POLAR REED-SOLOMON CONCATENATED CODES FOR OPTICAL COMMUNICATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

By
Yiğit Ertuğrul
July 2020

Polar Reed-Solomon Concatenated Codes for Optical Communications
By Yiğit Ertuğrul
July 2020

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Erdal Arıkan (Advisor)

Tolga Mete Duman

Ali Özgür Yılmaz

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

POLAR REED-SOLOMON CONCATENATED CODES FOR OPTICAL COMMUNICATIONS

Yiğit Ertuğrul
M.S. in Electrical and Electronics Engineering
Advisor: Erdal Arıkan
July 2020

A concatenated forward error correcting (FEC) code is developed by targeting optical communications. Polar and product Reed-Solomon (RS) codes are used as inner and outer codes, respectively. An interleaver block is designed to align multiple inner code blocks. Target key parameter indicators (KPIs) for the decoder circuitry are set to 1 Tb/s throughput and 10 mm^2 area occupation. These KPIs narrowed the design space down to the simplest decoding algorithms in moderate block-lengths. Soft information from the channel is collected by a polar decoder. Minimum distance between codewords is increased by a product code with two error correcting RS codes. Performance of the developed FEC code is evaluated based on its communications performance, decoding complexity and area occupation. Code configurations are designed with overheads of 15%, 20%, 24% and 28% supporting 1 Tb/s throughput. In one configuration, 11.3 dB net coding gain is estimated at 10^{-15} bit error rate (BER). Area of the decoder circuitry is estimated to be 14.27 mm^2 in 28nm while supporting 1 Tb/s throughput.

Keywords: error-correcting codes, polar codes, Reed Solomon codes, optical communications.

ÖZET

OPTİK HABERLEŞME İÇİN UÇ UCA EKLEMELİ KUTUPSAL REED-SOLOMON KODLAR

Yiğit Ertuğrul
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Erdal Arıkan
Temmuz 2020

Optik hatlarda haberleşmeyi hedefleyen uç uca eklenmiş bir hata düzeltme kodu geliştirildi. Kutupsal ve Reed-Solomon (RS) çarpım kodlar iç ve dış kodlar olarak kullanıldı. Birden fazla iç kodu dizmek için bir harmanlayıcı bloğu tasarlandı. Kod çözücü devresi için, saniye başına 1 terabit veri hızı ve 10 milimetrekare alan performans belirleyici hedefler olarak belirlendi. Bu performans belirleyici hedefler, tasarım uzayını orta blok uzunluklarında en basit kod çözme algoritmalarına kadar daralttı. Kanaldaki yumuşak bilgi bir kutupsal kod çözücü tarafından toplandı. Kod kelimeleri arasındaki en az mesafe iki hata düzelten RS kod kullanan çarpım kodları ile artırıldı. Geliştirilen hata düzeltme kodunun performansı; iletişim performansı, kod çözme karmaşıklığı ve kapladığı alan baz alınarak değerlendirildi. Saniye başına 1 terabit veri hızında %15, %20, %24 ve %28 fazlalık oranlarında kod tasarımları yapıldı. Bir tasarımda, 10^{-15} bit hata oranında 11.3 dB net kodlama kazancı kestirimi yapıldı. Saniye başına 1 terabit veri hızını sağlayan kod çözücü devresinin alanı 28nm teknolojisinde 14.27 milimetrekare olarak kestirildi.

Anahtar sözcükler: hata düzelten kodlar, kutupsal kodlar, Reed-Solomon kodlar, optik haberleşme.

Acknowledgement

I would like to thank my supervisor, Prof. Erdal Arıkan for his persistent support, encouragement and patience during my thesis. I would like to thank my thesis jury members, Prof. Tolga Mete Duman and Prof. Ali Özgür Yılmaz for their comments that elevated the quality of this thesis.

I would like to thank Bilkent University for giving me the opportunity to study here. I would like to thank EPIC project which is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 760150.

In particular, I would like to thank Ertuğrul Kolağasıoğlu for his support and teachings that motivated this thesis.

I am indebted to my colleagues in Polaran. I would like to thank Altuğ Süral and Evren Göksu Sezer for the invaluable technical discussions. I would like to thank my family for their support throughout my educational career.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In his seminal paper [1], Shannon presented an upper limit on the information rate, called channel capacity, below that limit reliable communication is possible with vanishing error probability. Figure 1.1 depicts a generic communication system where data source generates K information bits $\mathbf{u}^K$ and channel encoder adds $N-K$ bits of redundancy to generate N coded bits $\mathbf{x}^N$. The ratio of $\frac{K}{N}$ is called the code rate. Coded bits pass through a channel and channel decoder receives N channel outputs $\mathbf{y}^N$. Channel decoder generates an estimate $\hat{\mathbf{u}}^K$ of $\mathbf{u}^K$ from the channel output. Proper design of channel encoder and decoder blocks is essential for an efficient use of channel resources. Forward error correction (FEC) deals with design methodology of channel encoder and decoder blocks.

Figure 1.1: Block diagram of a generic communication system.

Performance of a FEC code is evaluated based on its error correction performance against noise and complexity. Bit error rate (BER) and frame error rate (FER) are two common metrics to assess the error correction performance of the FEC

code. BER is the ratio of the number of bit errors to the number of information bits. FER is the ratio of corrupted frames to the transmitted frames where a frame is corrupted if at least one-bit error occurs. Complexity of a FEC code could be evaluated by asymptotic runtime complexity and hardware complexity in terms of area occupation. Landau notation is used to denote the asymptotic runtime complexity of a FEC code. Area occupation is the measure of required space for the decoding circuitry in mm^2 units.

In this thesis, we focus on additive white Gaussian noise (AWGN) channel and binary phase shift keying (BPSK) modulation. Define the AWGN channel as $y = x + z$ where y is the channel output, x is the channel input and z is the additive Gaussian noise term with zero mean and $N_o/2$ variance. Signal-to-noise ratio (SNR) is denoted by $E_s/N_o = \frac{1}{\sigma^2}$. In BPSK modulation, bit energy E_b (joule/bit) and signal energy E_s (joule/2D) is related as $2RE_b = E_s$ where R (bit/D) is the code rate.

Capacity formulas of well-known channels are listed below. Capacity of AWGN channel C_{AWGN}[1] is calculated using Equation 1.1. Consider binary signaling over AWGN channel with $0 \rightarrow 1$ and $1 \rightarrow -1$ symbol mapping. Resulting channel is binary input additive white Gaussian noise (BIAWGN) channel and capacity C_{BIAWGN} is calculated using Equation 1.2. Capacity of binary symmetric channel (BSC) C_{BSC} is calculated using Equation 1.3. Note that $\epsilon = Q(\sqrt{E_s/N_o})$ where $Q(.)$ is the tail distribution function of the standard normal random variable.

$$C_{AWGN} = \frac{1}{2}\log_2(1 + E_s/N_o) \tag{1.1}$$

$$C_{BIAWGN} = \frac{1}{\sqrt{2\pi}}\int e^{-z^2/2}(1 - \log_2(1 + e^{-2E_s/N_o + 2z\sqrt{E_s/N_o}}))dz \tag{1.2}$$

$$C_{BSC} = 1 + \epsilon\log_2(\epsilon) + (1-\epsilon)\log_2(1-\epsilon) \tag{1.3}$$

Capacity of well-known channels are plotted as a function of E_s/N_o (dB) in Figure 1.2. One can observe the capacity loss between BIAWGN channel and BSC due to cardinality of the output alphabet.

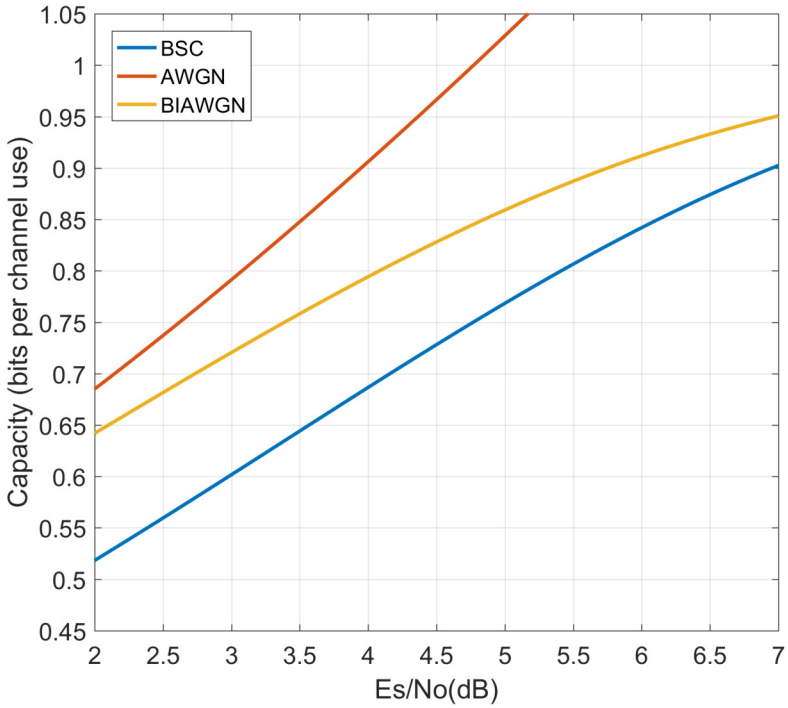

Figure 1.2: Capacity versus E_s/N_o (dB).

Dispersion approximation is used to estimate the achievable FER performance of finite block length codes [2]. Dispersion approximation for BIAWGN channel could be calculated using Equation 1.4 where n is the block length, R is the code rate, C is the channel capacity and V is the channel dispersion term [3].

$$\epsilon^*(R,n) = Q\left(\frac{n(C-R)+0.5\log_2(n)+O(1)}{\sqrt{nV}}\right) \tag{1.4}$$

$$V = \frac{1}{\sqrt{2\pi}}\int e^{-z^2/2}(1-\log_2(1+e^{-2E_s/N_o+2z\sqrt{E_s/N_o}})-C)^2dz \tag{1.5}$$

Achievable performances of finite block length codes are plotted in Figure 1.3 as a function of energy per bit to noise power spectral density ratio (E_b/N_o) where coding overhead is 20%. One can observe that there is a diminishing return of the code block length to the code performance.

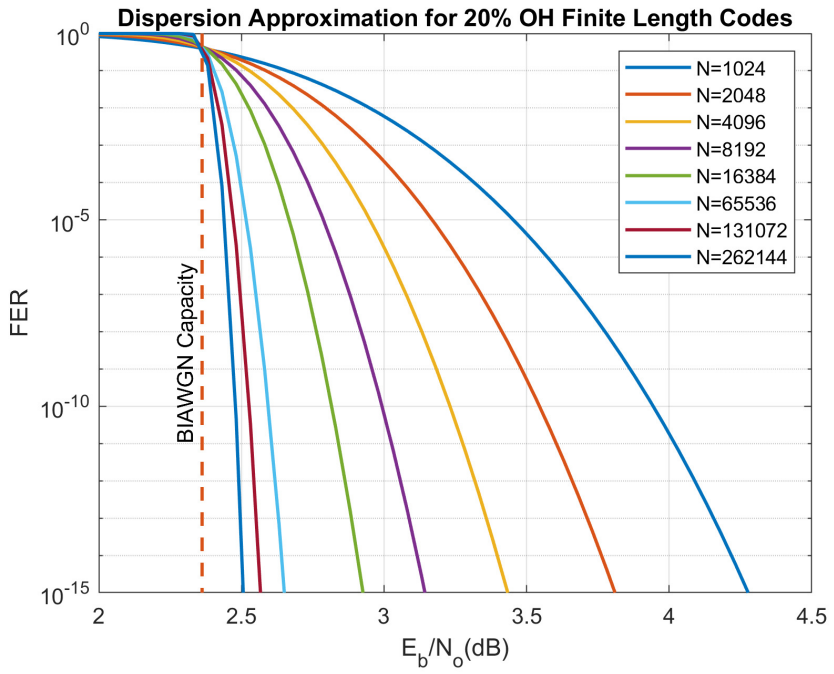

Figure 1.3: Achievable FER performance of finite block-length codes.

1.1 Fiber-Optic Use Case

Network operators are seeking ways of increasing the network capacity to address the traffic growth problem [4]. Communication systems with higher data rates have become favorable in this setting. A desirable solution would operate with the existing physical structures from cabling to bandwidth allocation. To that end, FEC is a viable component that compensates the non-ideal channel effects while progressing to higher data rates.

Optical transport network (OTN) is an interface between digital domain and optical media network [5]. This interface ensures multiplexing, routing, supervision and performance survivability for digital clients. Digital structure of OTN contains multiple optical transport units (OTUs). Ethernet protocol is developed by the Institution of Electrical and Electronics Engineers (IEEE) [6] which can operate on both coaxial cable and optical medium. Data rates from 50 Gb/s to 400 Gb/s are supported over a single-mode fiber in [7]. A comparison of OTN and Ethernet from network level could be found in [8]. FEC is a significant component of OTN since transmission ranges are over 10 km where channel impairments occur. Moreover, retransmission may not be favorable due to large propagation delays. On the other hand, Ethernet does not use FEC to protect the data in the short-range transmissions.

Roadmap of optical communications [4] identifies that the energy efficiency will be one of the biggest challenges of FEC algorithms while supporting higher data rates. In [9], authors discuss the implementation challenges for energy efficient FEC. Impact of FEC on the energy consumption of optical transmitters is analyzed in [10] for short-range optical links. Authors show that FEC can reduce the energy consumption of the transmitter by reducing the transmit power. There is no doubt that energy efficiency will be a distinctive metric among candidate FEC codes.

