TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

LIST OF ABREVIATIONS

ADC	Analog-to-Digital Converter
BEC	Binary Erasure Channel
BER	Bit-Error Rate
BLER	Block-Error Rate
BPSK	Binary Phase-Shift Keying
BSC	Binary Symmetric Channel
CP	Cyclic Prefix
CRC	Cyclic Redundancy Check
DAC	Digital-to-Analog Converter
DARC	Data Radio Channel
DCS	Digital Communication system
DMT	Discrete Multi-Tone
FPGA	Field-Programmable Gate Array
FFT	Fast Fourier Transform
GF	Galois Field
HW	Hamming Weight
HD	Hamming Distance
IDT	Initial Detection Time
IFFT	Inverse Fast Fourier Transform

IMT-2020	Internation Mobile Telecommunications for 2020
ISI	Inter-Symbol Interference
ITU	International Telecommunication Union
LDPC	Low-Density Parity-Check
LLR	Log-Likelihood Ratio
LR	Likelihood Ratio
LTE	Long-Term Evolution
ML	Maximum-Likelihood
MSPRT	Multi-hypothesis Sequential Ratio Test
M2M	Machine-to-Machine
OFDM	Orthogonal Frequency-Division Multiplexing
PC	Polar Code
PDF	Probability Density Function
PMF	Probability Mass Function
PSK	Phase-Shift Keying
P/S	Parallel-to-Serial
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
SC	Successive-Cancellation
SNR	Signal-to-Noise Ratio

SSC	Simplified Successive-Cancellation
S/P	Serial-to-Parallel
WCS	Wireless Communications System
5G	Fifth Generation

INTRODUCTION

In the last three decades, although it is true there have been considerable improvements in the performance of a wireless communication system (WCS) in terms of bandwidth, throughput, jitter and error rate, latency has been mostly neglected, (Rumble *et al.* (2011)). Nowadays there are real-time applications such as high-frequency trading, vehicle to vehicle communications, interactive online gaming (Nikaein *et al.* (2011)), among others, that require very low latencies and evidently this subject requires more attention. For example, in high-frequency trading, a reduction of 1 ms of latency could represent a profit of 100 million of dollars per year, or a fraction of millisecond over a multihop network of radio links could mean a game-changer, (Martin (2007)). It is envisioned that traffic efficiency and traffic safety applications, based on machine-to-machine (M2M) communication, require less than 5 ms end-to-end latency for exchanging information and avoiding traffic accidents, (Osseiran *et al.* (2014)).

The recommendation ITU-R M.2083-0, published by the International Telecommunication Union (ITU) Radiocommunication sector (ITU-R (2015)), indicates that the reduction of latency can be achieved through technical developments covering the radio interface and network architecture aspects. Au & Gagnon (2016) have proposed an innovative strategy to decrease the detection latency of a multicarrier communication system using a scheme of sequential early detections. However, the configuration of this scheme has not yet been found.

On the other hand, after Claude Shannon has shown that there is a tight upper bound on the transmission rate through a noisy communication channel, known as the channel capacity, researchers have tried to achieve this maximum rate employing different mechanisms of error-correcting codes. An important and recent code family are polar codes, proposed by Erdal Arikan (2009), which asymptotically achieve the symmetric capacity of any given binary-input discrete memoryless channel (B-DMC) with a low encoding and decoding complexity, in the order of $O(Nlog_2N)$.

Polar codes discovery is a significant accomplishment since first-ever capacity-achieving has been proved theoretically, in contrast to empirical approaches developed by known high performance codes like turbo and low-density parity-check (LDPC) codes. The construction of these latter codes involves randomness which implies more complex implementations (Sarkis (2016)). The encoder and decoder of polar codes in terms of practical implementation have a recursive structure, a fixed and deterministic complexity, and an explicit construction, consequently they are easy to implement. But not everything is beneficial, the serial nature of the successive-cancellation (SC) decoder results in low decoding throughput and the error rate performance is mediocre for short to moderate code lengths. These issues have been overcome with a fast simplified successive-cancellation (Fast-SSC) decoding algorithm (Sarkis *et al.* (2014)) and a cyclic redundancy check (CRC)-list decoding algorithm (Tal & Vardy (2015)), respectively. The low decoding speed of the CRC-list decoder is improved with a fast SC-list decoder algorithm based on unrolling the decoding tree of the code (Sarkis *et al.* (2016)). Hardware implementations of polar decoders achieve very good results in the order of micro seconds of latency, specifically with fast-SSC decoders (Sarkis *et al.* (2014); Giard *et al.* (2015a)) and unrolled fast-SSC polar decoders (Giard *et al.* (2015b)).

The original construction algorithm of polar codes is based on the Bhattacharrya parameter for binary erasure channels (BEC), (Arikan (2009)). However, Vangala *et al.* (2015) have proved heuristically that Bhattacharyya construction method can be used for additive white Gaussian noise (AWGN) channels if the state of the channel is correctly preestablished.

Motivation

Current WCSs are designed to satisfy the instantaneous connectivity that human mind perceive without waiting times for improving the quality of user experience. However, near-future applications will be designed based on a M2M communication approach where the required latency is even more demanding. Examples of these applications are emergency disaster relief, military surveillance, e-health, cloud services, vehicle collision detection and avoidance, telesurgery, efficient industrial communication or environmental monitoring. Considering the evolution of these applications over the fifth generation (5G) of wireless networks, the International Telecommunication Union Radiocommunication sector has established in the recommendation ITU-R M.2083.0 that the International Mobile Telecommunications for 2020 (IMT-2020) should be able to provide at most 1 ms over-the-air latency (ITU-R (2015)), which represents a big challenge for all the telecommunication industry community.

Another valuable reason for decreasing the latency of a WCS is to contribute with the promotion of new information and communication technologies for rural areas where sensitive-delay applications are affected by latency due to long distance connections (Gagnon (2016)). It is specially applied in developing countries where it is necessary to deploy multihop networks employing microwave radio links because of the difficult access to these areas.

Problem overview

The *system latency* or end-to-end delay of a digital communication system (DCS) is defined as the time interval between the transmission and reception of a message $\mathbf{m}$.

It means that the system latency results from addition of delays at each phase of the DCS (summarized as transmission and reception delays) and the propagation delay imposed by the medium. Then, there are some phases in a DCS where could be possible to decrease the latency. Recently, Au & Gagnon (2016) have proposed a novel approach for decreasing the detection

latency of the receiver. This strategy consists of making early detections of the received signal in a sequential manner before the end of the symbol duration, given that a short message can be transmitted simultaneously through orthogonal-frequency-division-multiplexing (OFDM) signals. The early detection scheme can be performed by multi-hypothesis sequential probability ratio tests based on list-decoding or by sequential detections guided by error-detecting codes such as cyclic redundancy check (CRC) codes, (Au & Gagnon (2016)).

According to Au & Gagnon (2016) the sequential CRC-based early detection with 8-bits CRC and 16-bits CRC reduces the average latency while maintaining the block-error probability of a system without early detections. However, the proper early detection scheme configuration has not been found to obtain these results. Since too early detections in noisy channels generate a bad error performance, an initial detection time (IDT) must be set as part of the detection scheme configuration.

Objectives

The purpose of this research is to decrease the detection latency of a multicarrier communication system and maintain the error rate of a synchronous detection system by using an early detection scheme based on CRC-polar codes.

The specific objectives of this investigation are:

- Determine the best design signal-to-noise-ratio (SNR) for the construction of polar codes over AWGN channels.
- Analyze and establish CRC polynomials that minimize the average latency and block-error rate of an early detection scheme.
- Define the initial detection time of the sequential detection to optimize the performance of the early detection scheme under noisy channels.

- Analyze different detection distributions for the early detection scheme based on their statistical average latencies.

Hypothesis

Based on our objectives, we have the next hypothesis:

- The design-SNRs found by Vangala *et al.* (2015), for Bhattacharyya and Tal & Vardy's construction methods, provides the best block-error performance of polar codes.
- CRC polynomials published by Koopman & Chakravarty (2004) are suitable to minimize the average latency and block-error rate of the early detection scheme.
- The best initial detection time of the sequential detection is around 50% of the symbol period.
- The theoretical or statistical average detection latency proposed by Au & Gagnon (2016) increases as the symbol period decreases.

Assumptions

This work is under the following assumptions:

- The communication system works with a fixed code rate and does not have a channel status feedback.
- Message symbols are simultaneously transmitted in parallel over additive white Gaussian noise channels using orthogonal-frequency-division-multiplexing signals.
- The demodulation and decoding latency is much shorter than symbol duration. Otherwise, the sequential early detection is not feasible.

- OFDM signals are spectrally efficient while early detections are developed.

Methodology

This research begins with a study of the strategy for decreasing latency through sequential early detections, and a brief literature review of OFDM systems, polar codes and CRC codes. A communication system with polar codes and binary phase-shift keying (BPSK) modulation is simulated over an AWGN channel. Polar codes are created by Bhattacharyya and Tal & Vardy construction methods under a design parameter called *design-SNR*. Based on heuristic Vangala's algorithm, the resulting block-error rates of a set of possible design-SNRs are compared to determine the best design-SNR of these construction methods. A construction method of polar codes with its best design parameter is chosen for next stages of this work.

Based on Au & Gagnon's early detection strategy, a scheme of sequential early detections guided by concatenated CRC and polar codes is proposed for an OFDM system. The early detection scheme is simulated under a parallel transmission of BPSK symbols over AWGN channels. In order to determine a suitable setting of the early detection scheme, two selection processes are developed to define the CRC polynomial and the IDT. Before applying these selection processes, we set a scenario for performance comparisons. This scenario is established in terms of the priority of polar bit-channels, a specific polar bit-arrangement, an SNR range of interest, block lengths and code rates of polar codes. The selection processes consist of performance comparisons of block-error rates and average detection latencies from a set of pre-established CRC polynomials or IDTs. Forty CRC polynomials with different degrees and error-detection probabilities are considered for the first selection process, which are taken from Koopman (2016) and Cook (2016). The polynomial selection process chooses the CRC polynomials with the lowest block-error rate (BLER) and low average detection latencies for specific ranges of SNR. Expressly, three CRC polynomial are selected for low, medium and high ranges of SNR. In addition to these three polynomials, another CRC polynomial is taken

into account to analyze the behavior of IDTs under different CRC sizes. The second selection process determines an IDT for each CRC polynomial by making a trade-off between the error and latency performances of seven IDTs. Finally, three CRC polynomials with their respective IDTs are declared for the early detection scheme. Each pair of parameters provides the best possible error and latency performance for low, medium and high SNRs, respectively.

The resulting BLERs are compared with the error performance of a synchronous detection system, while the obtained average detection latencies are compared with the theoretical or statistical average detection latency proposed by Au & Gagnon (2016) in the finite-blocklength regime and using CRC-polar codes. To complete this work, three different detection distributions are compared through their statistical average latencies in the finite-blocklength regime and using CRC-polar codes.

Thesis contributions

The main contribution of this thesis is the proposal of heuristic procedures to establish the configuration parameters of a CRC-based sequential early detection scheme. Two comparison processes to select CRC polynomials and initial detection times of the sequential early detection scheme are proposed. Another contribution is the validation through simulations that a sequential early detection scheme, based on concatenated CRC-polar codes, is able to reduce the detection latency of a multicarrier communication system over AWGN channels. In addition, we determined that this early detection scheme is able to approximate to the error performance of a system without early detections by increasing the block length of the polar code while the polar code rate remains fixed.

Despite the significant reduction of latency achieved by the CRC-based early detection scheme, it does not achieve the optimal or theoretical latency in the finite-blocklength regime. This is due to the regular error performance of the successive-cancellation decoder used in this work.

In other words, the detection latency is significantly affected by the error performance of the used code. To approach to the optimal latency performance, we suggest for future works to use another type of polar decoder that offers a better error performance without neglecting its decoding speed. On the other side, it is verified that as the time interval between early detections decreases the average detection latency improves.

This work also corroborates Vangala's study, based on block-error rates. That is, for a specific range of signal-to-noise ratios (SNR), the Bhattacharyya construction method of polar codes can generate a good error performance over AWGN channels similar as Tal&Vardy construction method if the design-SNR parameter is optimized.

Thesis organization

This work is organized into three chapters, the first one is a literature review of the concepts managed in this thesis, the second chapter is focused on the selection of a design parameter to obtain good polar codes, and the last chapter details a process to determine the configuration parameters of the CRC-based early detection scheme proposed by Au & Gagnon (2016).

Chapter 1 presents how an OFDM system works, the latencies present in this system and the calculation of the average detection latency through a statistical process in the finite-blocklength regime. The sequential detection guided by CRC codes for decreasing latency in a multicarrier communication system, background of CRC codes, and a particular selection of CRC polynomials are also reviewed in this chapter. This chapter ends with a brief description of polar codes, their construction, encoding and successive cancellation decoding.

In chapter 2, Bhattacharyya and Tal&Vardy construction methods of polar codes over AWGN channels are introduced. Based on the search method of Vangala *et al.* (2015) and employing BLER performances, the best design-SNRs are determined for both polar code construc-

tion methods. Under the obtained design-SNRs, a comparison between Bhattacharyya and Tal&Vardy construction methods is developed.

Chapter 3 describes the proposed scheme of sequential early detections based on the concatenation of CRC and polar codes in an OFDM communication system over AWGN channels. A sufficient statistic is determined through a mathematical analysis to take decisions at the receiver. Two possible polar bit-arrangement formats are analyzed. The selection of CRC polynomials and initial detection times are performed to obtain the best possible setting of the CRC-based early detection scheme. Three possible time detection distributions are analyzed through their statistical average latencies. The resulting latency performances are compared with statistical average latencies in the finite-blocklength regime and using CRC-polar codes. The state of the art of decoding implementations is presented, to analyze if the latency reduction is possible in practice.

Finally, we present conclusions and recommendations of our work.

CHAPTER 1

THEORETICAL BACKGROUND

In this chapter, we present a brief review of the OFDM multicarrier communication system. Then, the latencies of a digital communication system (DCS) are defined, as well as the reception and detection latency of an OFDM system. Along with these definitions, we describe latency parameters used throughout this thesis for synchronous and asynchronous detections schemes. A novel idea for decreasing the detection latency on multicarrier communication systems is presented. Since this strategy is concentrated on decreasing the detection latency at the receiver, it is shown how to calculate the average detection latency based on the error probability as a function of the signal-to-noise-ratio. There are two strategies to reduce the detection latency, multi-hypothesis sequential ratio tests guided by list decoding and sequential detections guided by error-detecting codes. Considering that our work is focused on the second strategy, we review the CRC error-detecting code and how to chose CRC polynomials. Finally, the theory of polar codes is explained, this comprises their basic construction, encoding, and decoding processes.

1.1 OFDM transmission scheme

Multicarrier communications are used to achieve high data transmissions while is maintained a negligible inter-symbol interference (ISI). There are different ways of implementing multicarrier communications such as frequency division multiplexing (FDM), discrete multi-tone (DMT) and orthogonal frequency division multiplexing (OFDM). In OFDM, the information is transmitted in parallel form over L different subchannels centered at different orthogonal subcarrier frequencies. The orthogonal subcarrier signals are overlapped in frequency spectrum for improving the spectral efficiency and they are spaced by $\Delta f = 1/T$ to maintain the orthogonality during the symbol interval T. In other words, the total system bandwidth W is divided into $L = W/\Delta f$ subchannels of width Δf. The bandwidth of each subchannel Δf is chosen sufficiently small to ensure that each subchannel experience flat fading, so that ISI is

mitigated on each subchannel. Furthermore, OFDM transmissions add a guard interval called cyclic prefix (CP) to each transmitted OFDM symbol in order to completely eliminate the ISI.

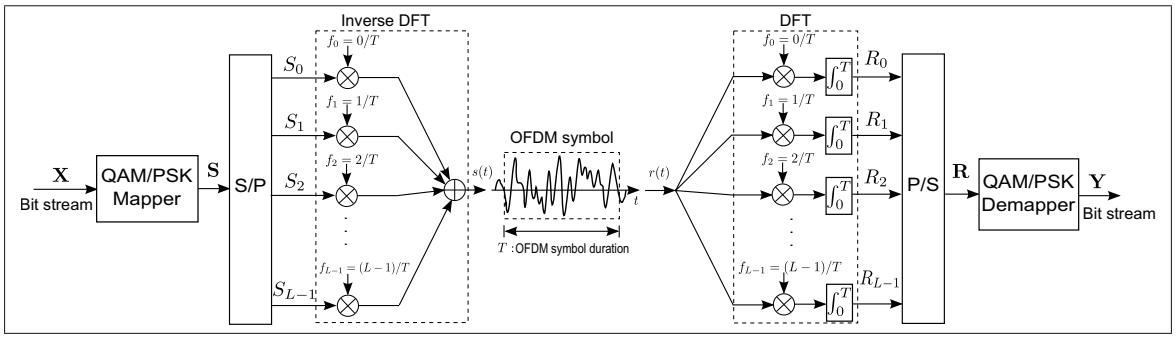


Figure 1.1 OFDM modulation/demodulation
Adapted from Cho *et al.* (2010)

For the construction of OFDM signals with L subchannels, a bank of $2L$ filters are required for the modulation and $2L$ matched filters or cross-correlators for the demodulation, (Proakis & Masoud (2008)). This implementation is relatively complex. The work developed by the demodulator and modulator is equivalent to the calculation of the discrete Fourier transform (DFT) and its inverse, respectively, see Figure 1.1. The DFT is computed by the fast Fourier transform (FFT) algorithm, thus the modulation and demodulation processes of an OFDM transmission scheme are efficiently implemented by the inverse fast Fourier transform (IFFT) and FFT algorithms, respectively.

The block diagram of the entire OFDM process employing FFT/IFFT algorithms is ilustrated in Figure 1.2. The bit stream $\mathbf{X}$ is the input of a quadrature amplitude modulation (QAM) or phase-shift keying (PSK) mapper. After the mapping process, a vector $\mathbf{S} = (S_1, S_2, ..., S_L)$ of complex symbols is obtained, which pass through a serial-to-parallel (S/P) converter. The L QAM/PSK symbols represent frequency-domain symbols of the OFDM output signal $s(t)$ that are transmitted over each subcarrier. Then, the IFFT algorithm is executed on these L symbols to convert frequency components into time samples $(s_1, s_2, ...s_L)$ of the signal $s(t)$. After time samples pass through the addition of a cyclic prefix, a parallel-to-serial (P/S) converter, and a digital-to-analog converter (DAC), a baseband OFDM signal $s(t)$ is obtained, which is

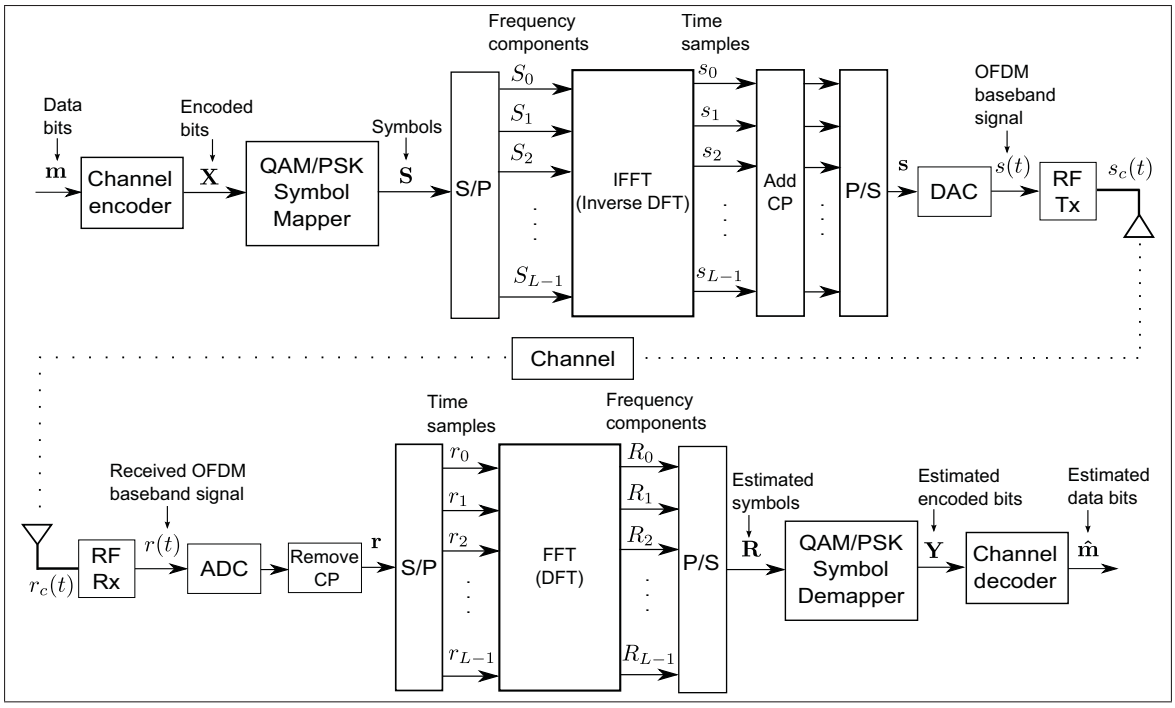


Figure 1.2 Transmitter and receiver of an OFDM system

expressed as

$$s(t) = \frac{1}{\sqrt{L}} \sum_{i=0}^{L-1} S_i e^{2\pi t(i/T)}, \qquad 0 \le t \le T, \tag{1.1}$$

where $1/\sqrt{L}$ is a scale factor, T is the duration of OFDM symbols, and the subcarrier frequencies are given by $f_i = i/T$, $i = 0, 1, ..., L-1$. Time samples $(s_1, s_2, ...s_L)$ are taken every T/L seconds, (Goldsmith (2005)). Therefore, L QAM/PSK symbols are simultaneously transmitted over each subcarrier with OFDM symbol duration T. The baseband OFDM signal $s(t)$ must be upconverted to a carrier frequency f_c for being transmitted over the channel, resulting in the transmitted signal $s_c(t)$. At the receiver, operations are performed to reverse the transmission and obtain the original sequence of data, as shown in Figure 1.2.

Also note that in OFDM transmission schemes where L symbols are transmitted in parallel, the symbol duration T_s of the original symbol X_i is extended to $T = LT_s$. In other words, the OFDM symbol is a composite signal of L symbols in a parallel form with duration T (Cho *et al.* (2010)), see Figure 1.1. The number of subchannels L is selected sufficiently large to make the

symbol time T significantly larger than the delay spread of the channel. This allows to obtain a small ISI in each subchannel, *i.e.* flat fading.

1.2 Latency of a digital communication system

The determination of the communication latency depends on the observed system. Take as reference the digital communication system at physical layer level, shown in Figure 1.3

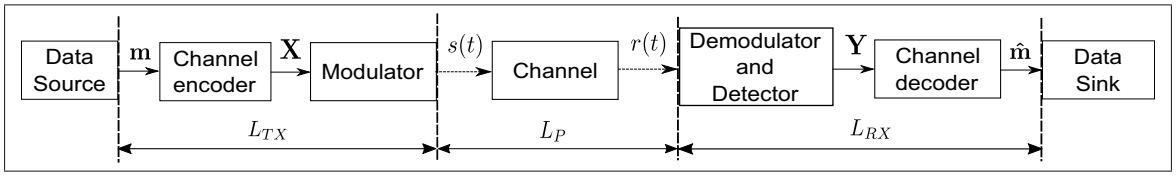


Figure 1.3 Digital communication system

The latency of a digital communication system or *system latency* is the time taken to deliver a message **m** across a physical medium from source to destination. The system latency can be referred also as *end-to-end delay* or *one-way delay*. The system latency is then calculated as

$$L_{DCS} = L_{TX} + L_P + L_{RX}. \tag{1.2}$$

where L_{TX}, L_{RX} and L_P denote the transmission latency, the reception latency and the propagation latency, respectively, see Figure 1.3.

The *propagation latency* L_P is the time duration to propagate a signal (message) on a physical medium from a transmitter to a receiver. It depends on the distance d between the communication nodes and the wave propagation speed s, as follows $L_P = d/s$. Note that in wireless communications, the propagation speed is better than in fiber-optic communications due to high refraction indices of optical fiber. (Kawanishi *et al.* (2012)). The *transmission latency* is the amount of time required by the transmitter to push out an entire message onto the channel, whereas the *reception latency* is the time length that the receiver needs to estimate the transmitted message.

1.3 Reception latency of an OFDM system

Taking into account the OFDM receiver shown in Figure 1.2, the reception latency of an OFDM system results from the sum of the processing delays of each element that constitutes the receiver and the received message latency, as follows

$$L_{OFDM\_RX} = L_{CONV} + L_{RXM} + L_{DEM} + L_{DECO}, \tag{1.3}$$

where L_{CONV} denotes the OFDM signaling down-conversion latency, L_{RXM} is the received message latency, L_{DEM} is the OFDM demodulation latency and L_{DECO} is the channel decoding latency. Note that OFDM signaling down-conversion latency encompasses the frequency down-conversion, the analog-to-digital conversion and the cyclix prefix removing operation; while the OFDM demodulation latency includes the FFT and demapping process.

The *received message latency* (L_{RXM}) is the time consumed by the analog-to-digital converter (ADC) to sample the received OFDM signal that carries a message (**m**), or is the time taken by the serial-to-parallel (S/P) converter to parallelize the samples that constitute a message (**m**) transmitted through an OFDM signal. Hence, the received message latency is equal to the OFDM symbol duration T in a common OFDM receiver (without early detections).

1.4 Detection latency of an OFDM system

Let us define the *total detection latency* of an OFDM system as the needed time to do a correct detection of the transmitted message counted from the beginning of the parallelization of the message samples. In other words, it is the time interval from the instant the first sample of the received OFDM baseband signal (**r**) is available at the S/P converter output until the transmitted message (**m**) is completely decoded or detected, see Figure 1.4. Therefore, the total detection latency is calculated as follows

$$L_{TOTAL\_DET} = L_{RXM} + L_{DEM} + L_{DECO}. \tag{1.4}$$

Given that the received message latency is equal to the OFDM symbol period ($L_{RXM} = T$) in a normal OFDM receiver, the received message latency is much longer than the processing delays L_{DEM} and L_{DECO}. That is, the total detection latency depends mainly on L_{RXM}. Due to this fact, we refer to the received message latency as the *detection latency*, which is denoted by τ. Therefore, the total detection latency from Equation 1.4 can be rewritten as

$$L_{TOTAL\_DET} = \tau + L_{DEM} + L_{DECO}. \tag{1.5}$$

From Equations 1.3 and 1.5, the detection latency (τ) is part of the reception latency. Consequently, an increment or decrement of the detection latency affects the reception latency and this in turn to the overall latency of a communication system.

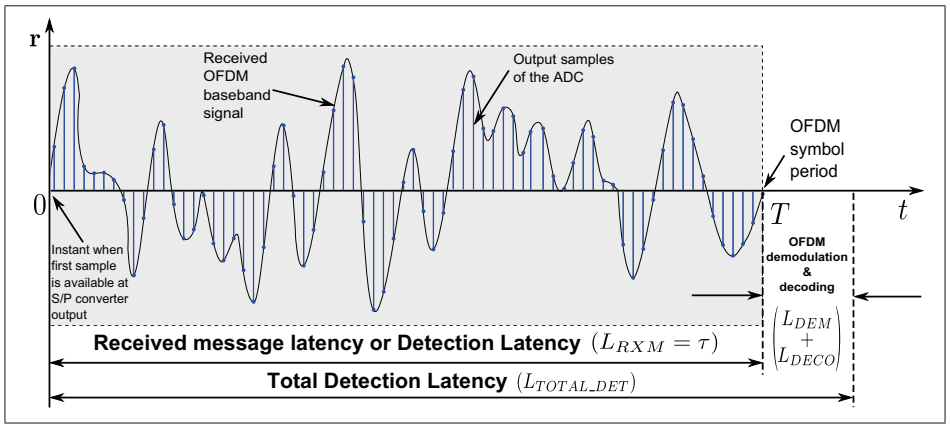


Figure 1.4 Latency parameters in an OFDM receiver with synchronous detections

The *detection latency* of an OFDM system is defined as the sufficient portion of the OFDM symbol period that allows the detection of the transmitted message after the demodulation and decoding processes, see Figure 1.5. The processing delays of the OFDM demodulator and channel decoder are considered in the calculation of the total detection latency as shown in Equation 1.5. Considering that $\tau = L_{RXM}$ and the detection latency starts the instant the first sample of the OFDM baseband signal is available at the S/P converter output, the *detection*

latency can also be defined as the the time used by the S/P converter to parallelize a set of OFDM signal samples that provide a correct detection of the message.

Note that the detection latency is less than or equal to the OFDM symbol period T, $(\tau \leq T)$. So that if $\tau = T$, the OFDM receiver performs *synchronous detections*, while if $\tau < T$, the OFDM receiver applies *early* or *asynchronous detections*.

1.5 How to decrease latency?

In order to decrease the system latency of a DCS, one of the delays described above can be reduced. Au & Gagnon (2016) have proposed to reduce the *detection latency* (τ) of a DCS by employing multicarrier communications, such as OFDM. The main idea is to develop *sequential early detections* ($\{\tau_1, \tau_2, \tau_3, ..., T\}$) of the message (**m**) implicitly transmitted through an OFDM signal. These early detections start from a preselected initial detection time (denoted by τ_1) until the end of the OFDM symbol duration T, see Figure 1.5.

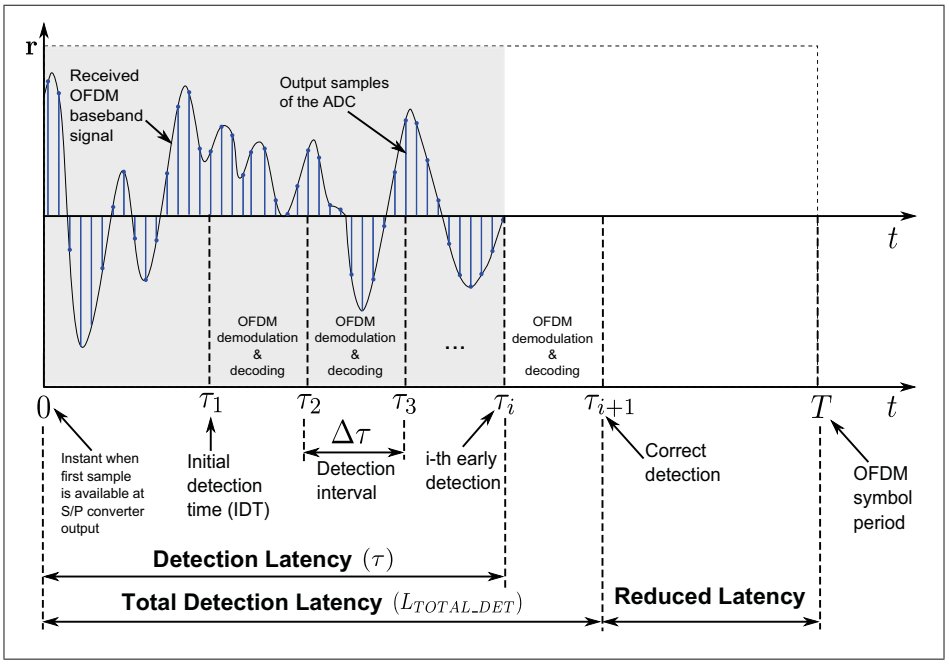


Figure 1.5 Latency parameters in an OFDM receiver with early or asynchoronous detections

It is expected that the transmitted message ($\mathbf{m}$) is correctly detected at an i-th early detection (τ_i) less than the OFDM symbol period T. The worst case is the message detection at $\tau = T$, since the latency is not reduced or is equal to the latency of synchronous detections. Advantageously, Au & Gagnon's results show that the expected detection latency of early detections ($\mathbb{E}_{\mathbf{m}}[\tau]$) is less than the OFDM symbol period T. An important parameter of the sequential detection is the detection interval ($\Delta\tau$) between early detections, which in practice depends on the processing delay of the OFDM demodulation and channel decoding, see Figure 1.5. More details of the sequential early detection are in Section 1.8.

