Energy Conservation and Security Enhancement in Wireless End-to-End Secure Connections

by

Kiarash Narimani

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2007

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Signature

Abstract

Wireless channels are vulnerable to interception. In some applications an end-to-end secure data transfer is required. However the use of cryptographic functions in communication over a wireless channel increases sensitivity to channel errors. As a result, the connection characteristics in terms of delay, throughput, and transmission energy worsen. Transmission energy is a key issue in some secure end-to-end wireless applications especially if they are running on mobile handheld devices with a limited source of energy such as batteries. That is why in most secure end-to-end wireless connections, the connection is dropped in poor channel conditions.

In this thesis, models are proposed by which the performance is improved and transmission energy can be lowered. A combination of a cross-layer controller, *K* Best Likelihood (*K*-BL) channel decoder, and a keyed error detection algorithm in the novel model supports the authorized receivers by a higher throughput, lower delay mean, and less transmission energy in a certain range of the Signal to Noise Ratio (SNR). This is done at the expense of additional computation at the receiving end. Ttradeoffs are examined and the simulation results of the new model are compared with those of conventional wireless communication systems.

Another model is devised to mitigate the energy consumption of the Turbo Code channel decoder. The overall decoding energy consumption for each packet can be lowered by reducing the average number of iterations in the Turbo Code channel decoder.

The proposed models achieve better energy consumption by reducing the number of iterations in a channel decoder that uses the Turbo decoder and by reducing the number of retransmissions in a trellis channel decoder. Furthermore, the security enhancement of the novel models is assessed in terms of the extent to which the enhancement is fully achieved.

Acknowledgements

I would like to thank Dr. Gordon Agnew, my supervisor, for his help and support through out my study.

Also, I would like to thank all the people who have directly or indirectly helped me in completing this thesis.

I am deeply indebted to Professor Hamidreza Jamali whose help, suggestions, and encouragement helped me in writing of this thesis.

I would like to thank all my friends (too many to mention) who made my studies at University of Waterloo a memorable experience.

The financial support of University of Waterloo is gratefully acknowledged.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

We are now in the age of digital information and processing. Information in the digital format is an asset not only to the business community, but also to other sectors of our society. The entire economy and the daily lives of people have become more and more dependent on data provided by the Internet, computers, and microprocessors.

The facilities and flexibilities that are provided by wireless networks have created considerable interest in wireless device usage. The wireless networks can typically provide easy access but are also characterized by slow transmission and lower accuracy. One does not need to be reminded of the importance of speed of transmission and accuracy of the transferred or processed information in wireless networks. Efficient methods and techniques are developed to manage, process, store, and transfer the digital information in wireless networks to achieve acceptable performance.

However, achieving such performance is not easy in wireless networks. The quality of a wireless connection changes frequently due to the variations in the surroundings. Routing through mobile nodes causes considerable variation in network topology, traffic loads, and channel conditions. As a result, it is difficult to support different applications. The link state metrics such as delay, bandwidth, jitter, packet loss rate, bit error rate, and throughput must be continuously monitored. Some of these metrics can be managed to achieve an optimal performance in the specific conditions.

However, getting and managing such link state information in a wireless network is not easy. In practice, there is a need for different levels of Quality of Service (QoS) in order to prioritize the parameters of an application. For instance, the vital parameter for Voice over IP (VoIP), real time image application and remote sensing [1]-[3] is delay; for telephony and video [4][5] is jitter, for banking and stock information [6][8] is security, for purchasing and file transferring is high accuracy, and for most networks of battery-powered devices such as sensor networks is energy consumption.

Some applications are very sensitive to delay. Almost all real-time applications are in this category, including video conferencing, real-time voice, real-time sensing (remote surgery), data base updating, and security cameras. Furthermore, applications that deal with financial information or health care data require a certain level of privacy. In these categories, some applications require authentication and encryption services, whereas others require only one of them [9]. Furthermore, handheld devices are typically powered by batteries, and energy conservation techniques need to be applied.

For wireless network applications, the combination of the characteristics of the environment, network, and application drives the performance. The environment in which mobile devices are randomly distributed, changes constantly, affecting the communication channel parameters. An example of the time variance effect on a channel is fading, caused by shadowing or multi-path. Also, the addition or loss of a device affects the interference level of communication channel. An application often requires shared limited resources such as energy or processing power, which, in turn, influences other applications. Therefore, the application's requirements can change the requirements of the connection.

Another example of the effects of an application is the increased sensitivity to error when a secure link is required on a wireless connection. Security functions such as confidentiality, authentication, and digital signatures increase the overhead of the channel and decrease the throughput. Moreover, the sensitivity of the connection to channel errors is increased; a single residual bit error in a received block of encrypted information can render the entire block unreadable [10]-[12]. This sensitivity is magnified within the constraints of a wireless channel.

Frequently, the network condition or the connection quality deteriorates so much that a viable performance is impossible. In most secure wireless connections, the connection is dropped in the harsh environments [13][14].

As wireless network usage expands, there is a need to support different applications on the same network. As a result, the network must change the sensitivity level of the parameters according to the application running at the time. It is possible that in one moment the link supports an application which is highly sensitive to jitter, but, at another moment the running application is more sensitive to accuracy. A network can be designed for a specific application in a certain environment, but it may not perform acceptably or maintain the quality for different environments.

There are different techniques to maintain the quality of communication when the channel condition varies. One way is to change the design so that the important parameters can be monitored and controlled. Another solution can be that a controller changes some parameters in the system to maintain the communication quality.

Almost all the protocols in wireless communication depend on feedback information to meet the reliability requirement of a connection. The information is processed at different levels and by different techniques.

Collision detection is a well-known technique that is largely employed in wireless protocols [15]. It is implemented in the physical layer of wireless communication systems. The recent transmitted packet is retransmitted on detection of a collision in the shared communication channel. The collision indicates that the shared communication channel is being used by another transmitter during the transmission of the packet.

The rate of collision is used to change the transmission power. It has been shown [16] that with this technique, the average transmission delay is lower than that of other techniques. Also, the average network power consumption can be reduced. More details about how to use the collision occurrence rate to lower power consumption can be found in [17]-[19].

At the receiver end, the Automatic Repeat reQuest (ARQ) techniques are implemented in the transport layer to ensure the quality of received information [20]. Such techniques use an error detection process to decide if the received packet needs to be resent. In a system with ARQ, the receiver returns one bit of information for each received packet to the sender to indicate the acceptance or rejection of the packet. For instance, Lu et al. [21] and Zorzi et al. [24] studied the throughput of different implementations of the Go-Back-N and Selective-Repeat ARQ.

Furthermore, more information about the channel state and the quality of the received packet can be returned to the sender to let it adaptively adjust the information format and transmission parameters. The Channel State Information (CSI) is sent back along with the packet acknowledgement to obtain a better performance in [25]-[28].

In many systems, a combination of error detection and correction techniques is used to handle the channel errors. This is called Hybrid ARQ (HARQ)[20]. To reduce the mean delay in mobile wireless communication, a truncated HARQ type II [25][26] has been studied in [27]. The idea of error control coding with a finite number of retransmissions has been introduced with more emphasis on the delay in [28]. In this technique the sender sends the error detection parity bits along with the original message. Upon the retransmission request, the sender transmits the error correction parity bits only once or twice. It is shown that it is possible to improve the delay characteristics of the connection [27].

M. Zorzi and R. Rao have described the use of an adaptive error control protocol with the Go-back-N ARQ technique in energy constrained systems [29]. They have studied the characteristics of the power provided by batteries, and analyzed the energy consumption of their adaptive error control technique, and compared it to the energy used in the original Go-back-N ARQ protocol.

The idea of reducing the transmission energy helps to extend the battery life of mobile battery-powered devices. One of the solutions to the channel fading problem is the Code Division Multiple Access (CDMA) systems. Their performance is directly affected by the level of noise in the channel. As a multiple access technique, most of the noise in the channel is generated by other transmitters. This explains why the signal power to the interference power ratio is so crucial in CDMA systems. Many studies indicate that lowering the power transmission in this technique reduces the level of the interference in the channel (for the other receivers). Zhuang and Kim et al. have explained that how CDMA networks achieve the lower level of the interference by reducing network's transmission energy, while the performance is maintained [30][31].

To reduce the transmission energy in CDMA systems by using Hybrid ARQ [31], the acknowledgement packets contain E_b/N_t (transmission energy per bit to interference noise density in working bandwidth ratio) information to control the sender's transmission power. As a result, the sender's energy consumption is reduced by not sending packets with more than the needed power. At the same time, the level of the interference noise in the working bandwidth (N_t) is dropped for the other receivers. The technique can result in less overall network energy consumption and a lower delay mean.

There has been much research on power management techniques in wireless connections. Each technique is developed to reduce the energy consumption and to extend the battery life. It is shown that how these techniques affect the other connection/network parameters such as interference [32][33], delay [34]-[36], reliability [37]-[40], and throughput [41][42].

A combined scheme for power and error control, recently proposed by N. Arulselvan and R. Berry [37], can reduce the energy consumption and the probability of error.

There are different approaches to reduce energy consumption and some require carefully compromising some parameters since they do not always lead to a desirable performance. The decision to make this compromise depends on the application and is performed by a controller. For example, M. N. Smadi and B. Szabados [43]-[45] have shown that the packet size can be adapted to the channel condition to achieve higher throughput and a lower delay. Also, an adaptive Forward Error Correction (FEC) technique has been studied [43][46]-[48] to increase the reliability of the channel during dynamic changes.

Recently, a cross-layer approach to optimize communication performance has attracted a great deal of attention. B. G. Lee et al. [49] have introduced an energy efficient and reliable method for a wireless sensor network.

In this thesis the idea of cross-layer control unit has been adopted to improve energy efficiency and performance. The controller optimizes the number of retransmission request by monitoring the information flow through different blocks and layers. A combination of a cross-layer controller, multi-output channel decoder, and a keyed error detection algorithm results in the authorized receivers realizing higher throughput, lower delay mean, and less energy consumption in a certain range of Signal to Noise Ratio (SNR).

A model is proposed to achieve a higher aggregate throughput over a certain range of residual channel error rates in secure end-to-end wireless connections. This is done at the expense of additional computational power at the receiving ends. These tradeoffs and the simulation results for the novel models are compared with those of conventional wireless communication systems. The newly devised model has lower retransmission rates which result in less energy consumption on transmission. The novel model also acquires a higher throughput and a less delay mean compared with those of the basic models in certain SNRs.

In addition, the efficiency of the proposed cross-layer controller and the advantage of the controller in terms of throughput and reliability in a range of SNR are explored. By using this cross-layer controller, a battery-powered device can still receive the information in the range of an acceptable delay mean and jitter. The cross-layer controller expands the acceptable performance of the model into lower SNR region. The model's improvement is achieved at the expense of additional computation at the receiving end.

The model is best used in the application were the cost of transmission energy is higher than the receiver's processing. In applications such as deep space communications, Wireless Sensor Networks (WSN), the proposed model can extend the battery life of the transmitter by reducing the energy consumption in transmission.

In another approach to reduce the energy consumption, the focus is on the extensive computing of the Turbo Code channel decoder. The overall decoding energy consumption for each packet can be lowered by reducing the average number of iterations in a Turbo code channel decoder. Battail et al.

[50] and Moher et al. [51] have demonstrated that the cross entropy is an important stop criterion in an iterative decoding algorithm. To reduce the number of decoding iterations in Turbo decoder, Hagenauer et al. [52] have introduced a cross entropy metric. Also, the similarity of the decoded block in consecutive iterations is used to detect the sufficiency of the decoding process [53]. In this thesis, a simple error detection algorithm, proposed by A. Banerjee et al. [54], is adopted to control the number of iterations in the Turbo decoder. The error detection algorithm is combined with security functions to enhance the privacy protection of the communication from a listening attack.

In applications such as personal wireless network or wireless sensor network where the wireless devices are supposed to work in a harsh environment such as, busy intersections, inside tornados, or crowded office spaces, the proposed model extends the battery life of the devices by using lower processing energy per bit.

The proposed models in the thesis convey an improved energy-consumption by reducing the number of iterations in a channel decoder using Turbo decoder and by reducing the number of retransmissions in a trellis channel decoder. Furthermore, the security enhancement of the proposed models and the point at which the enhancement is established is investigated.

The rest of the thesis is organized as follows: Chapter 2 is a review of the fundamental concepts of the techniques examined in the thesis in terms of achieving a better performance. In Chapter 3 the proposed models are introduced and details of the cross-layer control unit, controlling algorithm, flow of the information, and controlling signals are looked at. In addition, base models with conventional techniques and algorithms are described. The performances of the proposed models are compared to those of the base model later in Chapter 4. The simulation results in different channels are also discussed and the way the proposed models can achieve better energy consumption in range of Signal to Noise Ratio is demonstrated. Furthermore, the proposed models' security enhancement of the communication and protection against listening attacks are discussed. Chapter 5 provides the conclusion and suggestions for future works.

Chapter 2
Literature Review

Typically, wireless channels are subject to more problems than wired communication channels. Wireless channels are severely constrained by bandwidth limitations and bear rapid time-variant characteristics; mobile devices are usually constrained by extra limitations such as energy and processing capability caused by mobility, size, and weight limitations. The efficiency of the data transfer and the quality of the communication are significantly affected by these constraints. For example, the requirements for the hardware in terms of size, weight, and energy consumption on handheld battery-powered devices have severely restricts the employment of powerful processors, which in turn, limits the use of efficient algorithms due to their complexity.

Changes in the transmitter or receiver's location, speed, or direction; variations in the environment temperature; and the movements of nearby objects directly affect the characteristics of a wireless channel. This explains why the models that are viable for the wired channels are not viable for wireless channels. Due to the nature of wireless channels, the main characteristics of the channel vary rapidly over time, directly affecting the communication efficiency.

In this chapter the techniques and algorithms are reviewed of which are used in the proposed models. Also, the typical methods that are used to achieve a better performance in a communication over a wireless channel are explained.

2.1.1 Wireless Channel Modelling

There has been substantial research in the field of wireless communication channel modelling. The best models that fit a general wireless channel are fading models. In digital communications, the time required to transmit a symbol is called the *symbol time* [55]. If the symbol time is smaller than the average period of change, the characteristics of the channel are likely to be the same for several adjacent symbols and the channel is classified as slow fading. But if the rate of changes is close to the symbol time, the channel is classified as fast fading [56].

Figure 2.1 shows some of the wireless channels' problems that can be modelled by fading characteristics.

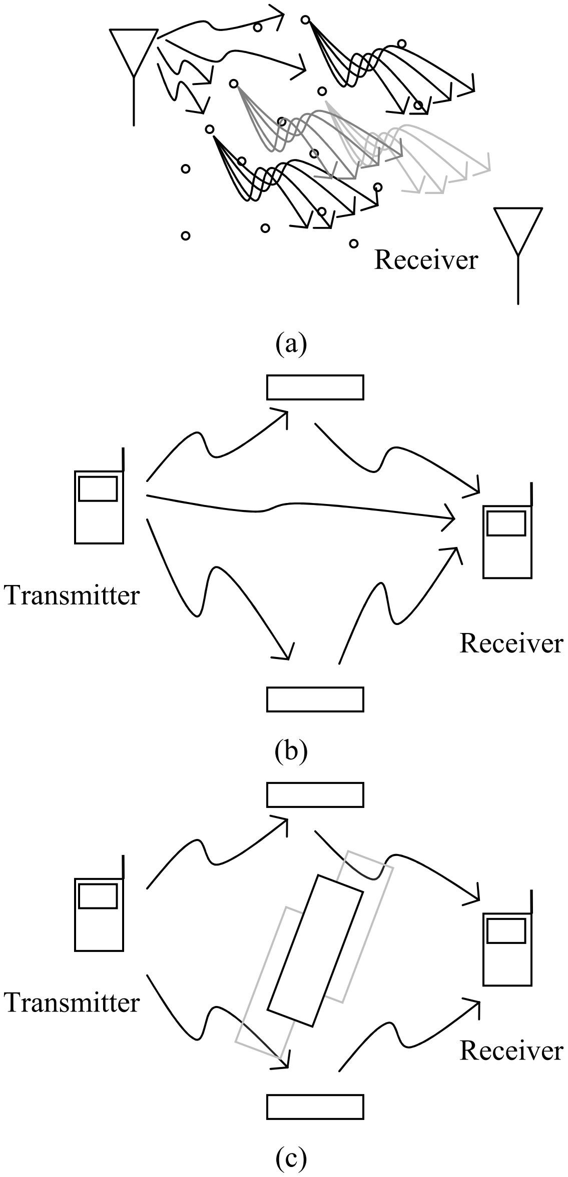


Figure 2.1 Examples of wireless channels' challenges: (a) scattering, (b) multi-path, and (c) shadowing

Time-variant characteristics of the channel, along with the resource constraints, reduce the reliability of transmissions over wireless channels. The reliability of a communication system is measured by the average number of errors in the receiver.

There are several error control techniques to overcome wireless channels' problems [57]. One technique that is used to increase the reliability of the channels in digital communication is the error handling technique [58]. There are several methods to implement error handling techniques, depending on the channel being simplex (one-way) or duplex (two-way) [56].

2.2 Error Handling Techniques

Error Control Coding (ECC) depends on controlled redundancy to detect or correct errors in the receiver. The requirements of the system (e.g., throughput, error rate, delay, and jitter) and the nature of the channel (e.g., bandwidth, fading, mobility, level of noise, and interference) play a major role in the selection of the error handling technique.

The block diagram in Figure 2.2 illustrates a simplex communication channel. The flow of data is strictly in one direction, from transmitter to receiver, or from the recording unit to the replaying device. Error control for a one-way system is accomplished by using the *Forward Error Correction* (FEC). This means that the receiver uses the arithmetic or algebraic structure of the code to determine what the original message is, given the (erroneous) received word.

FEC improves the error correction capacity of a system by adding some carefully designed redundant information to the data sent through the channel. The process of adding information is known as channel coding [20] and is categorized in two main groups: convolutional coding and block coding. Convolutional codes manipulate serial data, one or a few bits at a time. Block codes operate on relatively large blocks of data (typically, up to a couple of hundred bytes). There are a variety of useful convolutional and block codes and different coding and decoding algorithms to efficiently recover the original message from the received information [57]-[59].

Examples of FEC systems include magnetic tape storage systems in which the information recorded on tape can be replayed weeks or even months after it is recorded. Another example is deep space communications, where the *round-trip delay* (the time interval in which a message that has been sent from the transmitter, comes back again from the receiver) is too long and retransmission is not possible. Many systems in use today employ some form of FEC, even if the channel is not strictly one-way, for example, satellite communication, video conferencing [20][57][123].

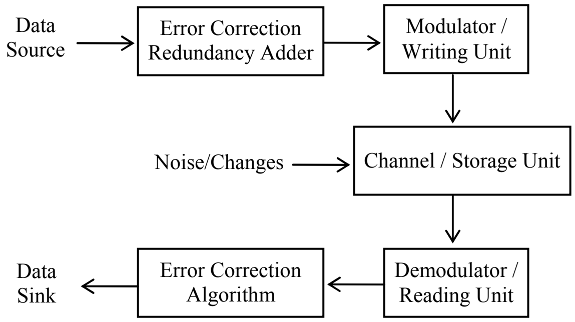


Figure 2.2 Block diagram of a typical one-way data transmission or storage system one-way channel, and the usage of FEC

Error control for a two-way system is accomplished by using error detection and retransmission, called the *Automatic Repeat reQuest* (ARQ). In an ARQ system, the receiver uses an error detection technique to detect errors in the receiving messages. The required information to distinguish a correct message from an incorrect one is added to the message before transmission. When the error(s) are detected at the receiver, a request is sent through the return channel to the sender. In order to ask for a specific message to be retransmitted in some ARQ systems, the messages carry an identifier. The error avoidance unit in the sender then transmits the requested message again by using the identifier. This can continue until the message is received correctly by the receiver [60].

Figure 2.3 shows the block diagram of a system using ARQ in a duplex channel.

In many systems, a combination of error detection and correction techniques is adopted to handle the channel errors, as shown in Figure 2.4, and is referred as *Hybrid ARQ* (HARQ). The structure of the HARQ system is similar to that of the ARQ system, but for the receiver. After the message is received, an error correction algorithm is employed to recover all the errors. Then the error detection algorithm checks if there are any more errors in the message. If so, a request for retransmission is sent to the sender; otherwise, the message is accepted [59].

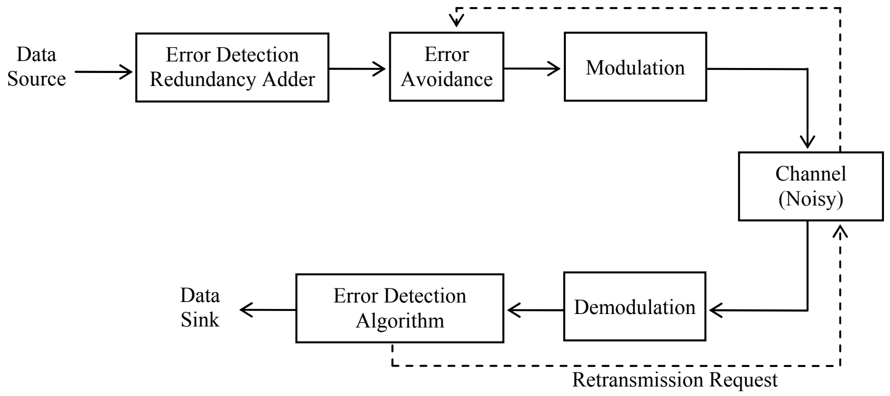


Figure 2.3 Two-way transmission system using ARQ

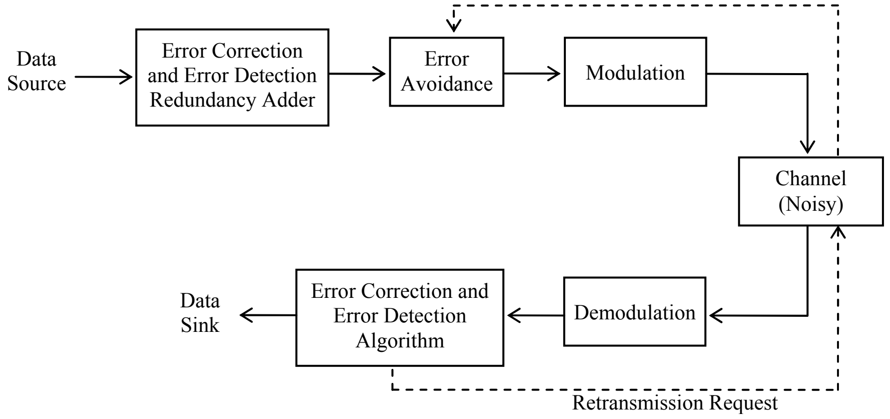

Data Source
Error Correction and Error Detection Redundancy Adder
Error Avoidance
Modulation
Channel (Noisy)
Demodulation
Error Correction and Error Detection Algorithm
Data Sink
Retransmission Request

Figure 2.4 Two-way transmission system using HARQ

In the next few sections, the concept of error correction and error detection techniques are discussed, and the structure of the implemented code are described.

2.2.1 Forward Error Correction (FEC)

Error correction is based on the Hamming distance of code words. Figure 2.5 is an example of an error correction technique for code words with 21 bits of *free distance* (d_{free} = 21 bits). If the transmitter sends one of these two code words, the receiver can perform error correction on the received words. The correction process corrects the word to the code word with the maximum probability of being transmitted, given the received word, which corresponds to the code word with the smallest Hamming distance to the received word [57][60]. In Figure 2.5, received word 1 and

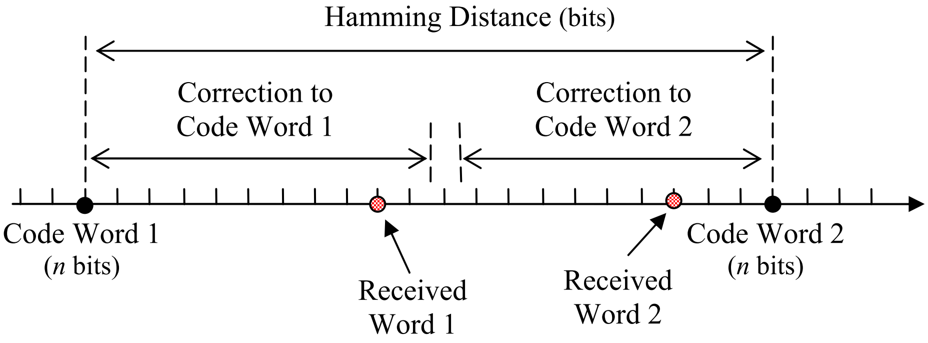


Figure 2.5 Correction of received code using ECC

received word 2 are corrected to code word 1 and code word 2, respectively. Of course, if the received word contains more error bits than the correction ability of the code (here 10 bits), the receiver is not capable of correctly recovering the message.

2.2.2 Automatic Repeat Request (ARQ) Technique

The principle of the ARQ technique is based on detecting the error(s) in the received block by adding extra bits, called parity bits, to the transmitted block. They are chosen in a way in which their values are highly correlated with the contents of the block. The error in the received block is detected by regenerating the parity bits in the receiver, and comparing them with the received parity bits. If any of the bits are decoded incorrectly, it is very likely to be detected [60]. The implementation of error detection is represented in Figure 2.6.

An example of error detection is shown in Figure 2.7. If the received word is not a valid code word, the receiver can conclude that it contains some error bits and requests for retransmission. Any error which does not change a valid code word to another can be detected by this technique. Then again, if the noise of the channel changes the transmitted code word to another valid code word, the error cannot be detected by this technique.

There are two types of ARQ systems: stop-and-wait ARQ and continuous ARQ [59]. With stop-and-wait ARQ, the transmitter sends a code word to the receiver and waits for a positive (ACK) or negative (NAK) acknowledgement from the receiver (Figure 2.8). If an ACK is received (no error detected), the transmitter sends the next code word. If an NAK is received (errors detected), the

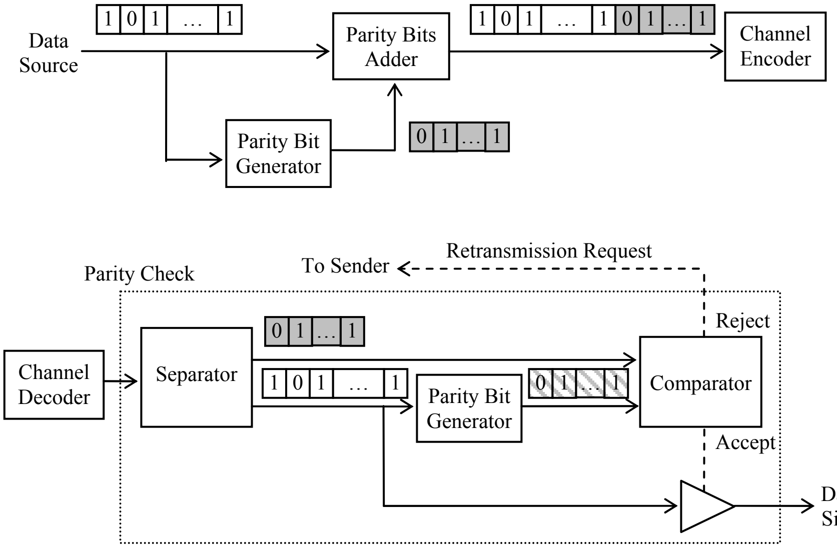


Figure 2.6 Error detection using parity bits implementation in sender and receiver

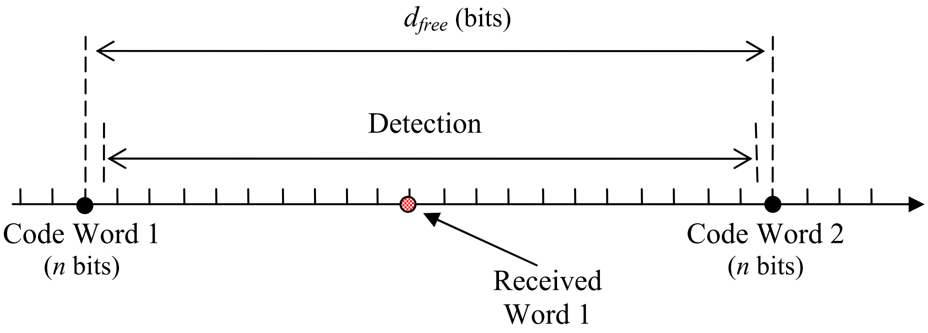


Figure 2.7 Error detection in a block of message using parity bits

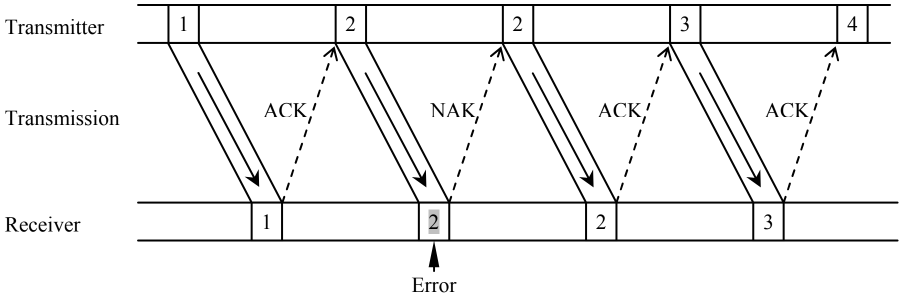


Figure 2.8 Stop-and-wait ARQ

transmitter resends the preceding code word. When the noise is persistent, the same code word might be transmitted several times before it is correctly received and acknowledged.

With the continuous ARQ, the transmitter sends code words to the receiver continuously and receives acknowledgements continuously. When NAK is received, the transmitter begins a retransmission. It can back up to the code word in error and resend that word plus the words that follow it. This is called go-back-N ARQ (Figure 2.9). In such a system, the acknowledgement for a code arrives after a round-trip delay, defined as the time interval between the transmission of a code and the receipt of an acknowledgment for that code. During this interval, N-1 other codes are also transmitted. When a NAK is received, the transmitter backs up to the code that is negatively acknowledged and resends it and N-1 succeeding code words that are transmitted during the round-trip delay.

Alternatively, the transmitter might simply resend only those code words that are negatively acknowledged. This is known as selective-repeat ARQ, shown in Figure 2.10, and is more efficient than go-back-N ARQ, but requires more logic and buffering; that is, as the receiver must be able to reassemble the memory blocks in order.

ARQ schemes are quite effective for throughput enhancement in time-varying mobile channel environments. The continuous ARQ is more efficient than stop-and-wait ARQ, but it is also more complex. In a satellite communication system where the transmission rate is high and the round-trip delay is long, continuous ARQ is often used. Stop-and-wait ARQ is used in systems where the time taken to transmit a code word is long compared to the time taken to receive an acknowledgment [59].

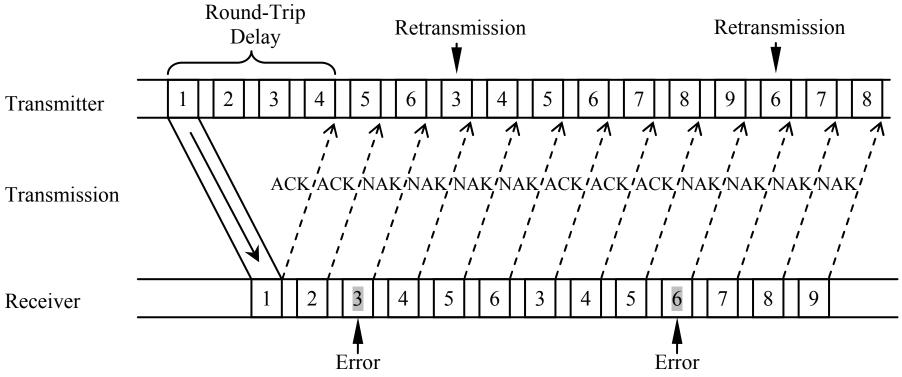


Figure 2.9 Go-back-N ARQ with N=4

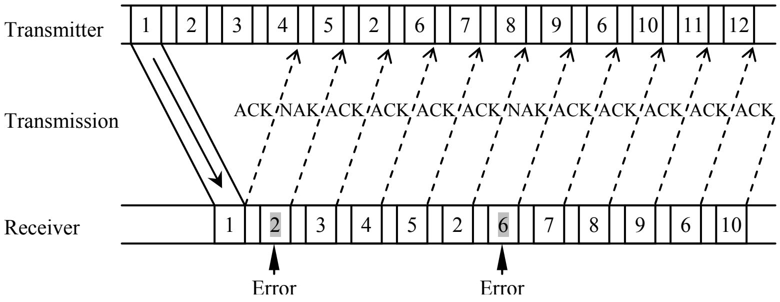


Figure 2.10 Selective-repeat ARQ

2.2.3 Hybrid ARQ

In many systems, including mobile phone networks [61], detection and correction techniques are combined to handle the channel errors. This is called Hybrid ARQ (HARQ) and is composed of three types of HARQ.

For type I HARQ, the error correction codes, and error detection codes are implemented in the same code word. In these systems, after the receiver decodes the error-correction code, it checks if the block still has an error using the error-detection decoder. Figure 2.11 depicts type I Hybrid ARQ systems work. Depending on the region that the received word is in; it is either corrected to a code word or detected as an erroneous word. Although this scheme is much less complex, it is an inefficient method of implementing ARQ. In fact, the main disadvantage of type I HARQ is that negative acknowledged packets are retransmitted entirely.

In type II and type III HARQ, after receiving a negative acknowledgement, the sender sends different bits, instead of simply repeating the same block. The information in the retransmitted block can be used to recover more errors in the original block. Therefore, the information from the first block is used, along with the new parity bits, to recover the errors. A simple way to implement the type II HARQ is to send the original message with error-detection on the first transmission and send the error-correction parity bits after the negative acknowledgement are received (Figure 2.12). If the receiver detects an error in the first block, the receiver uses the additional information of the second transmission to recover the original message. The difference between type II HARQ and type III HARQ is that the information in each retransmission block is different in type III HARQ. These schemes require additional complexity at the transmitter than the Type I HARQ, because packets must be reconfigured with the retransmissions.

Figure 2.12 illustrates the different types of HARQ schemes.

Since HARQ schemes benefit from both FEC and ARQ techniques, the HARQ technique generally improves the performance of the system in terms of the accuracy and delay. More details on HARQ can be found in the literature [59][60].

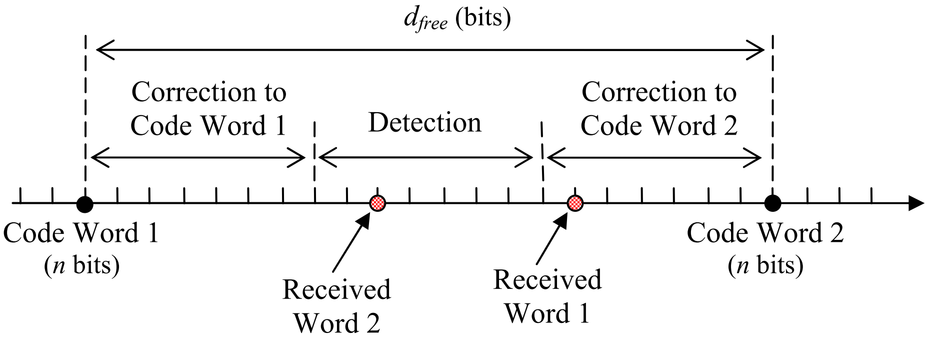


Figure 2.11 Detection and correction of received codes using ECC

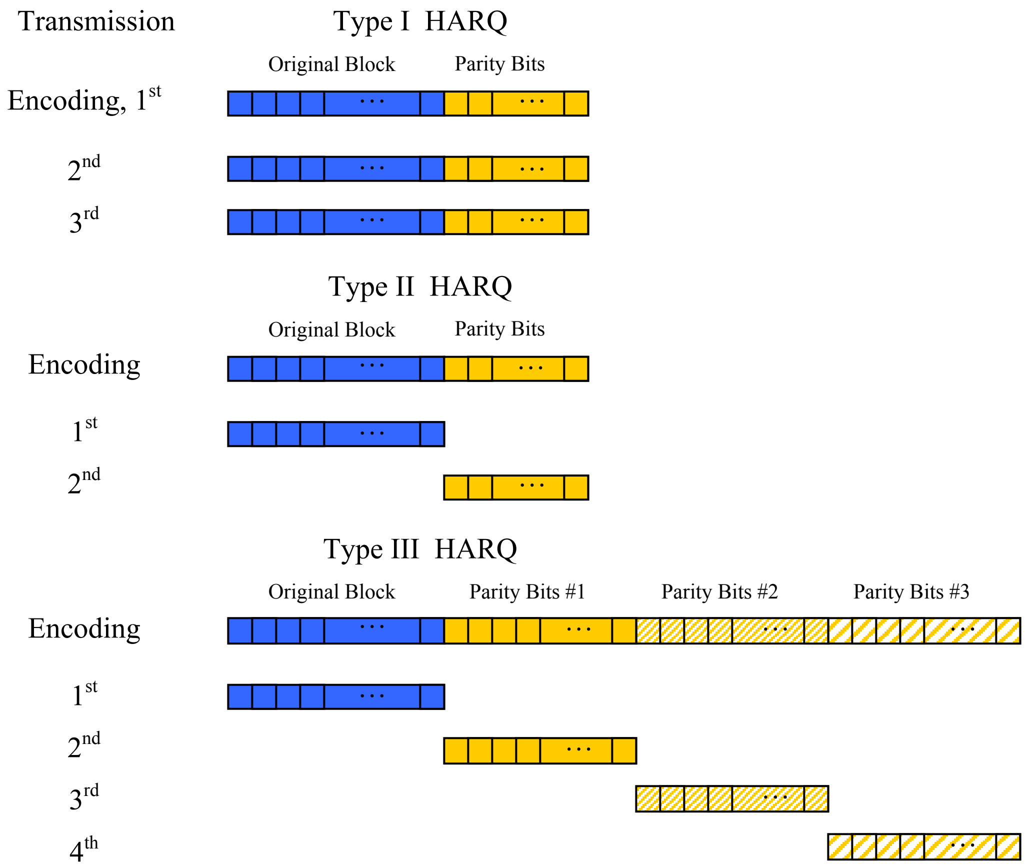


Figure 2.12 Different types of Hybrid ARQ schemes

In next section, some code structures that are used for detection and correction in FEC and ARQ systems are described.

2.2.4 Code Structure

The structure of the codes that are used in ECC systems is exemplified. For simplicity, it is assumed that the source of information are blocks of k-bit binary messages. Consequently, there are $M=2^k$ possible blocks, which are called block messages or blocks of information. A block code consists of a set of fixed-length vectors, called code words whose length n is the number of code elements in the vector. Since the elements of a code word are selected from the binary alphabet, there are 2^n possible words. From these, $M=2^k$ words are selected ($k<n$) to form code words. A block of k information bits is mapped into a code word of length n such that the resulting block code is a (n, k) code, and the

ratio R_c=*k/n* is defined to be the *code rate*. Thus, each ECC system selects a subset of $M=2^k$ code words of *n*-bit length to transmit a *k*-bit message block [59][60].

Besides the code rate parameter, R_c, an important parameter of a code word is its *Hamming* weight, which is simply the number of nonzero elements that it contains. A measure of the differences in the code words is the number of corresponding elements or positions in which they differ. This measure is the Hamming distance between the two code words [59][60].

If the original message block appears in code word *(n, k)*, the code is *systematic* [59][60]. In systematic codes, *n–k* redundant bits are added to each message to form a code word. The redundant bits provide the code with the capability of detecting and/or correcting some channel errors. The redundant bits, *parity bits*, are best chosen if the code words have the maximum Hamming distance from each other. The minimum of the Hamming distance between any two code words is called *free distance* and is denoted by d_{free}. A code with a free distance of d_{free} is capable of correcting up to $t=(d_{free}-1)/2$ random bit errors in each code word. Also, the code can be used to detect up to $d_{free}-1$ random bit errors in each erroneous code word.