International Telecommunication Union (ITU) defined a standard for optical fiber submarine cable systems targeting 10^{-15} post-FEC BER [11]. We will

benefit from certain metrics in comparison of candidate FEC codes for optical communications. Coding overhead (OH) denotes the amount of redundancy per information bit and is a function of code rate OH $= (\frac{1}{R} - 1) \times 100$ in percent. Pre-FEC uncoded BER (BER_{pre}) denotes the raw BER at the decoder output and is calculated as in Equation 1.6. Post-FEC BER (BER_{post}) denotes the BER at the decoder output, the target is 10^{-15}. Gap to Shannon limit could be calculated as in Equation 1.7 where $Q^{-1}(.)$ is the inverse of tail distribution function of standard normal random variable. Net Coding Gain (NCG) is the measure of reduction in the transmit power compared to uncoded transmission and could be calculated as in Equation 1.8.

$$\text{BER}_{\text{pre}} = Q\left(\sqrt{2RE_b/N_o}\right) \tag{1.6}$$

$$\text{Gap}_{\text{BIAWGN}} = C^{-1}_{BIAWGN}(R) - \left(Q^{-1}\left(\text{BER}_{\text{post}}\right)\right)^2/(2R) \tag{1.7}$$

$$\text{NCG} = \left(Q^{-1}\left(\text{BER}_{\text{post}}\right)\right)^2/(2R) - E_b/N_o \tag{1.8}$$

A FEC code could have different hardware realizations exploiting different architectural templates. In [12], authors identified the practical limits of the well-known channel codes from an implementation point of view. 10 mm^2 area occupation under 1 Tb/s throughput is identified by Enabling Practical Wireless Tb/s Communications with Next Generation Channel Coding (EPIC) [13] project. These targets are identified only for the decoding circuitry in advanced technology nodes (7/16/28nm). Keeping the data rate, performance and area metrics in mind, a desirable FEC code should exhibit the best of three metrics compared to others.

1.2 Literature Survey

A comprehensive survey of the FEC codes that are suitable for fiber-optic use case are presented in this section. In this field of study, FEC codes are divided into three generations namely first, second and third. First and second-generation

codes are explained in Section 1.2.1. Third generation codes propose more sophisticated FEC code schemes. These codes are presented in Section 1.2.2. Communications performance of the codes will be evaluated based on coding overhead, pre-FEC uncoded BER, post-FEC coded BER and gap to Shannon limit at BER 10^{-15}.

1.2.1 First and Second Generation Codes

First and second-generation codes are presented in this subsection. In [11], ITU defines a Reed-Solomon (RS) code for multigigabit-per-second optical fiber submarine cable systems. Systematic RS(255,239) code with 7% overhead is used to protect the data to be transmitted. This code is capable of correcting $2t + e \leq 16$ errors and erasures. Depending on the interleaver depth n, a FEC frame is $2040 \times n$ bits long.

In [14], ITU defines new FEC codes that are more advanced compared to first generation code. Performance of these FEC codes are summarized in Table 1.1 where one can observe that second generation codes have different overheads varying from 7% to 25%. Low-density parity-check (LDPC) [15] codes, Bose-Caudhuri-Hocquenghem (BCH) [16] codes and convolutional self-orthogonal codes (CSOC) are included in the standard. A review could be found in [17].

Code	Overhead	Block Length (bits)	BIAWGN Shannon Limit at R	Pre-FEC BER Uncoded Eb/No	Post-FEC BER Coded Eb/No	Gap to Shannon Limit dB
RS(255,239)	7%	2040	3.823	8.404e-5 8.499 dB	1e-15 8.781 dB	4.97
CSOC×RS	24.48%	32640	2.07	5.226e-3 5.156 dB	1e-15 6.107 dB	4.577
BCH×BCH	6.69%	32640	3.883	3.217e-3 5.696 dB	1e-15 5.997 dB	2.09
BCH×RS	7%	130560	3.823	2.332e-3 6.023 dB	1e-15 6.317 dB	2.494
pr.Hamming×RS	6.69%	261120	3.883	4.594e-3 5.305 dB	1e-15 5.587 dB	1.704
LDPC	6.69%	32640	3.883	1.533e-3 6.418 dB	1e-15 6.7 dB	2.81
BCH×BCH	7%	32640	3.823	1.245e-3 6.603 dB	1e-15 6.897 dB	3.074
BCH×BCH	11%	33536	3.212	4.443e-3 5.344 dB	1e-15 5.797 dB	2.585
BCH×BCH	25%	38016	2.04	1.286e-2 3.958 dB	1e-15 4.927 dB	2.887
RS	7%	32640	3.823	1.112e-3 6.699 dB	1e-15 6.98 dB	3.16

Table 1.1: Comparison with first and second generation coding schemes.

1.2.2 Third Generation Codes

Third generation codes are explained in this subsection. Figure 1.4 summarizes the code performances under study in terms of coding gain and overhead. Coding gain for soft and hard decision decoding is limited by BIAWGN and BSC capacities, respectively. Markers with blue and red color indicate codes with hard decision and soft decision decoding, respectively. A similar study could be found in [18] including codes in the previous generation. A guide on FEC design options for fiber-optic communications is presented in [19]. In [20] and [21], authors analyzed the performance iterative Bounded Distance Decoder (iBDD). In [22] polar coding for fiber-optical communications is discussed. A survey on FEC codes in optical links could be found in [23]. Performance of the third-generation codes is listed in Table 1.2.

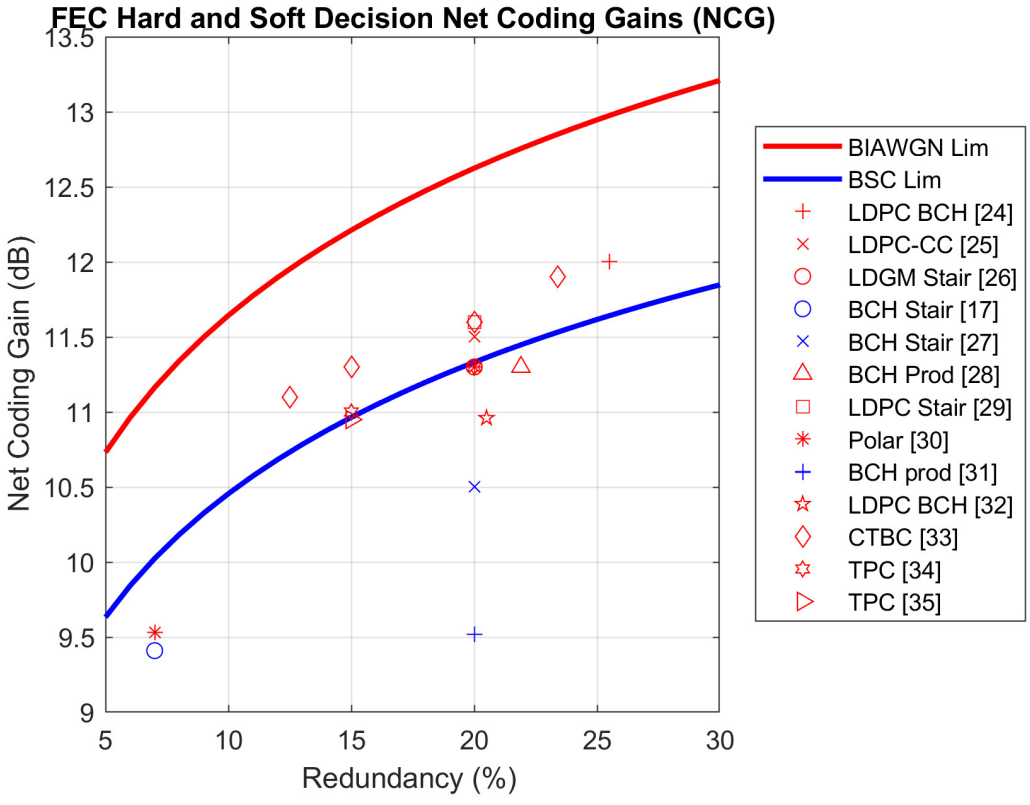

Figure 1.4: FEC hard and soft decision net coding gains.

Code	Overhead	Block Length (bits)	BIAWGN Shannon Limit at R	Pre-FEC BER Uncoded Eb/No	Post-FEC BER Coded Eb/No	Gap to Shannon Limit dB
[24]	25.5%	38400	2.011	3.75e-2 2.001 dB	1e-15 2.987 dB	0.976
[25]	20%	N/A	2.362	2.688e-2 2.695 dB	1e-15 3.487 dB	1.125
[26]	20%	N/A	2.362	2.421e-2 2.895 dB	1e-15 3.687 dB	1.325
[17]	7%	130560	3.823	4.686e-3 5.283 dB	1e-15 5.577 dB	1.754
[27]	20%	N/A	2.362	1.523e-3 3.695 dB	1e-15 4.487 dB	2.125
[28]	21.9%	65025	2.231	1.493e-2 3.727 dB	1e-15 4.587 dB	2.356
[29]	20%	N/A	2.362	2.828e-2 2.595 dB	1e-15 3.387 dB	1.025
[30]	7%	522240	3.823	5.195e-3 5.163 dB	1e-15 5.457 dB	1.634
[30]	20%	261120	2.362	2.421e-2 2.895 dB	1e-15 3.687 dB	1.325
[31]	20%	38025	2.362	7.706e-3 4.675 dB	1e-15 5.467 dB	3.105
[32]	20.5%	2359296	2.327	2.074e-2 3.177 dB	1e-15 3.987 dB	1.66
[33]	12.5%	32640	3.033	1.849e-2 3.375 dB	1e-15 3.887 dB	0.854
[33]	15%	32640	2.775	2.189e-2 3.08 dB	1e-15 3.687 dB	0.912
[33]	20%	32640	2.362	2.828e-2 2.595 dB	1e-15 3.387 dB	1.025
[33]	23.4%	32640	2.136	3.466e-2 2.174 dB	1e-15 3.087 dB	0.951
[34]	15%	65536	2.775	1.845e-2 3.38 dB	1e-15 3.987 dB	1.212
[34]	20%	65536	2.362	2.421e-2 2.895 dB	1e-15 3.687 dB	1.325
[35]	15%	65536	2.775	1.791e-2 3.43 dB	1e-15 4.037 dB	1.262

Table 1.2: Comparison with third generation coding schemes.

Spatially coupled LDPC codes are concatenated with BCH codes in [24] where code block length equals 38400 bits. LDPC(38400,30832) (24.5% overhead) code is concatenated with BCH(30832,30592) (0.78% overhead) code. BCH(30832,30592) code can correct 16-bit errors. Number of iterations is set to 32 for LDPC decoder. Instead of min-sum algorithm, δ-min algorithm is used. Simulation results are reported up to 10^{-11} BER.

LDPC convolutional code (LDPC-CC) is proposed in [25]. LDPC-CC(10032,4,24) code has constraint length L_c equals 10032. Number of ones at each column and row in the parity-check matrix is denoted by J and K, respectively. Here $J = 4$ and $K = 24$. Layered decoding algorithm is used with 12 iterations.

Low-density generator matrix (LDGM) codes are concatenated with staircase codes in [26]. Belief propagation algorithm with min-sum approximation is used at the decoder. Complexity of the decoder is estimated by a score function. Code performance is estimated by analytical methods.

BCH codes are concatenated with staircase codes in [17] where code block length equals 130560 bits. BCH staircase codes are used with a syndrome-based decoder. Component codes are BCH(1023,993,3). Performance of the decoder is verified by field programmable gate array (FPGA) simulations.

BCH codes are concatenated with staircase codes in [27] where component code is BCH(432,396,4). Number of iterations is set to 6. Coding gain is calculated analytically. Decoder throughput achieves 1 Tb/s with around 2 pJ/b energy efficiency at 28nm.

BCH codes are concatenated with product codes in [28] where code block length equals 65025 bits. Product code is used with BCH(255,231,3) component code. Iterative bounded-distance decoding with scaled reliability (iBDD-SR) decoder is used with 5 iterations. BER performance is estimated using extrapolation methods. Decoder throughput achieves 1 Tb/s with 0.63 pJ/b energy efficiency at 28nm.

LDPC codes are concatenated with staircase codes in [29]. Length 30000 and 6000 LDPC codes are used. Sum-product algorithm with floating-point message-passing is used in simulations. Complexity of the decoder is estimated by a score function. Code performance is estimated by analytical methods.

Polar codes are used in [30]. For 20% overhead, block length equals 522240 bits. Polar(522240,435200) code is decoded under successive cancellation (SC) decoding algorithm. For 7% overhead, block length equals 261120 bits. Polar(261120,244736) code is decoded under SC decoding algorithm. Communications performance is estimated with density evolution based Gaussian approximation (DE-GA).

BCH codes are concatenated with product codes in [31] where code block length equals 38025 bits. Product BCH code is used with BCH(195,178) component code. Extended BCH (eBCH) decoder is used for decoding. Decoder achieves 100 Gb/s throughput at 65nm. Decoder throughput is 162 b/cycle at worst case with post-processing iterations. Code performance is evaluated with FPGA simulations up to 10^{-13} BER.

LDPC codes are concatenated with product BCH codes in [32] where code block length equals 2359296 bits. Unequal error protection (UEP) BCH product code is used as outer code and LDPC code is used as inner code. Component codes are BCH(1632,1588) in one axis and BCH(1280,1236), BCH(1280,1255) in the other axis. Inner code is LDPC(4608,4080) with 16 decoding iterations. BCH product code is used with 8 decoding iterations. Extrapolation method is used to estimate the code performance. Simulations are done up to 10^{-8} BER.

Constrained turbo block convolutional codes (CTBC) are used in [33] where block length is 122368 message bits plus coding overhead bits. There are four code designs with 12.5%, 15%, 20% and 23.4% OH. A BCH outer code, a constrained interleaver and a recursive convolutional code is used as inner code. Bahl Cocke Jelinek Raviv (BCJR) and Pyndiah algorithms are used to decode inner and outer codes, respectively. Code performance is estimated analytically.

Turbo product codes (TPC) are used in [34] where block length is configurable with OTU4 frames. There are two code designs with 15% and 20% overheads. Performance of the codes is verified in software and hardware. In [35], a similar turbo product code is developed where code block-length equals 65536 bits and component code is an extended BCH(256,239) code.

1.3 Aim of the Thesis

Aim of this thesis is to develop a FEC code that could satisfy EPIC project targets in advanced technology nodes. EPIC project aims to show Tb/s communications is possible with well-known FEC codes. These targets are 10 mm^2 area occupation and 1 Tb/s throughput. 1024 block-length polar codes under SC decoding satisfied EPIC project targets. However, EPIC polar code has mediocre communications performance compared to codes in the literature. By using EPIC polar code, our aim is to design a concatenated code that satisfies 1 Tb/s throughput and has above 10.5 dB net coding gain at 10^{-15} BER. With the help of RS codes, communications performance of the EPIC polar code is increased. In order to concatenate these codes, RS codes must keep up the pace with EPIC polar codes in terms of throughput. To that end, two-error correcting RS codes are used to satisfy the throughput requirement. In this thesis, we will study the best way to concatenate two FEC codes such that 1 Tb/s throughput is satisfied.