Note that the detection latency can be decreased significantly with this method, given that a block of symbols is transmitted in parallel and its detection is performed simultaneously at the receiver. This means that Au & Gagnon's strategy takes advantage of implicit characteristics of multicarrier wireless systems. In a single carrier communication, symbols are received in serial, and for decoding a complete message it is necessary to wait until the reception of the last symbol. Consequently, the early detection method would not be a good idea to apply on single carrier systems.

Since the time to take decisions is reduced with early detections, the orthogonality of OFDM signals is not maintained. Recall that the minimum space between OFDM subcarriers should be $1/T$ for maintaining the orthogonality. Fortunately, Au & Gagnon (2016) show that this issue is overcome by using random coding or precoding random matrices. In other words, early detection schemes are feasible over OFDM systems.

1.6 Average detection latency

In order to obtain the *average detection latency* ($\bar{\tau}$) of a message $\mathbf{m}$ in a multicarrier communication, Au & Gagnon (2016) determine the *expectation of early detections* $\tau = \{\tau_1, \tau_2, ..., T\}$ as follows

$$\bar{\tau}_{\mathbf{m}} = E_{\mathbf{m}}[\tau] = \sum_{\tau_i=\tau_1}^{T} \tau_i p_{\tau}(\tau_i). \tag{1.6}$$

The probability mass function (PMF) of an early detection τ_i $(p_\tau(\tau_i))$ is interpreted as the probability of developing a correct detection of the received message when a previous detection τ_{i-1} was not successful. Considering that the average block-error probability of an optimal early detection scheme is given by $\varepsilon = 1 - \int_0^T p_\tau(\tau)d\tau$, the PMF at τ_i can be calculated as follows (Au & Gagnon (2016))

$$\begin{aligned} p_\tau(\tau_i) &= (1-\varepsilon(\tau_i)) - (1-\varepsilon(\tau_{i-1})) \\ &= \varepsilon(\tau_{i-1}) - \varepsilon(\tau_i) \end{aligned} \tag{1.7}$$

The expectation of early detections obtained with Equation (1.6) is for one message; however, there are M possible messages that could be sent with equal probability. Therefore, the *average detection latency* of all possible messages is determined by

$$\bar{\tau} = \frac{1}{M}\sum_{\mathbf{m}=1}^{M} \mathbb{E}_{\mathbf{m}}[\tau]. \tag{1.8}$$

In a DCS with an expected block-error probability under synchronous detections $\varepsilon(T)$, *i.e.* $\tau = T$, the total energy of the received signal $r_c(t)$ is used to take decisions. Consequently, the the received SNR per bit (E_b/N_0) is affected by the detection latency, or is a function of the detection latency $(SNR = f(\tau))$. An early detection $(\tau < T)$ implies receiving a portion of the symbol energy, which is equivalent as receiving a signal with lower SNR. On the other side, it is known that the block-error rate is a function of the received SNR, $\varepsilon = f(SNR)$. Therefore, the block-error probability is also a function of the detection latency, $\varepsilon = f(\tau)$. To better understand this, the following example shows three detection instants, which are denoted on Figure 1.6.

$$\begin{array}{llcccl} \text{If} & \tau = T & \rightarrow & SNR(T) = E_b/N_0 & \rightarrow & \varepsilon(T) \\ \text{at} & \tau = 3T/4 & \rightarrow & SNR(3T/4) = 3(E_b/N_0)/4 & \rightarrow & \varepsilon(3T/4) \\ \text{at} & \tau = T/2 & \rightarrow & SNR(T/2) = (E_b/N_0)/2 & \rightarrow & \varepsilon(T/2) \end{array}$$

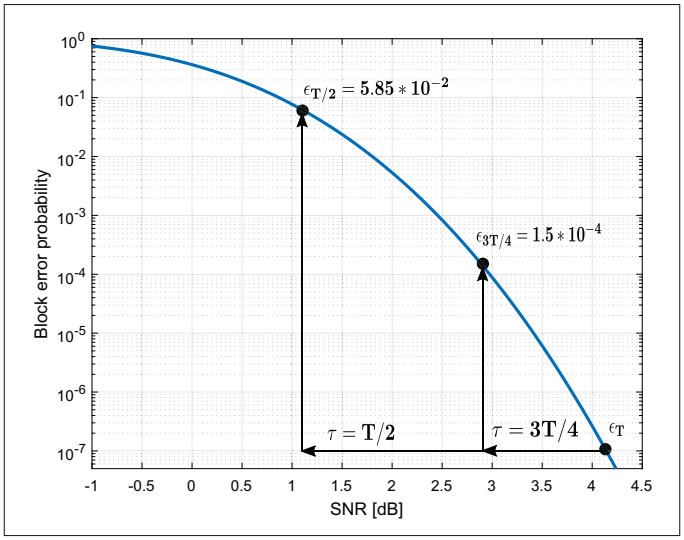


Figure 1.6 Block error rate in finite-blocklength regime for early detections $\tau = \{T/2, 3T/4, T\}$ with $R_c = 0.5$ [bits/channel use] and $N = 128$

According with the above explanation and considering the Equation (1.7), correct early detection probabilities can be calculated using the error performance (ε) as function of their respective SNRs (or detection times). As shown below

$$p(T/2) = 1 - \varepsilon(T/2) \tag{1.9a}$$

$$p(3T/4) = (1 - \varepsilon(3T/4)) - (1 - \varepsilon(T/2)) \tag{1.9b}$$

$$p(T) = (1 - \varepsilon(T)) - (1 - \varepsilon(3T/4)). \tag{1.9c}$$

In the finite-blocklength regime, for computing the average detection latency under asynchronous detections $\tau = \{T/2, 3T/4, T\}$, we take as reference an expected block-error probability for T. For this example, we assume $\varepsilon(T) = 1 * 10^{-7}$, see Figure 1.6. Using the Equation (1.10) of the block-error probability in the finite-blocklength regime, we compute $\varepsilon(T/2) = 5.85 * 10^{-2}$ and $\varepsilon(3T/4) = 1.5 * 10^{-4}$. Using the Equation (1.9) and the previous results, we calculate the PMF of early detections: $p(T/2) = 0.942$, $p(3T/4) = 5.83 * 10^{-2}$, and $p(T) = 1.5 * 10^{-4}$. Hence, by the expectation of early detections computed with Equation (1.6), it is necessary 51.47% of the symbol duration to detect correctly a message $\mathbf{m}$. From this

process, we conclude that the average detection latency for an expected error probability ε or specific SNR can be determined by the error probability of early detections or lower SNRs.

1.7 Error probability in the finite-blocklength regime

In the finite-blocklength regime, let us consider an arbitrary error correcting (N,M,ε) code implemented over a DCS with a *single carrier*. This code has a block length N, M possible codewords, and an average error probability less than ε. The number of information bits encoded in a codeword is calculated as $k = \log_2 M$, which is called as *code size*.

Polyanskiy *et al.* (2010) determine the maximal code size $\log_2 M^*$ and the maximal code rate R_c^* achievable with a block length N and an error probability ε. Based on the normal approximation of these results over AWGN channels, Au & Gagnon calculate the performance of (N,M,ε) codes in terms of the *block-error rate* for a given code rate R_c, an SNR ρ, and a block length N, with the following equation

$$\varepsilon^*(\rho,R_c,N) = Q\left(\frac{C - R_c + \frac{1}{2N}\log_2(N)}{\sqrt{V/N}}\right), \tag{1.10}$$

where Q-function denotes the complement of the standard normal distribution , C is the channel capacity in [*bits/channel use*] defined as $C = \frac{1}{2}\log_2(1+\rho)$ and V is the channel dispersion, which is calculated as

$$V = \frac{\rho}{2}\frac{\rho+2}{(\rho+1)^2}(\log_2 e)^2. \tag{1.11}$$

The channel dispersion measures the variability of the channel relative to a deterministic channel with the same capacity (Polyanskiy *et al.* (2010)).

1.8 Sequential early detection

The sequential early detection is a technique proposed by Au & Gagnon (2016) in order to decrease the *average detection latency* in a multicarrier system. It consists of doing progressive tests to know if the sent message **m** is correctly detected at the receiver. Sequential early detections can be represented as a random variable $\tau = \{\tau_1, \tau_2, ..., T\}$, where i-th detection time is denoted as τ_i. The evaluation of the received OFDM signal $r(t)$ or received symbols $(R_0, R_1, ..., R_{L-1})$ is performed at progressive detection times until it is correctly detected or the detection time τ_i reachs the end of the symbol interval T, see Figure 1.7. This means that sequential early detection is successful if $\tau_i < T$ and it is optimum when the detection time τ_i is as small as possible without exceeding an established error probability ε.

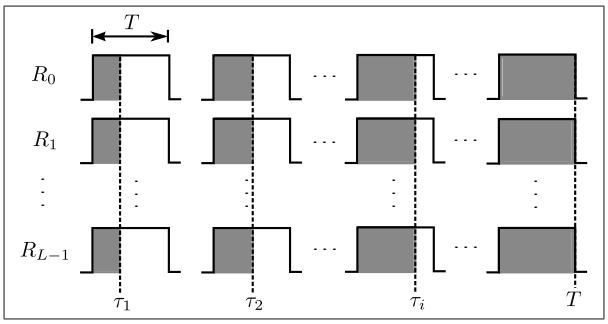


Figure 1.7 Sequential early detection

Au & Gagnon (2016) consider two *sequential detection schemes* for decreasing latency. A multihypothesis sequential probability ratio test (MSPRT) guided by a list decoder and a sequential detection guided by error-detecting codes.

1.8.1 Multi-hypothesis sequential ratio test guided by list decoding

Basically, MSPRT selects the transmitted message **m** among M possible messages as soon as the probability of its correct detection exceeds a threshold γ_m. One challenge of this sequential test is the determination of an optimal threshold γ_m given that it depends on the channel conditions that must be known at the receiver. Moreover, this threshold affects the error rate and

the achievable latency. Then γ_m should be correctly designed in order to achieve the minimum latency without exceeding a predefined error rate ε. On the other side, recall that $M = 2^k$, where k is the information block size. Then 2^k tests are necessary to decide which message has the largest probability. Therefore, when the information block size is large (*e.g.* $k = 100$) the implementation of MSPRT is not tractable. A *list decoder* is employed for reducing the number of hypothesis in sequential tests by giving a list l of the most probable messages, where $l < M$. In a MSPRT+list decoding scheme, it is necessary to determine a trade-off between the list size l and the achievable latency, since as l increases the average latency $(\bar{\tau})$ decreases, but the error rate also increases.

1.8.2 Sequential detection guided by error-detecting codes

The second option for decreasing the average detection latency under sequential detections is employing *error-detecting codes*. In this sequential detection scheme, the error-detecting code acts as a *'genie'* that identifies if the detected message is correct or not at certain detection time τ_i. Sequential tests using error-detecting codes are executed until there is not error on data, namely as soon as the decoder has not found errors in the received message. An error-detecting code commonly used is the *Cyclic-redundancy-check* (CRC) code. Error-detecting CRC codes are likewise used as outer codes to improve the error correction performance of inner codes. For instance, when list decoders are employed by inner codes such as turbo codes or polar codes (Narayanan & Stuber (1998); Tal & Vardy (2015)), the CRC code is used to choose a valid path from the generated list. With the purpose to recover an erroneous message instead of to discard it, Wang *et al.* (2008) proposes an iterative CRC-assisted decoding of convolutional codes.

A concern related with this CRC-based scheme is that too early detections produce a bad error performance over noisy channels. This issue will be managed in this project. Moreover, Au & Gagnon (2016) show that under an optimized CRC-based early detection scheme the average detection latency $(\bar{\tau})$ is decreased while the block-error probability is the same of

synchronous detections. In this work, we will try to determine the best possible setting to obtain the mentioned results.

1.9 Cyclic-redundancy-check codes

Cyclic-redundancy-check codes are a class of cyclic error-detecting codes, commonly used in digital networks. Note that cyclic codes are a kind of linear block codes. Basically, the CRC creates check-bits, which are appended to the message bits or data word. At the reception, it is checked whether or not the redundant bits agree with the received data. A detailed explanation of the CRC method is given next.

Let us assume an (n,k) CRC code with k message bits and a block length n. So that there are $n-k$ bits of redundancy, known as *checksum*, *check sequence* or *CRC*. The cyclic-redundancy-check employs polynomial arithmetic in Galois Field (GF) of two elements $\{0,1\}$. Message bits are treated as coefficients of a polynomial $\mathbf{U}(X)$ with degree $k-1$, as follows

$$\mathbf{U}(X) = u_{k-1}X^{k-1} + u_{k-2}X^{k-2} + ... + u_1X + u_0, \tag{1.12}$$

namely $\mathbf{U} = (u_{k-1}, u_{k-2}, ..., u_1, u_0)$ represent the k information bits. Under cyclic codes theory proposed by Peterson & Brown (1961), CRC codes are constructed by a *generator polynomial* $\mathbf{G}(X)$ of degree $n-k$ and a division of polynomials, as follows

$$\frac{\mathbf{F}(X)}{\mathbf{G}(X)} = \mathbf{Q}(X) + \frac{\mathbf{R}(X)}{\mathbf{G}(X)}, \tag{1.13}$$

where $\mathbf{F}(X)$ is the dividend, $\mathbf{Q}(X)$ is the quotient and $\mathbf{R}(X)$ is the remainder. The CRC encoder creates a block polynomial $\mathbf{F}(X)$ based on $\mathbf{U}(X)$ and $\mathbf{G}(X)$, such that $\mathbf{F}(X)$ is divisible by $\mathbf{G}(X)$. In other words, the remainder $\mathbf{R}(X)$ is equal to zero in Equation (1.13). This is achieved by multiplying the message $\mathbf{U}(X)$ by the factor X^{n-k}, as shown below

$$\mathbf{F}(X) = X^{n-k}\mathbf{U}(X). \tag{1.14}$$

The polynomial multiplication in (1.14) represents appending $n-k$ 0-bits to the k-bit message. After $\mathbf{F}(X)$ is obtained, the encoder performs the polynomial division over GF(2) (or modulo-2 division) shown in Equation (1.13). The quotient $\mathbf{Q}(X)$ is ignored and the remainder $\mathbf{R}(X)$ becomes the result. The $n-k$ coefficients of the remainder polynomial $\mathbf{R}(X)$ constitute the *checksum*. The remainder polynomial $\mathbf{R}(X)$ has a degree less than $n-k$. The output of the CRC encoder results in the original k-bit message followed by the $n-k$ redundancy bits, *i.e.* $X^{n-k}\mathbf{U}(X)+\mathbf{R}(X)$. Take into account that the generator polynomial $\mathbf{G}(X)$ is fixed for a given CRC scheme and it must be known by both encoder and decoder. Moreover, note that the degree of the generator polynomial defines the size of the CRC or check sequence.

The CRC decoder verifies the correctness of the transmission by repeating the calculation (1.13). The received codeword represented by polynomial $\hat{\mathbf{F}}(X)$ is divided by the generator polynomial $\mathbf{G}(X)$. If the remainder polynomial $\mathbf{R}(X)$ (or all its $n-k$ coefficients) is zero, then the received message $\hat{\mathbf{F}}(X)$ is accepted as the one which was transmitted otherwise $\hat{\mathbf{F}}(X)$ has errors. In other words, when $\hat{\mathbf{F}}(X)$ is not divisible by $\mathbf{G}(X)$, ($\mathbf{R}(X)\neq 0$), an error has occurred.

It is evident that CRC method for detecting errors is not foolproof. If the transmitted message is garbled across the communication channel, there is a possibility that the new version of the message $\hat{\mathbf{F}}(X)$ is divisible by the generator polynomial $\mathbf{G}(X)$, although it is not right. Therefore, when $\hat{\mathbf{F}}(X)$ is divisible by $\mathbf{G}(X)$, ($\mathbf{R}(X)=0$), either no error or undetectable error has occurred. The received encoded message with errors can be represented by

$$\hat{\mathbf{F}}(X)=\mathbf{F}(X)+\mathbf{E}(X), \tag{1.15}$$

where $\mathbf{F}(X)$ is the correct encoded message and $\mathbf{E}(X)$ is the error polynomial with nonzero terms in erroneous positions. $\mathbf{E}(X)$ is detectable if and only if it is not divisible by the generator polynomial $\mathbf{G}(X)$. Hence, the generator polynomial should be carefully chosen to ensure that $\mathbf{E}(X)/\mathbf{G}(X)$ gives a remainder different of zero, for errors we wish to detect. According with the form of the generator polynomial $\mathbf{G}(X)$, it is possible to detect different types of errors

such as single, double, triple, odd or burst errors. For example if $\mathbf{G}(X)$ contains a factor $1+X$, then any single errors or any odd number of errors will be detected.

The implementation of CRC codes is relatively simple using shift registers with feedback connections and modulo-2 adders or exclusive-OR operations, (Sklar (2001)).

1.10 Selection of CRC polynomials

Considering that the size of the CRC or check sequence is equal to the degree of the generator polynomial $(r = n-k)$ of an (n,k) CRC code, we refer to this code as CRC-r. The normal representation of a CRC polynomial is the binary or hexadecimal notation of its coefficients. For example the CRC-5/ITU uses the CRC polynomial $X^5+X^4+X^2+1$, which is represented by 110101 or 0x15 in binary or hexadecimal notation, respectively. Alternatively, the CRC polynomial representation can omit the coefficient of the term X^0, since it is always 1. This representation is proposed by Koopman (2002). Hence, the polynomial of the example also can be denoted as 11010 or 0x1A.

It is well known that different cyclic redundancy checks are employed in technical standards. For the same CRC size there are different *CRC polynomials* or generator polynomials $\mathbf{G}(X)$, Cook (2016). For instance, there are three, fourteen, and twenty-eight CRC polynomials reported for CRC-5, CRC-8 and CRC-12, respectively. Considering that there is a variety of published CRC polynomials, now the question is which polynomial do we choose?

Koopman & Chakravarty (2004) describe how to select good CRC polynomials based on the desired Hamming distance, the data word length and the desired CRC size. They have found there are other CRC polynomials that provide a better error detection capacity. The performance of CRC polynomials is evaluated in terms of the probability of undetected errors (P_{ud}) under an assumed random independent bit-error rate (BER). The best achievable performance is the lowest bound computed by an exhaustive search of all polynomials for each data word length assuming a BER of 10^{-6} as shown in Figure 1.8. The CRC polynomial performance depends on its Hamming weight. *Hamming weight* (HW) is the number of undetectable errors for

a given number of bit-errors, *e.g.* 1267 undetectable 2-bit errors. The undetected error probability decreases when the Hamming distance increases as we see in Figure 1.8; accordingly, for the selection of the CRC polynomial it is necessary to maintain high Hamming distance values. *Hamming distance* (HD) is defined by Koopman & Chakravarty (2004) as the minimum number of bit-errors that is undetectable, *e.g.* assume a polynomial with Hamming weights: 0 for 1-bit error, 0 for 2-bit errors, 452 for 3-bit errors, 0 for 4-bit errors and 356 for 5-bit errors; then HD = 3, since all 1 and 2-bit errors are detectable. In other words, the Hamming distance of a CRC polynomial is given by the first non-zero Hamming weigth.

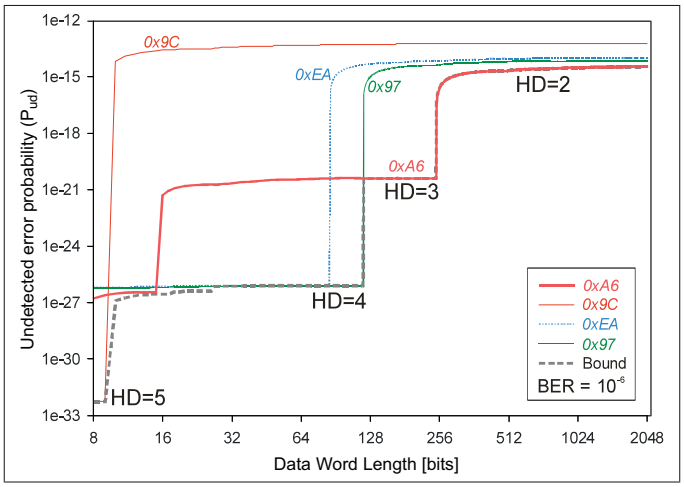


Figure 1.8 Performance of CRC-8 polynomials
Adapted from Koopman & Chakravarty (2004)

As shown in Figure 1.8, the performance of each CRC-8 polynomial is different and depends on the employed data word length. The performance level of each CRC polynomial in specific range of lengths is associated with a Hamming distance and tries to approximate to the optimal bound. In fact, most of the polynomials obtain good performance, but only under a specific range of data word lengths. For example, the polynomial 0x97 has a better performance than 0xEA in the range of lengths from 86 to 119. The polynomial 0x97 provides a HD = 4 whereas 0xEA has a HD = 2. Moreover, for lengths larger than 119, although 0x97 provides a HD = 2, it has a lower undetected error probability than 0xEA.

On the other side, the new polynomial 0xA6 proposed by Koopman & Chakravarty (2004) has a good performance in lengths between 120 and 247 with HD = 3. This means that the polynomial 0xA6 attains the breakpoint in the bound, where the HD changes from 3 to 4. However, for lower lengths between 15 and 119 the performance is not optimal because the Hamming distance is maintained (HD=3) whereas the polynomial 0x97 provides a better HD = 4. Besides, the error detection in 0xA6 is better than in 0x97 for messages longer than 247, it achieves the lowest bound. The good performance of the polynomial 0xA6 might compensate its increase of undetected error rate for short messages. Consequently, an strategy to choose a good CRC polynomial could be to select a polynomial with a good performance for long messages at the expense of a regular performance for short messages, or vice versa. This last decision implies that the selection of CRC polynomial also depends on the application, whether the application uses long or short messages.

The generic selection process of good CRC polynomials proposed by Koopman & Chakravarty (2004), under an evaluation in terms of the probability of undetected errors use the following guidelines. First, achieve maximum Hamming distance for the longest possible data word length. Second, obtain good performance at shorter lengths, and third, try to attain good performance at longer lengths than the stated maximum usage length, as a safety net. The result of this selection process are "good" CRC polynomials candidates which are summarized in tables published in (Koopman (2016)). In each cell of the tables, the maximum data word length is indicated at a specific HD, and the "good" CRC polynomial that gives at least the specific HD up to the indicated data word length. Depending on the application, these tables can be used in two ways. For finding a "good" polynomial (which provides the best HD) given the data word length and CRC size, or for determining the minimum CRC size required to achieve a given Hamming distance at a specific data word length.

1.11 Polar Codes

Polar codes are based on the phenomenon of channel polarization, which is developed by recursively combining and splitting individual channels. Some channels become noiseless or

perfect while others turn into completely noisy or useless. As the codelength N increases, a fraction of reliable channels approaches to the channel capacity. Thus, the data are sent through the reliable channels, Arikan (2009).

1.12 Channel polarization

Let be $W : \mathscr{X} \rightarrow \mathscr{Y}$ a binary discrete memoryless channel (B-DMC), with input alphabet $\mathscr{X} = \{0,1\}$, output alphabet $\mathscr{Y}$=$\{$arbitrary$\}$, and the probability of observing y given that x was transmitted defined as the *transition probability* $W(y|x) \stackrel{\Delta}{=} P(y|x)$, where $x \in \mathscr{X}$ and $y \in \mathscr{Y}$. Since the channel W is symmetric, such as a binary symmetric channel (BSC) or a binary erasure channel (BEC), the symmetric capacity $I(W)$ is equal to the Shannon capacity (Arikan (2009)). Symmetric capacity $I(W)$ is the highest rate over W with reliable communication using inputs with equal probability, it takes values inside [0,1] and is given by

$$I(W) = \sum_{y \in \mathscr{Y}} \sum_{x \in \mathscr{X}} \frac{1}{2} W(y|x) log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)}. \tag{1.16}$$

Channel polarization consists of converting n independent copies of B-DMC W into a polarized channel set $W_N^{(i)}$, where $0 \leq i \leq N-1$. Each polarized channel becomes either noisy or noiseless as the codelength N goes to infinity and the symmetric capacity terms $I(W_N^{(i)})$ tend towards 0 or 1, respectively. Thus, a fraction of noiseless channels goes to $I(W)$ and a fraction of noisy channels goes to $1 - I(W)$. Information bits are sent through the noiseless bit-channels in order to achieve the symmetric capacity of B-DMC, and over the noisy bit-channels are fixed values, which are known by the transmitter and the receiver. The channel polarization is divided in two stages, a channel combining and a channel splitting, as is depicted in Figure 1.9a.

1.12.1 Channel Combining

In this stage, N copies of B-DMC W, denoted as W^N, are combined in a recursive manner to produce a synthesized vector channel $W_N : \mathscr{X}^N \rightarrow \mathscr{Y}^N$, where $n = 2^p$, $p \geq 0$. The first level of

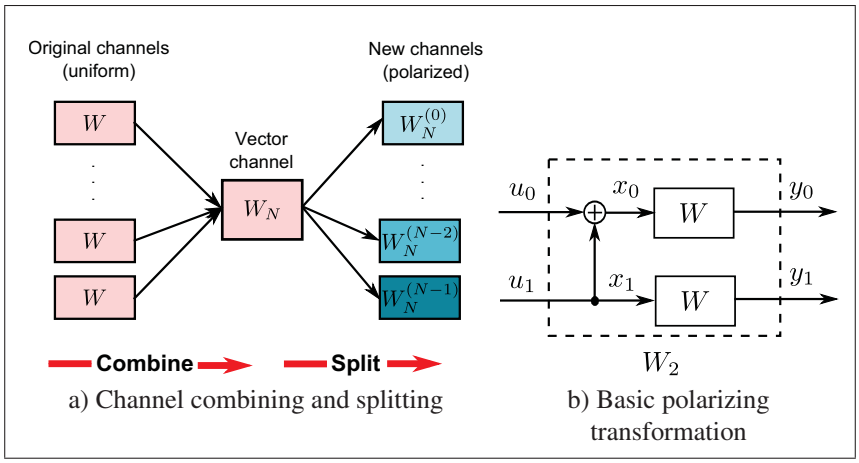


Figure 1.9 Channel Polarization
Adapted from Arikan (2009)

the recursion is $W_2 : \mathscr{X}^2 \to \mathscr{Y}^2$ as shown in Figure 1.9b, with transition probability

$$W_2(y_0, y_1|u_0, u_1) = W(y_0|u_0 \oplus u_1)W(y_1|u_1). \tag{1.17}$$

In general, the transition probability of the combining operation $(W^N \longrightarrow W_N)$ is

$$W_N(y_0^{N-1}|u_0^{N-1}) = W^N(y_0^{N-1}|u_0^{N-1}G_N), \tag{1.18}$$

where G_N is the *generator matrix* of size N, which represents a linear mapping from the input vector u_0^{N-1} to the output vector x_0^{N-1}, so that $x_0^{N-1} = u_0^{N-1}G_N$.

1.12.2 Channel Splitting

In a second step of the process, the synthesized channel W_N is split into a set of N polarized *bit-channnels* $W_N^{(i)} : \mathscr{X} \to \mathscr{Y}^N \times \mathscr{X}^{i-1}$, $1 \le i \le N$. The transition probability of the splitting operation $(W_N \longrightarrow W_N^{(i)})$ is determined as

$$W_N^{(i)}(y_0^{N-1}, u_0^{i-1}|u_i) = \sum_{u_{i+1}^N \in \mathscr{X}^{N-i}} \frac{1}{2^{N-1}} W_N(y_0^{N-1}|u_0^{N-1}), \tag{1.19}$$

where u_i is the input and (y_0^{N-1}, u_0^{i-1}) is the output of $W_N^{(i)}$. It means, at the successive cancellation decoder, that the estimation of the input-bit u_i is carry out after observing y_0^{N-1} and the past inputs u_0^{i-1}.

1.12.3 Recursive channel transformation

It is important to consider that bit-channels can be constructed recursively using the single-step channel transformations W^+ and W^-. These polarized bit-channels, defined as $W^- : \mathscr{X} \to \mathscr{Y}^2$ and $W^+ : \mathscr{X} \to \mathscr{Y}^2 \times \mathscr{X}$, come from the transformation of two independent copies of a binary-input channel $W : \mathscr{X} \to \mathscr{Y}$. This basic transformation is expressed as $(W,W) \longrightarrow (W^-, W^+)$ or $(W,W) \longrightarrow (W_2^{(1)}, W_2^{(2)})$, with their respective transition probabilities

$$W^-(y_0, y_1|u_0) = \frac{1}{2} \sum_{u_1 \in \mathscr{X}} W(y_0|u_0 \oplus u_1) W(y_1|u_1), \tag{1.20a}$$

$$W^+(y_0, y_1, u_0|u_1) = \frac{1}{2} W(y_0|u_0 \oplus u_1) W(y_1|u_1), \tag{1.20b}$$

where, W^- is "worse" than W, then $I(W^-) \leq I(W)$ and W^+ is "better" than W, then $I(W^+) \geq I(W)$; consequently, $I(W^-) \leq I(W) \leq I(W^+)$. Moreover, this basic channel transformation conserves the channel capacity, then $I(W^-) + I(W^+) = 2I(W)$.

1.13 Construction of polar codes

Although the construction phase of polar codes is treated separately from encoding, note that the construction phase is part of the encoder, as we can see in Figure 1.10. The construction of a polar code (N,K) of block length N and *dimension* K consists in the selection of the "best" (noiseless) K *bit-channels* among N, where $N = 2^n$ for some $n \geq 0$. The code rate is K/N. The selection of the K "best" bit-channels $W_N^{(i)}$ involves the localization of the *information bits* $(u_{\mathscr{I}})$ in the input vector u_0^{N-1} shown in Figure 1.10. The subset of indices $\mathscr{I} \subseteq \{0, 2, ...N-1\}$ are called *non-frozen bit indices*. In the remaining $N-K$ *non-reliable* bit locations of the vector

u_0^{N-1}, *frozen bits* ($u_{\mathscr{J}^c}$) are placed with values equal to '0'. The complement subset of indices $\mathscr{J}^c$ are known as *frozen bits indices*.

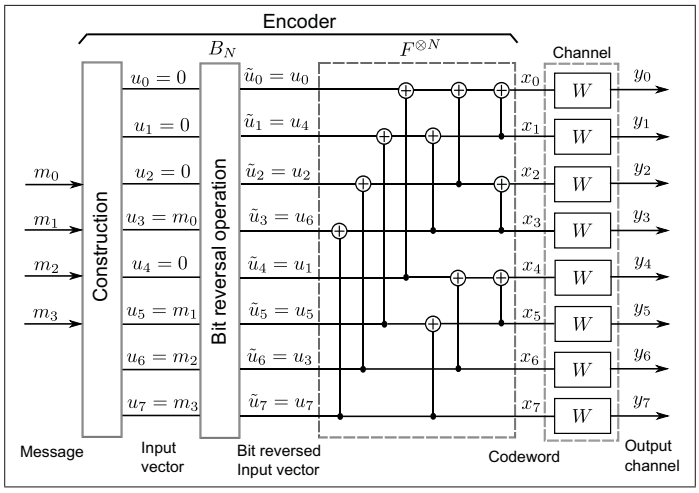


Figure 1.10 Polar Coding: $N = 8$, $K = 4$.