To show all the important parameters, binary block codes with *k* bits message length, *n* bits code words length, and free distance of d_{free} are denoted by (n, k, d_{free}).

There has been much work done on finding block codes with good properties. Some examples are given in Table 2.1 [59][62][63].

The codes in Table 2.1 are called *block codes*, because a fixed size *block* of *k*-bit data is examined to compute the required error control bits. There are applications, such as audio and video with a continuous data stream, which the block codes are not suitable unless the data is chopped in *k*-bit blocks. In these cases, *convolutional codes* are well suited for error control purposes [64].

Typically, convolutional codes are described by two parameters: the code rate and the constraint length. The code rate, R_c=*k/n*, is expressed as a ratio of the number of bits in the convolutional encoder (*k*) to the number of channel symbols output by the convolutional encoder (*n*) in a given encoder cycle. The other parameter is *m* which indicates how many cycles an input bit is retained and used for encoding, after it first appears at the input to the convolutional encoder. The *m* parameter can be thought of as the memory length of the encoder.

Figure 2.13 illustrates a convolutional encoder that is implemented by a two-bit memory, where the relations between the source input and the encoder outputs sequence are exhibited.

Table 2.1 Some binary block codes and their properties implemented on GF (2^m)

Name of the code	***n* (bits)**	**k (bits)**	**d_{free} (bits)**
Original Hamming	2^m-1	$n-m$	3
Extended Hamming	2^m	$n-m$	4
BCH*	2^m-1	$n-mt \leq$	$2t+1 \leq$
RS**	$m(2^m-1)$	$n-2mt$	$m(2t+1)$

All codes are defined in the extended binary field GF(2^m)

* BCH : Bose, Chaudhuri and Hocquenghem codes; *t* is the maximum bits that are correctable

** RS : Reed-Solomon codes; *t* is the maximum bits that are correctable

The redundancy, added by the parity bits, is used to detect the errors in the code words (error detection) and/or recover the original information (error correction) by different methods.

In communications, parity checking refers to the use of parity bits to check that the data is transmitted accurately. On the receiver end, the device checks each block to ensure that it has matching parity bits. If there is a mismatch, the receiver knows there is an error during transmission.

The sender and receiver must both agree regarding parity checking and the set of code words used. If the two ends are not configured with the same code word set, communication is impossible.

There are many other more sophisticated protocols for ensuring transmission accuracy such as MNP [65] and CCITT V.42 [66]. Parity checking is used not only in communications but also to test memory storage devices. Many PCs, for example, perform a parity check on the memory every time a byte of data is read.

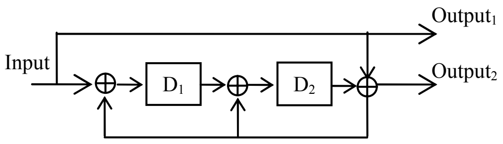


Figure 2.13 Convolutional encodes with R_c $=1/2$ ($k=1$, $n=2$) and $m=2$

2.3 Trellis Coded Modulation (TCM)

The innovative aspect of Trellis Coded Modulation (TCM) is that convolutional encoding and modulation [67] are viewed as one operation. Therefore, the received signal is processed by combining the demodulation and decoding in a single step, instead of being first demodulated and then decoded. Consequently, the parameter governing the performance of the transmission system over the channel is not the free Hamming distance of the convolutional code words but is the free Euclidean distance [64] between the transmitted signal sequences. The key to this integrated modulation and coding approach is to form an effective method for mapping the coded bits into signal points such that the minimum Euclidean distance is maximized. Such a method has been developed by Ungerboeck (1982), based on the principle of mapping by set partitioning [68].

Typically, the TCM encoder consists of a convolutional encoder cascaded to a memory-less mapper. The signal transmitted at each discrete time depends not only on the source symbol at the same time instant, but also on a finite number of the previous source symbols (Figure 2.13).

A convenient way of describing a set of signal sequences is by a trellis diagram [64]. The distance properties of a TCM scheme can now be studied in the same way as those of convolutional codes. The memory state and outputs in Figure 2.13 reflects the function of the input and current memory state of the encoder by using the trellis diagram in Figure 2.14.

For binary convolutional codes, TCM schemes have an underlying trellis structure that can be decoded by the Viterbi decoding algorithm as described in the next section.

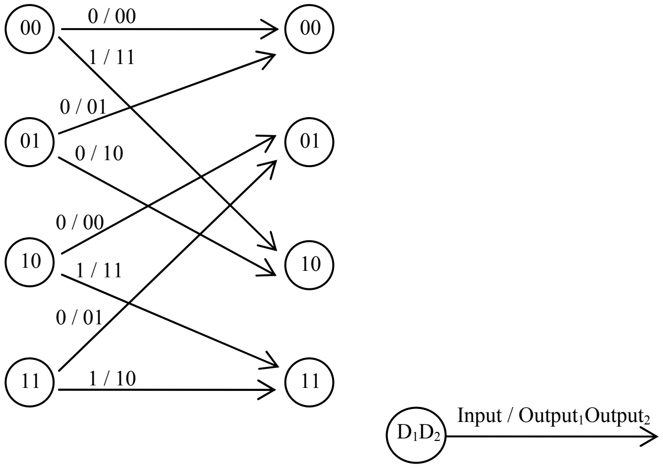


Figure 2.14 Trellis diagram of the convolutional encoder in Figure 2.13

2.3.1 Decoding TCM

Convolutional encoding with Viterbi decoding is an FEC technique, and is employed in the TCM decoder. Due to the one-to-one correspondence between the signal sequences and paths traversing the trellis, Maximum-Likelihood (ML) decoding is adopted to search for the trellis path with the minimum Euclidean distance to the received signal sequence. The optimum decoding is the search for the most likely path through the trellis tree once the received sequence is observed at the channel output.

Because of the noise, the chosen path might not coincide with the correct path. That is, the path traced by the sequence of source symbols, but occasionally diverges from it and remerges at a later time (Figure 2.15). The diverging and remerging time depends on the noise/fading level of the channel at the time of the symbol transmission.

Considering the convolutional coder in Figure 2.13, If the sequence *10010* is sent to the receiver, it is encoded to *11 00 10 11 01* and transmitted. Figure 2.15 illustrates the correct path in the ML receiver. If the noise in the channel changes the transmitted sequence and the ML receiver sees *11 11 10 10 01* at the output of the channel, it decodes the sequence as *11110* which minimizes the Euclidian distance to the received sequence.

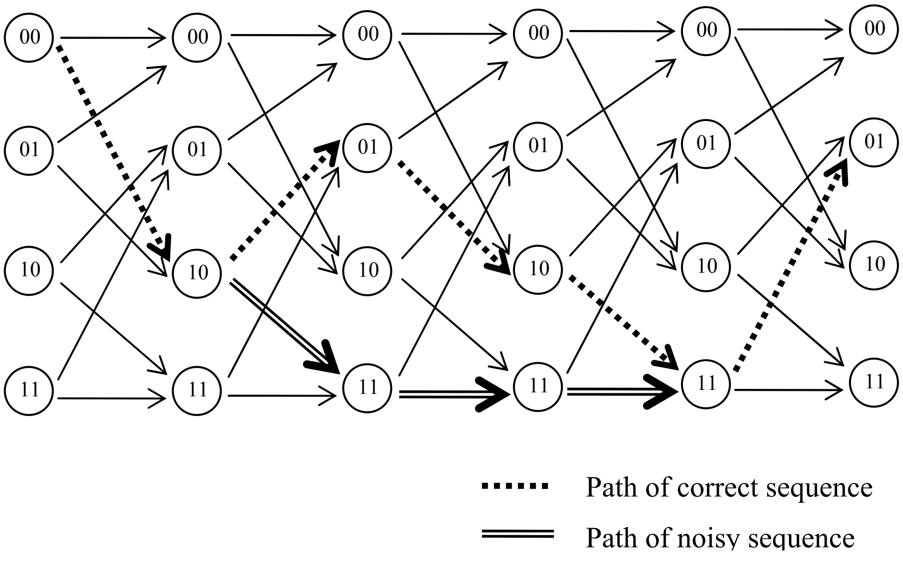


Figure 2.15 Viterbi decoder diagram

Viterbi decoding was introduced by Andrew J. Viterbi [69]. Since then, many researchers have extended his work by finding good convolutional codes, exploring the performance limits of the technique, and varying the decoder design parameters to optimize the implementation of the technique in hardware and software [70].

To measure the improvement of the system by using channel codes, the *coding gain* is considered. The difference in the SNR to achieve a certain bit error probability between a system with the channel coder and the same system without the channel coder is coding gain [59]. TCM allows the system to achieve the Bit Error Rate with a lower transmission power. Therefore the power saving for the TCM (in decibels) is expressed by its coding gain.

However, there is a cost; the increased bandwidth of the transmitted signal and, of course, higher receiver complexity [67]. Thus, coded modulation provides an effective method for trading off the bandwidth and implementation complexity for the transmission power. This situation is especially important for digital communication systems that are designed to operate in the energy limited regions. TCM finds applications on bandwidth limited channels such as voice-band telephone, terrestrial microwave, satellite and mobile channels [71].

2.4 Turbo Codes

In 1993, Turbo Codes [72] were first introduced and have provided significant improvements in the coding gain over the other channel codes. Turbo codes use two Recursive Convolutional (RC) encoders [73], separated by an interleaver, for encoding and multiple iterations of the Maximum A-posterior Probability (MAP) algorithm, as seen in Figure 2.16 [55].

Turbo Codes significantly outperform conventional codes, and rely on interleavers to reduce the effect of burst errors. This is particularly important when communication is conducted over a fading channel where burst errors happen more frequently.

On the receiver end, the distinctive characteristic of the Turbo Code is the use of iterative decoding, in which the results of one MAP decoder are passed to another MAP decoder for the next decoding iteration, as depicted in Figure 2.16. The iteration process is useful in sharing the information from one decoding to another. The first decoder does not have the information from the second decoder output in the first iteration. Then, the output of the second decoder provides feedback into the input

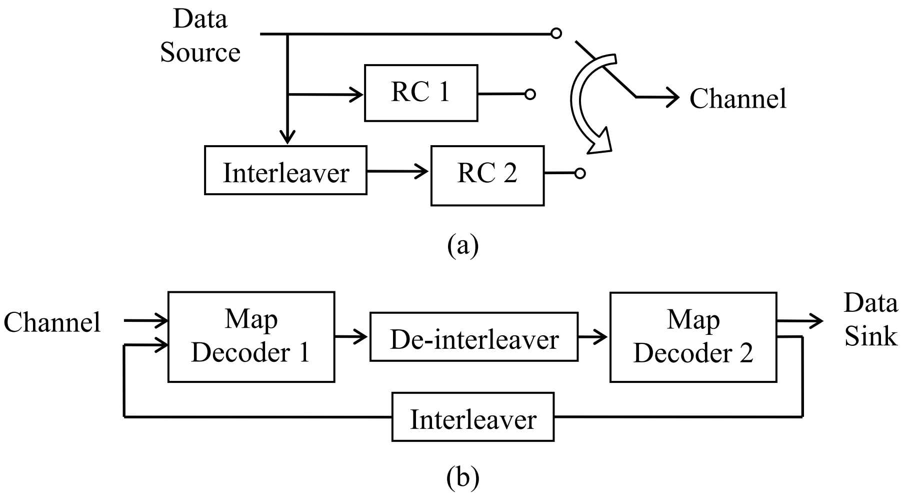


Figure 2.16 Turbo codes; (a) a general encoder and (b) a general decoder

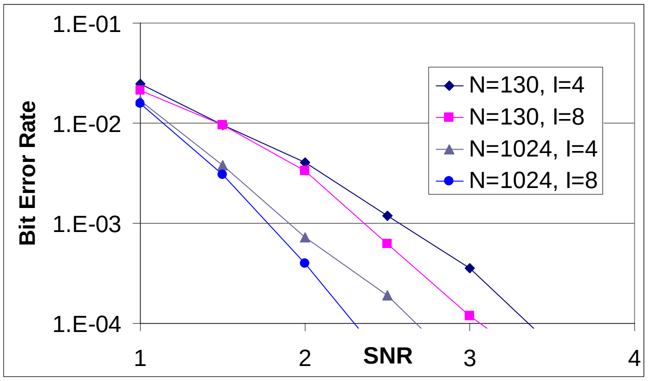


Figure 2.17 Bit Error Rate vs. SNR for two block sizes *N*=130, and 1024 bits and the number of iterations of *I*=4, and 8

of the first decoder. Thus, the first decoder has more information in the second iteration, and the decoding performance should improve. Intuitively, with more iterations the error probability decreases in Turbo Code decoder.

Figure 2.17 offers a comparison of the Bit Error Rate performance of a Turbo Code for various block sizes and decoding iterations. Here, *I* is the number of decoding iterations, and *N* is the block size assuming the channel is subject to Additive White Gaussian Noise channel (AWGN).

In Figure 2.17, it is evident that performance improves as the length of the blocks (*N*) and the number of decoding iterations (*I*) increase. This occurs since more information is shared between the decoders, and they have more information about the input, resulting in more accurate decisions. The impact of this extra information, though, decreases as the number of iterations increases, since the decoders have already exchanged whatever information is useful to the decoding process [74]. This implies that, beyond a certain point, more iterations do not improve the error performance and only lengthens the decoding time. It is also noteworthy that the decoding time increases as the number of decoding iterations increase.

To further reduce the energy consumption, researchers focus on the complex computing processes of the Turbo Code channel decoder. The overall decoding energy consumption for each packet can be lowered by reducing the average number of iterations in a Turbo code channel decoder. The methods which are used to reduce the number of iterations in Turbo Code channel decoder are now introduced.

2.4.1 Iteration Reduction Techniques

Battail et al. [50] and Moher et al. [51] have reported that the cross entropy is an important criterion in the iterative decoding algorithm. Hagenauer et al. [52] have introduced a cross entropy metric to reduce the number of decoding iterations in the Turbo decoder. They have used the log likelihood ratio, and the extrinsic value of each bit at each iteration to approximate the cross entropy of the whole block. If the cross entropy of the probability distribution between the two decoders at each iteration drops below a certain threshold, the decoder concludes that running more iterations does not recover more accurate bits.

To stop the iteration in the Turbo Code channel decoder, J. Hamorsky and U. Wachsmann [53] have used different criteria. In their approach, the results of the decoders are transformed to bits by hard decisions at the end of each iteration. The block of data which consists of these bits is compared to the last iteration's block. The authors suggest the decoder can stop the decoding process if the block values do not change in $i_{thershold}$ consecutive iterations. The value of $i_{thershold}$ can be adaptively changed according to channel condition to increase the efficiency of the technique, as signified in Figure 2.18.

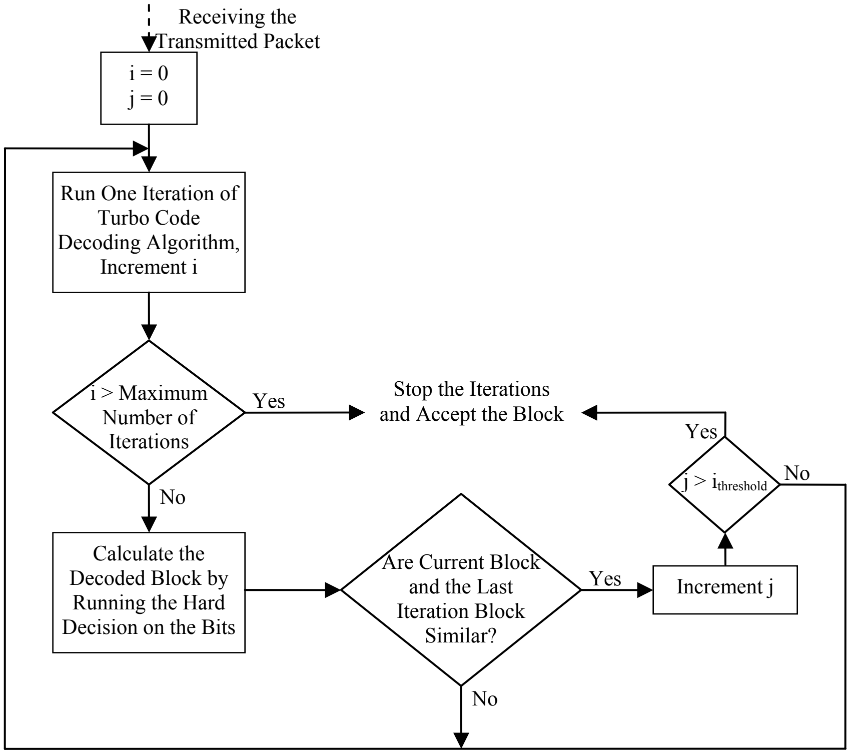

Receiving the Transmitted Packet
i = 0
j = 0
Run One Iteration of Turbo Code Decoding Algorithm, Increment i
i > Maximum Number of Iterations
Yes
No
Stop the Iterations and Accept the Block
Yes
j > $i_{threshold}$
No
Calculate the Decoded Block by Running the Hard Decision on the Bits
Are Current Block and the Last Iteration Block Similar?
Yes
No
Increment j

Figure 2.18 Iteration stop criteria using the similarity of the decoded block in $i_{thershold}$ consecutive iterations [53]

A. Banerjee et al. [54] have proposed a simple error detection algorithm, similar to CRC check, to control the number of iterations in the Turbo decoder. The iteration stops when the error detection algorithm finds no error in the decoded block. In Chapter 3, the error detection algorithm is combined with the security functions to enhance the privacy protection of the communication from a listening attack.

The interleaver is one of the essential components in the Turbo code design, and is also used to permute the data bits in the transmitted block so that the channel noise spreads over all the data blocks. The error correction technique use in the Turbo Code can recover data more effectively, when the noise effect randomly moved in the code.

The next section is devoted to the interleaver and how it is adopted to recover data in the blocks with burst errors.

2.5 Interleaver

Most of the well-known codes, which have been devised for increasing reliability in the transmission of information, are effective when the errors caused by the channel are statistically independent. However, there are channels that exhibit bursty error characteristics [55]. One example is the class of wireless channels characterized by multi-path and fading. Signal fading, due to time-variant multi-path propagation, often causes the signal to fall below the noise level, thus resulting in a large number of consecutive errors. This is the case for the fading channels which are examples of memory-channels. It is assumed that a slow fading channel exhibits fades with durations that exceed the time required to transmit several channel symbols. Thus, a fade that affects one channel symbol is very likely to affect a number of adjacent channel symbols. As a consequence, slow fading transmission errors often occur in bursts. A second example is the class of magnetic recording channels (tape or disk) in which the defects in the recording media result in clusters of errors. Usually, they are not corrected by codes that are optimally designed for statistically independent random errors.

Most block codes and convolutional codes are efficient at random-error correcting. A random error-correcting code corrects up to *t* symbols per code word, regardless of their placement in the code word. An error burst consists of several symbol errors within a small number of received code words. The error-correction capacity, required to correct these bursts, is large and is included in blocks that are not affected by the burst. However, this is inefficient in terms of overhead.

Therefore, a sophisticated error control scheme must be capable of correcting both random and burst errors. Many communication systems contain ARQs along with FECs. Such a Hybrid ARQ is a potential solution, but requires a feedback channel. The disadvantage of introducing a variable information rate to the system is the fact that the retransmission rate is dependent on channel errors and characteristics which are random time variables. However, for some applications such as a compact-disc player, digital radio, or television broadcasting, where ARQ is not applicable, alternative techniques to ARQ are needed to handle burst errors that do not require a feedback channel. Much effort has gone into the development of codes or coding techniques that can specifically deal with channel burst errors. Probably one of the best known burst error correcting codes is the subclass of cyclic codes, called Fire codes, named after P. Fire [75], who discovered them. Another class of cyclic codes for burst error correction has been subsequently discovered by Burton [76].

In a one-way channel, the combination of the FEC technique with interleaving, as depicted in

Figure 2.19, is a solution for handling burst errors [57]. It should be noted that interleaving does not change the constant throughput of a FEC system, but does produce an additional delay.

An interleaver reorders the channel-encoded symbols, before transmitting them over the channel. This operation is neutralized or reversed by the deinterleaver at the receiver before the channel decoding. If the parameters of the interleaver are chosen properly with respect to the channel characteristics, the reordering results in a spread of burst errors over a long time interval in the channel-encoded symbol sequence. Such errors then appear as random single errors to the error correction block as signified in Figure 2.20. However, the resulting additional transmission delay for such a system can become quite long, if a large interleaver is used.

A burst of errors of length *b* is defined as a sequence of *b*-bit errors, the first and last of which are *1*s. The burst error correction capability of a code is defined as one less than the length of the shortest uncorrectable burst. It is relatively easy to show that a systematic (*n, k*) block code, which has *n-k* parity check bits, can correct bursts of length $b < [(n\text{-}k)/2]$.

An effective method for dealing with burst error channels is to interleave the coded data in such a way that the bursty channel is transformed into a channel with independent errors. Thus, a code

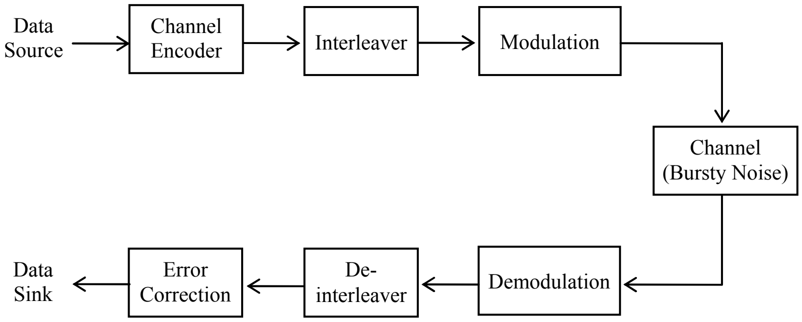


Figure 2.19 Channel coding with interleaving

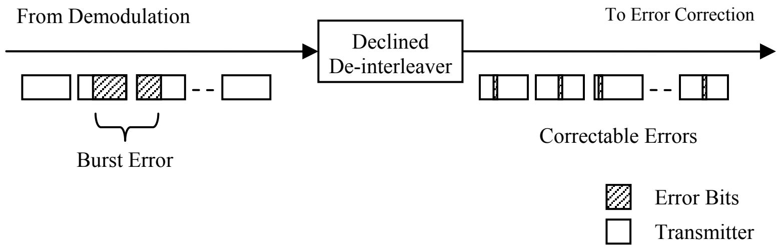


Figure 2.20 Breaking the burst error to correctable size by interleaving

that is designed for independent channel errors (short burst) can be used.

Suppose that *L* code words are interleaved. If the original code corrects any single burst of length *t* or less, the interleaved code corrects any single bursts of length *Lt* or less. This is due to the fact that if a burst error occurs, it is distributed over *L* code words equally, and appears as independent random bit errors. So, if each code word with a burst error of length *t* is capable of recovering data, the interleaver and the decoder can recover data in a block of code words when single burst error of length *Lt* occurs [57].

A block diagram of a simple interleaving process is portrayed in Figure 2.21. In this example, the input blocks are written horizontally in memory cells, while the output blocks are read out vertically (arrows) or vice versa.

Another type of block interleaver is semi-random interleavers. The input bits are read out in a

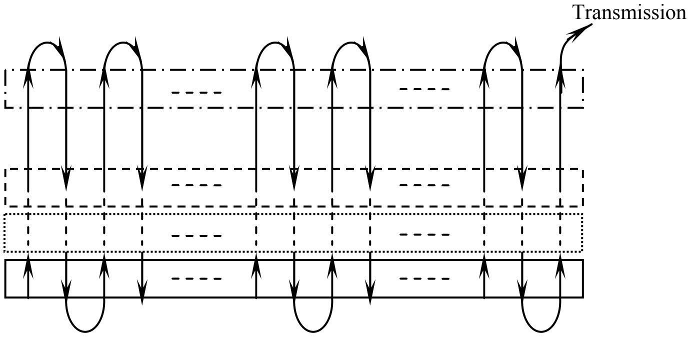


Figure 2.21 Example of interleaving process

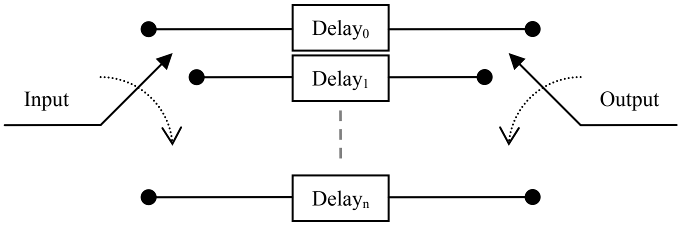


Figure 2.22 General structure of a convolutional interleaver

semi-random manner (instead of vertically or horizontally).

There are non-block interleavers such as convolutional interleavers that consist of *n* parallel lines with various digital delays. The input bits are sequentially written to the lines and the output is the bits which are synchronously read from the lines. Figure 2.22 depicts the structure of a convotional interleaver [77]. Generally convolutional interleavers have a better performance in terms of delay and overall Bit Error Rate.

If the delay in each line of the convotional interleaver is increased by one, the interleaver is called a Declined Interleaver. Declined interleavers are utilized to reduce the delay and jitter caused by the interleavering process. Figure 2.23 is a block diagram of a simple declined interleaving process. The input blocks are written horizontally in the memory cells where they have been shifted (or delayed) incrementally, and are read vertically (arrows).

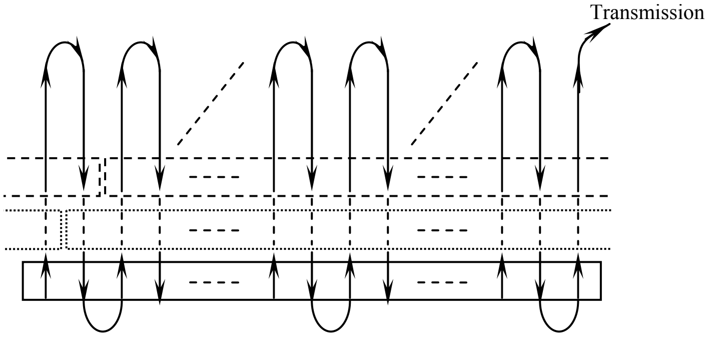


Figure 2.23 Example of a declined interleaver

2.6 Security Services

Many of the protocols in communication systems do not provide security. Tools to "sniff" passwords on the network are in common use by malicious hackers. Thus, applications which send an unencrypted password over the network are extremely vulnerable. Worse yet, other client/server applications rely on the client program to be honest about the identity of the user. Other applications rely on the clients to restrict their activities to those which they are allowed to do, with no other enforcement by the server.

There are four concerns that a security system can address [11][78]:

- Confidentiality
- Authenticity
- Data integrity
- Non-repudiation

Confidentiality is the protection of transmitted data from attackers with respect to the release of message contents. Usually, this is done by encryption functions which transform the message into unreadable information. The function has a property that only someone in the possession of a specific key can reverse the effect (decryption) and recover the message. With no knowledge of the key no one can extract the message from the unreadable information by using reasonable feasible resources [11], which means it is computationally safe. Therefore, if the sender encrypts all private messages before transmitting them through an insecure channel, only the receivers that know the specific key can extract the original message.

User *Authenticity* is a process where a user establishes the right to an identity; that is, in essence, the right to use a name. There are a large number of techniques that can be applied to authenticate a user, including passwords, biometric techniques, smart cards, and digital certificates; the last of which is used widely in cryptosystems [81].

Even after a user is authenticated, it is necessary for the data that is sent to be checked continuously to verify that it has been sent by the authorized user. Due to the use of unreliable communication channels, it is likely that the received data has not been sent by an authorized (expected) user or data source. The eavesdropper, the attacker of the system, can change the message that has been sent through the channel or even completely replace it with a new message. To protect the system from this kind of attack, *data authentication* is employed to ensure the integrity of the received message.

Data Integrity or *Data Authentication* is a critical component in acquiring and maintaining trustworthy data. When it is transferred between locations where it can be intercepted and/or substituted, the potential for unlawful modification of the data exists. Data authentication is a cryptographic technique used for detecting this modification or replacement. Note that the data authentication does not change the data in any way (i.e., the data is not encrypted); rather, adds an authentication tag that accompanies the data. Generally, data authentication algorithms are used to detect unauthorized modifications, both intentional and accidental, to data. Hashed Message Authentication Control (HMAC), digital signatures are examples of protocols that can be used in data integrity.

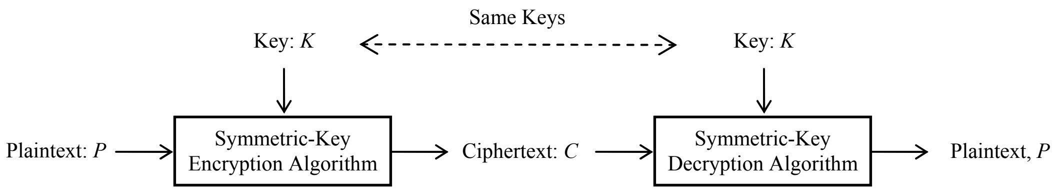
Non-repudiation prevents legitimate parties, involved in a communication, from denying its existence or the contents of the data they sent. Thus, when a message is sent, the receiver can prove that the message is, in fact, sent by the alleged sender. In a cryptosystem, this is usually handled by *digital signatures*. It indicates who has signed the data, and ensures that the contents have not been altered, and is also used as a means of user identification. In schemes, where digital IDs have been assigned to all authorized parities in the communication, a certificate is issued to authenticate the valid users and their public keys. These are commonly referred to as *Public Key Certificates.* They and their infrastructure, Public Key Infrastructure (PKI), are briefly explained in the next section.

Digital signature and encryption systems are expected to become an essential part of doing business via digital links. Based on a range of encryption techniques, digital signature systems allow people and organizations to electronically certify such features as their identity, their ability to pay, or the authenticity of an electronic document. Policies governing the collection of information for digital signatures, and the architecture and legal liabilities associated with these technologies, raise important privacy and consumer protection issues. Encryption and authentication are discussed in the next two sections.

2.6.1 Encryption

Encryption is the process of encoding information with a key in such a way that only the user (or computer) with the specific key and process algorithm can decode the encryption. Encryption algorithms are characterized in two categories: *symmetric-key* and *asymmetric-key* encryptions, as illustrated in Figure 2.24 and Figure 2.25 [11].

In a symmetric-key algorithm, the encryption key is related to the decryption key in the way that they are identical or there is a simple transformation between the two keys. The keys, in practice, represent a shared secret between two or more parties that want to maintain a private information link.



Same Keys
Key: K
Key: K
Plaintext: P
Symmetric-Key Encryption Algorithm
Ciphertext: C
Symmetric-Key Decryption Algorithm
Plaintext, P

Figure 2.24 Symmetric key cryptography

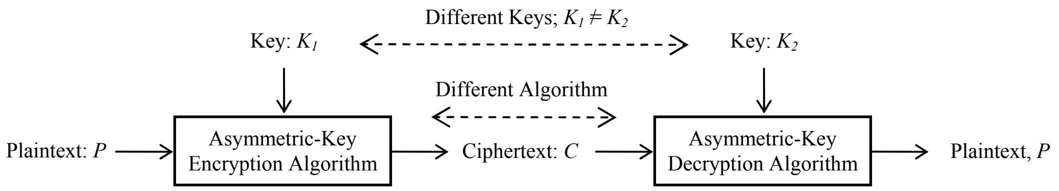


Figure 2.25 Asymmetric key cryptography

For Symmetric-key algorithms there are two formats: *stream cipher* and *block cipher* [11].

In the latter, a block of plaintext is treated as a whole and used to produce a ciphertext of equal or greater length as signified in Figure 2.26. This is convenient when dealing with stored data or data processed by a computer.

As shown in Figure 2.27, in a stream cipher, a digital data stream is encrypted one bit or one byte at a time. This is a convenient form when the data is already present as a serial data stream, such as in the serial transmission systems.

As mentioned, stream ciphers are a type of symmetric encryption algorithm. Stream ciphers are designed to be exceptionally fast, much faster than any block cipher. Block ciphers operate on large blocks of data, and stream ciphers, typically, operate on smaller units of plaintext, usually bits. The encryption of any particular plaintext with a block cipher results in the same ciphertext, when the same key is used. With a stream cipher, the transformation of these smaller plaintext units varies, depending on when they are encountered during the encryption process. A stream cipher generates what is called a *key stream* (a sequence of bits used as a key). Encryption is accomplished by combining the key stream with the plaintext, usually with the bitwise XOR operation, illustrated in Figure 2.27.

The disadvantage of symmetric-key algorithms is the requirement of a *shared secret key*, with one copy at each end. Since keys are subject to potential discovery by a cryptographic adversary, they need to be changed often and kept secure during distribution and in service. The consequent requirement to choose, distribute, and store keys without error and without loss is difficult to achieve confidently.

Some examples of popular and well-respected symmetric algorithms include: AES (aka Rijndael) [82], Blowfish [83], CAST5 [80][83], IDEA [84], RC4[83], Serpent [85][81], 3DES [86], and Twofish [85]. In 2001, the Advanced Encryption Standard algorithm was approved by NIST [81].

Asymmetric-key encryption is also known as a *Public key cryptography*. Typically, it allows users to communicate securely without having access to a shared secret key. This is accomplished by using a pair of cryptographic keys, designated as a *public key* and a *private key*, which are related

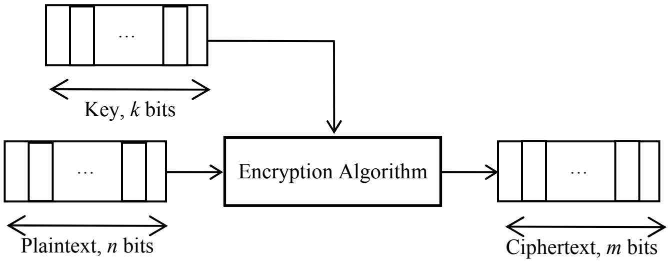


Figure 2.26 Block cipher

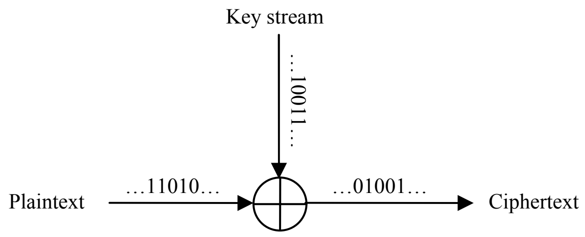


Figure 2.27 Stream cipher

mathematically [11]. Such keys enable users to make some parameters, including the encryption key, public. If the users want to send an encrypted message to a member of the public key system, they can take the corresponding public parameters of the member and encrypt their information with them. The public key schemes ensure that only those, whose public parameters are used for encrypting the information and possess a unique secret key, can decrypt the information (Figure 2.28). Also, the scheme ensures that by knowing the public information and possessing the encrypted data, it is computationally infeasible for some one to decrypt data without knowledge of the secret parameters and secret key. There is no need to say that the public and secret parameters are securely correlated, that is, for each member #*i* and each message *P*, the following

$$g_{m(i)}(f_{m(i)}(P)) = P$$

is valid.

Typically, public key algorithms are computationally complex, compared with many symmetric key algorithms of equivalent security. In practice, this means that a quality asymmetric key algorithm is hundreds or thousands of times slower than a quality symmetric key algorithm. Examples of well-regarded public key techniques include: the Diffie-Hellman, ElGamal, Elliptic Curve encryption algorithm (ECC), and RSA encryption algorithms [11][80][81][83]. Symmetric-key algorithms requir less computation than asymmetric key algorithms of equivalent secuirty strength.

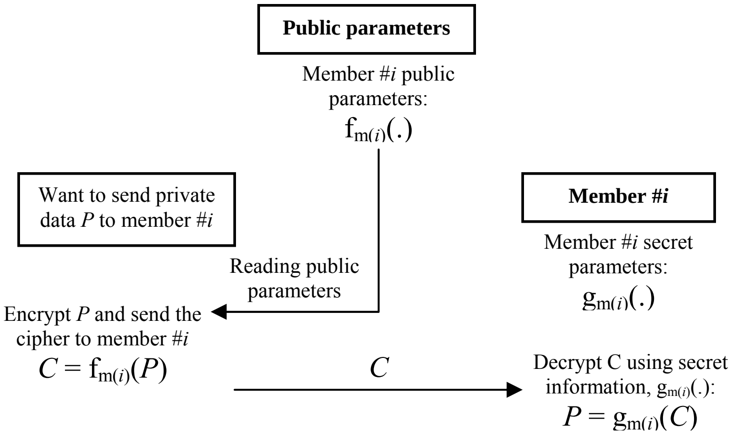

Public parameters
Member #*i* public parameters:
$f_{m(i)}(.)$
Want to send private data *P* to member #*i*
Member #*i*
Member #*i* secret parameters:
$g_{m(i)}(.)$
Reading public parameters
Encrypt *P* and send the cipher to member #*i*
$C = f_{m(i)}(P)$
C
Decrypt C using secret information, $g_{m(i)}(.)$:
$P = g_{m(i)}(C)$

Figure 2.28 Typical public key scheme

2.6.2 SAC Property

Far more effort has gone into analyzing block ciphers. They are applicable to a broader range of applications than stream ciphers. The vast majority of network-based conventional cryptographic applications make use of block ciphers. Accordingly, the concern in this chapter and the discussion throughout the thesis is solely on block ciphers.

A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext. In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext. This is called the *avalanche effect* [11][87]. If the change is small, this might provide a way to reduce the size of the plaintext or the key space to be searched [88]. This concept can be extended to *Strict Avalanche Criterion* (SAC), where changing a single bit in an information-block results in changing almost half of the bits on average in the output block [11]. If a cryptographic function is to satisfy the strict avalanche criterion, then each output bit should change with a probability of one half, whenever a single input bit is changed. The SAC was introduced by Webster and Tavares [116].

This is also true for decryption. If a small change occurs in the ciphertext, the plaintext, produced by that ciphertext and the correct key, should not be similar to the original plaintext as seen in Figure 2.29. In this way, an attacker cannot estimate the plaintext by examining different forms of the ciphertext. Due to this property, if a channel error occurs in the ciphertext, the recovered plaintext is corrupted. This is why, when cryptographic functions such as encryption are employed, the sensitivity of the system to errors increases, and it is crucial that the blocks which are delivered to the decryption block do not contain any error.

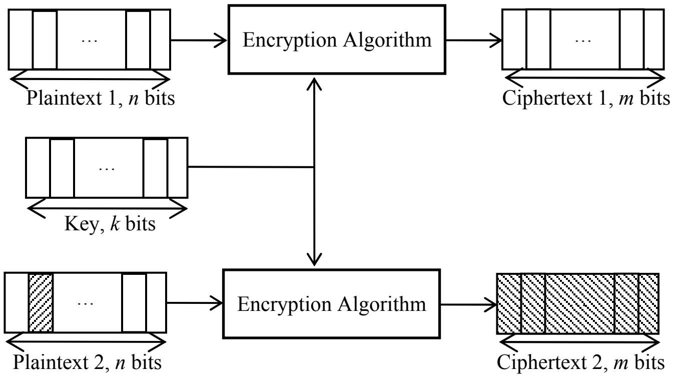


Figure 2.29 SAC property, plaintext 1 and 2 are different in only one bit. Bits with the same position in ciphertext 1 and 2 are different with probability of 1/2

2.6.3 Data Integrity Checks

Data integrity checks in secure connections are performed by either digital signature or hash functions.