1.4 Summary of Main Results

A concatenated FEC code and architecture is developed that achieve 1 Tb/s throughput, 11.3 dB coding gain under 10 mm^2 in 16nm technology. Polar and product-RS codes are used as inner and outer codes, respectively. Due to implementation constraints, a combination of low-complexity polar codes and simple RS codes is used as component codes for iterative decoding. An interleaver between inner and outer code is employed such that polar frames are distributed into product code in a specific way.

In one configuration, developed FEC code can achieve 11.5 dB net coding gain at 10^{-15} BER with 28% overhead. Table 1.3 summarizes the code parameters and communications performance of the developed codes. One of the developed codes and corresponding architectures are synthesized using Cadence Genus with Taiwan Semiconductor Manufacturing Company (TSMC) 28nm high performance

computing (HPC) library. Area of the developed algorithm is estimated as 14.27 mm^2 in 28nm technology. Area scaling methods [36] are used to project the area of the designed circuitry to 16nm. All the design targets are satisfied in 16nm technology where the area of the decoder is estimated as 4.65 mm^2.

Block Length (bits)	Rate R	Overhead	Gap to Shannon Limit dB	Inner Code	Outer Code
425,984	0.78	28%	1.622	Polar(512,416) R_i=0.8125	RS8(208,204) ×RS8(208,204) R_o=0.9619
425,984	0.78	28%	1.811	Polar(2048,1664) R_i=0.8125	RS8(208,204) ×RS8(208,204) R_o=0.9619
129,024	0.80	24%	2.088	Polar(1024,882) R_i=0.8613	RS7(126,122) ×RS7(126,122) R_o=0.9375
436,224	0.80	24%	1.638	Polar(2048,1704) R_i=0.832	RS8(213,209) ×RS8(213,209) R_o=0.9627
454,656	0.83	20%	1.825	Polar(2048,1776) R_i=0.867	RS8(222,218) ×RS8(222,218) R_o=0.964
468,992	0.87	15%	1.512	Polar(2048,1832) R_i=0.8945	RS8(229,225) ×RS8(229,225) R_o=0.9825

Table 1.3: Communications performance and parameters of developed FEC codes.

1.5 Outline of Thesis

In Chapter 2, a review of FEC codes are presented including polar, RS, product and concatenated codes. SC and successive cancellation list decoding algorithms are explained for polar codes. Analytical error performance and emulation methods are explained for RS codes. Chapter 3 motivates the developed FEC code and possible code configurations. In Chapter 3, design methodology is described to satisfy the targets of the thesis. Simulation results are presented in Section 3.5. Complexity analysis and comparison with other codes are presented in Sections 3.6 and 3.7, respectively. Hardware complexity is analyzed in Section 3.6.1. Finally, in Chapter 4, main results and achievements are summarized, and possible research paths are expressed.

Chapter 2

Review of Codes

A review of FEC codes are presented in this chapter. Codes and techniques that have been described in this chapter serve as a basis for the code design stage in Chapter 3. Polar codes are described in Section 2.1, including code construction techniques and decoding algorithms. Reed-Solomon codes are briefly described in Section 2.2, including accurate performance estimations. Product codes are described in Section 2.3. Concatenated codes are described in Section 2.4.

2.1 Polar Codes

Polar codes are a class of linear block codes that provably achieve symmetric binary input memoryless channel capacity [37]. Low complexity encoding and decoding methods make polar code a favorable FEC code. Third Generation Partnership Project (3GPP) selected polar codes to protect the control channel transmissions in fifth generation (5G) standards [38].

2.1.1 Notations

Random variables and realizations are denoted by upper-case and lower-case italic letters such as X and x respectively. Sets are denoted by upper case calligraphic letters such as $\mathcal{X}$. Probabilities will be denoted by $\mathbb{P}(.)$ or $W(.)$. Uppercase bold letters will be used to denote matrices (eg. $\mathbf{G}$). A length-N vector is denoted using a superscript as $\mathbf{x}^N$. Subscripts and superscripts are used to denote the start and end indexes of a sub-vector such as $\mathbf{x}_i^j$ meaning $(x_i, x_{i+1}, ..., x_j)$. Set notation in the subscripts of vectors indicate collection of certain elements i.e. $\mathbf{x}_{\mathcal{A}} = \{x_i : i \in \mathcal{A}\}$.

2.1.2 Preliminaries

Tools for understanding the polar codes are defined in this subsection.

Definition 1. Binary-Input Discrete Memoryless Channel (B-DMC). Denote a generic B-DMC as $W : \mathcal{X} \to \mathcal{Y}$ where $\mathcal{X}$ is the input alphabet and $\mathcal{Y}$ is the output alphabet with underlying $W(y|x)$ channel transition probabilities such that $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Input alphabet $\mathcal{X}$ will be $\{0, 1\}$. Output alphabet and channel transition probabilities may be arbitrary.

Definition 2. Memoryless Channel. Denote a generic memoryless channel as $W : \mathcal{X} \to \mathcal{Y}$ where $\mathcal{X}$ is input alphabet and $\mathcal{Y}$ is output alphabet with channel transition probabilities $W(y|x)$ such that $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Denote N uses of this channel W as $W_N : \mathcal{X}^N \to \mathcal{Y}^N$ and channel transition probabilities $W(y^N|x^N)$. Memoryless channel W satisfies

$$W_N(y^N|x^N) = \prod_{\forall i} W(y_i|x_i) \tag{2.1}$$

Definition 3. Symmetric B-DMC Channel. Denote a generic B-DMC as $W : \mathcal{X} \to \mathcal{Y}$ where $\mathcal{X}$ is the input alphabet and $\mathcal{Y}$ is the output alphabet with channel transition probabilities $W(y|x)$ such that $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. If there exists a permutation π of the output alphabet $\mathcal{Y}$ such that $\pi^{-1} = \pi$ and $W(\pi(y)|0) = W(y|1)$ for all $y \in \mathcal{Y}$.

Definition 4. Symmetric capacity of a B-DMC with channel $W : \mathcal{X} \to \mathcal{Y}$.

$$I(W) = \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)} \tag{2.2}$$

Definition 5. Kronecker Product. The Kronecker product of two arbitrary matrices $\mathbf{A}$ and $\mathbf{B}$ with dimensions $m \times n$ and $p \times q$ is another matrix $\mathbf{C}$ with dimensions $mp \times nq$ and defined as

$$\mathbf{C} \triangleq \mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} A_{11}\mathbf{B} & \dots & A_{1n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ A_{m1}\mathbf{B} & \dots & A_{mn}\mathbf{B} \end{bmatrix} \tag{2.3}$$

In addition, n^{th} Kronecker power of an arbitrary matrix $\mathbf{A}$ is denoted by using a superscript i.e. $\mathbf{A}^{\otimes n} = \mathbf{A} \otimes \mathbf{A}^{\otimes(n-1)}$.

2.1.3 Channel Polarization

Channel polarization transforms N independent copies of a given B-DMC W to N polarized channels $\{W_N^{(i)} : 1 \leq i \leq N\}$ such that a great fraction of symmetric capacities $I(W_N^{(i)})$ tend to 0 or 1. Channel polarization is achieved by two channel transformation operations namely channel combining and channel splitting.

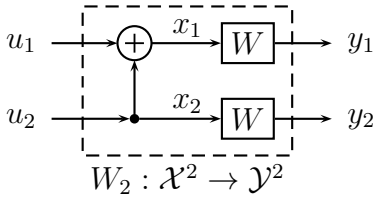

Figure 2.1: The channel W_2 is constructed from two independent copies of W.

1)Channel combining operation combines N independent copies of W and generates $W_N : \mathcal{X}^N \to \mathcal{Y}^N$ which is a vector channel of length-N. Construction of W_2 from independent copies of W is shown in Figure 2.1. Channel transition probabilities of W_2 could be calculated as the following.

$$W_2(y_1, y_2|u_1, u_2) = W(y_1|u_1 \oplus u_2)W(y_2|u_2) \tag{2.4}$$

2)Channel splitting operation splits the synthesized vector channel W_N into N binary-input DMC $W_N^{(i)} : \mathcal{X} \to \mathcal{Y}^N \times \mathcal{X}^{i-1}$. For length 2 vector channel W_2, one can produce two binary-input DMC i.e. $W_2^1 : \mathcal{X} \to \mathcal{Y}^2$ and $W_2^2 : \mathcal{X} \to \mathcal{Y}^2 \times \mathcal{X}$. Channel transition probabilities of W_2^1 and W_2^2 could be calculated as the following.

$$W_2^1(y_1, y_2|u_1) = \sum_{u_2 \in \mathcal{X}} \frac{1}{2} W(y_1|u_1 \oplus u_2) W(y_2|u_2) \tag{2.5}$$

$$W_2^2(y_1, y_2, u_1|u_2) = \frac{1}{2} W(y_1|u_1 \oplus u_2) W(y_2|u_2) \tag{2.6}$$

2.1.4 Polar Encoding

A length N polar code has a generator matrix $\mathbf{G}_n$ where $2^n = N$. Block length of polar codes are restricted to powers of two. Base case of the recursive construction is $\mathbf{G}_0 = 1$ and the recursion is the following

$$\mathbf{G}_n = \begin{bmatrix} \mathbf{G}_{n-1} & 0 \\ \mathbf{G}_{n-1} & \mathbf{G}_{n-1} \end{bmatrix} \tag{2.7}$$

It is also common to denote the recursion with Kronecker power operation i.e. $\mathbf{G}_n = \mathbf{G}^{\otimes n}$ where $\mathbf{G} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ for $n > 1$. This Kronecker product operation is performed in Galois field (GF) GF(2) arithmetic. A length-N polar encoder takes vector $\mathbf{u}^N$ as input and maps it to another vector $\mathbf{x}^N$ i.e. $\mathbf{x}^N = \mathbf{u}^N \mathbf{G}_n$. Factor graph representation of length-8 polar encoder is shown in Figure 2.2. One can observe that encoding operation has asymptotic complexity $\mathcal{O}(N \log N)$ where N is the block length.

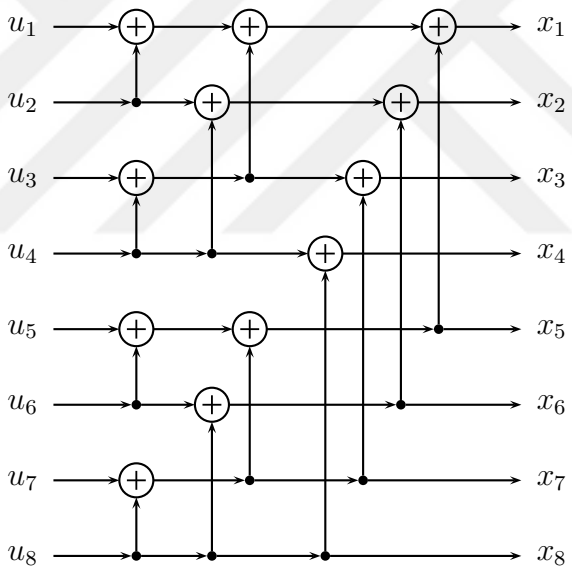

Figure 2.2: Length 8 polar encoder.

2.1.5 Code Construction Methods

Aim of the polar code construction is the identification of $\mathcal{A}$ i.e. the set of free-bit channels. For length-N and payload K polar code, K bit-channels with highest capacity is included in the set $\mathcal{A}$. Bit-channels in $\mathcal{A}$ will carry information with rate one while bit-channels in $\mathcal{A}^c$ does not carry information. Thus, data to be transmitted is inserted into $\mathbf{u}_{\mathcal{A}}$ sub-vector and remaining bit channels $\mathbf{u}_{\mathcal{A}^c}$ are set to a predetermined value. Calculation of bit-channel capacities $I(W_N^{(i)})$ are channel dependent. Exact calculation for binary erasure channel (BEC) BEC(ϵ) is derived in [37]. This calculation becomes infeasible when the channel is not BEC due to exponential increase in the output alphabet proportional block length. Density Evolution method [39] is used in LDPC code designs where several convolution operations are involved. Density Evolution method for polar codes is described in [40] and [41], authors developed an algorithm that limits the output alphabet size to compute bit-channel capacities.

2.1.6 Systematic Polar Coding

Systematic polar codes are introduced in [42]. BER performance of the polar codes could be increased while preserving the complexity of encoding and decoding same as non-systematic polar coding. Consider the composition of polar codeword $\mathbf{x}$ as in Equation 2.8 where set $\mathcal{A}$ is the indices of free-bit channel and $\mathbf{G}_{\mathcal{A}}$ is the sub-matrix of $\mathbf{G}$ consisting of row indices in $\mathcal{A}$.

$$\mathbf{x} = \mathbf{u}_{\mathcal{A}}\mathbf{G}_{\mathcal{A}} + \mathbf{u}_{\mathcal{A}^c}\mathbf{G}_{\mathcal{A}^c} \tag{2.8}$$

Now partition the polar codeword $\mathbf{x} = (\mathbf{x}_{\mathcal{B}}, \mathbf{x}_{\mathcal{B}^c})$ where $\mathcal{B}$ is an arbitrary subset of indices. Equation 2.8 could be rewritten in terms of $\mathbf{x}_{\mathcal{B}}$ and $\mathbf{x}_{\mathcal{B}^c}$ as in Equations 2.9 and 2.10 where $\mathbf{G}_{\mathcal{AB}}$ is a sub-matrix of $\mathbf{G}$ with elements $\mathbf{G}_{i,j} : i \in \mathcal{A}, j \in \mathcal{B}$. We will set $\mathcal{A} = \mathcal{B}$ to establish a one-to-one correspondence between $\mathbf{u}_{\mathcal{A}}$ and $\mathbf{x}_{\mathcal{B}}$.

$$\mathbf{x}_{\mathcal{B}} = \mathbf{u}_{\mathcal{A}}\mathbf{G}_{\mathcal{AB}} + \mathbf{u}_{\mathcal{A}^c}\mathbf{G}_{\mathcal{A}^c\mathcal{B}} \tag{2.9}$$

$$\mathbf{x}_{\mathcal{B}^c} = \mathbf{u}_{\mathcal{A}}\mathbf{G}_{\mathcal{AB}^c} + \mathbf{u}_{\mathcal{A}^c}\mathbf{G}_{\mathcal{A}^c\mathcal{B}^c} \tag{2.10}$$

Figures 2.3 and 2.4 show the difference in the BER and FER between the standard and systematic polar code under SC decoding. Simulations are carried out using AWGN channel and BPSK modulation.