According to Arikan (2009), the bit-channel choice depends on the channel conditions, so that the K bit-channels $W_N^{(i)}$ with the lowest Bhattacharyya bound $Z_N^{(i)}$ on the probability of decision error ($P_e^{(i)}$) are chosen, since they are the noiseless channels. Bhattacharyya parameter is an upper bound on the probability of maximum-likelihood (ML) decision error, namely a measure of channel error performance is considered. It takes values in [0,1] and it is given by

$$Z(W) = \sum_{y \in \mathscr{Y}} \sqrt{W(y|0)W(y|1)}. \tag{1.21}$$

In a B-DMC the symmetric channel capacity $I(W)$ is related with the Bhattacharyya $Z(W)$ constant through the following expressions

$$I(W) \geq log\frac{2}{1+Z(W)}, \quad I(W) \leq \sqrt{1-Z(W)^2}. \tag{1.22}$$

Therefore, as $N \to \infty$, $I(W) \approx 1$ (noiseless channel) iff $Z(W) \approx 0$, and $I(W) \approx 0$ (unreliable channel) iff $Z(W) \approx 1$. The Bhattacharyya parameter for a single step transformation is de-

fined as $Z(W^+) = Z(W)^2$ and $Z(W^-) \leq 2Z(W) - Z(W)^2$. Consequently, the Bhattacharyya parameter of bit-channels $Z_N^{(i)}$ are calculated recursively as

$$\begin{aligned} Z_N^{(2j)} &= \left(Z_{N/2}^{(j)}\right)^2, \\ Z_N^{(2j-1)} &= 2Z_{N/2}^{(j)} - \left(Z_{N/2}^{(j)}\right)^2, \end{aligned} \tag{1.23}$$

where the initial value $Z_1^{(1)}$ is the erasure probability ε of the binary erasure channel. Note that Arikan's construction is only explicit and efficient for binary erasure channels, *i.e.* $Z_N^{(i)}$ are calculated exactly. The time complexity of Bhattacharyya-based construction is $O(N)$.

A drawback with the recursive construction of polar codes is that the output alphabet of the bit-channels grows exponentially with the block length. Therefore, the exact estimation of the bit-channels is intractable in practice. That is why there are several works proposing to approximate bit-channels for improving the polar code construction. Arikan (2009) proposed a Monte-Carlo estimation of bit-channels, which can be applied to different type of channels, (not only BEC). However, it has a greater complexity $O(MNlog_2N)$ among all, where M is the number of Monte Carlo iterations that restrict the construction accuracy (Vangala *et al.* (2015)). The determination of the noiseless bit-channels is based on their bit-error rate rather than their Bhattacharyya parameters.

Another construction for the estimation of bit-channels is one based on Gaussian approximations proposed by Trifonov (2012). Similar methods are presented by Li & Yuan (2013) and Wu *et al.* (2014). Polar codes constructed with these proposals have almost the same performance compared with Monte Carlo method of Arikan (2009). The complexity of the Gaussian approximation algorithm is $O(N)$, (Vangala *et al.* (2015)).

The best construction of polar codes available in literature is known by the name of its authors, Tal & Vardy (2013) construction. It is based on the quantization operation for estimating upper and lower bounds on the error probability of each bit-channel. The time complexity of this construction method is $O(N\mu^2 log_2\mu)$, where μ represents the number of quantization

symbols. It is the second largest complexity after Monte Carlo construction, (Vangala *et al.* (2015)).

1.14 Polar Encoder

Based on the channel combining process (Arikan (2009)), the encoding operation (represented in Figure 1.10) uses the *generator matrix* G_N to encode the input vector u_0^{N-1} and obtain the codeword

$$x_0^{N-1} = u_0^{N-1} G_N, \tag{1.24}$$

where $G_N = B_N F^{\otimes n}$, B_N is a permutation matrix (*bit-reversal*), $F^{\otimes n}$ is the Kronecker power defined as $F^{\otimes n} = F \otimes F \otimes ...$ (n times), and the polarization kernel matrix is

$$F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{1.25}$$

However, to simplify actual implementations of polar codes, it is better to use $F^{\otimes n}$ instead of $B_N F^{\otimes n}$. This implies that the successive-cancellation decoder should decode the source vector u_0^{N-1} in bit-reversed index order.

Alternatively, a polar codeword can be generated as a G_N-*coset code*:

$$x_1^N = u_{\mathscr{I}} G_N(\mathscr{I}) \oplus u_{\mathscr{I}^c} G_N(\mathscr{I}^c), \tag{1.26}$$

where $G_N(\mathscr{I})$ and $G_N(\mathscr{I}^c)$ are submatrices of G_N formed by the rows with indices in $\mathscr{I}$ and $\mathscr{I}^c$, respectively. Note that the second term $u_{\mathscr{I}^c} G_N(\mathscr{I}^c)$ is a fixed vector. Polar codes represented as a G_N-coset code can be identified by the parameter vector $(N, K, \mathscr{I}, u_{\mathscr{I}^c})$. The SC decoder has a low encoding complexity of $O(N log_2 N)$.

The permutation matrix B_N is a *bit-reversal* operator that inverts the bits position b_j of the binary representation of an input vector index u_{b_j}. Thus $\tilde{u}_{b_N \ldots b_1} = u_{b_1 \ldots b_N}$, after the operation $\tilde{u}_0^{N-1} = u_0^{N-1} B_N$, for all $b_j \in \{0,1\}$. Bit-reversed indexing is shown in Figure 1.10.

1.15 Successive-cancellation decoder

Arikan (2009) proposed a non-systematic successive-cancellation (SC) decoder with low decoding complexity, which is also a basic structure for more advanced and efficient decoders. The function of the decoder is to generate an estimate $\hat{u}_0^{N-1}$ of u_0^{N-1}, given knowledge of frozen bits ($u_{\mathscr{J}^c}$) and observing the vector y_0^{N-1}. The decoder considers likelihood ratios (LRs) $L_N^{(i)}$ to generate a decision. If the estimated element $\hat{u}_i$ is frozen ($i \in \mathscr{J}^c$), then $\hat{u}_i = u_i$, otherwise

$$\hat{u}_i = \begin{cases} 0, & \text{if } L_N^{(i)}(y_0^{N-1}, \hat{u}_0^{i-1}) \geq 1 \\ 1, & \text{otherwise} \end{cases} \tag{1.27}$$

where the likelihood ratio is defined as

$$L_N^{(i)}(y_0^{N-1}, \hat{u}_0^{i-1}) = \frac{W_N^{(i)}(y_0^{N-1}, \hat{u}_0^{i-1} | u_i = 0)}{W_N^{(i)}(y_0^{N-1}, \hat{u}_0^{i-1} | u_i = 1)} \tag{1.28}$$

The algorithm of the SC decoder is two-way recursive where *estimated bits* $\hat{u}_0^{N-1}$ are computed sequentially, one bit at a time. The decoding starts with the calculation of the likelihood ratios of the channel observations y_0^{N-1}, as follows

$$L_i(y_i) = \Lambda_i(y_i) = \frac{W(y_i|0)}{W(y_i|1)}. \tag{1.29}$$

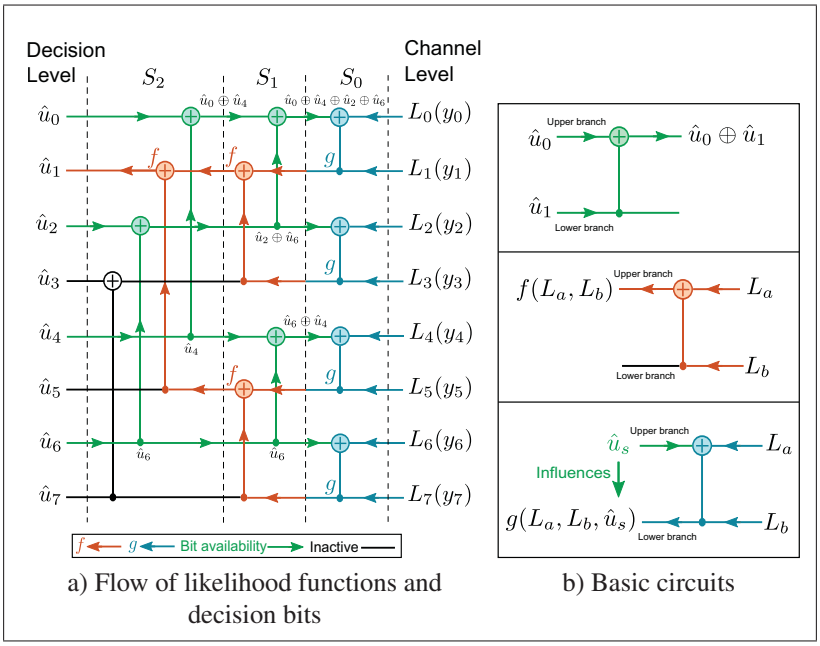
Then, likelihoods are computed through the circuits composed of XOR $\oplus$ operators, from the right to the left on the graph shown in Figure 1.11a. Once the likelihood is achieved at the left end of the graph a decision is made and it is spread from left to right to actualize the bits on the rest of the graph, (Vangala *et al.* (2014)). Note that the decoding order of bits is given by

the *bit-reversal* order if it was not previously applied the permutation matrix B_N in the encoder. Likelihoods inputs (L_a, L_b) of the basic circuit, shown in Figure 1.11b, are transformed through the functions f and g:

$$f(L_a, L_b) = \frac{L_a L_b + 1}{L_a + L_b}, \quad g(L_a, L_b, \hat{u}_s) = \begin{cases} L_a \cdot L_b, & \text{if } \hat{u}_s = 0 \\ L_b / L_a, & \text{if } \hat{u}_s = 1 \end{cases}, \tag{1.30}$$

where the function g also depends on the *upper branch state* $(\hat{u}_s)$ of the evaluated basic circuit.

This bit $\hat{u}_s$ represents the partial module-2 sum of previously estimated bits propagated from left to right. Therefore, the lower branch likelihood or result from g function is computed only after an available decision on the upper branch.



a) Flow of likelihood functions and decision bits

b) Basic circuits

Figure 1.11 Flow of likelihoods in decoding schemes

For the implementation of the decoder take into account that likelihood ratios are underflowing with large block lengths, so that it is necessary to work with log-likelihood ratios (LLRs) in order to avoid this numerical issue. Moreover, a successive-cancellation-decoder working in

the log-domain reduces the hardware implementation complexity of f and g functions (Leroux *et al.* (2011)), since multiplication and division operation are eliminated. In the logarithmic domain f and g functions become:

$$f(l_a, l_b) = 2\tanh^{-1}\left(\tanh\left(\frac{l_a}{2}\right)\tanh\left(\frac{l_b}{2}\right)\right), \quad g(l_a, l_b, \hat{u}_s) = \begin{cases} l_a + l_b, & \text{if } \hat{u}_s = 0 \\ l_b - l_a, & \text{if } \hat{u}_s = 1 \end{cases} \tag{1.31}$$

In order to reduce the complexity of the function f, it can be approximated using the *minimum function*, such that $f(l_a, l_b) \approx \text{sign}(l_a)\text{sign}(l_b)\min(|l_a|, |l_b|)$.

1.16 Decoding performance

Considering that a block-error event happens when the decoder output is different from the transmitted data vector ($\hat{u}_{\mathscr{I}} \neq u_{\mathscr{I}}$), Arikan (2009) determines that the upper bound on block-error probability under the successive cancellation decoding algorithm is the sum of the resulting Bhattacharyya parameters of non-frozen bit-channels, as follows

$$P_e(N, K, \mathscr{I}, u_{\mathscr{I}^c}) \leq \sum_{i \in \mathscr{I}} Z\left(W_N^{(i)}\right), \tag{1.32}$$

while the block-error probability of an $(N, K, \mathscr{I}, u_{\mathscr{I}^c})$ code is

$$P_e(N, K, \mathscr{I}, u_{\mathscr{I}^c}) = O(N^{-1/4}), \tag{1.33}$$

which is independently of the code rate. The decoding complexity is similar to the encoder, it is equal to $O(Nlog_2N)$. The low-complexity of encoding and decoding is due to the recursive structure of the channel polarization construction.

Although the decoding complexity of the SC decoder is low, it has drawbacks in practice. The error correction performance for short or moderate block lengths is not suitable for practical implementations. As we can see in Figure 1.12, the error rate of LDPC is better than the BER

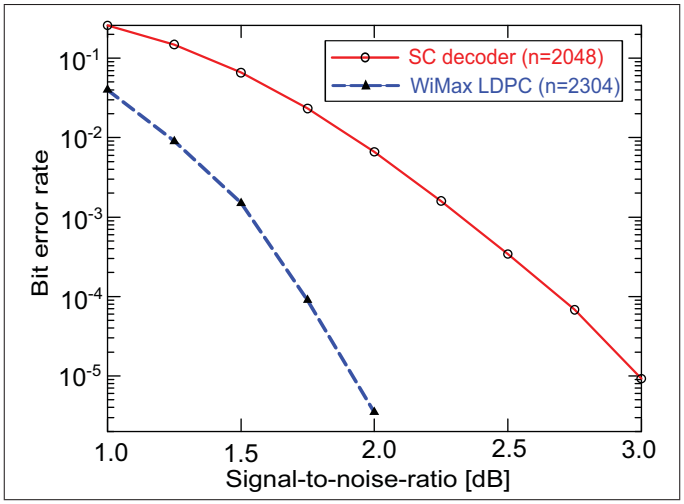


Figure 1.12 Polar codes versus LDPC
Taken from Tal & Vardy (2015)

of polar codes when its block length is $N = 2048$ and code rate is $R = 0.5$. In order to improve the error rate of polar codes, it is necessary to increase the block length, but it implies that the decoding latency of polar codes also will increase due to the sequence nature of the SC decoder. In other words, the SC decoder has low decoding throughput at long block lengths.

CHAPTER 2

DESIGN-SNR FOR CONSTRUCTION OF POLAR CODES

Polar codes are constructed based on a design-channel parameter. For AWGN channels, this parameter is given by the SNR per bit of the channel. We consider Bhattacharyya and Tal&Vardy construction methods to analyze the selection of *design-SNRs*. Based on an heuristic method proposed by Vangala *et al.* (2015) and considering the block-error performance, the best design-SNRs are determined for both constructions over AWGN channels. Finally, we perform a comparison of both construction methods with their corresponding design-SNRs. The results show that polar codes constructed with Bhattacharyya and Tal&Vardy algorithms have similar performances for a specific range of SNRs if optimized design-SNRs are used.

2.1 Bhattacharyya-based construction

As we see in section 1.13, the original construction of polar codes is based on the Bhattacharyya parameter, proposed by Arikan (2009). This method determines exactly the Bhattacharyya parameter of bit-channels only over bit erasure channels. However, Arikan *et al.* (2008) have proposed to use the same recursion (Equation (1.23)) for any arbitrary binary-input channel in order to approximate the Bhattacharyya parameter of bit-channels.

The Bhattacharyya parameters for BSCs and binary-input AWGN channels are derived from Equation (1.21), (Lin & Costello (2004)), which results in

$$\begin{aligned} Z_{BSC} &= \sqrt{P_{y|x}(0|0)P_{y|x}(0|1)} + \sqrt{P_{y|x}(1|0)P_{y|x}(1|1)} \\ &= \sqrt{(1-p)p} + \sqrt{p(1-p)} \\ &= 2\sqrt{p(1-p)}, \end{aligned} \tag{2.1}$$

where p represents the crossover probability of a BSC, and

$$\begin{aligned} Z_{AWGN} &= \int_{-\infty}^{+\infty} \sqrt{f_{y|x}(y|\sqrt{E_c}) f_{y|x}(y|-\sqrt{E_c})}\, dy \\ &= \int_{-\infty}^{+\infty} \sqrt{\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-\sqrt{E_c})^2}{2\sigma^2}} \cdot \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y+\sqrt{E_c})^2}{2\sigma^2}}}\, dy \\ &= e^{-\frac{E_c}{2\sigma^2}}, \end{aligned} \tag{2.2}$$

where E_c denotes energy per component of the codeword (or encoded bit) over an AWGN channel with zero mean and variance $\sigma^2 = \frac{N_0}{2}$, N_0 denotes power spectral density of the additive noise. BPSK modulation is employed, which maps 0 and 1 into $-\sqrt{E_c}$ and $+\sqrt{E_c}$, respectively. Considering Equation (2.2) and given that $E_c = R_c E_b$ (Proakis & Masoud (2008)), where R_c is the code rate and E_b is the energy per information bit, the initial Bhattacharyya parameter for construction rule (1.23) over AWGN channels is determined by $Z_1^{(1)} = e^{-R_c(E_b/N_0)}$, where (Eb/N_0) is the SNR per bit of the channel; while for BSCs it is calculated as $Z_1^{(1)} = 2\sqrt{p(1-p)}$.

The underlying calculation of the Bhattacharyya parameter over an AWGN channel shown in Equation (2.2) is also employed in other constructions, for instance the Gaussian-approximation-based construction over AWGN channels proposed by Li & Yuan (2013). This construction estimates intermediate log-likelihood ratios (LLRs) as Gaussian variables and computes directly the Bhattacharyya parameters of every i-th bit channel $W_N^{(i)}$ under the recursion (2.2) with $E_c = 1$, as follows

$$Z_N^{(i)} = e^{-1/(2(\sigma_N^{(i)})^2)}. \tag{2.3}$$

The variance of each bit-channel is determined as $\sigma_N^{(i)} = 2/m_N^{(i)}$ and the mean of the LLR density function $m_N^{(i)}$ at each stage of decoding is calculated employing formulas derived in (Chung *et al.* (2001)) for LDPC codes. After the computation of the Bhattacharyya parameter of each bit-channel, the noiseless ones are chosen to transmit the message. The results of this proposition using Gaussian approximations are not much different from the heuristic method of Arikan *et al.* (2008).

2.2 Tal & Vardy's construction

Mori & Tanaka (2009a), using density evolution, develop the construction of polar codes for a symmetric B-DMC with linear complexity in the block length, but with an exponential requirement in memory and a high computation complexity which increases with the code length. Based on these studies and using the recursive basic transformations (W^-, W^+) of Arikan, Tal & Vardy (2013) have proposed two close approximations of a bit-channel using *quantization*. With these approximations one can obtain a lower and upper bound on the probability of error ($P_e^{(i)}$) of each bit-channel. This implies the creation of two versions of a bit-channel, an upgraded $\mathscr{Q}_i$ and a degraded $\mathscr{Q}_i'$ one, with smaller output alphabets limited by the number of quantization symbols μ. The i-th bit channel W_i is thus "sandwiched" between them,

$$I(\mathscr{Q}_i) \leq I(W_i) \leq I(\mathscr{Q}_i'). \tag{2.4}$$

The desired error probabilities of bit-channels are estimated from their transition probabilities but some estimates are lost. Consequently, it is proposed to complement the estimation using Bhattacharyya parameters of Arikan (2009), whenever they are better, to improve the BER estimates. The final hybrid algorithm in (Tal & Vardy (2013)) is the most accurate construction available with theoretical guarantees, Vangala *et al.* (2015). The running time complexity of Tal & Vardy construction is $O(N\mu^2 log_2 \mu)$.

2.3 Design-SNR of polar codes over AWGN channels

Polar codes are known to be *non-universal*. This means that the generation of polar codes depends on channel condition, which is represented as a *design-channel parameter* of the construction phase, and defined as erasure probability (ε) in BECs, crossover probability (p) in BSCs or SNR per bit (E_b/N_0) in AWGN channels. So that the initial Bhattacharrya parameter for AWGN channels under the *design-*E_b/N_0 or *design-SNR* term can also be expressed as

$$Z_1^{(1)} = e^{-R_c(\text{design-}SNR)}. \tag{2.5}$$

In other words, the selection of the noiseless K *bit-channels* among N, depends on the given *design-SNR* over AWGN channels.

When polar code constructions are dynamically tailored to specific channel conditions according to where the code will be used, the expected performance in terms of bit-error rate is enhanced at the expense of more complexity in code construction (Arikan *et al.* (2008)). This channel-specific construction can achieve channel capacity in bit-erasure-channels, which is proved in (Arikan (2009)). On the other hand, Arikan *et al.* (2008) also propose to construct polar codes tailored to a fixed channel condition in BECs, with an erasure probability of 0.5 ($Z_1^{(1)} = \textit{design-}\varepsilon = 0.5$), which corresponds to the worst BER.

Channel-specific construction of polar codes results in optimal performance over BECs; however, according to Vangala *et al.* (2015) it does not happen over AWGN channels. In this chapter, after extensive simulations, we observe that dynamic updating of the design-SNR for Bhattacharyya constructions of polar codes does not guarantee the optimal performance at their corresponding SNRs over AWGN channels. Nevertheless, this is not true for Tal&Vardy constructions. Vangala *et al.* (2015) suggest to use one design-SNR for a set of SNRs, but it should be correctly chosen.

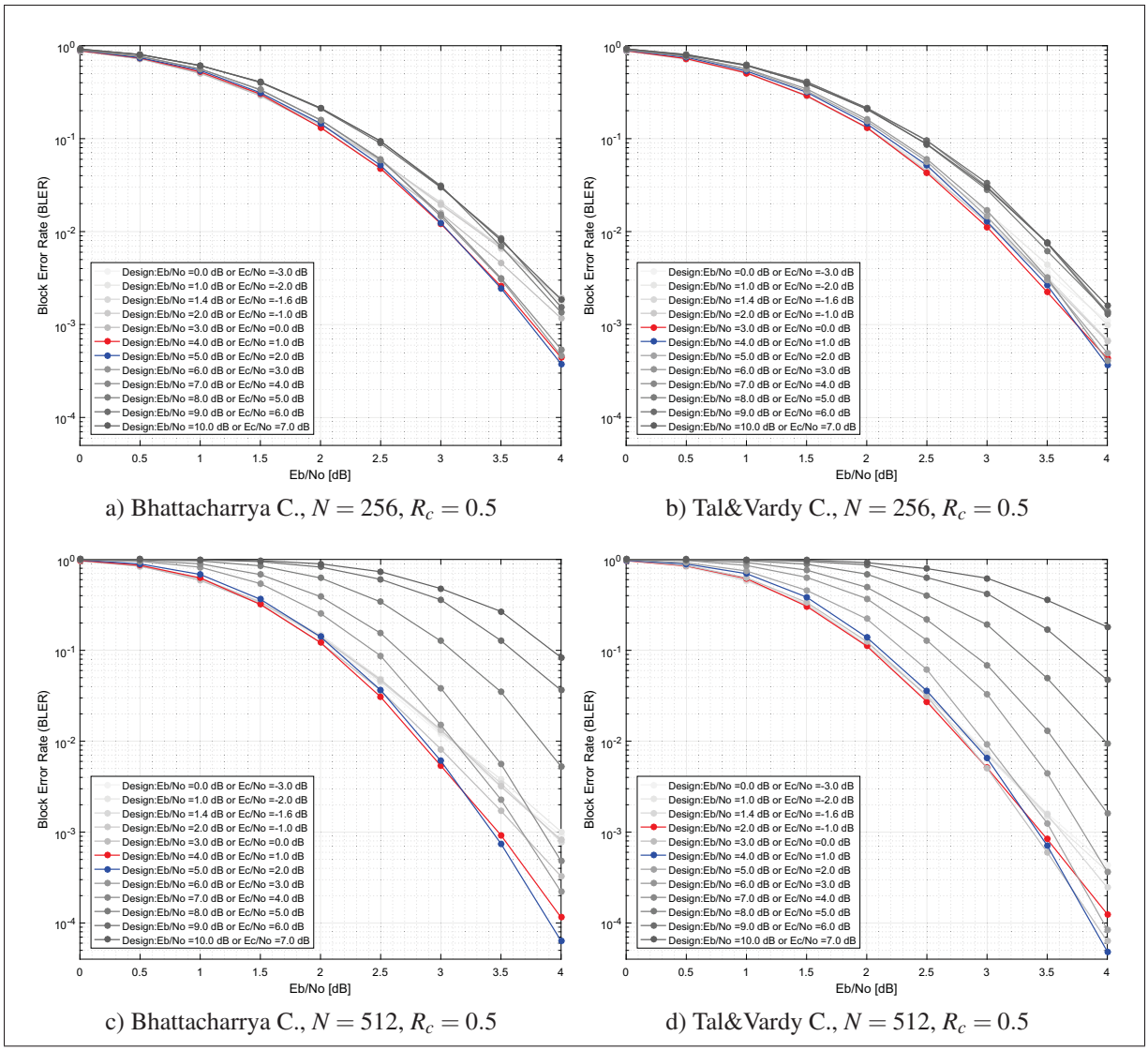
Vangala *et al.* (2015) compare the bit-error rate performances of different polar code construction algorithms over AWGN channels. They have concluded that all construction algorithms produce polar codes of the same good performance if they are constructed with the best design-SNR. The best design-SNR in terms of E_c/N_0 for different algorithms are: 0dB for Bhattacharyya method, 1dB for Monte-Carlo construction, -1.5917dB for Tal&Vardy construction, and -1.5917dB for Gaussian-approximation-based construction. This approach, of using a fixed design-SNR and consequently an invariant arrangement of bit-channels, is beneficial when there is no knowledge of the communication channel at the encoder (at construction phase), decoder or at both sides.

There are other researches concentrated in the design of universal polar codes, but they have usually obtained higher encoding and decoding complexities (Hassani & Urbanke (2014)), (Şaşoğlu & Wang (2016)).

2.4 Searching of the best design-SNRs

Considering that our main goal is to decrease the detection latency of a block of symbols, the analysis for determining the design-SNRs should be under the frame or block-error-rate (BLER). We use the *search algorithm*, proposed by Vangala *et al.* (2015), for determining the best design-SNRs of construction methods. In our work, we analyze the performance of polar codes employing Bhattacharyya and Tal&Vardy constructions for different block lengths over AWGN channels. This means that for each pre-established block length using any given construction method, we determine the design-SNR for a range of SNRs or BLERs. The determination of the best design-SNR is carried out by comparing performance curves at several possible design-SNRs, which are obtained using extensive simulations. We also perform a comparison between performances of polar codes under Bhattacharyya and Tal&Vardy construction methods employing the obtained design-SNRs.

Following *Vangala's search-algorithm*, first, we establish a set of design-SNRs of our interest. We consider a set of *design-*E_b/N_0 equal to $\{0,1,...,10\}$ in decibels (dB). Similar as Vangala's work, we also consider a design-E_b/N_0 = 1.42dB (design-E_c/N_0 = -1.59dB), which is equivalent to the worst initial value ($Z_1^{(1)} = 0.5$) proposed by Arikan *et al.* (2008) for BEC channels. Constructions of polar codes at the set of *design-*E_b/N_0 are performed with a fixed code rate, a fixed block length, and a particular method of construction. The design parameters used in our simulations are a code rate $R_c = 0.5$, block lengths $N = \{256, 512, 1024, 2048\}$, and Bhattacharyya and Tal&Vardy construction algorithms. After extensive simulations, performances of polar codes are plotted as curves of BLER vs. E_b/N_0, shown in Figures 2.1 and 2.2. We compare these performance curves and select one with the lowest BLER for a specific SNR or a range of SNRs of our interest. In particular, we select the performance curves with the lowest BLER for *a relative low/medium range and a high range of SNRs*, whose results are summa-



a) Bhattacharrya C., $N = 256$, $R_c = 0.5$

b) Tal&Vardy C., $N = 256$, $R_c = 0.5$

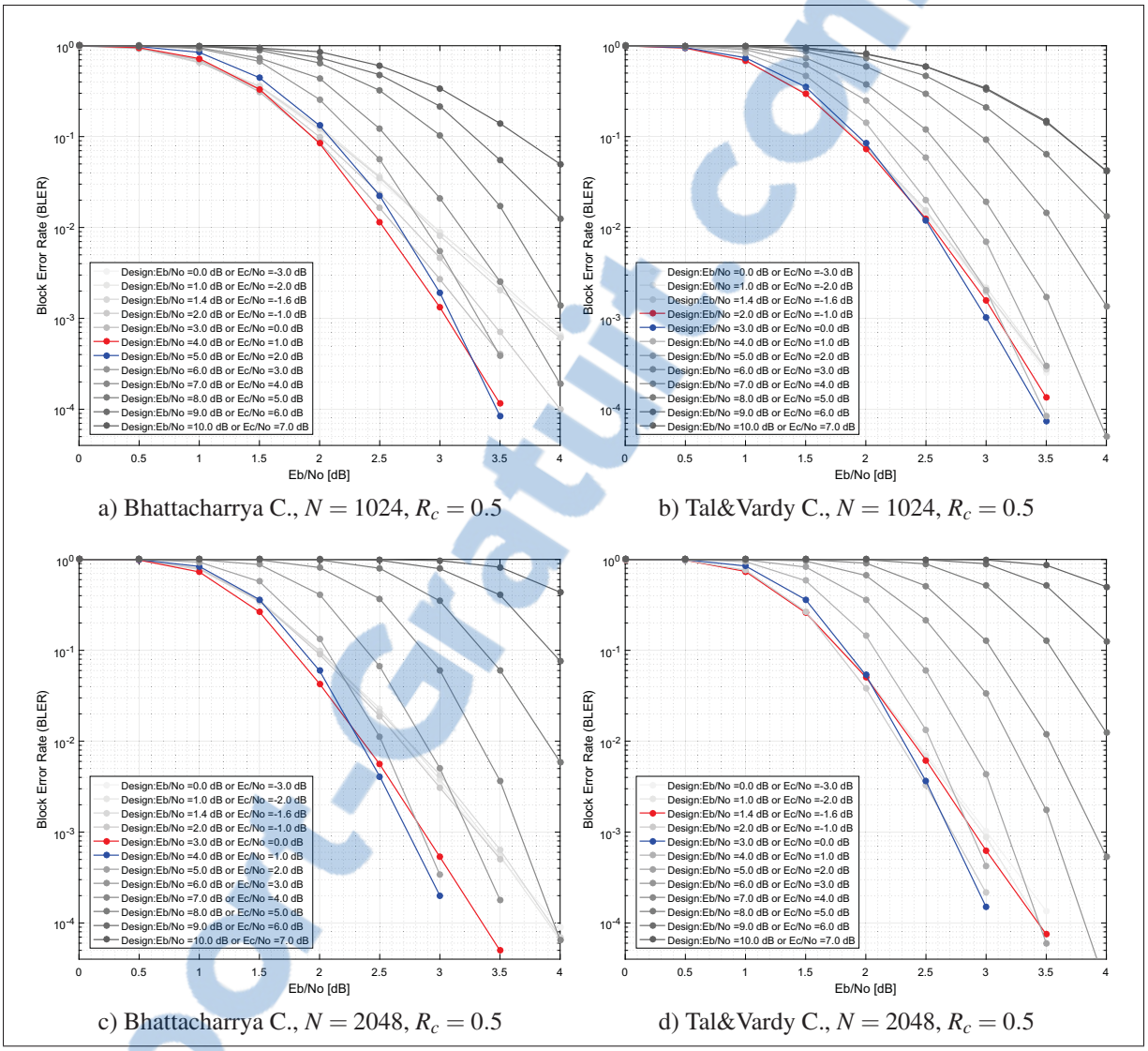
c) Bhattacharrya C., $N = 512$, $R_c = 0.5$

d) Tal&Vardy C., $N = 512$, $R_c = 0.5$

Figure 2.1 Performance of polar codes at different design-SNRs, under Bhattacharyya and Tal&Vardy constructions, $N = \{256, 512\}$ and $R_c = 0.5$

rized in Table 2.1. The selected curves are highlighted with red and blue colors in Figures 2.1 and 2.2. Therefore, the design-SNRs of the selected curves are declared as the "best" design-SNR for a specific range of SNRs under a fixed code rate and a fixed block length, either for Bhattacharyya or Tal&Vardy constructions.