Typically, digital signatures are more complex and require more processing power compared with hash functions. Digital signatures also provide non-repudiation services as well, and are used to detect unauthorized modifications of the data and to authenticate the identity of the signatory. In addition, the recipient of the signed data can use a digital signature for proving to a third party that the signature is, in fact, generated by the signatory. This is known as non-repudiation since the signatory cannot, at a later time, repudiate the signature**.**

A hash function is an essential element in digital signatures, and is present in almost all digital signature algorithms. As shown in Figure 2.30(a), Hash function, H(.), is a one-way function with a SAC property which converts any arbitrary length input to a fixed output size [11]. The output of the hash function is referred as the *hash value*. It is obvious from a mathematical perspective that there are many inputs that have the same hash value (Collision, Figure 2.30(b)) [11][78][89]. One of the key properties of hash functions is that it must be computationally difficult to find two inputs which have the same hash values. SHA-1 and MD5 are examples of hash functions.

If a secret key is used to calculate the hash value, the function is referred to as a keyed hash function.

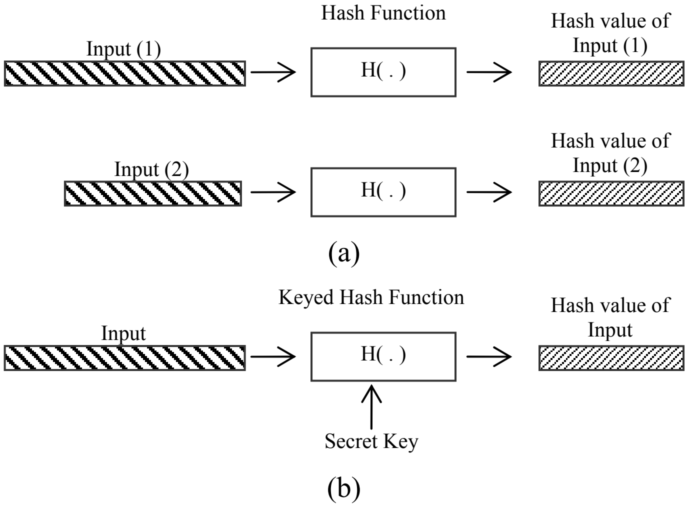


Figure 2.30 Hash function: (a) a collision in hash functions occurs when the hash values of two different inputs are identical and (b) keyed hash function

In the next sections a few well-known digital signature schemes are explained. Also it is demonstrated how hash functions are used in signature generation and verification processing.

2.6.3.1 RSA Signature

In addition to public key encryption, RSA is employed to sign a message. Suppose Alice wishes to send a signed message to Bob. She produces a hash value of the message, raises it to the power of *d* mod *n* (as she does when she is decrypting a message), and attaches the resulting number as a signature to the message. When Bob receives the signed message, he raises the signature to the power of *e* mod *n* (as he does when he is encrypting a message), and compares the resulting hash value with the message's actual hash value. If the two agree, Bob knows that the author of the message is in possession of Alice's secret key, and that the message has not been tampered with. Anyone can verify a signed message, but only someone in possession of the correct private key can sign a message. By using RSA, authentication occurs without any sharing of private keys: each user uses only other's public key and/or his own private key [11].

2.6.3.2 DSA

In 1994, the National Institute of Standards and Technology (NIST) [81] published the Digital Signature Algorithm (DSA) in the Digital Signature Standard (DSS). DSA is being incorporated into a number of systems and specifications. A minor revision was issued in 2000 as FIPS 186-2 [90], and the standard was expanded further in 2006 as FIPS 186-3 [91].

The DSA is based on the discrete logarithm problem and is related to signature schemes that have been proposed by Schnorr and ElGamal [81]. Although RSA can be used for both encryption and digital signatures, the DSA can only be used to provide digital signatures.

The Key Generation to establish a DSA digital signature scheme follows:

- Choose a 160-bit prime q.
- Choose an L-bit prime p such that $p=qz+1$ for some integer z and such that $512 \leq L \leq 1024$ and L is divisible by 64.
- Choose h, where $1 < h < p - 1$ such that $g = h^z \bmod p > 1$.
- Choose x by some random method, where $0 < x < q$.
- Calculate $y = g^x \bmod p$.

Public key is (p, q, g, y). Private key is x.

The DSA standard specifies the Secure Hash Algorithm, SHA1. Note that FIPS-186-2, change notice 1 specifies that L should assume the value 1024 only, and the FIPS 186-3 uses SHA-224, SHA-256, SHA-384, and SHA-512 as a hash function, q of size 224, 256, 384, and 512 bits with L equal to 2048, 3072, 7680, and 15360, respectively.

To sign a message m, the following steps are taken

- Generate a random value k where $0 < k < q$.
- Calculate $r = (g^k \bmod p) \bmod q$.
- Calculate $s = (k^{-1}(\mathrm{H}(m) + x*r)) \bmod q$, where $\mathrm{H}(m)$ is the hash function applied to the message m .
- Recalculate the signature in the unlikely case that $r = 0$ or $s = 0$.
- The signature is (r, s) .

The verification is as follows

- Reject the signature if either $0 < r < \mathrm{q}$ or $0 < s < \mathrm{q}$ is not satisfied.
- Calculate $w = (s)^{-1} \bmod q$.
- Calculate $u1 = (\mathrm{H}(m)*w) \bmod q$.
- Calculate $u2 = (r*w) \bmod q$.
- Calculate $v = ((g^{u1}*y^{u2}) \bmod p) \bmod q$.
- The signature is valid if $v = r$.

In DSA, the signature generation is faster than the signature verification, whereas with RSA, the signature verification is much faster than the signature generation (if the public and private exponents, respectively, are chosen for this property, which is the usual case [92]). It might be claimed that it is advantageous for the signature to be the faster operation, however, since in many applications a block of digital information is signed once, but verified often, it might well be more advantageous to have a faster verification. Wiener has explored the trade-offs and issues involved in DSA [93]. There has been work by many authors, including Naccache et al. [94], on developing techniques to improve the efficiency of DSA, both for signature and verification.

2.6.3.3 ECDSA

The Elliptic Curve Digital Signature Algorithm (ECDSA) is a variant of the DSA which is based on Elliptic Curves [81]. Elliptic curve cryptosystems (ECC) were first proposed independently by V. Miller [95] and N. Koblitz [96]. At a high level, ECCs are analogs of the existing public-key cryptosystems in which modular arithmetic is replaced by operations defined over elliptic curves [11][81][83].

Like the security of all public-key cryptosystems, the security of ECC relies on the underlying difficult-to-solve mathematical problems. The security of such systems depends on the following hard problem: Given two points, G and Y, on an elliptic curve such that $Y = kG$ (i.e., Y is G added to itself k times [11]), find the integer k. This problem is commonly referred to as the *elliptic curve discrete logarithm problem*.

Presently, the methods for computing general elliptic curve discrete logarithms are much less efficient than those for factoring or computing conventional discrete logs [11]. As a result, shorter key sizes can be used to achieve the same security as that of conventional public-key cryptosystems. Recently, ECC has attracted the attention of both the academic community and industry.

Elliptic curves can be also used for digital signatures. The next few paragraphs summarize the ECDSA scheme.

Suppose Alice wants to send a signed message to Bob. Initially, the curve parameters (including generating point G) must be shared [11][80]. Also, Alice must have a key pair, suitable for ECC that consists of private key d_A (a randomly selected integer in the interval $[1, n-1]$) and public key Q_A (where $Q_A = d_AG$).

For Alice to sign a message m, she follows these steps

- Calculate $e = \mathrm{H}(m)$.
- Select a random integer k from $[1,n - 1]$ (nonce).
- Calculate $r = x_1 \pmod{n}$, where $(x_1, y_1) = kG$. If $r = 0$, select another nonce and calculate r again.
- Calculate $\mathrm{s} = k^{-1}(e + d_Ar) \pmod{n}$. If $s = 0$, select another nonce and recalculate s and r.

The signature is the pair (r, s).

To authenticate Alice's signature, Bob must have a copy of her public key Q_A. He proceeds with the following.

- Verify that r and s are integers in $[1,n - 1]$. If not, the signature is invalid.
- Calculate $\mathrm{e} = \mathrm{H}\ (m)$.

- Calculate $w = s^{-1}$ (mod n).
- Calculate u_1 = ew (mod n) and $u_2 = rw$ (mod n).
- Calculate $(x_1, y_1) = u_1G + u_2Q_A$.

The signature is valid if $x_1 = r$ (mod n).

There are techniques to increase the calculation speed in ECDSA. For instance, it is possible to calculate $u_1G + u_2Q_A$ faster by using Straus's algorithm (a.k.a. Shamir's trick) than to calculate the sum by using conventional scalar multiplications [97].

2.6.3.4 Public Key Certificate

A public key certificate (or identity certificate) is a certificate, where a digital signature is combined with a public key. Therefore, public keys are binned with an identity, such as the name of a person or an organization, and their addresses. Thus, the certificate can be used to verify that a public key belongs to a user.

In a typical Public Key Infrastructure (PKI) scheme [11][81], the signature is of a *Certificate Authority* (CA). In a secure network, the signature is either that of the user (a self-signed certificate) or of other users (endorsements). In either case, the signatures on a certificate are authenticated by the certificate issuer such that the identity and the public key are combined.

Figure 2.31 conveys how a CA is used to establish an authenticated private channel. Assume Bob, a CA member, wants to establish an authenticated secure channel to Alice, another member of CA. First, he sends his ID (ID_{Bob}) and public key (PK_{Bob}), along with his certificate ($\text{Sign}_{\text{CA}}(\text{ID}_{\text{Bob}}, \text{PK}_{\text{Bob}})$) to Alice (1). Alice verifies the integrity of Bob's ID and public key by using the membership list of the CA, and verifying CA's signature on Bob's ID and public key (2). Then, she generates a random number ($\text{nonce}_{\text{Alice}}$) and encrypts it with Bob's public key. She sends her ID, public key, and her certificate, with the encrypted random number and her signature on it, to Bob (3). After Bob's verification of Alice's ID and public key (4), he generates his own random number and encrypts it using Alice's public key (5). At this time, both Alice and Bob can decrypt the received random numbers and generate a shared key (session key) by using a pre-defined function in the network (f(.)). Then, the session key is used to encrypt the rest of the communication. In some networks, the session key might change several times during a communication. Also, the certificate itself has a life time and needs to be renewed.

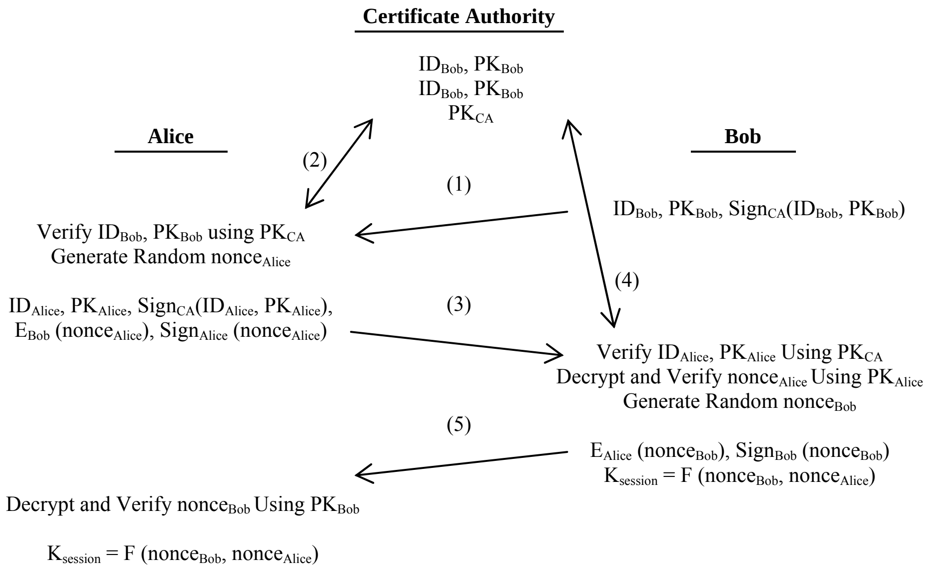

Certificate Authority
ID_{Bob}, PK_{Bob}
ID_{Bob}, PK_{Bob}
PK_{CA}
Alice
Bob
(2)
(1)
ID_{Bob}, PK_{Bob}, $Sign_{CA}(ID_{Bob}, PK_{Bob})$
Verify ID_{Bob}, PK_{Bob} using PK_{CA}
Generate Random $nonce_{Alice}$
(4)
ID_{Alice}, PK_{Alice}, $Sign_{CA}(ID_{Alice}, PK_{Alice})$,
E_{Bob} ($nonce_{Alice}$), $Sign_{Alice}$ ($nonce_{Alice}$)
(3)
Verify ID_{Alice}, PK_{Alice} Using PK_{CA}
Decrypt and Verify $nonce_{Alice}$ Using PK_{Alice}
Generate Random $nonce_{Bob}$
(5)
E_{Alice} ($nonce_{Bob}$), $Sign_{Bob}$ ($nonce_{Bob}$)
$K_{session}$ = F ($nonce_{Bob}$, $nonce_{Alice}$)
Decrypt and Verify $nonce_{Bob}$ Using PK_{Bob}
$K_{session}$ = F ($nonce_{Bob}$, $nonce_{Alice}$)

Figure 2.31 Authentication and secure communication of the Certificate Authority (CA) scheme

Digital signatures are widely used to protect data in secure e-mail systems [98][99].

2.7 Wireless Application Protocol Architecture

In wireless networks, there are several protocols and standards. Typically, wireless protocols are implemented in the physical and transport layers. The Wireless Application Protocol (WAP) architecture [100] is shown in Figure 2.32. The wireless bearer deals with the actual connection of the devices. The primary concern is to change the bits to signals that can be sent to the antenna. Channel coding and signalling techniques are implemented in this layer.

The transport layer involves flow control, error detection, error correction, retransmission, Multiple Accessing (MA), and carrier sensing to increase the transmission reliability. Since all the wireless devices share the transmission medium to send and receive information, collision detection and collision avoidance must be employed in the transport layer. The transaction layer manages the data and creates the data packets.

In the next section, a few energy conservation techniques, to extend the mobile devices' battery-life, in secure wireless networks are reviewed.

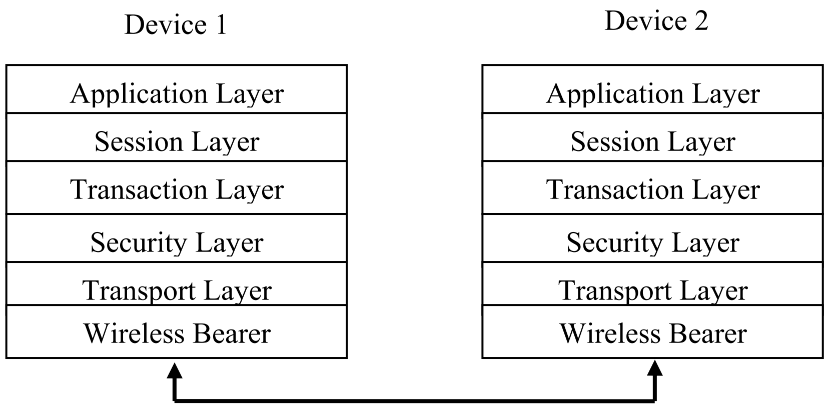


Figure 2.32 Wireless Application Protocol (WAP) architecture

2.8 Energy Costs and Security Requirements

In order to use security services, the system requires more algorithms and specific processes that results in time and energy costs increase. Setting up the security requirements and manipulating the data delays the transmission, and performing the security processes consume additional energy. When the security processes are included in time and power limited systems such as battery-powered mobile handheld devices, these costs significantly increase. The energy cost is the most important performance metric in wireless networks that consist of limited energy resource devices, such as WSN (Wireless Sensor Network) [101][102], WLAN (Wireless Local Area Network) [103], and WPAN (Wireless Personal Area Network) [104].

Also, it has been demonstrated that a system using security services can consume almost twice the energy of a system which does not use the service [105]. Different approaches have been taken to minimize the energy consumption of the security services. P. Prasithsangaree et al. have concentrated on the symmetric-key algorithm [106], and Potlapally et al. [107] has studied the power consumption of asymmetric-key algorithms. The energy cost of different AES modes has also been shown in [106]. The AES in the Electronic Code Book (ECB) mode consumes less energy than Cipher Block Chaining (CBC), Cipher FeedBack (CFB), and Output FeedBack (OFB) modes ([11][80][106]).

Key setup refers to the operation in which the input key is expanded in order to derive a distinct and cryptographically strong key for each of the rounds. Key setup is another issue which must be considered. Symmetric-key algorithms usually run for several iterations of a substitution and/or permutation on their input to produce the output [11][80]. Since different algorithms require different key setup operations, the energy consumption of the key setup should be taken into consideration [106].

In some applications, the trade-off between energy consumption and the security services' strength is an option as well [106].

A thorough study has been conducted by Karri et al. [108] who have considered the total energy needed for the Message Authentication Code (MAC) generation/verification using SHA-256, data encryption/decryption, transmission/reception. They have compared a system with security enabled services with the same system when all the security services are disabled. A more detailed description of the test bed and measurement results can be found in [109].

Security can be provided at different layers of the protocol stack. Karri et al. [110] has employed IPSec security protocol to measure the increase in energy consumption of a mobile device where a secure connection is maintained and compared their results to the non secure connection. WTLS, as security provider for WAP services, has been explored in [92] where the time efficiency of different asymmetric-key algorithms is compared.

Also, the energy costs of digital signature algorithms have been measured [105]. A client-server communication, where the server is a PC and a Compaq iPAQ H3670 Handheld is used as the client, is explored. Table 2.2 lists the energy costs of various digital signature algorithms at different steps at that setup [105].

Table 2.2 Energy cost of different digital signature algorithms

Scheme	Key Size	Signing	Verifying
RSA	1024	546.5 mJ	15.97 mJ
ECDSA	160	134.2 mJ	196.23 mJ

As mentioned, the most constraints are on handheld devices, and it is desirable to shift the heavy computation and time consuming processes to a fast server. For a digital signature protocol, a handheld depends on ECDSA for signing messages and RSA for verifying the digital signatures.

Since the total energy for signing and verification can be minimized on a handheld device, the handheld device must consume the minimum energy and time to perform the ongoing digital signature scheme.

Table 2.3 summarizes the energy consumption based on the results on [105] and [111] for three different cases by using the 1024-bit RSA and 160-bit ECDSA digital signature scheme:

- 1-both sides use RSA,
- 2-both sides use ECDSA, and
- 3-Slower party, the handheld device uses ECDSA to sign and RSA to verify messages and the fast party, the server, uses RSA and ECDSA to verify and sign, respectively.

Table 2.3 Total energy used in client and server on performing one digital signature and one verification

Sign and Transmit + Receive and Verify a Signature			
Scheme	Key Size	Handheld	Server
RSA/RSA	1024	563.82 mJ	563.82 mJ
ECDSA/ECDSA	160	330.65 mJ	330.65 mJ
RSA/ECDSA	1024/160	150.95 mJ	743.51 mJ

It can be seen that the RSA/ECDSA scheme has the lowest energy cost for handheld devices. The efficiency of RSA/ECDSA scheme on more platforms has also been studied by Daswani [92].

Also, the processing power limit affects the required time for performing complex calculations of such algorithms as digital signatures. Since, the hybrid RSA/ECDSA scheme shifts the heavy computation processes to the server, the handheld requires less time to generate or verify a digital signature than the other schemes. H. Little [111] measurements have demonstrated that the hybrid RSA/ECDSA can run 20% faster than the ECDSA for the digital signature generation and verification as indicated in Table 2.4.

By considering both the reliability and security, different protocols can be defined to maximize the overall performance of a secure connection. Sometimes privacy is achieved without the use of encryption. Rivest [112] has introduced *chaffing and winnowing* in order to achieve confidentiality without using encryption. There, the transmitter and the receiver share a secret key that is used with a keyed-hash function. The transmitter sends real and fake messages. For the real messages, the transmitter calculates the hash value and attaches it to the message. For the fake messages, the transmitter sends a packet of random bits. The receiver compares the hash it calculates with the hash values that are received, and retains only the messages where the hashes match. In this way, an attacker cannot determine which packets are the ones of the real message. One of the interesting advantages of this technique is that it can be used in the network's lower layers such as the Data Link Layer. Also, Chaum [113] has developed a data reordering technique to obtain confidentiality. In this technique each packet contains unencrypted data and an encrypted order number. Although the mixed ordered packets are not meaningful, the receiver with the shared key can recombine the packets back in the correct order by decrypting the order of the packets. This transposition can be carried out on an even finer level such as portions of bytes or bits. A hybrid approach of chaffing and winnowing and Chaum's order mixing concept, has also been included to create more confusion [114]. Privacy can be further improved by increasing the resources of the system. For example, Simov et al. [115] have reported that privacy can be provided in multi-channel systems without using encryption.

The mobility of devices and time-variant characteristics of a channel in most wireless connections affects the reliability and the minimum transmission power channel needed to achieve the QoS. By examining the power needed to maintain the QoS and security level in a wireless communication, the investigation described in this thesis, is an attempt to devise techniques to conserve energy.

Table 2.4 Client and server time on performing digital signature and verification on a Blackberry device [111]

Scheme	Key Size	Handheld	Server
RSA/RSA	1024	747 ms	747 ms
ECDSA/ECDSA	160	118 ms	118 ms
RSA/ECDSA	1024/160	98 ms	767 ms

2.9 Summary

There are more concerns about wireless channels than wired channels. The wireless channel parameters are time variant. Changes in the transmitter's or receiver's location, speed, or direction; variations in the environment's temperature; and the movement of nearby objects directly affect the characteristics of the wireless channel. In this chapter, the models which are a better fit for wireless channels, their categories, and the different techniques which are used to handle the errors in a connection are explored.

In addition, the structures of the error handling techniques and their codes are reviewed, since they are used in the proposed models later in Chapter 3. The principal concepts of two renowned decoding algorithms, the Viterbi and the Turbo Code, are explained, since they are used in the proposed models. It is also demonstrated how different techniques can be used to reduce the energy consumption in Turbo Code channel decoders. Moreover, the effect of the security functions on the performance and the energy consumption of the systems is studied. In the next chapter, the modification of the decoding algorithms and the newly devised error handling technique to construct the proposed models are presented.

Chapter 3

Retransmission Reduction in Secure End-to-End Wireless Communication

Applying security functions such as privacy, authentication, and digital signatures on a wireless connection increases the error sensitivity and overhead and decreases the overall performance of the communication system. An increase in error sensitivity is primarily caused by the security functions; a single residual bit error in a received block of encrypted information renders the entire block unreadable [11][117][118].

In applications which require secure end-to-end wireless connection, the complexity is increased to obtain a higher level of reliability and system performance. The added complexity affects various aspects of the communication. In this thesis, it is proven that under certain conditions of the communication channel the overall performance of the system improves by increasing the complexity.

In this chapter, a number of techniques are presented to achieve better performance in a secure end-to-end wireless connection by reducing the number of retransmissions such that the energy consumption for retransmission is reduced. This is especially important in wireless mobile devices such as cell phones and PDAs which have a limited access to energy.

3.1 Retransmission Rate Reduction Techniques

In most communication systems, such as TCP/IP, ARQ schemes are used to augment reliability. As explained in Section 2.2.2, such techniques are based on the retransmission of corrupted packets by sending back a negative acknowledgment to the sender. The receiver must use error detection coding techniques to highlight the corrupted packets. Such codes add more bits to the transmitted packet. When the received packets are likely to have few bit errors, HARQ [Section 2.2.3] techniques are usually used. They decrease the number of retransmissions by reconstructing the correct information from the corrupted packets [59]. This leads to better delay characteristics at the expense of more overhead and complexity.

In these systems, the message cannot be delivered until either a correct copy arrives at the receiver end or it is recovered from the errors, which increase the average delay of the system. The aggregate throughput of the system with retransmission depends first on the number of retransmissions and secondly, on the packet overhead [59]. Generally, more retransmissions lead to a lower throughput, higher average delay, and higher energy consumption.

In the next section, a technique, which uses the principle of channel coding to reduce the number of retransmissions when the communication channel is noisy, is described. By avoiding retransmissions, the communication system attains a higher throughput and better delay characteristics.

3.1.1 *K* Best Likelihood (*K*-BL) Block Decoder

Noise in a channel introduces errors in the transmitted information in such a way that it might not be decoded into a correct block in the receiver. When the ratio of the receiving signal power to the channel noise power, SNR, decreases, more blocks are likely to be decoded incorrectly. In other words, the probability that the number of errors occurre in one block is higher than the correction capability of the channel decoder increases. To deliver error-free messages in the ARQ, the receiver must request the sender to retransmit the block each time an error in the decoded blocks is detected. In HARQ systems, the retransmission is requested when the number of errors in the decoded block is larger than the number of correctable bits. In this case, the rate of the block retransmission goes up, and the delay and throughput characteristics of the communications system deteriorate. When the receiver detects uncorrectable errors in the decoded block, it is possible that the receiver checks a few other probable blocks before discarding all the information of the decoded block. If any of the probable blocks is the correct one, the receiver avoids sending retransmission requests to the sender.

For the realization of this system, a channel decoder, which produces *K best likelihood* (*K*-BL) blocks after receiving a codeword from the channel, is chosen. Note that for K=1, the channel decoder is exactly the same as that of a conventional channel decoder which outputs only the maximum likelihood block.

The probability that the correct block appears among the K best likelihood decoded blocks is higher than or equal to the probability that the correct block is actually the best likelihood decoded block. In the other words,

$$\begin{aligned}&\text{Probability (correct block = maximum likelihood decoded block)} \leq \\ &\qquad\qquad \text{Probability (correct block} \in \{K\text{-BL decoded blocks}\}). \end{aligned} \tag{3.1}$$

To prove this inequality, let P_i denote the probability that the correct block being decoded is the i^{th} maximum likelihood block in the K best likelihood channel decoder. According to this definition, the probability that the correct block being decoded as the maximum likelihood block, is P_1, such that

$$\text{Probability (correct block = maximum likelihood decode block)} = P_1 \, . \tag{3.2}$$

This is the probability that a conventional channel decoder delivers the correct block. The probability that the correct block appears among the K best likelihood decoded blocks is the summation of the all P_{i} s from 1 to K. Thus,

$$\text{Probability (correct block} \in \{K\text{-BL decoded blocks}\}) = \sum_{i=1}^{K} P_i \tag{3.3}$$

Considering $0 \leq P_i \leq 1$ and $K \geq 1$, it is concluded that

$$P_1 \leq \sum_{i=1}^{K} P_i = \text{Probability (correct block} \in \{K\text{-BL decoded blocks}\}). \tag{3.4}$$

This indicates that the probability that the K-BL channel decoder delivers the correct block is either equal to or higher than the probability of the correct block being decoded by a conventional maximum likelihood channel decoder.

The K probable blocks in the output of the K-BL channel decoder can contain the correct block. The blocks which contain errors must be detected and eliminated so that the system delivers the correct block, if it appears in the K probable blocks. From here on in the thesis, the blocks in the receiver that contain errors, are called *spurious* blocks. In fact, there is either K-1 or K spurious blocks among the K probable blocks. To eliminate the spurious blocks, an error detection technique must be used. It can eliminate the spurious blocks and find the correct block among the probable blocks. Therefore, a system, enhanced with an error detection algorithm as depicted in Figure 2.13, can examine the probable blocks to find the correct block. After failing to pass the error detection process, the spurious blocks are eliminated. The system delivers the block that passes the detection test to the data sink.

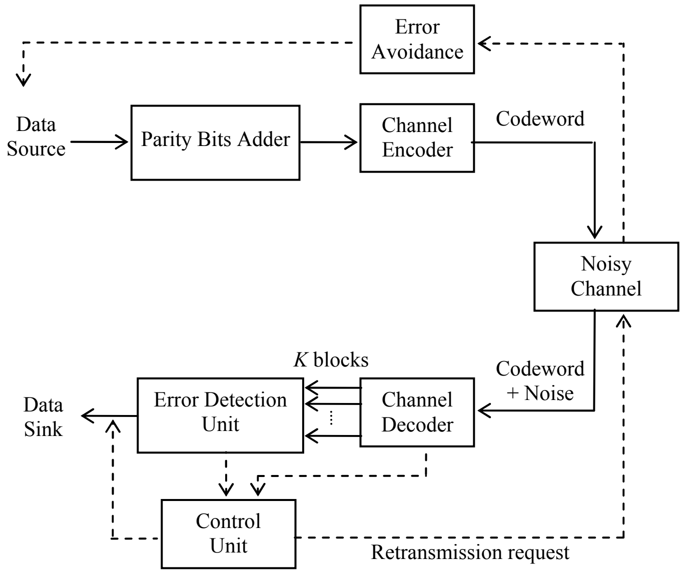


Figure 3.1 Communication system using *K*-BL channel decoder and error detection

In practice, error detection techniques might not detect the errors in the block. Consequently, it is probable that more than one block successfully passes the error detection process. In this case, the block with the higher likelihood probability is delivered. If none of the blocks passes the detection test, the system's control unit should request retransmission of the block as the last resort. Figure 3.1 is a diagram of a system with the *K*-BL channel decoder.

The performance of the model in Figure 3.1 depends on the accuracy of the error detection process. In next section, a technique, which improves the error detection capability in the receiver, is introduced. It uses the characteristics of the security functions to gain more detection power. Also, further explanations to develop an error detection technique, which is limited to legitimate receivers only, are presented. This improves the reliability of the communication for the legitimate receivers and increases the confusion for the unauthorized receivers.

3.1.2 Private Error Detection (PED)

Error detection techniques are not ideal, and they can miss the detection of a spurious block. The spurious block, which passes the detection algorithm without being detected, is called *false positive* whose probability is denoted as $P_{f+,}$.

Due to the error occurrence in the parity bits, sometimes a correct block is rejected by the detection algorithm and is mistakenly considered as a spurious block.

Different detection techniques have different rates of false positive. To lessen the probability of a false positive, more parity bits must be included which increases the overhead of the system (decreasing the throughput). There is always a trade-off between the overhead of the parity bits and the accuracy of the detection algorithm [59]. Generally, if the patterns of the errors in the blocks are random, each extra parity bit cuts the probability of a false positive by half [11].

Usually, the data transmission is in a block format; for example, when the communication medium is shared. Consider a data stream from a source. To form a block, the source data is segmented. The size of these segments must be in a range which is determined by the block format. In most communication systems, the receiver performs error handling techniques on each received block; that is, the receiver checks each block of data individually and independently. Therefore, to combine the cryptographic functions with error handling techniques, block ciphers are more suitable. The reason is that the data in each block can be encrypted independently, and the errors in one block are not necessarily extended to those in other blocks. It is assumed that each block contains the encrypted message, along with the added parity bits which are used in error detection and error correction as portrayed in Figure 3.2.

The role of encryption is to conceal the contents of the messages from unauthorized receivers. Consequently, the encryption algorithms should have certain characteristics. One of them is the *Strict Avalanche Criterion* (SAC) [11][78][80][116]. It is possible to use the result of security function in a data integrity check process to enhance the error detection capability.

Assume the transmitter sends message *m* to the receiver and it is changed to *m'* at the receiver end. The change can be caused by noise, distortion, or any imperfection in the channel. Now, let $f_{SAC}(.)$

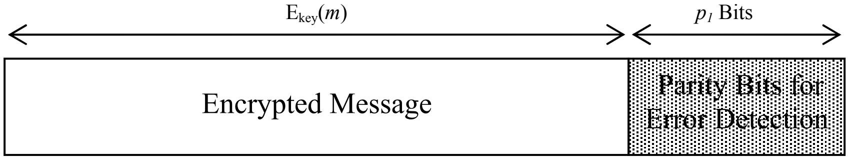


Figure 3.2 Example of a block used in secure reliable communication

represent a Boolean function with the SAC property, which accepts any arbitrary length of the input and converts it to a p_1-bit block. According to the SAC, if any bits of the transmitting message m are changed in the channel, the probability that $f_{SAC}(m)$ matches $f_{SAC}(m')$ is $(½)^{p_1}$. On the other hand, if the p_1-bit output of $f_{SAC}(m')$ matches the p_1-bits output of $f_{SAC}(m)$ then the message is decoded correctly with a probability of $1-(½)^{p_1}$. By generating the p_1-bit output of f_{SAC}(message) and sending it with the message, it is possible to detect errors at the receiver end with a probability of $1-(½)^{p_1}$ [58]. In this detection technique, only the SAC property of the $f_{SAC}(.)$ is used. Therefore, any function with the SAC property can be used in this technique.

In practice, each transmitted block of information contains several parity bits for the detection process. If the encryption of the blocks and/or the parity bits is not used, unauthorized devices can detect the correct blocks with the same efficiency and reliability as the authorized devices. If the parity-bit check algorithm uses a key to examine the correctness of the message, the error detection is only available to the parties that possess the correct key. This leads to a technique in which the error detection capability is granted only to the authorized receivers. The unauthorized receivers cannot check the correctness of the decoded message through the parity bits produced by the keyed data integrity function.

Since only the receivers who have the correct secret key can use the detection algorithm in this technique, it is referred as *Private Error Detection* (PED) technique. Also, the output of the keyed algorithm with the SAC property is called *private parity bits* because it is used to detect the error in the block of decoded data and they are only useable for the receivers who have the key. The private parity bits provide additional error detection capability to the legitimate receivers.

As mentioned, encrypted blocks are very sensitive to errors; a single bit error can affect the reliability of the whole block. Now, due to a noisy communication channel, the receiver requires additional error handling capability, which can be provided by the PED.

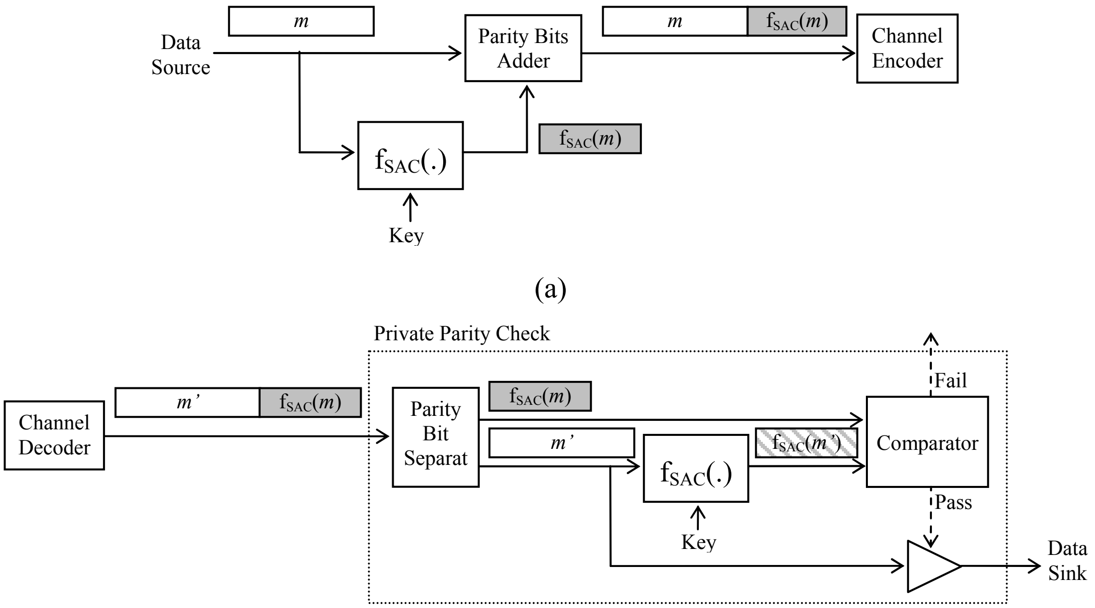
By using the PED technique, the authorized receiver holding the correct secret key is able to use private parity bits to detect and deliver the correct block with a lower probability of false positive than that of unauthorized receivers.

By using the functions in data integrity process and encryption process, PED technique are implemented. Figure 3.3 is a scheme of a communication system which relies on the PED technique. In this system the data integrity function is used to implement the PED. The system with the correct key is capable of including private parity bits to detect the spurious blocks.

In the systems which already use encryption to hide the contents of the transmitted block, the same functions can be used to conceal the private bits. In this case the private parity bits have fixed values, simplifying detection. It is noteworthy that the addition of the fixed bits or related bits in the encrypted block might reveal some information about the secret key in some cryptosystems. Figure 3.4 represents the flow of data in a system which uses data privacy functions (encryption and decryption) to implement the PED technique. In this figure, the private parity bits are assumed to be all zeros.

As shown in Figure 3.4, all of the decoded blocks are fed to the decryption process after they are processed by the detection algorithm. After decryption, the private parity bits of blocks will be checked to detect errors.

If the PED technique is used in parallel with another error detection technique, such as the CRC, the overall error detection of the system becomes more accurate. Figure 3.5 shows how the PED can be used along with another error detection technique.



(a)

(b)

Figure 3.3 PED technique implementation using data integrity function in (a) sender and (b) receiver

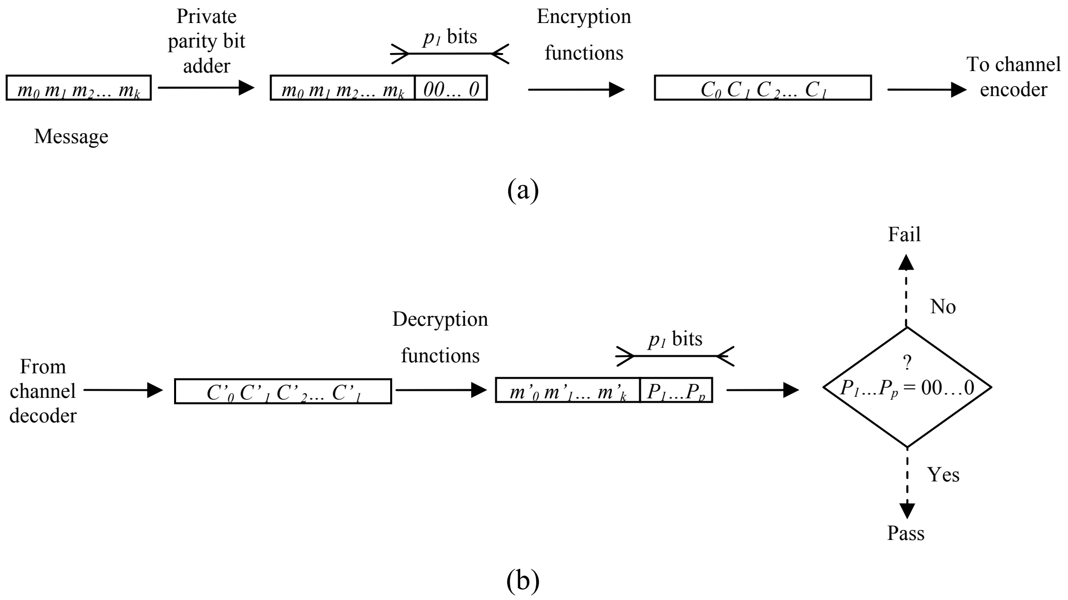

Private parity bit adder
p_1 bits
Encryption functions
To channel encoder
$m_0\, m_1\, m_2 \ldots m_k$
$m_0\, m_1\, m_2 \ldots m_k$ | $00\ldots 0$
$C_0\, C_1\, C_2 \ldots C_l$
Message
(a)
Fail
No
Decryption functions
p_1 bits
From channel decoder
$C'_0\, C'_1\, C'_2 \ldots C'_l$
$m'_0\, m'_1 \ldots m'_k$ | $P_1 \ldots P_p$
?
$P_1 \ldots P_p = 00\ldots0$
Yes
Pass
(b)

Figure 3.4 Flow of data in a system with PED technique implementation using data privacy functions in (a) sender and (b) receiver

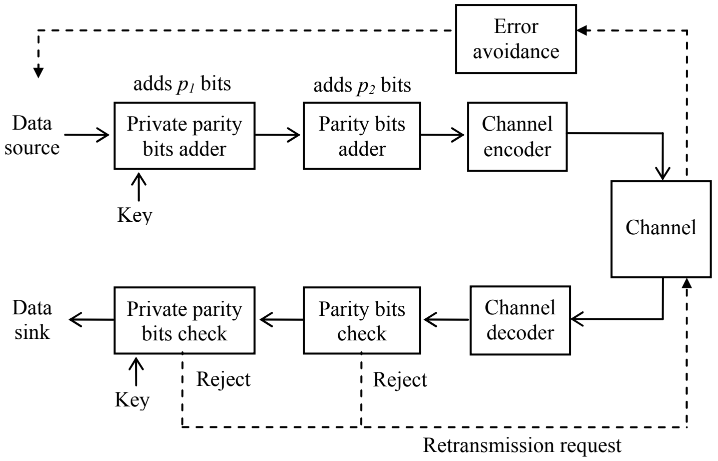


Figure 3.5 System which uses PED along with another error detection technique

If there are p_1 bits assigned as a private parity, the probability of the delivery of a spurious block, false positive, reduces 2^{p_1} times. The following equation shows the relation of the false positive probability of a system before and after using p_1 bits of private bits:

$$P_{f+,\text{ using PED}} = \frac{1}{2^{p_1}} P_{f+,\text{ without PED.}} \tag{3.5}$$

The enhancement of the error detection capability, provided by PED, is gained at the expense of a higher overhead caused by private parity bits and complexity. The numerical simulation results in Chapter 4 verifies that in certain channel conditions the overhead which is added by the private parity bits is far less than the overhead that is caused by a retransmission in the system with a less accurate error detection technique.

