicity is required it can also be used as a stand-alone mechanism and still offers beneficial gains. In this latter case packet merging is most effective after a first retransmission. Beyond this it is best used in conjunction with majority voting. Some systems don’t use majority voting for even numbers of transmissions [4] so packet merging is also an option in these cases.

In terms of the gains obtainable it was shown that packet merging approaches that of maximal ratio combining as the value of $w(V_A)_{max}$ increases. This is an interesting result as it implies that highly optimised code or hardware implementations may make these large gains more feasible. This represents interesting future work in this area.

The practical experiments carried out used a specific type of architecture. Each situation contains its own idiosyncrasies that need to be factored in to the system design making it difficult to draw general conclusions. However, it was shown that, for the specific device under test, packet merging was a beneficial combining technique to use. This was done through energy measurements of the node whilst running the algorithm developed and comparing the energy cost to that of a communication. The practical experiments also consisted of a real implementation of the algorithm. It was shown that significant reduction in PERs are possible when using packet merging.

Using packet merging will not only save energy in simple setups such as the ones described in this thesis. In more complex protocols that sense the channel to determine when to transmit, collisions very likely. By choosing to perform a merging operation rather a retransmission the number of collisions and energy wasted from transmission backoffs can be reduced significantly.

Chapter 6

Conclusions and Future Work

A highly improved implementation of a simple packet combining scheme referred to as *improved Packet Merging (iPM)* has been developed in this thesis. It has been shown that even though complex combining schemes such as maximal-ratio combining are not possible on resource-constrained wireless sensor networks, it is still possible to achieve quite significant performance gains using the packet combining scheme developed.

To motivate the need for a packet combining scheme a characterisation of the *transitional region* using a spread-spectrum radio was carried out. It was suggested in [22] that the use of a spread-spectrum architecture may help to reduce the extent of this region, possibly replacing the need for combining techniques. This thesis showed, through practical experimentation, that this region can still have significant extent, even with the use of a spread-spectrum architecture. The results obtained motivate the need for techniques such as iPM within this region. The experiments made use of the Received Signal Strength Indicator (RSSI) as an estimate of the Signal-to-Noise Ratio (SNR). The results suggested that RSSI can over-estimate the SNR if the level of interference is too large. Future work in this area might be to carefully examine the effects of a controllable source of interference on the effects of RSSI and to determine whether it is possible to estimate this interference in a real situation. In doing so a better estimate of the SNR may be possible. Other possible work, related to the transitional zone in general, might be to use different spread-spectrum architectures in which larger spreading bandwidths are used.

In chapter 4 two algorithms were developed both of which used a pair of packets, received over independent fading channels, for error correction. The first algorithm, referred to as Simple Packet Merging (sPM), chooses one channel, *a priori*, and always uses the packets from that channel in the error correction process. It also implements a simplistic search procedure for the correct packet by running through all possible error candidates in an incremental fashion. The second algorithm, referred to as Improved Packet Merging (iPM), not only intelligently chooses, in real time, which packet has less errors, but also carries out a more efficient search for the correct packet. It was shown that iPM improved upon the average number of iterations required for sPM by up to 21%. This value was then reduced further by using the parity equivalence technique where it was observed that the parity of the Hamming weight of the message was equivalent to that of the Frame Check Sequence (FCS) accompanying the message. Use of this particular technique was shown to reduce the required number of searches by one-half. Experiments then showed that by using the better of the two algorithms developed (iPM) more than an 85% reduction in the required number of iterations compared with a standard brute-force search was obtainable. The trade-off using this method, however, was a slightly increased probability of false-positives. For the packet size used, the number of false positives occurring was shown to be quite small when using iPM in its basic form. Even still, to satisfy the needs of more data critical applications a technique was developed whereby a second generator polynomial was used with the original packet's pair. This technique was shown to effectively eliminate false positives.

In iPM a packet selection metric such as RSSI/LQI is used for packet selection. It was noted that future chip designs may provide a means of sampling the RSSI/LQI for each symbol of the incoming packet. It is very likely that this would significantly improve the selection policy of iPM as the current algorithm relies on one sample only (an average RSSI value taken over the first 8 symbol periods of the packet) to provide accurate information about the rest of the packet. It is very likely that this will not only increase the likelihood of choosing the packet with the least number of errors but also, if the bits are then arranged in order of decreasing confidence, reduce the number of false-positives and increase the search efficiency. This also represents interesting future work.

In chapter 5 an analysis of packet merging was carried using simulations and practical experimentation. By employing an analytical technique for maximal-ratio combining, a comparison of the SNR gains obtainable using packet merging were determined. It was noted on a number of occasions that iPM is not intended as a replacement for MRC but as a technique to be used when MRC is not available. As a standalone device iPM was shown to offer quite significant gains considering its simplicity. It was suggested, however, that for maximum benefit iPM could be used in conjunction with other forms of combining, such as the packet decoding technique described in [18].

Packet merging, in general, was shown to be most effective after a first retransmission, as techniques such as majority voting are not available to use in this case. Potentially, iPM can obtain up to 2.47 dB gain (with a packet size of 320 bits) when only one retransmission is permitted. The suggestion was made that highly optimised code or hardware implementations may allow these gains to be obtained and represents interesting future work in this area. When more than one retransmission is used, iPM is still of benefit once the first retransmission has been received. Subsequent retransmissions provide the opportunity to do a majority vote on the received packets. Packet merging should be used in conjunction with majority voting for most benefit. Majority voting is most effective when an odd number of packets are available. It is possible that packet merging may be of more benefit when only an even number of packets are available.

Again, this represents interesting future work. Combining packet merging with a majority voting scheme offers potential gains of up to 2.72 dB (with a packet size of 320 bits) although current practical gains are approximately 2.52 dB, which is still quite significant. Merging was shown to be more effective strategy for smaller packet sizes. Currently many WSN implementations use packets of a size similar to the size used in the experiments. For larger packet sizes, on the order of the 1080 bit packets studied in this thesis, an interesting piece of future work would be to look at the possibility of splitting the application payload up into two, with a different FCS values for each half. In this way less undetected CRC errors would occur as a result of using a longer packet and the packet combining scheme would be more effective. The tradeoffs in energy/throughput when doing this would be an interesting study.

