Energy-Aware Network Coding Circuit and System Design

by

Georgios Angelopoulos

B.S. in Electrical and Computer Engineering,
University of Patras, Greece (2009)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2011

Author .....
Department of Electrical Engineering and Computer Science
May 20, 2011

Certified by... .....
Muriel Médard
Professor of Electrical Engineering
Thesis Supervisor

Certified by.....
Anantha P. Chandrakasan
Joseph F. and Nancy P. Keithley Professor of Electrical Engineering
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Chair, Department Committee on Graduate Students

Energy-Aware Network Coding Circuit and System Design

by

Georgios Angelopoulos

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2011, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Network Coding (NC) has been shown to provide several advantages in communication networks in terms of throughput, data robustness and security. However, its applicability to networks with resource constrained nodes, like Body Area Networks (BANs), has been questioned due to its complexity requirements. Proposed NC implementations are based on high-end CPUs and GPUs, consuming hundreds of Watts, without providing enough insight about its energy requirements. As more and more mobile devices, sensors and other low power systems are used in modern communication protocols, a highly efficient and optimized implementation of NC is required.

In this work, an effort is made to bridge NC theory with ultra low power applications. For this reason, an energy-scalable, low power accelerator is designed in order to explore the minimum energy requirements of NC. Based on post-layout simulation results using a TSMC 65nm process, the proposed encoder consumes 22.15 uW at 0.4V, achieving a processing throughput of 80 MB/s. These numbers reveal that NC can indeed be incorporated into resource constrained networks with battery-operated or even energy scavenging nodes.

Apart from the hardware design, a new partial packet recovery mechanism based on NC, called PPRNC, is proposed. PPRNC exploits information contained in partial packets, similarly to existing Hybrid-ARQ schemes, but with a PHY-agnostic approach. Minimization of the number of retransmitted packets saves transmission energy and results in higher total network throughput, making PPRNC an attractive candidate for energy constrained networks, such as BANs, as well as modern, high-speed wireless mesh networks. The proposed mechanism is analyzed and implemented using commercial development boards, validating its ability to extract information contained from partial packets.

Thesis Supervisor: Muriel Médard
Title: Professor of Electrical Engineering

Thesis Supervisor: Anantha P. Chandrakasan
Title: Joseph F. and Nancy P. Keithley Professor of Electrical Engineering

Acknowledgments

First, I would like to express my deepest gratitude to my advisors, Prof. Muriel Médard and Anantha P. Chandrakasan. Both of them provided precious guidance and endless support during my first two years at MIT, which were instrumental for the completion of this thesis, as well as for my personal development. I have been amazed and continuously motivated by their perspective and vision. I also consider myself very fortunate being able to collaborate with their groups of exceptionally talented students and multidisciplinary research interests.

I would also like to acknowledge Prof. Dina Katabi for including me in her research group during the first semester of my graduate studies and making my transition to graduate life and US smoother. Her dedication to work and constructive questioning of the traditional approach to several research problems taught me a lot and considerably influenced my perspective.

I wish to thank all students of Prof. Médard's group for making the lab such an incredibly enjoyable and interesting place to work on a daily basis. Laughs during breaks from research and lunchtime conversations were extremely beneficial. Special thanks to Shirley for being a wonderful officemate (who never complains about anything), for answering 'network coded' questions and, above all, for being willing to help with any kind of problem. I would also like to thank all *ananthagroup* students for transforming their cubicles to easily accessible sources of knowledge and fun; special thanks to Arun, Phil, Patrick, Nathan, Mahmut and Joyce. I extend thanks to all my friends here in Boston, sharing unforgettable moments during the last two years. I should also thank my good friends back in Greece for their support and my girlfriend, Konstantina, for her love and patience.

Many thanks go to Michael Lewy and Margaret Flaherty for being always willing to help with logistics and especially for scheduling appointments. I would also like

to thank FCRP IFC program for funding my second year of studies[1] and Texas Instruments for providing development boards in order to experiment with low power sensors and real channel measurements. Special thanks to Dr. Alice Wang, who was instrumental in getting the boards on time.

Not only the current thesis but also all the achievements in my life, academic or not, are (and will be) dedicated to my family. Mom, Dad and Aggeliki, thank you very much for supporting me in any decision of my life. Your endless love is always accompanying me, even thousands miles away.

[1]This material is based upon work funded by the Departmental Fellowship 'Paris Kanellakis' and the Georgia Institute of Technology Under Award Number: 017894-010.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A communication network is usually considered in Information Theory as a graph with edges the communication links and vertices the network nodes. A widely used network model, which was the only one for several decades, is the *routed-packet network model*. According to this model, packets within the network are routed towards their destination on a hop-by-hop basis; nodes forward received packets onto the appropriate outgoing link(s). However, Ahlswede *et al.* claimed that, from an information point of view, restricting intermediate nodes in only routing packets limits overall network's performance and introduced the *coded-packet network model*, as well as the notion of Network Coding [3].

Network Coding (NC) has received extensive research attention in Communications and Networking community the last years and the range of its applications has already been vast: from network management and network tomography [4, 5] to capacity achieving communication schemes [6, 7]. Its revolutionary idea is to allow intermediate nodes within a network to mix previously received or locally generated packets together and let the final destinations decode the mixtures. By encouraging intermediate nodes to perform coding operations on the content of packets throughout the network, the coded-packet network model can demonstrate several advantages, such as throughput gains, increase in data robustness, security and better utilization of network resources in a wide variety of scenarios [8–10]. The model of coded-packet network is a generalization of the routed-packet model and the ability of its nodes to

mix packets together is the key reason for superior performance.

However, several questions arise concerning this new model, such as which are the nodes within the network that should code, when coding results in significant performance benefits and which packets or flows should be mixed together. The answer in all these questions is usually called in Information Theory community as *code*. Some of the most important code design approaches found in the literature have been proposed by Ho *et al.* [11], Jaggi *et al.* [12], and Fragouli *et al.* [13]. Although these techniques show analytically that NC does provide advantages, it is generally true that it can not be applied with the same efficiency and performance benefits in any network and under any communication scenario. For instance, since coding is a more complex operation compared to routing, it increases the computational requirements of nodes and requires extra amount of energy, which may or may not be available. Thus, careful examination of the different trade-offs is required in order to get the maximum performance benefits of incorporating NC into a system architecture.

Particular attention should be paid to different type of applications and networks, which require different treatment because of their specific characteristics. For instance, in high-speed optical networks, due to the low propagation time of transmission links, the communication bottleneck is usually the delay introduced by processing the information signal, and so, not careful use of NC may considerably harm the overall throughput of the network. An other example is wireless sensor networks; NC may reduce their expected lifetime due to its increased computational requirements if its transmission energy savings are not larger than the extra required processing power. Thus, depending on the application specific characteristics, a detailed analysis of NC is required.

In this thesis, we study the applicability of NC in resource-constrained, low power, low data rate wireless networks, such as Body Area Networks, and we investigate its advantages in terms of energy and resource utilization. Our work is two-fold; we first design a low power NC accelerator and then, we develop a partial packet recovery mechanism based on NC which can minimize the number of retransmitted packets, resulting in significant energy savings and throughput gains.

1.1 Motivation

The continuous scaling of electronics and the ultra low power circuit design techniques, as well as the advances in the Wireless Communications field, such routing and congestion algorithms in dynamic ad-hoc networks, have enabled the development of a new type of network architecture, known as Wireless Sensor Networks (WSNs). WSN is a network of spatially distributed autonomous devices, which are usually equipped with multiple sensor interfaces, analog front-ends and low-power processing circuits, analyzing and transmitting information contained in specific signals of interest. Typically, each node is powered by a small battery or an energy harvesting source, so extremely strict requirements are associated with its energy consumption.

Considering human physiological activities and actions monitoring, system engineers and application developers have introduced a specific type of WSNs, known as Body Area Networks (BANs). BANs is a promising network architecture with several potential advantages. For instance, it may enable the development of new health-related applications, helping the increasingly number of aging population and the overloaded health-care system. BANs are usually composed of multiple nodes, around or in the human body, transmitting vital information or other signals of interest, and a base station, collecting the transmitted data and coordinating the nodes.

Taking into account the importance of BANs as well as the existing constraints and challenges, the applicability of Network Coding (NC) in this specific type of networks is investigated. The extent and frequency of benefits of NC are studied in such a resource-constrained environment. Although NC has been shown to provide several advantages in wireless networks, its incorporation in low power, low data rate sensor networks has not been justified yet. For this reason, a custom NC accelerator is designed in this thesis, exploring the minimum energy requirements of it.

In addition, a new partial packet recovery mechanism based on NC, called PPRNC, is proposed. Majority of current communication networks operate in an all-or-nothing mode, using CRC checks or other error detection codes at the link layer in order to determine the validity of received packets. In this way, no erroneous information is

propagated to higher layers of the protocol stack. However, dropping packets that do not satisfy the validity check and requesting a retransmission of the entire packet is inefficient in terms of resource utilization, since partial packets have usually large amount of useful information. Several techniques have been proposed to eliminate this inefficiency but most of them are not applicable to BANs due to their high computational requirements.

According to PPRNC, sensor nodes apply NC to their sensed data and transmit the coded packets, while the base station collects and decodes the mixtures. Processing partial packets and extracting useful information reduces the required number of retransmitted packet by the sensor nodes, resulting in significant energy savings. Moreover, PPRNC is an attractive candidate for modern, high-speed WLANs since reducing the number of retransmissions results in higher total throughput.

1.2 Thesis Contributions

The main contributions of this thesis can be summarized in the following lines:

- A detailed energy modeling of the required operations by Network Coding is presented, based on VLSI measurements. This allows the precise estimate of the energy consumed by Network Coding and the detailed analysis of a system's energy budget.

- Based on the created energy models, a systematic approach for specifying optimum algorithmic parameters of Network Coding in a typical BAN scenario is presented.

- A low power accelerator is designed using a TSMC 65nm CMOS process, performing the Network Coding encoding process. Its power consumption indicates that Network Coding can be successfully incorporated in extremely resource constrained networks, such as BANs.

- A new partial packet reception mechanism based on Network Coding (PPRNC)

is proposed in order to exploit information contained in partial packets and minimize the number of retransmitted packets.

Chapter 2

Background

This chapter presents briefly all the background information required to understand the material contained in this thesis. Although it is not a in-depth analysis, it gives an overview of the power consumption in digital CMOS circuits and energy sources in modern microsystems, introduces the required definitions about Network Coding and presents some of the key characteristics of Body Area Networks (BANs).

As shown in Figure 2-1, the simplistic block diagram of a typical micro-sensor is composed by four major parts: the interface to the analog world, the DSP or processing logic, the radio and the energy source. Although all of them are equally important, we focus on the processing logic and its power consumption since Network Coding can be implemented as a digital processing circuit.

2.1 Power Consumption and Energy Sources in Modern Microsystems

As more and more battery-operated devices, such as cameras, smart-phones and sensors, become increasingly popular nowadays, their power consumption has emerged as an issue of major importance. Because of commercial trends, aesthetic reasons, usability and cost of these devices, battery size and capacity has to be relatively small. Low power design techniques, reducing the amount of power required by their

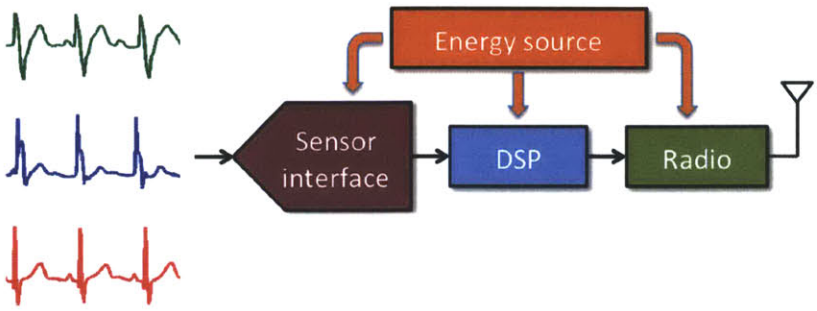

Figure 2-1: A simplistic block diagram of a typical microsensor.

integrated circuits, are usually the way to extend their expected lifetime.