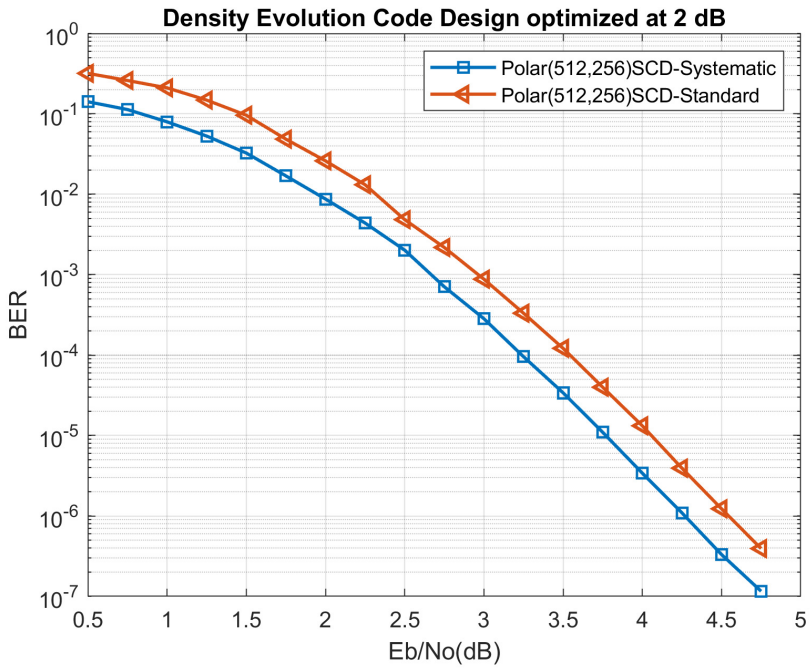

Figure 2.3: BER performance of rate 0.5 polar codes under SC decoding on AWGN channel and BPSK modulation.

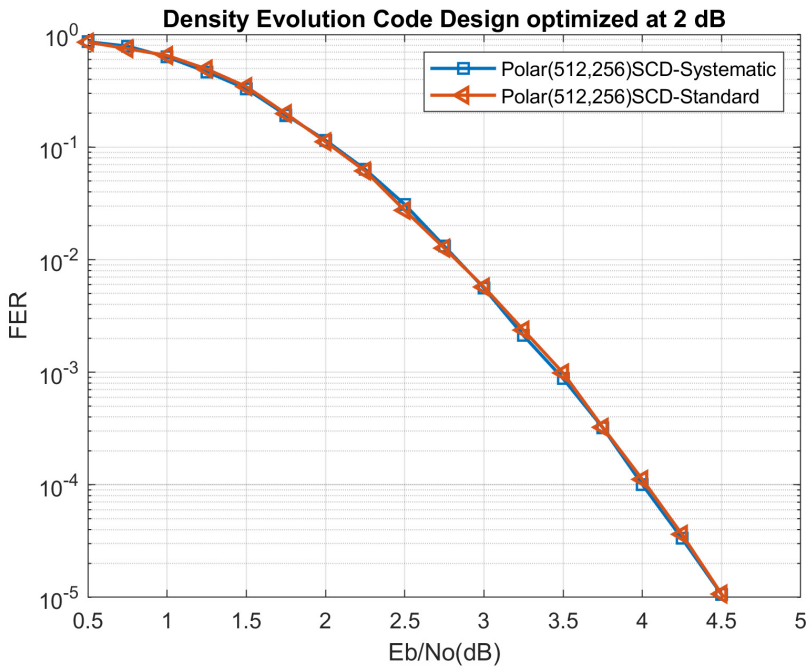

Figure 2.4: FER performance of rate 0.5 polar codes under SC decoding on AWGN channel and BPSK modulation.

2.1.7 Decoding Algorithms

Polar decoding algorithms are explained in this subsection. Successive cancellation (SC) and successive cancellation list (SCL) algorithms are two well-known polar decoding algorithms.

2.1.7.1 Successive Cancellation Decoding Algorithm

SC decoding algorithm is a depth-first search algorithm with asymptotic complexity $\mathcal{O}(N \log N)$ [37]. Data on the i^{th} bit-channel is estimated using Equation 2.11. One can observe that decoding of $\hat{u}_i$ depends on the previous bit channel estimates. Thus, decoding takes place one by one from the first bit channel to the last.

$$\hat{u}_i = \begin{cases} \text{argmax}_{u_i} = \{W(y^N, u_1^{i-1}|u_i = 0), W(y^N, u_1^{i-1}|u_i = 1)\} & i \in \mathcal{A} \\ 0 & i \in \mathcal{A}^c \end{cases} \tag{2.11}$$

We will use log-likelihood ratio (LLR) representation of the channel probabilities as in Equation 2.12 where $W(y_i|x_i)$ is the channel transition probability density function. Decision LLRs are shown in Equation 2.13 where positive decision LLR will be decoded as $\hat{u}_i = 0$ and negative decision LLR will be decoded as $\hat{u}_i = 1$.

$$\ell_i = \log\left(\frac{W(y_i|x_i = 0)}{W(y_i|x_i = 1)}\right) \tag{2.12}$$

$$\ell_i = \log\left(\frac{W(y_i, u_1^{i-1}|u_i = 0)}{W(y_i, u_1^{i-1}|u_i = 0)}\right) \tag{2.13}$$

SC decoder calculates the decision LLRs from channel LLRs using the update Equations 2.14 and 2.15 [43] where ℓ denotes the LLR values and $u \in \{0, 1\}$.

$$\text{f}(\ell_1, \ell_2) = 2\text{tanh}^{-1}\left(\text{tanh}(\frac{\ell_1}{2})\text{tanh}(\frac{\ell_2}{2})\right) \tag{2.14}$$

$$\text{g}(\ell_1, \ell_2, u) = (-1)^u\ell_1 + \ell_2 \tag{2.15}$$

Equation 2.16 approximates Equation 2.14. We will refer to Equation 2.16 as min-sum approximation of Equation 2.14.

$$\text{f}(\ell_1, \ell_2) \approx \text{sgn}(\ell_1\ell_2)\text{min}(|\ell_1|, |\ell_2|) \tag{2.16}$$

Algorithm 1 is the pseudo-code for LLR based successive cancellation decoder where the input ℓ, N and FI denotes the LLR, block length and frozen index vector, respectively. Frozen index vector is a binary vector where frozen bit-channels are indicated with a one and remaining bit-channels are indicated with a zero. Note that output vector $u_{\mathcal{A}}$ contains the estimate of the information bits. One can observe the recursive code structure where a length-N polar code consists of two length-$N/2$ polar codes.

Algorithm 1: LLR based successive cancellation Decoder.

Input: ℓ, N, FI **Output:** u
Function Call: u=**SC**(ℓ,N,FI)
if $N == 1$ **then**
    if $FI == 0$ **then**
        $u = (\ell > 0)\ ?\ 0 : 1;$
    else
        $u = 0;$
else
    $\ell' = \mathrm{f}(\ell_1^{N/2},\ \ell_{N/2+1}^{N});$
    $u' = \mathbf{SC}(\ell',\ N/2,\ FI_1^{N/2});$
    $\ell'' = \mathrm{g}(\ell_1^{N/2},\ \ell_{N/2+1}^{N},\ u');$
    $u_{N/2+1}^{N} = \mathbf{SC}(\ell'',\ N/2,\ FI_{N/2+1}^{N});$
    $u_1^{N/2} = u' \oplus u_{N/2+1}^{N};$

Certain algebraic simplifications are presented in [44] and [45]. In order to satisfy the throughput requirements, we will prefer simplified version of the SC decoding algorithm. Figures 2.5 and 2.6 demonstrate the performance of polar codes under SC decoding. All codes have rate 0.5 and simulations are carried out using AWGN channel and BPSK modulation. Density evolution method is used to determine the free bit-channels at 2 dB for all polar codes.

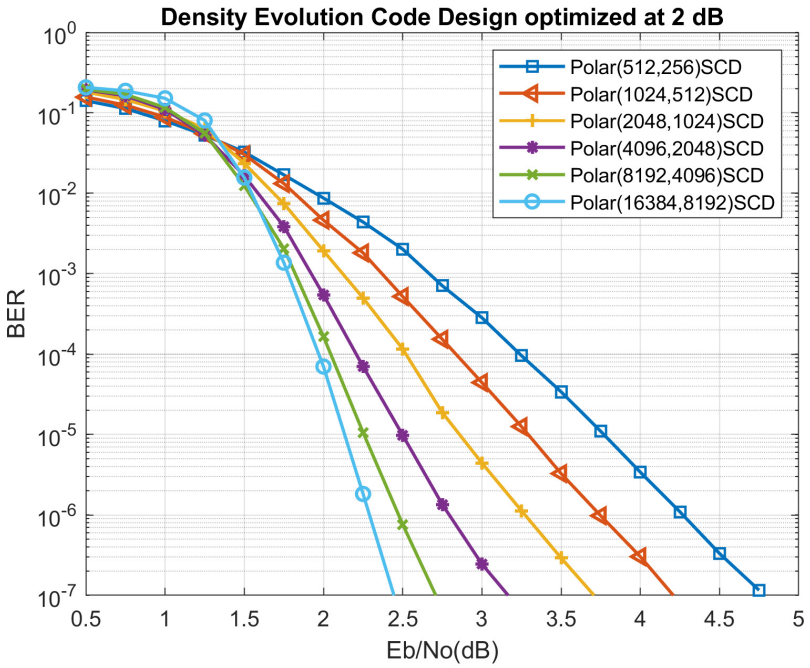

Figure 2.5: BER performance of rate 0.5 polar codes under SC decoding on AWGN channel and BPSK modulation.

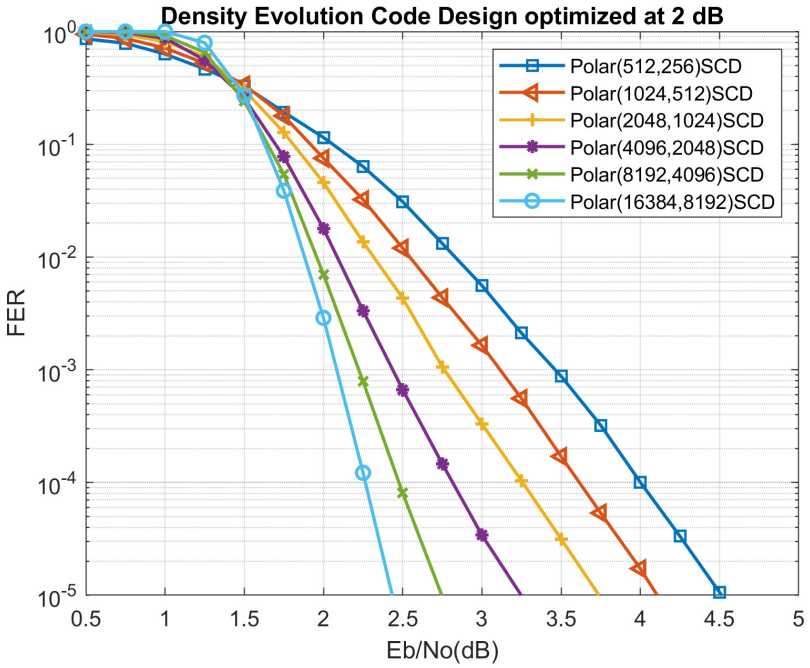

Figure 2.6: FER performance of rate 0.5 polar codes under SC decoding on AWGN channel and BPSK modulation.

2.1.7.1.1 Area Estimation of SC Decoder An area estimation methodology of SC decoding algorithm with unrolled and pipelined architecture is presented in this part. This estimation should serve as a basis for the code designer in assessment of the hardware complexity of the decoders without performing any synthesis. Number of logic gates and memory elements required for the decoder is calculated using software tools. We have expressed basic decoding functions described in Section 2.1.7.1 with the help of [46] as logic gates. Number of pipeline states is estimated as proportional to the total number of decoding operations. D-type flip-flops (FFs) are used for memory elements. By using TSMC 28nm HPC standard cell library, we have generated a data sheet of components and its area. Two area estimates are generated one consisting of only drive-one components and another consisting of mixture of drive-one to drive-eight components. Arithmetic mean of these two estimates (E_{avg}) is reported as the final area estimate. 70% utilization is assumed for wire area. In [47], area of the Polar(1024,512) code with unrolled and fully pipelined decoder is reported as 4.63 mm^2 in 28nm technology node. By using the developed methodology, area of the same decoder is estimated as $E_{avg} = 3.6$ mm^2 which is reasonably close.

2.1.7.2 Successive Cancellation List Decoding Algorithm

Successive cancellation list decoding algorithm is presented in [48]. Instead of tracking a single path in the depth-first search, SCL algorithm keeps track of L candidate codewords at any depth. This search method is known as the M-Algorithm [49]. Candidate codewords are compared by a metric which is a summary of the decoding history of the codeword. LLR-based SCL algorithm is presented in [50] where comparison metric simplifies to accumulation of contrasting decisions. Equations 2.17 and 2.18 are used to calculate the path metric PM_i of a candidate codeword at depth i. Here ℓ_i and u_i are the decision LLR and the decision of the i^{th} bit-channel, respectively.

$$PM_i = \phi(PM_{i-1}, \ell_i, u_i) \tag{2.17}$$

$$\phi(PM, \ell, u) = \begin{cases} PM & u = \frac{1}{2}(1 - \text{sgn}(\ell)) \\ PM + |\ell| & u \neq \frac{1}{2}(1 - \text{sgn}(\ell)) \end{cases} \tag{2.18}$$

Asymptotic run-time complexity of the SCL algorithm is $\mathcal{O}(LNlgN)$ where L is the list size and N is the block length. Complexity of path metric comparisons is excluded in this notation. One can observe that, for sufficiently large list sizes, comparison of path metrics between candidate codewords becomes the dominating factor for the decoding complexity. Algorithm 2 is the pseudo-code for LLR based SCL decoder where index i is used for bit-channels, index j is used for lists and $\mathcal{L}$ denotes the set of active decoding paths. Duplicate function appends the input to itself in the column direction such that the output has twice the column length of the input.