The practical experiments carried out used a specific type of architecture. However, it was shown that, for the specific device under test, packet merging was a beneficial combining technique to use. This was done through energy measurements of the node whilst running the algorithm developed and comparing the energy cost to that of a communication. The practical experiments also consisted of a real implementation of the algorithm. It was shown that significant reduction in PERs are possible when using packet merging. Although it was difficult to measure the actual gains obtainable in the practical experiments iPM was shown to reduce the PER by a factor of 3.7 when $w(V_A)_{max} = 10$ and 6.5 when $w(V_A)_{max} = 15$. This was an average value over different transmission powers.

The experimental setups were sufficiently representative of real deployment scenarios to give accurate results. The receiving node was always placed on the fringe of its radio range, in a highly cluttered environment, with plenty of surrounding movement. To ensure sufficient fading the nodes were then placed on a record player rotating at 33 1/3 rpm with randomly timed packet transmissions for maximum effect. Although these experiments do not cover all possible scenarios, such as nodes placed in a rainforest, it is not envisaged that any major difference will be perceived by the packet combining technique in these environments (with the exception of possibly no reception at all, in which case packet combining won't work anyway).

Choosing to carry out a packet merging procedure over a retransmission was shown to significantly improve the performance of a system, not only consuming less energy, but also by providing more than five times the SNR gain. It was noted that in multiple access networks interference and collisions are a major issue in many cases. By choosing to combine rather than retransmit increased levels of interference and collisions can be avoided. The less channel use taken up by retransmitted packets the more the network is free to transmit new data and the less probable collisions will be. The performance of packet merging from a network -wide perspective represents interesting future work in this regard as it will save energy and increase throughput.

From an applications perspective packet combining is very versatile and can be applied in most situations. However, it seems to be very well suited to underwater networks. Due to the difficult nature of the conditions and low-bandwidths available it is very likely combining will represent a significant system-wide improvement over retransmission. The exact savings in energy or increases in throughput represent very interesting future work.

In terms of hidden errors, it may be possible to deal with them by considering the correlation values returned by the demodulator. These can then be used to determine the likelihood that a hidden error occurred for a particular bit position. The bits could then be ordered in decreasing order of confidence and the search started from the appropriate end; this should possibly provide a further reduction in the total number of iterations required.

Finally, as noted in the introduction, this work makes the following main contributions to the field:

- An improved *practical* implementation of a packet combining scheme based on the Cyclic Redundancy Check has been developed in this thesis.
- This is shown to have a positive effect on the energy consumption of a wireless node as it lowers the number of transmissions required to achieve a given packet error rate
- It is shown that the improved packet combining scheme developed offers more than an 85% reduction in the number of iterations required to find the correct packet over current implementations. This leads to further significant energy savings.
- The use of a second generator polynomial has been shown to be of benefit in this scheme offering improved performance through the reduction of false-positives.
- A thorough energy analysis has been carried out for a particular type of wireless node.
- The benefits of packet merging in both cooperative and non-cooperative scenarios have been shown practically and in simulations, for different packet sizes.
- A motivating experimental study showed that, under certain circumstances, the transitional region still has significant extent even with the use of a spread-spectrum architecture.

Bibliography

[1] E. Callaway, P. Gorday, L. Hester, J. A. Gutierrez, M. Naeve, B. Heile, and V. Bahl, "Home networking with IEEE 802.15.4: A developing standard for low-rate wireless personal area networks," *IEEE Communications Magazine*, vol. 40, pp. 70–77, August 2002.

[2] Texas Instruments, *CC2420 Datasheet*. Available from: http://focus.ti.com/docs/prod/folders/print/cc2420.html [Accessed 2nd March 2007].

[3] T. Ramabadran and S. Gaitonde, "A tutorial on CRC computations," *IEEE Micro*, vol. 8, pp. 62–75, August 1988.

[4] T. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Wiley-Interscience, 2005.

[5] Institute of Electrical and Electronic Engineers, *IEEE 802.15.4 Std.*, 2006.

[6] *CRC Code for TinyOS*. Available from http://www.tinyos.net/tinyos-2.x/tos/system/crc.h [Accessed 05 September 2008].

[7] J. Paradiso and T. Starner, "Energy scavenging for mobile and wireless electronics," *IEEE Pervasive Computing*, vol. 4, pp. 18–27, Jan-March 2005.

[8] *Energy Harvesting White Paper*. Available from http://www.sentilla.com [Accessed 20 October 2008].

[9] S. Wicker, *Error Control Systems for Digital Communication and Storage*. Prentice Hall, 1994.

[10] A. Nosratinia, T. E. Hunter, and A. Hedayat, "Cooperative communication in wireless networks," *IEEE Communications Magazine*, vol. 42, pp. 74–80, October 2004.

[11] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[12] D. Varshney, C. Arumugam, V. Vijayaraghavan, N. Vijay, and S. Srikanth, "Space-time codes in wireless communications," *IEEE Potentials*, vol. 22, pp. 36–38, August-September 2003.

[13] Moteiv Corporation, *Tmote Sky Datasheet*. Available from http://www.moteiv.com [Accessed 27 September 2005].

[14] Crossbow Technology Inc. Available from http://www.xbow.com/ [Accessed 08 April 2005].

[15] J. Kahn, R. Katz, and K. Pister, "Next century challenges: Mobile networking for smart dust," in *MobiCom '99: Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 483–492, 1999.

[16] P. Sindhu, "Retransmission error control with memory," *IEEE Transactions on Communications*, vol. 25, pp. 473–479, May 1977.

[17] S. S. Chakraborty, E. Yli-Juuti, and M. Liinaharja, "An ARQ scheme with packet combining," *IEEE Communications Letters*, vol. 2, pp. 200–202, July 1998.

[18] H. Dubois-Ferrière, D. Estrin, and M. Vetterli, "Packet combining in sensor networks," in *SenSys '05*, November 2005.