The total power consumption of a digital CMOS circuit is given by Equation 2.1, where P_{active} is the power consumption for processing input data, charging and discharging different capacitances of the circuit, P_{leak} is the power consumed by the circuit in idle mode and $P_{short\_circuit}$ a process and input dependent power consumption component (it is usually negligible compared to the other components, so only the first two will be commented).

$$P_{total} = P_{active} + P_{leak} + P_{short\_circuit} \tag{2.1}$$

P_{active} represents the dynamic or active power of a circuit and is consumed when capacitors are charged (low-to-high transition). It is proportional to the total capacitance (C_L), the square of the supply voltage (V_{DD}), the switching activity (a) and the operating frequency (f). As shown in Equation 2.2, lowering the supply voltage is the most effective way to reduce the dynamic power consumption, due to the quadratic relationship. Supply voltages below transistors' threshold (V_t) lead to operation in the *sub-threshold* regime, which typically enables significant power savings with some performance degradation.

$$P_{active} = \alpha \times f \times C_L \times V_{DD}^2. \tag{2.2}$$

Leakage or static power consumption (P_{leak}) is the power consumed in the absence of any switching activity or execution of processing operations. It is proportional to

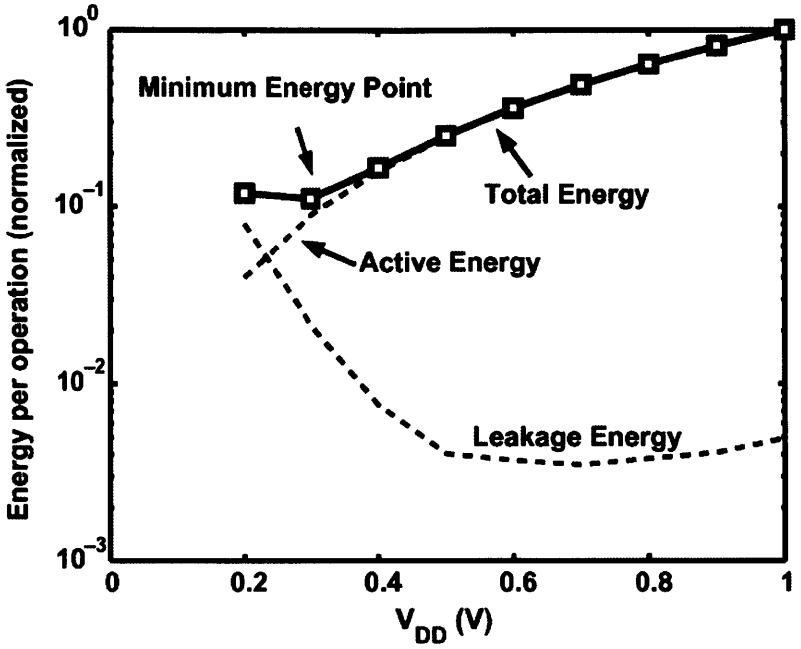

Figure 2-2: Energy per operation of a digital processing circuit in terms of different supply voltages, as shown in [1]. In above-threshold operation, active energy is the dominant component of the total energy but, in near and sub-threshold operation, leakage energy becomes the most significant.

leakage currents (I_{leak}) through the transistors' channel, substrate and gate, and the supply voltage (V_{DD}) :

$$P_{leak} = I_{leak} \times V_{DD}. \tag{2.3}$$

Lowering of V_{DD} results also in reduced leakage currents. Thus, supply voltage scaling is a very attractive solution for lowering both dynamic and leakage power, and sub-threshold operation has became a very popular design choice. However, the benefits of voltage scaling can not be exploited continuously because of the inverse relationship between the propagation delay (T_{delay}) and the supply voltage. As shown in Equation 2.4, operating in saturation region and lowering the supply voltage increases approximately linearly the propagation delay, while in sub-threshold region, the increase is exponential. Thus, lowering the supply voltage results in significant performance degradation which may or may not be acceptable. Other techniques, such as dynamic voltage scaling, parallelism, *etc.*, have to be used in order to amor-

Table 2.1: Typical energy sources in modern microsystems and their offered power densities [2].

Energy harvesting source	**Power density**
Micro-battery	50 mW/cm^3
Solar cell (direct sun light)	15mW/cm^2
Solar cell (indoors)	6$\mu W/cm^2$
Piezoelectric	330$\mu W/cm^2$
Thermoelectric (10°C gradient)	60$\mu W/cm^3$
Vibration	0.01-0.1mW/cm^2

tize for this performance penalty.

$$T_{delay} = \frac{C_L \times V_{DD}}{I_D} = \frac{C_L \times V_{DD}}{\mu C_{ox}(W/L)(V_{DD} - V_t)^{\alpha}}, \tag{2.4}$$

where $1 < \alpha \leq 2$.

Apart from the instantaneous power constraints, energy consumption limits are also of major importance. From a system's perspective, it would be desirable for a circuit to complete a processing task with the minimum amount of energy spent. So, examining the energy per operation of a circuit for different supply voltages, a sweet-spot or a *minimum energy point* generally exists near or below the transistors' threshold voltage, as shown in Figure 2-2 for a specific digital processing circuit [1]. In above-threshold operation, the active energy is the dominant component, but, in sub-threshold operation, leakage energy becomes the most significant. This is because the propagation delay increases exponentially as explained in the previous paragraph, and consequently, leakage energy, which is the integration of power over operating time, increases rapidly, as shown in Equation 2.5.

$$E_{leak} = \int_{T_{delay}} P_{leak}\, dt \tag{2.5}$$

Modern microsystems are usually powered by a combination of energy storage and energy harvesting sources. Common energy storage devices are batteries of different technologies (nickel-cadmium, lithium, etc.) and the more recently developed

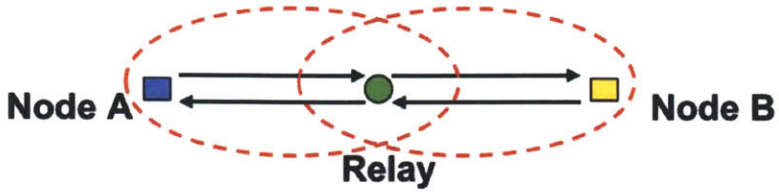

Figure 2-3: Node A and B are exchanging information using an intermediate relay since their transmission range is not enough to communicate directly. They also operate under half-duplex constraints, meaning that only one node can transmit at a given time slot.

ultra-capacitors. In Table 2.1, some of the energy sources, including some harvesting sources, are presented with their power densities [2]. As it becomes obvious, microsensors and other mobile devices operate under strict energy limits and their power consumption usually limits their functionality and lifetime.

2.2 Introduction to Network Coding

Ahlswede *et al.* introduced the notion of Network Coding (NC) in their seminal paper [3], claiming that, from an Information Theory point of view, there is no reason to restrict intermediate nodes in only forwarding packets at outgoing links (traditional routing), without allowing them to mix packets. Actually, it was shown that the mixing ability of nodes can result in higher network performance with several advantages. Its revolutionary idea is to allow intermediate nodes within a network to mix previously received or locally generated packets together and let the final destinations decode the mixtures. This results in several advantages, such as throughput gains, increase in data robustness, security and better utilization of network resources in a wide variety of scenarios [8–10].

The basic idea of NC is illustrated in Figures 2-3 and 2-4. In Figure 2-3, two nodes (A and B) are shown exchanging information through the use of a relay. Because the nodes are sharing the same common medium and they are subject to half-duplex transmission constraints, only one node is allowed to transmit information at a specific

time slot. Thus, for the exchange of a packet from node A to node B and vice versa, four time slots are required, as depicted in Figure 2-4a.

Examining in more details the transmission of packets in the previous scenario, it can be pointed out that there is an inefficiency. During the second time slot, when the intermediate node transmits the packet to node B, node A is considered to sleep and does not receive any packet since the transmission is not innovative for it. However, it would be desirable from a throughput perspective to make this transmission useful to node A as well. NC successfully deals with this inefficiency. As shown in Figure 2-4b, both nodes are transmitting their packets to the relay in the first two time slots and, during the third one, the relay codes these packets together and broadcasts the mixture. Assume for the time being that the mixing process is a simple addition. By receiving the mixture and subtracting their initial transmitted packet, node A and B can recover the packet sent from the other node.

This is an example illustrating throughput benefits and minimization of transmitted packets using NC. Since the mixing operation is performed in the digital domain, or equivalently, operations are over bits, it is called Digital Network Coding. An other type of NC has been proposed in the literature, called Analog Network Coding, which mixes the packets through the superposition of their physical layer signals. In the rest of this thesis, only Digital Network Coding is examined. In the following paragraphs, the encoding and decoding process is presented.

2.2.1 Encoding Process

Assume that a node has to transmit n packets, $\{p_1, p_2, \ldots, p_n\}$, each of them with length of L bits. With NC, the node creates and transmits n coded packets, $\{p'_1, p'_2, \ldots, p'_n\}$, as shown in Figure 2-5. Although several ways have been proposed to create these coded packets, it has been shown that linear codes are sufficient to achieve full advantages of NC [15]. Moreover, Trace Ho *et al.* developed an elegant, distributed Random Linear NC (RLNC) approach [11], according to it nodes within the network randomly choose set of coefficients from a Galois Field, multiply and add their packets, and send linear combinations at the outgoing links. This corresponds

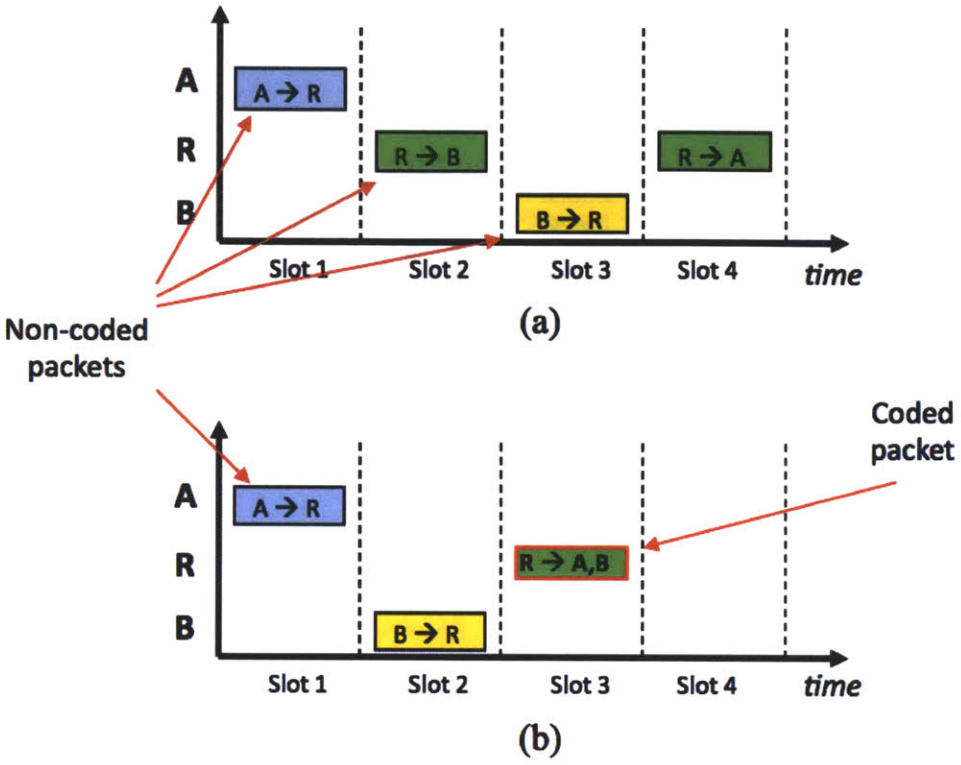

Figure 2-4: (a) Operation without Network Coding - intermediate node just forwards packets to their destination. Four time slots are required in total for the transmission of two packets, one from node A to node B and vice versa. (b) Operation with Network Coding - intermediate node is allowed to mix packets together. The total number of required time slots is reduced in three.

to the encoding process, which can be viewed as a matrix multiplication:

$$P' = C \times P, \tag{2.6}$$

where C is the coefficients' matrix, P is the matrix with the initial packets ($P = [p_1, p_2, \ldots, p_n]$) and P' is the matrix of coded packets ($P' = [p'_1, p'_2, \ldots, p'_n]$).

All operations are performed over Galois Fields, explained in more details in Section 3.2. A packet is considered to contain L/q symbols, where a symbol is formed by q consecutive bits. So, $p_i = [p_{i1}, \ldots, p_{iL/q}]$, where $p_{ij} \in \mathrm{GF}(2^q)$ and $j \in [1, L/q]$. The number of bits (q) in each symbol is usually called *field size* and the number of packets coded together is called *generation size*. Every coded packet p'_i, being a linear combi-

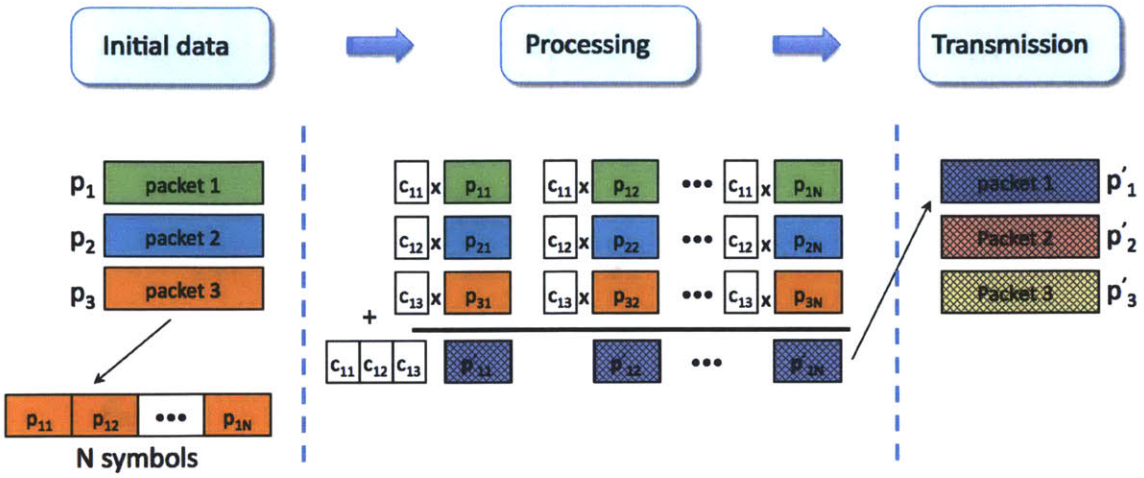

Figure 2-5: Encoding process of the Network Coding.

nation of the initial packets, is associated with a set of coefficients $c_i = [c_{i1}, \ldots, c_{in}]$, where $c_{ij} \in \mathrm{GF}(2^q)$ and $j \in [1, L/q]$. These coefficients are usually calculated through a pseudo-random number generator, such as a linear feedback shift register (LFSR), or stored in look-up tables. Thus, for the creation of a new coded packet p'_i, also called *degree of freedom* (dof), a node randomly picks a new set of coefficients c_i and calculates the following sum:

$$p'_{ij} = \sum_{k=1}^{n} c_{ik} p_{kj}, \tag{2.7}$$

where $i \in [1, n]$ and $j \in [1, L/q]$.

2.2.2 Decoding Process

As soon as a node receives n linear independent coded packets, it can start the decoding process, which is actually a problem of solving n linear equations with n unknowns. The coefficients associated with every coded combination are either transmitted with the actual packet or fetched through a look-up table (the same look-up table used at the encoding process). Thus, through the following equation:

$$P = C^{-1} \times P', \tag{2.8}$$

the initial packets can be recovered, given that the matrix C is invertible. The probability of matrix C being invertible depends on the field size; a regular Gaussian elimination method over Galois Fields can be used for its inversion.