Algorithm 2: LLR based successive cancellation list decoder.

```
if i ∈ A^c then
    foreach ℓ_{i,j} ∈ L do
        PM_{i,j} = φ(PM_{i-1,j}, ℓ_{i,j}, u_i);
    u_{i,j} = u_i;
else
    Duplicate(PM_i, L);
    foreach ℓ_j ∈ L do
        if j < |L|/2 then
            PM_{i,j} = φ(PM_{i-1,j}, ℓ_{i,j}, 0);
            u_{i,j} = 0;
        else
            PM_{i,j} = φ(PM_{i-1,j}, ℓ_{i,j}, 1);
            u_{i,j} = 1;
    if |L| > ListSize then
        [ v,idx] = sort(PM_{i,j});
        PM_{i,j} = v_1^{ListSize};
        u_{i,j} = u_{i,idx_1^{ListSize}};
        L = L_{idx_1^{ListSize}};
```

Similar to SC decoding algorithm, certain algebraic simplifications are introduced

to SCL algorithm [51]. In order to satisfy the throughput requirements, we will prefer simplified version of the SCL algorithm. Figures 2.7 and 2.8 demonstrate the performance of rate 0.5 polar codes under SCL decoding. Simulations are carried out using AWGN channel and BPSK modulation. Density evolution method is used to determine the free bit-channels at 2 dB.

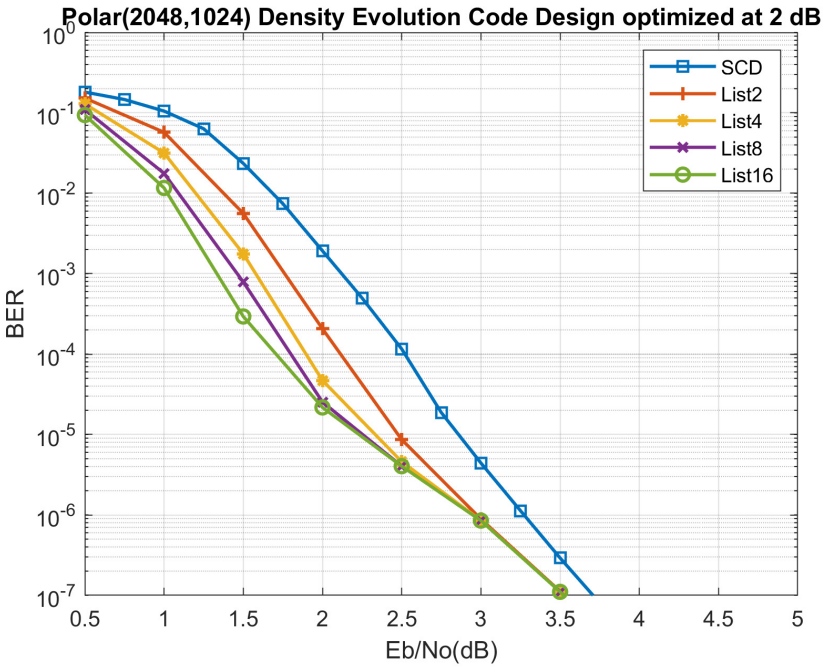

Figure 2.7: BER performance of rate 0.5 polar codes under SCL decoding on AWGN channel and BPSK modulation.

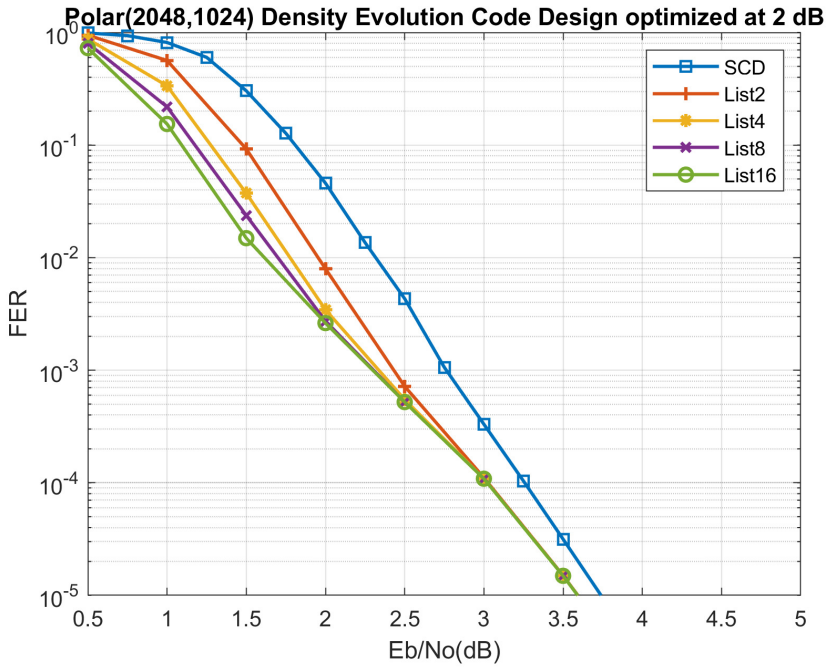

Figure 2.8: FER performance of rate 0.5 polar codes under SCL decoding on AWGN channel and BPSK modulation.

2.2 Reed-Solomon Codes

Reed-Solomon codes are described in [52]. This code becomes desirable when correlated binary errors or burst errors occur. An RS code is described as RS$m(N,K)$ where m is the order of the finite Galois field GF(2^m), N is the block length in symbols and K is the payload in symbols. Note that GF(2^m) contains $2^m - 1$ distinct elements, thus block length of the RS code is bounded by $N \leq 2^m - 1$. RS codes exploit the maximum distance separable (MDS) property since they achieve Singleton bound. An RS$m(N,K)$ code can correct up to $2t + e \leq N - K$ error and erasure combinations where t is the number of errors and e is the number of erasures. For example, RS8(255,223) code is defined over GF(2^8) and can correct up to 16 errors $t \leq 16$ or 32 erasures $e \leq 32$. This code is known as the National Aeronautics and Space Administration (NASA) code which is used in deep space communications.

2.2.1 Analytical Error Performance and Emulation Methods

Solid behavior of RS decoders allows performance estimation by analytical calculations with negligible errors. In [53], authors derived the exact probability of undetected error for a given RS code. Undetected errors occur if the received pattern falls into the radius t Hamming sphere of another codeword in the codebook. In Equation 2.19, $P_E(u)$ denotes the probability of undetected error event within a distance u codeword ($u > t$), D_u denotes the weight enumerator function of the RS code and $d = n - k + 1$. $P_E(u)$ can become significantly large if t is small compared to m.

$$P_E(u) = \frac{D_u}{\binom{N}{u}(m-1)^u} \quad d - t \leq u \leq N \tag{2.19}$$

Keeping $P_E(u)$ in mind, BER characteristics of RS codes can be estimated for given symbol error probability P_{SE}. In Equation 2.20, P_{UE} is the probability of

an uncorrectable symbol error.

$$P_{UE} = \sum_{i=t+1}^{N} \frac{i}{N} \binom{N}{i} \times (P_{SE})^i \times (1 - P_{SE})^{(N-i)} \tag{2.20}$$

Input and output BER of the RS decoder could be calculated by

$$BER_{in} = 1 - (1 - P_{SE})^{\frac{1}{m}} \tag{2.21}$$

$$BER_{out} = 1 - (1 - P_{UE})^{\frac{1}{m}} \tag{2.22}$$

Figures 2.9 and 2.10 show the comparison between performance estimates and simulations. FER is estimated using [54] assuming FER $= 1 - P_{CD}$ where P_{CD} is the probability of correct decoding of a RS codeword. Simulations have been carried out using AWGN channel with BPSK modulation.

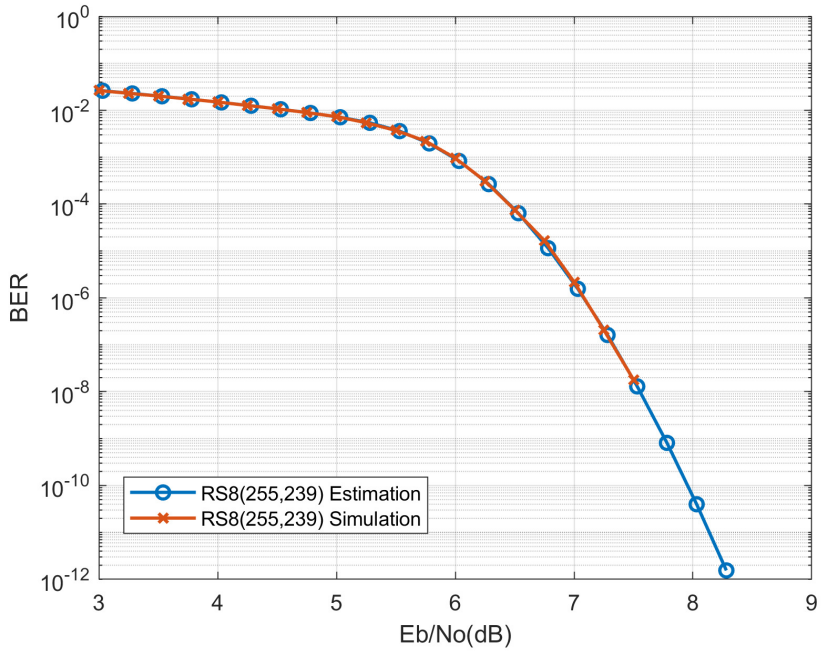

Figure 2.9: BER performance comparison between simulations and analytical estimations.

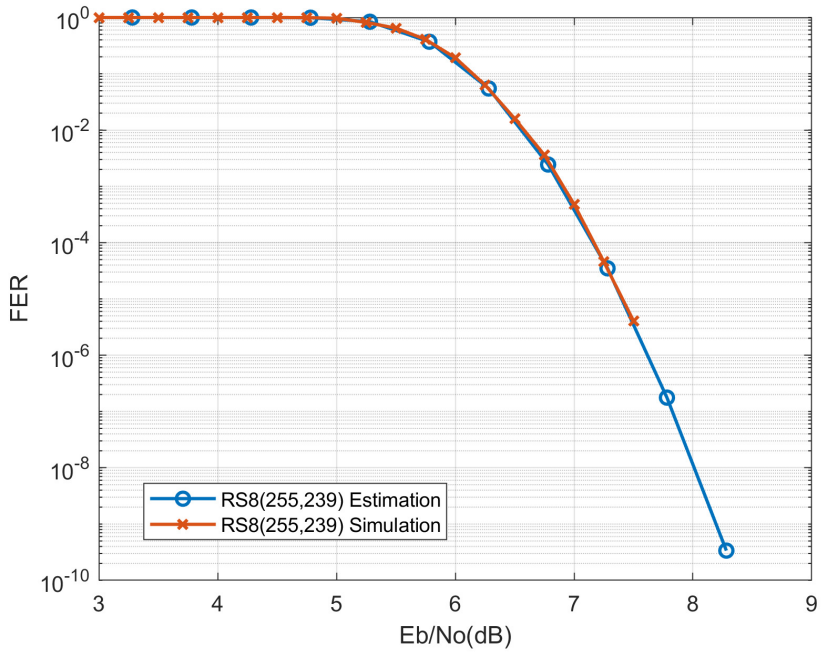

Figure 2.10: FER performance comparison between simulations and analytical estimations.

2.3 Product Codes

Product codes are first introduced by Elias in [55]. Product codes benefit from combined error correction capability of simple component codes. A product code can be built by interlacing two codes C_1 and C_2, called as component codes. In Figure 2.11, product code structure is shown with component codes $C_1(N_1, K_1)$ and $C_2(N_2, K_2)$. In this thesis, focus will be on product codes with two dimensional structures as in Figure 2.11. Features of the product codes are described in [56], [57] and [58].

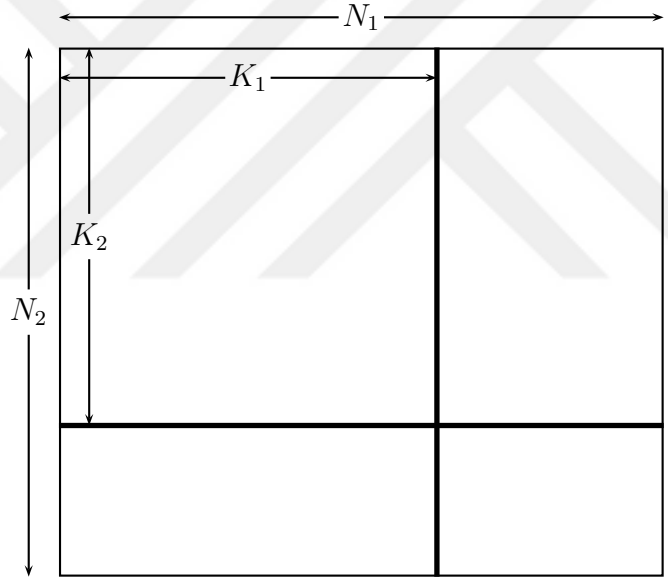

Figure 2.11: Product code structure with component codes $C_1(N_1, K_1)$ and $C_2(N_2, K_2)$.

Product codes can correct at least $(t_1+1)(t_2+1)-1$ errors by using t_1 and t_2-error correcting component codes. $P_{e,j}$ denotes the probability that a j-error pattern could not be decoded. $P_{e,j}$ is upper bounded in Equation 2.25 where component codes are t_1 and t_2-error correcting codes.

$$\tau = (t_1 + 1)(t_2 + 1) - 1 \tag{2.23}$$

$$P_{e,\tau+1} \leq \frac{\binom{N_1}{t_1+1}\binom{N_2}{t_2+1}^{t_1+1}}{\binom{N_1 N_2}{\tau+1}} \tag{2.24}$$

$$P_{e,j} \leq \binom{j}{\tau+1} P_{e,\tau+1}, \quad j > \tau + 1 \tag{2.25}$$

2.4 Concatenated Codes

Concatenated codes are presented in [59] where long codes are built from shorter ones. Aim of the concatenation is to combine the good parts of two codes to diminish the weakness of the overall code. A concatenated FEC code consists of at least two component codes. One of the codes could operate above the channel capacity while the other code is operating near channel capacity in order to satisfy the target coding rate. In the previous chapter, we have seen many concatenated codes where soft and hard information is exchanged between codes. Concatenated codes become favorable due to implementation constraints compared to longer codes. In [60], authors have developed a concatenation scheme where polar and RS codes are involved. Several RS codes operate on different parts of the multiple polar blocks. A common concatenation for polar code is cyclic-redundancy-check (CRC) when SCL algorithm is used to decode polar code [48].