[19] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, E. D., and S. Wicker, "Complex behavior at scale: An experimental study of low-power wireless sensor networks," Technical Report UCLA/CSD-TR 02-0013, UCLA CS, 2002.

[20] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pp. 1–13, 2003.

[21] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges for reliable multihop routing in sensor networks," in *ACM Sensys '03*, pp. 14–27, 2003.

[22] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links," in *SECON '04: First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, pp. 517–526, 2004.

[23] A. Cerpa, J. Wong, M. Potkonjak, and D. Estrin, "Temporal properties of low power wireless links: Modelling and implications on multi-hop routing," in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 414–425, 2005.

[24] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, pp. 102–114, August 2002.

[25] K. Ong, C. Grimes, C. Robbines, and R. Singh, "Design and application of a wireless, passive, resonant-circuit environmental monitoring sensor," *Sensors and Actuators A: Physical*, vol. 93, pp. 33–43, September 2001.

[26] A. Benbasat, S. Morris, and J. Paradiso, “A wireless modular sensor architecture and its application in on-shoe gait analysis,” in *Proceedings of IEEE Sensors*, vol. 2, pp. 1086–1091, October 2003.

[27] T. Wark, M. Karunanithi, and W. Chan, “A framework for linking gait characteristics of patients with accelerations of the waist,” in *Engineering in Medicine and Biology 27th Annual Conference*, pp. 7695–7698, September 2005.

[28] Y. Zhang, L. Ackerson, D. Duff, C. Eldershaw, and M. Yim, “Stam: A system of tracking and mapping in real environments,” *IEEE Wireless Communications*, vol. 11, pp. 87–96, December 2004.

[29] P. Sikka, P. Corke, and L. Overs, “Wireless sensor devices for animal tracking and control,” in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, vol. 1, pp. 446–454, November 2004.

[30] J. Burrell, T. Brooke, and B. R., “Vineyard computing: Sensor networks in agricultural production,” *Pervasive Computing*, vol. 3, no. 1, pp. 38–45, 2004.

[31] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, “Wireless ad-hoc sensor and actuator networks on the farm,” in *IPSN ‘06: The Fifth International Conference on Information Processing in Sensor Networks*, pp. 492–499, April 2006.

[32] T. Wark, P. Corke, P. Sikka, L. Klingbeil, Y. Guo, C. Crossman, P. Valencia, D. Swain, and G. Bishop-Hurley, “Transforming agriculture through pervasive wireless sensor networks,” *IEEE Pervasive Computing*, vol. 6, pp. 50–57, April-June 2007.

[33] *Riga Development, “WiSuite - The Freedom and Simplicity of Wireless Control”*. Available from http://www.wisuite.com/ [Accessed 05 September 2008].

[34] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, “A wireless sensor network for structural monitoring,” in *SenSys ‘04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 13–24, 2004.

[35] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, “Wireless sensor networks for habitat monitoring,” in *WSNA ‘02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pp. 88–97, September 2002.

[36] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, “Habitat monitoring with sensor networks,” in *Communications of the ACM*, vol. 47, pp. 34–40, June 2004.

[37] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *ACM Sensys '05*, (San Diego, California, USA), pp. 51–63, November 2005.

[38] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 81–94, 2004.

[39] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh, "Monitoring volcanic eruptions with a wireless sensor network," in *Proceedings of the Second European Workshop on Wireless Sensor Networks*, pp. 108–120, January-February 2005.

[40] X. Hong, M. Gerla, R. Bagrodia, T. J. Kwon, P. Estabrook, and G. Pei, "The Mars sensor network: efficient, energy aware communications," *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE*, vol. 1, pp. 418–422, 2001.

[41] H. Kulah and K. Najafi, "Energy scavenging from low-frequency vibrations by using frequency up-conversion for wireless sensor applications," *IEEE Sensors Journal*, vol. 8, pp. 261–268, March 2008.

[42] S. Chalasani and J. Conrad, "A survey of energy harvesting sources for embedded systems," in *IEEE SoutheastCon*, pp. 442–447, April 2008.

[43] P. Corke, P. Valencia, P. Sikka, T. Wark, and L. Overs, "Long-duration solar-powered wireless sensor networks," in *EmNets '07*, pp. 33–37, June 2007.

[44] A. N. Knaian and J. A. Paradiso, "Wireless roadway monitoring system." US Patent No. 6,662,099, Dec. 9, 2003.

[45] C. Enz, N. Scolari, and U. Yodprasit, "Ultra low-power radio design for wireless sensor networks," in *IEEE International Workshop on Radio-Frequency Integration Technology*, pp. 1–17, Nov-Dec 2005.

[46] I. Demirkol, C. Ersoy, and F. Alagöz, "Mac protocols for wireless sensor networks: A survey," *IEEE communications magazine*, vol. 44, pp. 115–121, April 2006.

[47] S. Zhou, R. Ping Liu, and Y. Guo, "Energy efficient networking protocols for wireless sensor networks," in *IEEE International Conference on Industrial Informatics*, pp. 1006–1011, August 2006.

[48] J. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter, and B. Heile, “IEEE 802.15.4: A developing standard for low-power low-cost wireless personal area networks,” *IEEE Network*, vol. 15, pp. 12–19, Sept/Oct 2001.

[49] A. Goldsmith and S. Wicker, “Design challenges for energy constrained ad-hoc wireless networks,” *IEEE Wireless Communications*, vol. 9, pp. 8–27, August 2002.

[50] K. Schwieger and G. Fettweis, “Power and energy consumption for multi-hop protocols: A sensor network point of view,” in *IWWAN ‘05: International Workshop on Wireless Ad-Hoc Networks*, May 2005.

[51] R. Min and A. Chandrakasan, “Mobicom poster: top five myths about the energy consumption of wireless communication,” *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 1, pp. 65–67, 2003.

[52] W. R. Heinzelman, C. A, and B. H, “Energy-efficient communication protocol for wireless microsensor networks,” in *Proceedings of the 33rd Hawaii International Conference on System Sciences*, January 2000.