2.3 Body Area Networks Overview

The emerging field of wireless sensor networks (WSNs) involves sensing, computation and communication performed by spatially distributed autonomous devices, which are usually equipped with multiple sensor interfaces, analog front-ends and low-power processing circuits. Although the area of WSNs is relatively young, an increasingly number of applications are based on this type of networks, mainly because of their unique characteristics, such as low cost, large area and distributed network deployment. Some of their applications are environmental monitoring, smart grids and buildings, military surveillance and health monitoring systems.

Considering human physiological activities and actions monitoring, system engineers and application developers have introduced a specific type of WSNs, known as Body Area Networks (BANs). The reason for introducing a new category of sensor networks is because of the unique characteristics and application requirements of BANs. Some of them are the following:

- **Deployment and Density**: In BANs nodes are placed in or on the human body, contrary to WSNs, which are often deployed in much larger areas. Also, because of aesthetic reasons, BANs do not make use of redundant nodes to deal with device failures, which is a common design choice in several WSNs applications, so BANs are usually less node-dense.

- **Mobility**: In the majority of WSNs nodes are usually considered stationary, while in BANs nodes move with the human body, so they experience approximately the same mobility patterns during a day as a human, influencing their communication performance.

- **Distance**: Communication distance between nodes in BANs is usually on the

Table 2.2: Typical health related applications in BANs and their functional requirements.

Application	Target data rate	Latency
Drug delivery	< 10 Kbps	< 1 s
Deep brain stimulation	< 320 Kbps	< 250ms
Capsule endoscopy	> 1 Mbps	≈ 100ms
ECG	72 Kbps (6 Kbps, 12 channels)	< 250ms
EEG	≈ 100 Kbps	< 250ms
EMG	≈ 1.5 Mbps	< 250ms
Glucose level monitor	<1 Kbps	< 250ms
Audio	1 Mbps	< 20ms
Video/Medical imaging	<20 Mbps	<10 ms

range of 1cm to 3m, while in typical WSNs, distance can be up to 100 or 1000m.

- **Data Rate and Latency**: Although data rate and latency are mainly application specific, majority of WSNs are employed for event-based monitoring, while BANs are usually used for continuously monitoring humans' physiological activities and actions with specific data rates and strict latency requirements.

BANs' applications include personal entertainment, military surveillance and medical systems; some of the health related applications with their functional requirements are shown in Table 2.2 [16].

Because of the new challenges that BANs face, there is an ongoing effort for standardization of this specific type of networks, optimized for ultra low power devices and operation on, in or around the human body. Current wireless protocols have been designed and optimized for different purposes; for instance, WiFi for data networks, Bluetooth for voice links, Zigbee for industrial sensor applications, *etc.* Thus, due to their unique characteristics, BANs are considered as a promising solution and an engineering challenge, attracting the interest of both academia and industry.

Chapter 3

Design of a Network Coding Accelerator

In this chapter, after a short literature review, we present the need for a custom low power Network Coding implementation, designed for the increasingly number of mobile devices and sensors. In order to achieve the desired energy efficiency, Galois Field operations are analyzed and different implementation architectures are examined, modeling and comparing their energy consumption. After the analysis of Galois Field operations, we make use of the energy models to specify the optimum algorithmic parameters for a system using Network Coding, in order to achieve minimum system energy consumption. Finally, the architecture and implementation results for a low power accelerator performing Network Coding encoding are presented.

3.1 Motivation

In the literature, a few papers deal with implementation issues of Network Coding (NC), and almost all of them use high-performance CPUs or GPUs, focusing on the maximum achievable throughput, without analyzing the energy trade-offs of its incorporation into a system architecture. For instance, in [17] a 3.6 GHz Xeon Dual-Core processor is used to perform NC, achieving a coding throughput of approximately 5MB/s, while, for similar settings, in [18] NC is implemented using a 800MHz

Celeron CPU, achieving a throughput of 44MB/s. In addition, in [19] a special type of systematic NC over GF(2) is implemented, both on a cell phone and a laptop, achieving maximum reported throughput of 40MB/s and 1.5GB/s, respectively.

However, authors of mentioned works do not consider the energy analysis neither of the coding process, nor of the implications that the specific algorithmic parameters may have in the total system's energy; for instance, a possible increase in packet retransmissions due to linear dependent packets. Authors in [20–23] make use of multi-core CPUs and GPUs to speed up both encoding and decoding of NC. While remarkable effort is required to achieve this coding performance, the power budget of these approaches is in the order of 100 to 500W, number which is generally prohibitive for low power systems. Finally, an iPhone is used as the implementation device in reference [24] , where a maximum throughput of 420 KB/s is reported, while NC is responsible for approximately 33% of the reduction in the total battery life-time.

In Figure 3-1 the throughput of the encoding process of published NC implementations versus the required power budget of their hardware platforms is plotted. It is also emphasized in the graph the target area that would make NC applicable in BANs and other next generation wireless networks. Since different implementation solutions (i.e. CPUs, GPUS and custom ASICs) are used in these implementations, a direct comparison of them would be unfair and misleading. However, the graph emphasizes the following two observations: first, current implementations report widely varying results, without providing much of insight regarding the energy consumed by NC, and second, a few orders of magnitude more energy efficient implementations are required for next generation mobile devices.

As described in Section 2.2, a coded packet is created as a linear combination of packets and the decoding process is equivalent of a matrix inversion problem. Operations like addition, multiplication and division over Galois Fields are involved in the en-/decoding process. As a result, for an energy efficient implementation of NC, a detailed examination of these operations should be done in advance.

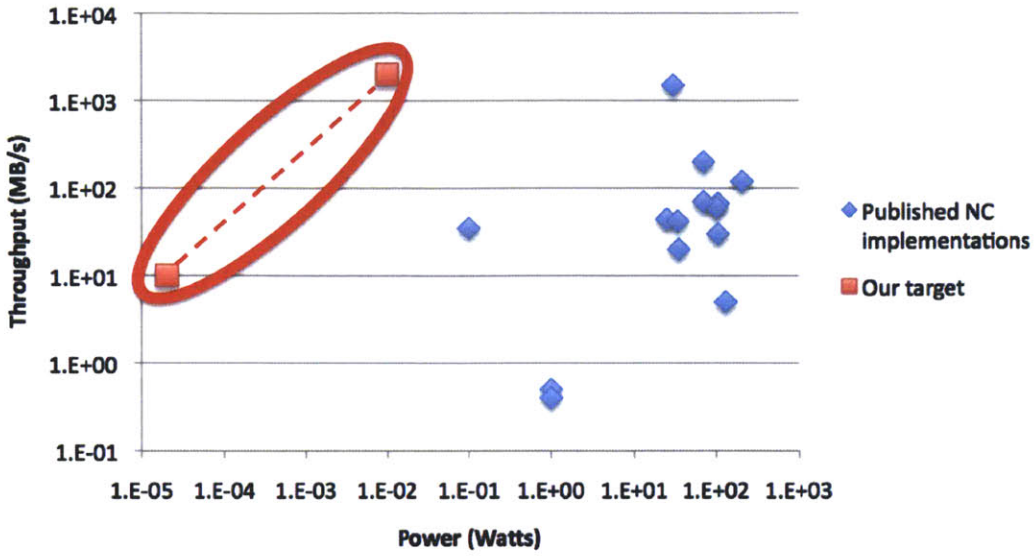

Figure 3-1: Reported Network Coding implementations, based on MCUs, CPUs and GPUs, with their power requirements and throughput of the encoding process, as well as our target low power implementation, based on a custom accelerator.

3.2 Galois Fields Arithmetic

In this Section, we provide a brief description of the most important concepts of Abstract Algebra related to NC. Our description serves only the purpose of explaining the decisions made during the design steps; for a more mathematically rigorous approach readers are referred to Algebra books.

3.2.1 Galois Fields Fundamentals

A *field* $\mathbb{F}$ is a set of at least two elements, with the operations $\otimes$ and $*$ (often called addition and multiplication), satisfying certain properties. One of these properties is that $\mathbb{F}$ is closed under the two operations, meaning that when an operation is applied to some elements of $\mathbb{F}$, the result will also be an element of this field. The number of elements in $\mathbb{F}$ is called *order* and, when this number is finite, the field is called *finite field*, denoted also as *Galois field* (GF). For any prime number p, it is always defined a GF with order p, represented as GF(p), having exactly p elements:

$$\mathrm{GF}(p) = \{0, 1, \ldots, p-1\}.$$

We can also create a GF(p^q), for any $q>0$, called *extension field* of GF(p). The definition of *field size* is often used to characterize the size of a field, denoted as q, where $q = log_p p^q$. Elements from GF(p^q) are usually considered and treated as vectors $[a_{q-1}, \ldots, a_0]$ or polynomials of degree at most (q-1) with coefficients from GF(p) :

$$GF(p^q) = \{A | A = a_{q-1}x^{q-1} + \ldots + a_1 x^1 + a_0, \text{ for } a_i \in GF(p), 0 \leq i \leq q-1\}. \quad (3.1)$$

Finally, all GFs contain a zero, an identity and a primitive element, and have at least one primitive polynomial of degree q associated with them.

The representation basis of the elements in a field is a crucial aspect, determining the efficiency and complexity of the implementation of different arithmetic operations. In general, different bases result in quite different representation and implementation of arithmetic operations (apart from addition/subtraction). There are several representation bases; the more popular among them are the standard (or polynomial) and the normal basis. The standard basis is the set of elements $\Omega = \{1, \omega, \omega^2, \ldots, \omega^{q-1}\}$, where ω is a primitive element of the field GF(p^q), while the normal basis is the set $\Psi = \{\psi, \psi^p, \psi^{p^2}, \ldots, \psi^{p^{q-1}}\}$, where ψ is a generator of the basis. Although the normal basis is considered as more suitable for squaring or multiplying two numbers [25], we choose for our implementation to work entirely on the standard basis in order to avoid conversions when data are exchanged between the accelerator and other hardware modules. This is because standard basis represents numbers in the same way as fixed-point representation does and the power consumption of the conversion unit is comparable to the power consumed for arithmetic operations [26].

In general, GFs play an important role in several key parts of modern communication systems, such as forward error correction and cryptographic schemes. Since digital computing machines use Boolean logic, the binary field GF(2)=$\{0, 1\}$, and its extension fields GF(2^q), are widely used, due to the direct map between their elements and the Boolean values. In the rest of this work we consider only binary fields.

3.2.2 Addition over Galois Fields

As mentioned previously, each element from GF(2^q) can be represented as a q-bit vector or polynomial of degree $(q-1)$. Adding two elements is equivalent of adding the coordinates of each vector, or adding the two polynomials, using GF(2) arithmetic. This means that the implementation of addition over GF(2^q) is equivalent to a q-bit-wise XOR operation. According to finite field definition and its properties, the result of addition is an element of the field, with the same q-bit representation; no carries or overflow issues exists in GF operations (the same hold for multiplication, as explained in the next subsection). This makes finite field operations simpler to implement, faster and less energy demanding compared to standard arithmetic operations.

In order to visualize this comparison, we summarize in Table 3.1 the area, power and delay requirements of a standard (a q-bit carry ripple adder) and a GF adder. In this table, area is calculated as the required number of gates, power is approximated to be analogous to area and delay is considered the maximum time for each operation (critical path), assuming that AND, OR and XOR gates have the same area, power and delay.

Table 3.1: Comparison between Standard and Galois Field addition in terms of area, power and delay.

q-bit arithmetic		Area	Power	Delay
Addition	*Standard*	$5q$	$5q$	$2q+1$
	Galois Field	q	q	1

3.2.3 Multiplication over Galois Fields

Using standard basis representation, multiplication over GF(2^q) of two elements, b and c, can be computed as:

$$D(x) = (B(x)C(x)) \textbf{ mod } p(x), \tag{3.2}$$

Algorithm 1 : Pseudocode of Rijndael's algorithm, multiplying two operands over $GF(2^q)$ with primitive polynomial $p(x)$.

Input: $oper1$, $oper2$
Output: $prod$

```
1: temp_op1 = oper1;
2: temp_op2 = oper2;
3: prod = 0;
4: for i = 0 : 1 : q do
5:    if temp_op2[0] = 1 then
6:       prod = prod xor temp_op1;
7:    end if
8:    temp_bit = temp_op1[q − 1];
9:    temp_op1 = shift_left(temp_op1, 1);
10:   if temp_bit = 1 then
11:      temp_op1 = temp_op1 xor p(x);
12:   end if
13:   temp_op2 = shift_right(temp_op2, 1);
14: end for
```

where $p(x)$ is primitive polynomial of the field and $B(x), C(x)$ are the polynomial representations of b and c respectively. As we can see, GF multiplication is equivalent of polynomial multiplication followed by polynomial modulo reduction. These two operations can be performed both jointly and independently, leading to fully parallel, modular and standard multiplication architectures [27].

In general, there are several ways to implement a GF multiplication and the resulting performance is highly dependent on the underlying hardware platform. One approach is the use of look-up tables; under the appropriate representation basis, multiplication can be implemented by the following equation:

$$\text{oper1} \times \text{oper2} = exp(log(\text{oper1}) + log(\text{oper2})), \tag{3.3}$$

where *exp()* and *log()* are look-up tables.

An algorithmic approach to GF multiplication is the use of Rijdael's algorithm, shown in Algorithm 1, which is widely used in error correction and cryptographic applications. In this work, trying to minimize the energy consumption of NC and having the flexibility to create specialized hardware, we build a custom, low power

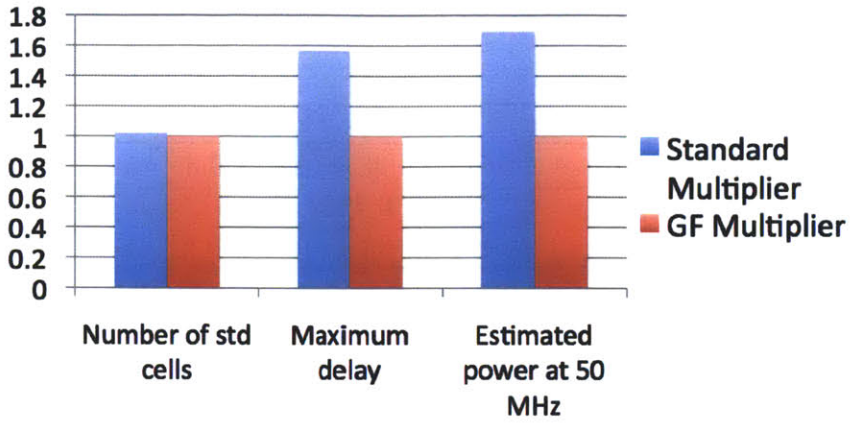

Figure 3-2: Standard array and Galois Field multipliers comparison. Area, delay and power metrics are shown, normalized in respect of the GF multiplier. Numbers are based on post-synthesis results, using Cadence RTL compiler.