Regarding the resulting BLER performance curves (shown in Figures 2.1 and 2.2), as it was expected, the BLER performance of polar codes depends on the block length N and the *design-*



a) Bhattacharrya C., $N = 1024$, $R_c = 0.5$

b) Tal&Vardy C., $N = 1024$, $R_c = 0.5$

c) Bhattacharrya C., $N = 2048$, $R_c = 0.5$

d) Tal&Vardy C., $N = 2048$, $R_c = 0.5$

Figure 2.2 Performance of polar codes at different design-SNRs, under Bhattacharyya and Tal&Vardy constructions, $N = \{1024, 2048\}$ and $R_c = 0.5$

E_b/N_0 employed in their construction. We can see in simulations of Bhattacharyya and Tal& Vardy construction methods, that their respective BLER performances get worse as the design-SNR increases from 3dB to 10dB or when design-SNR decreases from 3dB to 0dB. Although the latter varies in a lower magnitude. Moreover, as the block length N increases, there are larger BLER performance variations with design-SNRs, specially when design-SNRs are higher than 3dB. When design-SNRs are lower than 3dB, their corresponding BLER performance curves maintain relative small variations to each other. Thus, when the block length is

small, *e.g.* $N = 256$ (Figures 2.1a or 2.1b), the determination of the best design-SNR is tighter than when the block length is bigger, *e.g.* $N = 2048$ (Figures 2.2c or 2.2d).

2.4.1 Influence of the design-SNRs on polar code performance

Let's compare the performance of polar codes with the same design-SNR for different block lengths $(256, 512, 1024, 2048)$, the same construction method, and a fixed code rate $Rc = 0.5$. When the selected design-SNRs are approximately equal or lower than 5dB, we can see that the block-error performance improves progressively from certain SNR to 4dB as the block length N increases, see Figures 2.3a and 2.3b. Otherwise, performance of polar codes are degraded as shown in Figure 2.3c, where a design-SNR equal to 8dB is used as an example.

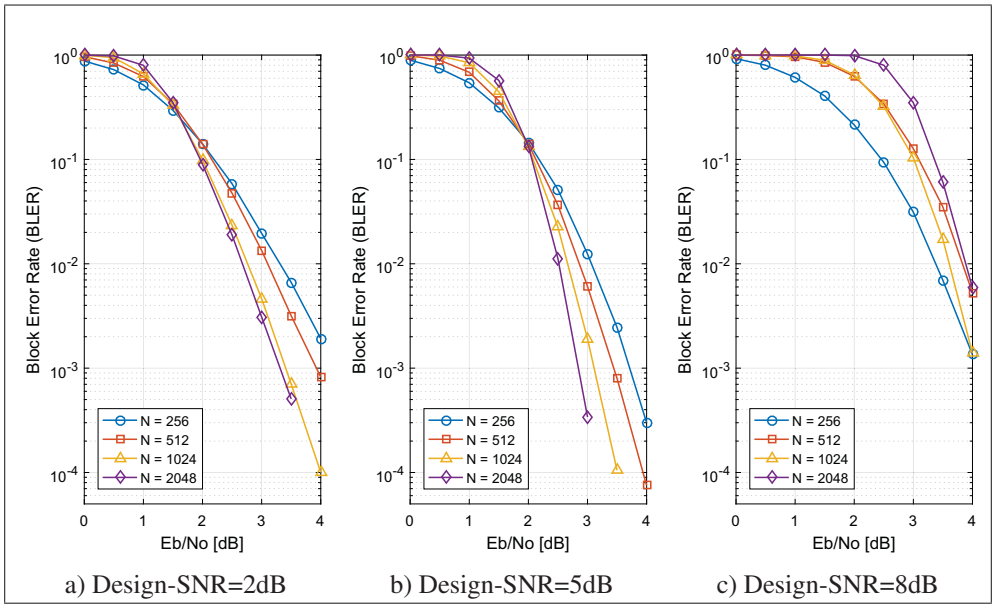


Figure 2.3 Performance of polar codes under Bhattacharyya construction, a fixed design-SNR and different block lengths

This behavior is common in both construction methods. Therefore, although a large block length and significant redundancy information (since $R_c = 0.5$) is used to construct polar codes, they do not guarantee an efficient performance of these codes. Accordingly, it is important to

choose correctly the design-SNR parameter under Bhattacharyya and Tal&Vardy constructions over AWGN channels.

2.4.2 Results

According to the obtained error performance curves, a design-SNR parameter does not result in low error rates for the complete range of SNRs from 0 to 4dB. A design-SNR parameter is only optimized for a specific range of SNRs. For example, in Bhattacharyya construction with block length $N = 2048$, see Figure 2.2c, the best design-SNR is 3dB for a low range of SNRs $\in [0.6; 2.25]$dB, while for a relative high range of SNRs $\in (2.25; 3]$dB, the best design-SNR is 5dB. There are also design-SNRs optimized for an intermediate range of SNRs. For example, in Tal&Vardy construction with block length $N = 512$, see Figure 2.1d, the best design-SNR is 2dB for a medium range of SNRs $\in [1.3; 2.95]$dB, but for upper or lower ranges of SNR, the best design-SNRs are 4dB and 1dB, respectively.

Table 2.1 The best design-SNRs to construct polar codes for specific ranges of SNR, under a specific construction method, block length and code rate

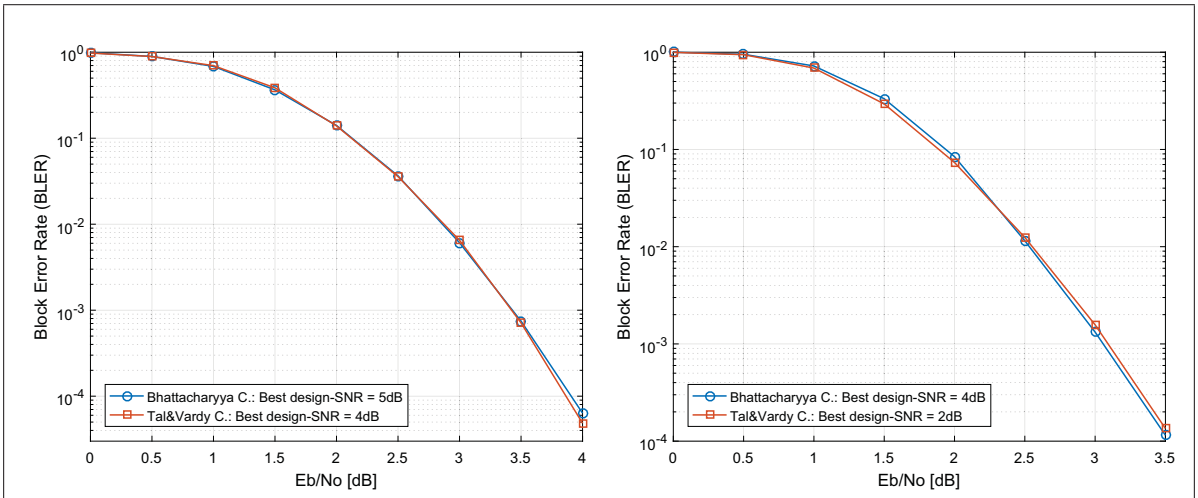
Construction	**N**	**R_c**	**Low/medium SNRs**		**High SNRs**	
			Design-E_b/N_0 [dB]	**E_b/N_0 range [dB]**	**Design-E_b/N_0 [dB]**	**E_b/N_0 range [dB]**
Bhattacharyya	256	0.5	4	[1.9;3.1]	5	(3.1;4]
	512		4	[1.45;3.2]	5	(3.2;4]
	1024		4	[1.9;3.25]	5	(3.25;4]
	2048		3	[0.6;2.25]	5	(2.25;3]
Tal&Vardy	256	0.5	3	[2.1;3.75]	4	(3.75;4]
	512		2	[1.3;2.95]	4	[3.7;4]
	1024		2	[1.25;2.4]	3	(2.4;3.5]
	2048		1.42	[0.8;1.55]	3	[2.65;3]

In *Bhattacharyya construction*s, we have found that performance curves designed with SNRs equal to or lower than 4dB are good candidates for being the design-SNR for relative low or medium ranges of SNR. While for relative high ranges of SNR we can find suitable design-SNRs at 5dB. On the other side, in *Tal&Vardy constructions*, performances at lower design-SNRs from 1.42dB to 3dB are good design-SNRs for relative low or medium ranges of SNR.

With design-SNRs between 3dB and 4dB, we obtain the best design-SNRs for relative high ranges of SNR, see Table 2.1.

2.4.3 Error performance comparison under the best design-SNRs

To end the analysis of the determination of design-SNRs, we compare the results obtained between Bhattacharyya and Tal&Vardy constructions. For this, we must take the obtained design-SNRs (shown in Table 2.1) for similar ranges of SNR of each construction method.



a) $N = 512$, optimized at high SNRs

b) $N = 1024$, optimized at low SNRs

Figure 2.4 Performance of polar codes with different construction methods, the best design-SNR and different block lengths

For instance, if we contrast performance of polar codes with $N = 512$, which are optimized for a relative high range of SNRs, then the best design-SNRs should be equal to 5dB and 4dB under Bhattacharyya and Tal&Vardy constructions, respectively. Or if we consider to compare performance of polar codes with $N = 1024$ for a relative low/medium range of SNRs, 4dB and 2dB are the best design-SNRs under their respective constructions, see Table 2.1. As shown in Figures 2.4a and 2.4b, both construction methods result in similar good performance, although they have different design-SNRs.

Table 2.2 Block error rate performance of Bhattacharyya an Tal&Vardy constructions, at design$-E_b/N_0 = \{0,1,...,5\}$, $N = 512$ and $R_c = 0.5$

Construction	**Design-E_b/N_0 [dB]**	**E_b/N_0 [dB]**				
		0	**1**	**2**	**3**	**4**
Bhattacharyya	0	0.964000*	0.619000	0.133600	0.011826	0.000915
	1	0.968000	0.606000	0.148100	0.012441	0.001010
	2	0.966000	0.623000	0.140300	0.013419	0.000820
	3	0.966000	0.589000*	0.121800	0.008170	0.000325
	4	0.974000	0.626000	0.121700*	0.005377*	0.000115
	5	0.986000	0.684000	0.140800	0.006082	0.000064*
Tal&Vardy	0	0.965917*	0.592000	0.120000	0.007567	0.000430
	1	0.967333	0.578000*	0.114000	0.007046	0.000370
	2	0.971000	0.616000	0.111385*	0.005099	0.000125
	3	0.982000	0.629000	0.125000	0.005052*	0.000063
	4	0.978000	0.698000	0.139000	0.006509	0.000049*
	5	0.991000	0.747000	0.222000	0.009133	0.000085

* Lowest block-error rate at a specific SNR

If we compare performance at design-SNRs that are not the best for similar ranges of SNR, such as for the high ranges of SNR with $N = 1024$ in Table 2.1, then one construction method will produce a better performance than the other. To contrast what construction method (Bhattacharyya or Tal&Vardy) is better with every best design-SNR at each SNR (every 0.5dB), we show their respective performance curves for different block lengths in Figure 2.5. As we see in this figure, although both constructions seem to be similar at first sight, we note that performance of Tal&Vardy constructions are slightly better than Bhattacharyya constructions as the SNR increases. Moreover, after analysing the data of the extensive simulations, we have noticed that in Tal&Vardy constructions the best BLER performance at each SNR matches with the design-SNR, as we can see in Table 2.2 for $N = 512$. On the other hand, in Bhattacharya constructions, performance of polar codes are not the best at their corresponding SNRs. For example, as shown in Table 2.2, if the design-$E_b/N_0 = 3$dB, the minimum BLER is at $E_b/N_0 = 1$dB, or the best BLER at $E_b/N_0 = 3$dB is when the design-$E_b/N_0 = 4$dB.

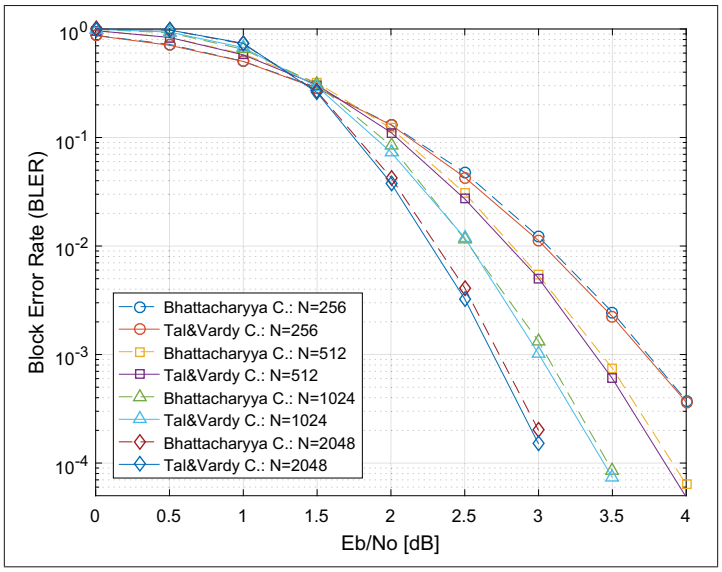


Figure 2.5 Performances of polar codes under Bhattacharyya and Tal&Vardy constructions with every best design-SNR at each SNR

2.5 Conclusion

Considering the obtained results, we corroborate that different construction algorithms can produce similar good performance of polar codes over AWGN channels if the design-SNR is optimized, Vangala *et al.* (2015). Moreover, Vangala's algorithm is valid for determining the best design-SNR under the comparison of block-error performances. It is to be noted that the best design-SNR allows to construct the best possible polar codes just for a specific range of SNRs or BLERs. We also conclude that despite the small advantage of performance of Tal&Vardy constructions over Bhattacharrya constructions, the Bhattacharyya construction is a good alternative for the design of polar codes. Take into account that Bhattacharrya construction has a very low complexity, and it obtains such a good performance as Tal&Vardy construction under a correct selection of the design-SNR. On the other side, although the performance of polar codes under Tal&Vardy construction is slightly better than Bhattacharyya construction, it has a larger complexity than Bhattacharyya construction. Besides, polar codes are optimized at the same SNRs for which they were designed with Tal&Vardy constructions. One of the best

design-SNRs determined in this section will be used to construct polar codes for decreasing the detection latency of a DCS through sequential early detections.

CHAPTER 3

SEQUENTIAL EARLY DETECTION BASED ON CRC-POLAR CODES

In the present chapter, it is introduced a sequential early detection scheme based on the concatenation of CRC codes and polar codes in an OFDM system over AWGN channels. Considering the implementation scheme of an OFDM system with IFFT/FFT algorithms, the implementation of the sequential early detection scheme is also presented. On the other side, taking into account the probability density function of the Gaussian noise and the mathematical representation of the received signals for every early detection, a general expression of the likelihood ratio to take decisions at the receiver is deduced. The expression to calculate the LLR of each channel symbol is derived under BPSK mapping, since our simulations are based on this mapping. Before developing selection processes to obtain the best possible setting of the early detection scheme, a suitable scenario is established to carry out this work. Data bits are located at the most reliable bit-channels of the output vector of the polar code construction, while CRC bits are in second priority. Besides, a polar bit-arrangement is chosen to make performance comparisons under a fixed polar code rate and different block lengths of the polar code. The best CRC polynomial for a specific CRC size is selected among four generator polynomials, three are published by Koopman (2016) and one is a standard polynomial. In total, forty CRC polynomials of different sizes are evaluated through simulations. From the set of the best CRC polynomials with different degrees, four polynomials with particular characteristics are selected to determine their suitable initial detection times (IDT). At the end, we obtain CRC polynomials with their respective IDTs that offer the best possible error and average latency performances on an early detection scheme under a specific block length and code rate of the polar code. Furthermore, based on the analysis of statistical latencies of three possible detection distributions, it is observed that small intervals between detections improve the average detection latency. The resulting average latencies are compared with the theoretical average latency defined by Au & Gagnon (2016) in the finite-blocklength regime and by CRC-polar codes. The optimal latencies in the finite-blocklength regime are not achieved, whereas the resulting latencies through simulations are similar to statistical latencies under CRC-polar codes.

3.1 Proposed scheme for decreasing the average detection latency

The proposed scheme for decreasing the average detection latency in a multicarrier communication system is based on Au & Gagnon's idea. Basically, this scheme consist of a concatenation of two codes and the creation of a message-checking-loop between the outer decoder and a sampling buffer, see Figure 3.1. The concatenated code is formed by an error-correcting code as the inner code and an error-detecting code as the outer code. Specifically, a polar code and a CRC code are used. The aim of the loop at the receiver is to check iteratively if the received message is possibly correct or not through the CRC code, at different early detection times τ_i or different number of received samples. The number of samples sent to the OFDM demodulator is controlled by a buffer that temporarily stores samples delivered by the ADC. As was mentioned in Section 1.9, the CRC error detection is not infallible, the undetected errors and false alarms depends on the employed CRC polynomial. The loop finishes when the CRC decoder obtains a remainder polynomial equal to zero or when the detection time τ_i reachs the OFDM symbol period T. The multicarrier demodulation, the polar decoding, and the CRC decoding are inside of this iterative process. Consequently, the detection interval $\Delta\tau$ between early detections is given by the processing delay of these operations, as we saw in Section 1.5.

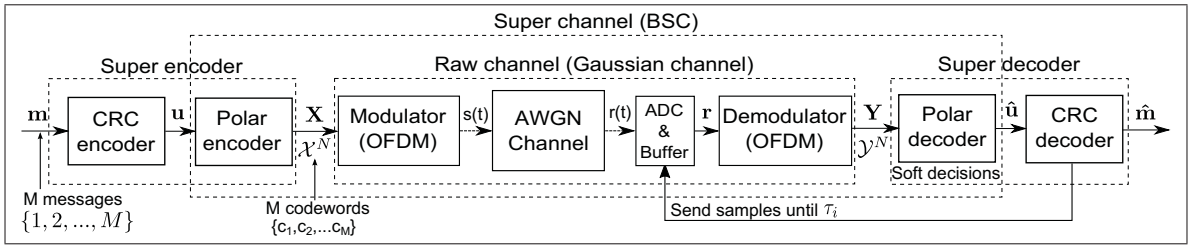


Figure 3.1 Basic scheme of sequential early detections based on the concatenation of CRC codes and polar codes

Under the proposed concatenated code scheme, two types of channels are identified. A raw channel, which is composed of an AWGN channel, the modulator, and the demodulator. The raw channel interacts directly with the encoder and decoder of polar codes. Note that the polar decoder receives from the demodulator soft values (greater than two quantization levels

or unquantized values) and consequently it develops a *soft-decision decoding*. The second channel in this scheme is a BSC, which receives binary values from the CRC encoder and yields binary values for the CRC decoder. The BSC encompasses the described raw channel and the encoder and decoder of polar codes. From concatenated codes point of view (Forney & Forney (1966)), the BSC of the scheme is known as a *super channel*, while CRC and polar encoders and decoders constitute a *super encoder* and a *super decoder*, respectively.

Consider an orthogonal frequency division multiplexing system with L subcarriers, and the concatenation of a (n,k) CRC code and a (N,K) polar code shown in Figure 3.2. The CRC encoder input is a binary message denoted as $\mathbf{m} = (m_0, m_1, ..., m_k)$, which previously passes through a serial to parallel converter. There are $M = 2^k$ possible messages, considering that the CRC encoder input is a block of k data bits. The CRC encoder transforms the binary k-tuple $\mathbf{m}$ into a codeword of block length n. The n-tuple CRC codeword results from appending the redundancy check bits or *CRC* to the k message bits. The n CRC-encoded bits constitute the input message of the polar encoder, *i.e.* $K = n$. The binary input of the polar encoder is represented by the vector $\mathbf{u} = (u_0, u_1, ..., u_{K-1})$. The output of the polar encoder is also a set of M possible codewords $\{c_1, c_2, ..., c_M\}$ of block length N, corresponding to the M possible input messages.

Polar encoded bits, also known as channel symbols, are the input bits of the OFDM modulation, and they are denoted by the vector $\mathbf{X} = (X_0, X_1, ..., X_{N-1})$. The N-tuple polar encoded bits $\mathbf{X}$ are mapped into L PSK or QAM frequency-domain symbols. The L complex numbers, represented by the vector $\mathbf{S} = (S_0, S_1, ..., S_{L-1})$, are processed by the inverse discrete-Fourier-transform to obtain time samples $(s_1, s_2, ...s_L)$ of the OFDM baseband signal $s(t)$. In other words, each frequency-domain symbol S_i employs a different subcarrier frequency for being transmitted during a symbol interval T. The L PSK/QAM symbols are transmitted at the same time over each subchannel. After the baseband signal $s(t)$ is upconverted to the carrier frequency f_c, the resulting bandpass signal $s_c(t)$ is transmitted over a channel with additive Gaussian noise $z(t)$. The frequency upconversion can be done by a quadrature modulator or by using any other kind of modulation, see Figure 3.2. At the receiver, inverse operations of the transmitter are

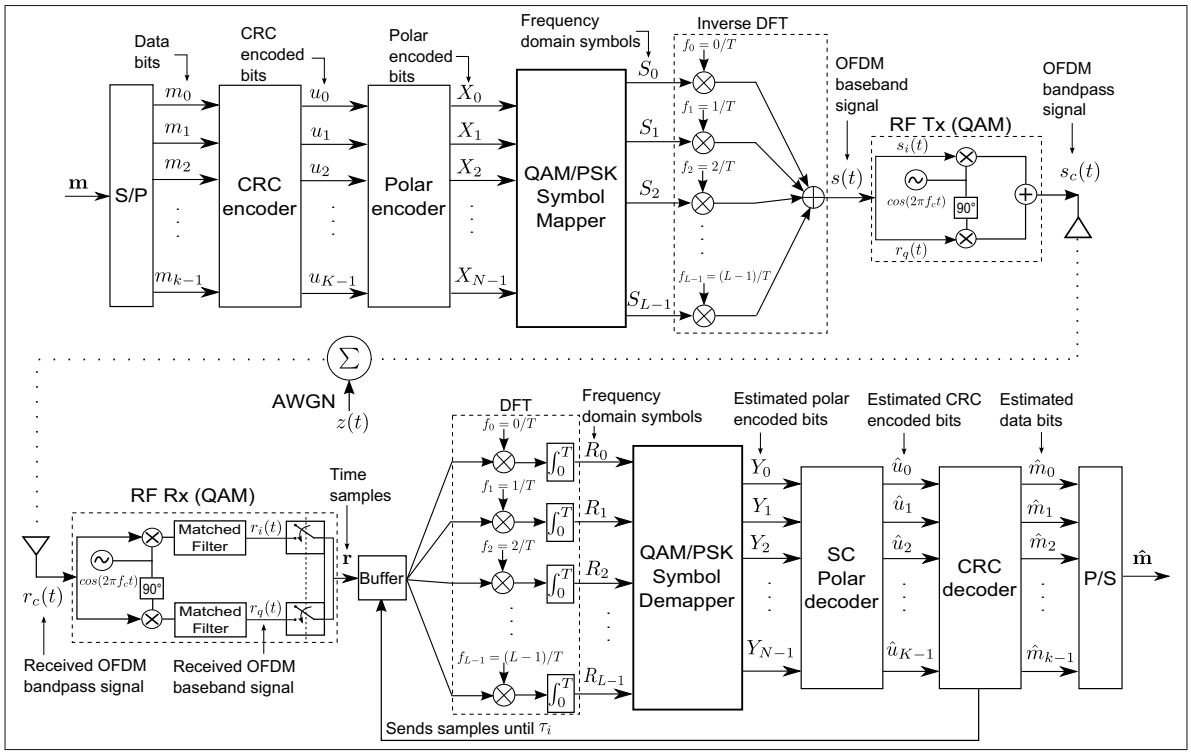


Figure 3.2 Sequential early detections scheme based on the concatenation of CRC codes and polar codes in an OFDM system

respectively performed to do early estimations of the original message. Specifically, there is a feedback from the CRC decoder to a buffer (that controls the flow of OFDM signal samples) in order to carry out iterative early detections τ_i of the received message during the period T.

It is important to note that the number of useful subcarriers should match with the number of PSK or QAM symbols that constitute an N-tuple codeword $\mathbf{X}$, namely L. To decrease the detection latency of the transmitted message $\mathbf{m}$, it is necessary to receive at the same time all the N polar encoded bits $(X_0, X_1, ..., X_{N-1})$ that make up a codeword $\mathbf{X}$. This is because the N polar encoded bits contain implicitly the n CRC encoded bits, and these in turn contain the k data bits that constitute the whole message $\mathbf{m}$. Assuming that the number of transmitted symbols is equal to L, if the number of subcarriers of the OFDM modulation/demodulation is greater or less than L, we will detect simultaneously incomplete codewords at the receiver. Consider the number of symbols of the employed digital modulation is $Q = 2^l$, where l is the number of polar encoded bits that compose a PSK/QAM symbol, and the block length

of the polar codeword is N. Then the number of Q-ary symbols transmitted per codeword is (Proakis & Masoud (2008))

$$L = \frac{N}{log_2 Q}, \tag{3.1}$$

which should be equal to the number of useful subcarriers of the OFDM system. When the number of symbols defined by the symbol mapper is equal or higher than four $(Q \geq 4)$, it is possible to send two or more encoded bits X_i over each subcarrier f_i of the OFDM system.

The total transmitted power of a word of L QAM/PSK symbols through an OFDM system is given by (Proakis & Masoud (2008))

$$P = \sum_{i=0}^{L-1} P_i, \tag{3.2}$$

where P_i represents the power assigned to each i-th subcarrier. Assuming that each subcarrier has the same power P_{sc}, then the total power of the L transmitted symbols is $P = LP_{sc}$. Taking into account that the L QAM/PSK symbols are transmitted in parallel during the OFDM symbol period T, the total energy required to transmit a word of L symbols is given by

$$\begin{aligned} E &= PT \\ &= LP_{sc}T \\ &= \left(\frac{N}{\log_2 Q}\right) P_{sc}T. \end{aligned} \tag{3.3}$$

In Equation (3.3), the number of transmitted symbols is replaced from Equation (3.1). From the resulting energy under synchronous detections, shown in Equation (3.3), the energy per each transmitted word of symbols under an early detection τ is given by

$$E_\tau = \left(\frac{N}{\log_2 Q}\right) P_{sc}\tau, \tag{3.4}$$

where τ is a shorter duration than the symbol period T.

3.2 Implementation of the proposed scheme

From the implementation point of view, the proposed scheme in an OFDM system employs IFFT/FFT algorithms, see Figure 3.3. Moreover, in order to eliminate all ISI, a cyclic prefix (CP) or more samples are added to the time samples $(s_0, s_1, ..., s_{L-1})$ yielded by the IFFT. This new set of samples are then ordered by a parallel to serial converter and processed by a digital-to-analog converter (DAC) to obtain the OFDM baseband signal $s(t)$.

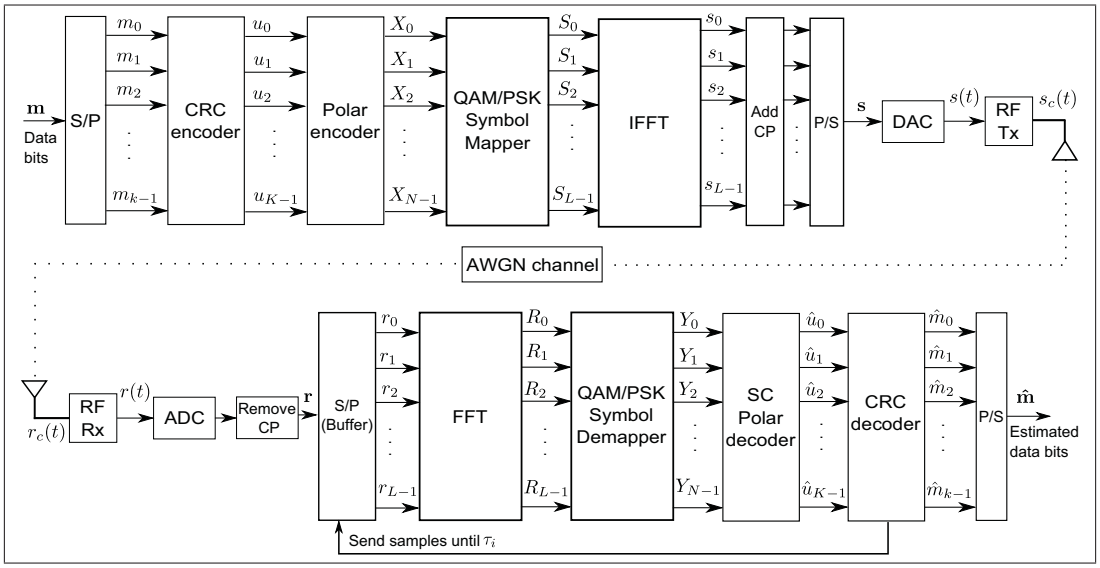


Figure 3.3 Implementation scheme of sequential early detections based on CRC and polar codes under an OFDM system with IFFT/FFT

At the receiver, the recovered OFDM baseband signal $r(t)$ is sampled by an analog-to-digital converter (ADC). The number of time samples delivered to the FFT is controlled by the serial-to-parallel (S/P) converter, which releases different quantities of samples according to the early detection times $\{\tau_1, \tau_2, ..., T\}$. These early detections are given when the CRC decoder sends the feedback signal indicating that the received message $\hat{\mathbf{m}}$ is not correct. This means that the S/P converter requires some type of buffering to temporarily store samples that were already sent to the FFT and other new samples that will arrive while the detection (demodulation and decoding) process is developed. The S/P converter output is used many times before a complete parallelization, where outputs are zeros for non-converted samples.

The initial detection time τ_1 is pre-established by the scheme design, typically $\tau_1 = T/2$. For example, assuming that the OFDM symbol period T is spanned by 1000 samples, if the initial early detection time is set to $\tau_1 = T/2$, the corresponding number of samples delivered by the S/P converter is 500. These 500 samples are processed by the FFT, the symbol demapper, the polar decoder and the CRC decoder. The first two operations can be seen as one process, the OFDM demodulation.