[53] M. Vuran and I. Akyildiz, “Cross-layer analysis of error control in wireless sensor networks,” in *SECON ‘06: Sensor and Ad Hoc Communications and Networks*, vol. 2, pp. 585–594, September 2006.

[54] V. Srivastava and M. Motani, “Cross-layer design: A survey and the road ahead,” *IEEE communications magazine*, vol. 43, pp. 112–119, December 2005.

[55] A. Sendonaris, E. Erkip, and B. Aazhang, “User cooperation diversity - part I system description,” *IEEE Transactions on Communications*, vol. 51, pp. 1927–1938, November 2003.

[56] A. Sendonaris, E. Erkip, and B. Aazhang, “User cooperation diversity - part II implementation aspects and performance analysis,” *IEEE Transactions on Communications*, vol. 51, pp. 1939–1948, November 2003.

[57] J. N. Laneman, *Cooperative Diversity in Wireless Networks: Algorithms and Architectures*. PhD, Massachusetts Institute of Technology, September 2002.

[58] Institute of Electrical and Electronic Engineers, *IEEE 802.15.4 Std.*, 2003.

[59] J. Zheng and M. J. Lee, “Will IEEE 802.15.4 make ubiquitous networking a reality?: A discussion on a potential low power, low bit rate standard,” *IEEE Communications Magazine*, vol. 42, pp. 140–146, June 2004.

[60] Zigbee Alliance. Available from http://www.zigbee.org/en/ [Accessed 13 May 2005].

[61] A. Wheeler, “Commercial applications of wireless sensor networks using zigbee,” *IEEE Communications Magazine*, vol. 45, pp. 70–77, APRIL 2007.

[62] A. Cunha, A. Koubâa, R. Severino, and M. Alves, "Open-ZB: an open-source implementation of the IEEE 802.15.4/Zigbee protocol stack on TinyOS," in *IEEE Southeastcon*, pp. 442–447, April 2008.

[63] J. G. Proakis, *Digital Communications*. McGraw Hill, 4th ed., 2001.

[64] S. Pasupathy, "Minimum shift keying: A spectrally efficient modulation," *IEEE Communications Magazine*, vol. 17, no. 4, pp. 14–22, 1979.

[65] W. C. Chung, N. J. August, and D. S. Ha, "Signalling and multiple access techniques for ultra wideband 4g wireless communication systems," *IEEE Wireless Communications*, vol. 12, pp. 46–55, April 2005.

[66] B. Sklar, *Digital Communications: Fundamentals and Applications*. Prentice Hall, 2nd ed., 2001.

[67] S. Haykin, *Communication Systems*. Wiley, 4th ed., 2001.

[68] S. Lin, D. Costello Jr, and M. Miller, "Automatic-repeat-request error control schemes," *IEEE Communications Magazine*, vol. 22, pp. 5–17, December 1984.

[69] S. Lin and D. Costello, *Error Control Coding*. Prentice Hall, 2nd ed., 2004.

[70] Y. Sankarasubramaniam, I. Akyildiz, and S. McLaughlin, "Energy efficiency based packet size optimization in wireless sensor networks," in *Sensor Network Protocols and Applications*, pp. 1–8, 2003.

[71] J. Wolf and R. I. Blakeney, "An exact evaluation of the probability of undetected error for certain shortened binary CRC codes," in *MILCOM '88.*, vol. 1, pp. 287–292, October 1988.

[72] W. Peterson and D. Brown, "Cyclic codes for error detection," *Proceedings of the IRE*, vol. 49, pp. 228–235, January 1961.

[73] G. Castagnoli, J. Ganz, and P. Graber, "Optimum cyclic redundancy-check codes with 16-bit redundancy," *IEEE Transactions on Communications*, vol. 38, pp. 111–114, January 1990.

[74] K. Witzke and C. Leung, "A comparison of some error detecting CRC code standards," *IEEE Transactions on Communications*, vol. 33, pp. 996–998, September 1985.

[75] T. Fujiwara, T. Kasami, A. Kitai, and S. Lin, "On the undetected error probability for shortened hamming codes," *IEEE Transactions on Communications*, vol. 33, pp. 570–574, June 1985.

[76] T. Baicheva, S. Dodunekov, and P. Kazakov, "Undetected error probability performance of cyclic redundancy-check codes of 16-bit redundancy," *IEE Proceedings on Communications*, vol. 147, pp. 253–256, October 2000.

[77] S. Leung-Yan-Cheong and M. E. Hellman, "Concerning a bound on undetected error probability," *IEEE Transactions on Information Theory*, vol. IT-22, pp. 235–237, March 1976.

[78] J. Mietzner and P. Hoeher, "Boosting the performance of wireless communication systems: theory and practice of multiple-antenna techniques," *IEEE Communications Magazine*, vol. 42, pp. 40–47, October 2004.

[79] S. Haykin and M. Moher, *Modern Wireless Communications*. Prentice Hall, 2005.

[80] S. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1451–1458, October 1998.

[81] S. Gravano, M. Doggett, and P. McDougall, "Comparison of a cyclic code and a repetition code with the same code rate in the presence of single-bit errors," *Int. J. Electronics*, vol. 67, no. 4, pp. 495–502, 1989.

[82] P. Loskot and N. Beaulieu, "A family of rate 1/2 modified binary block repetition codes," in *IEEE*, pp. 1985–1989, 2004.

[83] W. Lee, "The advantages of using repetition coding in mobile radio communication," in *Proc. 36th IEEE Vehicular Technology Conf.*, vol. 36, pp. 157–161, May 1986.

[84] A. Ali and I. Al-Kadi, "On the use of repetition coding with binary digital modulations on mobile channels," *37th IEEE Vehicular Technology Conference*, vol. 37, pp. 59–65, June 1987.

[85] A. Ali and I. Al-Kadi, "On the use of repetition coding with binary digital modulations on mobile channels," *IEEE Transactions on Vehicular Technology*, vol. 38, pp. 14–18, February 1989.

[86] D. Tse and P. Viswanath, *Fundamentals of Wireless Communications*. Cambridge, 2005.

[87] B. Friedlander and S. Scherzer, "Spatial diversity vs. array gain in cellular communication systems," in *Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 584–588, November 2002.