GF multiplier, as a modified version of Rijndael's algorithm. In previous implementations of NC [20,22,24], the algorithm was implemented as an iterative process. Our proposed multiplier is implemented as a combinational circuit which computes the product of two elements in one clock cycle, implementing the algorithm in a fully parallel way. The reason for such a choice is that our design aims an ultra low power operation without high frequency requirements. Direct implementation of Rijndael's algorithm would result in approximately q times less critical path delay, but also in larger overall power consumption.

Figure 3-2 compares a standard 8-bit array multiplier[1] with our GF 8-bit multiplier. Different implementation metrics are used, having as baseline the performance of the Galois Field multiplier. Results are based on synthesized measurements using standard cells from TSMC 65nm CMOS process. Table 3.2 summarizes the implementation results of our multiplier, based on post-layout simulations using the same process.

[1]A standard q-bit array multiplier is the simplest type of multiplier, implemented by q^2 AND gates, q half-adders and $q(q-2)$ full-adders.

Table 3.2: Implementation results of our proposed 8-bit Galois Field multiplier.

Voltage	**0.4V**	**1V**
Critical path delay	28.9 nsec	0.65 nsec
Total power @ 10MHz	0.416 μW	41.34 μW
Leakage power	0.013 μW	5.41 μW

3.2.4 Division over Galois Fields

As described in Section 2.2, the decoding process requires inversion of the coefficients matrix and then multiplication of the result with the coded packets. Gauss-Jordan elimination, converting a matrix to its reduced row echelon form and calculating its inverse, involves successive divisions over a Galois Field. There are several ways to perform division over a finite field; some of them are based on the Euclidean and the Binary division algorithm, while the rest are look-up table based approaches. Exploiting the logarithmic propriety that converts division into subtraction, two *log* and one *exp* tables can be used, followed by a subtraction circuit, as shown in the next equation:

$$\frac{\text{oper1}}{\text{oper2}} = exp(log(\text{oper1}) - log(\text{oper2})). \tag{3.4}$$

However, instead of performing consecutive divisions during the Gauss-Jordan elimination between the pivot and the rest elements of each row, an equivalent way is to firstly calculate the inverse of the divider and then perform multiplications. From a circuit design perspective, this significantly reduces the overhead of implementing division, given that a highly efficient implementation of a GF multiplier exists. Moreover, potential re-use between the circuitry of the encoder and decoder would increase even more the benefits of this approach.

Iterative methods for calculating the inverse of an element are the extended Euclidean algorithm, shown in Algorithm 2, and Binary inversion algorithm, which is similar to the previous one but without the degree calculation task. Apart from these algorithmic methods, an other approach to calculate the inverse of an element over GF(2^q) is the use of a look-up table of size 2^q. Because of its simplicity and due to

Algorithm 2 : Pseudocode for calculating the inverse of an element over $GF(2^q)$, with primitive polynomial $p(x)$, using the extended Euclidean Algorithm.

Input: $oper1$
Output: inv

```
1: temp_op1 = oper1;
2: temp_poly = p(x);
3: inv = 1; temp = 0;
4: while temp_op1 ≠ 1 do
5:    j = degree(temp_op1) − degree(p(x))
6:    if j < 0 then
7:       temp_op1 = p(x);
8:       inv = temp;
9:       j = −j;
10:   end if
11:   temp_op1 = temp_op1 + x^j p(x);
12:   inv = inv + x^j temp;
13: end while
```

the fact that NC requires relatively small values of field size, look-up tables are a very attractive solution. Indeed, contrary to GF multiplication, calculating the inverse of an element during the decoding process using look-up tables is more energy efficient compared to an iterative approach because, firstly, Euclidean algorithm is more complex than Rijndael's algorithm and, secondly, the size of the look-up table should be $O(2^q)$, not $O(2^{2q})$ as required for multiplication.

3.3 Energy Modeling and Optimization of Network Coding

In this Section, we present the modeling of a typical communication scenario in BANs and the energy consumption of NC, in order to specify optimum algorithmic parameters, such as field size, taking into consideration the total system's energy, including processing and transmission energy.

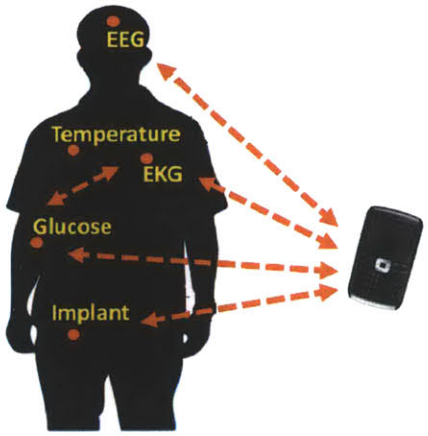

Figure 3-3: A typical Body Area Network with sensors around and in the human body communicating with a base station.

3.3.1 System Model

As shown in Figure 3-3, a typical BAN is composed by multiple sensors sending health related information to a base station. Most of the communicated data are from the sensors to the base station, with only a few control packets flowing in the opposite direction. This means that the nodes, which are the most critical parts of the network in terms of energy consumption, encode and send information, and the base station collects and decodes the mixtures during the largest portion of their operation. Even in the case of having more than one hop to the base station (i.e. allowing a relay node to help with the transmission of data), the intermediate node do not need to decode the data, since NC does not follow the end-to-end coding paradigm; intermediate nodes are allowed to re-encode on the fly packets without decoding them first, and only final destination nodes have to decode the mixtures.

3.3.2 Energy Modeling and Optimization

For the system described earlier, it is desirable to minimize the total energy consumption (E_{TOT}) for the critical parts of the network (i.e. the sensor nodes), given by Equation 3.5.

$$E_{TOT} = \sum_{i=1}^{N} E_{TOT}^{node_i}. \tag{3.5}$$

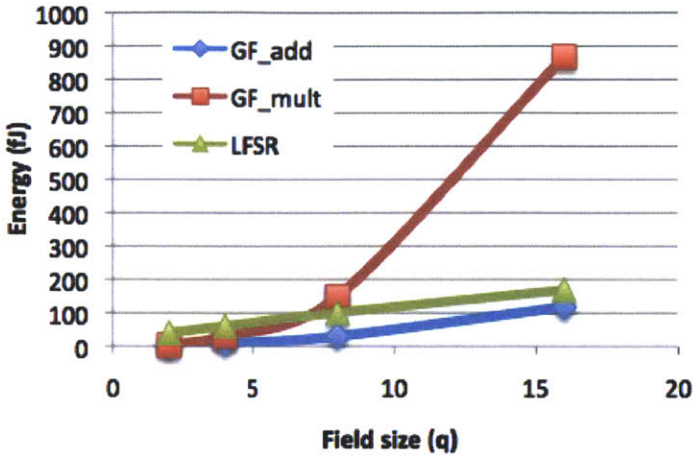

Figure 3-4: Energy per operation for different values of q consumed by a Galois Field adder, multiplier and LFSR.

Assuming that only direct links to the base station exist (Star topology), minimizing the total energy is equivalent of minimizing the energy for each node separately. The total energy consumption of a node ($E_{TOT}^{node_i}$) is given by Equation 3.6, considering the energy consumed for coding ($E_{COD}^{node_i}$) and transmission of packets ($E_{TX}^{node_i}$) per node.

$$E_{TOT}^{node_i} = E_{COD}^{node_i} + E_{TX}^{node_i}. \tag{3.6}$$

Assume that a node wants to transmit n packets, each of them with length L bits, using RLNC over GF(2^q). The encoding process is equivalent of generating a new packet as a linear combination of the existing blocks, weighted according to some random coefficients. The required energy per packet for the encoding process (E_{COD}^{pack}) is given by the following equation:

$$E_{COD}^{pack} = nE_{LFSR} + \frac{L}{q}(nE_{MULT} + (n-1)E_{ADD}), \tag{3.7}$$

where E_{MULT} and E_{ADD} is the energy consumed per multiplication and addition, respectively, and E_{LFSR} is the energy consumed for generating a q-bit coefficient using a LFSR (Linear Feedback Shift Register). In Figure 3-4 is shown how the choice of field size affects the energy for each operation. The energy numbers presented have been obtained after modeling every circuit component using Verilog, synthesizing

and performing post-layout simulations with a TSMC 65nm CMOS process, using standard VLSI CAD tools, such as Cadence RTL Compiler, Cadence Encounter and Nanosim. In this comparison, we keep the processing data rate same since, doubling the field size q results in doubling the critical path delay of the GF multiplier, which is used as the reference time period for our circuits; in other words, the ratio of the number of processed bit per operation over the required time for each operation, remains constant. The energy consumed by the multiplier increases much faster compared to the adder and the LFSR, as expected.

Apart from the processing energy, field size also affects the probability of two coded packets being linearly dependent and resulting in a non-successful decoding, requiring the transmission of extra coded packets. It can be shown that the expected number of transmitted packets until the reception of n linear independent combinations by the base station, using RLNC over GF(2^q) and assuming perfect links, is given by Equation 3.8.

$$\bar{n} = \sum_{i=1}^{n} \frac{1}{1 - (\frac{1}{2^q})^i}. \tag{3.8}$$

Now it becomes clear that a small field size lowers the required processing energy but results in extra retransmitted packets, increasing the total system's energy. This trade-off can be further explained by considering the expected number of transmitted packets ($\bar{n}$) and examining the expected total energy per node ($E_{TOT}^{node_i}$):

$$E_{TOT}^{node_i} = \bar{n}(E_{COD}^{pack} + E_{TX}^{pack}), \tag{3.9}$$

where E_{TX}^{pack} is the transmission energy per packet.

The total normalized system energy, including both processing for NC and transmission energy, is plotted in Figure 3-5. Due to the low data rate and the latency constraints of typical applications supported by BANs, the generation size is expected to be small and the packet length medium to small. In our analysis, we assuming that 8 packets are coded together, each of 1KB length, and a transmission energy per bit of 500 pJ/bit, which is close to the state of the art low power transmitters [28].

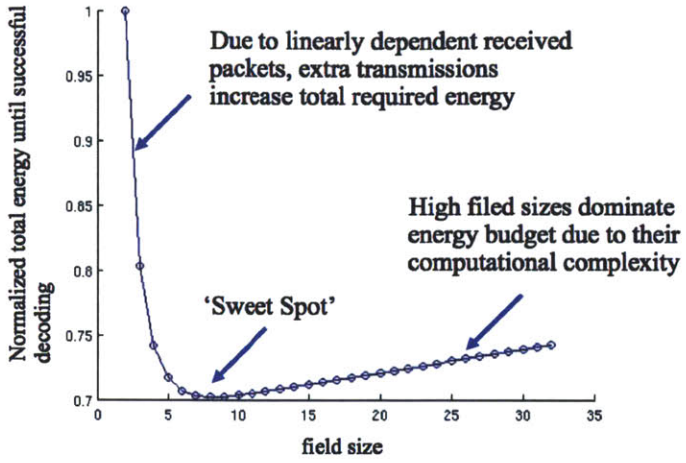

Figure 3-5: Normalized total system's energy of a node transmitting coded packets to the base as a function of the field size, using RLNC over GF(2^q).

Examining the plot we confirm that, when a small field size is used, the total system's energy in dominated by the extra radio energy due to packet retransmissions. However, as field size becomes large, increased energy is required for performing the coding process, without significantly affecting the expected number of transmitted packets, resulting in higher system's energy consumption. Field size values close to eight ($q = 8$) are around the optimum value, given the low data rate and low power constraints of examined scenario.

3.4 Implementation Results

In this subsection, the architecture and implementation results of our energy-aware VLSI implementation of Network Coding are presented. Based on the analysis of the previous section, we use GF(2^8) arithmetic and we assume that the maximum number of coded packets together is eight.

Our encoder architecture is shown in Figure 3-6. It is a parallel implementation of the encoding process and it composed by memory modules, Linear Feedback Shift Registers (LFSR), GF multipliers and adders. The memory is organized as multiple one-port modules, each of them with 1KB maximum length and 8-bit width. The LFSRs are responsible for creating the pseudo-random coefficients and they are im-

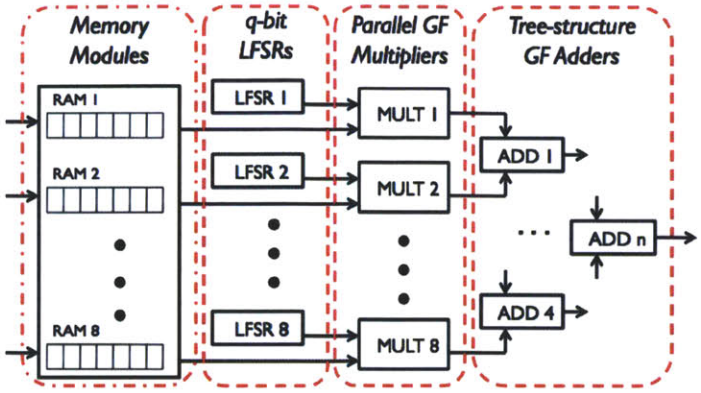

Figure 3-6: Block diagram of the proposed Network Coding encoder.