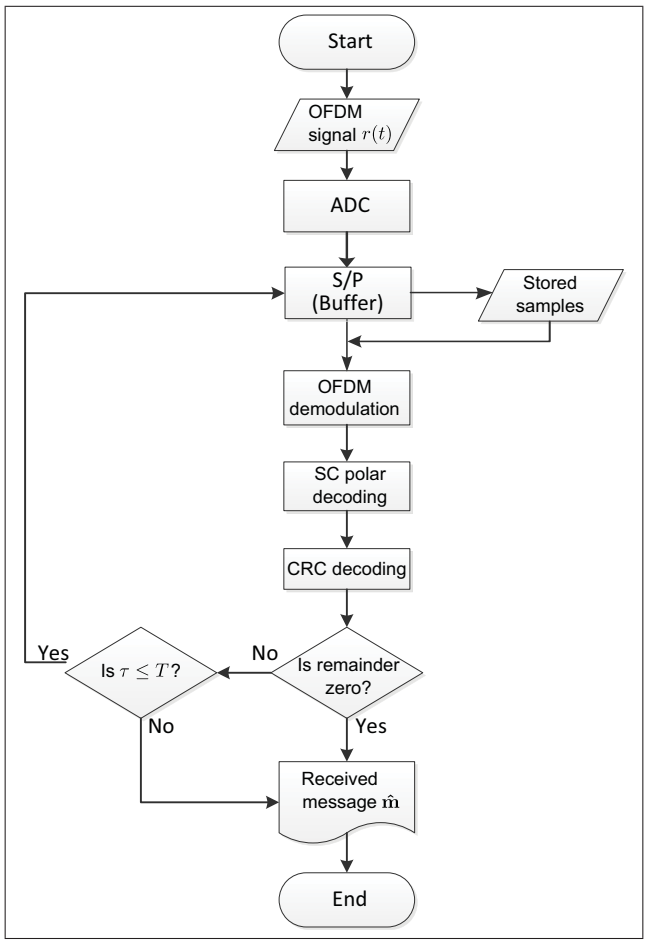


Figure 3.4 Flow diagram of CRC-based sequential early detections

If the CRC decoder indicates that the received message is incorrect (through the CRC-decoding feedback), then the S/P converter transfers the stored samples of the signal $r(t)$ up to that time (τ_2). Otherwise, the reception process continues normally, that is, the decoded message of

the CRC decoder $\hat{\mathbf{m}}$ is accepted as the transmitted original message. Continuing with the last example, suppose the feedback was done at $\tau_2 = 3T/4$, then the new number of delivered samples is 750. The new set of samples delivered by the S/P converter pass through all the operations described before, up to the CRC decoder. The described process is repeated until $\tau_i = T$ in the worst case. The message detection is expected that happens before τ_i achieves T. Evidently, this iterative process works assuming that the total processing time of the involved operations is much less than the OFDM symbol duration. A flow diagram of the iterative process to do early detections in an OFDM system is shown in Figure 3.4.

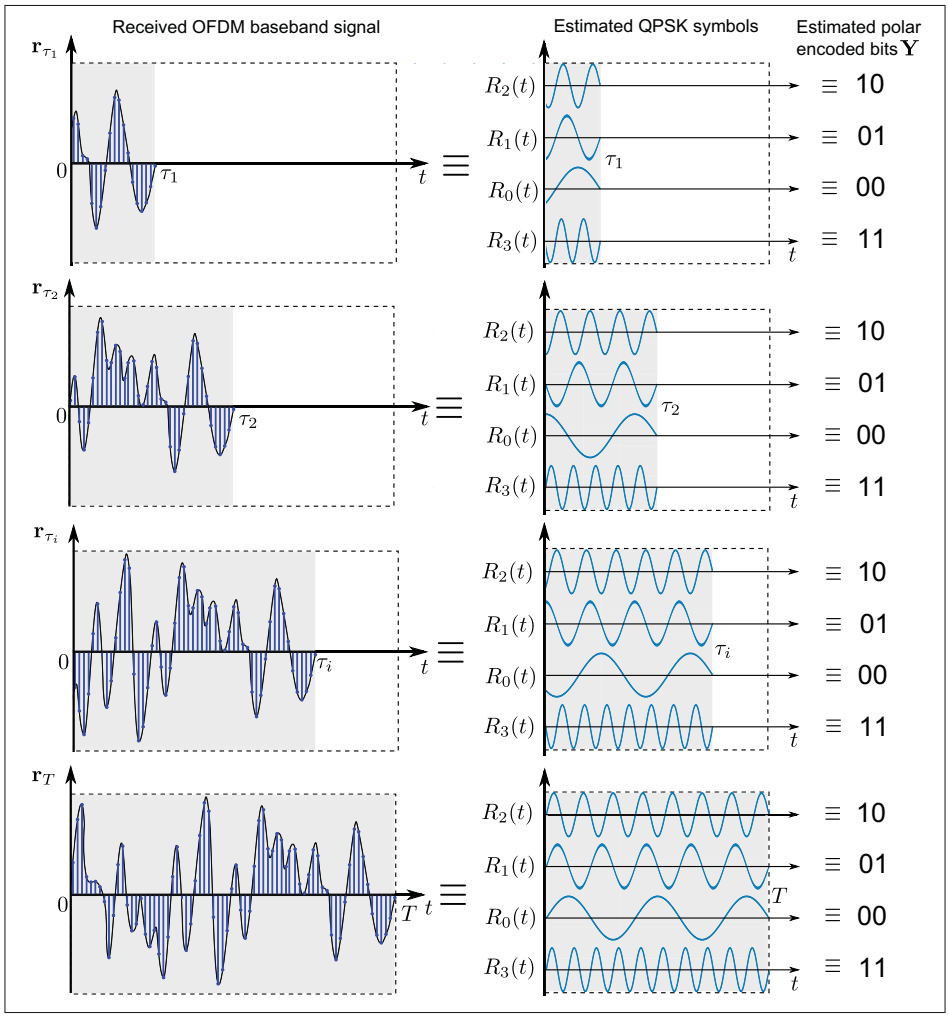


Figure 3.5 Sequential early detection of QPSK symbols under OFDM modulation

Recall that the L QAM/PSK symbols, denoted by the vector $\mathbf{S}$, are transmitted in parallel through an OFDM baseband signal $s(t)$. That is, the L QAM/PSK symbols are implicitly represented by the OFDM signal $s(t)$. This means, if the OFDM signal $s(t)$ is affected by an operation, then the L QAM/PSK symbols (S_i) are also affected by this operation. This is reciprocal with the received OFDM baseband signal $r(t)$ and the estimated QAM/PSK symbols, denoted by the vector $\mathbf{R}$. The early detection on the received OFDM baseband signal $r(t)$ implies a simultaneously early detection on the L estimated QAM/PSK symbols (R_i). This is graphically depicted in Figure 3.5, where different early detections τ_i are applied on the OFDM signal $r(t)$ and implicitly on its corresponding quadrature phase shift keying (QPSK) symbols $R_i(t)$ in the time domain. In fact, $r(t)$ is depicted as a set of samples limited by sequential early detections $\{\tau_1, \tau_2, ..., T\}$, denoted as $\{\mathbf{r}_{\tau_1}, \mathbf{r}_{\tau_2}, ..., \mathbf{r}_T\}$, since they are outputs of the ADC. In this graphic illustration, each symbol represents a pair of polar encoded bits, since QPSK symbols are used.

3.3 Problem formulation

Assume that each polar codeword $\mathbf{X}$ is sent in "one-shot" through L modulated symbols, which are transmitted in parallel with symbol duration T. The L QAM or PSK transmitted and received symbols are complex numbers in frequency domain, *i.e.* $\mathbf{S}$ and $\mathbf{R} \in \mathbb{C}^L$. The message of transmitted symbols is represented by a L-tuple random vector $\mathbf{S} = [S_0, S_1, ..., S_{L-1}]^{\text{tr}}$ where each component S_i can take any of the Q possible symbol values, $\{a_0, a_1, ..., a_{Q-1}\}$. The term 'tr' denotes the transpose of a vector. At the receiver, let's consider a finite set of early detections $\tau = \{\tau_1, \tau_2, ..., \tau_t\}$ performed on each message, where τ_t represents the last early detection when the transmitted message is detected or the symbol duration is achieved, *i.e.* $\tau_t \leq T$. Consequently, there is a set of vector random variables $\{\mathbf{R}(\tau_1), \mathbf{R}(\tau_2), ..., \mathbf{R}(\tau_t)\}$ determined by these early detections. The message of received symbols with an early detection τ_j, where $j \in \{1, 2, ..., t\}$, is represented by a vector random variable $\mathbf{R}(\tau_j) = [R_0(\tau_j), R_1(\tau_j), ..., R_{L-1}(\tau_j)]^{\text{tr}}$. The channel noise at an early detection τ_j is denoted by a vector random variable $\mathbf{Z}(\tau_j) = [Z_0(\tau_j), Z_1(\tau_j), ..., Z_{L-1}(\tau_j)]^{\text{tr}}$ composed of L components or Gaussian random variables with

i.i.d. real and imaginary parts, each $\mathcal{N}(0, N_0/2)$. Therefore, the observed vector at an early detection τ_j can be modeled as

$$\mathbf{R}(\tau_j) = \mathbf{S}(\tau_j) + \mathbf{Z}(\tau_j). \tag{3.5}$$

The observation of one component or random variable of the receive message $\mathbf{R}(\tau_j)$ at an early detection τ_j is represented by

$$R_i(\tau_j) = S_i(\tau_j) + Z_i(\tau_j), \tag{3.6}$$

where i is an index that point out the i-th component of the received message, $i \in \{1, 2, ..., L\}$. Now consider simultaneously the L components of a received message $\mathbf{R}$ and the t early detections applied on each component of the received message. From this point of view, we define a matrix of received symbols $\mathbf{R}_{\mathrm{L\ x\ t}}$, with L rows and t columns. The matrix $\mathbf{R}_{\mathrm{L\ x\ t}}$ is composed by Gaussian random variables R_{ij}. The index i denotes which component of the received message $[R_0, R_1, ..., R_{L-1}]^{\mathrm{tr}}$ represents the random variable R_{ij}, and the index j indicates which early detection, from $\{\tau_1, \tau_2, ..., \tau_t\}$, was applied to the i-th component of the received message.

$$\mathbf{R}_{\mathrm{L\ x\ t}} = \begin{bmatrix} R_{11} & R_{12} & \dots & R_{1j} & \dots & R_{1t} \\ R_{21} & R_{22} & \dots & R_{2j} & \dots & R_{2t} \\ \vdots & \vdots & & \vdots & & \vdots \\ R_{i1} & R_{i2} & \dots & R_{ij} & \dots & R_{it} \\ \vdots & \vdots & & \vdots & & \vdots \\ R_{L1} & R_{L2} & \dots & R_{Lj} & \dots & R_{Lt} \end{bmatrix}. \tag{3.7}$$

The vector of random variables $\mathbf{R}(\tau_j)$ of Equation (3.5) represents the j-th column of the matrix $\mathbf{R}_{\mathrm{L\ x\ t}}$, whereas the random variable $R_i(\tau_j)$ of Equation (3.6) denotes the component R_{ij} of the matrix $\mathbf{R}_{\mathrm{L\ x\ t}}$. The probability density function (PDF) of the Gaussian noise corresponding to

the i-th component of the received vector at an early detection τ_j can be expressed as

$$f_{Z_{ij}}(z_{ij}) = \frac{1}{\sqrt{\pi N_0}} \exp\left(-\frac{z_{ij}^2}{N_0}\right). \tag{3.8}$$

Consider a transmitted component $S_i(\tau_j)$, also denoted as S_{ij}, that takes a specific symbol value a_v from the set of possible transmitted symbols, *i.e.* $a_v \in \{a_0, a_1, ..., a_{Q-1}\}$. From the Equation (3.6), the corresponding received component $R_{ij} = a_v + Z_{ij}$. Since the component Z_{ij} of the noise vector is a Gaussian random variable with zero mean, the component R_{ij} of the received vector is also a Gaussian random variable with mean equal to a_v. Namely $R_{ij} \sim \mathcal{N}(a_v, N_0/2)$. Therefore, from Equation (3.8), the conditional PDF of the random variable R_{ij}, given that the symbol a_v was sent, is calculated as

$$f_{R_{ij}|S_{ij}}(r_{ij}|a_v) = \frac{1}{\sqrt{\pi N_0}} \exp\left(\frac{-(r_{ij} - a_v)^2}{N_0}\right), \tag{3.9}$$

where r_{ij} is the complex number of the random variable R_{ij} that represents the received symbol. The conditional PDF of Equation (3.9) is also called the *likelihood of* a_v, which is commonly used in hypothesis testing. The *Q-ary hypothesis testing* present in our problem can be managed as multiple binary hypothesis testing problems. Thus, the *likelihood ratio* (LR) between two different symbols a_v and $a_{v'}$ is given by

$$\Lambda(r_{ij}) = \frac{f_{R_{ij}|S_{ij}}(r_{ij}|a_v)}{f_{R_{ij}|S_{ij}}(r_{ij}|a_{v'})} \tag{3.10}$$

The obtained LR is a sufficient statistic that depends on the observed component R_{ij} and it is used to make decisions at the symbol demapper.

3.4 Basic setting of the early detection scheme

The sequential early detection scheme, based on concatenated CRC and polar codes, is simulated under specific conditions. Taking into account the results obtain in Chapter 2, for the

construction of polar codes we employ the Bhattacharrya algorithm with a design-SNR equal to 4dB. This means, we are optimizing our results around 2.5dB of SNR when the block length is 256 or 512, see Table 2.1. The decoding process of polar codes is performed by a successive cancellation decoder. The OFDM modulation manages BPSK symbols, which are transmitted in parallel over an AWGN channel with zero mean and variance $\sigma^2 = N_0/2$. Since one symbol S_i represents one polar encoded bit X_i with BPSK mapping $(Q = 2)$, the number of symbols transmitted at the same time is equal to the number of polar encoded bits $(L = N)$. This is corroborated with Equation (3.1). The two symbol values used by the BPSK mapping are $a_0 = -\sqrt{E_s}$ and $a_1 = +\sqrt{E_s}$, which represent 0 and 1, respectively. In general terms, the energy of a symbol QAM/PSK is denoted by E_s and is given by

$$E_s = lE_c, \tag{3.11}$$

where l is the number of encoded bits per symbol, with $l = log_2(Q)$, and E_c is the energy of each encoded bit. With BPSK $l = 1$, consequently $E_s = E_c = R_c E_b$. Moreover, since the constellation points of BPSK symbols are positioned on the real axis, then $\mathbf{S}, \mathbf{R}$, and $\mathbf{Z} \in \mathbb{R}^L$.

In this scenario with an AWGN channel, the successive cancellation decoder is able to receive soft (un-quantized) values and deliver hard values for the outer CRC decoder. Under these conditions, it is not necessary to carry out a hard decision at the demapper. However the decoding process starts with the determination of the channel LLRs. Considering that BSPK mapping is applied, the received vector $\mathbf{R}$ is equivalent to the vector $\mathbf{Y}$, as well as the computation of their LLRs with Equations (3.10) and (1.29), respectively. Therefore, the channel LLRs are calculated based on the observations of the Gaussian random vector $\mathbf{R} = [R_0, R_1, ..., R_{L-1}]$ at every early detection τ_j as follows

$$\Lambda(r_i(\tau_j)) = \frac{f_{R_i(\tau_j)|S_i(\tau_j)}(r_{ij}| - \sqrt{E_s})}{f_{R_i(\tau_j)|S_i(\tau_j)}(r_{ij}| + \sqrt{E_s})}. \tag{3.12}$$

Consider the Equation (3.9) to replace the likelihoods of the symbols $a_0 = -\sqrt{E_s}$ and $a_1 = +\sqrt{E_s}$, such that

$$\begin{aligned}\Lambda(r_i(\tau_j)) &= \frac{\frac{1}{\sqrt{\pi N_0}}\exp\left[\frac{-(r_{ij}+\sqrt{E_s})^2}{N_0}\right]}{\frac{1}{\sqrt{\pi N_0}}\exp\left[\frac{-(r_{ij}-\sqrt{E_s})^2}{N_0}\right]} \\ &= \exp\left[\frac{-(r_{ij}+\sqrt{E_s})^2+(r_{ij}-\sqrt{E_s})^2}{N_0}\right]. \end{aligned} \tag{3.13}$$

After the natural logarithm is applied on Equation (3.13), the LLR of each i-th component of the received vector $\mathbf{R}$ with an early detection τ_j is given by

$$\mathrm{LLR}(r_i(\tau_j)) = -\frac{4r_{ij}\sqrt{Es}}{N_0} \tag{3.14}$$

It is to be noted that these channel LLRs are the inputs of the functions f and g of the SC decoder, seen at section 1.15.

3.4.1 Assignation of bit channels

Consider that the transmission of a data message $\mathbf{m}$ involves the concatenation of an (n,k) CRC code and an (N,K) polar code. Where the N polar-encoded bits contains n or K CRC-encoded bits, and this in turn has k data bits, as shown in Figure 3.6. The code rate of a concatenated code, denoted by R_{cc}, is defined by the product of the inner and outer code rates (Proakis & Masoud (2008)), so that

$$R_{cc} = R_{pc} \cdot R_{crc}, \tag{3.15}$$

where R_{pc} and R_{crc} denotes the polar and CRC code rates for the proposed scheme, respectively. Consequently, the concatenated code rate is $R_{cc} = k/N$, which means that k information bits are contained in N polar encoded bits.

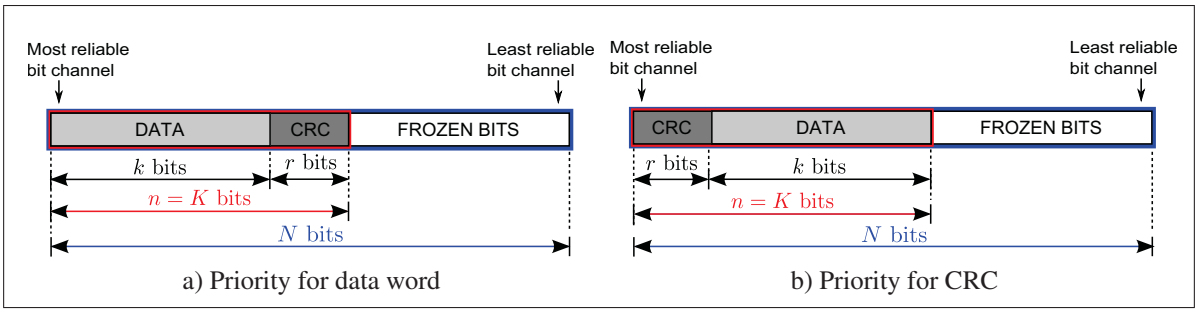


Figure 3.6 Assignation of bit channels in polar codeword

Since the CRC check sequence is consider as part of the data $u_{\mathscr{I}}$ to be encoded by the polar encoder, a question arises. What is the right assignation of *bit-channels* $W_N^{(i)}$ for the r CRC-bits in the polar code construction output vector u_0^{N-1}? To answer this question, two possible locations for CRC-bits are assumed. The first option is to give absolute priority to the k data bits and then to the r CRC-bits, as shown in Figure 3.6a. This means that the k information bits are located at the most reliable bit-channels. The second option is the opposite of the first one, that is, the r CRC-bits occupy the most reliable bit-channels and then the k data bits, see Figure 3.6b. The frozen bits of both options are always the least reliable bit-channels.

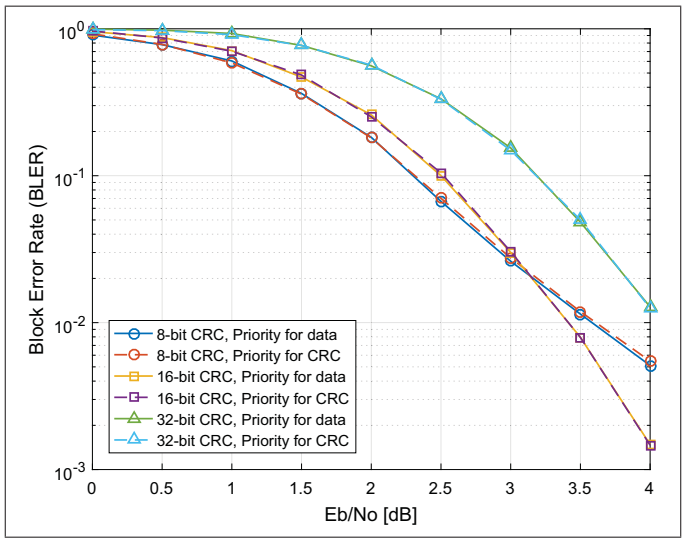


Figure 3.7 Sequential early detection performance based on CRC-polar codes with different bit channel assignation, $N = 256$, $R_{pc} = 0.5$, $\tau_1 = 50\%$, $\Delta\tau = 5\%$

The results show in Figure 3.7 indicate that both options give similar error performance regardless the CRC-bits location at the K most reliable bit-channels, when the block length is equal to 256. For further analysis, we choose the first option, since data bits are located at the most reliable bit-channels and the k data bits are larger targets for errors. Note that usually $k > r$. In the second option, the protection of the CRC-bits is easier because the number of bits is less. However, it is more likely that the data word gets errors due to its large length and location at less reliable bit-channels. Hence, we decide to give more preference or protection to data bits.

3.4.2 Bit-arrangement formats

Two bit-arrangement formats are proposed and analyzed. At the end, one bit-arrangement format is selected for the comparison of CRC polynomials. Of course, these formats take into account the pre-established data word preference, but they differ from each other regarding the CRC-bits location in the vector u_0^{N-1}. Both formats work with different CRC sizes. The first bit-arrangement format maintains constant the block length of the CRC codeword (n), whereas the second format maintains constant the block length of the data word (k), see Figure 3.8. This corresponds to fix the polar code rate R_{pc} or the concatenated code rate R_{cc} of the concatenated CRC-polar code in the detection scheme, these rates are underlined in Table 3.1.

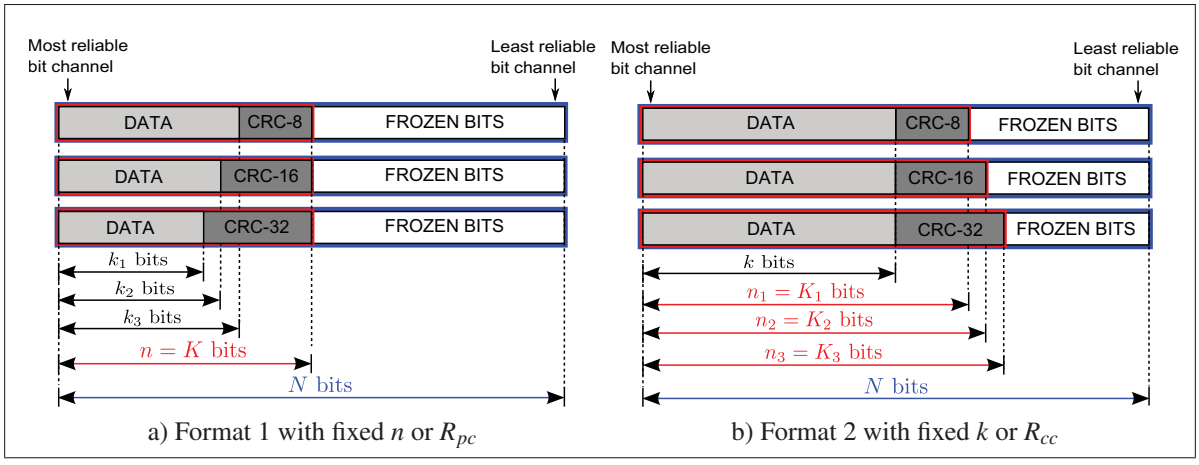


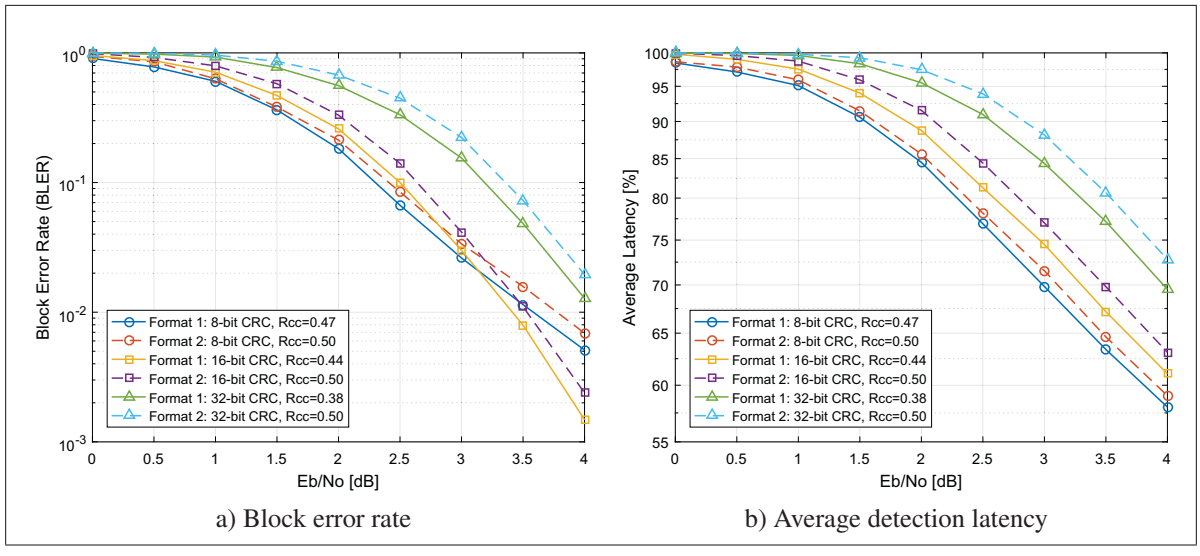
Figure 3.8 Bit-arrangement formats

Moreover, in the first arrangements, the concatenated code rate varies according with the CRC code rate. Hence, when the CRC size increases or the R_{crc} decreases, the R_{cc} also decreases. In the second arrangements, when the CRC size increases, the R_{pc} also increases. In other words, these two formats represent variations of codes rates, as shown in Table 3.1.

Table 3.1 Code rates under different bit-arrangements and CRC sizes, $N = 256$

Bit-arrangement format	**CRC size**	**R_{crc}**	**R_{pc}**	**R_{cc}**
1	8	0.94	0.50	0.47
	16	0.88	0.50	0.44
	32	0.75	0.50	0.38
2	8	0.94	0.53	0.50
	16	0.89	0.56	0.50
	32	0.80	0.63	0.50

A comparison of the block-error rates and the average detection latencies of the two proposed formats for each CRC size, is shown in Figure 3.9.



a) Block error rate b) Average detection latency

Figure 3.9 Performances of sequential early detections based on 8,16 and 32-bit CRCs, different concatenated code rates, $N = 256$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

The arrangements with constant polar code rate deliver a better error and detection latency performance than the arrangements with fixed concatenated code rate. This is due to the fact that the concatenated code rates of the first arrangements are lower than the concatenated code rates of the second arrangements. Format 1 transmits more redundancy or frozen bits than format 2. Therefore, a better error performance of the first bit-arrangement format is achieved at the expense of a decreased concatenated code rate or less transmitted information per frame, see Table 3.1. On the other side, the second bit-arrangement format carries more data at the expense of a degraded error and latency performance. For the following sections, the format 1 is used. Independently of the proposed formats but under a fixed number of CRC bits and a fixed block length, the CRC-based sequential early detection scheme with a lower concatenated code rate (R_{cc}) generates a better error and detection latency performance than the scheme with a higher R_{cc}, see Figure 3.9.

3.4.3 Concatenated CRC-polar code without early detections

It is important to know the effect of different CRC sizes on the error performance of the concatenated CRC-polar code, but still without applying asynchronous detections. The bit-arrangement of format 1 seen in Section 3.4.2 is used to obtain the error rate of the concatenated code with 8, 16, 24 and 32-bit CRCs, $N = 256$ and $R_{pc} = 0.5$.

Based on the results shown in Figure 3.10, it is observed that the error performance of the CRC-polar code gets worse as the CRC size increases. To understand the reason of this behavior, take into account that the CRC code is not doing any work for improving the error detection and the CRC-bits occupy positions of the data word in the polar codeword, see Figure 3.8a. That is, the CRC is just decreasing the concatenated code rate. In other words, as the CRC gets larger the data word is smaller. This implies that one error in a shorter transmitted data word affects (or increases) more the error rate than one error in a larger transmitted data word.

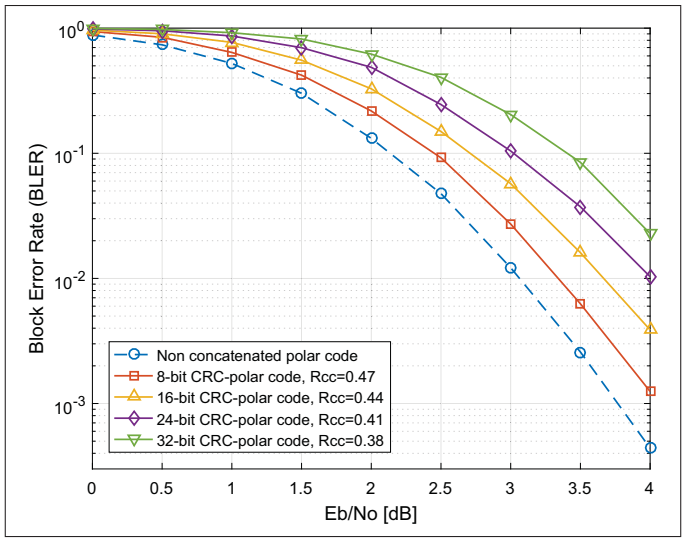


Figure 3.10 Error performance of CRC-polar codes without early detections, $N = 256$, $R_{pc} = 0.5$

3.5 Selection of CRC polynomials

Assume the CRC encoder and decoder observe a binary symmetric channel with crossover probability p, as shown in Figure 3.1. That is, the CRC outer code operates on a BSC regardless of the inner code. Under this initial assumption, to develop the selection of CRC polynomials on a CRC-based early detection scheme, consider the CRC sizes: 8, 10-16, 24 and 32 bits. For each selected CRC size, a standard CRC polynomial taken from Cook (2016) and three of the best CRC polynomials taken from Koopman (2016) tables are considered. Taking into account the data word length and the available polynomials published by Koopman, CRC polynomials with Hamming distances between three and eight are considered to be analyzed. These polynomial are detailed in Table 3.2.

The selection of CRC polynomials of a specific CRC size is initially based on the comparison of their error and average latency results obtained by simulations on the early detection scheme. This work is developed for two polar codeword block lengths, 256 and 512, under an observed SNR range from 0 to 4 dB and employing the bit-arrangement of format 1. Moreover, it is important to declare that the simulations of the early detection scheme are developed under an

initial detection time (τ_1) of 50% of the symbol period T and posterior detections with fixed increments ($\Delta\tau$) of 5% of T, these variables are analyzed in Sections 3.7 and 3.9.

To start the CRC polynomial selection, consider the results obtained when $N = 256$ and a CRC size is equal to 8, shown in Figure 3.11a. The error performance of different 8-bit CRC polynomials is similar for low SNRs (< 1.5dB). However, for higher SNRs there are polynomials with different error performances. This means that the selection of a good generator polynomial for 8-bit CRCs is only justifiable for the higher SNRs of the observed range. On the other hand, the average latencies of the four 8-bit CRC polynomials have similar performance throughout the observed range of SNRs, see Figure 3.12a.

In general, as the CRC size increases, the difference between the error performance of the available polynomials disappears. Specifically, when $N = 256$ and the CRC size is less than 12, there are different polynomial error performances at some point within $[0,4]$dB. However, if the CRC size is equal or larger than 12, the error rates of the different polynomials are practically the same in all points of the observed SNR range, see Figure 3.11. For $N = 512$, the error performance of the polynomials is similar starting from CRC-15, see Figure 3.13. Moreover, seeing all the error performances of the different CRC sizes shown in Figures 3.11 and 3.13, the block-error rate improves as the CRC sizes increases until certain CRC size, and after this, the error rates degrade. These findings are described in Table 3.4 and are reviewed in the next section. Regarding the average latency results of the selected polynomials, they are similar at each CRC size, from 8 to 32 bits, either for $N = 256$ or $N = 512$, see Figures 3.12 and 3.14. On the other side, the average latency of all CRC sizes degrades as the CRC size increases. Therefore, in the setting of the CRC-based sequential early detection scheme, a CRC polynomial selection should be performed when the chosen CRC size is less than 12 and 15 for $N = 256$ and $N = 512$, respectively. Otherwise, the CRC polynomial selection is not necessary.