[88] G. Foschini and M. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, no. 3, pp. 311–335, 1998.

[89] E. Telatar, "Capacity of multi-antenna gaussian channels," *European Transactions on Telecommunications*, vol. 10, no. 6, pp. 585–596, 1999.

[90] D. Gesbert, M. Shafi, Da-shan Shiu, P. Smith, and A. Naguib, "From theory to practice: an overview of MIMO space-time coded wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 281–302, April 2003.

[91] C. Shuguang, A. Goldsmith, and A. Bahai, "Energy-efficiency of MIMO and cooperative MIMO techniques in sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, pp. 1089–1098, August 2004.

[92] V. Tarokh, N. Seshadri, and A. Calderbank, "Space-time codes for high data rates wireless communications: Performance criterion and code construction," *IEEE Trans. Inform. Theory*, vol. 44, no. 2, pp. 744–765, 1998.

[93] H. Bölcskei, "MIMO-OFDM wireless systems: Basics, perspectives, and challenges," *IEEE Wireless Communications*, vol. 13, pp. 31–37, August 2006.

[94] J. Laneman, D. Tse, and G. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," *IEEE Transactions On Information Theory*, vol. 50, pp. 3062–3080, December 2004.

[95] E. C. van der Meulen, "Three-terminal communication channels," *Advances in Applied Probability*, vol. 3, no. 1, pp. 120–154, 1971.

[96] T. Cover and G. A.E., "Capacity theorems for the relay channel," *IEEE Trans. on Information Theory*, vol. 25, pp. 572–584, September 1979.

[97] E. Zimmermann, P. Herhold, and G. Fettweiss, "On the performance of cooperative relaying protocols in wireless networks," *European Transactions on Telecommunications (ETT)*, vol. 16, pp. 17–35, January-February 2005.

[98] T. E. Hunter, *Coded Cooperation: A New Framework For User Cooperation in Wireless Networks*. PhD, The University of Texas at Dallas, May 2004.

[99] A. S. Avestimehr and D. N. C. Tse, "Outage capacity of the fading relay channel in the low-SNR regime," *IEEE Transactions on Information Theory*, vol. 53, pp. 1401–1415, April 2007.

[100] T. E. Hunter and A. Nosratinia, "Cooperative diversity through coding," in *Proc. IEEE ISIT*, (Laussane, Switzerland), p. 220, July 2002.

[101] P. Herhold, E. Zimmermann, and G. Fettweiss, "A simple cooperative extension to wireless relaying," in *International Zurich Seminar on Communications*, February 2004.

[102] M. Tacca, P. Monti, and A. Fumagalli, "Cooperative and reliable ARQ protocols for energy harvesting wireless sensor nodes," *IEEE Transactions on Wireless Communications*, vol. 6, pp. 2519–2529, July 2007.

[103] D. Woldegebreal, S. Valentin, and K. Holger, "Outage probability analysis of cooperative transmission protocols without and with network coding: Inter-user channels based comparison.," in *MSWiM '07: Proceedings of the 10th ACM Symposium on Modelling, analysis, and simulation of wireless and mobile systems*, pp. 36–44, 2007.

[104] X. Li, M. Chen, and W. Liu, "Application of STBC-encoded cooperative transmissions in wireless sensor networks," *IEEE Signal Processing Letters*, vol. 12, pp. 134–137, February 2005.

[105] A. Bletsas, A. Khisti, D. Reed, and A. Lippman, "A simple cooperative diversity method based on network path selection," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 659–672, 2006.

[106] M. Zuniga and B. Krishnamachari, "An analysis of unreliability and asymmetry in low-power wireless links," *ACM Transactions on Sensor Networks (to appear)*, 2007.

[107] X. Zeng, R. Bagrodia, and M. Gerla, "Glomosim: A library for parallel simulation of large-scale wireless networks," in *The 12th Workshop on Parallel and Distributed Simulations*, pp. 154–161, 1998.

[108] *TinyAODV Implementation, TinyOS Source Code Repository*. Available from http://tinyos.cvs.sourceforge.net/tinyos/tinyos-1.x/contrib/hsn/ [Accessed 05 September 2008].

[109] M. Busse, T. Haenselmann, and W. Effelsberg, "Energy-efficient forwarding schemes for wireless sensor networks," in *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pp. 125–133, 2006.

[110] T. S. Rappaport, *Wireless Communications: Principles and Practice*. Prentice Hall, 1996.

[111] K. Srinivasan and P. Levis, "RSSI is under appreciated," in *Proc. of the Third Workshop on Embedded Networked Sensors, EmNets*, May 2006.

[112] Texas Instruments, *MSP430 Datasheet*. Available from: http://focus.ti.com/lit/ds/symlink/msp430f1611.pdf [Accessed 14th March 2007].

[113] D. Chase, "Code combining – a maximum-likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Transactions on Communications*, vol. 33, pp. 385–393, May 1985.

[114] S. Yi, B. Azimi-Sadjadit, S. Kalyanaraman, and V. Subramanian, "Error control code combining techniques in cluster-based cooperative wireless networks," in *IEEE International Conference on Communications*, pp. 3510 – 3514, 2005.

[115] *TinyOS Website*. Available from http://www.tinyos.net [Accessed 20 September 2007].

[116] G. Yu, Z. Zhang, and P. Qiu, “Efficient ARQ protocols for exploiting cooperative relaying in wireless sensor networks,” *Computer Communications*, vol. 30, no. 14-15, pp. 2765–2773, 2007.

[117] Q. Liu, S. Zhou, and G. Giannakis, “Cross-layer combining of adaptive modulation and coding with truncated ARQ over wireless links,” *Wireless Communications, IEEE Transactions on*, vol. 3, pp. 1746–1755, September 2004.

[118] Nordic Semiconductor, *NRF905 Datasheet*. Available from: http://www.nordicsemi.com/files/Product/data_sheet/nRF905_rev1_3.pdf [Accessed 15th June 2008].

[119] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter, “Luster: wireless sensor network for environmental research,” in *SenSys ‘07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pp. 103–116, 2007.