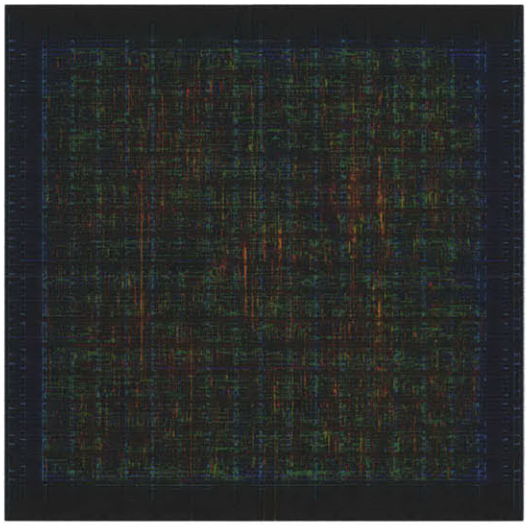

Figure 3-7: Layout of the proposed Network Coding accelerator.

plemented as 8-bit shift-registers with some extra logic gates at the feedback path. LFSRs of maximum length (maximum sequence of possible states) can provide coefficients which result in independent linear coefficients with very high probability. Design considerations of GF multipliers and adders were covered in the previous sections.

In order to achieve low power operation, voltage scaling is applied. The functionality of all components separately and the whole encoder was successfully verified down to 0.3V. Clock gating is also used in order to save energy from idle components. Finally, energy scalability is achieved by designing the encoder in such a way that it is able to encode different number of packets together with the same efficiency.

Table 3.3: Implementation results of our energy-aware, custom Network Coding accelerator encoder, using a TSMC 65nm process.

	Accelerator performing NC encoding	
Supply Voltage	$0.4V$	$1.0V$
Frequency	10 MHz	250 MHz
Throughput	80 MB/s	2 GB/s
Power	22.15 uW	10.98 mW

Multiplexers and AND gates are inserted between the LFSRs and the GF calculation components in order to minimize non-desired propagation of glitches. Table 3.3 summarizes the most important simulations results of our proposed encoder for two different supply voltages.

Chapter 4

Harnessing Partial Packets with Network Coding

In this chapter we present a new partial packet recovery mechanism, called PPRNC. PPRNC is based on Network Coding and is designed as a data link layer process, without requiring access to PHY's soft information. It can extract useful information from partial packets, dealing with the inefficiency of discarding packets which include a few erroneous bits, resulting in considerable reduction in the number of retransmitted packets. Although initially designed for resource constrained networks, such as BANs, improving their resource utilization, it can also be used in modern, high-speed wireless mesh networks, such as WLANs, improving their total throughput. The design of the recovery mechanism, as well as simulation and implementation results are presented and discussed in the next sections.

4.1 Motivation

Transmitting data over wireless channels has some fundamental differences compared to wired transmission. Channel attenuation, fading, noise and interference degrade signals' quality, often resulting in erroneous received bits. However, the vast majority of current wireless networks tend to imitate a wired network model and operate in an all-or-nothing mode, using CRC checks and other error detection codes in order to

determine the validity of received packets, before forwarding them to higher layers. Thus, in a typical wireless system, a packet, after its reception, usually followed by a FEC decoding step, is discarded if its number of erroneous bits is equal or greater than one, in order not to propagate erroneous information to higher layers of the protocol stack. Throughout this thesis, we call these packets *partial*, contrary to ones satisfying their CRC rule, called *valid*.

However, dropping partial packets and requesting a retransmission of the entire packet is inefficient in terms of resource utilization, since partial packets have usually large amount of useful information, especially in wireless networks. Lower resource utilization and increased overall power consumption is the result of not exploiting the information received during the first transmission. For this reason, several techniques have been proposed to eliminate this inefficiency.

4.1.1 Overview of the proposed scheme

In this thesis, we propose a new partial packet recovery framework, called *Partial Packet Reception with Network Coding* (PPRNC). PPRNC is based on Network Coding (NC) and is a physical layer agnostic recovery mechanism. This means that it is designed as a data link layer process and it can be efficiently incorporated in any firmware or driver of current wireless cards since it is physical layer independent and can also be transparent to higher layers.

The use of NC in wireless networks has been shown to provide great advantages in terms of throughput and data robustness [18,29,30]. PPRNC extends these results and uses NC also for leveraging information contained in partial packets. According to our simulation and implementation results, it can push even higher the aggregate network capacity by reducing the required number of retransmitted packets and increase the effective link reliability. The benefits of any partial packet recovery technique depends on the portion of received partial packets compared to total number of transmitted ones, which, in turn, depend on protocol and channel specific parameters; in this thesis we present and implement a general partial packet recovery framework, designed to be incorporated in a wide variety of wireless networks, such as BANs and

Figure 4-1: Types of networks in which PPRNC may be applicable, providing significant energy and throughput benefits through its efficient way of reducing packet retransmissions.

WLANs, with the main characteristics outlined in the next lines and explained in the following sections:

- PPRNC does not require any physical layer soft information to be exposed to it, making the proposed algorithm easily deployed in current wireless networks. Actually, it treats physical layer as a black box, receiving its output as the best estimate about the transmitted information, without having any knowledge about its implementation details (interleaving, FEC code, etc.).

- The recovery mechanism introduces no transmission overhead for correctly received packets. If a packet does not satisfy the CRC check, it is buffered and automatically becomes redundancy. No extra information is transmitted per packet, such as block-based CRCs or known pilot bits, that could potentially indicate the erroneous parts of a packet.

- PPRNC considerably simplifies the feedback process, since the receiver does not have to explicitly declare the parts of packets or even the specific packets which got corrupted during the transmission, making the amount of information sent by the receiver as feedback minimum.

- The computational requirements of PPRNC's recovery algorithm can be easily

adapted to the available resources, balancing the recovery performance with the algorithmic complexity.

We analyze the performance of our proposed recovery mechanism and we present our simulation results, providing useful insight about the choice of different algorithmic parameters. We test PPRNC in a very simple communication setup composed by only two wireless nodes. We make use of TI Development Boards, equipped with CC2500 2.4 GHz RF radio modules, to create real data-traces. We present the results of processing the collected traces off-line with PPRNC's recovery algorithm, comparing also its performance with other reported partial recovery mechanisms.

Our simulation and implementation results reveal the following findings:

- Exploiting information contained in partial packets can significantly increase (close to a factor of 2×) the throughput of a wireless communication pair, especially under challenged channel conditions.
- PPRNC outperforms the ability of proposed partial packet recovery mechanisms to extract useful information from partial packets, which can result in approximately 20% reduction in the required number of retransmitted packets in highly challenged environments.

4.2 Related Work

Trying to increase the energy efficiency of resource constrained networks and throughput of wireless mesh networks, the exploitation of correct information contained in partial packets has attracted a lot of attention. In the following paragraphs, we present some of the reported approaches.

4.2.1 Forward Error Correction

Firstly, one widely used approach for increasing the reliability of links in wireless networks is the use of FEC codes [31]. According to this technique, apart from the information bits, extra redundancy is inserted in transmitted packets, enabling the

receiver to successfully recover the transmitted information, only if a maximum number of errors has occurred. Different channel conditions require different amount of redundancy. Having a physical layer equipped with a powerful FEC scheme could potentially eliminate the need of retransmissions. However, since, in general, the wireless channel is not known a priori, designing codes for worst-case scenario can be quite inefficient because, most of the time, extra bits are transmitted but not used at the recovery process, reducing the effective transmission rate. Also, the incorporation of channel measurement mechanism in a system for appropriately adapting the amount of redundancy to different channel conditions is expensive, especially when resource-constrained networks are considered, such as BANs, or even inefficient when fast varying channels are used.

FEC codes are traditionally designed to operate very well in an end-to-end fashion, but they can not exploit with the same efficiency information existed in partial packets. Authors in [32] use a rate 1/2 Hamming code to create redundancy for transmitted information and an incremental CRC algorithm to combine the received erroneous packets. In [33], authors take advantage of spatial diversity to create an equivalent repetition code and, using a modified version of majority voting algorithm, try to recover the transmitted packets.

4.2.2 H-ARQ schemes

An other popular approach, adopted in several modern cellular networks [34,35], is the use of hybrid ARQ (H-ARQ) schemes [36], which combine the advantages of both FEC codes and packet retransmissions (ARQ). Soft information from the physical layer corresponding to incoming and previously received partial packets can be combined in an incremental way, increasing the probability of a successful reception.

According to Chase Combining H-ARQ [37], when an erroneous packet is received, the same copy of it is retransmitted; the receiver stores and combines soft information from the physical layer across these transmissions, increasing the effective SNR of the specific packet. A different approach is the Incremental Redundancy H-ARQ [38]. According to this technique, different coded versions of the received partial packet

or extra amount of redundancy is transmitted, gradually increasing the correcting ability of a FEC scheme.

4.2.3 Cross-Layer schemes

Although the previous schemes have considerably improved the reliability of wireless links and increased the error correction capability of the physical layer, several mechanisms have been proposed to operate on top of them, providing significant performance benefits. ARQ and multiple CRCs within a packet are used in [39], where a data link layer mechanism for sensor networks was presented, reducing the retransmission traffic. However, transmitting multiple CRCs per packet results in a fixed overhead, even for correctly received packets, and specifying the position of erroneous bits in a feedback packet may require large amount of information.

An other cross-layer approach is the use of physical layer confidence metrics, such as soft information or confidence hints, as shown in Figure 4-2. These metrics are provided by the physical layer to higher layers, in addition to its estimate about the received information, to express how close the bit passed to them was to the received channel value corresponding to that bit. All these metrics will be called as *PHY soft information* in the rest of the thesis. In [40] authors presented PPR, a system which annotates every decoded bit from the physical layer with a confidence hint, in order to enable the link layer to identify which chunks of a packet have high probability of being erroneous and request only these parts. An other protocol is presented in [41], called SOFT, which takes advantage of wireless spatial diversity and uses soft information from the physical layer to combine and recover corrupted packets. However, recovering mechanisms using PHY soft information violate the standard layering abstraction and can not be deployed in current wireless network cards (to the best of our knowledge, none commercial wireless card exposes symbol-level PHY soft information to higher layers).

The closest work to our proposed approach is by Lin *et al.*, who presented a partial packet recovery mechanism (ZipTx) [42] for IEEE 802.11 networks that does not require access to PHY soft information. Instead, a systematic block code is

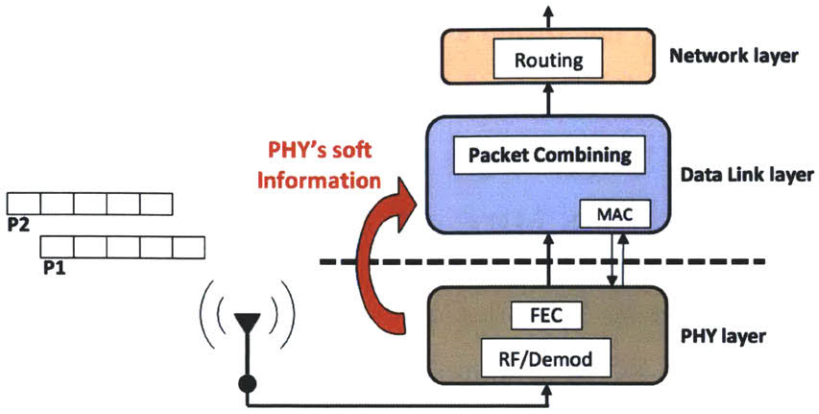

Figure 4-2: Block diagram of a protocol stack in a receiver, where higher layers use soft information directly from the PHY.

Table 4.1: Summary of partial packet recovery mechanisms.

	Zero Overhead for correctly received packets	PHY independent
H-ARQ [37, 38]	YES	NO
Block-based CRCs [39]	NO	YES
PPR [40]	YES	NO
SOFT [41]	YES	NO
ZipTx [42]	NO	YES
PPRNC	YES	YES

used per packet and the redundant information is sent in two rounds, imitating an Incremental Redundancy H-ARQ scheme. In addition, known pilot bits are inserted in transmitted packets in order to enable the receiver to estimate their BER.

PPRNC is also a software-based recovery mechanism, in the sense that does not require access to PHY soft information. However, since PPRNC does not include per-block CRCs or known pilot bits, a challenging question arises; how the recovery mechanism will distinguish between the correct and erroneous bits within a partial packet? PPRNC is able to identify correct and erroneous parts of partial packets with an elegant mechanism based on NC, described in Section 4.4.2. In addition, using an iterative algorithm, it tries to correct the erroneous parts according to information contained in other valid packets, since the coding process of PPRNC is

performed across packets and not within a packet, as in the referenced works. Table 4.1 summarizes several partial packet recovery mechanisms and their characteristics.

4.3 System Architecture of PPRNC

PPRNC's core idea is based on Network Coding (NC). Several research works have analyzed the advantages of NC in communication networks, both theoretically and through real implementations [18, 29, 30]. In this thesis, we take a slightly different direction and we extend the use of NC to take advantage of its implicit error detection and correction capability, to further increase its potential benefits to a network.

Considering a typical communication pair using NC, the receiver's physical layer captures the RF signal corresponding to a transmitted coded packet or degree of freedom (dof) and, after the demodulation and FEC decoding step, pushes up its best estimate about the received information to data link layer. If this estimate satisfy a checksum rule, it is stored and marked as valid dof, otherwise it is discarded. The NC decoding process starts as soon as the number of valid dofs becomes equal to the generation size, as explained in Section 2.2 and shown in Figure 4-3a. After the successful decoding, the initial packets can be delivered to higher layers of the protocol stack.

However, a communication system leveraging the information contained in partial packets, could potentially start the NC decoding process with less valid dofs, as shown in Figure 4-3b. PPRNC is such a mechanism, designed as a data link layer process, which can be easily incorporated into any communication protocol. In this work we restrict our analysis and implementation results considering a trivial network topology, with only a source and a destination. In this toy example, previously proposed NC techniques could not offer any advantage in terms of retransmitted packets. Thus, any potential benefit from the use of PPRNC in this scenario would be due to its novel characteristic of exploiting NC's error detection and correction capability.