2.5 Summary of the Chapter

A review of FEC codes are presented in this chapter including polar, RS, product and concatenated codes. SC and SCL decoding algorithms are preferred for polar codes. Density Evolution code construction along with systematic polar coding is used to improve the BER performance. Analytical error performance and emulation methods are used for RS code analysis. These codes will serve as a basis to develop the polar product RS concatenated code.

Chapter 3

RS2-Polar Concatenation Scheme

Polar product RS (RS2-Polar) concatenation scheme is explained in this chapter. Outer encoder and decoder pairs are described in Sections 3.1 and 3.4, respectively. Interleaver and segmentation block is described in Section 3.2. Inner encoder and decoder pairs are described in Section 3.3. Polar codes are used as inner codes with rates slightly above the channel capacity. RS codes are used as outer codes in a product form with rates closer to unity. Our aim is to collect the soft information from the channel output as much as possible using polar codes. Two-error correcting RS codes are used as component codes. We will present code configurations with 28%, 24%, 20% and 15% overheads.

Figure 3.1 presents the transmitter and receiver blocks. Transmitter chain begins with the product RS encoder and finishes with the modulator. Receiver chain begins with the demapper and finishes with the product RS decoder. BPSK modulation is used in simulations. Demapper calculates LLRs from channel output. We have used the MDS property of RS codes to emulate the error correction performance. In Figure 3.1, we denote the input and output of the outer encoder with K_r and N_r, respectively. We denote the input and output of the inner encoder with K_p and N_p, respectively.

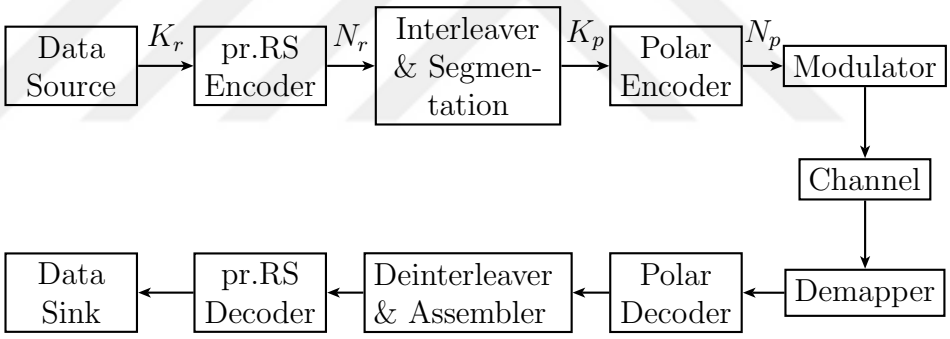

Figure 3.1: Transmitter and receiver chain.

Code parameters with a short description is listed below.

- Outer Encoder
  - m: Symbol length of the component codes.
  - n: Block length of the component codes in symbols.
  - k: Payload of the component codes in symbols.
  - K_r: Total number of input bits.
  - N_r: Total number of output bits.
- Interleaver & Segmentation
  - S_b: Code strand, a collection of n symbols from $n \times n$ matrix ($n \times m$ bits).
  - b: Code strand index ($0 \leq b < n$).
  - $S_b^{q'}$: Segmented portion of code strand S_b ($m \times n/q$ bits).
  - q: Number of segments derived from S_b.
  - q': Segmented code strand index ($0 \leq q' < q$).
- Inner Encoder
  - T: Polar block index.
  - K: Payload of the inner encoder in bits.
  - N: Block length of the inner encoder in bits.

We present an example of the developed RS2-Polar code scheme. Product RS code is used with RS4(15,11) component codes. Polar(32,20) code is used as inner code and SC decoder is used as inner decoder. Four bits are grouped to generate symbols to be encoded. Please note that m equals 4. 121 data symbols (484 data bits) are put into 11×11 matrix as in Figure 3.2. Next, row and parity symbols are generated as explained in Section 3.1. 225 symbols are sent to the interleaver and segmentation block with parameters n equals 15 and q equals 3. Figure 3.3 shows the polar block indexes in each cell. For example, S_2^0 is the first sub vector

the second code strand and could be calculated by applying Equations 3.1, 3.2 and 3.3. We begin with the calculation of S_2 by using Equation 3.1.

$$S_2 = \{(0,2),(1,3),(2,4),(3,5),(4,6),(5,7),(6,8),...,(13,0),(14,1)\}$$

Next, Equation 3.2 is used to align the starting index.

$$S_2 = \{(2,4),(3,5),(4,6),(5,7),(6,8),...,(13,0),(14,1),(0,2),(1,3)\}$$

Finally, Equation 3.3 is used to split S_b into three equal pieces.

$$S_2^0 = \{(2,4),(3,5),(4,6),(5,7),(6,8)\} = \{6_0, 6_1, 6_2, 6_3, 6_4\}$$

Thus, first polar block encodes 1/3 of the diagonal which corresponds to 20 bits (5 symbols). Polar encoder is used 45 times to encode all symbols. 1440 coded bits are generated by multiple use of polar encoder and sent to modulator. Modulated symbols pass through AWGN channel. Inner decoder receives LLRs calculated by demapper block. SC decoder calculates an estimate of the transmitted codeword per received polar frame. SC decoder is used 45 times and 900 bits (225 symbols) are sent to deinterleaver and assembler block. Received bits are formed into groups of four bits to generate symbols. These symbols are placed into a 15 $\times$ 15 matrix as in Figure 3.3. Next, outer decoding operation begins with the RS4(15,11) decoders in the horizontal direction and followed by the codes in the vertical direction. After decoding on horizontal and vertical axes several times, 484 data bits are sent to data sink.

3.1 Outer Encoder

Outer encoder is explained in this section. Product RS encoder is used with two-error correcting identical component codes on both axes. Figure 3.2 depicts the product code structure where n is the block length of the component code and k is the payload of the component code. Dashed lines indicate the position of component codes in both dimensions. Cross hatched box indicates a collection of m-bits. K_r bits are received by outer encoder and placed into $k \times k$ matrix.

Component codes in the horizontal direction generate row parity symbols by encoding each row in $k \times k$ matrix. Generated row parity symbols are written into $(n-k) \times k$ matrix. Component codes in the vertical direction generate column parity symbols by encoding each column in $k \times n$ matrix. Generated column parity symbols are written into $(n-k) \times n$ matrix. After generating row and column parity symbols, outer encoder delivers N_r bits to the next block. Table 3.1 shows the component codes that are used in different code configurations. Minimum distance of the designed product codes is equal to 9.

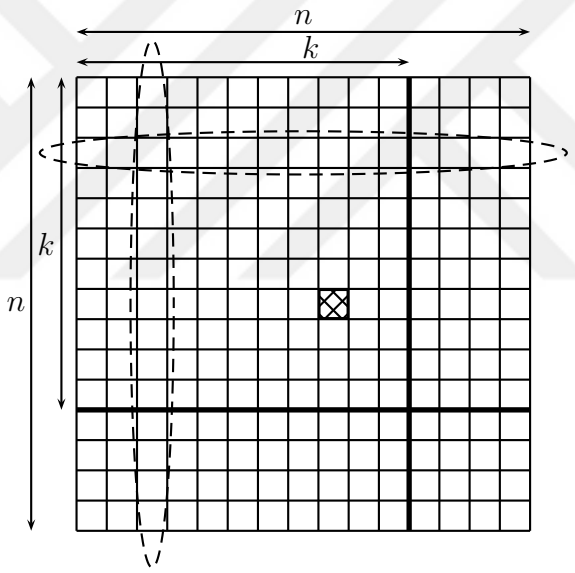

Figure 3.2: Product RS code structure.

Block-length N_r (bits)	Payload K_r (bits)	Rate (R_o)	Component Code
346,112	332,928	0.9619	RS8(208,204)
346,112	332,928	0.9619	RS8(208,204)
111,132	104,188	0.9375	RS7(126,122)
362,952	349,448	0.9627	RS8(213,209)
394,272	380,192	0.964	RS8(222,218)
419,528	405,000	0.9825	RS8(229,225)

Table 3.1: Outer code parameters.

3.2 Interleaver & Segmentation

Interleaver and segmentation block is explained in this section. This block is designed to align multiple polar blocks in a way that each component RS code protects different symbols of different polar blocks. Functionality of the designed interleaver and segmentation block could be summarized by applying Equations 3.1, 3.2 and 3.3 to N_r input bits. S_b is a vector of two tuple indexes. Equations 3.1 and 3.2 are used to form S_b where i and j denotes the row and column indexes, respectively. Subscript b denotes the code strand index where b is a natural number less than n.

Equation 3.1 selects a diagonal for each code strand where the starting point is the first row. For example, S_0 contains elements that are in the diagonal starting from first row and first column and S_1 contains elements that are in the sub diagonal starting from first row and second column. After applying Equation 3.1, n code strands are generated from an $n \times n$ matrix. Equation 3.2 shifts the starting point of each code strand by b elements which is a cyclic permutation of the corresponding code strand. For example, each element of S_1 is circular shifted by one and each element of S_2 is circular shifted by two. After applying Equation 3.2, S_1 begins at the second row and third column and S_2 begins at the third row and fifth column. $S_b^{q'}$ is generated by applying Equation 3.3 which divides S_b to q sub vectors. Equation 3.3 is optional for some code configurations.

$$S_b = \{(i,j) : 0 \leq i < n, j = \mathrm{mod}(i+b, n)\} \tag{3.1}$$

$$S_b = \mathrm{circshift}(S_b, b) \tag{3.2}$$

$$S_b^{q'} = \{S_b(q'\lfloor\frac{n}{q}\rfloor), ..., S_b((q'+1)\lfloor\frac{n}{q}\rfloor - 1)\} \quad 0 \leq q' < q \tag{3.3}$$

Figure 3.3 shows the realization of the interleaver and segmentation block with parameters $n = 15$ and $q = 3$. Each cell is enumerated by T_d where T corresponds to the segmented strand index ($T = b \times q + q'$) and subscript d corresponds to the m bits symbol index within a segmented strand. Subscript d is a natural number less than or equal to $\frac{n}{q}$. Please refer to Appendix A for detailed analysis of the interleaver block.

0_0	5_4	8_3	11_2	14_1	17_0	19_4	22_3	25_2	28_1	31_0	33_4	36_3	39_2	42_1
42_2	0_1	3_0	8_4	11_3	14_2	17_1	20_0	22_4	25_3	28_2	31_1	34_0	36_4	39_3
39_4	42_3	0_2	3_1	6_0	11_4	14_3	17_2	20_1	23_0	25_4	28_3	31_2	34_1	37_0
37_1	40_0	42_4	0_3	3_2	6_1	9_0	14_4	17_3	20_2	23_1	26_0	28_4	31_3	34_2
34_3	37_2	40_1	43_0	0_4	3_3	6_2	9_1	12_0	17_4	20_3	23_2	26_1	29_0	31_4
32_0	34_4	37_3	40_2	43_1	1_0	3_4	6_3	9_2	12_1	15_0	20_4	23_3	26_2	29_1
29_2	32_1	35_0	37_4	40_3	43_2	1_1	4_0	6_4	9_3	12_2	15_1	18_0	23_4	26_3
26_4	29_3	32_2	35_1	38_0	40_4	43_3	1_2	4_1	7_0	9_4	12_3	15_2	18_1	21_0
21_1	24_0	29_4	32_3	35_2	38_1	41_0	43_4	1_3	4_2	7_1	10_0	12_4	15_3	18_2
18_3	21_2	24_1	27_0	32_4	35_3	38_2	41_1	44_0	1_4	4_3	7_2	10_1	13_0	15_4
16_0	18_4	21_3	24_2	27_1	30_0	35_4	38_3	41_2	44_1	2_0	4_4	7_3	10_2	13_1
13_2	16_1	19_0	21_4	24_3	27_2	30_1	33_0	38_4	41_3	44_2	2_1	5_0	7_4	10_3
10_4	13_3	16_2	19_1	22_0	24_4	27_3	30_2	33_1	36_0	41_4	44_3	2_2	5_1	8_0
8_1	11_0	13_4	16_3	19_2	22_1	25_0	27_4	30_3	33_2	36_1	39_0	44_4	2_3	5_2
5_3	8_2	11_1	14_0	16_4	19_3	22_2	25_1	28_0	30_4	33_3	36_2	39_1	42_0	2_4

Figure 3.3: Interleaver & segmentation alignment for $n = 15$ and $q = 3$.

3.3 Inner Encoder & Decoder Pairs

Inner encoder and decoder pairs for various code configurations are presented in this section. Table 3.2 summarizes the code and decoder pairs. 28% overhead is targeted with the codes on the first and second row. 24% overhead is targeted with the codes on the third and fourth row. 20% overhead is targeted with the code on the fifth row and 15% overhead is targeted with the code on the last row. Polar codes are designed in low-SNR since we prefer improved BER performance at operating points. We could sacrifice from minimum distance of the inner code to improve the BER performance in the low-SNR.

Block Length (bits)	Payload (bits)	Code Rate	Code Design	Design E_b/N_o (dB)	Decoder
512	416	0.8125	Density Evolution	2.4	SCL4
2048	1664	0.8125	Gaussian Approximation	2.4	SC
1024	882	0.8613	Gaussian Approximation	2.1	SC
2048	1704	0.832	Gaussian Approximation	2.8	SC
2048	1776	0.8672	Gaussian Approximation	3.1	SC
2048	1832	0.8945	Gaussian Approximation	3	SC

Table 3.2: Inner code parameters of various code configurations.

Figures 3.4 and 3.5 show the BER and FER performance comparison of inner polar codes and decoder pairs. One can observe that even the minimum distance of the Polar(512,416) is worse than Polar(2048,1664), SCL algorithm improves the error correction performance in the desired SNR points.