Also, since the implementation of the PED technique involves security functions, which are already used in the secure communication, the complexity and energy consumption of the sender and receiver are not changed dramatically.

3.1.3 Application Layer Pre-Coding and Interleaving

Generally, it is desirable to recover correct messages even after the receiver fails to decode the received packet correctly. The problem in an end-to-end secure connection is that the security function (e.g., decryption) converts a spurious block with a few error bits to a complete random block in the application layer. This block then acts like a burst error.

Burst errors are common in communication systems with fading channels. Usually, the fading effect is caused by multi-path, scattering, or blockage of the transmitted signal. Fading deteriorates the quality of the signal in the receiver for several bit times.

Principally, error correction codes are designed to correct a few randomly positioned errors in the block. Generally, they cannot correct burst errors, if the error size is more than their correction ability. Therefore, when a transmitted packet is affected by a burst error completely, block error correction techniques cannot recover the information, and the receiver has to request a retransmission of the packet.

However, there is another solution. With an interleaver, a burst error can effectively be distributed over several blocks. That is, an interleaver can be used to reduce the effective size of the burst errors. The erroneous bits appear in several blocks but are in much smaller size. Now, an error correction code can be designed to recover the correct information. Error correction codes are broadly used in different communication protocols and standards. The codes are well known and well studied, and often have straight forward implementation both in the software and hardware environments. Thus, pre-coding and the interleaving technique are used to recover the information in burst affected blocks, if the number of encrypted blocks with errors is not high.

The computation time and memory size are increased but the performance is improved. Better performance results in lower retransmission rates, which in turn, reduces the overall delay of the system in a noisy environment, that is a low SNR. For the added overhead, the pre-coding and interleaving is smaller than that of other coding techniques.

Let us assume that in the transmitter, l_1 blocks of code (n_1, k_1) are interleaved, before they are delivered to the encryption unit in the application layer. The combination of a coding module with s_1 bit correction ability and an interleaver of l_1-block depth is capable of recovering error-free data from blocks with a single burst error of maximum length of $(l_1\text{-}1)s_1$+1 bits [119]. If the burst error consistently begins at the beginning of a block and ends at the end of another block, the maximum length of the correctable burst error increases to l_1s_1 bits.

In the receiver, after decryption, the blocks are de-interleaved and decoded by the *pre-decoder* to extract the message [119]. Figure 3.6 signifies how this technique is implemented to correct burst errors before delivering the message.

If there are any spurious blocks among the n_1 blocks after passing the decryption process, the bits of the spurious blocks randomly change. For example, if a spurious block is delivered to the decryption process, individual bits of the output block will change with a probability of ½ [11][80][119]. It can be said that each spurious decoded block acts like a burst error of length of n_1 and always starts and ends at the beginning and the end of the blocks, respectively. Therefore, if the

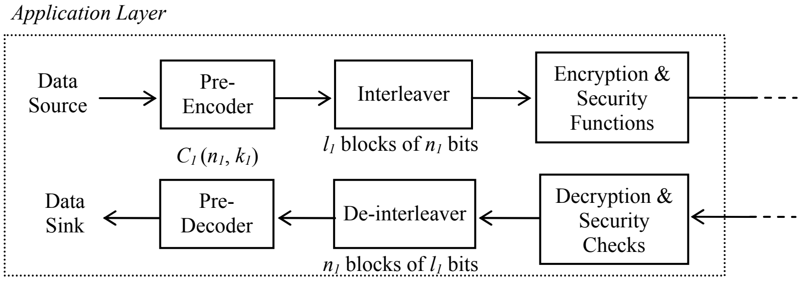

Application Layer
Data Source
Pre-Encoder
Interleaver
Encryption & Security Functions
C_1 (n_1, k_1)
l_1 blocks of n_1 bits
Data Sink
Pre-Decoder
De-interleaver
Decryption & Security Checks
n_1 blocks of l_1 bits

Figure 3.6 Application layer pre-coding and interleavering

number of spurious blocks in the input of de-interleaver is not more than $\lfloor l_1 s_1/n_1 \rfloor$ bits, where $\lfloor x \rfloor$ is the largest integer less than *x*, it is possible to recover the message without errors.

With a fixed s_1, the maximum length of the correctable burst error is determined by the ratio of the interleaver's depth and span; that is, l_1/n_1. Although the larger ratio of the depth to the span size of the interleaver increases the correction capability of the system, the decoding delay is higher. On the contrary, the implementation of the declined interleaver with same depth and span size is very easy in systems with block format data, and also has better delay characteristics. Therefore, declined interleavers with the same depth and span are commonly used in practice.

The pre-coding and interleaving technique can be adopted in HARQ systems with a bursty channel to reduce the number of retransmission by recovering burst errors. If the retransmission cost, in terms of delay and energy consumption, is higher than the cost of the extra overhead of the technique, the communication system's performance is improved by conserving energy and enhancing delay. These parameters are very important in real-time applications on battery-powered devices.

Figure 3.7 portrays how the information in a decrypted spurious block is recovered by pre-coding and interleaving technique. To improve the jitter characteristics of the system, a declined interleaver is used.

In contrast to conventional block interleavers which need to fill their memory before outputting the interleaved blocks, a declined block interleaver outputs an interleaved block as soon as a block is received. A simple interleaving method is presented in Figure 2.21. The best ratio of the de-interleaver's depth (n_1) and span (l_1) is a number close to 1 for the coding and interleaving

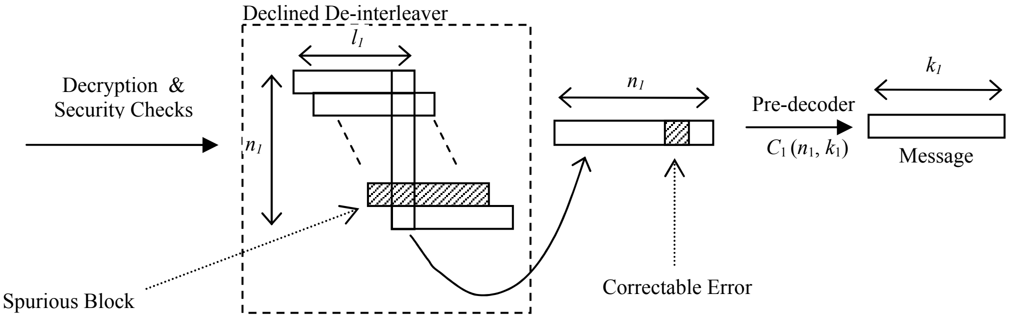


Figure 3.7 Flow of data in the receiver using declined de-interleaver and pre-decoding in application layer

technique fixed error correction capability (s_1).

If it is assumed that this technique is implemented in the applications layer, then, by using the interleaver and pre-decoding, the error occurs only if there are more than s_1 blocks of error in n_1 consecutive blocks. The probability of error improves according to the following.

$$P_e = \sum_{i=s_1+1}^{n_1} \binom{n_1}{i} P_{e_1}^{\ i} \left(1 - P_{e_1}\right)^{n_1 - i} \qquad (3.6)$$

where P_{e_1} and P_e are the probability of block errors before and after the de-interleaver and pre-decoding, respectively.

In the next section, a model that includes all of the previous techniques is introduced. In the next chapter, it is shown that although coding and interleaving increase the mean delay and overhead, the overall performance of the model is better than that of the base model with certain values of SNR.

3.2 Model

Figure 3.8 denotes the model which uses all the techniques that are previously discussed.

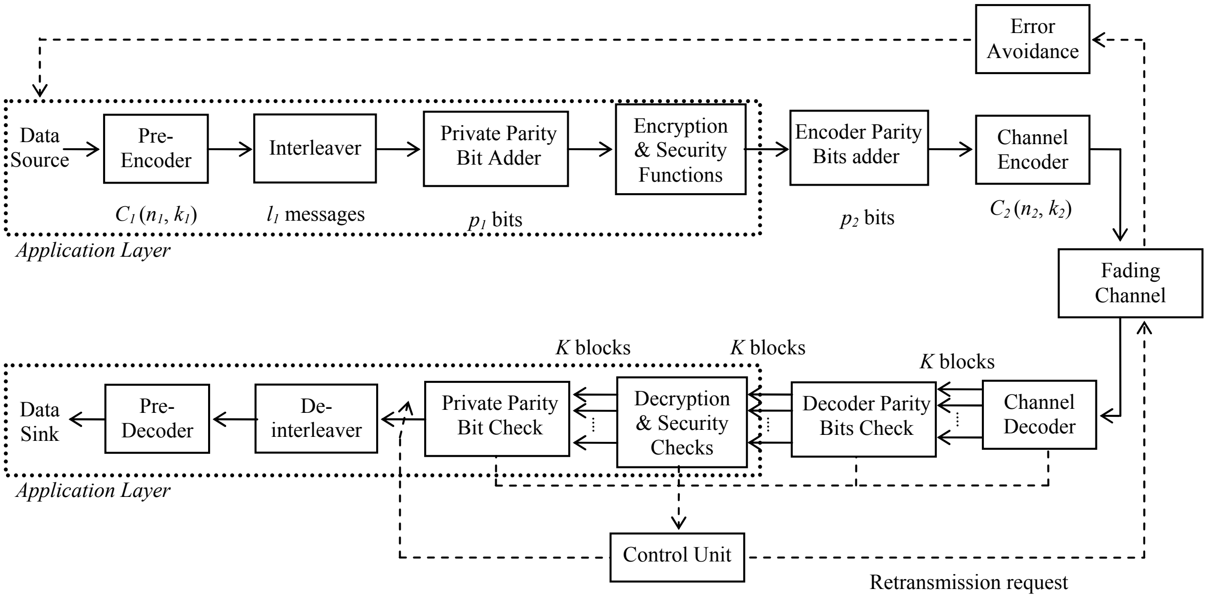


Figure 3.8 Proposed retransmission reduction model

It is assumed that the data source generates information in the form of messages of k_1 bits. Figure 3.9 conveys the flow of the data in block format at the transmitter end of the model.

As shown in Figure 3.9, each message is first encoded by the pre-coding C_1 (n_1, k_1) code that maps each message to a block of n_1 bits. Then, l_1 blocks are interleaved together to form n_1 blocks of length of l_1 bits in such way that each block in the output of the interleaver contains only one bit of each block that has been delivered to the interleaver. Consequently, the system has the capability to recover the correct messages from s_1 spurious blocks among l_1 blocks, where s_1 is the burst error correction capability of the combined pre-coding and interleaving. To improve the delay characteristics, a declined interleaver is used in the model.

After interleaving, p_1 bits of private parity are added to each block, changing the length of the blocks from l_1 to l_2 ($=l_1+ p_1$) bits. These parity bits are used to detect spurious blocks only in an authorized receiver, as discussed in Section 3.1.2. The blocks are then processed by the encryptor which outputs encrypted blocks of length k_2 bits.

The *encoder parity bits adder* appends p_2 bits of parity to the end of the block. These bits are used after the channel decoder to check the integrity of decoded blocks. Therefore, the length of the blocks

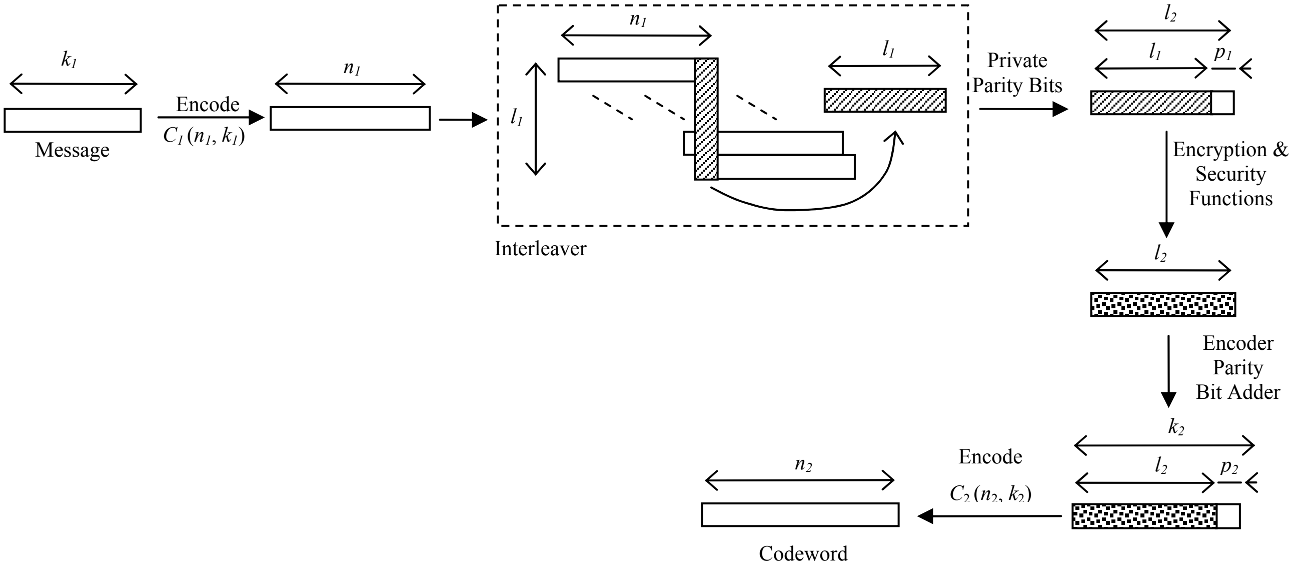


Figure 3.9 Flow of data in the proposed retransmission reduction model’s transmitter

increases to k_2 (=l_2+ p_2) bits. The channel encoder then encodes the block by using the C_2 (n_2, k_2) codes and produces codewords of length n_2 bits. These codewords are sent to the receiver through the channel.

Figure 3.10 reflects the structure of the sent packet in the model.

In the receiver, all of the processes are reversed. Figure 3.11 shows the process of decoding in the receiver. First the received codewords are delivered to the channel decoder, where the *K* best likelihood blocks that can produce the closest codeword to the received codeword, are produced.

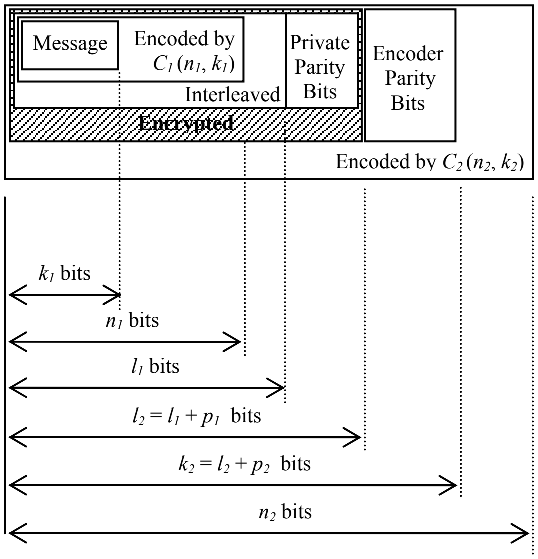
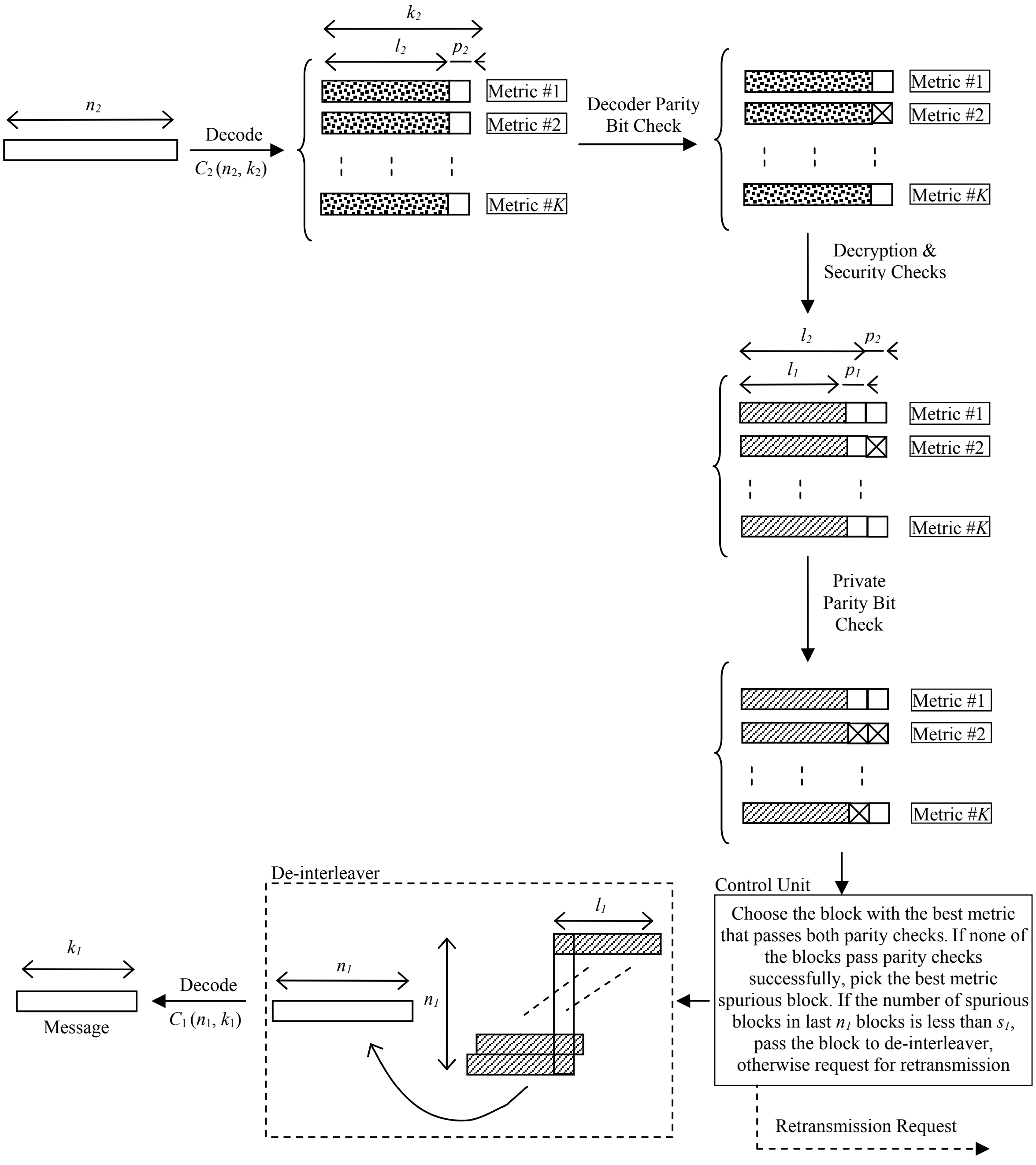


Figure 3.10 Structure of the packet in the model in Figure 3.8



Choose the block with the best metric that passes both parity checks. If none of the blocks pass parity checks successfully, pick the best metric spurious block. If the number of spurious blocks in last n_1 blocks is less than s_1, pass the block to de-interleaver, otherwise request for retransmission

Figure 3.11 Flow of data in the proposed retransmission reduction model's receiver

In addition to delivering the probable blocks, the channel decoder associates each block with a *metric*. It is a scale that shows how similar the produced codeword by each block is to the received codeword. It should be mentioned that any bits of the codeword which are affected by the noise on the channel, cause the differences. An example for the metric is the Hamming distance [67]. In this case, the metric related to each block is defined as the Hamming distance of the codeword, produced by the block and the received codeword after the hard decision process [57]. The two types of metrics are the distance metric and maximum likelihood metrics [67]. If the metric is a distance metric, the decoded block is most probably the correct block when its metric is the smallest. But if the metric is the maximum likelihood metric, the probability that the decoded block is the correct one is higher if its metric is larger. Assume that the smaller metric is assigned to the block that produces an output which is more similar to the received codeword. Here, the metric is a distance metric. Thus, the bigger the metric, the less likely the chance the block is the correct block. All the metrics are reported to the control unit to sort the blocks in a list from highly probable to less probable of being the correct answer. The control unit uses this information, along with other information, to decide if retransmission is required. The control unit is detailed later.

The *channel decoder parity check* tests all of the blocks. If it detects an error, it marks the block and notifies the control unit. Then, the receiver chops P_2 bits of the encoder parity bits from the block and passes the remaining to the decryption and security function process. At this time, the lengths of the blocks are l_2 bits. All the K blocks are processed by the *decryption and security check* unit. The spurious blocks appear as a completely random block, after passing the decryption block. The blocks, after the security check, are l_2 bits in length of which p_1 bits are the private parity bits. Random bit blocks are detected with the probability of $1-(1/2)^{p_1}$. After, the parity bits are checked by the parity check block, all of the blocks with non-matching private parity bits are marked and the control unit is notified. The most likely block of the remaining blocks is chosen by the control unit to proceed to the next level. The parity bits are removed from the selected blocks. At this point the blocks are delivered to the de-interleaver. Its output is a block of l_1 bits where the bits are interleaved from n_1 consecutive blocks, as seen in Figure 3.7. These blocks are decoded by pre-decoder block by using C_1. The result is a message in k_1 bits.

As shown in Figure 3.13, a *cross-layer control unit* supervises the performance of the system, and controls the retransmission request signal. The goal of this unit is to minimize the number of retransmissions. After the received codeword is decoded by the channel decoder, the control unit sorts the output blocks by using their metrics, from the best at the top to the worst in the bottom. After passing through the channel decoder detection process, all of the blocks that fail are marked and dropped to the bottom of the list. The list of blocks is sorted again by putting the blocks with better metrics and no detected errors at the top followed by the blocks with the detected errors. The erroneous blocks are sorted by their metrics. After, the private parity bit checking, the list is updated again in the same manner.

The block at the top of the list is selected. If it passes through both detection processes, it is delivered to the de-interleaver. But if the block fails even one of the processes, the control unit needs to decide whether to accept the block or reject it, and asks for a retransmission. This decision is dependent on the number of the blocks with detected errors that have been accepted previously. If the number of the blocks with detected errors is less than s_1 in the last l_1 blocks, the control unit sends the block to the de-interleaver, knowing that it is corrected by the pre-coding and interleaving processes. Otherwise, the control unit, in Figure 3.12, sends a negative acknowledgement to the sender and requests a retransmission.

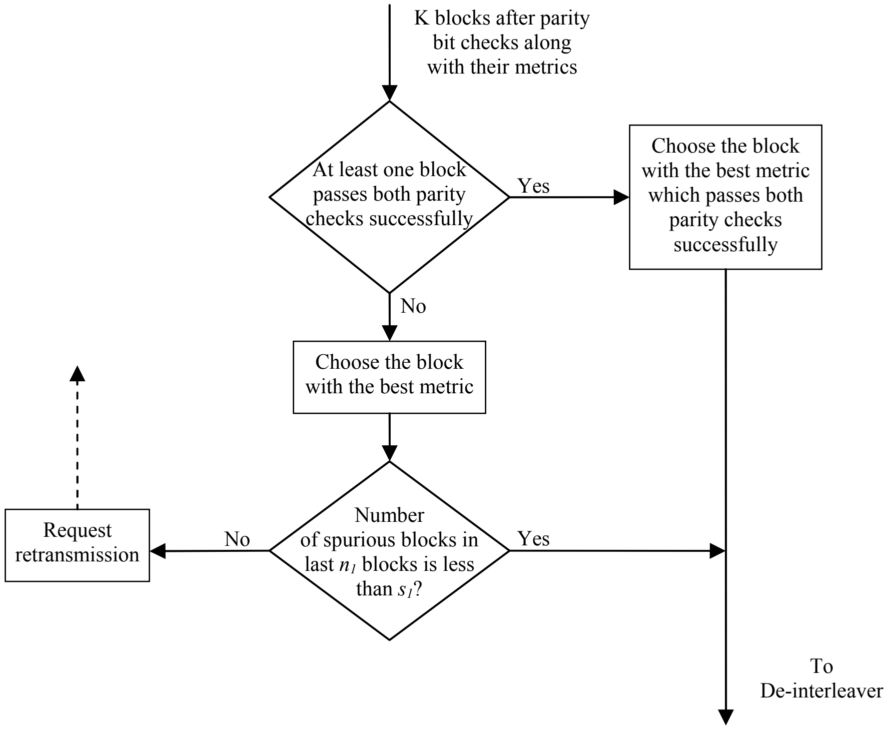

K blocks after parity bit checks along with their metrics
At least one block passes both parity checks successfully
Yes
Choose the block with the best metric which passes both parity checks successfully
No
Choose the block with the best metric
Request retransmission
No
Number of spurious blocks in last n_1 blocks is less than s_1?
Yes
To De-interleaver

Figure 3.12 Block diagram of control unit

In a secure end-to-end communication, the data encryption must be handled by the applications. This means that the information is encrypted in the application layer of the transmitter and decrypted by the receiving application. Thus, all the units, before encryption and the security function unit in the transmitter, and after decryption and the security function unit in the receiver, are implemented in the application layer. Figure 3.13 indicates the location of the implementation of the different units of the model in the WAP architecture model [100]. Since the control unit requires access to the different parts of the model implemented in different layers, the unit needs to be implemented as a cross-layer unit.

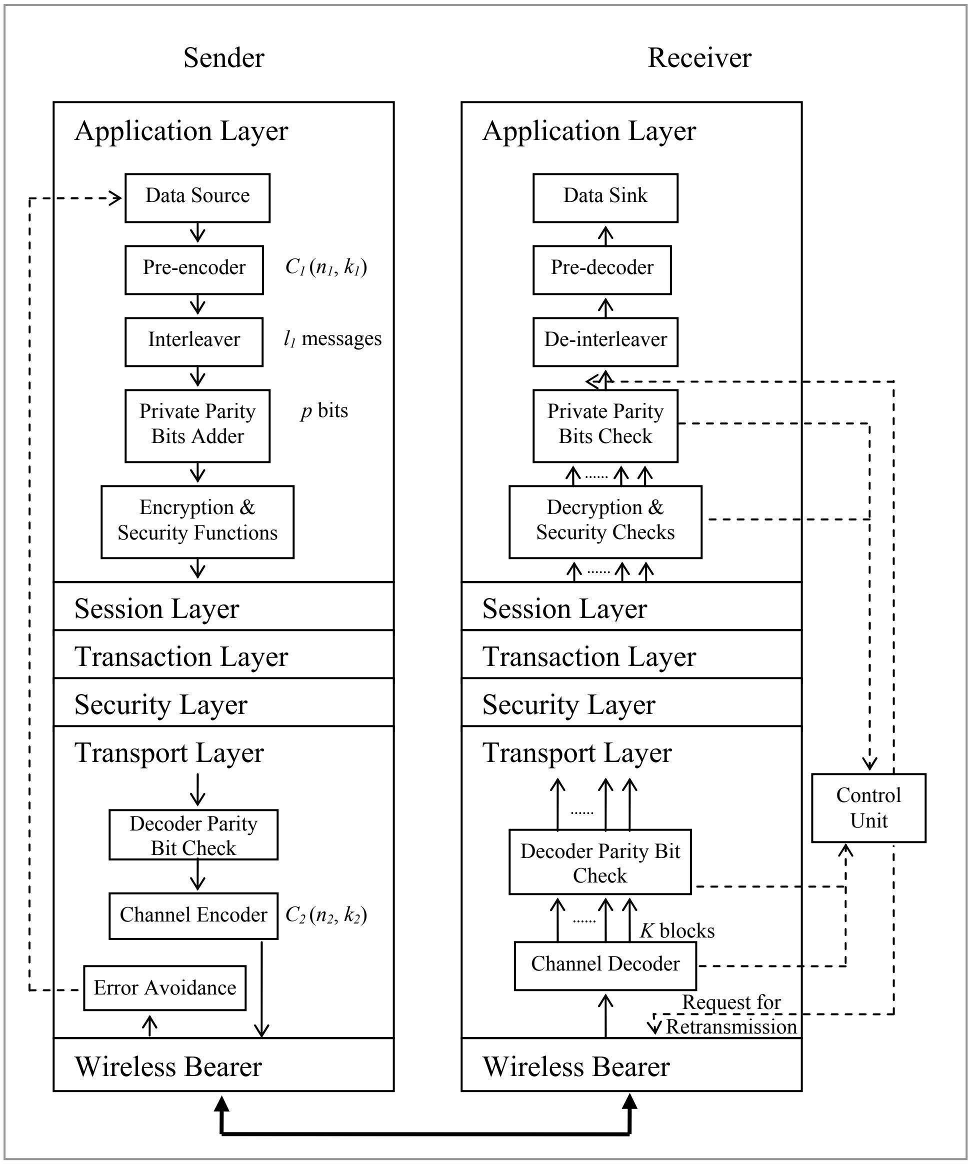


Figure 3.13 Location of different blocks of the model in Wireless Application Protocol (WAP) architecture

Figure 3.14 signifies the base model to which the results of the model are compared. The model is the system from the security functions and encryption block in the transmitter to the security checks and decryption block in the receiver. The main difference is that the channel decoder delivers only the most probable block (K=1); that is, the decoder has only one output. The interleaver and pre-coding technique is used to reduce the effect of the burst errors. Retransmission is requested if

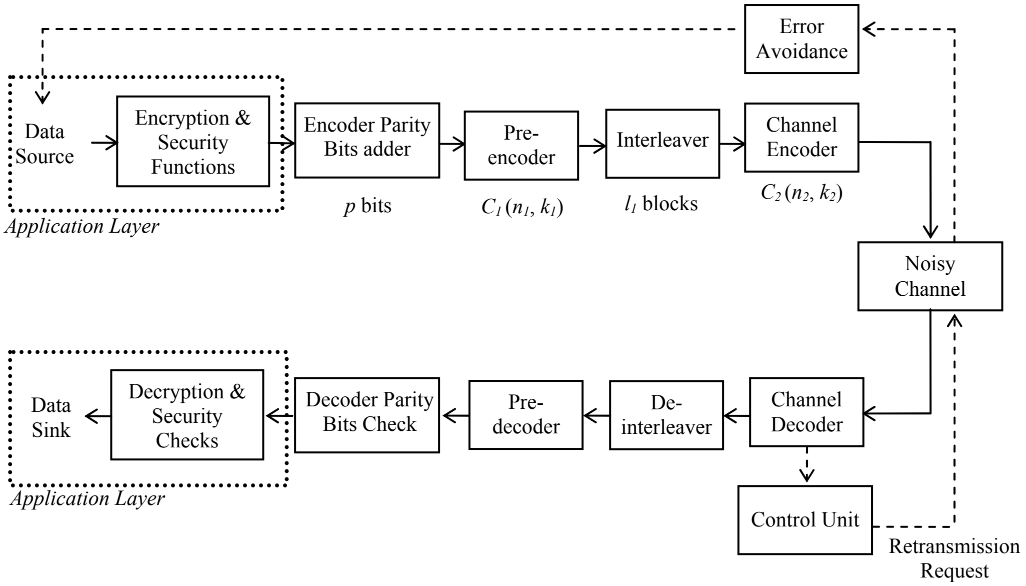


Figure 3.14 Base model

the block fails the parity bits process before delivering it to the decryption and security functions.

3.3 Analysis

Before studying different characteristics of the model the parameters that are used throughout the analyses need to be defined. Later, the assumptions and the inter-relationship of the parameters in the models are discussed.

P_i is the probability of the correct block being decoded in the i^{th} maximum likelihood position in the K best likelihood decoder. When the channel noise is very small, compared with the signal, it is expected that the channel decoder outputs the correct block in the first position. Therefore, it is expected that $P_1 >> P_j$ for high SNRs, where j ranges from 2 to K. As the SNR decreases, the probability that the correct block appears in the first position (P_1) decreases. Let P_{f+} be the probability of the false positive in the overall detection process. Also, let P_c, P_f, and P_d be the probability of correct decoding, delivering a spurious block, and not detecting the correct block among all K probable blocks in each case, respectively.

3.3.1 Block Error Rate (BER)

The channel decoder is the first module that is affected by the channel noise in the model. Due to the different forms of the received codewords, the *K*-BL channel decoder can react in three ways.

In the first case, the number of errors that occur in the receiving codeword is within the designed error correcting limit of the channel decoder. Hence, the decoder can decode the correct block as the first maximum likelihood block. This is accomplished by using the redundancy added by the channel encoder in the sender.

Assume that the probability that case 1 happens is *P*(*K*, 1), then

$$P(K,1) = P_1 \ ,$$

and the probability of the correct block delivery to the decryption and security check process is

$$P_c(K,1) = P_1 \ .$$

The probability of a spurious code delivery is

$$P_f(K,1) = 0 \ .$$

The probability of not detecting the correct block is

$$P_d(K,1) = 0 \ .$$

The second case occurs when the channel decoder decodes the correct block not as the first but as the i^{th} best likelihood block, where $1<i\leq K$. If this happens during the parity check process, two sub-cases can occur. For all *j* that $1<j<i$, either none of blocks in the j^{th} position have the correct parity bits, or at least one of them passes the parity check process successfully. The first sub-case guarantees that the correct block is selected. In this case, the proposed model can avoid the retransmissions and has a better aggregate throughput. But, if the other sub-case occurs, an incorrect block might be selected to pass to the next block. Case 2 occurs with probability

$$P(K,2) = \sum_{i=2}^{K} P_i \ ,$$

and the probabilities of the correct and spurious block delivery are:

$$P_c(K,2) = \sum_{i=2}^{K} \left(1 - P_{f+}\right)^{i-1} P_i \ ,$$

and

$$P_f(K,2) = \sum_{i=2}^{K} \left(\sum_{j=0}^{i-2} \left(1 - P_{f+}\right)^{j} \right) P_{f+} P_i \ ,$$

where i is the position of the correct block among the K decoded blocks, and P_{f+} is the probability false positive (incorrectly passing a spurious block in the detection process). The spurious block delivery occurs when a spurious block, which is positioned before the correct block, is passed through the detection process successfully. Also, the probability of not detecting the correct block at all, $P_d(K, 2)$ is

$$P_d(K,2)=0 \ .$$

In the third case, the correct block does not appear among the K best likelihood blocks. The probability case 3 happens is $P(K, 3)$, and can be calculated in terms of P_i as follows:

$$P(K,3)=1-\sum_{i=1}^{K} P_i \ .$$

Again two sub-cases can occur: either all the blocks fail to pass the parity process and the receiver requests a retransmission, or at least one of them passes which means that a spurious block is going to be selected. The probability that all the decoded blocks fail in the parity check process is

$$P_d(K,3)=\left(1-P_{f+}\right)^K P(K,3) \ .$$

The probability of spurious block delivery is

$$P_f(K,3)=\sum_{i=1}^{K}\left(\left(1-P_{f+}\right)^{i-1} P_{f+}\right)P(K,3) \ ,$$

and of course, there is no probability that correct block can be delivered, thus,

$$P_c(K,3)=0 \ .$$

It should be mentioned that these three cases cover all the possibilities, that is,

$$P(K,1)+P(K,2)+P(K,3)=1 \ .$$

Thus, the probability that the result of the parity check block is the correct block after decryption and the security check is

$$P_{c,s}(K)=P_c(K,1)+P_c(K,2) \ ,$$

and the probability of delivery of a spurious block is

$$P_{f,s}(K)=P_f(K,2)+P_f(K,3) \ .$$

The probability of not finding the correct block and mark all blocks as erroneous is

$$P_{d,s}(K) = P_d(K,3) \ .$$

The de-interleaver and pre-decoding block can recover error-free messages, if the total number of spurious blocks and blocks with detected errors are not larger than s_1 in an interval of n_1 consecutive blocks.

If 1-P_{f+} is approximated by 1, then

$$P_{c,s}(K) = \sum_{i=1}^{K} \left(1 - P_{f+}\right)^{i-1} P_i \ ,$$

$$P_{f,s}(K) \approx P_{f+}\left(K - \sum_{i=1}^{K} (K - i + 1) P_i\right) ,$$

and

$$P_{d,s}(K) \approx 1 - \sum_{i=1}^{K} P_i \ .$$

If the number of detected erroneous blocks is more than s_1, the receiver requests retransmission. Retransmission rate is calculated as follows:

$$P_d(K) = \sum_{i=s_1+1}^{n_1} \binom{n_1}{i} P_{d,\mathrm{s}}(K)^{\mathrm{i}} \left(1 - P_{d,\mathrm{s}}(K)\right)^{n_1 - i} \ .$$

To find the overall probability of error in the message, the probability of block error and block correctness, considering the retransmission, are needed. The probability of a correct block to be delivered to de-interleaver is

$$P_{c,d}(K) = P_{c,s}(K) + P_d(K)\, P_{c,s}(K) + P_d(K)^2\ P_{c,s}(K) + \ldots = P_{c,s}(K) \sum_{i=0}^{\infty} P_d(K)^i = \frac{P_{c,s}(K)}{1 - P_d(K)}.$$

With the same calculation, the probability of spurious block delivery is

$$P_{f,d}(K) = \frac{P_{f,s}(K)}{1 - P_d(K)} \ .$$

Now, the model delivers a spurious message if the total number of spurious and detected blocks in n_1 adjacent block is more than s_1, such that

$$P_e(K)=\sum_{l=0}^{s_1}\left[\binom{n_1}{l}P_{d,\mathrm{s}}(K)^l\left(1-P_{d,\mathrm{s}}(K)\right)^{n_1-l}\sum_{j=s_1+l+1}^{n_1-l}\binom{n_1-i}{j}P_{f,d}(K)^{\mathrm{j}}\left(1-P_{f,d}(K)\right)^{n_1-l-j}\right]$$

where l is the number of blocks with errors that are detected, and j is the number of spurious blocks in total n_1 consecutive blocks in the input of the de-interleaver.

By looking at $P_e(K)$, it is evident that it increases as $P_{f,\,d}(K)$ increases. $P_{f,\,d}(K)$ increases as K increases. This is why a better detection algorithm is required in order to use a channel decoder with more outputs, to compensate for the higher probability of false positive.

The probabilities of a correct delivery, spurious message, and the overall error detection change, as K increases. If the detection process is kept constant (fixed P_{f+}), the probabilities of false positive message, retransmission rate, and correct message delivery change as follows:

$$P_c(K) < P_c(K+1) \ ,$$

$$P_e(K) < P_e(K+1) \ ,$$

and

$$P_d(K) > P_d(K+1) \ .$$

If a PED is added to the system to check the integrity of blocks, the authorized receivers can test the blocks with more confidence, lower the probability of false positive. In this case, the authorized receiver can use the K-BL channel decoder without compromising reliability.