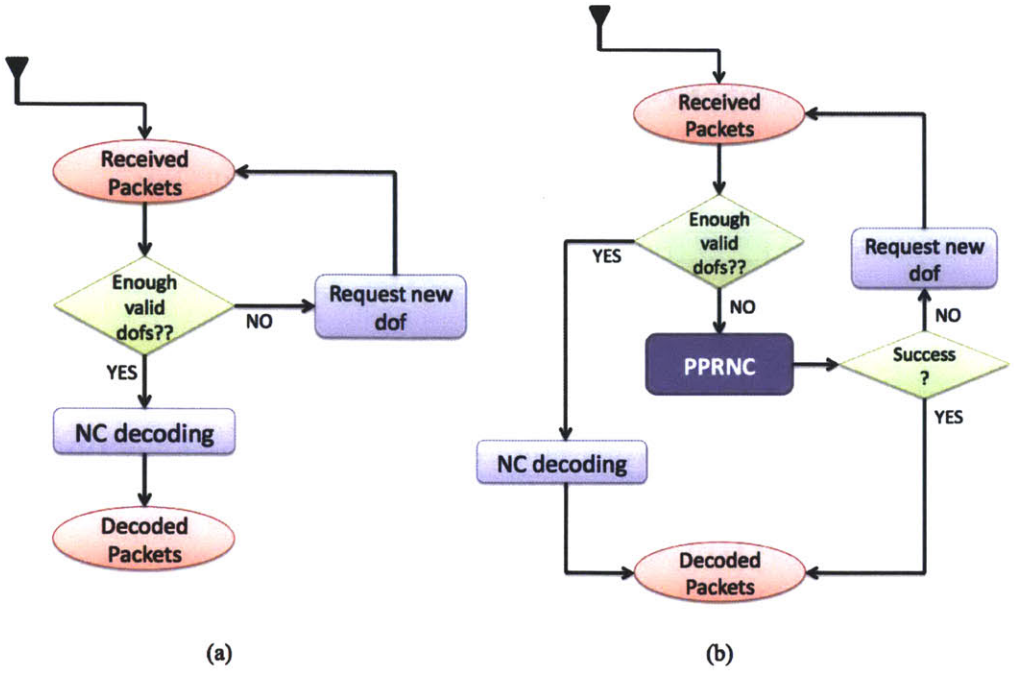

Figure 4-3: Simplistic flow chart of two systems: (a) Using traditional Network Coding and waiting the reception of the required number of valid coded packets, (b) Attempting to decode the received packets even when the number of valid packets is less than the required. This can be possible because PPRNC makes use of the information contained in partial packets.

4.3.1 PPRNC in a Protocol Stack

A simplistic block diagram of a potential protocol stack in a receiver that uses PPRNC is shown in Figure 4-4. The transmitter is not shown, since it is assumed that the standard NC encoding process is used in its data link layer. As mentioned earlier, our recovery mechanism does not require access to any PHY soft information. This means that PPRNC is physical layer independent, i.e. any modulation, FEC or H-ARQ scheme can be used in the lower layers. It can also be transparent to higher layers; the only consideration is that packets are delivered to higher layers in a batch mode, right after a successful decoding. If the CRC rule is not satisfied for a specific estimate of the physical layer, PPRNC buffers this packet. The recovery algorithm tries to correct its erroneous bits and start the NC decoding process, even when the number of valid dofs is less than the generation size.

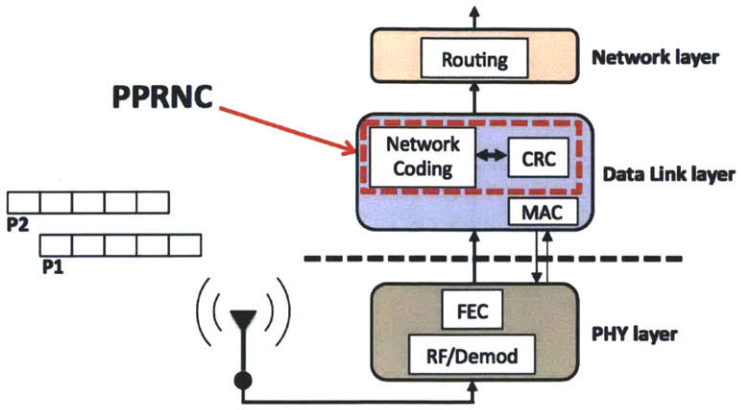

Figure 4-4: Block diagram of a protocol stack in a receiver using PPRNC.

4.3.2 FEC, H-ARQ and PPRNC

PPRNC's main advantage comes from the fact that the coding process is performed across packets using NC, a rateless coding technique, contrary to traditional FEC and, the more recently proposed, H-ARQ schemes, which attempt to recover individual packets. Although a direct comparison of these scheme would be considered as subtle and unfair, since they belong to different layers, it would illustrate the different characteristics of each approach. FEC schemes use a fixed amount of redundancy to recover a packet, even when it is not needed. H-ARQ techniques selectively ask for extra information when received packets are corrupted, accumulating also information across retransmissions to decode a specific packet. In PPRNC, when the physical layer returns a partial packet, it is buffered by the link layer and automatically becomes redundancy for all the other packets across its generation. However, when packets are received without errors, no extra overhead is associated with them. The challenging questions of how the receiver will be able to detect the wrong parts of partial packets and how it will correct them are analyzed in the next Section.

4.4 Harnessing Partial Packets

In this Section, PPRNC's recovery algorithm is described, providing also pseudocode for its main functions. In order to visualize PPRNC we consider the simplest form of a network, composed by two nodes; a source, which has to transmit a certain number

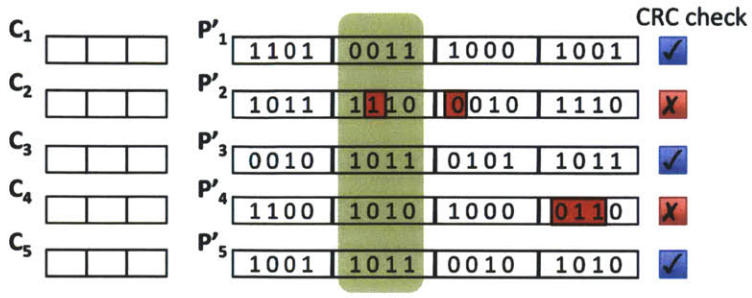

Figure 4-5: Received packets and their associated coefficients. Erroneous bits are marked with red color.

of packets, and a destination. In this case, if the communication pair uses ARQ, partial packets are discarded at the destination and requested to be re-sent by the source. If N packets have to be transmitted through a link with erasure probability p, then the expected number of transmitted packets is:

$$\bar{N} = \frac{N}{1-p}. \tag{4.1}$$

Making use of NC wouldn't reduce the expected number of transmitted packets. We intentionally adhere to such a simple scenario in order to decouple PPRNC's advantages with already reported benefits of NC in wireless mess networks. Considering a general case with a multi-terminal wireless network, the benefits would be multiplicative.

4.4.1 Recovery Algorithm

As mentioned earlier, an ARQ-like operation of a communication pair using NC would be the following: when the number of valid packets becomes equal to the generation size, the NC decoding process, having enough valid dofs, can recover the initial packets, using the equation:

$$P\_init = C\_val^{-1} * P'\_val, \tag{4.2}$$

Algorithm 3 : Pseudocode of the link layer process running at the receiver, triggered when a new packet is received.

Input: *pack_received*
Output: $P\_init \equiv Initial\_packets$

```
1: #valid_packs = 0, count = 0;
2: while #valid_packs < Gen_size do
3:    p = pack_received;
4:    count++;
5:    P'_inc = [P'_inc; p];
6:    #valid_packs = check_CRC(P'_inc);
7:    if count > Gen_size then
8:       [success, P_init] = Recover(P'_inc);
9:       if success is true then
10:         return P_init;
11:      end if
12:   end if
13: end while
14: P_init = NC_Decoding(P'_inc);
15: return P_init;
```

where P_init are the initial packets, P'_val are the valid coded packets and C_val are the coefficients associated with them. Assume that the packets shown in Figure 4-5 are transmitted (P'_1, ..., P'_5) and the channel corrupts some of their bits. In this particular example, we assume that the generation size equals three, the Galois Field $GF(2^4)$ is used and every packet has 16 bits. The standard NC decoding process will be initiated right after the reception of P'_5 (P'_1,P'_3 and P'_5 are the three required valid dofs). However, in this way information contained in packets P'_2 and P'_4, which are the partial packets, is not exploited. In the following paragraphs, we describe how PPRNC takes advantage of this information, and in Section 4.6.2 we present its performance benefits.

As soon as the total number of incoming packets (including valid and partial) becomes greater than the generation size, PPRNC tries to start the decoding process, utilizing the partial packets, as described in Algorithm 3. For instance, assume that only the first four packets of Figure 4-5 have been received ($P'_1, \ldots, P'_4$). PPRNC processes these buffered packets (P'_inc), working across columns, and finishes the recovery process only when all columns are successfully decoded, as shown in Algo-

Algorithm 4 : Pseudocode of the partial packet recovery process (**Recover**).

Input: $P'\_inc \equiv Incoming\_packets$
Output: $success, Recovered\_packets \equiv P\_recov$

```
1: success = 1;
2: count = 1;
3: while (count ≤ #cols_P'_inc)&&(success ≠ 0) do
4:    [flag, P_recov_col] = Recover_column (P'_inc(:, count));
5:    if (flag ≠ 0) then
6:       P_recov = [P_recov, P_recov_col];
7:       count++;
8:    else
9:       success = 0;
10:   end if
11: end while
12: return [success, P_recov];
```

rithm 4. Otherwise, it declares a failure and a new packet is requested. Thus, for this example, there is a certain probability that PPRNC would be able to recover the three initial packets based on the two valid (P'_1, P'_3)and two partial packets (P'_2, P'_4).

Considering the first column of our example, i.e., the first symbol of every packet, since no error has occurred, the receiver should ideally be able to recover this part of the packets. But, since P'_2 and P'_4 do not satisfy the CRC rule, the receiver does not know if p'_{21} and/or p'_{41} are erroneous. PPRNC tries to check the validity of these symbols using a *consistency check* rule, which is based on NC and analyzed in Section 4.4.2. Since symbols of the first column are correct, the result of the check is positive and the symbols are considered correct. So, the standard NC decoding process can be initiated with any three out of the four correct symbols and their associated coefficients, recovering the first column of the initial packets.

After a successful column recovery, PPRNC continues with the next one. Considering the second column, the consistency check is not satisfied due to the error at the second packet. Since the receiver knows that P'_2 and P'_4 contain errors, an iterative *correction process* is initialed, analyzed in Section 4.4.3, trying to identify the correct values of symbols p'_{22} and p'_{42}. The consistency check will be satisfied only when the correction process finds the correct symbol values. Then, the NC decoding process can be initiated, recovering the symbols of the second column. PPRNC, after

Algorithm 5 : Pseudocode of the per column recovery process (**Recover_column**).

Input: $coef, Incoming\_col \equiv Inc\_col$
Output: $flag, Recovered\_col \equiv Recov\_col$

```
1: [val_col, inv_col] = seperate(Inc_col);
2: [val_coef, inv_coef] = seperate(coef);
3: test_col = [val_col; inv_col(1 : Gen_size − #val_col)];
4: test_coef = [val_coef; inv_coef(1 : Gen_size − #val_col)];
5: [check, gues_col] = consistency_check (test_col, test_coef);
6: if check is true then
7:    Recov_col = gues_col;
8: else
9:    count = 0;
10:   while (check is false) && (count < limit) do
11:      test_col = correction_process (test_col);
12:      [check, gues_col] = consistency_check (test_col, test_coef);
13:      if check is true then
14:         Recov_col = gues_col;
15:      else
16:         count + +;
17:      end if
18:   end while
19: end if
20: return Recov_col;
```

specifying the correct values of the examined symbols, updates their CRC status, potentially reducing the number of packets marked as partial at the next steps of the recovery algorithm and significantly speeding up the recovery algorithm. In our example, since P_2' and P_4' contain also other errors, their CRC status is not changed after the end of second column's recovery. Working in the same way for the next columns, the described algorithm can successfully recover the initially transmitted packets.

4.4.2 Consistency Check Rule

Detection of corrupted information in partial packets is one of the key novel characteristics of PPRNC. In [40, 41], PHY soft information is used in order to identify erroneous parts of packets, whereas in [39], every block of packet has its own CRC. However, since PHY soft information is not always available to higher layers and the

block-based CRC approach results in extra overhead for every transmitted packet (even for the successfully received ones), PPRNC checks the validity of symbols in received packets through a consistency check rule based on NC. The rule exploits a NC property, according to it any set of enough valid dofs can be decoded, recovering the initial packets. Then, given the coefficients associated with an other coded packet, the rule can examine their correctness by regenerating the packet and comparing it with its current value.

For instance, consider the first symbols of packets P'_1, P'_2 and P'_3, namely p'_{11}, p'_{21} and p'_{31}. Assuming[1] that they are correct and performing the NC decoding process:

$$\begin{bmatrix} p^*_{11} \\ p^*_{21} \\ p^*_{31} \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \end{bmatrix}^{-1} \times \begin{bmatrix} p'_{11} \\ p'_{21} \\ p'_{31} \end{bmatrix}, \tag{4.3}$$

we guess the first symbol of initial packets P_1, P_2 and P_3. Multiplying these symbols with the coefficients associated with P'_4, we get:

$$p'^*_{41} = C_4 \times \begin{bmatrix} p^*_{11} \\ p^*_{21} \\ p^*_{31} \end{bmatrix}. \tag{4.4}$$

The final result p'^*_{41} is compared with the first symbol of the last packet, i.e. p'_{41}. If the comparison result is invalid, then an error exists in p'_{21} and/or p'_{41} (p'_{11} and p'_{31} belong to valid packets so, they are considered correct). Otherwise, with high probability, the first column of packets does not contain erroneous symbols and the guessed symbols (p^*_{11}, p^*_{21} and p^*_{31}) are correct. In the latter case, there is also a probability for a false positive event (p_{fpe}), as with any other checking rule. Assuming that p'^*_{41} and p'_{41} are independent, the false positive event rate depends only on the value of field size. So, by increasing the field size, we can set p_{fpe} under any desired threshold. An other

[1]In order to avoid confusion, we use the following notation: $'$ for coded symbols and packets, $*$ for guessed or estimated symbols and regular notation for initial packets.

elegant way to lower p_{fpe} is by running multiple consistency checks, called consistency check *rounds*, for the same column but considering different set of symbols each time. In our example, the second round could be performed by using p'_{11}, p'_{21} and p'_{41} at the NC decoding process and p'_{31} at the final comparison. We can continue performing consistency check rounds, decreasing p_{fpe} exponentially.