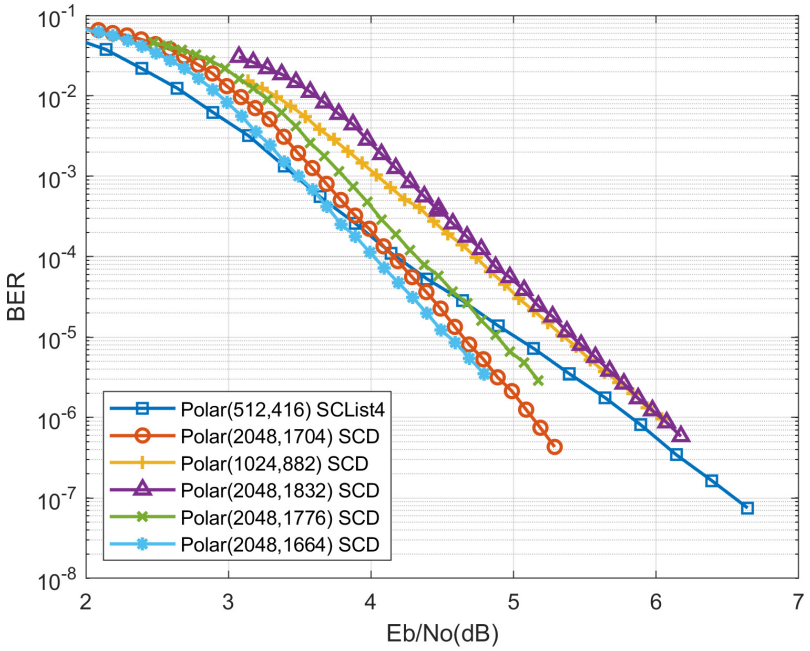

Figure 3.4: BER performance comparison of inner polar codes and decoder pairs on AWGN channel and BPSK modulation.

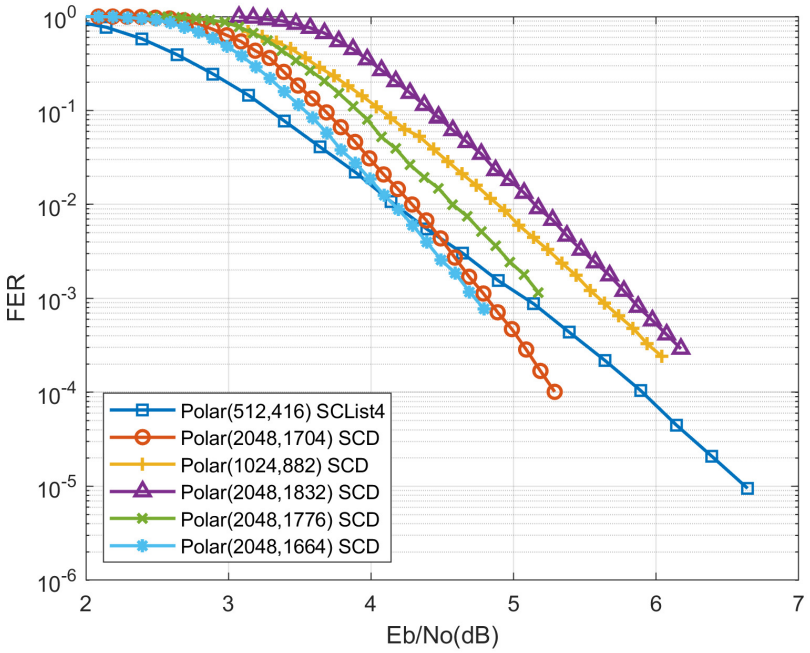

Figure 3.5: FER performance comparison of inner polar codes and decoder pairs on AWGN channel and BPSK modulation.

3.3.1 Polar(512,416) SC List4

Systematic polar encoding is performed over each $S_b^{q'}$ and sent to modulator. Polar(512,416) code is optimized at 2.4 E_b/N_o (dB) with density evolution method. Enumeration in Figure 3.3 also explains the polar encoding strategy where variable T indicates the collection $m \times d$ bits to be encoded. Received LLRs are decoded using polar successive cancellation with a list of 4 decoder and sent to deinterleaver block. Inner and outer code rates are $R_i = 0.8125$ and $R_o = 0.9619$, respectively.

3.3.2 Polar(2048,1664) SC

Systematic polar encoding is performed over each S_b after applying Equation 3.2. In other words, code strands are used without further segmentation. Polar(2048,1664) code is optimized at 2.4 E_b/N_o (dB) with Gaussian approximation method. Received LLRs are decoded using polar SC decoder and sent to deinterleaver block. Inner and outer code rates are $R_i = 0.8125$ and $R_o = 0.9619$, respectively.

3.3.3 Polar(1024,882) SC

Systematic polar encoding is performed over each S_b after applying Equation 3.2. Polar(1024,882) code is optimized at 2.1 E_b/N_o (dB) with Gaussian approximation method. Received LLRs are decoded using polar SC decoder and sent to deinterleaver block. Please note that product RS code with RS7(126,122) component code is used as outer code. Here $m = 7$, $n = 126$, $k = 122$ and $q = 1$. Inner and outer code rates are $R_i = 0.8613$ and $R_o = 0.9375$, respectively.

3.3.4 Polar(2048,1704) SC

Systematic polar encoding is performed over each S_b after applying Equation 3.2. Polar(2048,1704) code is optimized at 2.8 E_b/N_o (dB) with Gaussian approximation method. Received LLRs are decoded using polar SC decoder and sent to deinterleaver block. Please note that product RS code with RS8(213,209) component code is used as outer code. Here $m = 8$, $n = 213$, $k = 209$ and $q = 1$. Inner and outer code rates are $R_i = 0.832$ and $R_o = 0.9627$, respectively.

3.3.5 Polar(2048,1776) SC

Systematic polar encoding is performed over each S_b after applying Equation 3.2. Polar(2048,1776) code is optimized at 3.1 E_b/N_o (dB) with Gaussian approximation method. Received LLRs are decoded using polar SC decoder and sent to deinterleaver block. Please note that product RS code with RS8(222,218) component code is used as outer code. Here $m = 8$, $n = 222$, $k = 218$ and $q = 1$. Inner and outer code rates are $R_i = 0.867$ and $R_o = 0.964$, respectively.

3.3.6 Polar(2048,1832) SC

Systematic polar encoding is performed over each S_b after applying Equation 3.2. Polar(2048,1832) code is optimized at 3 E_b/N_o (dB) with Gaussian approximation method. Received LLRs are decoded using polar SC decoder and sent to deinterleaver block. Please note that product RS code with RS8(229,225) component code is used as outer code. Here $m = 8$, $n = 229$, $k = 225$ and $q = 1$. Inner and outer code rates are $R_i = 0.8945$ and $R_o = 0.9653$, respectively.

3.4 Outer Decoder

Outer decoder is presented in this section. This block takes input from the inner polar decoder. Product RS code is used as outer code with two-error correcting component codes. Product code structure allows iterative decoding between component codes in different axes. A decoding iteration begins with the RS decoding of component codes on the horizontal axis followed by component codes on the vertical axis. There could be further simplifications in the decoding circuitry by processing the information between decoding iterations. For example, component codes that have performed error correction in the previous iteration may not be activated in the current iteration. Required decoder circuitry is diminished by using identical component codes on vertical and horizontal axes. Figure 3.6 shows the BER performance of rate 0.96 product RS code with RS8(208,204) component code.

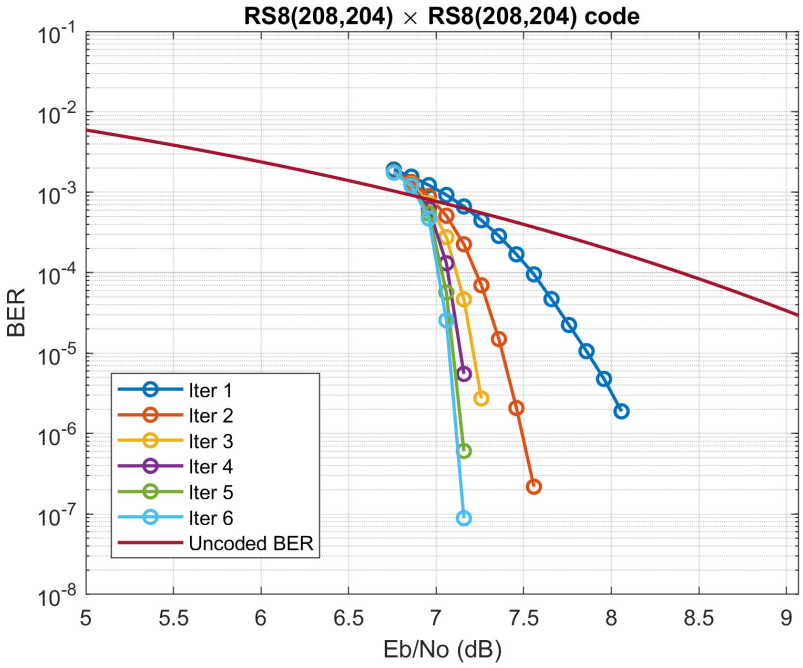

Figure 3.6: BER performance of rate 0.96 product RS code with RS8(208,204) component code. 6 decoding iterations on AWGN channel and BPSK modulation.

3.5 Simulation Results

Simulation results are presented in this section. Simulations are carried out using MATLAB. Channel outputs are represented as floating point numbers. Extrapolation methods are used to estimate the net coding gain at BER 10^{-15}. Table 3.3 summarizes the performance of the developed RS2-Polar codes. We have used MDS property of the RS codes and emulated the decoder accordingly.

Code	Overhead	Rate R	BIAWGN Shannon Limit at R	Pre-FEC BER Uncoded Eb/No	Post-FEC BER Coded Eb/No	Gap to Shannon Limit dB
3.3.1	28%	0.78	1.874	3.08e-2 2.423 dB	1e-15 3.496 dB	1.622
3.3.2	28%	0.78	1.874	2.8e-02 2.615 dB	1e-15 3.687 dB	1.811
3.3.4	24%	0.80	2.099	1.986e-02 3.253 dB	1e-15 4.187 dB	2.088
3.3.3	24%	0.80	2.099	2.294e-02 2.995 dB	1e-15 3.787 dB	1.638
3.3.5	20%	0.83	2.362	1.828e-02 3.395 dB	1e-15 4.187 dB	1.825
3.3.6	15%	0.8695	2.775	1.537e-02 3.680 dB	1e-15 4.287 dB	1.512

Table 3.3: Performance of developed RS2-Polar codes.

Figure 3.7 shows the first configuration of RS2-Polar code. Inner code is Polar(512,416) with SC List 4 decoding and outer decoder employs six iterations. This code operates 1.622 dB away from the Shannon limit.

Figure 3.8 shows the second configuration of RS2-Polar code. Inner code is Polar(2048,1664) with SC decoding and outer decoder employs six iterations. This code operates 1.811 dB away from the Shannon limit.

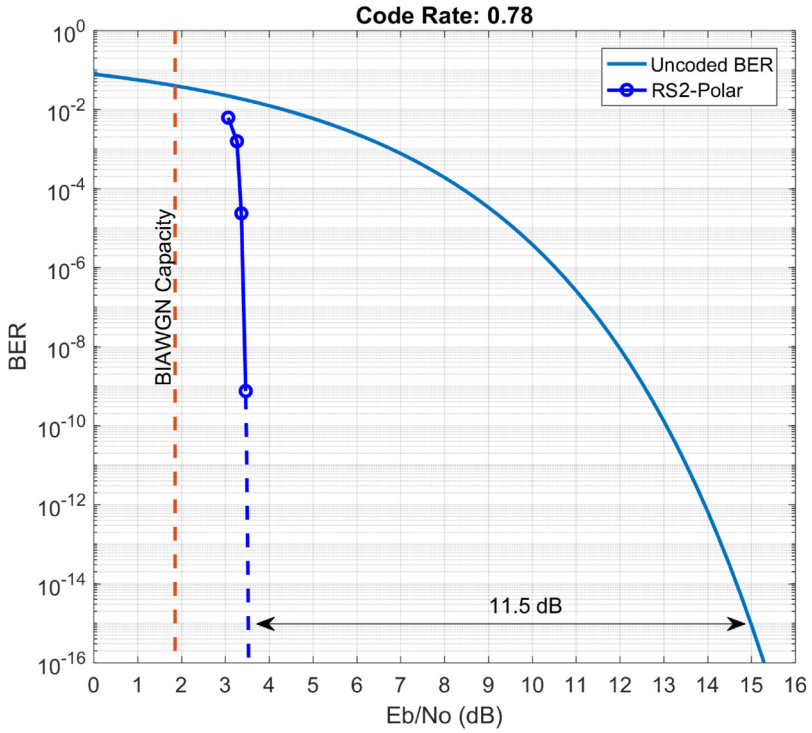

Figure 3.7: BER performance of rate 0.78 (28% OH) RS2-Polar code under SC List 4 decoding and 6 iterations on AWGN channel and BPSK modulation.

Figure 3.9 shows the third configuration of RS2-Polar code. Inner code is Polar(1024,882) with SC decoding and outer decoder employs six iterations. This code operates 2.088 dB away from the Shannon limit.

Figure 3.10 shows the fourth configuration of RS2-Polar code. Inner code is Polar(2048,1704) with SC decoding and outer decoder employs six iterations. This code operates 1.638 dB away from the Shannon limit.

Figure 3.11 shows the fifth configuration of RS2-Polar code. Inner code is Polar(2048,1776) with SC decoding and outer decoder employs six iterations. This code operates 1.825 dB away from the Shannon limit.