3.3.2 Throughput

Typically, the aggregate throughput is defined as follows:

$$Aggregate\ throughput = \frac{number\ of\ \text{message}\ bits}{number\ of\ transmitted\ bits} \ .$$

If it is assumed that the size of the acknowledgement packets is much smaller than that of the message, their sizes can be omitted in the calculations. In the models, the number of retransmission requests is counted for sending a block of information of n messages of k_1 bits with the *selective repeat* ARQ technique [57]. The aggregate throughput of the proposed model, then, is calculated as

$$Aggregate\ Throughput_P = \frac{k_1 n}{\left(n + ret_P\right)n_2} \ ,$$

where ret_P is the number of retransmissions for sending all the messages, and n_2 is the size of the codewords. The redundancy, which the pre-coding adds is n_1 - $k_1 = m$ bits, and the overheads of the error detection and PED techniques are p_1 and p_2 bits, respectively. Thus, each block, at the input of the channel encoder, contains (m+p_1+p_2) bits of redundancy. The channel encoder expands each bit to $1/R_2 = n_2/k_2$ bits, increasing the overall overhead in the transmitted packet to $(m + p_1 + p_2)/R_2$ bits.

For the base model, the aggregate throughput is

$$Aggregate\ throughput_B = \frac{nk_1}{(n + ret_B)n_2} \ ,$$

where ret_B is the number of retransmissions for successfully passing all the n blocks of information through the communication channel in base model, shown in Figure 3.14. The share of the overhead, added by the detection scheme in the transmitted packet, is p/R_1 . The overall overhead, considering both the pre-coder and the detection block, is $(m + p)/R_1$ bits.

When the channel residual error rate is very small, retransmission rarely occurs. In that case, it is assumed that ret_P and ret_B are negligible, compared with n, and the ratio of the base model throughput to the throughput of the proposed model is calculated as follows:

$$\frac{Aggregate\ throughput_B}{Aggregate\ throughput_P} = \frac{k_1 + m + p_1 + p_2}{k_1 + m + p} \ .$$

For the value of p_1+p_2 equal to p, the ratio is 1. This means in high SNRs (SNR>10^{dB}), the proposed model and base model have the same throughput efficiency.

To compare throughputs, they are normalized by eliminating the constant terms. Therefore, the throughput of the base model and the proposed model are

$$Normalized\ throughput_B = Aggregate\ throughput_B \frac{n_2}{k_1(p + m)} = \frac{n}{n + ret_B} \ ,$$

and

$$Normalized\ throughput_P = Aggregate\ throughput_P \frac{n_2}{k_1(p_1 + p_2 + m)} = \frac{n}{n + ret_P} \ .$$

Then, the normalized aggregate throughput is applied to compare the throughput of the proposed model with that of the basic model.

3.3.3 Delay and Jitter

The different actions during the error-free transmission of a block require different amounts of time. In the base model, if an uncorrectable error in the decoded block is detected, the receiver requests the sender to retransmit the block again. This retransmission takes time, and increases the delay mean and variance of the system. This is especially important if the round-trip time is long, or the energy consumption of the transmission is more costly than that of the processing.

In the newly devised models a selective-repeat ARQ is used by which on receiving a negative acknowledgement, the sender backs up to the codeword that is negatively acknowledged and resends it. As defined in Section 3.3.1, the probability of detecting errors in the block is $P_d(K)$. For a codeword to be successfully accepted by the receiver, the average number of transmissions, including the original transmission, required is

$$N_{average} = 1\left(1-P_d(K)\right) + 2\,P_d(K)\left(1-P_d(K)\right) + 3\,P_d(K)^2\left(1-P_d(K)\right) + \cdots \ ,$$

$$N_{average} = \left(1-P_d(K)\right)\sum_{i=0}^{\infty} i\,P_d(K)^{i-1} \quad ,$$

and [Appendix B]

$$N_{average} = \frac{1}{1-P_d(K)} \ .$$

Another important factor is the variance of the delay, called jitter. Its value in the novel models is

$$N_{Variance} = \frac{P_d(K)}{\left(1-P_d(K)\right)^2} \ .$$

By reducing the number of retransmissions, the delay mean and variance characteristics of the system are improved, but this reduction comes with the cost. It adds extra processing, which in turn, increases the overall processing time. It is proven that this trade improves the overall delay characteristics of the communication system. T_1 is the interval from the time that the sender transmits a codeword till the time that the receiver delivers the message to the de-interleaver, when retransmission does not occur. Also, let T_2 be the interval from the moment of the receiver request for retransmission till the time the sender has the codeword ready to be resent. T_1 includes all the time that the codeword needs to pass through the channel till the time the receiver requires to receive, decode, check the parity bits, and decrypt the blocks. The average delay time for a codeword to be accepted by the receiver in terms of T_1 and T_2 is [Appendix B]

$$Delay(K) = (1 - P_d(K))T_1 + P_d(K)(T_2 + T_1 + P_d(K)(T_2 + T_1 + P_d(K)(T_2 + T_1 + \cdots))) \ ,$$

$$Delay(K) = \frac{1}{1 - P_d(K)} T_1 + \frac{P_d(K)}{1 - P_d(K)} T_2 \ ,$$

$$Delay(K) = T_1 + \frac{P_d(K)}{1 - P_d(K)} (T_1 + T_2) \ ,$$

and

$$Delay(K) = T_1 + N_{average}(T_1 + T_2) \ .$$

The delay variance is

$$Jitter(K) = \sum_{i=0}^{\infty} ((i+1)T_1 + i\,T_2)^2 \, (P_d(K))^i (1 - P_d(K)) - (Delay(K))^2 \ ,$$

$$Jitter(K) = \frac{P_d(K)}{(1 - P_d(K))^2} (T_1 + T_2)^2 \ ,$$

and

$$Jitter(K) = N_{Variance}(T_1 + T_2)^2 \ .$$

These equations indicate that by decreasing $P_d(K)$, the delay average and jitter decrease. Also by increasing *K*, $P_d(K)$ shrinks. If the time required to process more blocks is considerably smaller than the retransmission time, $T_1 << T_2$, it is possible to achieve a lower delay average and jitter by increasing *K*.

Figure 3.15 shows the jitter and delay mean, when P_d is changing, that is, the delay characteristics of the system become worse, when P_d is growing. It can be seen that the system with a higher value of T_2 is more sensitive to P_d. In the proposed model, P_d is kept as small as possible to achieve better characteristics.

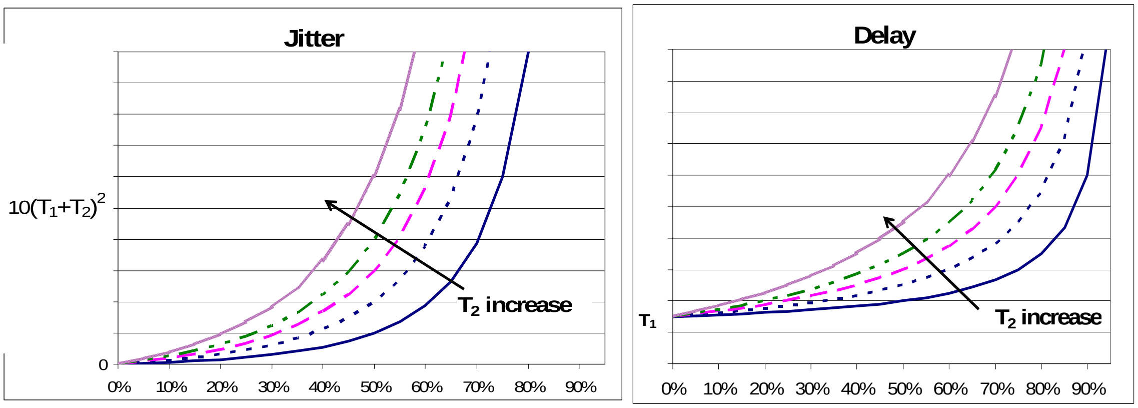


Figure 3.15 Increase in the delay mean and jitter when P_d increases

3.4 Summary

In this chapter, different methods are introduced to decrease the number of the retransmissions in an end-to-end secure wireless communication through a channel under noisy conditions. Also, a model which uses all the techniques is proposed.

In the proposed model, the channel noise is exploited to increase the attacker's difficulty in recovering the transmitted information. This is accomplished by hiding the parity bits and the acknowledgement in the communication. Although the original message can be left unencrypted, the reliability of the decoded information, available to the unauthorized receiver, is lower. In the networks for which processing time is smaller than the time required for the retransmission, it is possible to achieve lower number of transmission by using the novel model. The proposed model is best used in wireless environments.

In the next chapter, the simulation results and a comparison of the aggregate throughput and the delay characteristics of the proposed and base models are demonstrated.

Chapter 4
Energy Conservation and Security Enhancement Techniques in Wireless Connections

In this chapter, novel models, comprised of different techniques to improve the energy consumption, and enhance security over a certain range of the channel signal to noise ratio are established by reducing the number of retransmissions or avoiding unnecessary iterations. In addition, a cross-layer controller for monitoring the performance of the different parts of the model in the different layers is explained.

In this chapter, the implementation of the *K*-BL channel decoders is exhibited. We pursue the idea into more details and complete the implementation of proposed models. Further, simulation results of the proposed models are shown and compared with those of the base models.

For the realization of the *K*-BL channel decoder, two well-known channel decoders are chosen: the Viterbi decoder, and the Turbo decoder. In the next section, realization of the *K*-BL decoder by using the concept of the Viterbi decoder is described. Several modifications of the algorithm are applied to output the *K* best likelihood blocks. In the second section, the implementation of the *K*-output channel decoder is demonstrated by using the principle of Turbo Codes. The detailed simulation parameters of models, along with the results, are presented in following sub-sections.

4.1 Modified Viterbi Algorithm

Originally, the Viterbi decoder outputs the maximum likelihood block. To find it, the decoder associates each block with a metric which exhibits the similarity of the codeword generated by the decoded block and the received codeword [67]. Therefore, the better the metric, the more similar the decoded block is to the original block. Depending on the metric type, better metric means differently. If the metric is the measurement of the Hamming distance, the smaller its value the better the metric. For the maximum likelihood metrics, the larger the value, the better the metrics.

In the Viterbi algorithm the decoder retains the path of the best metric blocks in each state to find the maximum likelihood block [67]. Figure 2.13 reflects an example of the trellis in the Viterbi decoding algorithm. The bold arrows indicate the selected paths emerging to each state at the current

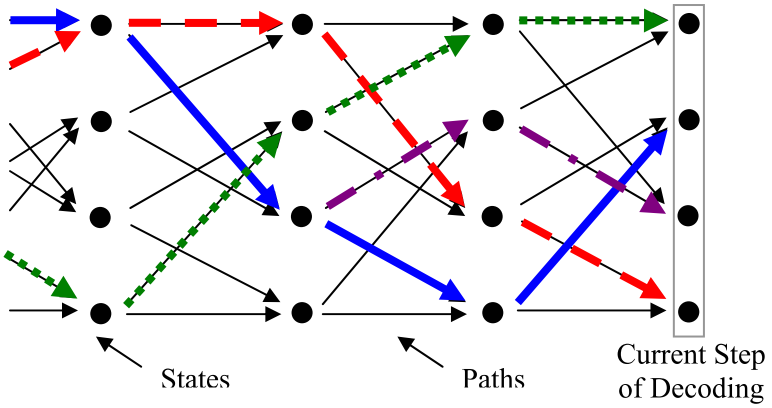


Figure 4.1 Different paths in the trellis used in the Viterbi algorithm

step of decoding.

To realize the *K*-BL channel decoder in the newly developed model, a modified version of the Viterbi decoding algorithm is adopted [119]. Like the original algorithm in which the best metric paths and sequences are kept in each state, the modified algorithm holds the paths of the *K* best metric sequences.

At the end of the decoding, the original algorithm determines the most likelihood block by looking at the blocks' metrics. The best-metric block is then outputted from the channel decoder. The modified Viterbi algorithm chooses its block similarly. The algorithm determines *K*-BL blocks by choosing *K* blocks with the best metric values and outputs them, as represented in Figure 4.2.

In some implementations, the block of data in the transmitter is forced to end by zeros so that the path of the correct codeword ends in a specific state (all-zero state). Knowing this the receiver has extra information for codeword decoding [67]. Also, this method is similarly used in the modified Viterbi algorithm. Such a method, in the receiver can narrow its selection by the assumption that the blocks path must end at a specific state at the end of the decoding process.

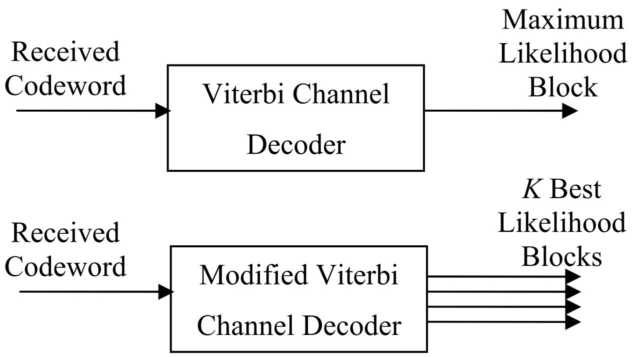
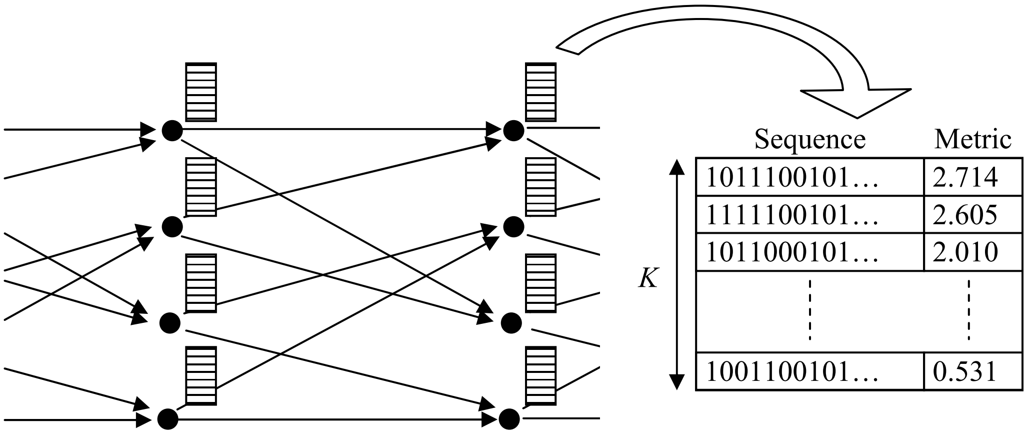


Figure 4.2 Channel decoder using the original Viterbi algorithm and the modified one which outputs *K*-BL blocks



Sequence	Metric
1011100101…	2.714
1111100101…	2.605
1011000101…	2.010
⋮	⋮
1001100101…	0.531

Figure 4.3 *K*-BL channel decoder using Viterbi algorithm

Figure 4.3 shows the amount of data that is stored at each step of the modified Viterbi decoding algorithm. Each row of the table stores a decoded sequence of bits, corresponding to a unique path. Each path links the beginning state to the current state on the trellis [67]. These sequences are accompanied by their metrics which indicate the similarity of the codeword generated by the sequence and the received codeword up to the current stage of decoding. The better the metric, the more similar the generated codeword is to the received codeword and the higher the probability that the original message looks like the current sequence.

Obviously, the modified Viterbi channel decoder needs more memory and processing capability, increasing the complexity of the receiver. The memory, the computation power, and the decoding time are measured and compared with those of in the base model.

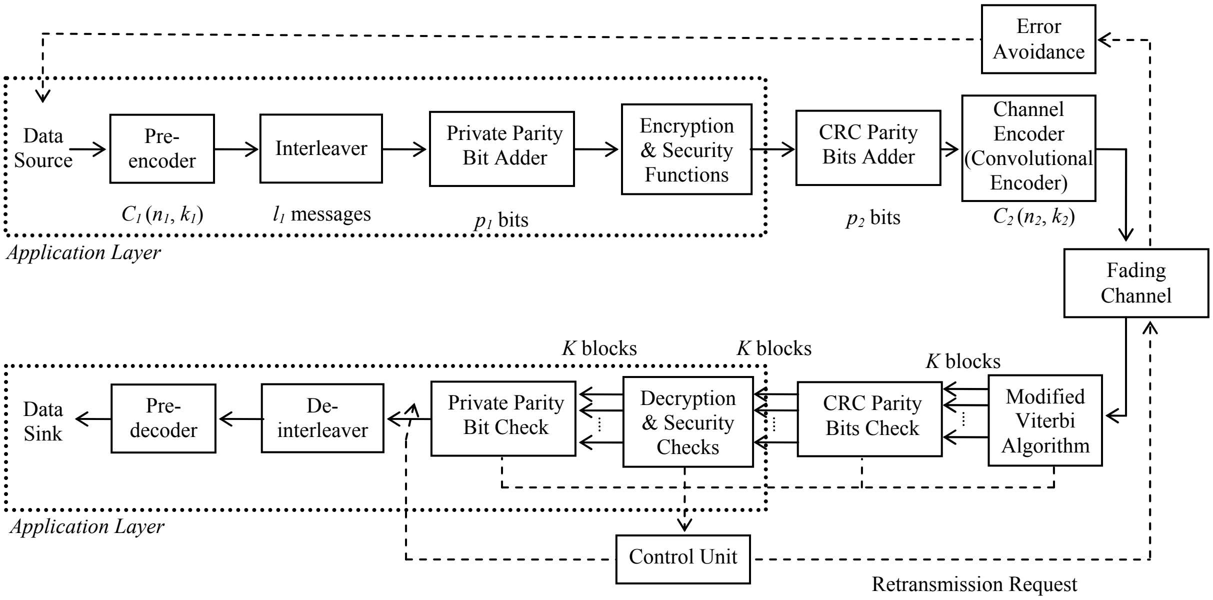


Figure 4.4 *K*-output channel decoder using the modified Viterbi algorithm

To study the performance of the reduced retransmission technique, the modified Viterbi algorithm is investigated in the channel decoder of the model in Figure 3.8. Figure 4.4 offers the schematic of the complete proposed model by using the modified Viterbi algorithm in the channel decoder.

As shown in Figure 4.4, to detect the spurious blocks, a *Cyclic Redundancy Code* (CRC) [20] detection algorithm is used. All the receivers can use the CRC error detection. In the authorized receivers, the PED technique is used to detect spurious blocks, along with CRC error detection process. Table 4.1 lists all the irreducible generator polynomials which are used in the CRC with different parity bits.

Table 4.1 CRC parameters used in simulation

Number of parity bits (p_2)	CRC parity generator function
1	$x+1$
2	x^2+x+1
3	x^3+x+1
5	x^5+x^2+1
6	x^6+x+1
7	x^7+x+1
9	x^9+x+1
10	$x^{10}+x^3+1$
12	$x^{12}+x^3+1$

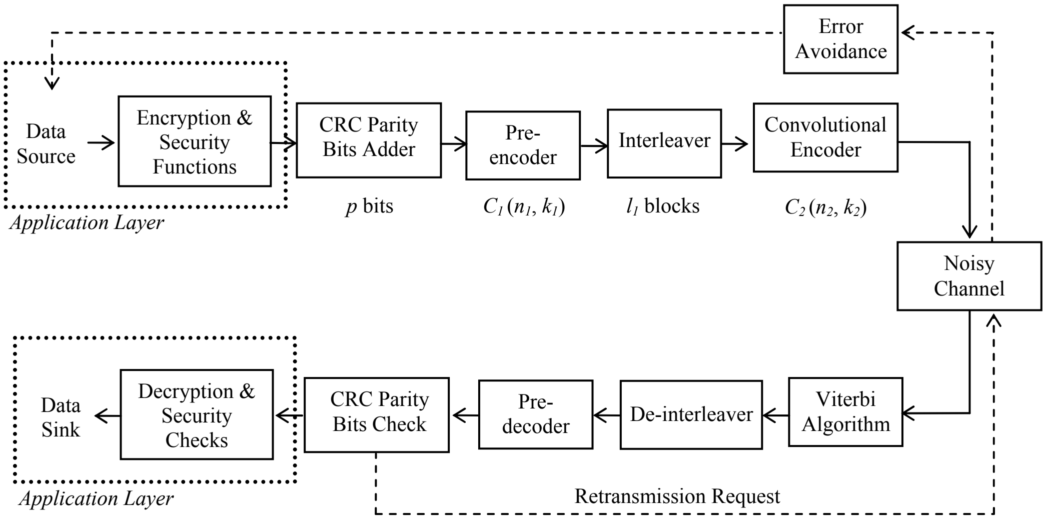


Figure 4.5 Base model using the original Viterbi algorithm

The base model, in Figure 4.5, consists of the original Viterbi algorithm in the channel decoder.

In the following sections, the models are simulated and their results are compared.

4.1.1 Simulation

In the simulations, a *shortened Hamming code*, C_1 (2^m - 1 - p_1, 2^m - m - 1 - p_1, 3), is used in the pre-coding, where m is the degree of the code generator polynomial, and p_1 is the number of the shortened bits. The shortening is conducted for the purpose of having a preset number of bits, which is 128, for encryption. The Hamming distance of this code is 3 and the code can correct only one bit error, that is $s_1 = 1$.

The interleaver is a declined interleaver with the same span and depth, that is $n_1 = l_1$. The depth (and span) of the interleaver is chosen according to the encryption function input size. For the encryption process the *Advance Encryption Standard* (AES) or *Rijndael* algorithm [82], with a block and key size of 128 bits, are chosen. Prior to encryption, p_1 bits of private parity are added to the output block of the interleaver, thus, the depth of the interleaver is set to 128 - p_1 which implies that $n_1 = l_1 = 128 - p_1$. This, in turn, determines the degree of the shortened Hamming code. Selecting $2^m - 1 - p_1 = 128 - p_1$, yields $m = 7$.

The CRC code appends p_2 parity bits to the encrypted block which increases the size of the block to $128+p_2$ bits, which are passed to a channel encoder.

For the channel encoder, a convolutional code $C_2\,(n_2, k_2, 5)$, with the generator polynomial $G(D)=[1+D^2 \;\; 1+D+D^2]$ is selected. This code has a rate of $R_2 = ½$ and its free distance is 5 [67]. The encoder of this code is exhibited in Figure 4.6. To return the state of the code to an all-zero state, two zero bits are appended to the input block, hence $n_2=2(k_2+2)$. The structure of the transmitted packet in the simulation is shown in Figure 4.7.

p_1 and p_2 are the variable parameters in the study, and the results are reported for different values.

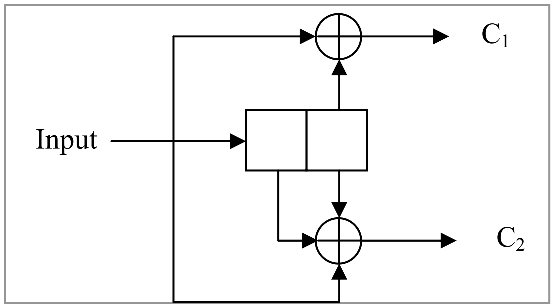


Figure 4.6 Convolutional encoder with a generator polynomial of $G(D) = [1+D^2 \;\; 1+D+D^2]$

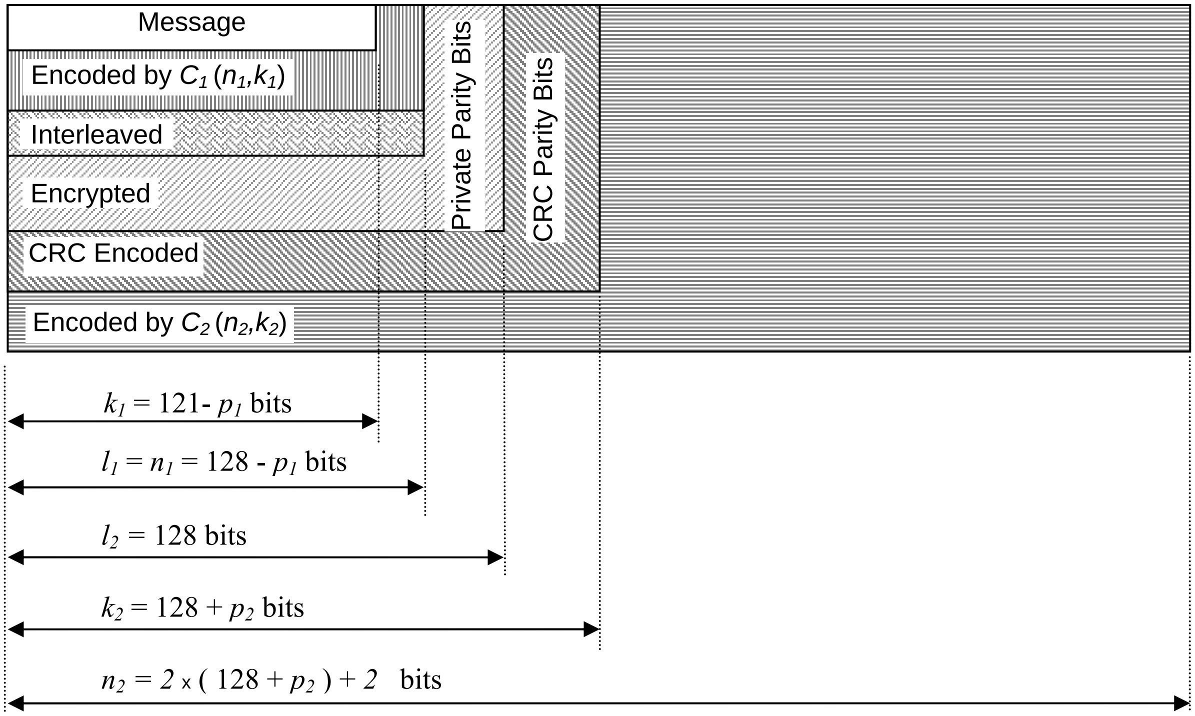


Figure 4.7 Structure of the simulated transmitted packet

The simulated channel is a Rayleigh fading channel. This model is used, since the wireless channels are negatively affected by different effects such as shadowing, fading, and multi-path [58]. In the model, the output of the channel is $y=a.x+n$, where a and n are random variables with Rayleigh and normal distributions, respectively, and x is the input of the channel.

Generally, there are two different channel decoders: *hard-decision* and *soft-decision*. In hard-decision decoding, the hard-decision of the received signal is done prior to the decoding. Thus, the hard-detection operation is considered a part of the communication channel. Then, the combination of the waveform channel and demodulator can be represented as a Binary Symmetric Channel (BSC) model, whose results are shown.

The channel decoder in the receiver uses the modified Viterbi algorithm to find the *K*-BL blocks [119], increasing the complexity of the decoder. Computation power, the required decoding time, and the memory size are measured as *K* is increased. For the simulation results, the range of *K* is from 1 to 4.

Then, the *K* suggested blocks are checked by a CRC parity check. If error bits are detected in the blocks, they are marked by the control unit. The receiver then starts decrypting the blocks with no errors and in the order of the best metric value. As soon as the blocks are decrypted their private parity bits are checked. If any block fails the private parity bit check, the block is marked and the control unit is notified.

The control unit monitors the number of erroneous detected blocks (the marked blocks). Consistently, it retains the number of spurious blocks, which are delivered to pre-coding and interleaving, smaller or at most equal to the correction ability of it, s_1, in l_1 consecutive blocks. Therefore, the probability that the correct bits are recovered by the pre-coding and interleaving technique is maximized.

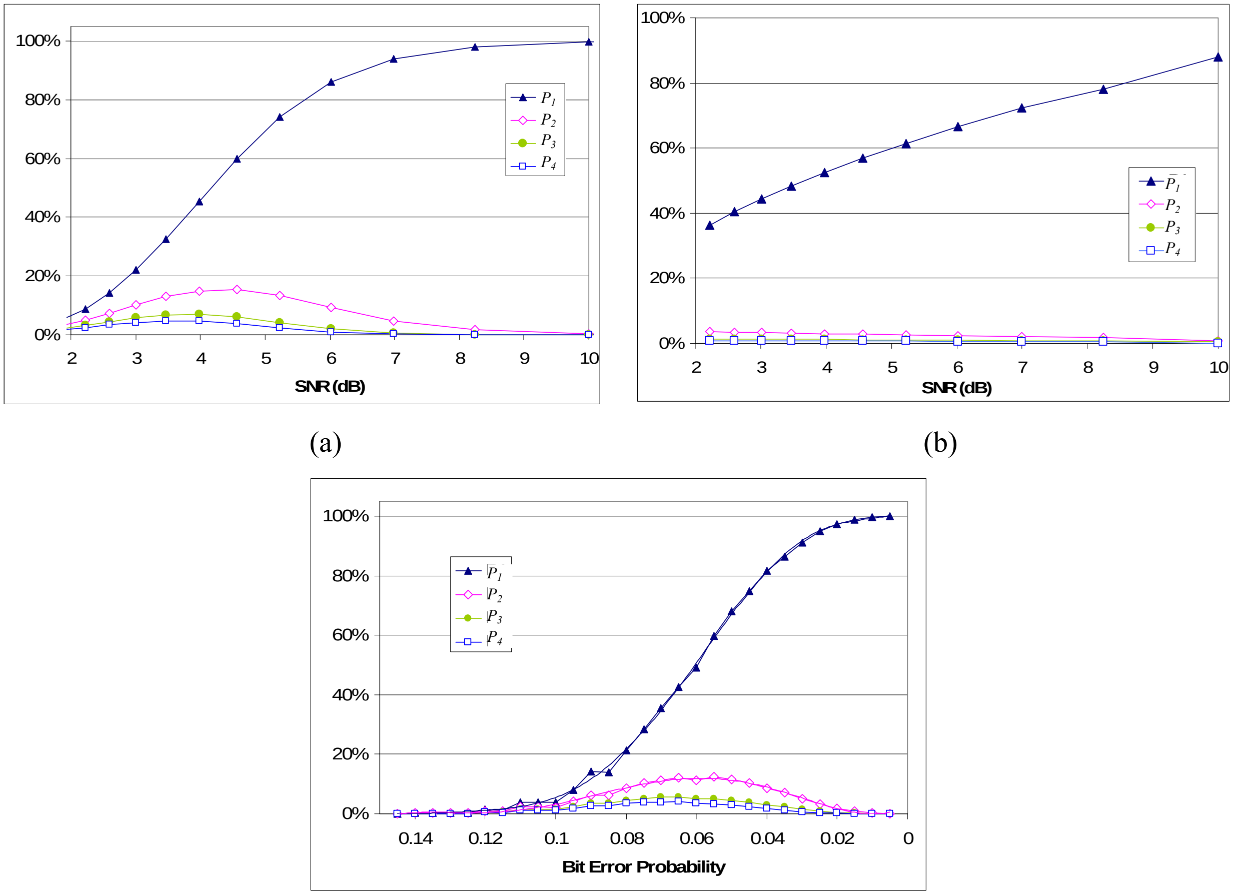
If the control unit detects more than s_1 erroneous block in l_1 adjacent blocks, the unit sends a retransmission request to the receiver. The flow of data in the receiver is shown in Figure 3.11.

Figure 4.8 shows P_i for the *K*-BL channel decoder by using the *K*-output Viterbi algorithm for $K = 4$ (for the definition of P_i refer to Section 3.3). Figure 4.8(a), (b), and (c) illustrate P_i for fast Rayleigh fading, slow block Rayleigh fading[1], and memory-less BSC channel models, respectively.

Since the P_i values for $i > 1$ are never comparable with P_1 in a slow block fading Rayleigh channels, the added computation cost in the proposed model does not provide additional reliability. Thus, the proposed model is not beneficial in channels with slow block fading characteristics.

In fast fading channels, on the other hand, the P_i's are comparable in low SNRs, as observed in Figure 4.8(a). But, P_1 is much larger than P_i for $i > 1$, when SNR is more than 7^{dB}. Since for $i > 1$, the model uses the sum of P_is to improve the performance, it is concluded that the improvement, obtained by this model and compared to the base model, is trivial for high SNRs.

[1] This model assumes that the fading gain process is piecewise constant on blocks of n_2 symbols.



(a)

(b)

(c)

Figure 4.8 P_i for $i = 1$ to 4 for the modified Viterbi channel decoder where the communication channel is (a) a fast Rayleigh fading channel, (b) a slow block Rayleigh fading channel, and (c) a memory-less BSC channel

Figure 4.8 signifies that when the channel becomes noisy, P_1 (the probability that the correct block is decoded as the maximum likelihood block) becomes comparable for P_i's , where i>1. The base model, in where the original Viterbi algorithm is deployed, can deliver the correct block only when P(1, *K*) happens. (For the definition of P(*i*, *K*) refer to Section 3.3.)

The concept behind the proposed model is to capture the correct block, even if it is not the maximum likelihood decoded block. The advantage of the novel model appears when the correct block is not decoded as the highest probable block, but as the i^{th} probable block where $2 < i < K$. This is represented by case 2 in the analysis in Chapter 3 (P(2, *K*)). Here, the base model is required to send a retransmission request or to deliver an incorrect block, whereas the proposed model can deliver the correct block, if it can eliminate the spurious blocks.

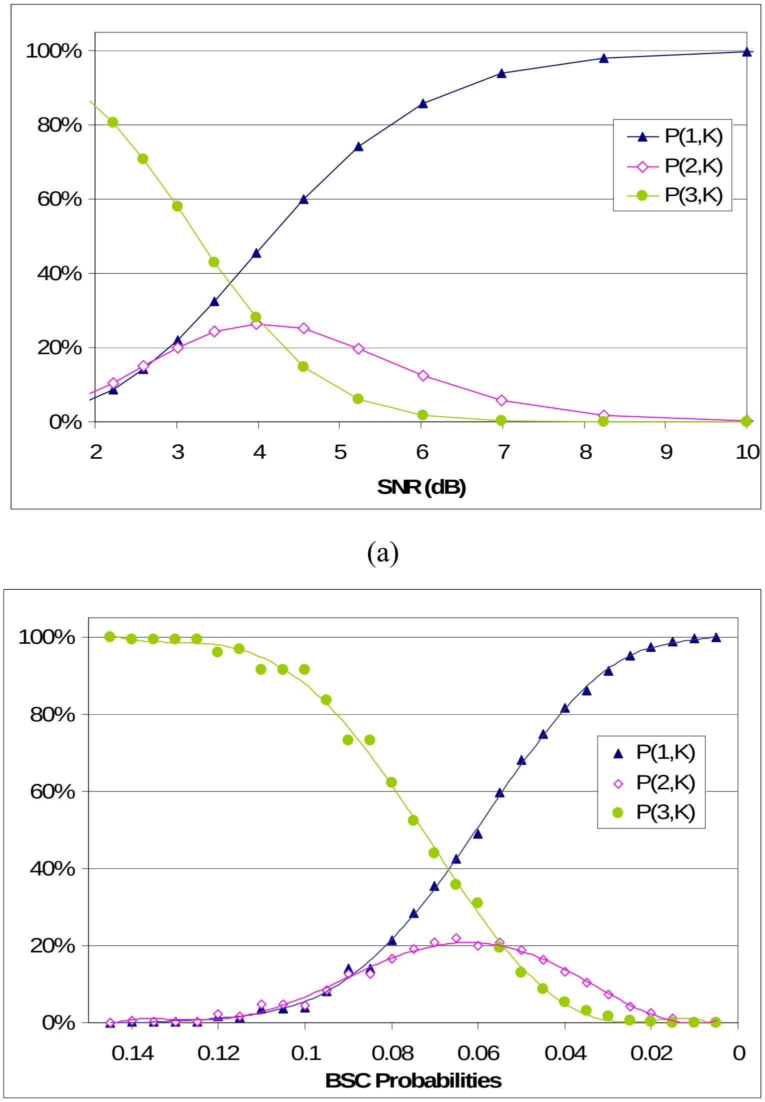
To recognize the correct block in the *K* outputs of the modified Viterbi channel decoder, the receiver uses error detection techniques. If it is assumed that the error detection algorithm can detect spurious blocks with 100% accuracy, P(1, *K*) represents the base model probability of delivering the correct block. P(1, *K*) + P(2, *K*) is, then, the probability of a correct block delivery in the proposed model. Thus, P(2, *K*) is interpreted as the advantage of the proposed model over the base model.

Figure 4.9(a) shows P(1, 4), P(2, 4), P(3, 4) for a modified Viterbi channel decoder with 4 outputs, when the channel is modelled as a fast Rayliegh fading channel. The number of private and CRC parity bits are set to $p_1 = 5$ and $p_2 = 5$, respectively. Figure 4.9(b) shows the same results with the same parameters for a Binary Symmetric Channel.

The behaviour of the P(*i*, *K*) probabilities in the models for a soft-decision and a hard-decision are the same. By studying one of them, the conclusions can be extended to include the other one. Since the performance of the soft-decision channel decoder is superior to the performance of the hard-decision channel decoder, we use soft-decision channel decoder to study the performance of the proposed model.

Now, by considering the soft-decision in the proposed and base models, it can be said that in a very poor channel condition, that is SNR < 3^{dB}, the probability of error is too high, and both models fail to decode the correct block. When the channel condition is good, that is SNR > 7^{dB}, the added complexity of the proposed model cannot improve the performance of the system. Under these conditions, the performance of the base model is as good as that of the proposed model. Therefore, the proposed model improves only the performance in the region, where the channel condition changes from very bad to good, that is 3^{dB} < SNR < 7^{dB}. In this region, the value of P(2, *K*) is comparable to the value of P(1, *K*) proving that the proposed model has the better performance.

The focus is on the channel condition, where the SNR ranges from 3^{dB} to 7^{dB}. In the next few sections, the improvements on the throughput and delay characteristics that are achieved by the proposed model are described.



(a)

(b)

Figure 4.9 P(1, *K*), P(2, *K*) and P(3, *K*) for the proposed model where *K*=4 and p_1=p_2=5: (a) soft-decision and (b) hard-decision channel decoder

4.1.2 Throughput

Figure 4.10 illustrates the normalized aggregate throughput of the proposed model compared with the normalized aggregate throughput of the base model. It is evident that the novel model outperforms

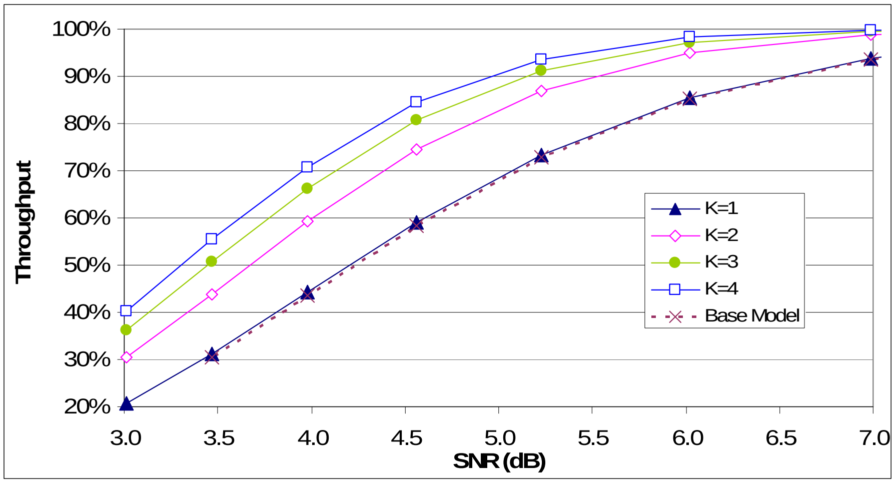


Figure 4.10 Aggregate throughput of the proposed model with $K = 1$ to 4, and $p_1 = p_2 = 5$, compared with the base model with $p = 10$

the base model in the region that the base model uses retransmissions with a high probability. It occurs when the channel decoder in the base model has an unacceptable probability of failure.