If q is the length of each symbol or equivalently, the size of the Galois Field used, and r is the number of consistency check rounds, the probability of a false positive event (p_{fpe}) of our check rule is:

$$p_{fpe} = (\frac{1}{2^q})^r, \tag{4.5}$$

This means that, by using a field size of four and two rounds of consistency check, p_{fpe} equals $1/2^8$, which is the same as the false positive event rate of 8-bit CRC rule. In Figure 4-6, the false positive event rate is shown for different values of field size and number of consistency check rounds. The desired value of p_{fpe} is an application specific choice and can be adjusted depending on the available computational resources and the desired algorithmic performance.

4.4.3 Correction Process

After detecting an erroneous column, the communication protocol should either request for retransmission of the symbols belonging to partial packets or trigger a correction mechanism to recover their initial values. Depending on the application's specific parameters, one of these approaches may be preferable. Since our framework's goal is to minimize retransmitted packets, the detection of an erroneous column initiates an iterative algorithm (lines 10-17, Algorithm 5), trying to correct the symbols of this specific column corresponding to partial packets. In our example, after the invalid result of second column's consistency check, the iterative process attempts to identify the correct values of the symbols contained in partial packets, namely p'_{22} and p'_{42}. A searching algorithm tries different possible combinations, starting with the ones of highest probability of occurrence, until the consistency is satisfied.

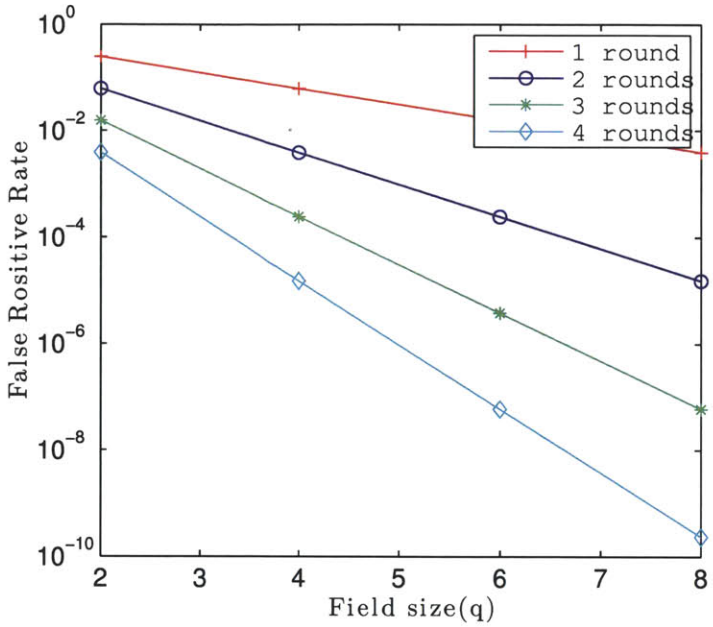

Figure 4-6: False positive event rate for different values of field size and number of consistency check rounds.

4.5 Experimental Measurements

In this Section, we provide details about our experimental setup and present some of our channel measurement results. Several research works have analyzed bit error characteristics and patterns of received packets in wireless networks [32, 40, 42]; we briefly present our findings, not as a complete error characterization study, but to give a clear idea about the experienced channel during our experiments and to reason the choice of our algorithmic parameters.

4.5.1 Experimental Setup

We used development boards from Texas Instruments [43] equipped with CC2500 RF radio modules, transmitting at the 2.4-2.483 GHz ISM band, shown in Figure 4-7. The boards communicated transmission data to a PC through USB connection. We also used *SmartRF Studio* to configure CC2500's registers and *uVision Programming Suite* to program microcontroller's firmware. Supported modulation schemes by the RF radio modules are OOK, 2-FSK and MSK, while the output transmission power

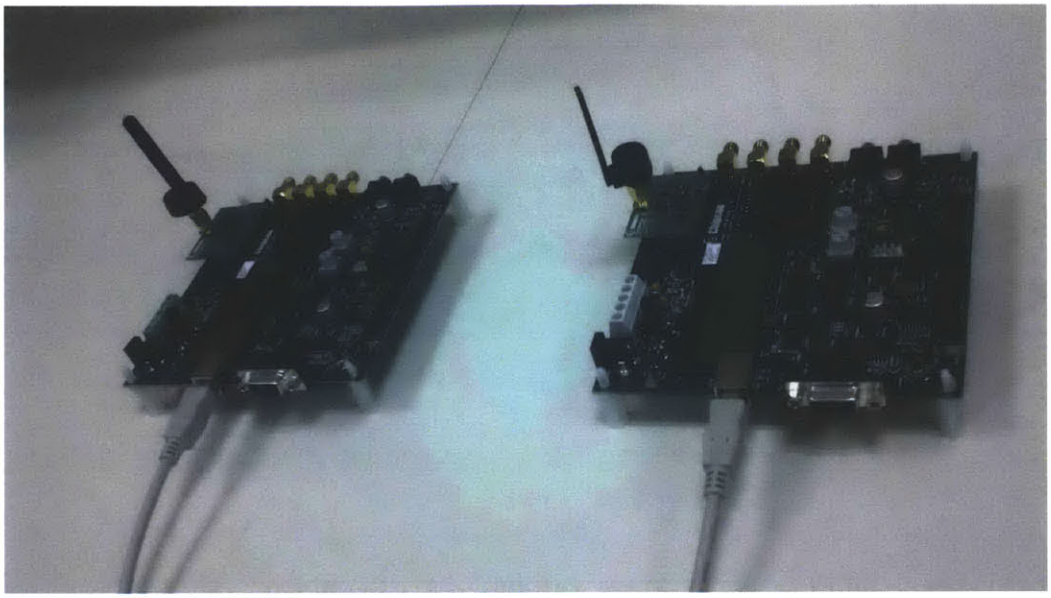

Figure 4-7: Development boards from Texas Instruments, equipped with CC2500 2.4 GHz radio modules.

can be programmed in the range of -30dBm to +1dBm, and the transmission rate up to 500KBaud.

For our experiments, we used FSK modulation, 250 kBaud transmission rate, variable output power as well as packet length. The packet format was the following: 6-byte preamble, 2-byte synchronization word, variable length data field and optional 2-byte CRC check. Transmitter and receiver had a 5m line-of-sight distance and they were located in a typical indoor campus environment, shown in Figure 4-8. At the physical layer, a rate 1/2 convolutional code and an interleaver were used. Depending on the experiment, we selectively activated and deactivated the 2-byte CRC field, in counting the valid and erased packets at the receiver. Measurements were taken across several days and hours; their average is calculated and presented in next subsection. Several traces were created by capturing transmission data and processed off-line.

4.5.2 Channel Measurement Results

Fading, noise, interference and other wireless channel characteristics degrade signal's quality and usually cause corruption of the transmitted information. Errors in the payload of the packets result in the reception of partial packets, whereas errors in

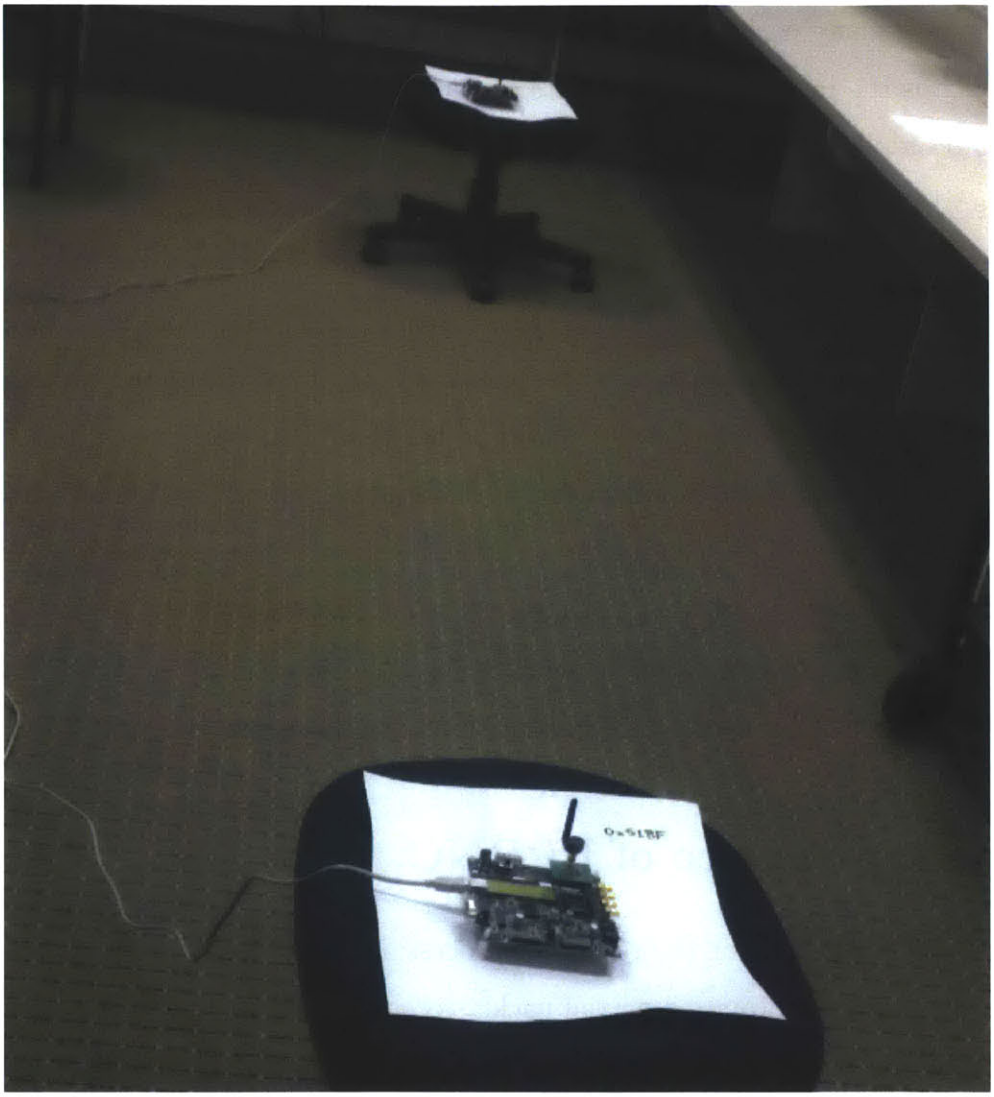

Figure 4-8: Indoor testing environment in a typical campus office.

the preamble or the sync word usually result in erased packets, i.e. no information is captured for these packets. The performance of any partial packet recovery mechanism depends on the portion of transmitted packets which are received as partial, because if the number of erased packets is large or if the number of partial packets is small, then the benefits of such a mechanism would be very limited. So it is critical to examine how many packets are received as partial across different channel conditions and which is the ratio of erased packets over the partial ones.

Based on our collected traces with the CRC check enabled, we count how many packets where successfully received. In Figure 4-9, we plot the probability of a packet being either erased by the channel or discarded as partial by the link layer, which is usually called erasure probability. Due to our static, short range communication link, we had to significantly lower transmitter's output power to challenge our system in the low SNR regime. For output power levels higher than $-10dBm$, only a few packets were not received and are not included in the graph. As expected, the probability increases as we lower the transmission power and transmit longer packets. In addition, our measurements indicate that the ratio of erased packets over the partial ones for various output power levels is very small (approximately $\approx 10^{-3}$). Thus, a partial packet mechanism would highly improve system's performance.

4.6 Performance of PPRNC

In this Section, we present the results of processing our collected traces. We implement in software our proposed partial packet recovery framework and we evaluate its performance offline. The reduction of retransmitted packets compared to a traditional scheme which discards partial packets clearly illustrates the amount of information contained in these packets. PPRNC is also compared with other partial packet recovery mechanisms in terms of their ability to exploit this particular kind of information.

4.6.1 Compared Approaches

We compare PPRNC with the following schemes:

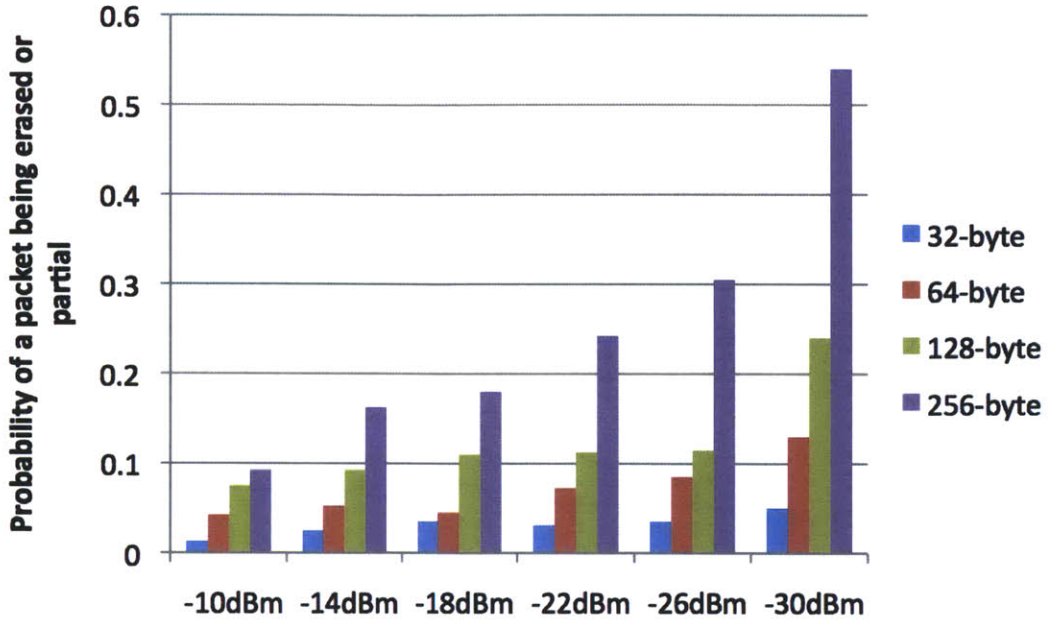

Figure 4-9: Average probability of a packet being erased by the channel or discarded after the CRC check, with varying packet lengths and output power levels.