According to the BLER and average latency results of sequential early detections under different CRC polynomials, at least two good generator polynomials are identified for each CRC

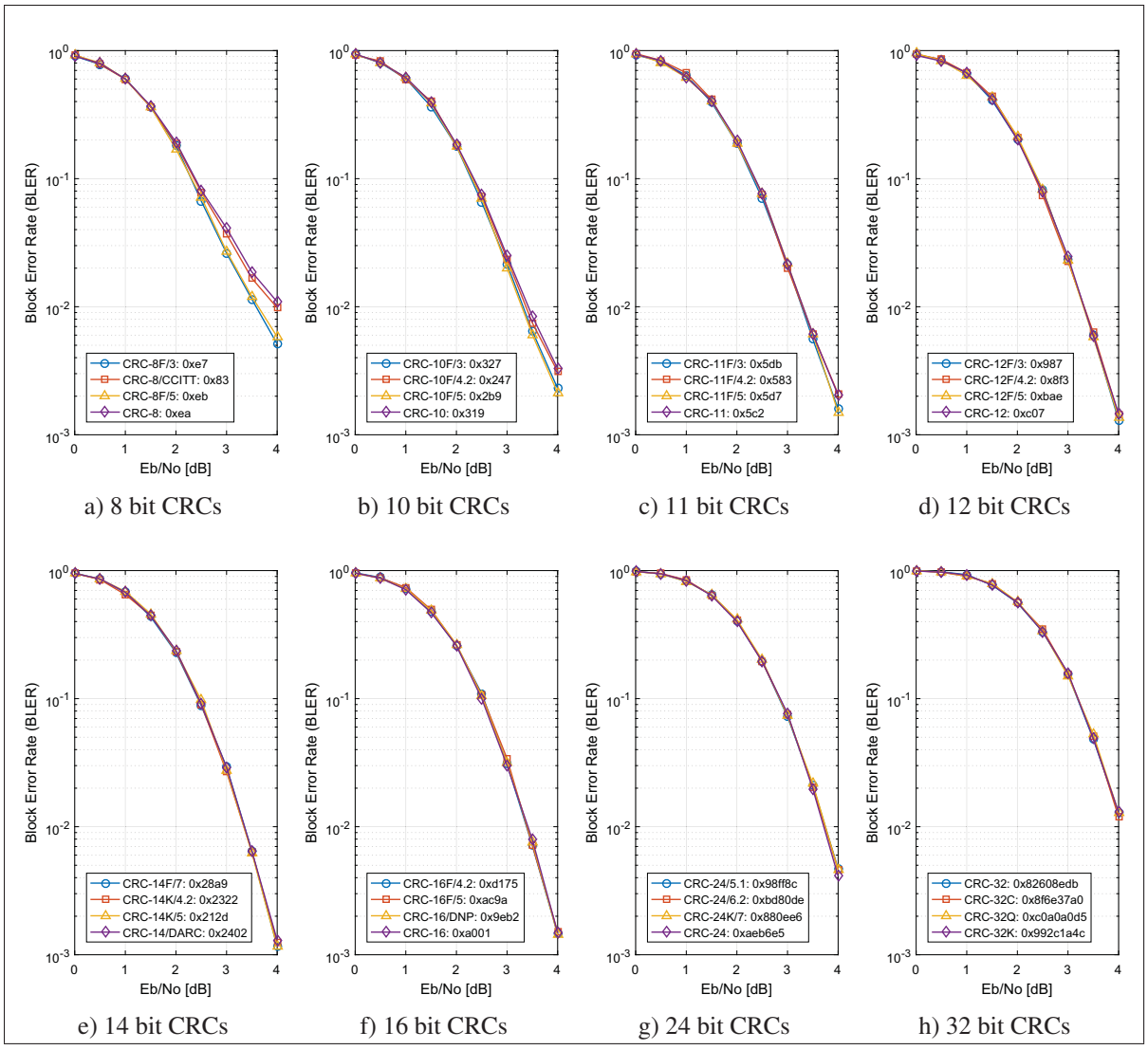


Figure 3.11 Error performance of sequential early detections with different generator polynomials of CRC codes, $N = 256$, $R_{pc} = 0.5$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

size. This determination is only based on the error rate comparison of the four polynomials evaluated at each CRC size, since their average detection performances are similar, see Figures 3.11, 3.12, 3.13 and 3.14. Therefore, the polynomials with the lowest error rates within [0,4]dB are selected as good generator polynomials of a specific CRC size and block length N. For example, for $N = 512$, the polynomials CRC-10F/3: 0x327 and CRC-10F/5: 0x2b9 are considered good generator polynomials for 10-bit CRCs, since their respective block-error rates are better than the polynomials CRC-10F/4.2: 0x247 and CRC-10: 0x319. Evidently, this

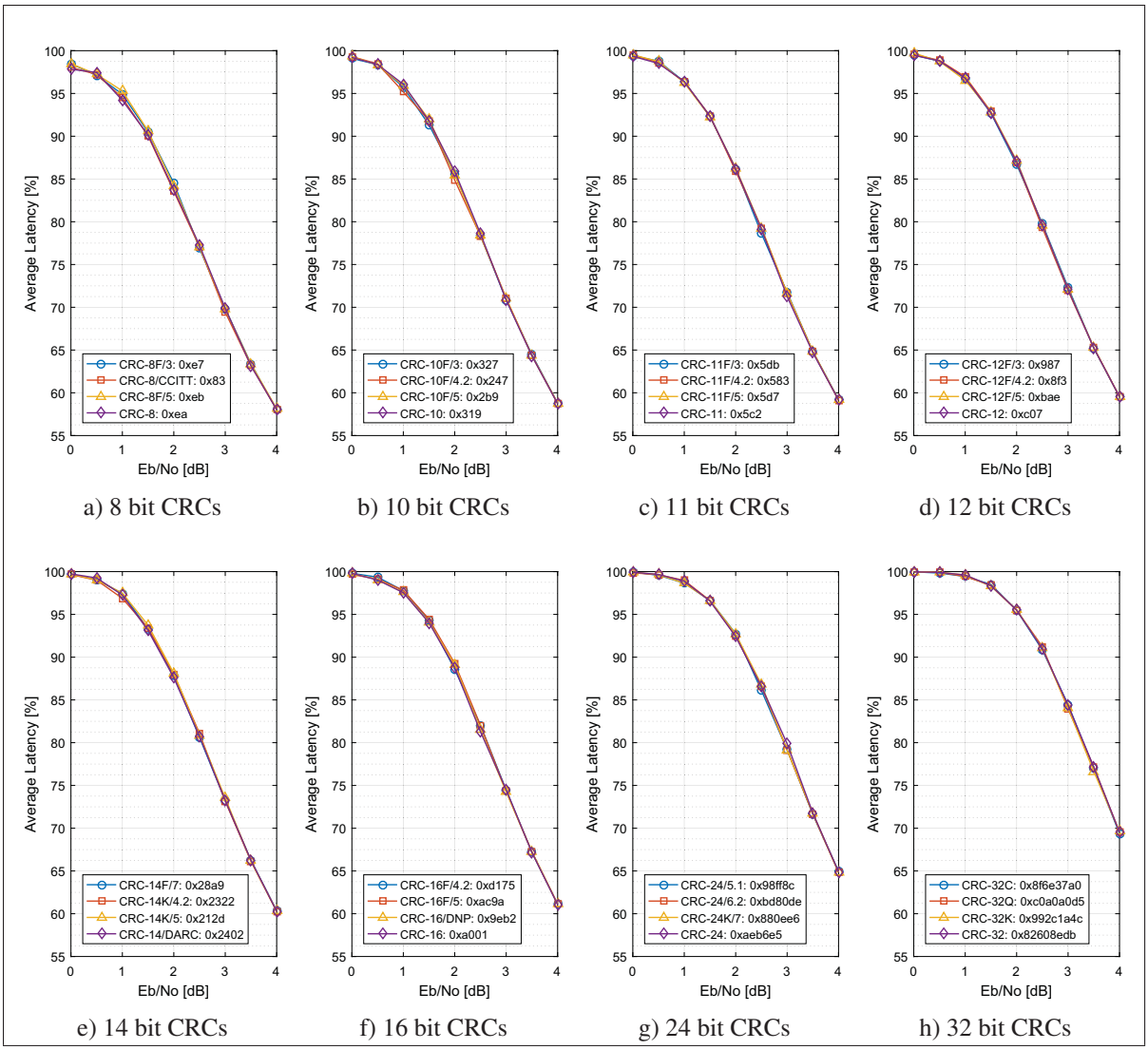


Figure 3.12 Average latency of sequential early detections with different generator polynomials of CRC codes, $N = 256$, $R_{pc} = 0.5$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

comparison is valid when the error performances are different, which happens until a specific CRC size, as was explained before.

Note that the error performance of the good polynomials is similar although they have different Hamming distances, as shown in Figures 3.11 and 3.13. This is because the polynomial with lower HD and longer data words works better (in terms of undetected error probability) than the

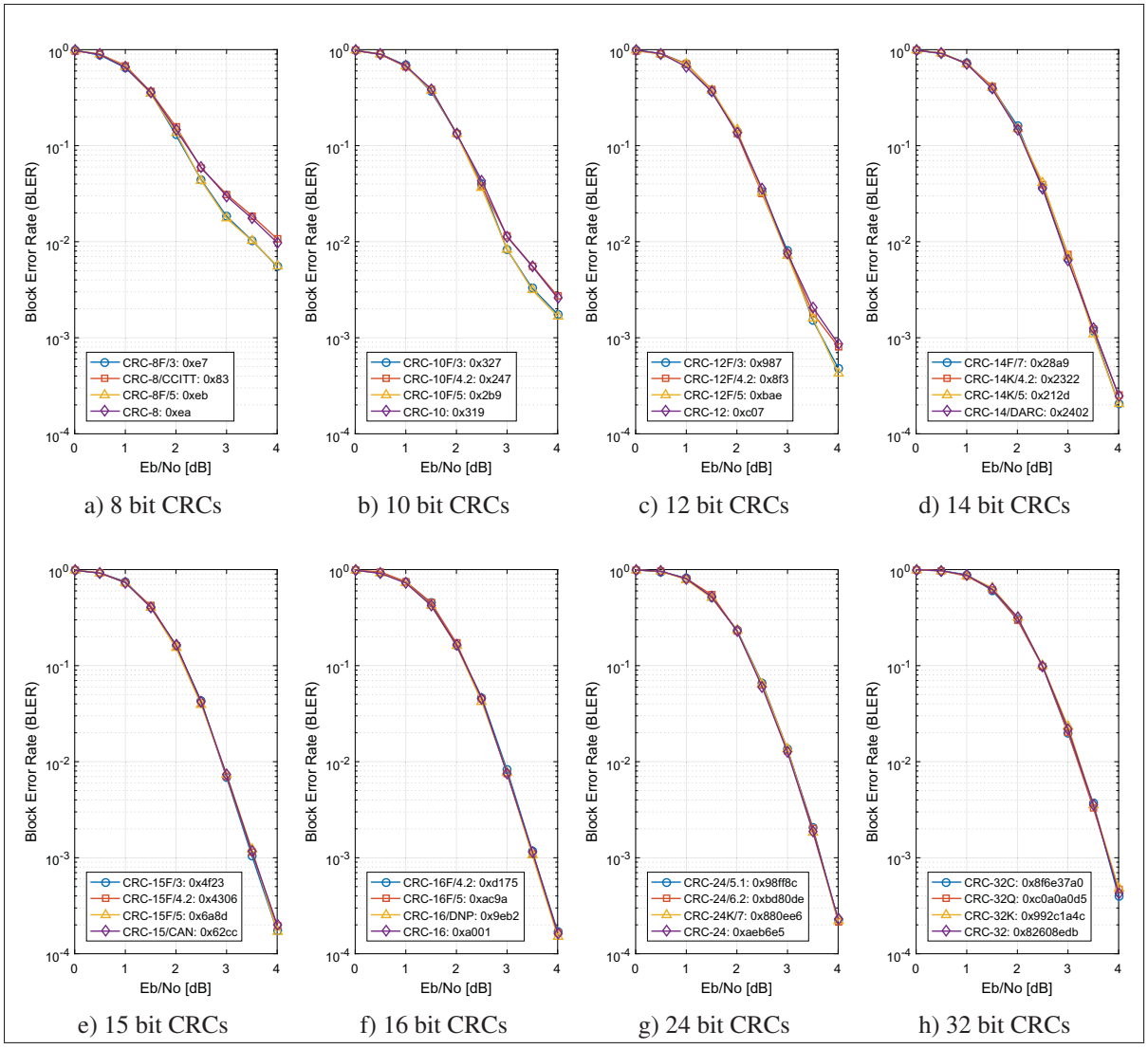


Figure 3.13 Error performance of sequential early detections with different generator polynomials of CRC codes, $N = 512$, $R_{pc} = 0.5$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

polynomial with higher HD and shorter data words. So that, there is a compensation between the HD and the maximum data word length, defined by Koopman & Chakravarty (2004).

For further analysis, one of the two good polynomials are selected at each CRC size where the CRC polynomial selection is necessary. For the rest of CRC sizes where is not required a polynomial selection, any of the polynomials can be considered. In this work, standard CRC polynomials are chosen, when a polynomial selection is not required. The list of the CRC

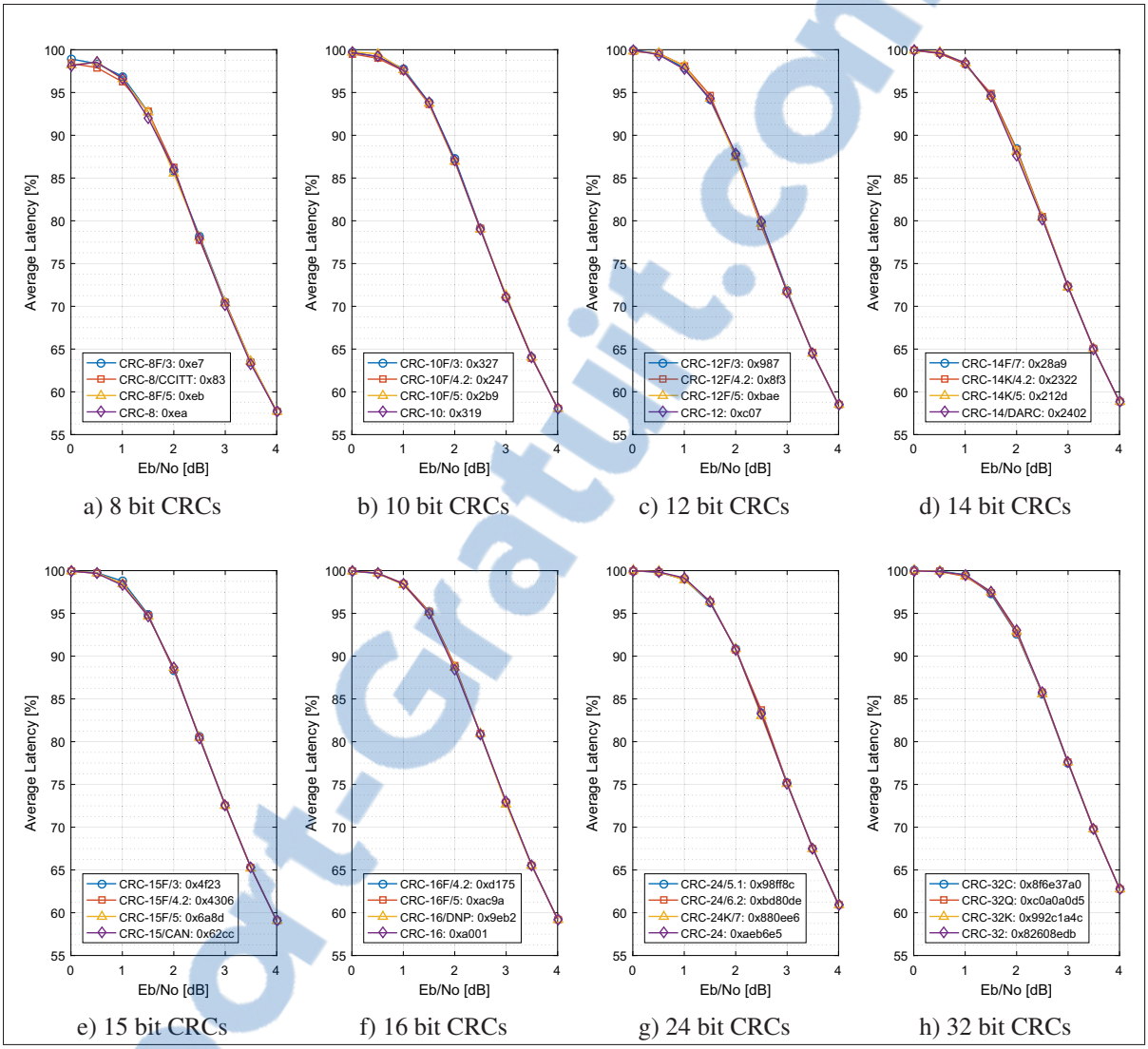


Figure 3.14 Average latency of sequential early detections with different generator polynomials of CRC codes, $N = 512$, $R_{pc} = 0.5$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

polynomials compared at each CRC size from 8 to 32 is shown at Table 3.2, where each polynomial is represented with Koopman notation. This table includes the nickname, maximum data word length, and Hamming Distance of each CRC polynomial. This information is taken from the best CRC polynomials published by Koopman (2016). Furthermore, the Table 3.2 indicates which CRC polynomial obtains a good block-error rate and which polynomial has been selected for each CRC size for a given blocklength, either $N = 256$ or $N = 512$.

Table 3.2 Selection of CRC polynomials for sequential early detection scheme
Adapted from Koopman (2016) and Cook (2016)

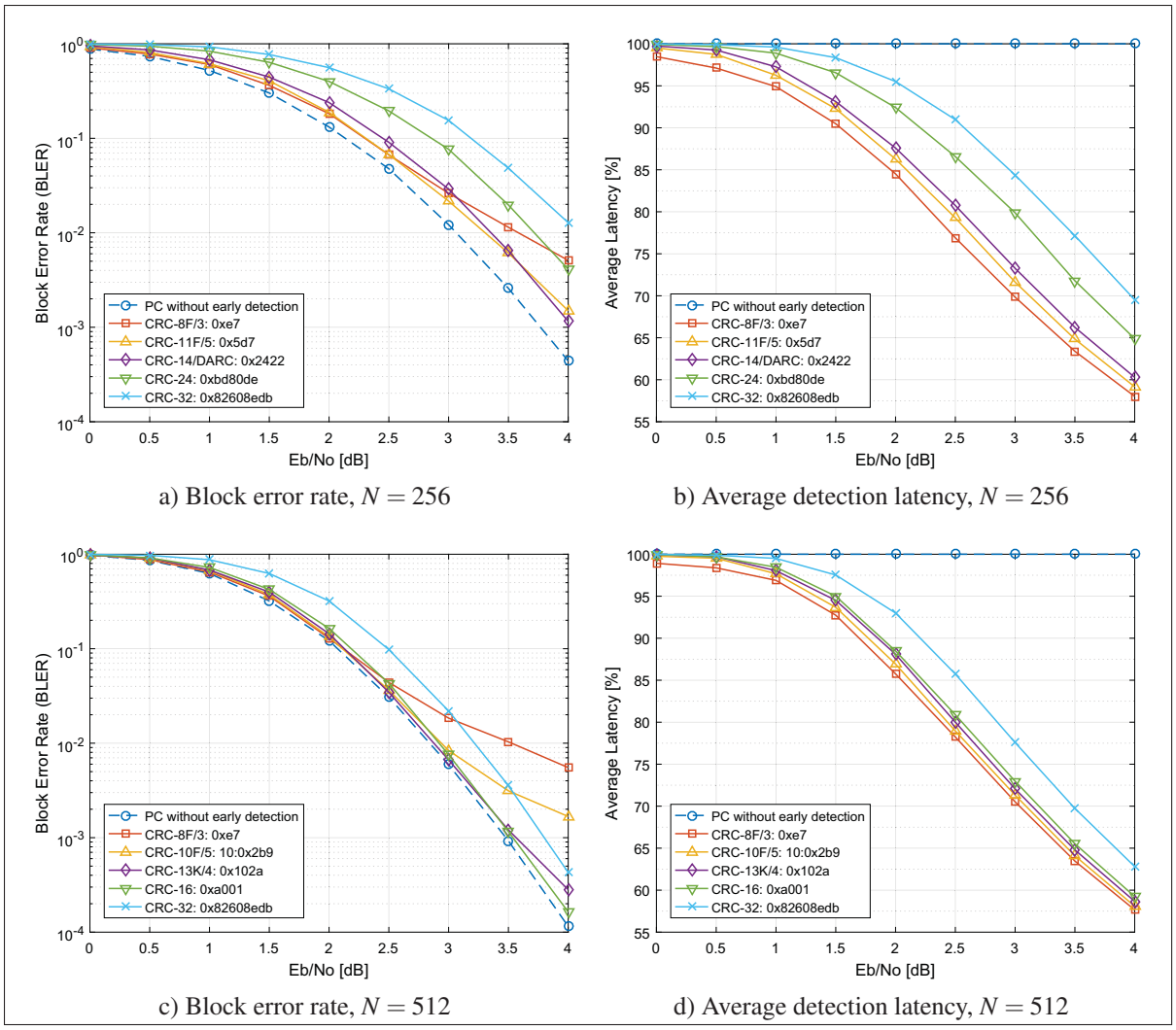
CRC size	Nickname	Polynomial	Max length at HD	HD	Good BLER		Selected poly.	
					N=256	N=512	N=256	N=512
8	CRC-8F/3	0xe7	247	3	X	X	X	X
8	CRC-8/CCITT	0x83	119	4				
8	CRC-8F/5	0xeb	9	5	X	X		
8	CRC-8	0xea	85	5				
10	CRC-10F/3	0x327	1013	3	X	X		
10	CRC-10F/4.2	0x247	501	4				
10	CRC-10F/5	0x2b9	21	5	X	X	X	X
10	CRC-10	0x319	501	4				
11	CRC-11F/3	0x5db	2036	3	X	X		
11	CRC-11F/4.2	0x583	1012	4				
11	CRC-11F/5	0x5d7	26	5	X	X	X	X
11	CRC-11	0x5c2	20	6				
12	CRC-12F/3	0x987	4083	3	X	X		
12	CRC-12F/4.2	0x8f3	2035	4	X			
12	CRC-12F/5	0xbae	53	5	X	X		X
12	CRC-12	0xc07	2035	4	X		X	
13	CRC-13F/3	0x1abf	8178	3	X	X		
13	CRC-13K/4	0x102a	2542	4	X	X		X
13	CRC-13F/6.2	0x1e97	52	6	X			
13	CRC-13/BBC	0x1e7a	165	4	X		X	
14	CRC-14F/7	0x28a9	16369	3	X	X		
14	CRC-14K/4.2	0x2322	8177	4	X			
14	CRC-14K/5	0x212d	113	5	X	X		X
14	CRC-14/DARC	0x2402	8177	4	X		X	
15	CRC-15F/3	0x4f23	32752	3	X	X		
15	CRC-15F/4.2	0x4306	16368	4	X	X		
15	CRC-15F/5	0x6a8d	136	5	X	X		
15	CRC-15/CAN	0x62cc	112	6	X	X	X	X
16	CRC-16F/4.2	0xd175	32751	4	X	X		
16	CRC-16F/5	0xac9a	241	5	X	X		
16	CRC-16/DNP	0x9eb2	135	6	X	X		
16	CRC-16	0xa001	32751	4	X	X	X	X
24	CRC-24/5.1	0x98ff8c	4073	5	X	X		
24	CRC-24/6.2	0xbd80de	2026	6	X	X		
24	CRC-24K/7	0x880ee6	231	7	X	X		
24	CRC-24	0xaeb6e5	2024	6	X	X	X	X
32	CRC-32C	0x8f6e37a0	5243	6	X	X		
32	CRC-32Q	0xc0a0a0d5	2275	6	X	X		
32	CRC-32K	0x992c1a4c	134	8	X	X		
32	CRC-32	0x82608edb	2974	5	X	X	X	X

3.6 Performance of the early detection scheme with appropriate CRC polynomials

After the selection of polynomials for each CRC size (if required according to CRC size), their block-error rates and average latency performances are compared. As shown in Figure 3.15a, when $N = 256$ and $R_{pc} = 0.5$, the best error performance at high SNRs ((3.55,4]dB) is achieved by 14-bit CRC. Whereas for $N = 512$, this happens at 16-bit CRC, see Figure 3.15c. However, the average latency performances of 14 and 16 bit CRCs, respectively, are not the best among all the simulated CRCs, as shown in Figures 3.15b and 3.15d. CRC codes with smaller check sequences obtain better average latency performances. For instance, when $N = 512$ and SNR = 4dB, the selected CRC-8 and CRC-12 obtain 57.65% and 58.48% of average detection latency, respectively, while the CRC-16 obtains 59.24% of average latency. On the other side, if a low CRC size (8-bit) is used to achieve the best possible average latency, a bad error performance is obtained at high SNRs. However, the CRC-8F/3 for both $N = 256$ and $N = 512$ provides the best BLER at low SNRs. Therefore, to obtain on average an acceptable error and detection latency along low and high SNRs ([0,4]dB), it is necessary to find a trade-off between these results. From the obtained results, for $N = 256$, we consider that an 11-bit CRC is a good option to maintain a relative low error rate in the whole SNR range and achieve a good average detection latency. In fact, the 11-bit CRC is the best CRC for intermediate SNRs when $N = 256$. For $N = 512$, a 13-bit CRC provides the lowest error rate at intermediate SNRs, while relative low error rates and average latencies are achieved in the rest of SNRs, see Figures 3.15c and 3.15d. Table 3.3 summarizes the CRC polynomials that obtain the lowest error rate and low average latency for different ranges of SNR, with an IDT fixed to $\tau_1 = 50\%$ and $\Delta\tau = 5\%$.

Table 3.3 CRC polynomials of the early detection scheme for low, medium and high SNRs from 0 to 4dB, $\tau_1 = 50\%$, $\Delta\tau = 5\%$, and $R_{pc} = 0.5$

N	Low SNRs		Medium SNRs		High SNRs	
	CRC Polynomial	Range [dB]	CRC Polynomial	Range [dB]	CRC Polynomial	Range [dB]
256	CRC-8F/3	[0;2.5]	CRC-11F/5	(2.5;3.55]	CRC-14/DARC	(3.55;4]
512	CRC-8F/3	[0;2.15]	CRC-13K/4	(2.15;3.35]	CRC-16	(3.35;4]



a) Block error rate, $N = 256$ b) Average detection latency, $N = 256$

c) Block error rate, $N = 512$ d) Average detection latency, $N = 512$

Figure 3.15 Performances of sequential early detections using preselected CRC polynomials, for block lengths 256 and 512, $R_{pc} = 0.5$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

For the analysis of the average latency results, consider that they are expressed in terms of percentage (%) with respect to the OFDM symbol period. That is, 100% represents the whole OFDM symbol duration T. Taking as reference the average latency value obtained by the CRC-16 with $N = 512$, a transmitted message might be detected using 59.24% of the duration of the symbol period. In other words, the detection of a polar codeword with block length 512 can be performed by saving 40.76% of time at 4dB, if a sequential early detection is applied with CRC-16. For $N = 256$, the obtained latency reduction is 39.69% at 4dB of SNR, if the best CRC polynomial determined for high SNRs is used, *i.e.* CRC-14/Data Radio Channel

(DARC). These results indicate that an early detection scheme based on CRC-polar codes can achieve a significant latency reduction at SNR equal to 4dB.

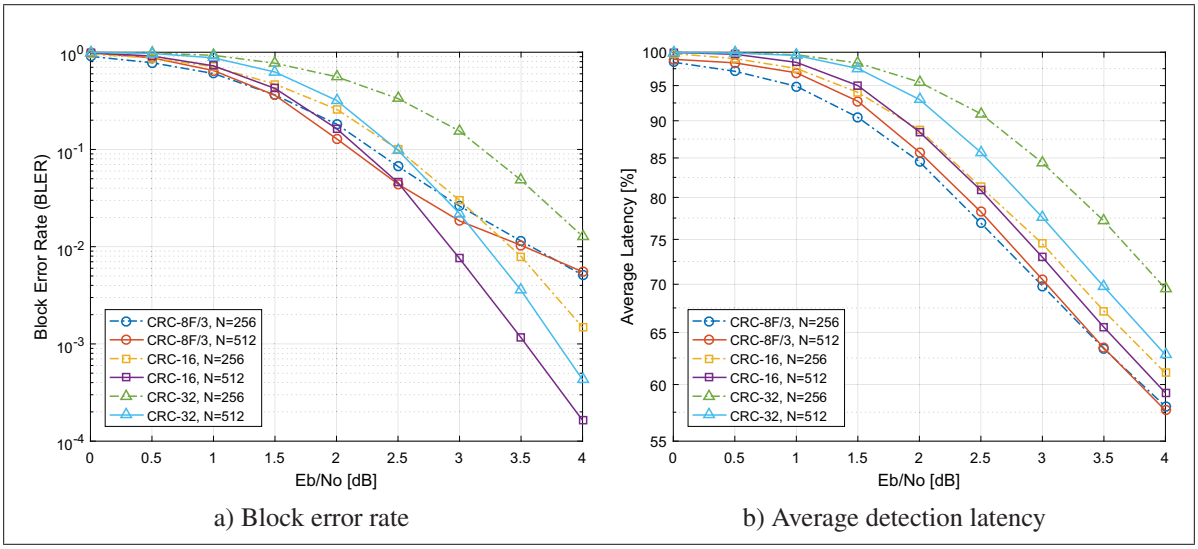
From Figures 3.15b and 3.15d , it is clear that average latency performances follow only one behavior. The average latency performance gets worse as the used CRC size grows. On the other side, block-error rates take two behaviors according with the length of the CRC check sequence. These behaviors are separated by the CRC that achieves the best error performance at the highest SNR value of the observed range. That is, the CRC breakpoint for $N = 256$ is CRC-14/DARC and for $N = 512$ is CRC-16, see Table 3.4. To show the first behavior, let's compare two CRC error performances, where the higher CRC is at most the CRC breakpoint. The block-error rate of the lower CRC size has a better performance than the higher CRC at low SNRs, whereas the higher CRC has a better error performance at high SNRs. For example, for $N = 256$, the CRC-8F/3 has a better error performance than the CRC-14/DARC from 0 to 3.1dB. Whereas the CRC-14/DARC has a better error performance than the CRC-8F/3 from 3.1 to 4dB, see Figure 3.15a. For the explanation of the second error rate behavior of CRCs, let's again compare two CRC error performances, but in this case, the lower CRC is at least the CRC breakpoint. In this situation, the higher CRC has a worse error performance than the lower CRC along the whole observed range of SNRs. An example of this can be seen by comparing the error performances of the CRC-16 and the CRC-32 when $N = 512$, as shown in Figure 3.15c. Findings of the CRC polynomial selection process are summarized in Table 3.4.