If the PED is added to the system in order to check the integrity of the blocks, the authorized receiver can test the blocks with more confidence than the unauthorized receivers. This is due to the lower probability of a false positive of the overall error detection in the authorized receivers.

When the channel residual error rate is very small (SNR > 8^{dB}), the throughput of the base model is the same as the throughput of the new model. When the channel decoder fails to decode the correct block, the correction ability of the pre-coding comes into play and reduces the number of retransmission requests.

4.1.3 Delay and Jitter

The various actions during the error-free transmission of a block require different amounts of time. In a HARQ system, if an uncorrectable error in the decoded block is detected, the receiver will request the sender to retransmit the block. This retransmission takes time and increases the delay mean and variance in the system. In the communication system where the retransmission consumes costly resources, such as the energy in battery powered mobile devices, and the waiting time in a long round-trip connection, the proposed model can improve the performance of the system. By reducing the number of retransmissions in the proposed model the delay mean and variance characteristics of

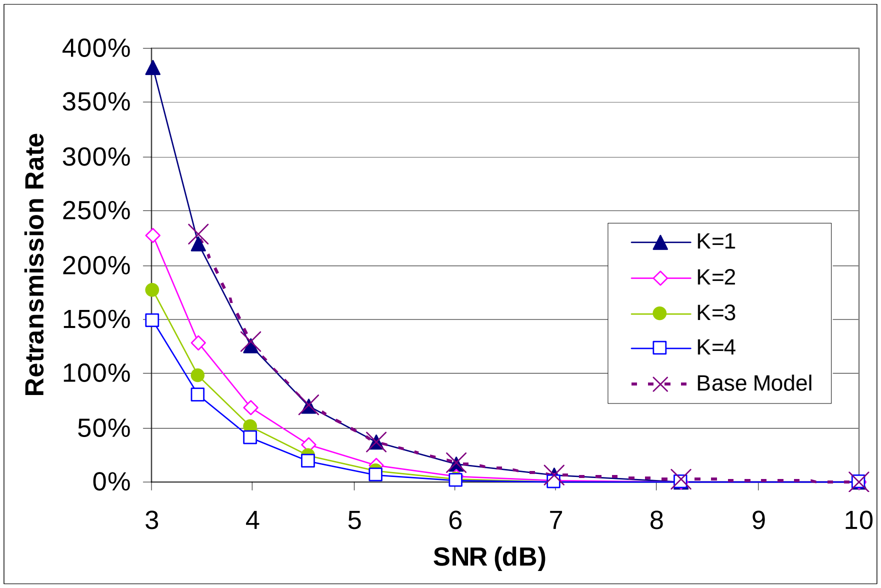


Figure 4.11 Retransmission rate of the proposed model for $K = 1 \ldots 4$ for $p_1 = p_2 = 5$, compared with the base model with $p = 10$

the system improve, but with a cost. More process(es) are added which in turn, increases the overall processing time. Therefore, the newly devised model has an advantage over the base model, when the processing capability is not limited.

Delay average and jitter are directly affected by the rate of retransmission (Section 3.3.3) as follows:

$$Delay(K) = T_1 + \frac{P_d(K)}{1 - P_d(K)}(T_1 + T_2) \ ,$$

and

$$Jitter(K) = \frac{P_d(K)}{(1 - P_d(K))^2}(T_1 + T_2)^2 \ .$$

Overall, a channel decoder with K maximum likelihood outputs, parity bits, and pre-coding blocks leads to a lower number of retransmissions, and therefore a lower mean delay and delay variance. In some ranges of residual errors in the channel, improved delay characteristics are possible. Figure 4.11 indicates that the proposed models have better retransmission rate $P_d(K)$, and thus, a better mean delay and jitter properties.

4.1.4 Block Error Rate

The Block Error Rate (BER) for the different parameters is plotted in Figure 4.13. The analytical results are shown in dotted lines for each value of *K*. The bound is not tight when the SNR is high. This is due to a higher probability of detection by the CRC, after the trellis decoder is employed in high SNRs. By the same detection algorithm, the probability of a failure rises as *K* increases.

As shown in the figures, the block error probability increases when *K* is increased. This is due to the fact that the number of error detection checkings per receiving codeword increases proportionally to *K* (Section 3.3.1). However, for the error detection algorithm with a fixed false positive probability, the higher the *K* becomes, the higher the rate of false positive. In order to compensate for the increase in the block error rate, the number of parity bits is increased. Figure 4.12 shows how the increase in parity bits can compensate for the effect of *K* on the block error rate.

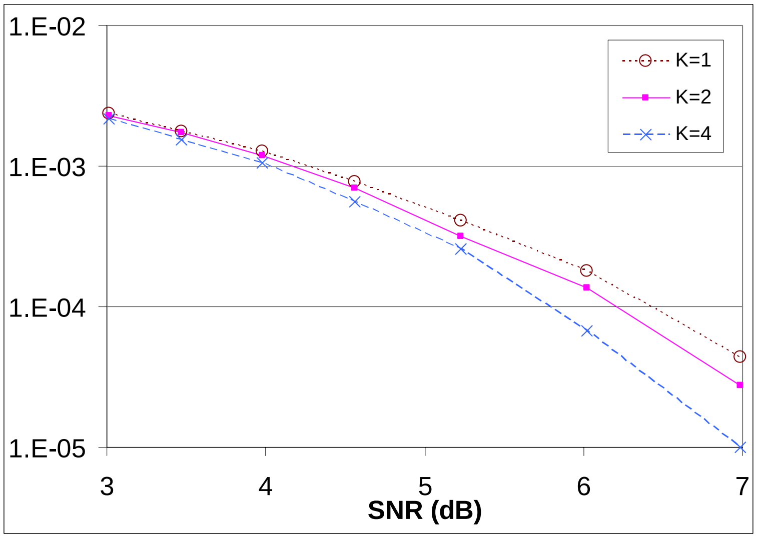
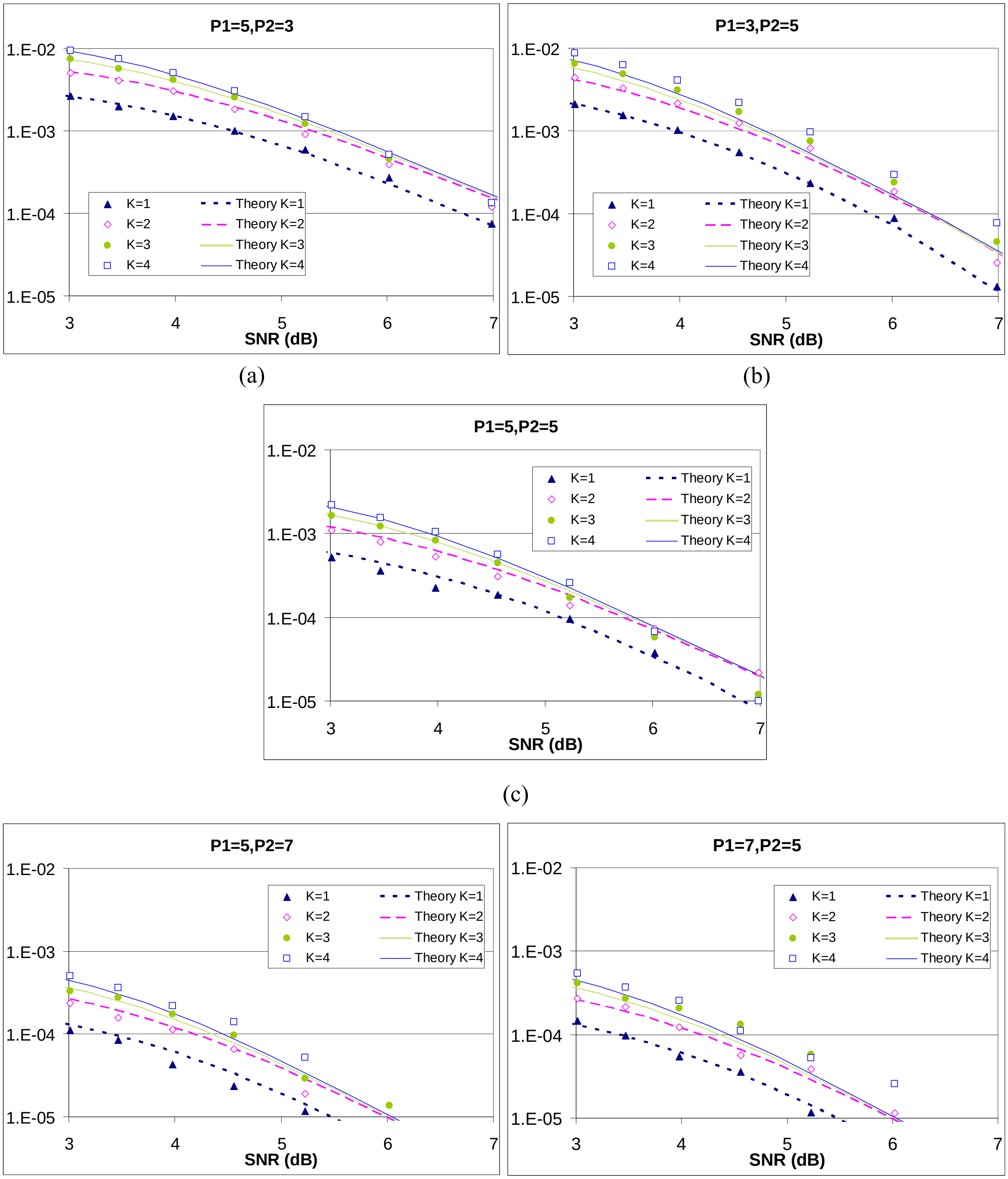


Figure 4.12 BER of the proposed model for $K = 1$, $p_2 = 3$ along $K = 2$, $p_2 = 4$ and $K = 4$, $p_2 = 5$ with a fixed $p_1 = 5$



(a)

(b)

(c)

(d)

(e)

Figure 4.13 BER results of the proposed model for $K = 1 \ldots 4$ from the calculation (lines) and simulation (symbols) (a) p_1=5, p_2=3, (b) p_1=3, p_2=5, (c) p_1=5, p_2=5, (d) p_1=5, p_2=7, (e) p_1=7, p_2=5

4.1.5 Security Enhancement

The private parity bits provide additional error detection capability, which are accessible by authorized receivers only. Therefore, the authorized receivers have a higher reliability level than the other ones.

When the channel noise level is not small (SNR < 5^{dB}), the PED technique plays a key role in the system's performance. Figure 4.14 traces the advantage of an authorized receiver over an unauthorized one. The Block Error Rate is compared with that of the proposed model with $K = 1$, $p_1 = 3$ and 5, and $p_2 = 10 - p_1$ with the base model where $p = p_1$, when the channel noise probability distribution is modelled with the fast Rayliegh fading. Also, it is possible to increase the number of private bits, when the total number of parity bits is kept constant ($p_1 + p_2 =$ cte.) to fix the error detection capability of the authorized receiver and change it for unauthorized receiver.

4.1.6 Performance Analysis

The proposed model achieves a better mean and variance of the number of retransmissions by using more computational power and memory. One of the main concerns about the K best likelihood channel decoder is the time it takes to decode a received codeword. Table 4.2 shows the average time required for a 2.4 GHz Pentium 4 to complete the decoding process. For a better comparison, it is worth noting that the encoding process takes 0.06 milliseconds by using the same hardware [Appendix A]. Also, the encoding and decoding times are almost the same when the SNR changes from 3^{dB} to 7^{dB}.

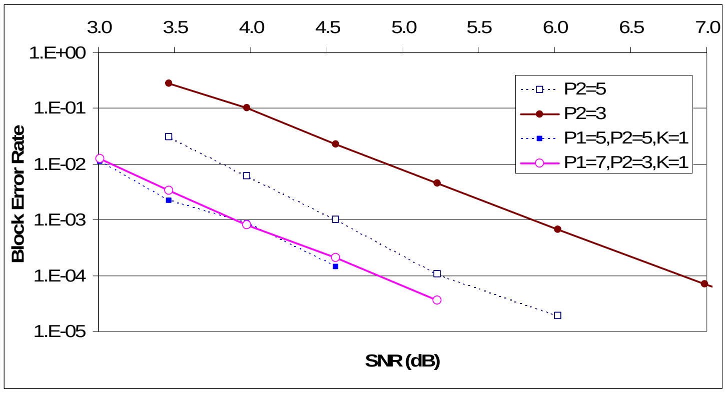
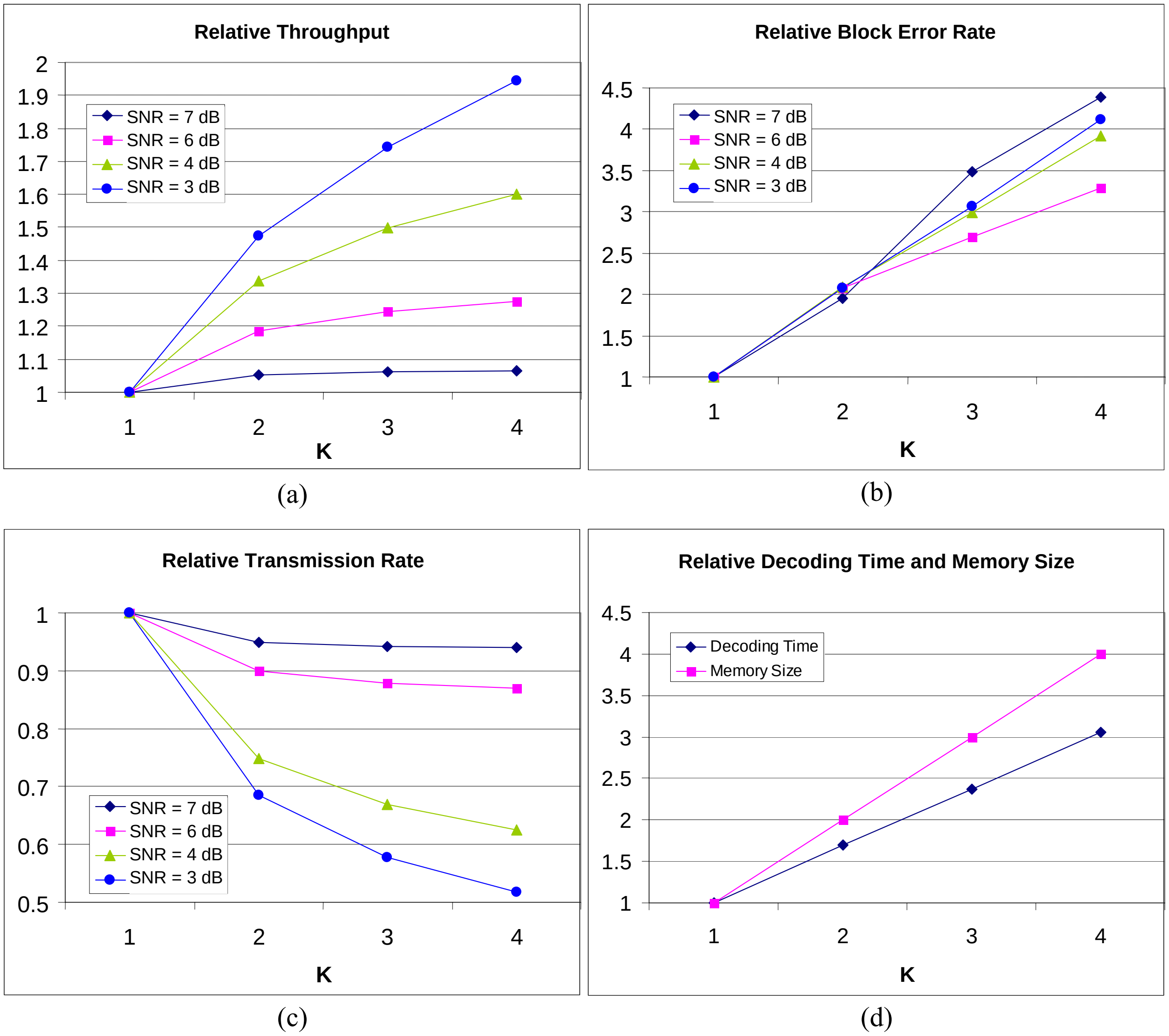


Figure 4.14 BER comparison of the proposed model of K=1, p_1= 3 and 5, and $p_2 = 10$-p_1 with the base model with $p = p_1$



(a) (b)

(c) (d)

Figure 4.15 Effect of increasing *K* on the normalized values of the relative (a) aggregate throughput, (b) BER, (c) transmission rate, and (d) decoding time, and memory size to their value in the base model

Moreover, the size of the memory in the decoder increases proportionally with that of *K*. This occurs because the channel decoder in the proposed model saves the *K* best paths in the trellis during the decoding process (Figure 4.3). Therefore the memory size doubles as *K* changes from 1 to 2.

In Figure 4.15, the results of this study are summarized. The figures show the increase of aggregate throughput, Block Error Rate (BER), computation time, memory size, and retransmission rate as a function of *K*. All of the values are shown in relation to the values when *K*=1, which is equivalent to the base model.

Table 4.2 Average decoding time in the proposed receiver for $K = 1 \ldots 4$

	$K = 1$	$K = 2$	$K = 3$	$K = 4$
Decoding Time (μs)	1203.6	2044.6	2860.3	3679.7

4.1.7 Security Analysis

In the proposed model, the effect of adding parity bits to the input of the security functions must be taken into account. In some security algorithms, this might weaken the security of the communication. If the attacker knows that the fixed bits are added at the end of each block, it might be easier to break the security of the communication.

Also, if unauthorized receivers know that a block has been accepted, they can conclude that by using a channel decoder with the maximum number of outputs, the receiver can decode the correct message. In fact, the only advantage that the authorized receivers have over the unauthorized ones is access to the detection bits which do not carry any information about the message bits. Therefore, by knowing that a block is accepted, it can be concluded that the packet is correctly decodable by a channel decoder with the maximum number of outputs. Usually, the rate of the acknowledged blocks is correlated to the channel conditions. Therefore, if the attacker realizes that the blocks are decoded with a high probability of success, the conclusion might be that the condition in the channel is good and there is no need for additional error detection. Therefore, the attacker chooses the maximum likelihood block, since knowing the probability that the correct block appears in the first output of the channel decoder (P_1) is high. In this case, the attacker decoding reliability is almost the same as the legitimate parties, and the model cannot further enhance the security of the communication anymore. To conceal the information in the message acknowledgements, they can be encrypted, such that it is more difficult for an attacker to break the privacy of the communication.

It is noteworthy that the proposed model only enhances the confidentiality, and does not provide one on its own.

4.1.8 Adaptability

More processing energy conservation can be achieved by augmenting the capability of the receiver's control unit.

If the number of private parity bits (p_2) increases, the probability that a spurious block passes the parity check successfully decreases exponentially. Since the private parity bits are set in the application layer of the sender, it is possible that the receiver's control unit selects the number of private parity bits in the sender's and receiver's, permitting the control unit to change the number of parity bits according to the channel condition.

Therefore, the control unit can increase the number of private parity bits, when the channel condition is poor, to provide greater error detection to the receiver. Similarly, when the channel condition is good, the receiver requires less error detection capability; and therefore, p_2 can be small. By selecting a small p_2 when accurate error detection is not needed, the error detection overhead decreases. Since the sender can send more data, the model has a higher throughput.

Also, it is possible to let the control unit select the number of outputs in the channel decoder. In good channel conditions, the receiver does not need to check other probable decoded blocks than the maximum likelihood block, since there is a very low probability that the correct block is not decoded as the maximum likelihood block. In these cases, the receiver can reduce K to 1 to reserve energy and reduce time in the receiver.

Thus, during the time that there are more errors, the control unit can select a larger K to increase the probability of decoding the correct block. Similarly, with a lower number of errors, the receiver can decrease K to 1. By increasing K, the receiver needs more error detection capability, because of the higher probability of the spurious block (Figure 4.15). As a result, when the receiver controls the number of the channel outputs, it needs to control the number of parity bits as well.

In addition, the control unit can modify the correction capability of the pre-coding and interleaver. This reduces the overhead of the communication and conserves the processing energy at a time when channel conditions are good.

To avoid a high mean delay and variance over a channel, the results indicate that the optimal strategy is to use a channel decoder with a two outputs (K=2) as illustrated in Figure 4.15. Consequently, the maximum reduction in delay characteristics is achieved by minimum changes in the hardware. Also, over a range of SNRs, an improvement in the aggregate throughput occurs. The primary drawback to this change is an increase in the hardware complexity in the proposed receiver by increasing K. The complexity grows slightly less than the linear growth.

Overall, the proposed model which contains the modified Viterbi K best likelihood channel decoder, private parity bits, and a pre-coding and interleaver has a lower number of retransmissions and therefore, a lower mean delay and variance of delay. Over a certain range of residual errors on the channel, a better aggregate throughput is achievable.

4.2 *K*-output Channel Decoder Using the Turbo Code Principle

Considering the blocks at the end of each decoding iteration in a Turbo Code channel decoder as an output, Turbo Code channel decoder can be applied in the proposed model. Then, the decoded block at i^{th} iteration is the i^{th} output of the channel decoder, and K is defined as the maximum number of iterations which the channel decoder executes. Considering the principle of the Turbo Code decoding algorithm, blocks with a higher number of iterations are more likely to be the correct block. Also, the probability that the correct block is decoded with fewer iterations increases, if the SNR of the channel increases.

Depending on the system requirements and the channel characteristics, an optimal number of iterations exists for the Turbo decoder, which ensures useful information is exchanged and used in the Turbo code iterations. This ensures the best possible block is decoded. The optimal number of iterations is affected by the cost of each iteration. Depending on the system resource constraint the energy consumption or the time required to execute one iteration can be included in the iteration cost calculation. The iterations are more sensitive to energy consumption, when the receiver is a battery-powered device. In real-time applications, the mean delay caused by the iteration execution, is the dominant factor in calculating the iteration cost. The cost of the iteration, and the acceptable reliability determines the optimal number of iterations.

Intuitively, with a constant signal power, the higher the noise of the channel, the higher the optimal number of iterations. In order to avoid unnecessary iterations and maintain the acceptable reliability, an error detection technique must be used to detect the correct block during the rounds of the iterations. If the correct block appears in an early iteration in the Turbo decoding, the error detection process can recognize the correct block and stops the decoder from further decoding iterations.

In newly developed model with the Turbo decoder, error detection techniques and the PED are employed to eliminate spurious blocks and detect the correct block. If it appears in the early iteration rounds, the model delivers the correct block with fewer iterations. In other words, the number of iterations in the Turbo decoder is adjusted by detection algorithm. This decreases the average number of iterations for the channel decoder, resulting in a lower delay average, and reduces the average energy consumption per block. By providing the error detection to the authorized receiver only (PED), the reliability of the communication for unauthorized receiver deteriorates.

The next section explains how to use the enhance detection capability provided by PED to eliminates the unnecessary decoding iterations.

4.2.1 Turbo Code Iteration Reduction Using PED

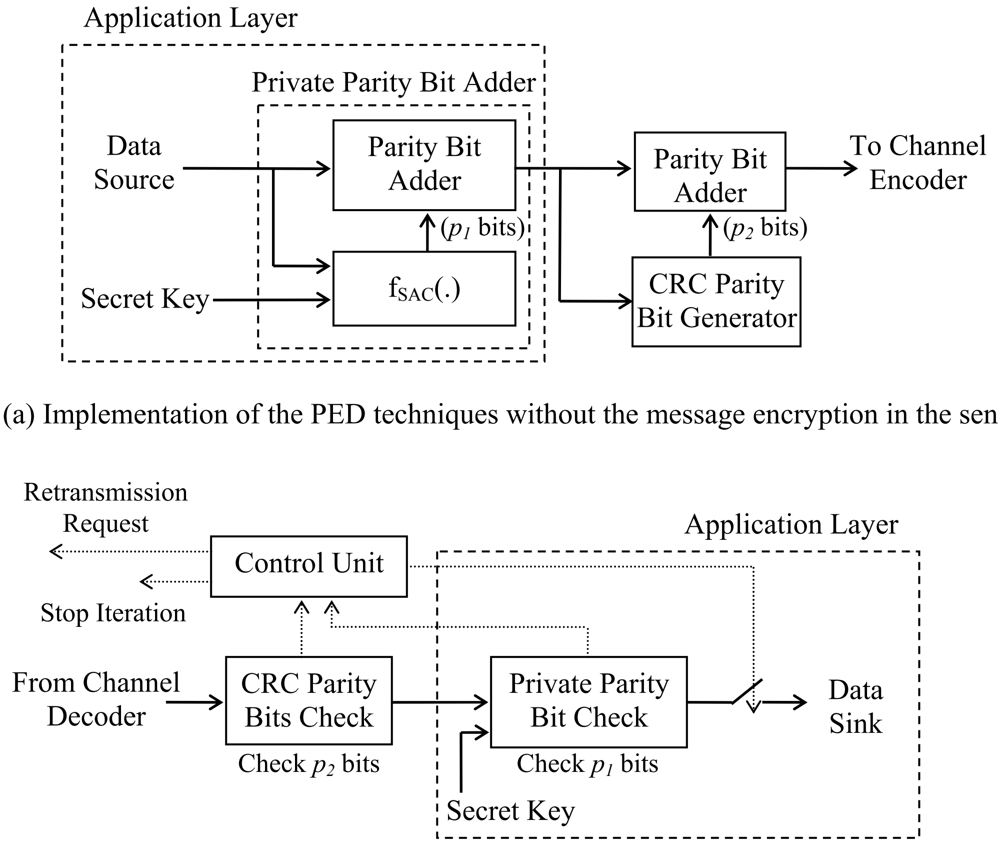
The use of a good detection algorithm in a system with Turbo codes enables the receiver to detect the correct block early in the decoding process and avoids running unnecessary iterations. To do this, an error detection technique, which its false positive probability is as small as possible, must be used.

Figure 4.16 shows how a detection algorithm is implemented without the message encryption in a Turbo code system. This is another way to implement PED technique. It uses the SAC property of the data integrity functions to detect error(s) in the block. Any data integrity process such as HMAC or digital signature can be used in PED. In the proposed model the PED technique is used to improve the error detection accuracy, leading to a higher reliability and better delay characteristics. Since the number of private bits is selected in the application layer, the PED parameters can be adjusted adaptively for the conditions of the communication channel. On the other hand, since the PED technique is useable only by the authorized receivers, the accuracy of error detection is not changed for the unauthorized receivers. This means the unauthorized receivers need to run the maximum number of iterations over all the conditions of the communication channel which leads to the higher delay mean and lower reliability.

The complete model with the Turbo Codes is presented in Figure 4.17.

Figure 4.18 displays the base model which is the equivalent of the proposed model without the PED error detection.

In the proposed model, the data is sent without being encrypted. The only part which is concealed from the unauthorized receiver is the private parity bits, generated by the PED. These bits are used to enhance the error detection capability which leads to a reduction in average number of iterations. Figure 4.20 shows the format of the codeword sent through the channel. The performance of the proposed and the base model is examined in the few next sections.



(a) Implementation of the PED techniques without the message encryption in the sender

(b) Implementation of the PED techniques without the message encryption in the receiver

Figure 4.16 General scheme of the PED techniques without message encryption in the proposed model: (a) sender and (b) receiver

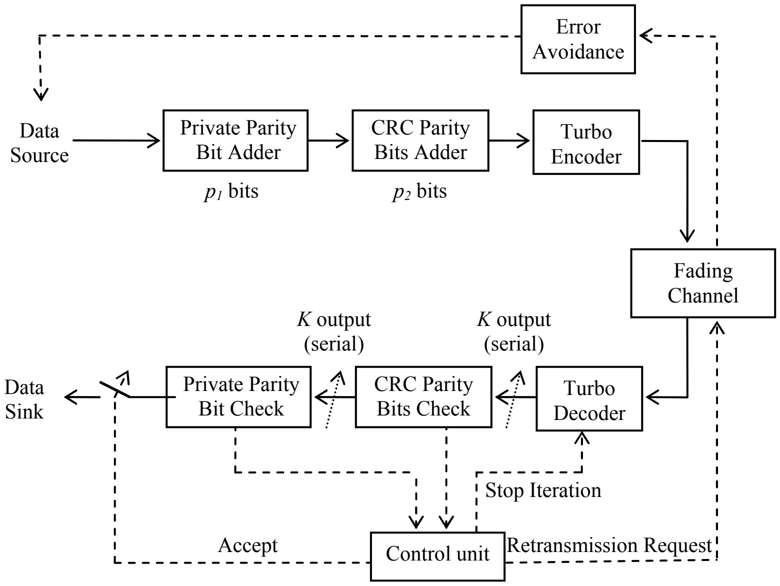


Figure 4.17 Proposed model using the *K* output channel decoder with the Turbo Code decoding algorithm

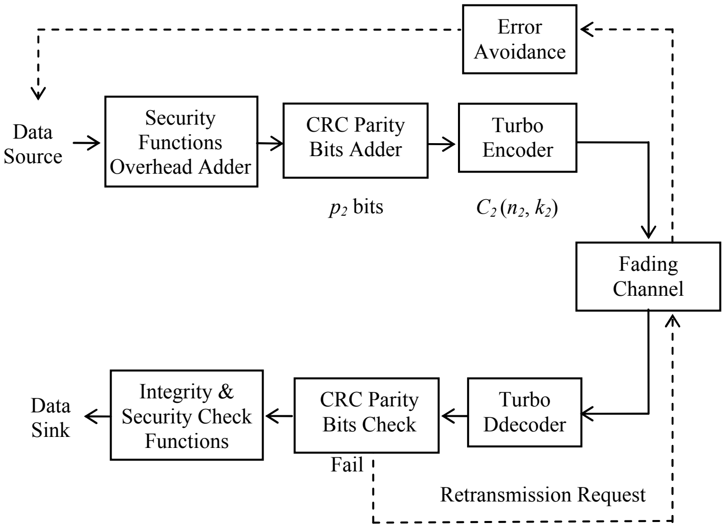


Figure 4.18 Base model using Turbo Codes

4.2.2 Simulation

To generate p_1 bits of private parity bits, the AES algorithm is adopted [82]. Here, the AES is used as the keyed algorithm with the SAC property. The secret key is shared between the sender and all the authorized receivers. After the message is fed into the AES algorithm, p_1-bits of the algorithm's 128-bit output are selected as private parity bits. In fact, any p_1 bit can be selected, but the receiver(s) should know the bit positions. The freedom in the selection of the parity bits position adds more confusion. Here, the last p_1 bits of the algorithm output are selected as the private parity bits. After the selected p_1 bits are appended to the end of the message in the parity bit adder, the CRC parity bit adder provides p_2 bits of parity to the end of the block.

The Turbo coder then encodes the block including private parity bits and produces codewords, as depicted in Figure 4.20.

The Turbo coder consists of two *Recursive Convolutional* (RC) coders with $y_n = x_n + y_{n-1} + y_{n-2}$ to generate codewords, where y_i and x_i are the i^{th} bit of the output and the input respectively. Thus,

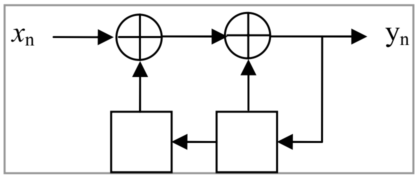


Figure 4.19 RC used in Turbo Code channel encoder with a generator function of G(D)= 1/(1+D+D^2)

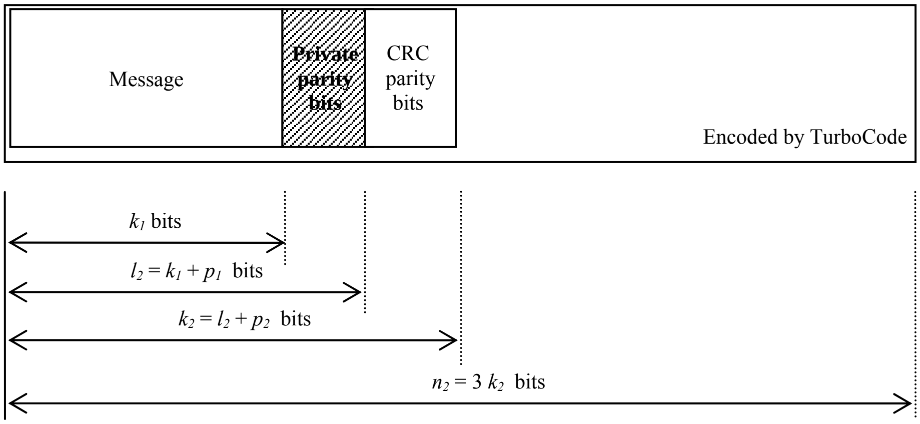


Figure 4.20 Format of the sending codeword in the proposed model

the generator polynomial is G(D)= 1/(1+D+D^2). Figure 4.19 signifies the implementation of the RC coder by using a 2-bit length shift register. The Turbo Code channel encoder contains a random mapping interleaver and two similar RC coders (Figure 2.16). They generate the codewords at a rate of 1/3 as seen in Figure 4.20.

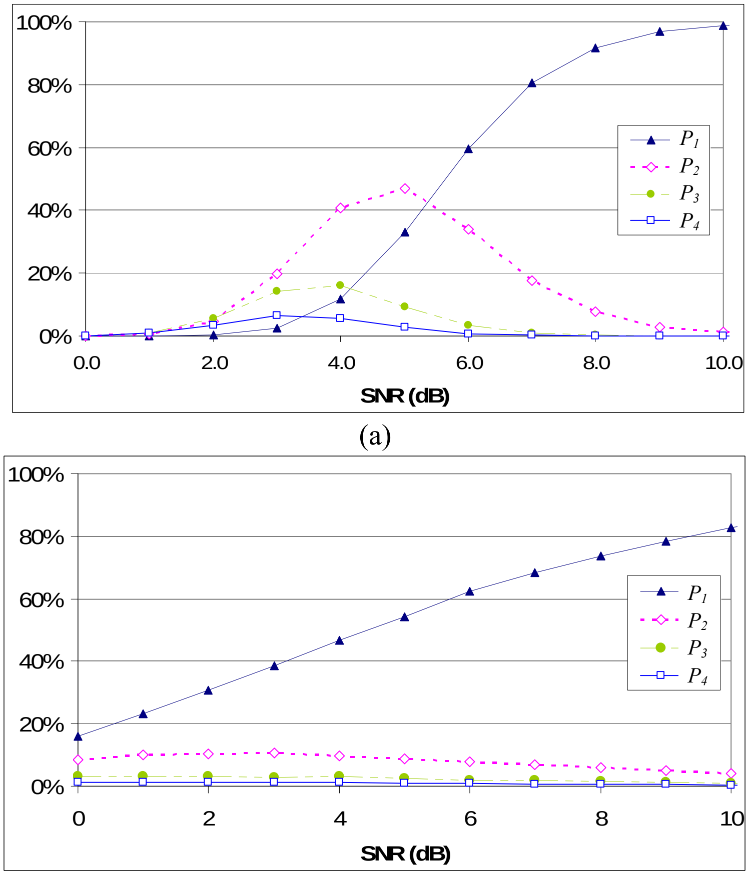
The channel modulation is Binary Phase Shift Keying (BPSK) [73][121]. The communication channel is modelled as a Rayleigh fading channel and is described in Section 4.1.1.

On the receiver's end, a Turbo decoder uses K iterations to decode the received codeword. The PED and CRC parity bit check examine the integrity of the decoded block after each iteration. If both of them confirm the correctness of the block, the block is accepted and the decoding process is stopped.

Figure 4.21 illustrates the P_i probabilities of the proposed model in the fast and the slow block fading channels. Since the values of P_i's for i>1 in the same SNR is bigger in a fast fading channel, better performance is expected under this channel condition, which is described in the next sections.

If a receiver does not possess the PED correct key, the receiver cannot use the private parity bits to detect spurious blocks. Thus, the receiver relies only on the CRC parity bits for error detection. If the size of the CRC parity bits is not enough to provide sufficient error detection capability, the unauthorized receiver(s) cannot exhibit acceptable reliability.

Figure 4.22 depicts the block error rates of the proposed model and the base model. The figure reflects the efficiency of the proposed model in low SNRs. It is observed that the unauthorized receivers have a higher false positive rate. Only the authorized receivers are able to perform with a lower false positive rate, and therefore, a higher reliability. The CRC error detection process has a high probability of a false positive (p_2=3). Therefore, the iterations are stopped before the correct block is reached, resulting in a poorer performance, as seen in Figure 4.22. But, with the appropriate error detection capability, the authorized receivers outperform in any fading condition. This is achieved by the combination of the CRC and the PED error detection processes.

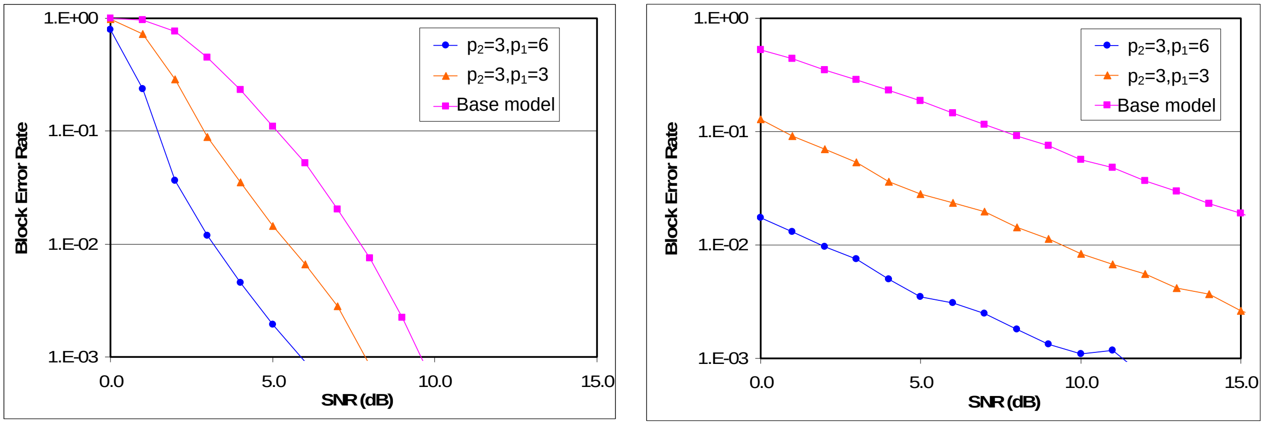


(a)

(b)

Figure 4.21 P_i for $i = 1 \ldots 4$ for modified Turbo Code channel decoder and (a) fast and (b) slow block Rayleigh channel

In a slow block fading channel, the authorized device decodes the transmitted blocks with a 99.7% reliability (for p_2=9 and SNR=5^{dB}), and the unauthorized devices deliver blocks of information with a 81.4% certainty. If the channel is fast fading, the reliability for the authorized and unauthorized devices is 99.8% and 89.0% in SNR=5^{dB}, respectively.