- **ARQ**: This scheme requests a new retransmitted packet every time a partial packet reaches receiver's data link layer. Its simplicity is the main reason for its popularity, but also for its poor performance since partial packets are discarded.

- **Ideal H-ARQ**: We consider a hypothetical recovery mechanism which has the ability to recover any received packet with at most two transmissions. So, if a packet is received partial, this ideal recovery mechanism guarantees that, with its upcoming retransmitted copy, successful recovering will take place, even if the second retransmitted packet is partial. As it becomes obvious, this is the best case scenario for the family of partial packet recovery mechanisms performing coding within a packet. In this category lie mechanisms such as *Type I* and *II* H-ARQ schemes and other cross-layer techniques, such as ZipTx.

4.6.2 Performance Evaluation

In order to compare the previous schemes, we plot in Figure 4-10 the expected number of transmitted packets (including the retransmitted ones) required by the source to communicate 5 packets to the destination. The performance of ARQ scheme is

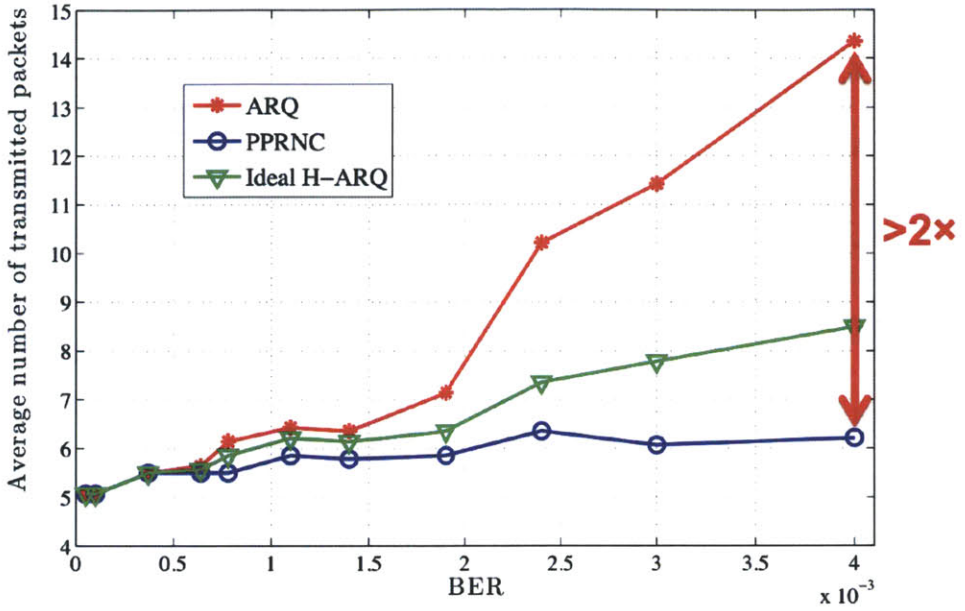

Figure 4-10: Performance comparison among ARQ, PPRNC and an ideal H-ARQ scheme.

expected to follow the curve of equation 4.1, while performance of the ideal H-ARQ scheme is expected to be much better. Indeed, the ideal H-ARQ scheme performs better, especially in the high BER regime.

PPRNC performs even better than this ideal scheme. With one or two extra partial packets, PPRNC can successfully recover the initial transmitted packets. The reduction in the total transmitted packets is more than a factor of two compared to the ARQ scheme, which represents the traditional all-or-nothing approaches discarding partial packets. Compared to the ideal H-ARQ scheme, the reduction is approximately 25% at the high BER regime, making PPRNC a very good candidate among the partial packet recovery mechanisms.

Our simulation and implementation results reveal the following findings:

- Exploiting information contained in partial packets can significantly improve the performance of wireless networks, resulting in better resource utilization and higher throughput.

- Our proposed recovery mechanism, called PPRNC, is a very good candidate

among the partial packet recovery mechanisms found in the literature. Based on our indoor channel measurements PPRNC outperforms the best partial packet recovery mechanism.

Chapter 5

Conclusion

This thesis examines the applicability of Network Coding in resource constrained wireless networks with battery-operated or even energy harvesting nodes. As a representative example of this type of networks, Body Area Networks (BANs) are considered. Potential feasibility of incorporating Network Coding into BANs would directly indicate its applicability in a wide range of wireless communication networks.

The main contributions of this thesis can be summarized in the following lines:

- A detailed energy modeling of the required operations by Network Coding is presented, based on VLSI measurements. This allows the precise estimate of the energy consumed by Network Coding and the detailed analysis of a system's energy budget.

- Based on the created energy models, a systematic approach for specifying optimum algorithmic parameters of Network Coding in a typical BAN scenario is presented.

- A low power accelerator is designed using a TSMC 65nm CMOS process, performing the Network Coding encoding process. Its power consumption indicates that Network Coding can be successfully incorporated in extremely resource constrained networks, such as BANs.

- A new partial packet reception mechanism based on Network Coding (PPRNC)

is proposed in order to exploit information contained in partial packets and minimize the number of retransmitted packets.

5.1 Future Directions

There are several exciting challenges related to Network Coding. A few of them, which are logical extensions of the current thesis and will be investigated in the future, are the following:

- Design and testing of a low power Network Coding accelerator, performing both the encoding and decoding process.
- Incorporation and simulation of PPRNC within a large scale testbed. Evaluation of a real time wireless network using PPRNC would result in significant performance benefits, both in terms of throughput and resource utilization, since already reported advantages of Network Coding would be combined with reduced number of retransmitted packets.

Bibliography

[1] A. Chandrakasan, F. Lee, D. Wentzloff, V. Sze, B. Ginsburg, P. Mercier, D. Daly, and R. Blazquez, "Low-power impulse UWB architectures and circuits," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 332 –352, Feb. 2009.

[2] M. Stojcev, M. Kosanovic, and L. Golubovic, "Power management and energy harvesting techniques for wireless sensor nodes," in *9th International Conference on Telecommunication in Modern Satellite, Cable, and Broadcasting Services, TELSIKS '09.* , pp. 65 –72.

[3] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204 –1216, Jul. 2000.

[4] T. Ho, M. Medard, and R. Koetter, "An information-theoretic view of network management," in *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1295 – 1312, april 2005.

[5] C. Fragouli and A. Markopoulou, "A network coding approach to overlay network monitoring," in *Proceedings of International Conference Allerton*, 2005.

[6] R. Gowaikar, A. Dana, R. Palanki, B. Hassibi, and M. Effros, "On the capacity of wireless erasure networks," in *Proceedings of International Symposium on Information Theory, ISIT'04*, p. 401.

[7] R. Khalili and K. Salamatian, "On the capacity of erasure relay channel: multi-relay case," in *Information Theory Workshop, 2005 IEEE.*

[8] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding," *IEEE/ACM Trans. Netw.*, vol. 16, pp. 497–510, June 2008.

[9] D. S. Lun, M. Médard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks," *CoRR*, vol. abs/cs/0510070, 2005.

[10] C. Fragouli and E. Soljanin, "Network Coding Fundamentals," *Found. Trends Netw.*, vol. 2, pp. 1–133, January 2007.

[11] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413 –4430, Oct. 2006.

[12] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1973 – 1982, June 2005.

[13] C. Fragouli and E. Soljanin, "Decentralized Network Coding," in *IEEE Information Theory Workshop* , Oct. 2004, pp. 310 – 314.

[14] G. Angelopoulos, M. Médard, and A. P. Chandrakasan, "Energy-aware hardware implementation of Network Coding," in *Proc. of Network Coding Applications and Protocols Workshop, NC-Pro*, May 2011.

[15] S.-Y. Li, R. Yeung, and N. Cai, "Linear Network Coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371 –381, Feb. 2003.

[16] "Body Area Networking Standardization: Present and future directions," *2nd International ICST Conference on Body Area Networks*, May 2007.

[17] M. Wang and B. Li, "How practical is Network Coding?" in *14th IEEE International Workshop on Quality of Service,IWQoS*, June 2006, pp. 274 –278.

[18] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *Proceedings of the 2007 conference*

on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '07. New York, NY, USA: ACM, 2007, pp. 169–180.

[19] J. Heide, M. Pedersen, F. Fitzek, and T. Larsen, "Network coding for mobile devices - systematic binary random rateless codes," in *IEEE International Conference on Communications Workshops,* June 2009, pp. 1 –6.

[20] H. Li and Q. Huan-yan, "Parallelized Network Coding with SIMD instruction sets," in *International Symposium on Computer Science and Computational Technology, ISCSCT '08.* , vol. 1, pp. 364 –369.

[21] P. Vingelmann, P. Zanaty, F. Fitzek, and H. Charaf, "Implementation of random linear network coding on openGL-enabled graphics cards," in *European Wireless Conference*, May 2009, pp. 118 –123.

[22] H. Shojania, B. Li, and X. Wang, "Nuclei: GPU-accelerated many-core Network Coding," in *INFOCOM 2009*, April 2009, pp. 459 –467.

[23] P. Vingelmann and F. H. P. Fitzek, "Implementation of random linear Network Coding using NVidia CUDA toolkit," in *Networks for Grid Applications*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer Berlin Heidelberg, 2010, vol. 25, pp. 131–138.

[24] H. Shojania and B. Li, "Random Network Coding on the iPhone: fact or fiction?" in *Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video*, NOSSDAV '09. New York, NY, USA: ACM, pp. 37–42.

[25] A. Reyhani-Masoleh, "Efficient algorithms and architectures for field multiplication using gaussian normal bases," *IEEE Trans. Comput.*, vol. 55, pp. 34–47, January 2006.

[26] I. Hsu, T. Truong, L. Deutsch, and I. Reed, "A comparison of VLSI architecture of finite field multipliers using dual, normal, or standard bases," *IEEE Transactions on Computers*, vol. 37, no. 6, pp. 735 –739, Jun. 1988.

[27] E. Mastrovito, "VLSI designs for multiplication over finite fields GF(2^m)," in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, Lecture Notes in Computer Science, T. Mora, Ed. Springer Berlin / Heidelberg, 1989, vol. 357, pp. 297–309.

[28] A. Chandrakasan, N. Verma, and D. Daly, "Ultra-low-power electronics for biomedical applications," *Annu. Rev. Biomed Eng*, vol. 10, pp. 247–274, 2008.

[29] R. Koetter and M. Médard, "An algebraic approach to Network Coding," *IEEE/ACM Trans. Netw.*, vol. 11, pp. 782–795, October 2003.

[30] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless Network Coding," *SIGCOMM Comput. Commun. Rev.*, vol. 36, pp. 243–254, August 2006.

[31] S. Lin and J. Costello, "*Error Control Coding: Fundamentals and Applications*", 2nd ed. Prentice Hall: Englewood Cliffs, NJ, 2004.

[32] H. Dubois-Ferrière, D. Estrin, and M. Vetterli, "Packet combining in sensor networks," in *Proceedings of the 3rd International Conference on Embedded networked sensor systems*, ser. SenSys '05. New York, NY, USA: ACM, 2005, pp. 102–115.

[33] A. Miu, H. Balakrishnan, and C. E. Koksal, "Improving loss resilience with multi-radio diversity in wireless networks," in *Proceedings of the 11th annual International Conference on Mobile computing and networking*, ser. MobiCom '05. New York, NY, USA: ACM, 2005, pp. 16–30.

[34] "IEEE standard for local and metropolitan area networks part 16: Air interface for broadband wireless access systems," *IEEE Std 802.16-2009 (Revision of IEEE Std 802.16-2004)*, pp. C1 –2004, 29 2009.

[35] "3GPP ts36.300, Evolved universal terrestrial radio access (e-utra) and evolved universal terrestrial radio access network (e-utran): Overall description."

[36] J. Costello, D.J., J. Hagenauer, H. Imai, and S. Wicker, "Applications of error-control coding," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2531 –2560, Oct. 1998.

[37] D. Chase, "Code combining–a maximum-likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Transactions on Communications*, vol. 33, no. 5, pp. 385 – 393, May 1985.

[38] N. V. Emina Soljanin and P. Whiting, "Incremental redundancy hybrid ARQ with LDPC and Raptor codes," *IEEE TSransactions on Information Theory*, 2005.

[39] R. K. Ganti, P. Jayachandran, H. Luo, and T. F. Abdelzaher, "Datalink streaming in wireless sensor networks," in *Proceedings of the 4th International Conference on Embedded networked sensor systems*, SenSys '06. New York, NY, USA: ACM, 2006, pp. 209–222.

[40] K. Jamieson and H. Balakrishnan, "PPR: partial packet recovery for wireless networks," in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '07. New York, NY, USA: ACM, 2007, pp. 409–420.

[41] G. R. Woo, P. Kheradpour, D. Shen, and D. Katabi, "Beyond the bits: cooperative packet recovery using physical layer information," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, ser. MobiCom '07. New York, NY, USA: ACM, 2007, pp. 147–158.

[42] K. C.-J. Lin, N. Kushman, and D. Katabi, "ZipTx: Harnessing partial packets in 802.11 networks," in *Proceedings of the 14th ACM International Conference on Mobile computing and networking*, MobiCom '08. New York, NY, USA: ACM, 2008, pp. 351–362.

[43] Texas Instruments Products, "CC2500/CC2550DK Development Kit." Datasheet available at: http://focus.ti.com/docs/toolsw/folders/print/cc2500-cc2550dk.html