Table 3.4 CRCs with special performance characteristics over a CRC-based sequential early detection scheme with different block lengths and $R_{pc} = 0.5$

Performance characteristic	***n*-bit CRC**	
	N=256	**N=512**
Similar BLERs of all evaluated CRC polynomials with the same degree	12-bit CRC	15-bit CRC
Best BLER at the highest SNR 4dB / Breaking point of BLER behaviors	14-bit CRC	16-bit CRC
Good error and latency performances in the whole range of SNRs [0,4]dB	11-bit CRC	13-bit CRC

At first sight of Figure 3.15, the performance curves of $N = 256$ look more disperse than the curves of $N = 512$. This means that the change of CRC size in the block length $N = 512$ affects

less the block-error rates and the average latencies than in the smaller block length $N = 256$. Moreover, the error performances provided by the sequential early detection scheme with block length 512 are quite close to the performance of the non-concatenated polar code (PC) without early detections. While the error performances with block length 256 are far from the error performance of the PC without early detections. Therefore, the sequential early detections schemes with higher block lengths are more likely to achieve the error performance of the PC without early detections.



a) Block error rate b) Average detection latency

Figure 3.16 Performance of sequential early detections based on 8,16 and 32 bit CRCs, block lengths 256 and 512, $R_{pc} = 0.5$, $\tau_1 = 50\%$, and $\Delta\tau = 5\%$

Now, let's compare the block-error rates and average latencies of sequential early detections with the same CRC size, the same polar code rate (R_{pc}) and different block lengths, $N = 256$ and $N = 512$, as shown in Figure 3.16. The error rate of the larger block length is better than the error rate with a small block length, specially as the CRC size increases. On the other side, the average latency performance depends significantly on the CRC size. When the CRC size is small, *e.g.* 8-bit CRC, the average latency of the smaller block length is better than the other one in almost the whole range of SNRs. But if the CRC size is large, *e.g.* 32-bit CRC, the average latency of the larger block length is better than the smaller block length. In accordance with this behavior, when the CRC size is an intermediate value such as 16, the average latency

of the larger block length is better than the other one just in the half of the SNR range, at high SNRs ([2,4]dB), as shown in Figure 3.16.

As a result, a large block length obtains a better average latency and error rate results than a small block length, specially at high SNRs. Therefore, for the design of a multicarrier system with sequential early detections, a large number of subcarriers is recommended.

3.7 Selection of the initial detection time of sequential early detections

Another important variable to consider in the setting of the CRC-based sequential early detection scheme is the initial detection time (IDT) denoted as τ_1, where $0 < \tau_1 < T$. All previous simulations of early detections start at 50% of the symbol period. This value was taken because the expected latency is around it, as we will see in the results of this section. On the other hand, to find a suitable initial detection time is necessary to determine a trade-off between the average detection latency and the BLER. Normally, a better average latency is obtained at the expense of a degradation in error performance. Therefore, the initial detection time can be defined as a time threshold that allows to achieve the best possible error and latency performances. In order to obtain the detection start point, we explore a little bit more the average latency and error performances with different initial detection times. From the lowest to the highest initial detection time, the detection time that generates simultaneously more decrement in BLER and less increment in average latency is selected as the appropriate initial detection time.

To determine the initial detection time of the sequential detection, consider the CRC polynomials that provide low average latency and the lowest error rate for low, medium and high SNRs (in the range from 0 to 4dB) determined in Section 3.6, see Table 3.3. We also consider a fourth polynomial with a higher degree for each block length, in order to analyze the IDT behavior in different CRC sizes. CRC-16 and CRC-32 are the selected polynomials for $N = 256$ and $N = 512$, respectively. The set of initial detection times used in this analysis is $\{20, 40, 45, 50, 55, 60, 80\}$, whose values are in terms of percentage of the symbol period. There are more detection times around 50% because it is the middle point to find a balance

between the BLER and the average latency. Starting with one of the selected initial detection times, the distribution of the rest of the detections is uniform with an interval of 5% of T between detections.

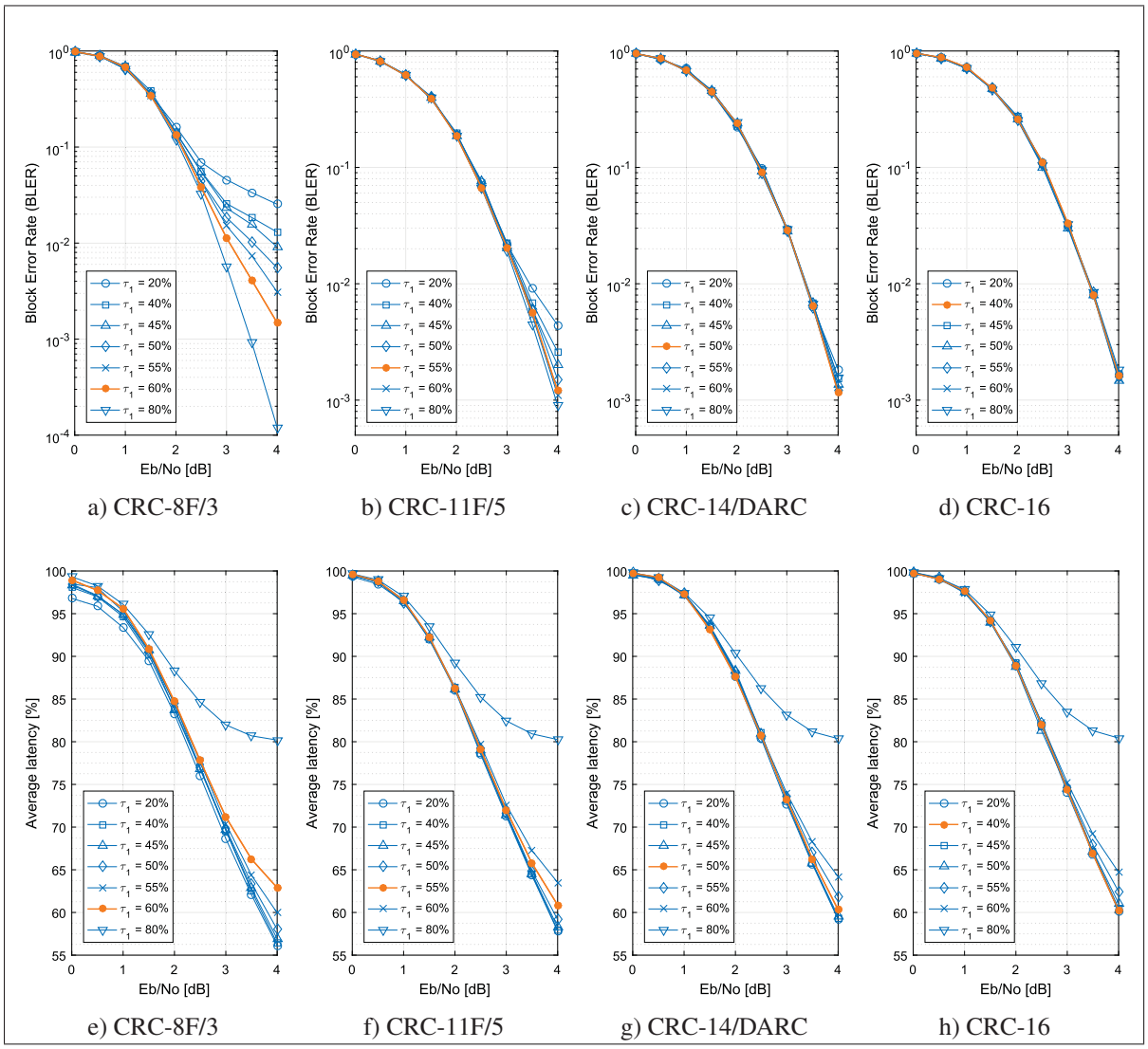


Figure 3.17 Error and latency performances of the early detection scheme under different initial detection times (τ_1) and CRC polynomials, $N = 256$, $R_{pc} = 0.5$, $\Delta\tau = 5\%$

As expected, the BLER decreases and the average latency increases as the initial detection time grows. This is clearly observed in performances obtained by CRC-8F/3, CRC-11F/5 and

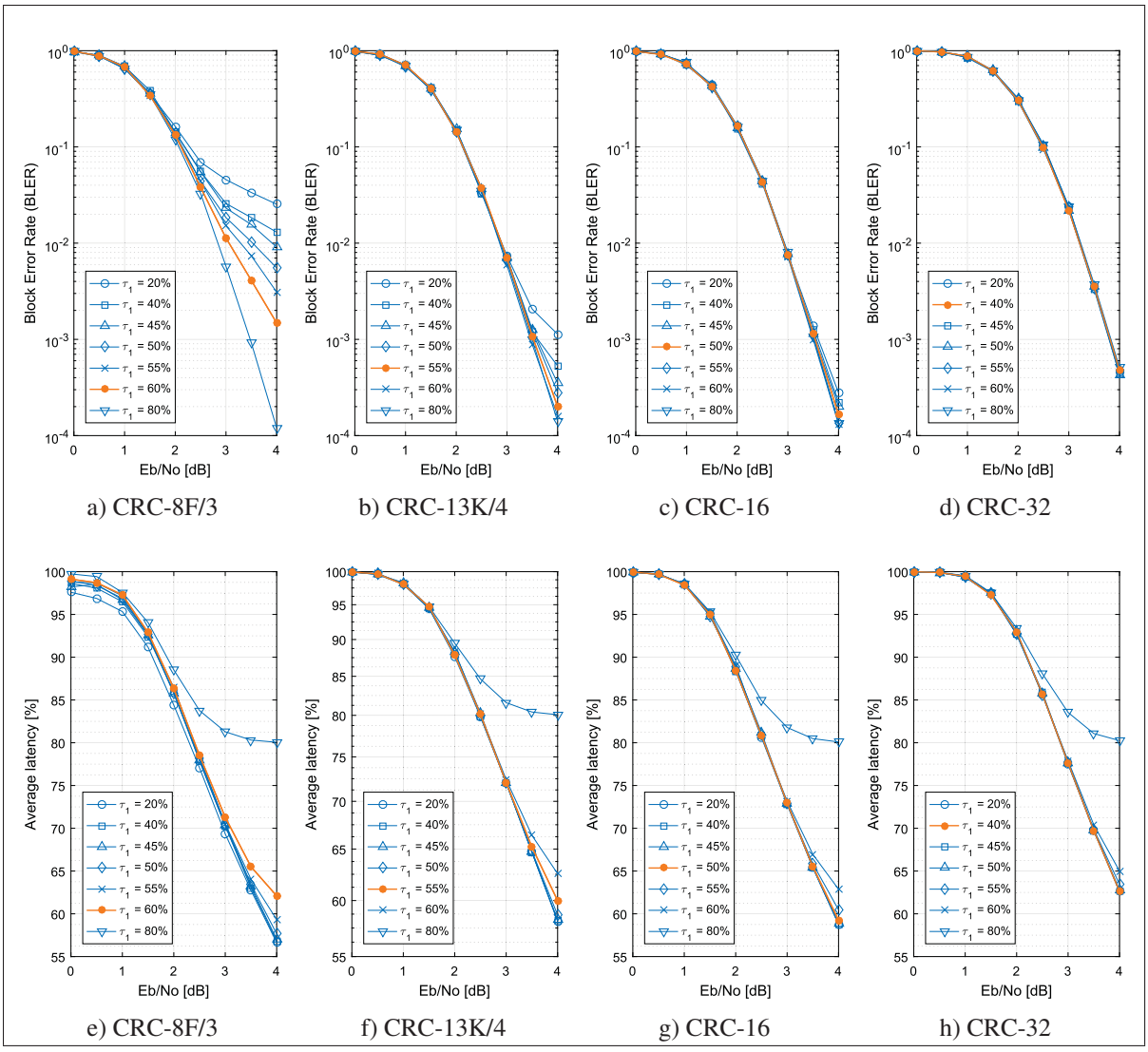


Figure 3.18 Error and latency performances of the early detection scheme under different initial detection times (τ_1) and CRC polynomials, $N = 512$, $R_{pc} = 0.5$, $\Delta\tau = 5\%$

CRC-13K/4 [1] shown in Figures 3.17 and 3.18. An initial detection time of 60% is selected for CRC-8F/3 with both block lengths. While for CRC-11F/5 with $N = 256$ and CRC-13K/4 with $N = 512$, the selected IDT is 55%. For CRC-14/DARC, the error performances of different detection times are quite similar, just at 4dB of SNR, they are slightly different. The lowest BLER of the 14-bit CRCs is achieved with an IDT of 50% at 4dB, see Figure 3.17c. On the other side, the resulting average latencies are low when the initial detection time is less than

[1] CRC-13K/4 is one of the CRC polynomials published by Koopman & Chakravarty (2004)

or equal to 50%, see Figure 3.17g. Thus, the selected starting detection time of the 14-bit CRC with $N = 256$ is 50%. The CRC-16 with $N = 512$ has the same behavior of the CRC-14/DARC with $N = 256$, see Figures 3.18c and 3.18g. Consequently, the recommended IDT for the CRC-16 with $N = 512$ is also 50%.

For CRC-16 with $N = 256$ and CRC-32 with $N = 512$, all the analyzed initial detection times have similar BLERs, even at 4dB, see Figures 3.17d and 3.18d. Besides, their average latencies are also similar, but just for IDTs equal to or less than 50%, see Figures 3.17h and 3.18h. Therefore, any IDT equal to or less than 50% is a good option for these polynomials. We have decided to declare 40% as the convenient IDT for CRC-16 and CRC-32 with block lengths 256 and 512, respectively. This decision is based on our premise to maintain a balance between the error and latency performances, although there is not a big difference.

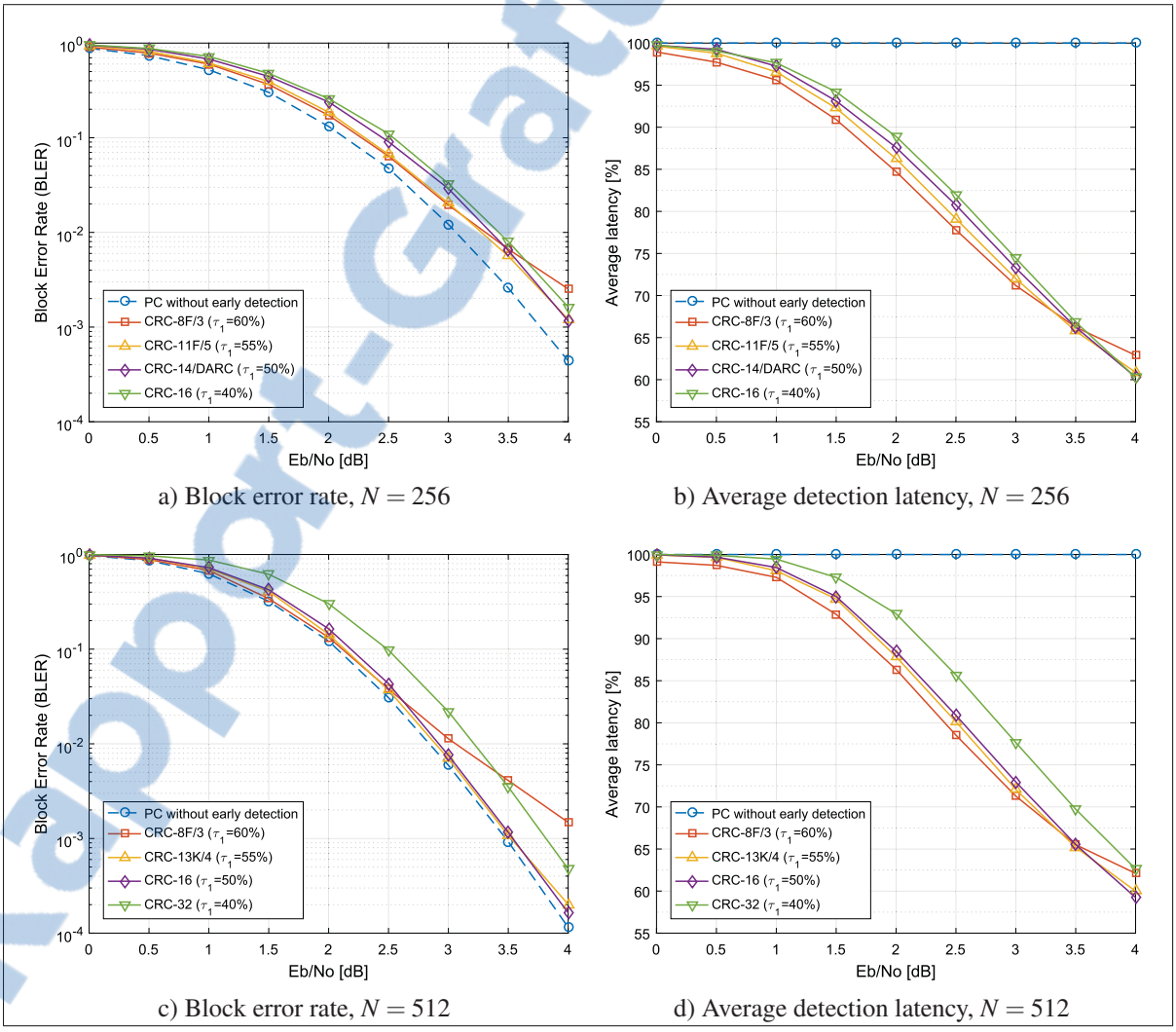
Based on these results, we conclude that the block-error rate of an early detection scheme is less influenced by the initial detection time as the CRC size is larger. The same happens with the average detection latency for IDTs less than 50%. Consequently, if an early detection scheme employs a CRC polynomial with a degree greater than 14 for $N = 256$ or 16 for $N = 512$, an IDT equal to 50% can be used to obtain good error and latency performances.

3.8 Performance of the early detection scheme with suitable IDTs

In this section, consider the final results of the CRC-based early detection scheme in terms of block-error rate and average latency for block lengths of 256 and 512, and $R_{pc} = 0.5$, shown in Figure 3.19. These results are obtained after the selection of the best CRC polynomials and their appropriate initial detection times.

For $N = 256$, the CRC-8F/3 with an initial detection time of 60% obtains the best performance in terms of block-error rate from 0 to 3.2dB. The best BLERs at high SNRs from 3.5dB to 4dB is obtained by the CRC-11F/5 and CRC-14/DARC, with initial detection times of 55% and 50%, respectively, see Table 3.5. Besides, the BLER of CRC-11F/5 follows very close the error performance curve of CRC-8F/3 at lower SNRs, see Figure 3.19a. Also note that

CRC-14/DARC only offers a good error performance at the highest SNRs, for SNRs lower than 3.5dB it results in bad BLERs similar as the CRC-16. For $N = 256$, the CRC-16 with an IDT of 40% generates the worst error performance among the four CRC polynomials selected for this analysis. This happens even if the IDT is higher, because as we see in Figure 3.17d all the IDTs from CRC-16 produce similar BLERs. Furthermore, no error performance of the analyzed CRCs achieves the BLER of a non-concatenated polar code (PC) without early detections, when the block length is 256.



a) Block error rate, $N = 256$

b) Average detection latency, $N = 256$

c) Block error rate, $N = 512$

d) Average detection latency, $N = 512$

Figure 3.19 Performance of sequential early detections after the selection of CRC polynomials and IDTs (τ_1), for block lengths 256 and 512, $R_{pc} = 0.5$, $\Delta\tau = 5\%$

With respect to the average detection latency obtained by the mentioned CRC polynomials shown in Figure 3.19b, the CRC-8F/3 has the lowest early detection at lower and medium SNRs, specifically from 0 to 3.3dB. The CRC-11F/5, CRC-14/DARC and CRC-16 result in similar good average latencies at high SNRs starting from 3.5dB. But the best average latencies at 4dB is achieved by CRC-14/DARC and CRC-16 with 60.31% and 60.26%, respectively, see Table 3.5.

Table 3.5 Error and latency performances of the early detection scheme under suitable IDTs of different CRCs when SNR = 4dB, for $N = \{256, 512\}$ and $R_{pc} = 0.5$

Blocklength (N)	**CRC polynomial**	**IDT** (τ_1)	**Performance**	
			BLER	**Avg. Latency**
256	CRC-8F/3	60%	$2.53 * 10^{-3}$	62.89%
	CRC-11F/5	55%	$1.21 * 10^{-3}$	60.87%
	CRC-14/DARC	50%	$1.16 * 10^{-3}$*	60.31%
	CRC-16	40%	$1.63 * 10^{-3}$	60.26%*
512	CRC-8F/3	60%	$1.48 * 10^{-3}$	62.10%
	CRC-13K/4	55%	$2.01 * 10^{-4}$	59.99%
	CRC-16	50%	$1.64 * 10^{-4}$*	59.24%*
	CRC-32	40%	$4.77 * 10^{-4}$	62.64%

* Best performance

For $N = 512$, the best BLER performance at low SNRs, from 0 to 2.45dB, and the worst BLER performance at the highest SNRs, from 3.4 to 4dB, results from the CRC-8F/3 with an IDT of 60%, see Figure 3.19c. The CRC-16 with an IDT of 50% obtains the best BLER at the highest SNRs from 3.7 to 4dB. Although, the BLER of CRC-16 is following closely the error performance curve of the PC without early detections, there are other polynomials that offer a slightly better BLER at intermediate and low SNRs, such as CRC-13K/4 and CRC-8F/3. At intermediate SNRs, the error performance of the CRC-13K/4 with an IDT of 55% is closer to the non-concatenated PC performance than the CRC-16. On the other side, the CRC-32 with an IDT equal to 40% has a remarkable difference with the rest of chosen CRCs, ending with the worst error performance from 0 to 3.4dB. With respect to the latency performances obtained by the CRC polynomials with a block length of 512, the best average latency at low and intermediate SNRs ([0,3.5]dB) is obtained by the CRC-8F/3, see Figure 3.19d. The CRC-

16 generates the best average latency at 4dB with 59.24% of T, see Table 3.5. The CRC-13K/4 has a similar latency performance with CRC-16 at high SNRs from 3.5 to 4dB (with 59.99% at 4dB), and it has a better performance than the CRC-16 at SNRs lower than 3.5dB. Table 3.5 presents the resulting error and latency performances of the early detection scheme under appropriate initial detection times for SNR equal to 4dB.

Comparing the best BLER performances of the two analyzed block lengths, the CRCs with the larger block length (512) obtain better results. Take as reference that the BLER of a non-concatenated PC without early detections with $N = 512$ has a better performance at intermediate and high SNRs than with $N = 256$. This is clearly observed at Figure 3.19c, where the error performances of CRC-13K/4 and CRC-16 are close to the non-concatenated PC performance. For instance, at the highest SNR (4dB) of our analysis, the best BLER with $N = 512$ $(1.64 * 10^{-4})$ approaches to the BLER of a non-concatenated polar code without early detections $(1.15 * 10^{-4})$. While CRC performances with the smaller block length (256) are further from the non-concatenated PC performance, specially at high SNRs, see Figure 3.19a. For example, the best BLER with $N = 256$ $(1.16 * 10^{-3})$ at 4dB is farther from the BLER of the scheme without early detections $(4.40 * 10^{-4})$. On the other hand, to compare the latency performances of both block lengths, suppose vertical lines are drawn at SNRs equal to 1, 2.5 and 3.5 dB on Figures 3.19b and 3.19d. This exercise shows that CRCs with smaller block lengths have better average latencies at low and intermediate SNRs, while CRCs with larger block lengths have better performances only at high SNRs.

In conclusion, if the objective is to obtain the best average latency and BLER at the highest SNRs by paying the price of a high error rate and a long average latency at lower SNRs, a good option is to choose the CRC-14/DARC with an IDT of 50% for $N = 256$. For $N = 512$, the same performance is given by the CRC-16 with an IDT equal to 50%. But if the goal is to achieve the best possible BLER and average latency at low SNRs regardless the performance at higher SNRs, then the CRC-8F/3 with an IDT of 60% should be selected for $N = 256$ or $N = 512$. The best error performance at intermediate SNRs is achieved by the CRC-13K/4

with an IDT of 55% for $N = 256$; while for $N = 512$, this is carried out by the CRC-11F/5 with the same IDT.

Table 3.6 Recommended setting for the CRC-based early detection scheme to obtain the best possible performance at low, medium and high SNRs, with $R_{pc} = 0.5$, and $\Delta\tau = 5\%$

N	Low SNRs			Medium SNRs			High SNRs		
	CRC Poly.	τ_1 **[%]**	**Range [dB]**	**CRC Poly.**	τ_1 **[%]**	**Range [dB]**	**CRC Poly.**	τ_1 **[%]**	**Range [dB]**
256	CRC-8F/3	60	[0;3.2]	CRC-11F/5	55	(3.2;3.85]	CRC-14/DARC	50	(3.85;4]
512	CRC-8F/3	60	[0;2.45]	CRC-13K/4	55	(2.45;3.7]	CRC-16	50	(3.7;4]

On the other hand, making a balance between the obtained error rates and average latencies throughout the observed range of SNRs, for sure, the best CRCs that we can suggest is CRC-11F/5 for $N = 256$, and CRC-13K/4 and CRC-16 for $N = 512$, with their respective IDTs shown in Table 3.6. In other words, if the CRC-based early detection scheme uses this last setting, good error rates and good average latencies are obtained from 0 to 4dB. The recommended CRC polynomials and IDTs for specific ranges of SNR are summarized in Table 3.6.

3.9 Statistical average latency of different detection distributions

The statistical average latency is defined as the average latency derived with probabilities, see Equation (1.6). The block-error rate of a code is used to determine the statistical average detection latency, specifically the PMFs of early detections. In this section, the statistical average latency of different detection distributions are analyzed under two approaches. The first approach employs an arbitrary code in the finite-blocklength regime, while the second approach is under a specific CRC-polar code. The block-error rate in the finite-blocklength regime is given by the Equation (1.10), while the error rate of the CRC-polar code is obtained through simulations of this scheme without applying early detections. In both cases, consider a block length of 256 and a data word length of 120, *i.e.* $R_{cc} \approx 0.469$. A CRC of 8 bits (CRC-8F/3) is used in the concatenated CRC-polar code system without early detections, whose error performance is shown in Figure 3.10. Thus, the polar code rate of this system is $R_{pc} = 0.5$, under the

bit-arrangement of format 1, see section 3.4.2. Besides, the polar code is constructed under the Bhattacharyya algorithm, with a design-E_b/N_0 = 4dB.

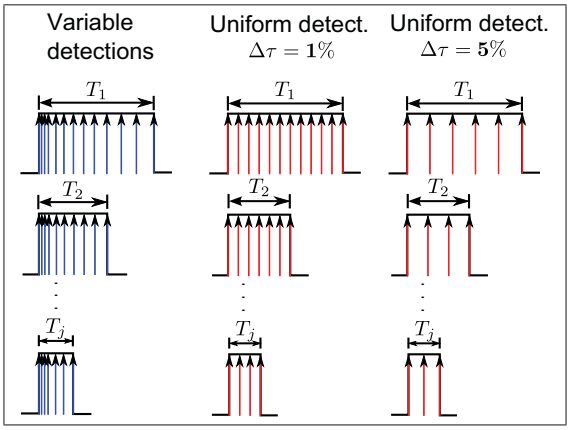
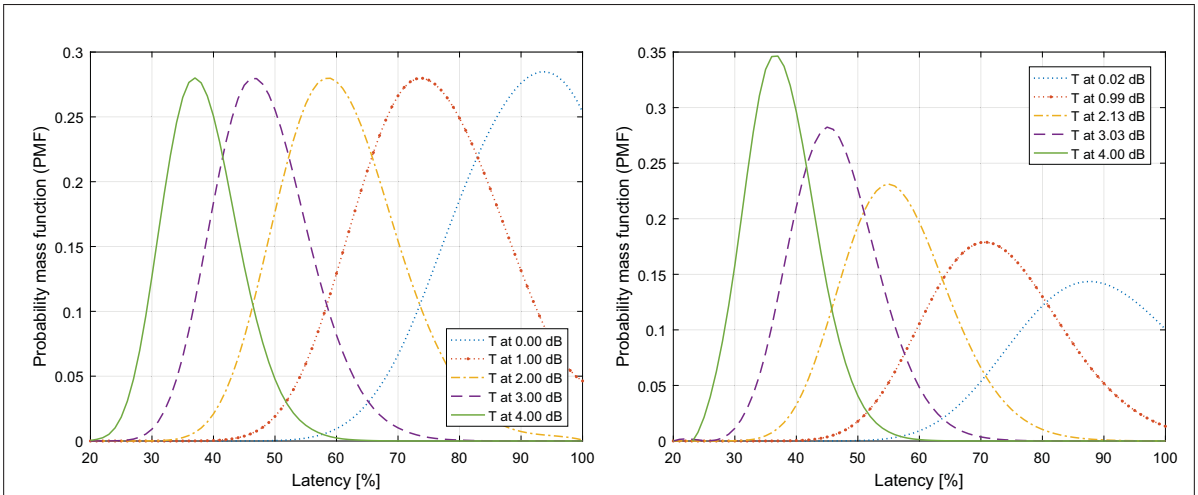


Figure 3.20 Time detection distributions

To show how the statistical average latency is affected by the distribution of asynchronous detections, three sets of detections (τ) are proposed from -3dB to 4dB in terms of SNR. The first set of detection times has increments of 0.5dB, which represents increasing intervals of time of $\Delta\tau = \{2.43\%, 2.73\%, ..., 10.87\%\}$, where each percentage is with respect to T. This means that the set of detection times is given by $\tau = \{19.95\%, 22.39\%, 25.12\%, ..., 89.13\%, 100\%\}$. The second set of asynchronous detection times has a fixed increment of $\Delta\tau = 5\%$. That is, these detections are uniform as $\tau = \{20\%, 25\%, 30\%, ..., 95\%, 100\%\}$. The third set of detection times is also uniform with increments of $\Delta\tau = 1\%$, *i.e.* $\tau = \{20\%, 21\%, 22\%, ..., 99\%, 100\%\}$. An illustration of the described detection distributions is shown in Figure 3.20. Note that the initial detection time (IDT) of both uniform detection distributions is τ_1 =20% and for the variable distribution, it is τ_1 =19.95%.

The expectation of early detections and their probability mass functions are computed similarly to the example shown in Section 1.6. The PMFs of early detections are calculated with the Equation (1.7) for specific symbol periods $\{T_1, T_2, ..., T_j\}$. The selected symbol periods match with the early detections; however, for a better illustration we show the PMF results of four of the selected symbol durations in Figure 3.21. The early detections of each new symbol period



a) PMFs using variable detections

b) PMFs using uniform detections

Figure 3.21 Probability mass functions using variable and uniform detections, deduced from BLER in the finite-blocklength regime with $N = 256$, $k = 128$ and an 8-bit CRC

maintain the variable and fixed intervals of the three proposed detection distributions. The PMFs under variable detections for symbols periods at 0, 1, 2, 3 and 4dB are shown in Figure 3.21a, while other four PMFs under uniform detections with increments of 5% are depicted in Figure 3.21b. The PMF curves shown in Figure 3.21 are obtained through spline interpolation of their discrete probabilities. From these results, it is noted that the average latency decreases as the symbol period increases or when more power is employed.

In order to determine the statistical average latency in a range of SNRs from 0 to 4dB, the expected latency of symbol periods $\{T_1, T_2, ...T_j\}$ inside this range should be calculated. Recall that receiving a signal with lower SNR is equivalent to receive a portion of the symbol energy. The longer symbol period T, at SNR = 4dB, represents an expected block-error rate of $\varepsilon(T) = 1.36 * 10^{-6}$ in the finite-blocklength regime and $\varepsilon(T) = 1.32 * 10^{-3}$ under the CRC-polar code. Moreover, consider that symbol durations are from -3dB (in terms of SNR) to each one of the selected symbol periods and the asynchronous detections of the three distributions start at specific IDTs. Therefore, the statistical average latency for the j-th symbol period T_j is

calculated by rewritten the Equation (1.6) as follows

$$\bar{\tau}_{T_j} = \sum_{\tau_i=\tau_1}^{T_j} \tau_i p_\tau(\tau_i). \tag{3.16}$$

The probability of each obtained average latency results from the sum of the PMFs belonging to their respective early detections, namely $\mathbb{P}[\bar{\tau}_{T_j}] = \sum_{\tau_1}^{T_j} p_\tau(\tau_i)$. This probability indicates the confidence degree of the statistical average latency, and according with this, the obtained average latency is classified as reliable or unrealiable. If $\mathbb{P}[\bar{\tau}_{T_j}] \geq 0.99$, then the obtained average latency is reliable otherwise it is unreliable. The unreliable average latencies arise from PMF curves that do not have a complete Gaussian form. That is, the area (or probability) under these curves is mutilated in the sense that the latency is bounded to 100%, see Figure 3.21. This happens because the chosen symbol period is too short and the symbol energy is not enough to do a correct inference where the error probability is very high (close to one).