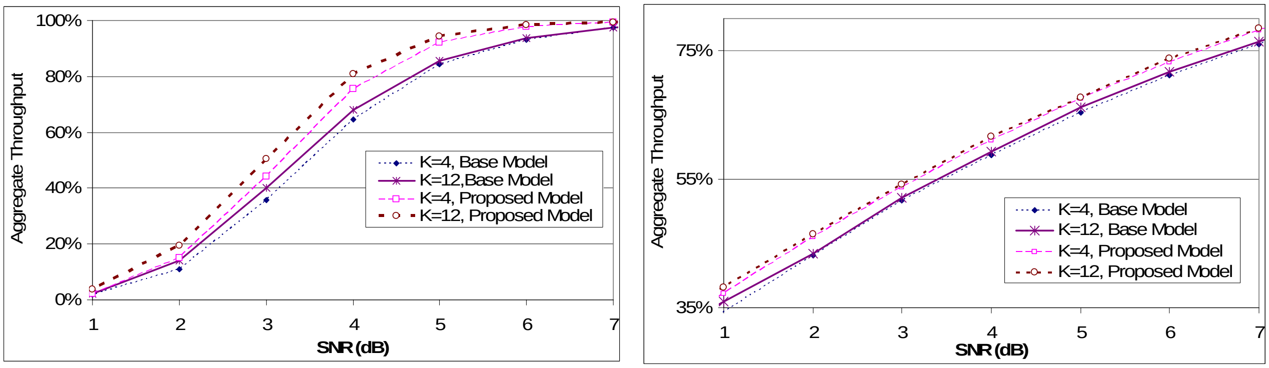
(a)

(b)

Figure 4.22 BER for a maximum iteration of $K = 4$ and block size of N=130 bits, using private parity length of $p_1 = 0$ (base model), 3, 6 and constant $p_2 = 3$, in (a) fast, and (b) slow block fading channels

4.2.3 Throughput

Figure 4.23 illustrates the aggregate throughput of the novel model, compared with the aggregate throughput of the base model under various channel conditions. Obviously, the proposed model outperforms in the region where the base model delivers spurious blocks more frequently, that is, 1^{dB}<SNR< 6^{dB}. This occurs when the channel decoder has an unacceptable probability of failure. The average number of decoding iterations in the proposed model and base model are depicted in Figure 4.24.



(a)

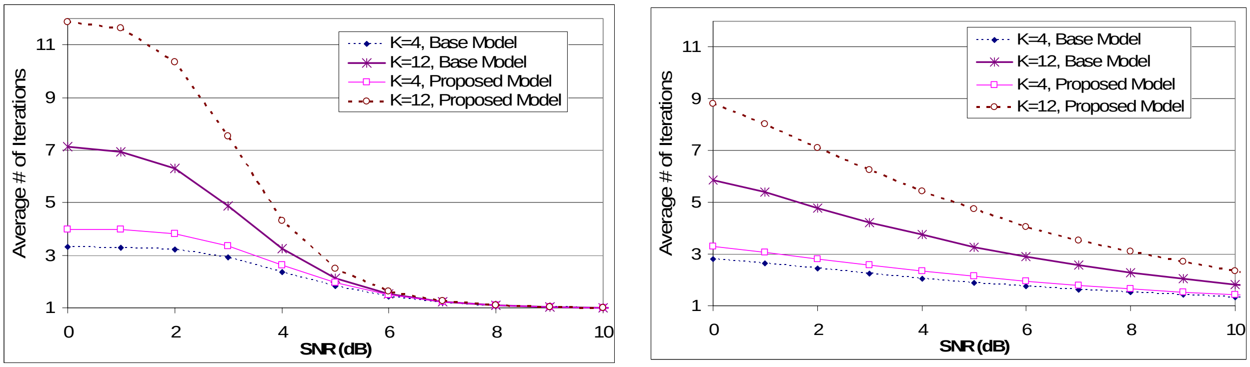
(b)

Figure 4.23 Throughput of the proposed model and base model for K= 4, 12, p_1=3 and p_2=6 in (a) fast and (b) slow block fading channels

4.2.4 Delay and Jitter

The delay mean is a key factor of the performance in wireless real-time applications. The higher the number of iterations, the longer the decoding delay in the system is. This is why designers of practical systems prefer to keep the number of iterations low. However, lowering the number of iterations causes a lower reliability [73]. Iteration execution takes time and increases the delay mean. By reducing the number of iterations, the delay mean of the system decreases. By using an extra error detection capability in the proposed model, the decoding iteration is always stopped at the time that the correct block is decoded. This reduces the average time that is needed to decode a block. However, as the blocks need different rounds of iterations to be decoded correctly, the variance of the decoding delay becomes larger. Therefore, this technique is best used in the applications which are not sensitive to jitter.

Figure 4.25 reflects the average number of iterations versus the false positive rate for the receivers in the fast and the slow block fading channels. It can be seen that with the same average number of iterations, that is, the same mean delay, a receiver with the PED has a lower Block Error Rate. Also, it is shown that by increasing the SNR the block error rate of the unauthorized and the authorized receivers becomes closer and closer. Therefore, in the case of a high SNR to keep the performance difference, the authorized party might want to use other techniques such as chaffing and winnowing, or reordering [112][120].



(a) (b)

Figure 4.24 Average number of iterations of the proposed model for K= 4, 12 and p_1+p_2=9 in (a) fast fading and (b) slow block fading channel

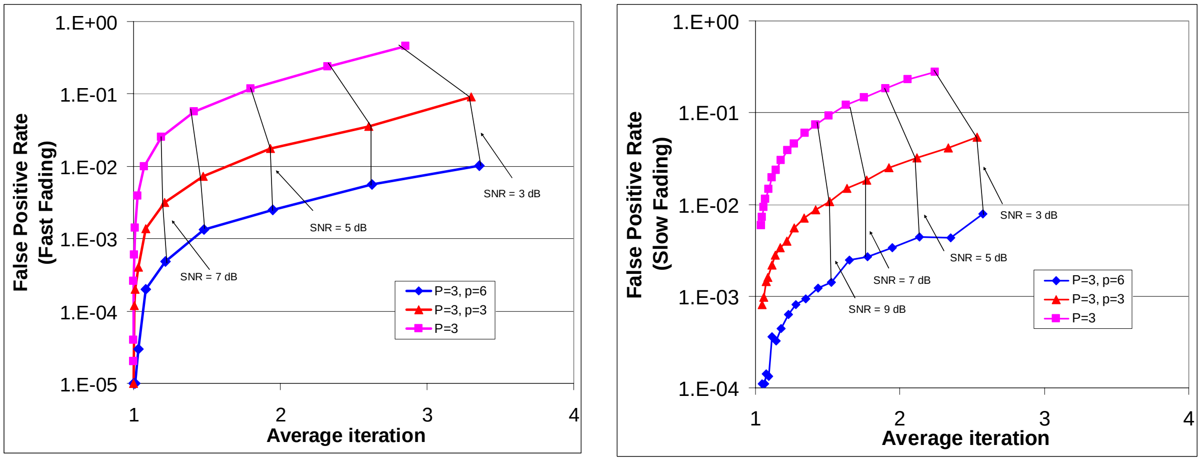


Figure 4.25 Block Error Rate versus average number of iterations in different SNR and different parity detection scheme (*K*=4)

4.2.5 Security Enhancement

In order to use the PED technique the input data stream should resemble a random stream to achieve the flattened probability of all the possible blocks. Here, the receiver expects all combinations of bits in decoded blocks with the same probability. This increases the confusion, since there is no expectation of receiving any particular message at any time and every decoded message, whether it is correct or not, looks like a valid message. Compression and bit-interleaving techniques can be used to convert the input data to a random-like stream [122].

Also, if an unauthorized receiver knows that a packet has been accepted, s/he can conclude that by running enough decoding iterations, the correct message can be decoded. In fact, the only advantage of the authorized receivers possess is access to the private parity bits for error detection which do not carry any information about the message bits. Therefore, by knowing that a packet is accepted, it can be concluded that the packet is correctly decodable by enough processing; that is, the maximum number of iterations. If unauthorized receivers know that a packet has been accepted, they run the maximum number of iterations, and they are sure that the packet is decoded correctly without the need of checking for errors. Then, the adversary decoding reliability is the same as that of the legitimate parties. Also, the adversary can compensate for the delay, associated with high average of iterations, by using a faster processor. Therefore, it is very important to hide the information in acknowledgement packets. By concealing the acknowledgement packet, not only is the reliability of the decoding without the key decrease dramatically, but also, packet reordering can be used to strengthen the security [113].

In addition, to increase the confusion by the noise in the communication channel, without encryption, the PED technique can be fully implemented at the hardware level of the connections, leaving more processing power for the higher levels. Also, other security protocols, running on a higher level of the connection, can co-exist with this technique.

It can be seen that the unauthorized receiver's error detection is less accurate (a lower number of parity bits) and has a higher false positive probability which deteriorate the performance. The unauthorized receiver which does not possess the correct key must run the maximum number of iterations to achieve the lowest error probability. This still does not improve the delay characteristics but does enhance the reliability. If the residual redundancy in the message is not high, the unauthorized receivers do not know if the decoded message is correct or not. If the level of noise in the channel is high, it might not be possible for the unauthorized receiver to have a high reliability. The low reliability of the unauthorized receivers enhances the protection of the sending information. In particular, the low reliability increases the protection against the listening attackers, where a passive attacker listens only to the communication to obtain private information.

The enhancement is only effective when the reliability of the communication is highly dependent on the accuracy of the error detection process.

If the noise level in the channel is low, there is a high probability that the correct block can be decoded without using any parity bits. This is why it is essential to use the PED technique in a noisy channel such as wireless channels. For a low channel noise, the sender can randomly insert some false messages with non-matching private parity bits to create the same effect as noise on the transmitting blocks. This merges the PED technique with chaffing and winnowing technique [112].

It is also possible to send the blocks' order by private parity bits. The messages can be sorted to their original order in the receiver. If the messages are sent in a shuffled order, only the authorized receivers are able to sort the messages into the correct order. The unauthorized receivers who perform the listening attack are not able to correctly order the messages since the receivers do not have access to the information in the private bits. It is noteworthy that in the reordering technique, the messages must be independent of each other to conceal any information about their orders.

4.3 Summary

Overall, a channel decoder that combines *K* maximum likelihood outputs, parity bits and a pre-coding block possesses a lower number of retransmissions, and therefore a lower transmission energy consumption, smaller mean delay and variance of delay. Over a certain range of residual errors on the channel, a better aggregate throughput is achievable. Figure 4.23 and Figure 4.24 display that the proposed models have a better mean delay and variance properties. Furthermore, if the number of parity bits (p_2) increases, the probability that a spurious block passes the parity check successfully decreases exponentially. This provides the freedom to adopt a smaller number of bits to detect the spurious blocks in the channel decoder. The number of parity bits is related to and depends on the detection ability of the channel decoder.

In order to avoid a high mean delay and variance over a channel, the results in this investigation show that the optimal strategy is to use a channel decoder with a double output (K=2) [119]. The maximum reduction in the delay characteristics can be achieved by minimum changes in hardware. Also, over a range of SNRs, an improvement in the aggregate throughput occurs. The higher aggregate throughput is the result of a smaller number of retransmissions, which in turn leads to less transmission energy consumption.

The principle drawback for this change is the higher probability of receiver failure. This can be compensated for by using more accurate error detection techniques which can be provided by PED. Also, the use of the PED gives the authorized receiver the advantage of having the exclusive access to the extra error detection capability. The increase in the hardware complexity in the proposed receiver, by increasing *K*, is a bit better than linear growth.

In the model with the Turbo Code channel decoder, the authorized devices not only have better reliability characteristics but also lower mean delays. This gives the authorized party an advantage over the unauthorized one. Also, over a certain range of SNR, it is always possible to keep the reliability of unauthorized devices lower than the acceptable threshold by using the novel method [120]. Avoiding unnecessary iterations conserves energy in mobile wireless devices.

Chapter 5
Conclusions

The goal of a wireless communications network is to provide a variety of services, including audio, video, and data with high throughput, reliably, and security. Although reliability and security are important, the other performance measures such as delay and jitter are concerns in delay-sensitive applications. In this thesis, techniques are proposed to improve the performance measures for wireless networks. Specifically, the techniques decrease the energy consumption, increase the throughput by reducing the number of retransmissions, enhance the security, and decrease the mean delay and jitter. However, improvements are achieved at the cost of increased processing and complexity.

Typically, to increase the reliability of wireless links, parity bits are added to the transmitted blocks to detect transmission errors. With the knowledge of the error detection technique and access to the parity bits, unauthorized receivers can detect the correct blocks with the same efficiency and reliability as the authorized receivers. If the parity bit generation involves a secret key, then the error detection can only be performed by the parties that possess the correct keys.

In Chapter three, a technique is introduced in which the error detection capability is granted to only the authorized receivers. Since, in this case, only the receivers who share the correct secret key with the sender are able to use the detection algorithm, it is called the *Private Error Detection* (PED) technique. To generate the parity bits, the output of the keyed algorithm with a SAC property is used. The resultant parity bits are called *private parity bits*, since they can be used by the legitimate users who have the correct secret key. Such bits provide extra error detection capability to only the legitimate receivers.

To improve the link quality and reduce the number of retransmissions, a novel technique, the *K Best Likelihood* (*K*-BL) channel decoder, is proposed. In this technique, instead of decoding in favour of the *Maximum Likelihood* (ML) codeword, the decoder provides the *K* most probable codewords (blocks) as the most likely transmitted ones. Since the set of candidate codewords in the *K*-BL decoder contains the maximum likelihood one, the *K*-BL performance is expected to be at least as good as that of the ML decoder. However, it is evident that for a range of low Signal to Noise Ratios (SNRs), employing the *K*-BL decoder results in a low retransmission rate, if a proper error detection strategy is devised. This is due to the fact that the K probable blocks in the output of the K-BL channel decoder contain the correct block with a higher probability than that of the ML one. Therefore, the system using the K-BL channel decoder has more opportunities to decode the correct block.

To implement the idea of the K best likelihood decoder, the well-known Viterbi decoder is modified. The analysis and simulations demonstrate the probability that the correct block appears in the output of the modified Viterbi decoder is higher than that of the original Viterbi decoder in a certain range of the SNR. The *spurious* block terminology is adopted to specify the *erroneous candidate blocks*, delivered by the K-BL decoder. To detect the spurious blocks, an error detection strategy, using the PED technique, is devised. This technique not only provides additional error detection capability to detect the spurious blocks, but also enhances the security.

Since the decryption process, applied on the spurious block, can create burst errors (due to the *Strict Avalanche Criterion*), an interleaver is required to distribute the burst error among the different blocks and deliver them with randomly distributed errors to the *pre-coder*. To improve the delay and jitter characteristics of the system, a *declined interleaver* with optimized depth and span is chosen. The combination of the declined interleaver and pre-coder helped the system to recover the transmitted codewords from the spurious blocks and lower the need for retransmission.

In a secure end-to-end communication, data encryption is handled by the applications. This means that the information is encrypted/decrypted in the application layer of the transmitter/receiver. Consequently, all the units, before encryption and the security functions in the transmitter, and after decryption and security functions in the receiver, should be implemented in the application layer.

In order to meet the secure end-to-end connection requirement, the interleaver and pre-coding technique is implemented in the application layer. This results in the capability to control the added overhead by the application layer in an adaptive manner. That is, when the channel condition is good and the probability that a spurious block reaches the application layer is very low, the application can eliminate the interleaver and pre-coding technique. For poor channel conditions, the application layer can increase the error correction capability of the pre-coder. This adaptive control of the overhead, based on the channel condition, improves the throughput.

The introduction of the new K-BL decoder causes more opportunity for spurious blocks to be selected as correct. Due to the limited correction capability of the pre-coder, the delivery of the spurious blocks with high numbers of errors should be avoided. Consequently, a control mechanism is required in order to prohibit the delivery of such blocks to the pre-coder.

To this end, a *cross-layer controller*, which monitors the flow of information in the receiver, is introduced. By selecting the appropriate criteria for accepting or rejecting the spurious blocks, the novel controller decides whether a retransmission is required or not.

Based on the proposed techniques, a new model is devised, and its performance is compared with that of a conventional model. Analysis and simulation studies prove there are many benefits of the proposed system, some of which are listed next.

- The proposed model requires fewer retransmissions than the base model. The reduction in the number of retransmissions depends on the SNR value. For the SNR range of 4~7 (dB), the reduction in the number of retransmissions is by 57.1% ~5.3%, respectively.
- Lowering the number of retransmissions affects the throughput, delay, and the required transmission energy. The transmission energy is especially important in wireless networks, where most of the energy of the battery-powered devices is claimed for transmission. Depending on the SNR value, the transmitted energy is reduced by 32.1% for SNR=3^{dB}, and 5.1% for SNR=7^{dB}.
- The proposed system reduces the delay mean and variance, crucial parameters in delay sensitive applications.
- The proposed model achieves a higher throughput. For instance, the throughput increases by 42.7% and 5.3% at the SNR values of 3^{dB} and 7^{dB}, respectively. The higher throughput increases the efficiency of the network and allows the network to utilize the applications with higher throughput requirements. A network with higher throughput links has fewer traffic problems such as congestion and connection drops.
- The security of the communication link increases in the proposed model. The security enhancement is the result of applying the PED technique, which prevents unauthorized users from readily detecting the correct blocks with the same efficiency and reliability as those of the authorized devices. It is worthwhile to emphasize that the proposed model does not provide additional confidentiality but does enhance the system security.

Table 5.1 Comparison of the proposed model characteristics (K=2) with the base model

Performance Measure	SNR (dB)				
	3	4	5	6	7
Retransmission Rate Reduction	154.6 %	57.1 %	21.2 %	11.7 %	5.3 %
Throughput Increase	47.2 %	33.8 %	20.9 %	11.3 %	5.3 %
Transmission Rate Reduction	32.1 %	25.3 %	17.4 %	10.1 %	5.1 %
Computational Work Increase	69.8 %	69.8 %	69.8 %	69.8 %	69.8 %

The previous benefits are summarized in Table 5.1. Note that the values are given for the K-BL decoder where K=2. Indeed, increasing K improves the performance measures of the proposed system at the cost of higher complexity. However, the investigation indicates that not much can be gained if this parameter is increased beyond a certain limit. An appropriate range for this parameter is found to be $2 \leq K \leq 4$.

All the above mentioned benefits are achieved at the cost of increased processing and complexity. The proposed scheme adds additional processing. The main processing burden is on the K-BL decoder which provides K-candidate codewords. As a measure of the complexity, the time required for a 2.4 GHz Pentium4 processor [Appendix A] to complete the decoding is measured. The results show that usually the complexity of the decoder increases linearly with K. The complexity is almost independent of the SNR value. As another measure, the memory requirement for different values of K is considered. It is observed that the required memory size is also increased linearly with K. To attain an acceptable complexity and good performance, K=2 is found to be proper value.

To improve the overall performance of the system, the convolutional code can be replaced by the Turbo code. The proposed ideas are extended to a system which utilizes the Turbo code as a channel coding strategy, instead of a convolutional code. To tailor the proposed K-BL decoder to this case, the iterative turbo decoder is restructured in a way to provide K-candidate codewords. For this purpose, a decoded block is declared at the end of the ith iteration as the ith candidate block.

Typically, the Turbo decoder performs a number of iterations in order to provide the final decision. The higher the number of iterations, the longer the decoding delay and the more the processing energy consumption is. Also, limiting the number of iterations degrades the decoding performance. It is desirable to find an optimum number of iterations to meet both the delay and the performance requirements. It is obvious that the number of required iterations depends on the channel noise. Therefore, for a higher channel noise, the Turbo decoder needs more iterations to meet the required performance.

To avoid unnecessary iterations, a stopping criterion should be devised. In this regard, many methods in the literature have been proposed. In the newly devised system, a simple stopping criterion is proposed. This enables the receiver to detect the correct block earlier in the decoding process and avoid running unnecessary iterations. The number of false positives is decreased as a result of using such a stopping criterion.

In the proposed method, error detection and PED techniques are used to eliminate the spurious blocks and detect the correct block. The number of iterations in the Turbo decoder is adjusted by the error detection strength. This results in a reduction in the average number of iterations, and consequently, reduces the energy consumption in decoding process and average decoding delay.

Furthermore, the devised stopping technique leads to an improved security. Since the number of private bits is set in the application layer, the PED parameters can be adjusted adaptively to meet the conditions of the communication channel. Due to the lack of access to the PED parameters, the unauthorized users need to run the maximum number of iterations over all conditions of the communication channel and that leads to a higher delay mean and lower reliability.

To examine the advantages of using Turbo codes and applying the concept of the *K*-BL decoder, the proposed model is evaluated and compared with a conventional-based model. The analysis and simulation studies demonstrate many benefits of the proposed system. Some of them are listed next.

- The Block Error Rate (BER) of the proposed system is much less than that of the base model in both the fast and slow block fading channels. In other words, in spite of using Turbo codes in both models, the proposed model outperforms the base one. This improvement is achieved due to using the PED technique. Increasing the number of parity bits, provided by PED, further improves the performance.

- The aggregate throughput of the proposed model, compared with that of the base model in different channel conditions, that is., fast and slow block fading, is higher. This improvement benefits the system in the SNR range 1^{dB}<SNR< 6^{dB}, in which the channel decoder uses maximum number of iterations and retransmissions occur with a high probability.

- The average number of iterations in the proposed model is fewer than those of the base model. This is due to the proposed stopping method. Since the number of iterations affects the decoding time, it is concluded that the proposed model requires less energy in decoding process than the base one.
- The proposed method has more enhanced confidentiality than the base one due the enhance error correction provided by PED. It is shown that with the same average number of iterations, a receiver with the secret key has a lower false positive rate. Also, it is shown that by increasing the SNR, the false positive rate of unauthorized and authorized receivers became closer and closer. Therefore, for high SNR values, the authorized party may use other techniques such as chaffing and winnowing or reordering to maintain the performance difference.

Some of the advantages of the proposed Turbo coder system are summarized in Table 5.2. One of the benefits of the novel model is security enhancement. In order to use the proposed model to suppress the unauthorized receivers, the input data stream should look like a random stream with uniformly distributed blocks. Then, the receiver expects all the combinations of the bits in the decoded block with the same probability. This increases the confusion, since there is no expectation of receiving any particular message at any time and every decoded message, correct or not, appears to be a valid message. Compression and bit-interleaving techniques can be used to convert the input data to a random-like stream.

Table 5.2 Comparison of the proposed Turbo coded model characteristics with the base model

Performance Measure	SNR (dB)			
	1	3	5	7
Average Iteration Reduction	40.5 %	35.2 %	14.2 %	0 %
Performance (BER) Improvement	24.2 %	2.7 %	1.8 %	0.9 %
Aggregate Throughput Increment	71.7 %	25.5 %	10.3 %	2.0 %
Decoding Processing Time Reduction	40.5 %	35.2 %	14.2 %	0 %

If unauthorized receivers know that a packet is accepted, they can conclude that by running enough decoding iterations, the correct message can be decoded. In fact, the only advantage of the authorized receivers over others is having access to the private parity bits which do not carry any information about the message bits. Therefore, by knowing that a packet is accepted, it is readily concluded that the packet is correctly decodable if the processing is enough; that is, the maximum number of iterations. If the unauthorized receiver knows that a packet has been accepted, he/she runs the maximum number of iterations and he/she is sure that the packet is decoded correctly without the need to detect errors. Here, the adversary decoding reliability is the same as the legitimate parties. The adversary can compensate the delay, associated with high average of iterations, by using a faster processor. Therefore, it is very important to hide the information in acknowledgement packets. By concealing them, the reliability of decoding without the key decreases significantly. In addition, packet reordering can be applied to strengthen the security.

It can be seen that the unauthorized receiver error detection is less accurate (the lower number of parity bits) and has a higher false positive probability which deteriorates its performance.

The proposed models achieve better energy consumption by reducing the number of iterations in a channel decoder that uses the Turbo decoder and by reducing the number of retransmissions in a Trellis channel decoder. Furthermore, the security enhancement of the novel models is an assessment where the enhancement is fully achieved.

5.1 Suggestions for Future Research

The following is suggested for future studies.

- Devise a more complex receiver control unit that can enhance the system performance. For example, selecting the number of channel decoder outputs in an adaptive manner will further improve the processing energy conservation. When the channel is in good condition, the receiver can lower the number of *K*-BL decoder outputs, because the probability that the correct block appears as the maximum likelihood one is very high. The *K* parameter in good channel conditions can even be set to one. This conserves the processing energy in the receiver.
- The transmitter can adaptively determine the number of private parity bits, based on the channel state information provided by the receiver. If the number of private parity bits increases, the probability that a spurious block passes the parity check successfully decreases exponentially.

Since the private parity bits are set in the application layer of the sender, it is possible that the receiver control unit selects the number of private parity bits and sends it back to the sender. This gives the newly developed control unit the ability of changing the number of parity bits according to the channel condition. Also, selecting a small number of private parity bits, when accurate error detection is not needed, decreases the error detection overhead. The sender can send more data, and therefore, the new model described in this thesis, will have a higher throughput.

- The adaptivity, changing the parameters, can be applied to the pre-coding and interleaver as well. The control unit, based on the channel condition, can change the correction ability of the pre-coding and interleaver. This reduces the required overhead and conserves the processing energy, at the time that the channel condition is good.
- In the model, by using the Turbo code, the receiver can change the number of private parity bits to control the error detection accuracy. It can increase the number of private parity bits, when the channel condition is poor, in order to provide more error detection capability to the receiver. Similarly, the receiver needs less error detection capability, when the channel condition is good, and therefore, the number of private parity bits can be small. This decreases the required overhead, and consequently, enhances the throughput.

Appendix A
Simulation

In this appendix, the detail of the simulations and models software implementations are presented. Also the confidence intervals of the results are discussed. The flow charts of the simulation codes are illustrated in the next section.

A.1 Flow Charts

The flow chart of the base model and proposed model are presented in Figure A.1 to Figure A.4. The data flows of the charts are explained in Chapter 4.

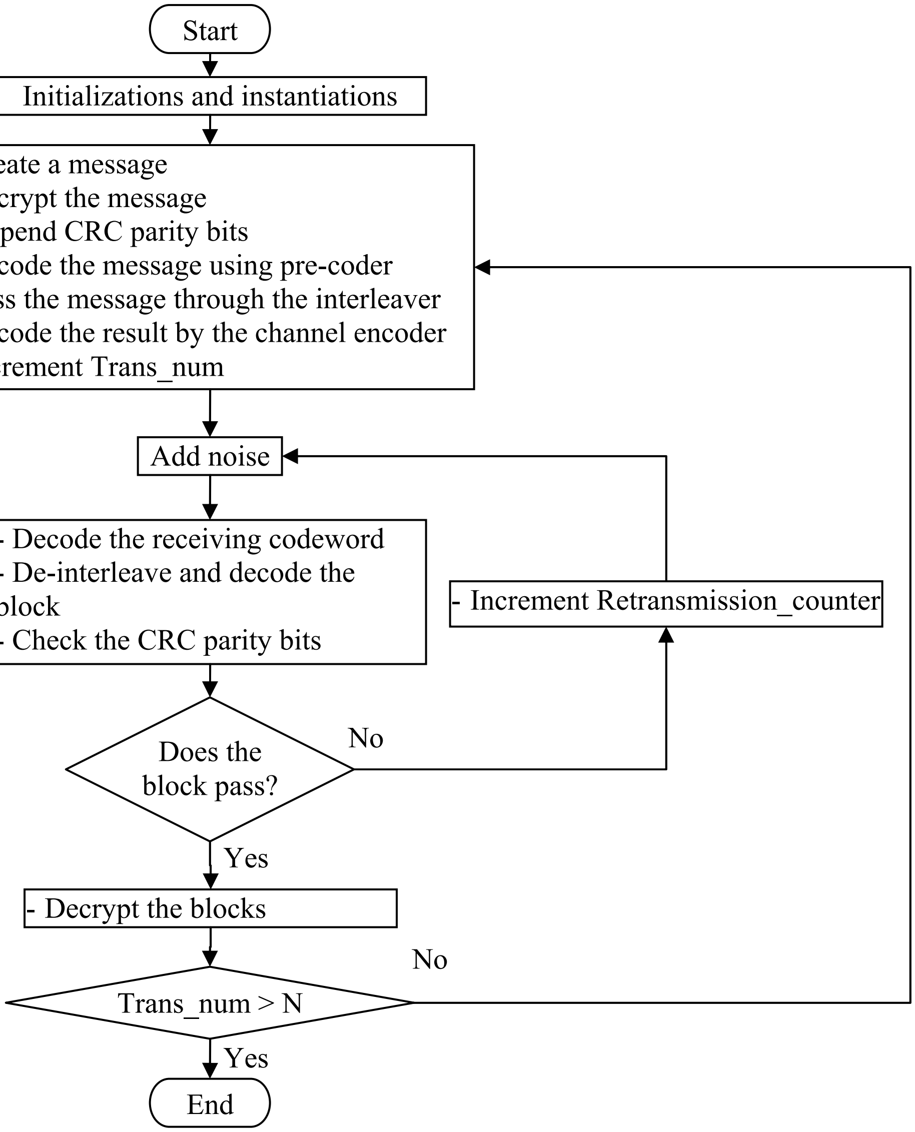

Start
Initializations and instantiations
- Create a message
- Encrypt the message
- Append CRC parity bits
- Encode the message using pre-coder
- Pass the message through the interleaver
- Encode the result by the channel encoder
- Increment Trans_num
Add noise
- Decode the receiving codeword
- De-interleave and decode the block
- Check the CRC parity bits
- Increment Retransmission_counter
Does the block pass?
No
Yes
- Decrypt the blocks
No
Trans_num > N
Yes
End

Figure A.1 Flow chart of base model using the original Viterbi algorithm as in Figure 4.5

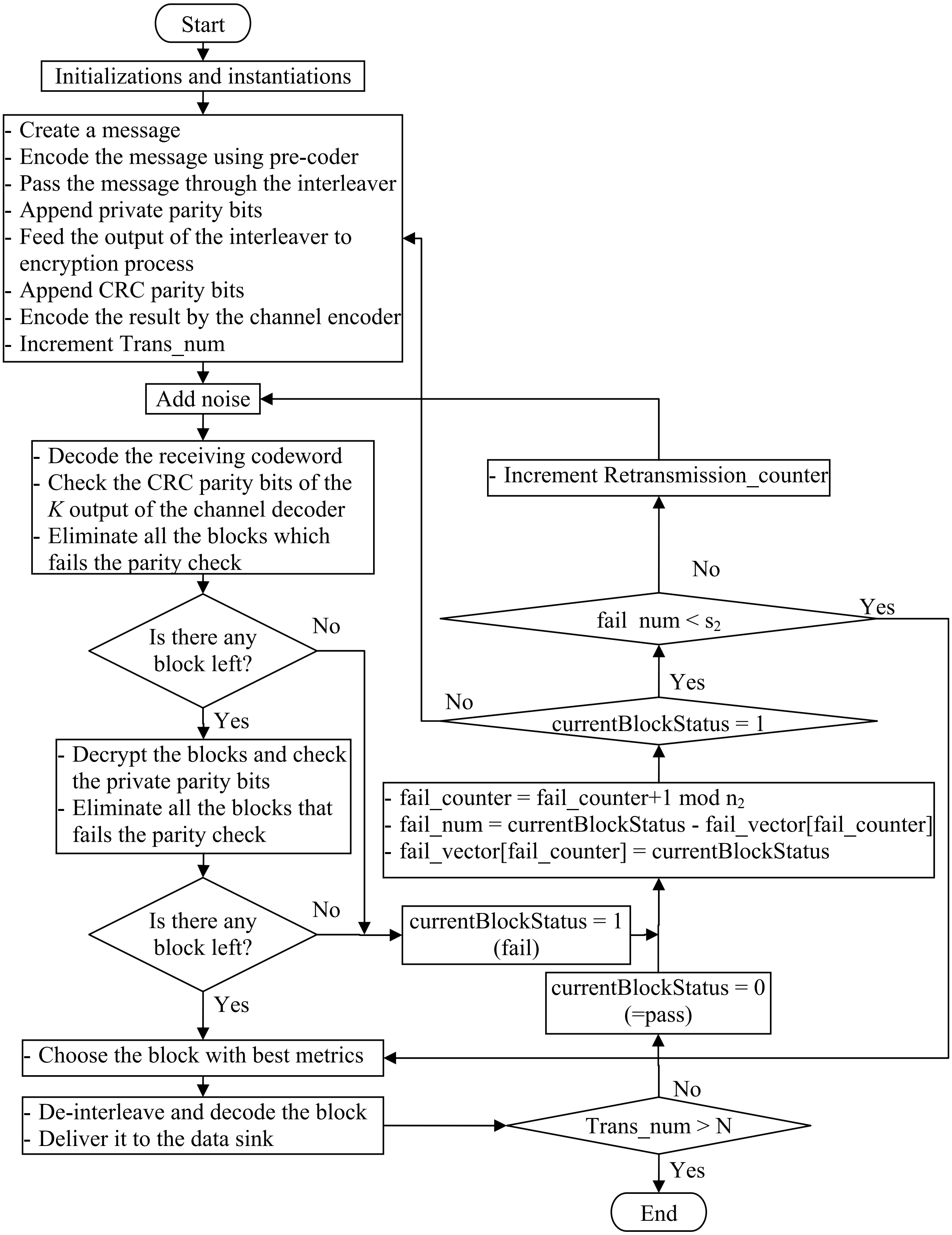

Start
Initializations and instantiations
- Create a message
- Encode the message using pre-coder
- Pass the message through the interleaver
- Append private parity bits
- Feed the output of the interleaver to encryption process
- Append CRC parity bits
- Encode the result by the channel encoder
- Increment Trans_num
Add noise
- Decode the receiving codeword
- Check the CRC parity bits of the *K* output of the channel decoder
- Eliminate all the blocks which fails the parity check
Is there any block left?
No
Yes
- Decrypt the blocks and check the private parity bits
- Eliminate all the blocks that fails the parity check
Is there any block left?
No
Yes
- Choose the block with best metrics
- De-interleave and decode the block
- Deliver it to the data sink
- Increment Retransmission_counter
No
fail num < s_2
Yes
Yes
No
currentBlockStatus = 1
- fail_counter = fail_counter+1 mod n_2
- fail_num = currentBlockStatus - fail_vector[fail_counter]
- fail_vector[fail_counter] = currentBlockStatus
currentBlockStatus = 1 (fail)
currentBlockStatus = 0 (=pass)
No
Trans_num > N
Yes
End

Figure A.2 Flow chart of *K*-output channel decoder using modified Viterbi algorithm as in Figure 4.4

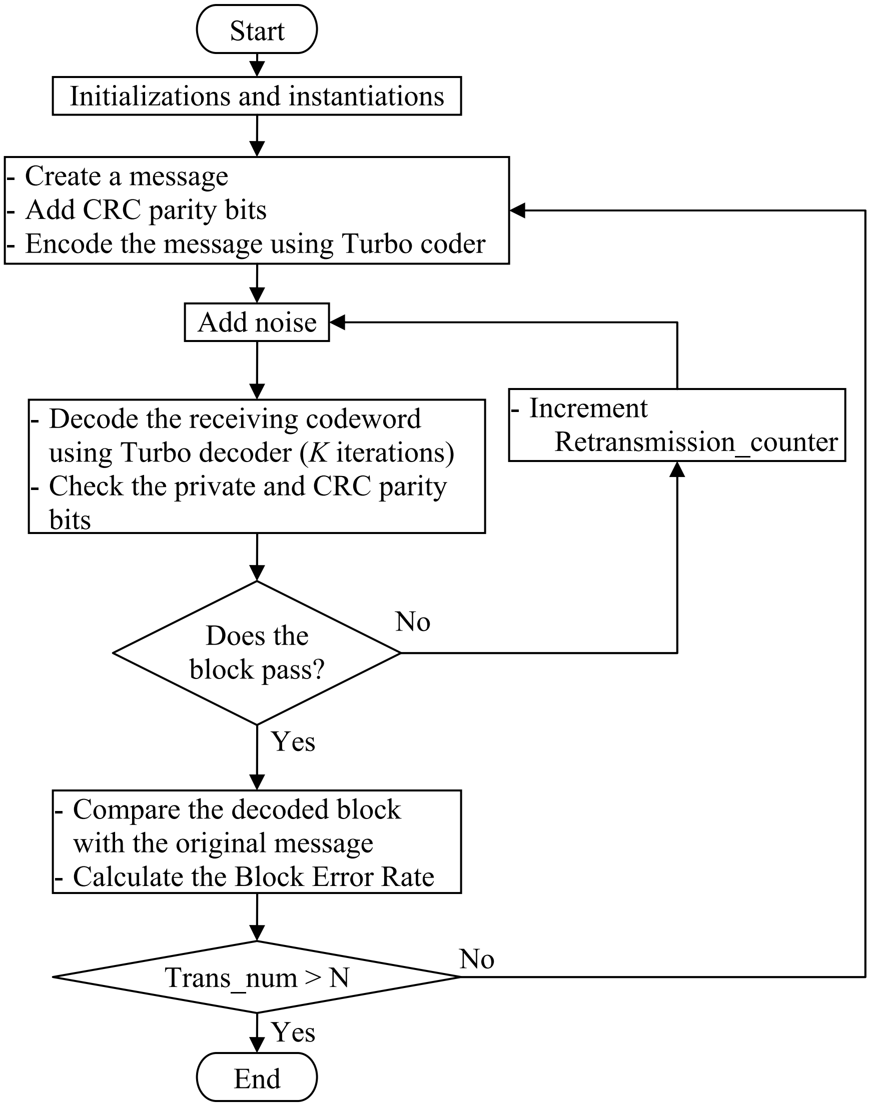

Start
Initializations and instantiations
- Create a message
- Add CRC parity bits
- Encode the message using Turbo coder
Add noise
- Decode the receiving codeword using Turbo decoder (K iterations)
- Check the private and CRC parity bits
- Increment Retransmission_counter
Does the block pass?
No
Yes
- Compare the decoded block with the original message
- Calculate the Block Error Rate
Trans_num > N
No
Yes
End

Figure A.3 Flow chart of base model using Turbo Codes as in Figure 4.18

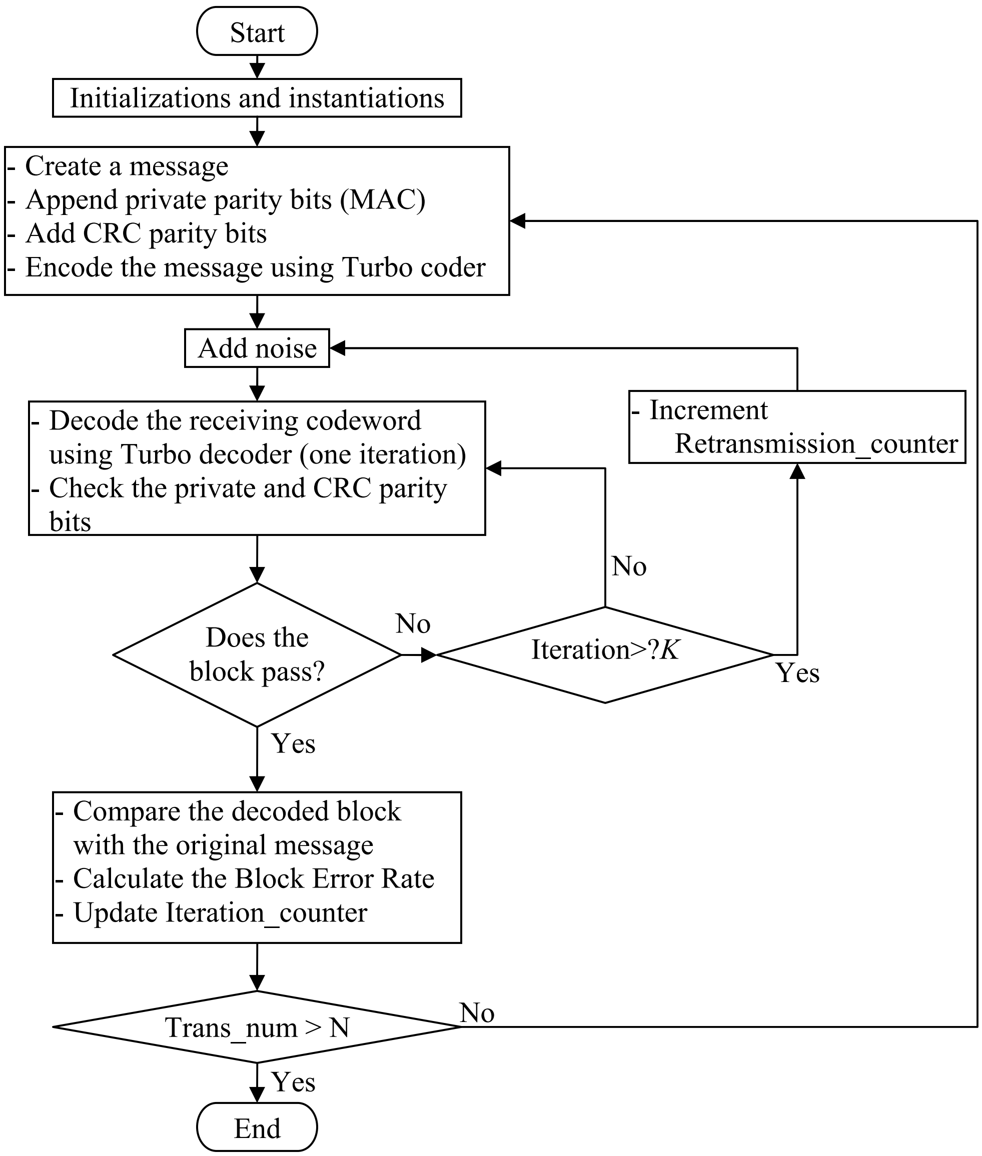

Start
Initializations and instantiations
- Create a message
- Append private parity bits (MAC)
- Add CRC parity bits
- Encode the message using Turbo coder
Add noise
- Decode the receiving codeword using Turbo decoder (one iteration)
- Check the private and CRC parity bits
- Increment Retransmission_counter
No
Does the block pass?
No
Iteration>?K
Yes
Yes
- Compare the decoded block with the original message
- Calculate the Block Error Rate
- Update Iteration_counter
Trans_num > N
No
Yes
End

Figure A.4 Flow chart of proposed model using the *K* output channel decoder with the Turbo Code decoding algorithm as in Figure 4.17

A.2 Testing the code

The performance of the program has been checked to ensure the correct data processing of each module in the simulation. The method that is used to test each module is listed as follows:

- Pre-coder and Pre-decoder

The $C_1(n, k)$ code should be able to retrieve a correct message from a blcok which contains at most t bit in error. The position of the error will be specified by the codeword's syndrome. All combination of the correctable error is generated by the controlled noise and the Pre-coder and Pre-decoder have been checked using the model in Figure A.5.