[120] T. Wark and P. Corke, “Springbrook: Challenges in developing a long-term rainforest wireless sensor network,” in *ISSNIP ‘08: Proceedings of the Fourth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2008.

[121] W. Hu and P. Corke, “secfleck: A public key technology platform for wireless sensor networks,” in *EWSN ‘08: Proceedings of the 6th European Conference on Wireless Sensor Networks*, 2008.

[122] Texas Instruments, *CC1000 Datasheet*. Available from: http://focus.ti.com/docs/prod/folders/print/cc1000.html [Accessed 8th July 2008].

[123] K. Schwieger, A. Kumar, and G. Fettweis, “On the impact of the physical layer on energy consumption in sensor networks,” in *IEEE Proceedings of the Second European Workshop on Wireless Sensor Networks*, pp. 13–14, 2005.

APPENDIX

Implementing the Cyclic Redundancy Check

This appendix discusses two main approaches for implementing a 16-bit CRC. The code described within is written in C but its simplicity is kept to a minimum. Although this section is discussed in as general a nature as possible reference will be made to the TinyOS implementation and that of the CC2420. The latter complies with the IEEE 802.15.4 standard [5] and uses a hardware shift register to calculate the FCS. However, the software version implemented by TinyOS and the hardware version of the CC2420 produce different results for the same data. The reason for this will become clear as this section develops and is mainly noted as it can be a source of confusion when comparing the two. The IEEE 802.15.4 standard specifies use of the CRC-ITU polynomial given in (2.17) and it is this that will be used in the discussion; use of a different degree-16 polynomial proceeds identically.

The parity bits $(r_{n-k-1}, \ldots, r_1, r_0)$ shown in equation (2.14) are commonly referred to as the Frame Check Sequence (FCS). As the degree of the CRC-ITU polynomial is $n - k = 16$ the FCS is two bytes long and is normally appended to the end of the packet being sent. The value of the FCS will depend on a number of factors, one of which is the particular fields of the packet it is calculated over. The IEEE 802.15.4 standard [5] defines the packet format shown in figure 1 and stipulates that the FCS should be calculated over the MAC Header (MHR) and MAC payload parts of the

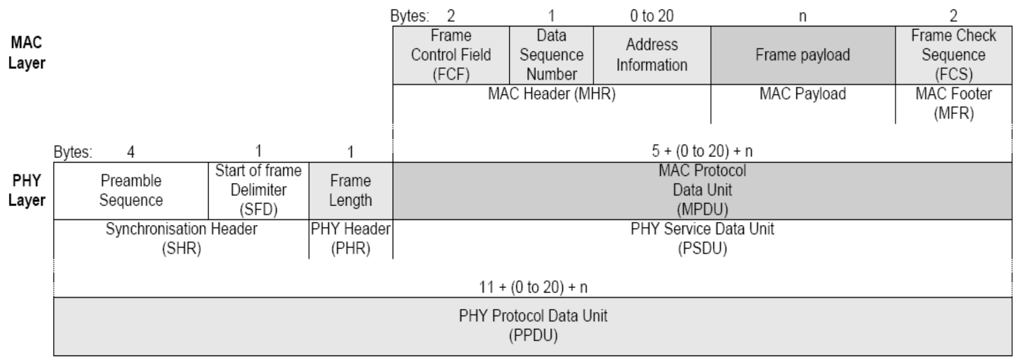


Figure 1: IEEE 802.15.4 Frame Format.

frame. The frame length field defines the number of bytes in the MAC Protocol Data Unit (MPDU) which consists of everything *after* the frame length field (including the FCS). It can be seen from the figure that the frame length is *not* part of the MPDU and therefore is not considered in the calculation of the FCS. The CC2420 complies with the IEEE 802.15.4 standard and the same rules therefore apply.

A.4.1 Software Implementations

The first approach described is a table-based approach (cf. figure 2). Of all the current approaches this one occupies the most space in memory but trades this off for high speed and a very simple coding procedure; it operates as follows. The table itself is an array of 256 entries, each entry consisting of the remainder when a particular byte value is divided by the CRC-ITU polynomial. As there are $2^8 = 256$ possible 8-bit sequences there are as many entries in the table. If, for example, the byte `b` is 0x00 (the prefix 0x representing a hexadecimal value in C) the remainder will be 0; if the byte `b` is 0xff the remainder will be 0x1ef0 — and similar for all values in between.

When the function `crcByte` is called, the current value of the FCS is input (`uint16_t` crc) along with the current byte (`uint8_t` b)[1]. An entry is then retrieved from the

[1]The type declarations uint8_t and uint16_t refer to 8 and 16-bit unsigned integers respectively.

```
uint16_t const tableITU[256] = {
  0x0000, 0x1021, ... , 0x60c6, 0x70e7,
  0x8108, 0x9129, ... , 0xe1ce, 0xf1ef,
                    .
                    .
                    .
  0xef1f, 0xff3e, ... , 0x8fd9, 0x9ff8,
  0x6e17, 0x7e36, ... , 0x0ed1, 0x1ef0
};

uint16_t crcByte(uint16_t crc, uint8_t b)
{
  crc = tableITU[(crc >> 8 ^ b) & 0xffU] ^ (crc << 8);
  return crc;
}
```

Figure 2: Table based CRC code. The table consists of a table of 256 entries and trades-off program space for speed.

table determined by the most significant byte of the current FCS (`crc >> 8`) bitwise XORed with the current byte (`^ b`). The least significant byte of the current FCS is then shifted into the position of the most significant byte replacing it (`crc << 8`). This is then bitwise XORed (^) with the value retrieved from the table and returned as the updated FCS. The reasoning behind this procedure is as follows [3].