Table 3.7 Statistical average latencies under uniform detection distributions at different symbol periods T, $N = 256$, $k = 120$, CRC=8 and $\tau_1 = 20\%$

Period T at E_b/N_0 [dB]	**Finite-blocklength regime**				**CRC-polar codes**			
	$\Delta\tau = 5\%$		$\Delta\tau = 1\%$		$\Delta\tau = 5\%$		$\Delta\tau = 1\%$	
	Avg. Lat. [%]	**Prob.**	**Avg. Lat. [%]**	**Prob.**	**Avg. Lat. [%]**	**Prob.**	**Avg. Lat. [%]**	**Prob.**
4.00	37.68	1.00	35.68	1.00	57.15	1.00	55.27	1.00
3.78	39.53	1.00	37.53	1.00	60.05	1.00	58.01	1.00
3.54	41.58	1.00	39.58	1.00	62.91	0.99	60.93	0.99
3.29	43.88	1.00	41.88	1.00	65.68	0.99	63.87	0.99
3.03	46.47	1.00	44.47	1.00	68.31*	0.97	66.53 *	0.97
2.75	49.40	1.00	47.40	1.00	70.20*	0.95	68.77 *	0.95
2.45	52.75	1.00	50.75	1.00	70.34*	0.90	69.20 *	0.91
2.13	56.61	1.00	54.61	1.00	67.40*	0.83	68.85 *	0.83
1.78	61.11	1.00	59.11	1.00	60.44*	0.71	59.49 *	0.72
1.40	66.30	1.00	64.31	1.00	47.29*	0.54	46.27 *	0.54
0.99	71.81	0.99	69.83	0.99	31.59*	0.35	30.52 *	0.35
0.53	75.16*	0.95	73.27*	0.95	16.56*	0.18	16.27*	0.18
0.02	69.24*	0.81	67.65*	0.81	6.73*	0.07	6.15*	0.07

* Unreliable average latency

The resulting statistical average latencies with their respective probabilities, under uniform detection distributions, are detailed at Table 3.7. Regarding these results, there is a larger set of reliable statistical average latencies in the finite-blocklength regime than using CRC-polar codes. In the finite-blocklength regime the statistical average latencies are reliable from 0.99 to 4dB, while CRC-polar codes provide reliable statistical average latencies only from 3.29 to 4dB. This is because the theoretical block-error rate in the finite-blocklength regime is much better than the block-error rate provided by the simulation of a communication system under CRC-polar codes. For the same reason, there is a large gap between the reliable latencies in finite-blocklength regime and the reliable latencies obtained through CRC-polar codes. At SNR = 4dB, the statistical average latency of CRC-polar codes has a difference of approximately 20% with the statistical average latency in the finite-blocklength regime. Take into account that the error rate (Equation (1.10)) used in the finite-blocklength regime is the optimal error performance of an arbitrary code, and consequently their statistical latencies shown in Table 3.7 are considered optimal latencies.

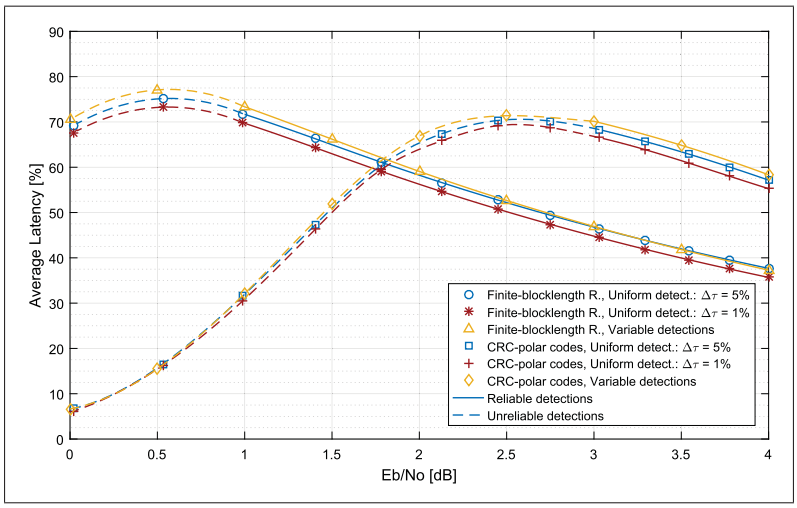


Figure 3.22 Statistical average latencies with different detection distributions, in the finite-blocklength regime and using CRC-polar codes

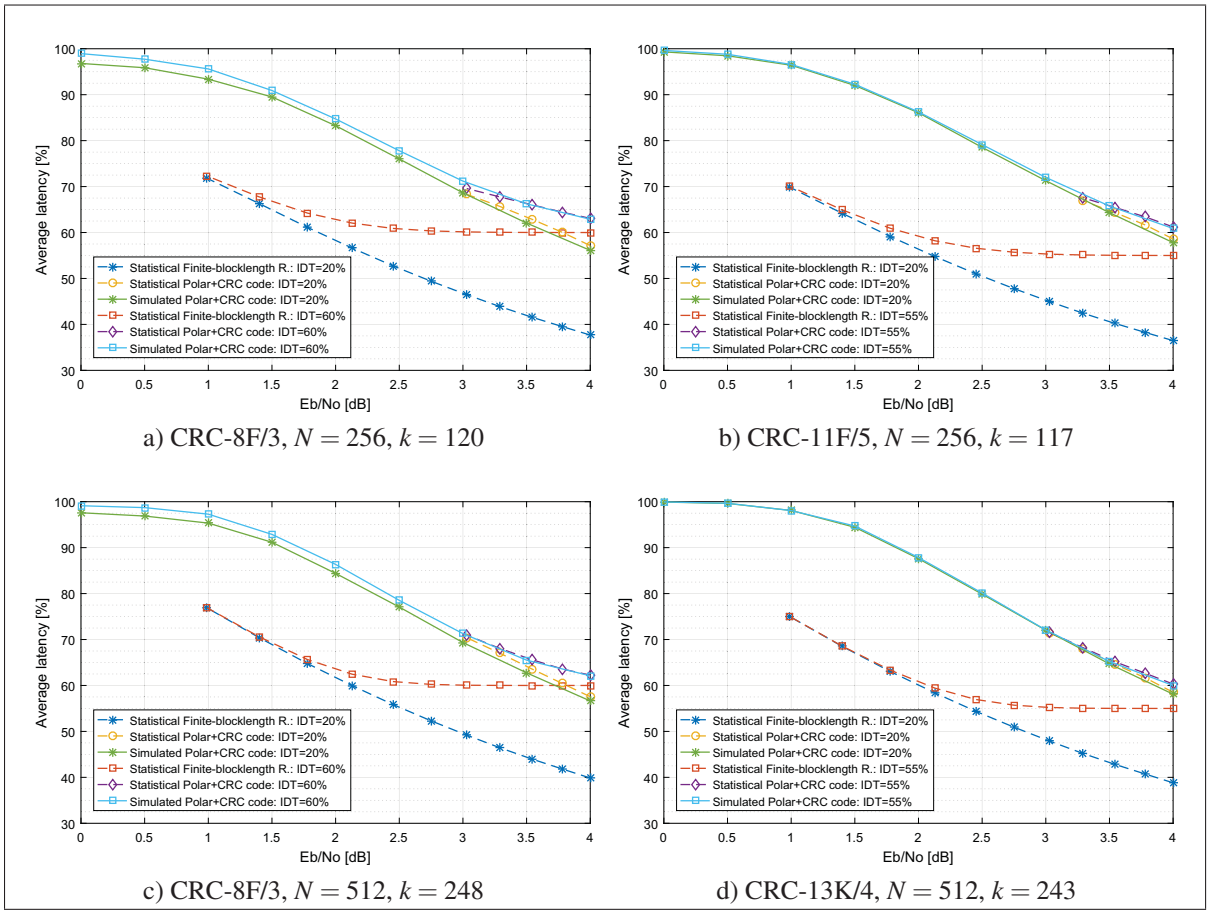
In Figure 3.22, as expected, the statistical average latencies increase as the SNR or symbol period decreases (from right to left). However, this behavior is only up to certain SNR values,

then the statistical average latencies decrease. This is because the error probabilities at lower SNRs are almost equal to one, which is more accentuated using CRC-polar codes. Comparing the latency performances obtained by the distribution with increasing variable detection times and the uniform detection distribution with fixed increments of 5%, they have similar performances in the finite-blocklength regime as SNR increases, see Figure 3.22. On the other side, the uniform detection distribution (5%) is slightly better than the variable detection distribution under CRC-polar codes. There is not a significant difference between the uniform distribution (5%) and the variable distribution because the latter distribution has fewer detections as the symbol energy is higher. Also, it is observed in Figure 3.22 that uniform detection distributions with fixed increments of 1% of T provide a latency improvement in both scenarios. For example, in the finite-blocklength regime, when the SNR = 4dB, the uniform detection distribution with increments of 1% offers a latency improvement of 2% with respect to the latency of the other uniform detection distribution. While using CRC-polar codes, this latency improvement is 1.88%, see Table 3.7. Therefore, the average latency degrades as the time interval between asynchronous detections increases. In real implementations, the interval between sequential detections depends on the processing latency of the decoder and the OFDM demodulator. For this reason, this processing latency must be very low to efficiently implement the sequential early detection scheme.

3.10 Comparison of the statistical and simulated average latency

The purpose of this section is to compare the average latency obtained through the simulations of the CRC-based sequential early scheme, with the statistical average latency proposed by Au & Gagnon (2016). As we saw in previous section, the statistical average latency can be calculated by employing the error probability of an arbitrary code in the finite-blocklength regime or by using the BLER of the CRC-polar code without early detections. For this analysis only the reliable statistical average latencies are considered. The final average latencies shown in Figure 3.19b and 3.19d result from uniform detections (with increments of 5% of T) and different IDTs according with the employed CRC polynomial. The comparison of these results

with the statistical average latency should be under the same conditions. Consider the CRC polynomials that generate balanced results between the error rate and average latency. That is, CRC-11F/5 for $N = 256$ and CRC-13K/4 for $N = 512$ with $R_{pc} = 0.5$ and an IDT of 55%. Also consider the CRC-8F/3 for both block lengths, which obtains the best error and latency performance at low SNRs with an IDT of 60% and $R_{pc} = 0.5$. Similar as in Section 3.9, statistical average latencies are computed under constant increments of 5% of T but with the same initial detection times of the simulated average latencies to do a fair comparison. In addition to the IDT that generates the best performance (*i.e.* trade off between BLER and average latency) of the early detection scheme with each CRC polynomial, we also present results with an IDT equal to 20%.



a) CRC-8F/3, $N = 256$, $k = 120$

b) CRC-11F/5, $N = 256$, $k = 117$

c) CRC-8F/3, $N = 512$, $k = 248$

d) CRC-13K/4, $N = 512$, $k = 243$

Figure 3.23 Statistical and simulated average latencies under uniform detections with $\Delta\tau = 5\%$ and different IDTs (τ_1), for block lengths of 256 and 512, $R_{pc} = 0.5$

As shown in Figure 3.23, both statistical and simulated average latencies degrade as the initial detection time increases. The simulated average latencies of the selected CRC polynomials do not achieve the performance of their respective statistical average latencies in the finite-blocklength regime, specially performances under lower IDTs (20%). However, the simulated average latencies are similar to the statistical average latencies under the BLER of CRC-polar codes without early detections. The latter confirms that the average latencies obtained through simulations are correct. The simulated average latencies obtained through CRC-polar codes need to improve in around 20% to achieve the optimal statistical latency performance of the finite-blocklength regime with IDTs of 20%. While statistical average latencies with IDTs of 55% or 60% get closer to their respective simulated average latencies as the SNR increases. Indeed, these statistical average latencies are truncated by the longer IDTs. For instance, when $N = 512$ and $k = 243$, the simulated average latency of the CRC-13K/4 with an IDT of 55% is 59.99% at 4dB, while the statistical average latency is 55% in the finite-blocklength regime, see Figure 3.23d. On the other hand, the simulated average latencies with a larger block length (512) are slightly closer to the statistical average latencies in the finite-blocklength regime, since the statistical latencies degrade at least 2% as N increases from 256 to 512. This behavior can be observed in Figures 3.23a and 3.23c, where average latencies are obtained under the same CRC polynomial and IDTs. Based on these results, we conclude that the obtained (simulated) average latencies of the CRC-based sequential early detection scheme are still far from the theoretical or optimal average latencies in the finite-blocklength regime. This is the price to maintain the block-error rate of a system with synchronous detections under a polar decoder (SC decoder) with regular error performance. Therefore, it is necessary to use a new polar decoder with better error performance to further decrease the average detection latency of a multicarrier communication system.

3.11 Is the latency reduction possible in practice?

Since this work employs the successive cancellation (SC) decoder of polar codes, which does not offer a low decoding latency, an evident question appears, the sequential early detection

based on a concatenated CRC-polar code can achieve a latency reduction in an OFDM system in practice? We believe that it is possible because there are hardware or software polar decoders that achieve high throughputs, and low latencies less than the OFDM symbol duration. The long-term evolution (LTE) system uses an OFDM symbol duration of $66.6\mu s$, Khan (2009). The fast simplified successive-cancellation (Fast-SSC) algorithm proposed by Sarkis *et al.* (2014) has been implemented over on an Altera Stratix IV field-programmable gate array (FPGA), and has been modified by Giard *et al.* (2015a) to further improve the decoding latency at low-rate codes, see Table 3.8. The Fast-SSC algorithm is also implemented under a fully-unrolled architecture, Giard *et al.* (2015b).

Table 3.8 Implementation results of SC-based polar decoders
Taken from Giard *et al.* (2016) and Sarkis *et al.* (2016)

Polar decoder	**N**	**Code rate**	**Frequency (Mhz)**	**Latency (μs)**	**Throughput (Gbps)**
Original Fast-SSC	1024	0.5	103	2.14	0.475
Modified Fast-SSC	1024	0.5	103	1.60	0.638
Unrolled Fast-SSC	1024	0.5	248	1.47	254.1
Unrolled Fast SC-list	1024	0.5	Intel Corei7	14	0.038

For polar codes with short or moderate lengths, the SC decoding algorithm provides an inferior error-correction performance than other codes. The CRC-aided successive cancellation list (SC-list) decoding algorithm (Tal & Vardy (2015)) solve this problem, offering even a better error performance than LDPC codes. However, this decoder has a low decoding latency. This issue is managed by adapting the fast-SSC decoding algorithm to the list-CRC decoder and unrolling the decoding tree of the code, resulting in a unrolled fast SC-list decoder, (Sarkis *et al.* (2015), Sarkis *et al.* (2016)). The results of the software implementation of this decoder are shown in Table 3.8.

The unrolled fast SC-list decoder might be a good option to implement the sequential early detection given that in this decoder the CRC code is also used as an outer code that helps to detect the transmitted codeword. Thus, the CRC code could be used for two purposes, identify

the transmitted codeword from the list of paths at an early detection time and give the feedback to close the loop of sequential asynchronous detections.

3.12 Conclusions

In this chapter, some selection processes have been developed to establish the best possible setting of a sequential early detection scheme based on CRC-polar codes. The main conclusions of these processes are detailed below.

- The location of data bits and CRC-bits inside the K most reliable bit-channels of the output vector of the polar code construction does not significantly affect the error performance of the early detection scheme. However, it is recommended to give priority to data bits.
- The sequential early detection scheme based on a concatenated CRC-polar code with a fixed polar code rate obtains a better error rate and average detection latency as the concatenated code rate decreases. This means that under a fixed number of CRC bits and a fixed block length, less data bits improve the error and latency performances of the early detection scheme.
- The BLER and average detection latency depend on the structure of the CRC-polar codeword, this means that the early detection scheme performance is affected by the number of data bits (k), the number of CRC bits (r), and the block length of the polar codeword (N).
- The error performance of a concatenated CRC-polar code without early detections get worse as the CRC size increases.
- In the setting of the CRC-based sequential early detection scheme with $R_{pc} = 0.5$, the selection of a CRC polynomial is only necessary if their degrees are less than 12 for $N = 256$ or 15 for $N = 512$. Otherwise, this is not necessary.
- A sequential early detection scheme under CRC polynomials with different coefficients but with the same degree (from 8-bit to 32-bit CRCs) generates the same average detection latency. However, the average detection latency degrades as the CRC size increases.

- The CRC polynomial selection of a specific CRC size is only based on their error performance comparison. The average latency is not considered.
- As the CRC size increases, the sequential early detection scheme results in two block-error rate behaviors, which are divided by a specific CRC size. This CRC size breakpoint obtains the lowest BLER at the highest SNR.
- The error and latency performances of a sequential early detection scheme based on CRC-polar codes, that manages a fixed polar code rate, improve as its block length increases. Consequently, a large number of subcarriers is recommended to use in OFDM systems under sequential early detections.
- In a sequential early detection scheme, the error performance decreases and the average latency increases as the initial detection time gets longer. However, this behavior is less noticeable as the CRC size increases, specially in error performances and in average latencies resulting from IDTs equal to or less than 50% of T. The selection of the initial detection time in an early detection scheme can be omitted for CRC polynomials with a degree higher than 14 for $N = 256$ or 16 for $N = 512$, and assume an IDT equal to 50%.
- To obtain the best possible average latency and BLER at the highest SNR (4dB), the CRC-14/DARC and CRC-16 are recommended to be applied in the sequential early detection scheme with block lengths 256 and 512, respectively. Take into account that these results are achieved if the initial detection time is 50% of T and uniform detections with increments of 5% of T are used.
- If early detection schemes with block lengths 256 and 512 employ respectively the CRC-11F/5 and CRC-13K/4 with an IDT of 55% of T and uniform detections with increments of 5%, relative good BLERs and average latencies in the whole range of SNRs from 0 to 4dB are generated.
- The statistical average latency proposed by Au & Gagnon (2016), is valid for specific ranges of SNR where the error probability is low. Otherwise, the statistical average latency is unreliable.

- The average detection latency of a CRC-based sequential early detection also depends on the time separation of the asynchronous detections. The average latency improves as the time interval of detections decreases.

- Although the reduction of the average detection latency is significant (40% at 4dB) using a CRC-based sequential early detection scheme, this is not enough to achieve the theoretical or statistical average latency in the finite-blocklength regime. Under an ideal scenario with an optimal code, the average latency could decrease by 20% more.

CONCLUSION AND RECOMMENDATIONS

One of the challenges to overcome by the telecommunication industry is to achieve low latencies for real-time applications. In this investigation, the aim was to decrease the average detection latency of a multicarrier communication system while maintaining the block-error rate of a synchronous detection system. For this purpose, we employ a sequential early detection scheme based on CRC-polar codes. This study set out to determine the configuration parameters of the early detection scheme based on CRC-polar codes. The first parameter is known as design-SNR, which is used to construct polar codes. The second parameter is the generator polynomial of the CRC code, and the third one is the initial detection time of the sequential detection. Basically, the selection processes of these parameters are based on performance comparisons of the early detection scheme under possible parameters. The sequential early detection scheme was simulated by using CRC-polar codes, parallel transmission of BPSK symbols over AWGN channels and the parameters under evaluation. After developing extensive simulations and the corresponding parameter selection processes, we have succeeded in decreasing the detection latency of a multicarrier system and approaching to the block-error rate of a system under synchronous detections.

In Chapter 2, we determined the best design-SNRs to construct polar codes over AWGN channels. This work was developed based on the comparison of block-error rates with Vangala's search method. Moreover, it was confirmed that Bhattacharyya and Tal&Vardy construction methods of polar codes can provide similar performances over AWGN channels if the design-SNR is optimized. This statement was first proved by Vangala *et al.* (2015), but under the comparison of BERs. Regarding the results obtained in Chapter 2, the design-SNRs obtained for Bhattacharyya and Tal&Vardy construction methods are optimized for specific SNR ranges and depend on the block length and code rate of the polar code. This means that the results obtained by Vangala *et al.* (2015) would not have been useful for our work.

The Tal&Vardy construction results in a slightly better block-error performance than the Bhattacharyya construction by making a comparison at every SNR with every best design-SNR. Another advantage of Tal&Vardy algorithm is that polar codes generated by this construction method are optimized at the same SNRs for which they were designed, while this does not happen with Bhattacharyya construction. Despite of the advantages of Tal&Vardy construction over Bhattacharrya construction, the Bhattacharyya algorithm can be considered as a good alternative for the construction of polar codes over AWGN channels. Taking into account that this construction has a very low complexity, and generates such a good error performance as Tal&Vardy construction if the selected design-SNR is the best one. This is why the Bhattacharyya construction was selected for the works developed in Chapter 3, with a design-SNR of 4dB to obtain the best possible BLERs around an intermediate SNR of 2.5dB for block lengths of 256 and 512.

In Chapter 3, the selection processes of CRC polynomials and IDTs conclude in suitable parameters for specific SNR ranges (low, medium and high), which are part of the observed range of SNRs from 0 to 4dB. In other words, the obtained CRC polynomials and IDTs generate the best possible average detection latency and error rate of the CRC-based early detection scheme for their corresponding SNR ranges. Advantageously, we identified that some of these CRC polynomials with their respective IDT can provide a relative good error and latency performance for the entire range of SNRs. By using an appropriate configuration of the early detection scheme, the achieved latency reduction is around of 40% of the OFDM symbol period when the required SNR is 4dB, for block lengths of 256 or 512 with a fixed polar code rate of 0.5. Regarding the error performance of the early detection scheme, it is significantly affected by the block length of the polar code, so that a larger block length provides a better error performance as the polar code rate remains fixed. The early detection scheme under a suitable configuration with a block length of 512 and a polar code rate of 0.5 provides block-error rates that approach to the target error performance of the scheme under synchronous detections. This is specifically achieved

by using the CRC-13K/4 with an IDT=55% or the CRC-16 with an IDT=50%, under uniform detection distributions with increments of 5% of T. Moreover, this study has also found that a larger block length improves the average detection latency at high SNRs. Consequently, the block-length of the CRC-polar codeword plays an important role to obtain good results with an early detection scheme. In the future, it will be important to study what are the limitations and what is the price for increasing the block length of the CRC-polar codeword in an early detection scheme over an OFDM system. Another future work is to perform the early detection scheme over an OFDM system with a higher polar-coded modulation level.

Although this study has successfully demonstrated that under a suitable setting of the early detection scheme based on CRC-polar codes, it is possible to decrease the average detection latency of a multicarrier system, it does not achieve the theoretical or statistical average detection latency in the finite-blocklength regime. This is due to the regular error performance of the successive-cancellation decoder used in our project. Future works should include the used of the fast version of the SC-list decoder (Sarkis *et al.* (2016)) for the early detection scheme based on the concatenation of CRC and polar codes.

In general, this research has demonstrated that both the error rate and the average detection latency of an early detection scheme based on the concatenation of CRC codes and polar codes are affected by the chosen CRC polynomial, the initial detection time and the separation between asynchronous detections. In addition, these performances are also influenced by the bit-arrangement of the CRC-polar codeword, by the construction of the polar code (design-SNR) and by the polar decoder performance.

BIBLIOGRAPHY

Arikan, E. (2009). Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7), 3051–3073.

Arikan, E. et al. (2008). A performance comparison of polar codes and reed-muller codes. *IEEE Commun. Lett*, 12(6), 447–449.

Au, M. & Gagnon, F. (2016). On optimal latency of communications. *arXiv preprint arXiv:1610.00017*.

Cho, Y. S., Kim, J., Yang, W. Y. & Kang, C. G. (2010). *MIMO-OFDM wireless communications with MATLAB*. John Wiley & Sons.

Chung, S.-Y., Richardson, T. J. & Urbanke, R. L. (2001). Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation. *Information Theory, IEEE Transactions on*, 47(2), 657–670.

Cook, G. (2016). Catalogue of parametrised CRC algorithms. Consulted at http://reveng.sourceforge.net/crc-catalogue/all.htm.

Forney, G. D. & Forney, G. D. (1966). *Concatenated codes*. Citeseer.

Gagnon, F. (2016). On wireless IP technology for developing countries. Consulted at http://en.etsmtl.ca/Chaires-de-recherche/Chaire-numerique-sans-fil/Enjeux.

Giard, P., Sarkis, G., Thibeault, C. & Gross, W. J. (2015a). A 638 Mbps low-complexity rate 1/2 polar decoder on FPGAs. *Signal Processing Systems (SiPS), 2015 IEEE Workshop on*, pp. 1–6.

Giard, P., Sarkis, G., Thibeault, C. & Gross, W. J. (2015b). Unrolled polar decoders, part i: hardware architectures. *arXiv preprint arXiv:1505.01459*.

Giard, P., Sarkis, G., Balatsoukas-Stimming, A., Fan, Y., Tsui, C.-y., Burg, A., Thibeault, C. & Gross, W. J. (2016). Hardware decoders for polar codes: An overview. *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, pp. 149–152.

Goldsmith, A. (2005). *Wireless communications*. Cambridge university press.

Hassani, S. H. & Urbanke, R. (2014). Universal polar codes. *2014 IEEE International Symposium on Information Theory*, pp. 1451–1455.

ITU-R. (2015). IMT Vision - Framework and overall objectives of the future development of IMT for 2020 and beyond. International Telecommunication Union. Recommendation ITU-R M.2083-0.

Kawanishi, T., Kanno, A. & Yamamoto, N. (2012). Low latency data transmission using wireless and wired communications. *International Conference on Space Optical Systems and Applications (ICSOS) 2012*.

Khan, F. (2009). *LTE for 4G mobile broadband: air interface technologies and performance*. Cambridge University Press.

Koopman, P. (2002). 32-bit cyclic redundancy codes for internet applications. *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, pp. 459–468.

Koopman, P. (2016). Best CRC polynomials. Consulted at https://users.ece.cmu.edu/~koopman/crc/index.html#notation.

Koopman, P. & Chakravarty, T. (2004). Cyclic redundancy code (CRC) polynomial selection for embedded networks. *Dependable Systems and Networks, 2004 International Conference on*, pp. 145–154.

Kurose, J. F. & Ross, K. W. (2007). *Computer networking: a top-down approach*. Addison Wesley.

Leroux, C., Tal, I., Vardy, A. & Gross, W. J. (2011). Hardware architectures for successive cancellation decoding of polar codes. *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pp. 1665–1668.

Li, H. & Yuan, J. (2013). A practical construction method for polar codes in AWGN channels. *TENCON Spring Conference, 2013 IEEE*, pp. 223–226.

Lin, S. & Costello, D. J. (2004). *Error control coding*. Pearson Education India.

Martin, R. (2007). Wall streets quest to process data at the speed of light. Consulted at http://www.informationweek.com/wall-streets-quest-to-process-data-at-th/199200297.

Mori, R. & Tanaka, T. (2009a). Performance and construction of polar codes on symmetric binary-input memoryless channels. *2009 IEEE International Symposium on Information Theory*, pp. 1496–1500.

Mori, R. & Tanaka, T. (2009b). Performance of polar codes with the construction using density evolution. *IEEE Communications Letters*, 13(7), 519–521.

Narayanan, K. R. & Stuber, G. L. (1998). List decoding of turbo codes. *IEEE Transactions on Communications*, 46(6), 754–762.

Nikaein, N., Knopp, R., Cipriano, A. M., Krco, S., Tomic, I., SVOBODA, P., LANER, M., LARSSON, E., Wu, Y., Fuertes, M. G. et al. (2011). Low-Latency in Wireless Communication. *VITEL Petindvajseta delavnica o telekomunikacijah*, 93–99.

Osseiran, A., Boccardi, F., Braun, V., Kusume, K., Marsch, P., Maternia, M., Queseth, O., Schellmann, M., Schotten, H., Taoka, H. et al. (2014). Scenarios for 5G mobile and wireless communications: the vision of the METIS project. *IEEE Communications Magazine*, 52(5), 26–35.

Peterson, W. W. & Brown, D. T. (1961). Cyclic codes for error detection. *Proceedings of the IRE*, 49(1), 228–235.

Polyanskiy, Y., Poor, H. V. & Verdú, S. (2010). Channel coding rate in the finite blocklength regime. *IEEE Transactions on Information Theory*, 56(5), 2307–2359.

Proakis, J. G. & Masoud, S. (2008). *Digital communications*. McGraw-Hill, New York.

Rumble, S. M., Ongaro, D., Stutsman, R., Rosenblum, M. & Ousterhout, J. K. (2011). It's Time for Low Latency. *HotOS*, 13, 11–11.

Sarkis, G. (2016). *Efficient encoders and decoders for polar codes: algorithms and implementations*. (Ph. D. thesis, PhD thesis, McGill University).

Sarkis, G., Giard, P., Vardy, A., Thibeault, C. & Gross, W. J. (2014). Fast polar decoders: Algorithm and implementation. *IEEE Journal on Selected Areas in Communications*, 32(5), 946–957.

Sarkis, G., Giard, P., Vardy, A., Thibeault, C. & Gross, W. J. (2015). Unrolled polar decoders, part ii: Fast list decoders. *arXiv preprint arXiv:1505.01466*.

Sarkis, G., Giard, P., Vardy, A., Thibeault, C. & Gross, W. J. (2016). Fast List Decoders for Polar Codes. *IEEE Journal on Selected Areas in Communications*, 34(2), 318–328.

Şaşoğlu, E. & Wang, L. (2016). Universal polarization. *IEEE Transactions on Information Theory*, 62(6), 2937–2946.

Sklar, B. (2001). *Digital communications*. Prentice Hall Upper Saddle River.

Tal, I. & Vardy, A. (2015). List decoding of polar codes. *IEEE Transactions on Information Theory*, 61(5), 2213–2226.

Tal, I. & Vardy, A. (2013). How to construct polar codes. *Information Theory, IEEE Transactions on*, 59(10), 6562–6582.

Trifonov, P. (2012). Efficient design and decoding of polar codes. *Communications, IEEE Transactions on*, 60(11), 3221–3227.

Vangala, H., Viterbo, E. & Hong, Y. (2014). Permuted successive cancellation decoder for polar codes. *Information Theory and its Applications (ISITA), 2014 International Symposium on*, pp. 438–442.

Vangala, H., Viterbo, E. & Hong, Y. (2015). A comparative study of polar code constructions for the AWGN channel. *arXiv preprint arXiv:1501.02473*.

Wang, C. D. & Thompson, J. P. (1997). Apparatus and method for motion detection and tracking of objects in a region for collision avoidance utilizing a real-time adaptive probabilistic neural network. Google Patents. US Patent 5,613,039.

Wang, R., Zhao, W. & Giannakis, G. B. (2008). CRC-assisted error correction in a convolutionally coded system. *IEEE Transactions on Communications*, 56(11).

Wu, D., Li, Y. & Sun, Y. (2014). Construction and block error rate analysis of polar codes over AWGN channel based on Gaussian approximation. *Communications Letters, IEEE*, 18(7), 1099–1102.