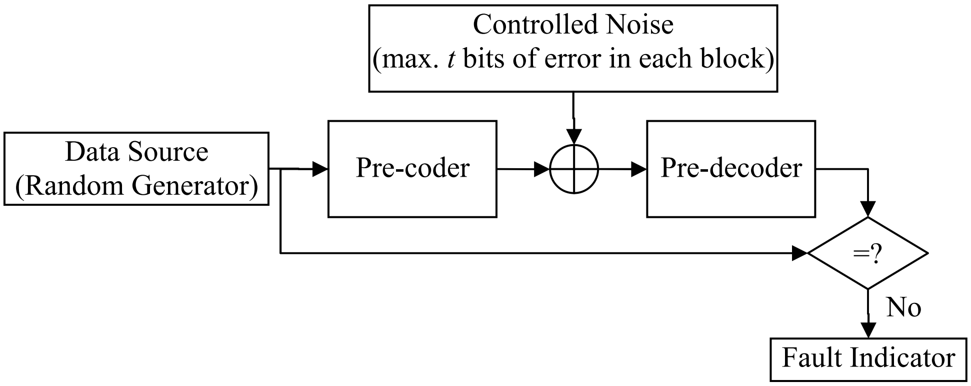


Figure A.5 Pre-coder and Pre-decoder testing model

- Interleaver and De-interleaver

Both declined interleaver and de-interleaver can be implemented as a similar module. To test the functionality of the modules, a block of data that contains random data has been used as the data source. The output of the de-interleaver module has been compared with the original data (Figure A.6).

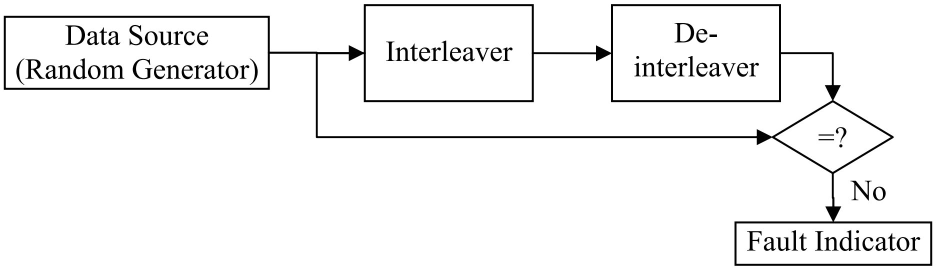


Figure A.6 Interleaver and De-interleaver testing model

- Encryption and Decryption Units

The encryption and decryption units have been tested using a random input and an arbitrary key. To ensure the functionality of the units, 50 different key for the iteration of 10000 have been tested. Figure A.7 illustrates the testing flow chart for the encryption and decryption units.

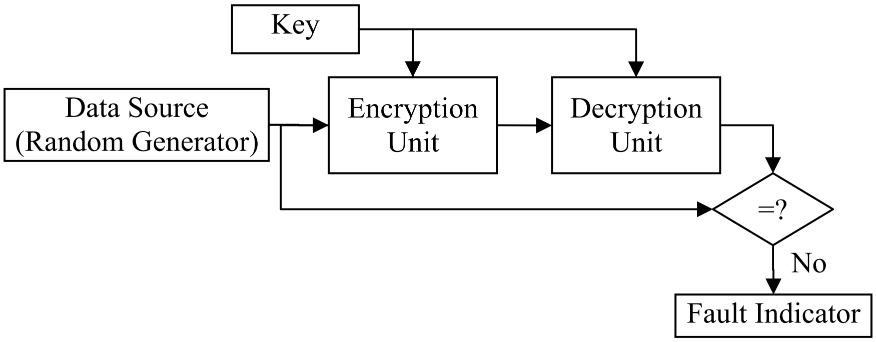


Figure A.7 Flow chart of testing Encryption and Decryption units

- Parity Bit Adder and Parity Bit Checkers

The PED and CRC parity bit adder and check units have been tested as shown in Figure A.8.

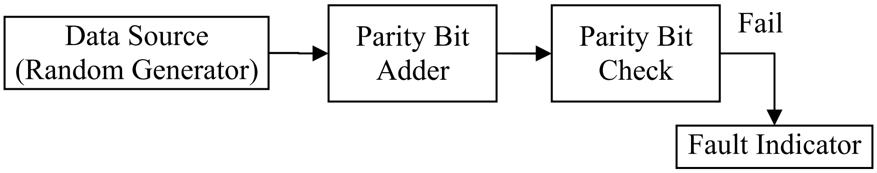


Figure A.8 Testing the functionality of the parity bit adder and parity bit check units

- Channel Encoder and Decoder

The performance of channel coder and decoder was checked in the model presented in Figure A.9. This test was used for both Modified Viterbi and Turbo Code channel decoders. Also the coding and decoding process has been compared with hand-calculated results to ensure the correctness of the program.

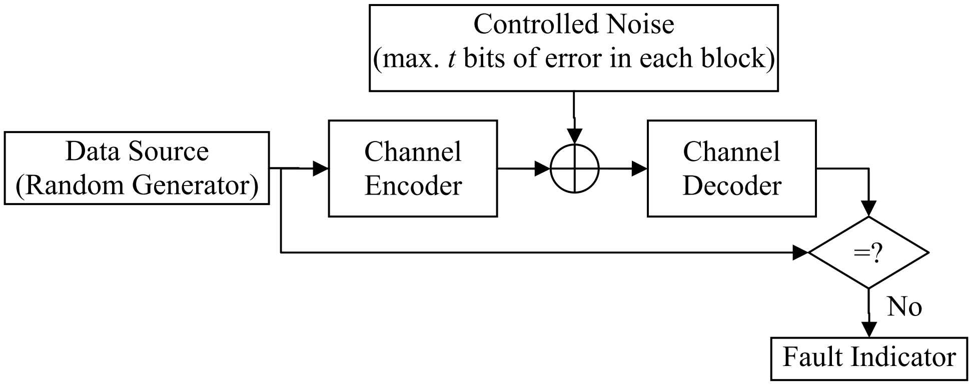


Figure A.9 Channel encoder and decoder testing flow chart

To check the overall model's functionality, the program was first tested under the noise-less channel condition to ensure the model works properly. On the next step, controlled errors were added to the packets in the channel and the results were checked for integrity in the output of the channel decoder. Also bit errors were added to one of the blocks in the input of the Decryption and Security check and the messages were compared to the original message in the receiver. This test checks the performance of the interleaver and pre-coding again.

A.3 Confidence Interval

In this section, the confidence interval of the simulation results is explained. The Confidence Interval (CI) is metric to show the repeatability and accuracy of the simulation. CI of a measurement with normal probability distribution is calculated as follows [125]:

$$CI = \Phi^{-1}\left(1-\frac{\alpha}{2}\right)\times\frac{\sigma}{\sqrt{n}}$$

Where α is the confidence percentage, Φ is the cumulative distribution function of the standard normal distribution, σ is the standard deviation of the measured variable, and n is the total number of simulation run.

As the total number of rounds in the simulations is set relatively high (1000000), it can be concluded from Central Limit Theorem [125][126] that the block error occurs according to the normal distribution [127].

Considering the independency of the block errors in one packet from others, the standard deviation of the error can be calculated as $\sigma = \sqrt{p(1-p)}$, where p is the ratio of the number of times block error is observed to the total number of simulation runs.

Figure A.10 and Figure A.11 depict the results in the Figure 4.13 to Figure 4.14 along with their 95% CI.

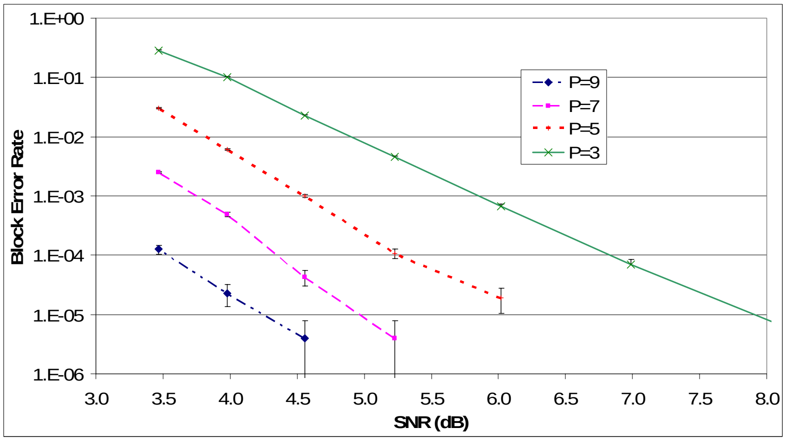


Figure A.10 BER along with 95% CI of the base model presented in Figure 3.14

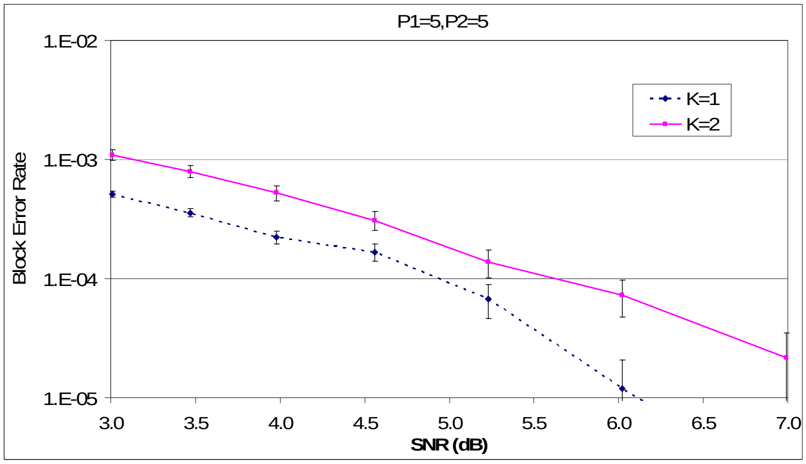
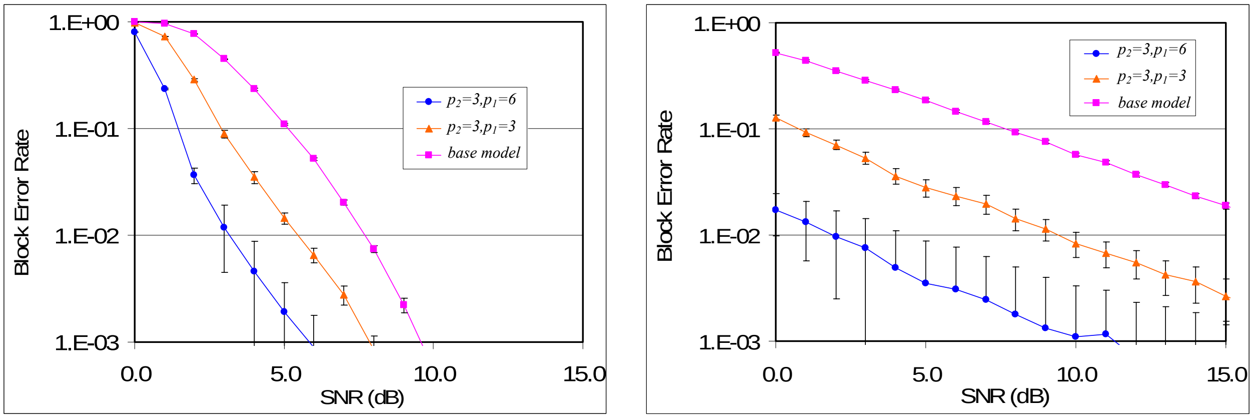


Figure A.11 95% Confidence Interval of the values depicted in Figure 4.13

The 95% Confidence Interval of the throughput figures in Section 4.1.2 are less than $CI_{ModifiedViterbi\_Throughput} \leq 10^{-2}$.

For the proposed model which uses Turbo Code, the 95% CI is depicted in Figure A.12.



(a) (b)

Figure A.12 95% CI for BER graphs in Figure 4.22 (a) fast fading, (b) slow block fading channel

The average number of iterations, presented in Figure 4.24 and Figure 4.25, are calculated for 100001 rounds of simulations. As the result, the throughput confidence intervals are less than $CI_{TurboCode\_Throughput} \le 8 \times 10^{-3}$.

A.4 The Hardware

The simulation was run on a 2.4 GHz Intel Pentium 4 processor. The computer has 512 Mb of RAM and it runs on the Windows XP Professional Operating System (OS).

The results in Table 4.2 shows the average time that is required for this system to execute decoding algorithm.

A.5 The Code

The simulation codes are in enclosed CD. Please refer to Readme.txt for more details.

A.6 Classes

To understand the code faster and easier, the implemented classes that are used in the codes are listed below. In addition, the input(s) and a brief description of the class are presented.

- "BinSeq.h"

Input: Sequence length

Instantiates a binary Sequence of arbitrary length and the binary operations such as AND, OR, XOR.

- "Interleaver.h"

Input: Span and deep of the interleaver

Instantiates declined interleaver and de-interleaver blocks with the given span and deep size.

- Parity.h

Input: Size of the private parity bits

Instantiates a private parity adder and private parity check blocks.

- MyAES.h

Input: Null

Instantiates the encryption and decryption blocks.

- Convolutional.h

Input: Size of the codeword, number of channel decoder output

Instantiates the channel encoder and decoder blocks.

- Channel.h

Input: Null

Instantiates the communication channel including noise generator function.

- time.h

Input: Null

Instantiates the time measuring functions.

- CRC.h

Input: Null

Instantiates the functions that generate, append, and check CRC parity bit.

- Random.h

Input: Null

Instantiates a uniform random number generator which outputs a random number between zero and 1 [127].

- TurboCPTCRC.h

Input: Null

Instantiates the Turbo Code channel encoder and decoder. The decoder has the stop condition based on the result of the CRC and private parity bits checks [125][127].

Appendix B

Calculation

Some of the mathematic close formulas that are used in the thesis are presented below:

$$\sum_{i=0}^{\infty} P^i = \frac{1}{1-P}$$

$$\sum_{i=0}^{\infty} iP^i = \frac{P}{(1-P)^2}$$

$$\sum_{i=0}^{\infty} i^2 P^i = \frac{P(1+P)}{(1-P)^3}$$

$$\sum_{i=0}^{\infty} i(i+1)P^i = \frac{2P}{(1-P)^3}$$

Bibliography

[1] S. Aissa, and E. Dubois, “Coding with Dynamic Rate Control for Low-Delay Image Transmission over CDMA Fading Channels,” Forty Eighth IEEE Vehicular Technology Conference, Volume 2, pp. 1488-1492, May 1998.

[2] Zhiwei Cen, Matt W. Mutka, Danyu Zhu, and Ning Xi, “Improved Transport Service for Remote Sensing and Control over Wireless Networks,” IEEE International Conference on Mobile Adhoc and Sensor Systems, Nov. 2005.

[3] K. M. Rege, and Dong Sun, “A Simple Analytical Model to Estimate VoIP Signaling Delays in an HFC Access Network,” IEEE Global Telecommunications Conference, Volume 1, 2005.

[4] C. Chiasserini, and Michela Meo, “Impact of ARQ Protocols on QoS in 3GPP Systems,” IEEE Transactions on Vehicular Technology, Volume 52, pp. 205-215, Jan. 2003.

[5] Nir Naaman, and Raphael Rom, “Bandwidth Scheduling for Multi-Channel Packet Cable Telephony,” in Proceedings of Eleventh International Conference on Computer Communications and Networks, pp. 537-542, Oct. 2002.

[6] M. L. Das, A. Saxena, and V. P. Gulati, “A security framework for mobile-to-mobile payment network,” IEEE International Conference on Personal Wireless Communications, pp. 420-423, Jan. 2005.

[7] Chung-Huang Yang, “A 6805-based security system for broadcasting stock information,” in Proceedings of IEEE Thirty Forth Annual 2000 International Carnahan Conference on Security Technology, pp. 238-241, Oct. 2000.

[8] S. S. Y. Shim, V. S. Pendyala, M. Sundaram, and J. Z. Gao, “Business-to-business e-commerce frameworks,” IEEE Computer Magazine, Volume 33, Issue 10, pp. 40-47, Oct. 2000.

[9] Kwei Tu, “A CCSDS command authentication scheme,” in Proceedings of IEEE Conference on Computers, Communications, Control and Power Engineering, Volume 1, pp. 141-144, Oct. 2002.

[10] A. Lenstra, and E. Verheul. “Selecting cryptographic key sizes,” Journal of Cryptology, Volume 14, No. 4, pp. 255-293, 2001.

[11] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of Applied Cryptography,* CRC Press, 1997.

[12] C. H. Cho, J. J. Won, and H. W. Lee. “Performance of Hybrid II ARQ Schemes Using Punctured RS Code for Wireless ATM,” in IEE Proceedings on Communication, Volume 148, No. 4, pp. 229-233, Aug. 2001.

[13] Yuan Chen, and Lemin Li, “A fair packet dropping algorithm considering channel condition in diff-serv wireless networks,” The Fourth International Conference on Computer and Information Technology, CIT '04, pp. 554-559, Sep. 2004.

[14] D. Grace, T. C. Tozer, and A.G. Burr, “Reducing call dropping in distributed dynamic channel assignment algorithms by incorporating power control in wireless ad hoc networks,” IEEE

Journal on Selected Areas in Communications, Volume 18, Issue 11, pp. 2417-2428, Nov. 2000.

[15] G. Caire, and D. Tuninetti, “ARQ Protocols for the Gaussian Collision Channel,” in Proceedings of IEEE Transactions on Information Theory, pp. 406, July 2001.

[16] Andrew S. Tanenbaum, *Computer Networks*, Forth edition, Prentice Hall, 2003.

[17] J. Sanchez, J.F. Troncoso, and J.R. Gallardo, “Retransmission algorithm base on power priorities for wireless networks,” The Thirteenth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Volume 3, pp. 1146-1150, Sept. 2002.

[18] M. Zorzi, “Capture Probabilities in Random-Access Mobile Communications in the Presence of Rician Fading,” IEEE Transactions on Vehicle Technology, Volume 46, No. 1, pp. 96-101, Feb. 1997.

[19] M. Zorzi, and R. R. Rao, “Capture and Retransmission Control in Mobile Radio,” IEEE Journal on Selected areas in Communications, Volume 12, No. 8, pp. 1289-1298, Oct. 1994.

[20] Irving S. Reed, and Xuemin Chen, *Error Control Coding for Data Networks*, Kluwer Academic Publishers, 1999.

[21] D. L. Lu, and J. F. Chang, “Performance of ARQ Protocols in Non-independent Channel Errors,” IEEE Transactions on Communication, Volume 41, Issue 5, pp. 721-730, May 1993.

[22] Y. J. Cho, and C. K. Un, “Performance Analysis of ARQ Error Controls under Markovian Block Error Pattern,” IEEE Transactions on Communication, Volume 42, Issue 234, Part 3, pp. 2051-61, Feb.-Apr. 1994.

[23] M. Zorzi, and R. R. Rao, “Throughput Analysis of ARQ Go-Back-N Protocol in Markov Channels with Unreliable Feedback,” in Proceedings of IEEE International Conference on Communications, pp. 1232-1237, June 1995.

[24] M. Zorzi, and R. R. Rao, “Throughput Analysis of ARQ Selective-Repeat Protocol with Time Diversity in Markov Channels,” in Proceedings of IEEE Global Telecommunications Conference, pp. 1673-1677, Nov. 1995.

[25] Shu Lin, and P. S. Yu, “A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels,” IEEE Transactions on Communications, Volume 30, Issue 7, Part 2, pp. 1701-1719, July 1982.

[26] J. Metzner, “Improvements in Block-Retransmission Schemes,” IEEE Transactions on Communications, Volume 27, Issue 2, Part 1, pp. 524-532, Feb. 1979.

[27] Q. Yang, and V. K. Bhargava, “Delay and coding gain analysis of a truncated type-II hybrid ARQ protocol,” IEEE Transactions on Vehicular Technology, Volume 42, Issue 1, pp. 22-32, Feb. 1993.

[28] C. Fuiiwara, S. Hirasawa, and W. W. Chu., “Feedback Error Control System with Limited Number of Retransmissions,” in Proceedings of Third Symposium of Information Theory and its Applications, Japan, pp. 328-332, Nov. 1980.

[29] M. Zorzi, and R.R. Rao, “Energy-constrained Error Control for Wireless Channels,” IEEE Personal Communications, Volume 4, Issue 6, pp. 27-33, Dec. 1997.

[30] W. Zhuang, “Integrated Error Control and Power Control for DS-CDMA Multimedia Wireless Communications,” in IEE Proceedings on Communications, Volume 146, Issue 6, pp. 359-365, Dec. 1999.

[31] S. G. Kim, Ki-Jun Kim, Youngwoo Yun, Soonyil Kwon, and B.K. Yi, “Synchronous H-ARQ with Energy Reduction (ER-HARQ) Retransmissions,” Fifth IEE International Conference on 3G Mobile Communication Technologies, pp. 227-229, 2004.

[32] N. Bambos, and J. M. Rulnick, “Mobile Power Management for Maximum Battery Life in Wireless Communication Networks,” in Proceedings of IEEE the Conference on Computer Communications, pp. 443-450, Mar. 1996.

[33] N. Bambos, and J. M. Rulnick, “Performance Evaluation of Power-Managed Mobile Communication Devices,” in Proceedings of IEEE International Conference on Communications, pp. 1477-1481, June 1996.

[34] D. L. Lu, and J. F. Chang, “Performance of ARQ Protocols in Non-independent Channel Errors,” IEEE Transactions on Communication, Volume 41, pp. 721-730, May 1993.

[35] Y. J. Cho, and C. K. Un, “Performance Analysis of ARQ Error Controls under Markovian Block Error Pattern,” IEEE Transactions on Communication, Volume 42, pp. 2051-2061, Feb.-Apr. 1994.

[36] M. Zorzi, and R. R. Rao, “Throughput Analysis of ARQ Go-Back-N Protocol in Markov Channels with Unreliable Feedback,” in Proceedings of IEEE International Conference on Communications, pp. 1232-1237, June 1995.

[37] N. Arulselvan, and R. Berry, “Joint Power-Error Control Schemes for Time-varying Wireless channels,” Fortieth Annual Conference on Information Sciences and Systems, pp. 492-497, Mar. 2006.

[38] H. E. Gamal, G. Caire, and M. O. Damen, “The diversity-multiplexing-delay tradeoff in MIMO ARQ channels ,” in Proceedings of International Symposium on Information Theory, ISIT 2005, pp. 1823-1827, Sept. 2005.

[39] E. Teletar, “Capacity of Multi-antenna Gaussian Channels,” AT&T-Bell Labs, Technical Report, 1995.

[40] G. J. Foschini, and M. Gans, “On the Limits of Wireless Communication in a Fading Environment When Using Multiple Antennas,” Wireless Personal Communication, Volume 6, pp. 311-335, Mar. 1998.

[41] Yu-Ming Wang, and Shu Lin, “A Modified Selective-Repeat Type-II Hybrid ARQ System and Its Performance Analysis”, IEEE Transactions on Communications, Volume 31, No. 5, pp. 593-608, May 1983.

[42] N. Arulselvan, and R. Berry, “Energy-throughput Optimization for Wireless ARQ Protocols,” IEEE International Conference on Acoustics, Speech, and Signal Processing, Volume 5, pp.

797-800, Mar. 2005.

[43] Mohammed N. Smadi, and Barna Szabados, "Error-Recovery Service for the IEEE 802.11b Protocol," IEEE Transactions on Instrumentations and Measurement, Volume 55, No. 4, Aug. 2006.

[44] E. Modiano, "An Adaptive Algorithm for Optimizing the Packet Size Used in Wireless ARQ Protocols," Wireless Networks, Volume 5, No. 4, pp. 279-286, Aug. 1999.

[45] S. Hsiao, and A. Hwang, "Bit-Error Recovery with Adaptive Packet Sizes for Wireless Network Environments," Harvard University Press, 2000.

[46] D. A. Eckhardt, and P. Steenkiste, "Improving Wireless LAN Performance via Adaptive Local Error Control," in Proceedings of IEEE Sixth International Conference on Network Protocols, pp. 327-338, 1998.

[47] E. Ayanoglu, "Adaptive ARQ/FEC for multi-tone transmission in wireless networks," in Proceedings of IEEE Global Telecommunications Conference, pp. 2278-2283, 1995.

[48] J.-S. Ahn, and J. Heidemann, "An adaptive FEC algorithm for mobile wireless networks," USC/Information Sciences Institute, Marina del Rey, CA, Tech. Rep. ISI-TR-555, 2002.

[49] H. Kwon, Tae H. Kim, and S. Choi, "A Cross-Layer Strategy for Energy-Efficient Reliable Delivery in Wireless Sensor Networks," IEEE Transactions on Wireless Communications, Volume 5, No. 12, Dec. 2006.

[50] G. Battail, and R. Sfez, Lecture Notes in Computer Science, "Sub-optimum Decoding Using Kullback Principle", pp. 93-101, Springer, 1988.

[51] M. Moher, "Decoding via cross-entropy minimization," in Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '93), pp. 809-813, Dec. 1993.

[52] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," IEEE Transactions on Information Theory, Volume 42, Issue 2, pp. 429-445, Mar. 1996.

[53] J. Hamorsky, and U. Wachsmann, "Criteria for minimum bit error rate decoding of turbo-codes,'' in Proceedings of Kleinheubacher Tagung, pp. 607-616, 1995.

[54] A. Banerjee, D. J. Costello Jr., and Thomas E. Fuja, "Comparison of Different Retransmission Strategies for Bandwidth Efficient Hybrid ARQ Schemes Using Turbo Codes," IEEE International Conference on Personal Wireless Communications, pp. 548-552, Dec. 2000.

[55] John G. Proakis, "*Digital Communications*," McGraw-Hill, 1995.

[56] Marvin K. Simon, Mohamed-SlimGordon, and L. Stuber, *Principle of Mobile Communication*, Klumer Academic Publichers, 1996.

[57] S. B. Wicker, "Error Control Systems for Digital Communication and Storage," Prentice Hall, 1995.

[58] Marvin K. Simon, and Mohamed-Slim Alouini, *Digital Communication over Fading Channels,*

A Unified Approach to Performance Analysis, John Wiley and Sons Inc., 2000.

[59] Shu Lin, and Daniel J. Costello. *Error Control Coding: Fundamental and Applications*, Prentice-Hall, 1983.

[60] Jorge C. Moreira, and Patrick G. Farrell, *Essentials of Error-Control Coding*, John Wiley and Sons Inc., 2006.

[61] H. Minn, M. Zeng, and Vijay K. Bhargava. "On ARQ scheme With Adaptive Error Control," IEEE Transactions on Vehicular Technology, Volume 50, No. 6, pp. 1426-1436, November 2001.

[62] T. R. N. Rao, and E. Fujiwara, *Error-Control Coding for Computer Systems*, Prentice-Hall, 1989.

[63] Stephen B. Wicker, and Vijay K. Bhargava (Editors). "Reed Solomon Codes and Their Applications," IEEE Press, 1994.

[64] Charles Lee, *Convolutional Coding: Fundamentals and Applications*, Artech House, 1997.

[65] Zhu Danyu, and M. Mutka, "Sharing presence information and message notification in an ad hoc network," in Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, pp. 351-358, Mar. 2003.

[66] T. Sato, M. Kawabe, T. Kato, and A. Fukasawa, "Data transmission protocol in integrated mobile communications network," IEEE Vehicular Technology Conference, Volume 2, pp. 656-659, May 1989.

[67] Shu Lin, *Trellises and trellis-based decoding algorithms for linear block codes*, Kluwer Academic, 1998.

[68] G. Ungerboeck, "Channel coding with multilevel/phase signals," IEEE Transactions on Information Theory, Volume 28, Issue 1, pp. 55-67, Jan 1982.

[69] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Transactions on Information Theory, Volume 13, Issue 2, pp. 260-269, Apr. 1967.

[70] Q. Huang, S. Chan, L. Ping, K. Ko, and M. Zukerman, "A hybrid ARQ scheme based on CT-TCM codes," IEEE Communications Letters, Volume 9, Issue 7, pp. 664-666, July 2005.

[71] E. Biglieri, D. Divsalar, P. J. McLane, and N. K. Simon, "*Introduction to Trellis Coded Modulation with Applications*," Prentice Hall, 1992.

[72] C. Rerrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting coding and decoding: Turbo-Codes," International Conference on Communications, pp. 1064-1070, May 1993.

[73] C. Heegard, S. B. Wicker, *Turbo Coding*, Kluwer Academic Publishers, 1999.

[74] Nam Yul Yu, Min Goo Kim, Yong Serk Kim, and Sang Uoon Chung, "Efficient stopping criterion for iterative decoding of turbo codes", Electronics Letters, Volume: 39, Issue: 1, pp.

73-75, 9 Jan. 2003.

[75] P. Fire, “A Class of Multiple Error Correcting Binary Codes for Non-independent Errors,” Sylvania Reconnaissance System Lab., Mountain View, Calif., Sylvania Rep. RSL-e-2, 1959.

[76] H. O. Burton, “A class of asymptotically optimal burst correcting block codes,” Presentation in International Conference on Computers and Communication, June 1969.

[77] G. D. Forney, Jr., “Burst-Correcting Codes for the Classic Bursty Channel,” IEEE Transactions on Communications, Volume 19, Issue 5, Part 1, pp. 772-781, Oct 1971.

[78] Douglas R. Stinson, *Cryptography Theory and Practice*, CRC Press, 1995.

[79] Kiarash Narimani, “Reliable End-to-End Secure Connections in an Ad Hoc Networking Environment”, Bell University Laboratories report, University of Waterloo, Waterloo, Ontario, Canada, Report NSSG-2003-01.

[80] William Stallings, *Cryptography and Network Security: Principles and Practice*, Prentice-Hall, 1999.

[81] National Institute of Standards and Technology (NIST), at http://www.nist.gov.

[82] J. Daemen, V. Rijmen, AES Proposal: Rijndael, AES Algorithm Submission, September 3, 1999, available at http://www.nist.gov/CryptoToolkit.

[83] B. Schneier, *Applied Cryptography*, John Wiley and Sons Inc., 1996.

[84] X. Lai, and J. Massey, “A Proposal for a New Block Encryption Standard,” in Proceedings of the EUROCRYPT 90 Conference, pp. 389-404, 1990.

[85] J. Soto, and L. Bassham, “Randomness Testing of the Advanced Encryption Standard Finalist Candidates,” available at http://csrc.nist.gov/publications/nistir/ir6483.doc

[86] R. Morris, “The Data Encryption Standard: Retrospective and Prospects,” IEEE Communications Magazine, Volume 16, No. 6, pp. 11-14, 1978.

[87] Howard M. Heys, and Stafford E. Tavares, “Avalanche Characteristics of Substitution-Permutation Encryption Networks,” IEEE Transactions on Computers, September 1995, Volume 44, No. 9, pp.1131-1139

[88] Xun Yi, Shi Xin Cheng, Xiao Hu You, and Kwok Yan Lam, “A method for obtaining cryptographically strong 8×8 S-boxes,” IEEE Global Telecommunications Conference, Volume 2, pp. 689-693, Nov. 1997.

[89] P. Gutmann, D. Naccache, and C.C. Palmer, “When hashes collide,” IEEE Security & Privacy Magazine, Volume 3, Issue 3, pp. 68-71, May-June 2005.

[90] FIPS-186-2 available at http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf

[91] FIPS-186-3 available at http://csrc.nist.gov/publications/drafts/fips_186-3/Draft-FIPS-186-3%20_March2006.pdf

[92] N. Daswani, “Cryptographic Execution Time for WTLS Handshakes on Palm OS Devices,”

Certicom Public Key Solutions, San Jose, CA, Sept. 2000.

[93] M. J. Wiener, "Cryptanalysis of short RSA secret exponents," IEEE Transactions on Information Theory, Volume 36, Issue 3, pp. 553-558, May 1990.

[94] D. Naccache, and D. M'Raihi, "Cryptographic smart cards," IEEE Micro, Volume 16, Issue 3, pp. 14, 16-24, June 1996.

[95] V. S. Miller, "Use of Elliptic Curves in Cryptography," in Proceedings of Advances in Cryptology, pp. 417-426, 1986.

[96] N. Koblitz, "CM-Curves with Good Cryptographic Properties," in Proceedings of Advances in Cryptology, pp. 279-287, 1992.

[97] A. Shamir, "A fast signature scheme," MIT Laboratory for Computer Science rep., Cambridge, MA, 1978.

[98] P.R. Zimmermann, *The Official PGP User's Guide*, MIT Press, 1995.

[99] http://www.rsa.com/glossary/?id=1051

[100] The Wireless Application Protocol (WAP) architecture http://www.wapforum.org/what/WAP_white_pages.pdf

[101] Akhilesh Shrestha, and Liudong Xing, "A Performance Comparison of Different Topologies for Wireless Sensor Networks", IEEE Conference on Technologies for Homeland Security, pp. 280-285, May 2007.

[102] Ruiping Ma, Liudong Xing, and Howard E. Mickel, "A New Mechanism for Achieving Secure and Reliable Data Transmission in Wireless Sensor Networks", IEEE Conference on Technologies for Homeland Security, pp. 274-279, May 2007.

[103] Z. I. Dafalla, T. C. T. Tan, M. B. R. Murthy, and S. Clement, "Radio resource management for WLAN networks through power control," Thirteenth IEEE International Conference on Communication Jointly held with the IEEE Seventh Malaysia International Conference on Networks, , Volume 2, Nov. 2005.

[104] Shao-Yi Hung, Shao-Yi Hungi, Po-Yu Chuang, Po-Yu Chuangi, Yi-Hsien Tseng, "Energy Efficient TCP Transmission for IEEE 802.15.3 WPAN," IEEE Seventeenth International Symposium on Personal, Indoor and Mobile Radio Communications, pp. 1–6, Sept. 2006.

[105] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, "Analyzing Energy consumption Security Protocols," in Proceedings of International Symposium on Low Power Electronics and Design (ISLPED'03), Seoul, Korea, pp. 30-35, Aug. 2003.

[106] P. Prasithsangaree, and P. Krishnamurthy, "Analysis of Energy Consumption of RC4 and AES in Wireless LANs" in Proceedings of IEEE Global Telecommunications Conference, pp. 1445-1449, 2003.

[107] N. R. Potlapally, S. Ravi, A. Raghunathan, and G. Lakshminarayana, "Optimizing public-key encryption for wireless clients," IEEE International Conference on Communications, Volume 2, pp. 1050-1056, 2002.

[108] R. Karri, and P. Mishra, "Modeling energy efficient secure wireless networks using network simulation," IEEE International Conference on Communications, Volume 1, pp. 61-65, May 2003.

[109] R. Karri, and P. Mishra, "Optimizing the energy consumed by secure wireless session – Wireless Transport Layer Security case study," Journal of Mobile Networks and Applications, Special Issue on Security, ACM/Kluwer Publications, Volume 8, No. 2., Apr. 2003.

[110] R. Karri, and P. Mishra, "An investigation into the design of energy-efficient session negotiation protocols for wireless networks," in Proceedings of IEEE Global Telecommunications Conference, Volume 6, pp. 3488-3492, Dec. 2003.

[111] H. Little (Research In Motion), Presentation in Certicom ECC Conference, 2004.

[112] R. L. Rivest, "Chaffing and Winnowing," MIT Lab for Computer Science, Mar. 1998, Available at http://theory.lcs.mit.edufrivest/chaffing.txt.

[113] D. Chaum, "Untraceable electronic mail, return addresses, and digital Pseudonyms," Communications of the ACM, Feb. 1981.

[114] C. F. Tschudin, "Header hopping and packet mixers," in Proceedings of Ninth International Conference on Computer Communications and Networks, pp: 316-319, Oct. 2000.

[115] B. H. Simov, and S.B Tridandapani, "Providing privacy without encryption in multi-channel systems," Conference Record of IEEE International Conference on Communications, Volume 1, pp. 270-274, June 1998.

[116] A. F. Webster, and S.E. Tavares, "On the Design of S-boxes," Advances in Cryptology, Springer-Verlag, 1985.

[117] Howard M. Heys, and Stafford E. Tavares, "Avalanche Characteristics of Substitution-Permutation Encryption Networks", IEEE Transactions on Computers, Volume 44, No. 9, pp.1131-1139, Sep. 1995.

[118] Xun Yi, Shi Xin Cheng, Xiao Hu You, and Kwok Yan Lam, "A method for obtaining cryptographically strong 8×8 S-boxes," in Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '97), Volume 2 , pp. 689-693, Nov. 1997.

[119] Kiarash Narimani, and G. B. Agnew, "Retransmission Reduction Technique for End-to-end Secure Wireless Connections", Submitted.

[120] Kiarash Narimani, and G. B. Agnew, "Cryptographic Stop Criterion for Turbo Decoder", Submitted.

[121] A. Bruce Carlson, *Communication Systems: An Introduction to Signals and Noise in Electrical Communications*, Third edition, McGraw-Hill Inc., 1986.

[122] T. M. Cover, J. A. Thomas, *Elements of Information Theory*, John Wiley and Sons Inc., 1991.

[123] "H323 standard", available at http://www.h323forum.org

[124] Gordon B. Agnew, “Cryptographic Systems Using Redundancy, IEEE Transactions of Information Theory, pp. 31-39, Jan. 1990.

[125] Henk Tijms, *Understanding Probability: Chance Rules in Everyday Life*,” Cambridge University Press, 2004.

[126] http://www.statisticalengineering.com/central_limit_theorem.htm .

[127] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*, John Wiley and Sons Inc., 2002.