The message is effectively operated on one byte at a time beginning with the MSB. For ease of exposition consider that the message $m(x)$ consists of only two bytes: m_u and m_l. This can be written as

$$m(x) = x^8 m_u(x) + m_l(x). \tag{1}$$

Using the systematic encoding procedure described in section 2.4.4.2 the following holds for $m_u(x)$:

$$x^{16} m_u(x) = q_u(x) g(x) + r_u(x) \tag{2}$$

for some $q_u(x)$ and $r_u(x)$. This implies that $r_u(x)$ is returned the first time the function `crcByte` is called (assuming the initial value of `crc` is set to 0). The second time this function is called the input parameters consist of $r_u(x)$ and $m_l(x)$, and the value returned is:

$$r(x) = R_{g(x)}[x^{16} m(x)] \tag{3}$$

where $r(x)$ represents the updated (and, for this two byte example, final) FCS. This can be expanded as

$$\begin{aligned} r(x) &= R_{g(x)}[x^{16} m_l(x)] + R_{g(x)}[x^8 q_u(x) g(x)] + R_{g(x)}[x^8 r_u(x)] \\ &= R_{g(x)}[x^{16} m_l(x)] + R_{g(x)}[x^8 r_u(x)] \\ &= R_{g(x)}[x^{16} m_l(x) + x^8 r_u(x)] \end{aligned} \tag{4}$$

due to the fact that $R_{s(x)}[t(x) + u(x)] = R_{s(x)}[t(x)] + R_{s(x)}[u(x)]$ for all $s(x), t(x)$ and $u(x)$.

Expanding $m_l(x)$ as $m_{l,7}x^7 + \ldots + m_{l,1}x^1 + m_{l,0}$ and $r_u(x)$ as $r_{u,15}x^{15} + \ldots + r_{u,1}x^1 + r_{u,0}$ the expression $x^{16} m_l(x) + x^8 r_u(x)$ in (4) can be expanded out resulting in

$$\begin{aligned} r(x) = &\underbrace{R_{g(x)}[(m_{l,7} + r_{u,15})x^{23} + \ldots + (m_{l,1} + r_{u,9})x^{22} + (m_{l,0} + r_{u,8})x^{16}]}_{\text{tableITU[(crc >>8 ^ b) \& 0xffU]}} + \\ &\underbrace{r_{u,7}x^{15} + \ldots + r_{u,1}x^9 + r_{u,0}x^8}_{\text{crc << 8}} \end{aligned} \tag{5}$$

```
uint16_t crcByte(uint16_t crc, uint8_t b)
{
  uint8_t i;

  crc = crc ^ b << 8;
  i = 8;
  do
    if (crc & 0x8000)
      crc = crc << 1 ^ 0x1021;
    else
      crc = crc << 1;
  while (--i);

  return crc;
}
```

Figure 3: Shift register based CRC code used in TinyOS 2.x (see [6])

where it is noted that $R_{g(x)}[r_{u,7}x^{15}+\ldots+r_{u,1}x^{9}+r_{u,0}x^{8}] = r_{u,7}x^{15}+\ldots+r_{u,1}x^{9}+r_{u,0}x^{8}$ as deg($g(x)$) = 16. Noting the fact that, in modulo-2 arithmetic, addition, subtraction and XOR are all the same operation, it can be seen from equation (5) why the code in figure 2 produces the correct result.

The second approach discussed is that used in TinyOS and is effectively a software version of the shift register method used in hardware. As before, a single byte (`b`) is accepted on each iteration and the variable `crc` updated accordingly. In C the bitwise XOR operator (^) has a lower precedence than the shift operator (<<) and therefore the assignment `crc = crc ^ b << 8` XORs the upper byte of `crc` with the incoming byte `b`. This is effectively the operation

$$(m_{l,7} + r_{u,15})x^{23} + \ldots + (m_{l,1} + r_{u,9})x^{22} + (m_{l,0} + r_{u,8})x^{16}$$

in equation (5). Following this, the 16-bit `crc` value is then checked to see if its most significant bit is a 1 (`crc & 0x8000`) and if so the CRC-ITU polynomial is subtracted from it (as would be done in normal long division). If the upper most bit is a 1 it will always go to zero when XORed with the upper most bit of $g(x)$. Therefore instead of explicitly carrying out this XOR operation with the full polynomial, the uppermost bit of `crc` is shifted off the end (`crc << 1`) and the result is XORed with the lower bits of $g(x)$, i.e. $x^{12} + x^5 + 1$ (0x1021). If the most significant bit is not a 1 then the algorithm simply removes this bit by shifting it left and loops back around to check the next bit. The value returned will be the FCS of the byte `b` provided `crc` was initially set to 0.

A.4.2 Operating on multiple bytes

It can be seen that both the table based and shift based approaches both operate on one byte at a time. To determine the FCS of a message with more than one byte, the function `crcByte` needs to be called for each byte of the message. The algorithm expects the MSB first and so the implementation will depend on which way the data is classified. One particular implementation example is shown in figure 4.

The function `calculateFCS` assumes that the MSB is contained in the *first* byte of the array (i.e. `msgPtr[0]`). The resulting FCS will later (after the function `calculateFCS` has returned) be appended to the end of this array. This then allows for integrity checking using the code of figure 5, which offers the advantage that only a one line comparison is required for the integrity check: `return FCS == 0`. If, on the other hand, the LSB had been stored in the first byte of the message, the `for` loop of figure 4 would have to have been run in the opposite direction with the FCS appended to the *beginning* of the array in order to retain the simple integrity check. Appending the FCS to the end of the array in this latter case would have meant explicitly accessing the received FCS for comparison with a recalculation of the FCS; this is slightly more cumbersome to do.

```
uint16_t calculateFCS(uint8_t* msgPtr){
  uint16_t FCS = 0x0000;

  for (n=0; n<DATA_SIZE; n++)
    FCS = crcByte(FCS, msgPtr[n]);

  return FCS;
}
```

Figure 4: Calculating the FCS of a message with more than one byte assuming the MSB is the *first* byte of the message.

```
bool checkFCS(uint8_t* msgPtr){
  uint16_t FCS = 0x0000;

  for (n=0; n<(DATA_SIZE + FCS_SIZE); n++)
    FCS = crcByte(FCS, msgPtr[n]);

  return FCS == 0;
}
```

Figure 5: Checking the FCS of a message with more than one byte assuming the MSB is the *first* byte of the message.

A.4.3 CRC on the CC2420

The normal way of implementing the CRC in hardware is to use a shift register. The CC2420 uses the shift register defined in the IEEE 802.15.4 standard to calculate the parity bits (cf. figure 6). The operation of this particular shift register has already been explained to a certain extent as the code in figure 3 is essentially a software implementation of the shift register.

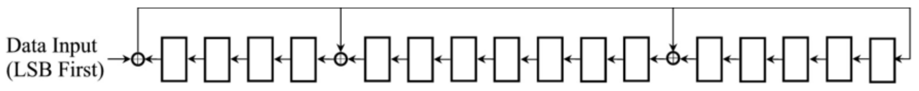


Figure 6: Hardware shift register described in IEEE 802.15.4 standard and used by the CC2420 chip.

As noted earlier however, for the code given in figure 3 the byte that is required first is the MSB whereas, in the hardware case (cf. figure 6), it is the LSB; in fact, this is not the only difference. Consider the code in figure 7. This is an exact software implementation of the shift register in figure 6. Notice however that the polynomial used is not 0x1021 as might be expected but 0x8048. This can be a source of confusion for designers comparing the values of the software approach described earlier and the hardware approach. The reason the CC2420 implements the CRC in this manner is due to the way the IEEE 802.15.4 standard [5] defines the order of bits in a message:

$$m(x) = b_0x^{k-1} + \ldots + b_{k-2}x + b_{k-1} \tag{6}$$

where b_0 is the least significant bit and b_{k-1} the most significant. This is different to the usual way the message is defined where the coefficient of the largest degree of x is generally the most significant bit. Using the notation of (6), however, the CRC-ITU polynomial (neglecting the x^{16} term), $x^{12} + x^5 + 1$, now represents the hexadecimal number 0x8408, as the coefficient of x^{16} represents the *least* significant bit.

So, for example, if the FCS of the polynomial 0x6A under this scheme is determined the result is 0xCC5C. If, on the other hand, the FCS of 0x56 is found using the code defined in figure 3 the result is 0x3A33. Careful inspection will reveal that the value 0x6A is simply 0x56 viewed from the opposite end and 0xCC5C is 0x3A33 viewed from the opposite end.

As can be seen, the code in figure 7 is almost identical to that of figure 3 except that the bit weightings are reversed. In the hardware version the polynomial is represented by the feedback taps taken at various points on the register. The left most tap representing the coefficient of x^{16}. Counting down it can be seen that the next tap represents that of x^{12} and the last two of x^5 and 1, respectively. Further details on shift registers can be found in any good textbook on error control systems (e.g. [4, 9, 69]).

```
uint16_t crcByte(uint16_t crc, uint8_t b)
{
  uint8_t i;

  crc = crc ^ b;
  i = 8;
  do
    if (crc & 0x0001)
      crc = (crc >> 1) ^ 0x8408;
    else
      crc = (crc >> 1);
  while (--i);

  return crc;
}
```

Figure 7: Software implementation of the shift register used in the CC2420. The hexadecimal value 0x8408 is 0x1021 viewed backwards.

A.4.4 CC2420 CRC configuration

In the CC2420 the register `MODEMCTRL0` contains the `AUTOCRC` bit (bit 5 in the register). When this bit is *set* the FCS is automatically generated and verified by hardware. During transmission the FCS is appended to the end of the packet: the position of which is determined by the `frame length` field. The FCS is not written to the RXFIFO, but stored in a separate 16-bit register. In receiver mode the FCS is verified by hardware but is *not* written to the RXFIFO. Instead, the two FCS bytes are replaced by the Received Signal Strength Indicator (RSSI), the average correlation value (CORR) and a single bit representing the outcome of the CRC check. As a result, the value of the FCS is not accessible to higher layers.

If required, it is possible to gain access to the FCS by clearing the `MODEMCTRL0.AUTOCRC` bit in the receiver. The value of the FCS can then be obtained by reading the last two bytes of the message in the RXFIFO. The `MODEMCTRL0.AUTOCRC` bit will, of course, still need to be set in the transmitter otherwise an FCS will not be generated in the first place. To effectively carry out this procedure in TinyOS 2.x requires some modification of lower level files. As is good practice when modifying these files, the version to be modified should be a copy of the original placed in the application directory. In this way the changes will not affect other applications which may not need them (also if a problem occurs the original files will still be intact). The following line of code needs to be removed from the file CC2420ControlP.nc: `1 << CC2420_MDMCTRL0_AUTOCRC`. The value of the right operand is defined in the header file CC2420.h and is set to the value 5 as `AUTOCRC` is the 5th bit in this register, as explained earlier. In the file CC2420ReceiveP.nc the line `metadata->lqi = buf[ length ] & 0x7f;` needs to be replaced with `metadata->lqi = buf[ length ];` to allow the entire upper byte of the FCS to be assigned. Although these changes refer specifically to TinyOS 2.0 later versions should be very similar in this respect.

A.4.5 Other implementations

Other possible implementations of the CRC are described in [3]. In this work a comparison of five different possible implementations was carried out while encoding a 512-byte message. The results of this comparison are shown in table 1. The algorithms CRCT (table based CRC) and CRCS (shift register based CRC) refer to those of figures 2 and 3 respectively; however, they were implemented in assembly using the CRC-ANSI polynomial. The other algorithms referred to in the table are an on-the-fly lookup table (CRCF) used to keep the storage requirements low, and reduced table lookup algorithms for both bytewise (CRCR(B)) and wordwise (CRCR(W)) operations. While CRCT is the fastest to implement it requires the greatest amount of storage. However, speed translates into reduced energy costs in this situation and it should therefore be considered as a candidate when implementing the CRC on WSNs. The on-the-fly algorithm offers the best overall trade-off. However, it is not as flexible as the others when different generator polynomials are required. Further details on these algorithms can be found in [3].

Algorithm	Time (ms)	Storage (bytes)
CRCS	61.2	56
CRCT	8.4	553
CRCF	18.8	67
CRCR(B)	53.3	72
CRCR(W)	50.9	88

Table 1: A comparison of five different possible implementations of the CRC algorithm with the time results measured over a 512-byte message (taken from [3]).