Figure 3.12 shows the sixth configuration of RS2-Polar code. Inner code is Polar(2048,1832) with SC decoding and outer decoder employs six iterations. This

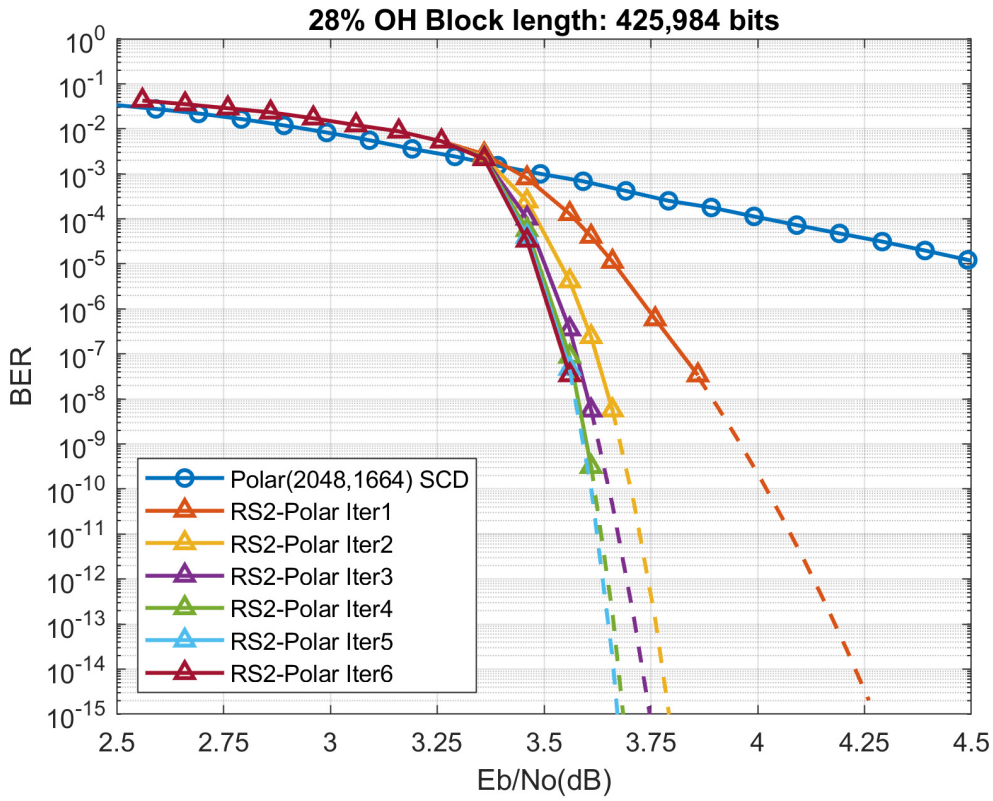

Figure 3.8: BER performance of rate 0.78 (28% OH) RS2-Polar code under SC decoding and 6 iterations on AWGN channel and BPSK modulation.

code operates 1.512 dB away from the Shannon limit.

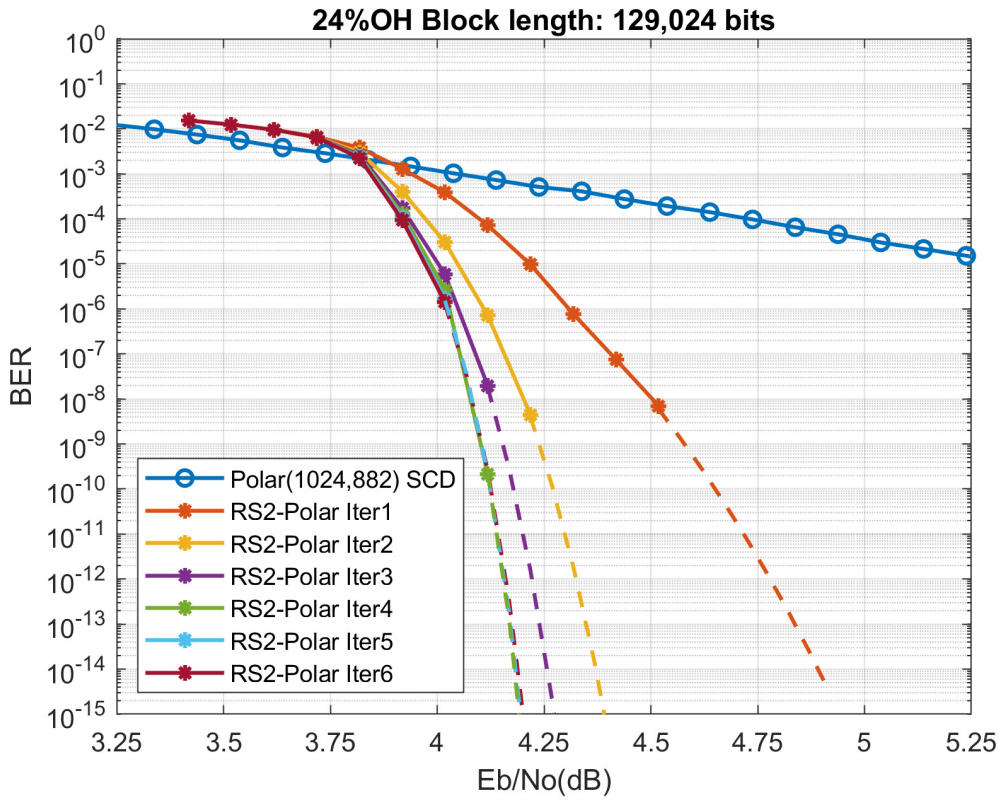

Figure 3.9: BER performance of rate 0.80 (24% OH) RS2-Polar code under SC decoding and 6 iterations on AWGN channel and BPSK modulation.

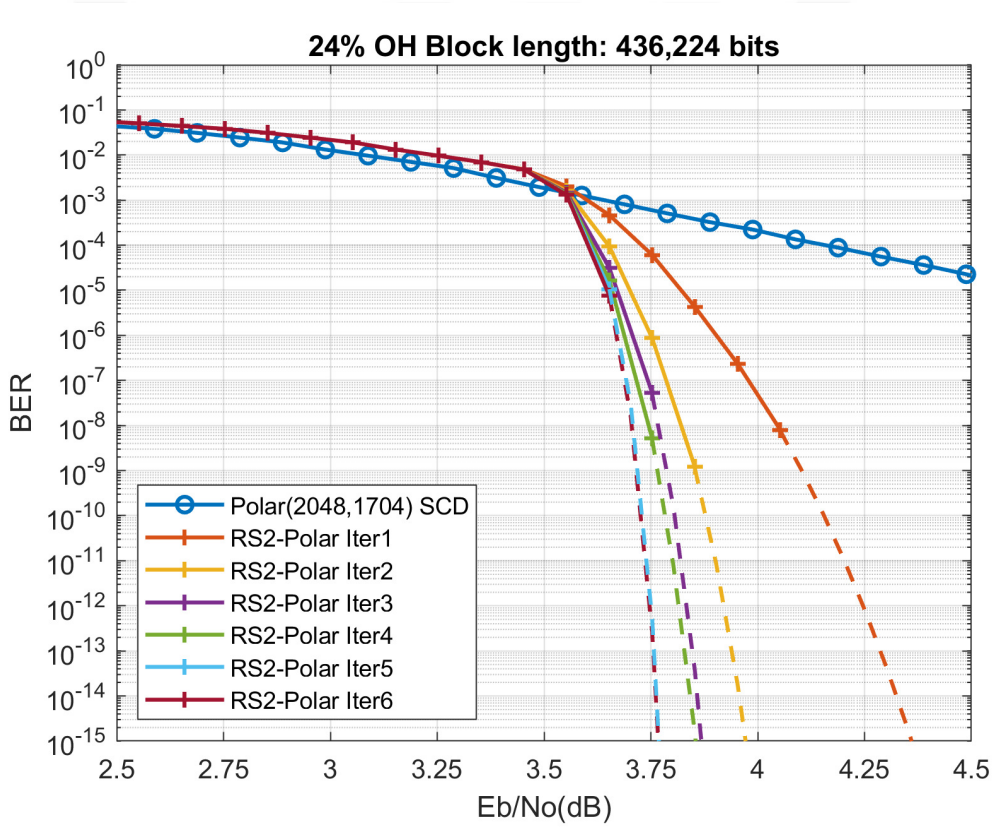

Figure 3.10: BER performance of rate 0.80 (24% OH) RS2-Polar code on AWGN channel and BPSK modulation.

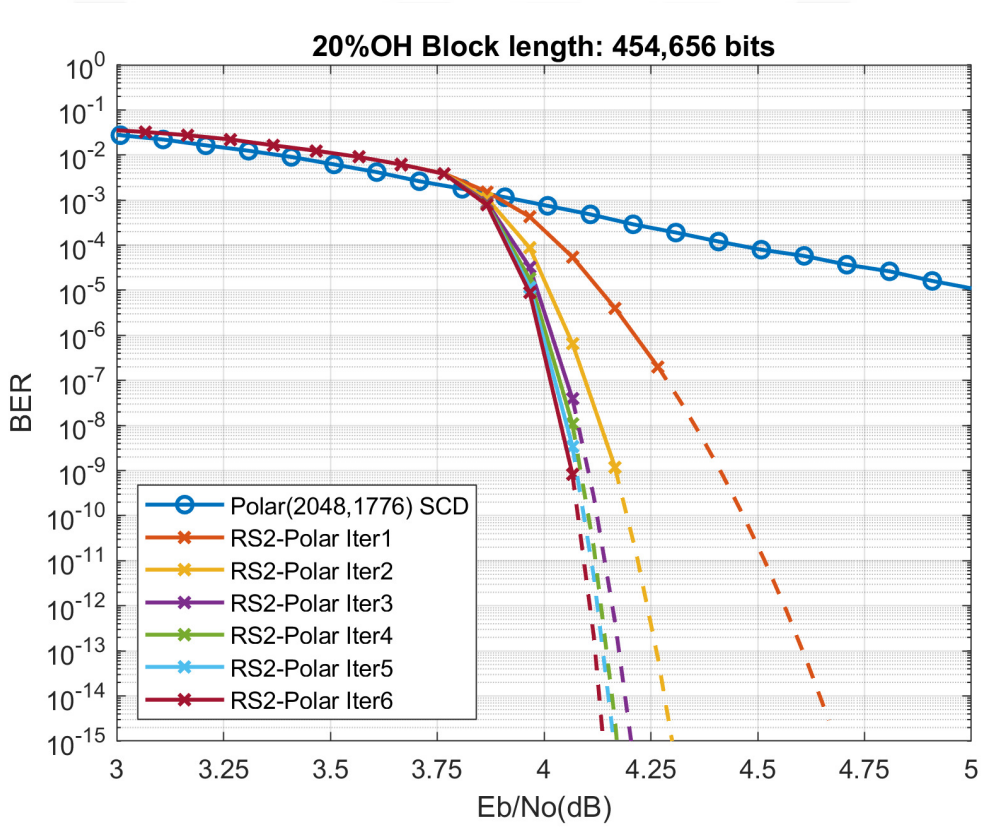

Figure 3.11: BER performance of rate 0.83 (20% OH) RS2-Polar code on AWGN channel and BPSK modulation.

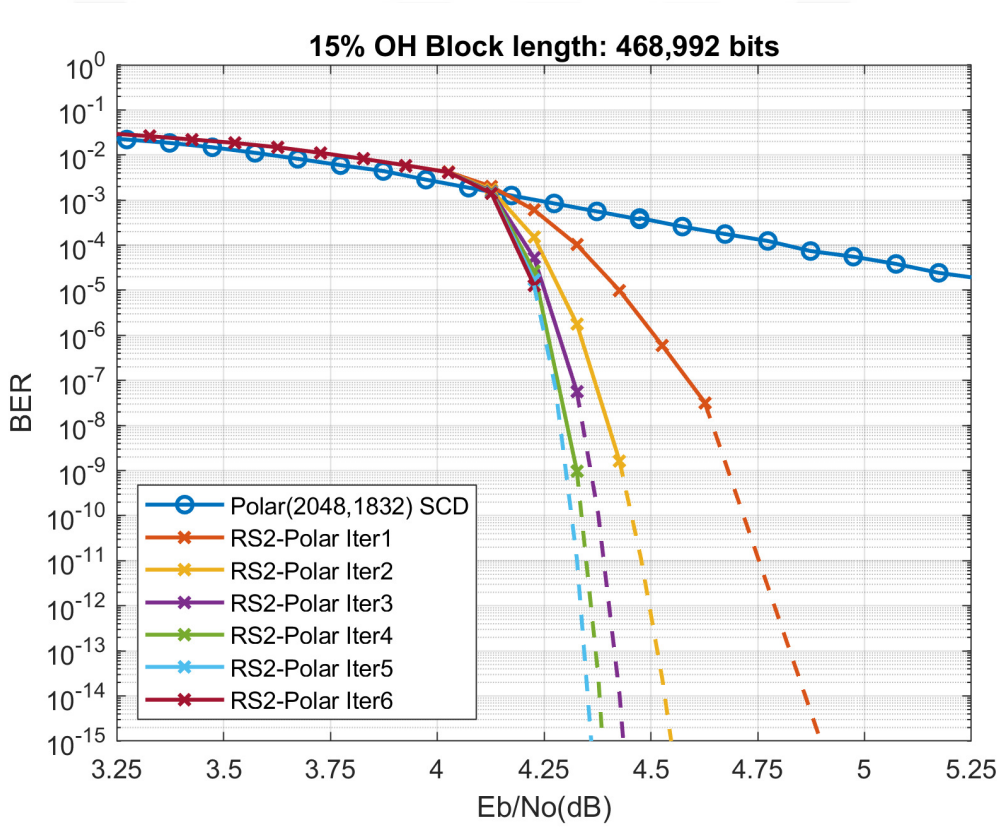

Figure 3.12: BER performance of rate 0.87 (15% OH) RS2-Polar code on AWGN channel and BPSK modulation.

Simulation results show that there is a diminishing return in the number of iterations where performance of 5 and 6 iterations may be negligible in some applications.

Gap to Shannon limit of the developed codes are higher than the codes in the literature since we have sacrificed from code complexity to satisfy the throughput requirements. Performance of the developed code could be increased by using soft information in a more efficient but costly way. First, SC decoder could be replaced with SCL decoder or more complex decoders. This will reduce the BER further at the expense of higher decoding complexity. Second, soft information collected from channel output could be reused by the polar decoder. By sending the information of corrected bits to inner code, polar decoder could override previous bit estimates and generate a better estimate of the transmitted codeword. However, second method is more complex since it requires storage of the LLRs and extra hardware to recalculate an estimate. In this thesis, we have used inner and outer codes as building blocks without introducing any modifications to satisfy the targets of the thesis.

3.6 Complexity Analysis

Complexity of the designed codes is analyzed in this section. Polar and RS decoders are the two components of the developed algorithm. Algorithms have been selected along with corresponding architectural templates considering EPIC project targets. We will assess the complexity of the polar decoders by counting the number of operations to decode the codeword. We will consider simplifications on the GF multipliers in RS decoders. In [61], authors show that polar coding is feasible at Tb/s throughput. In this thesis, we will consider similar block lengths and corresponding architectural templates to support Tb/s throughput. RS codes on GF(2^8) become favorable for two reasons (i) in [62], authors showed the simplified version of multiplier architectures for GF(2^{4n}), (ii) code rate of the product code could be selected in the desired range for $t = 2$ RS codes. Complexity of the RS codes with low t is dominated by syndrome calculation

and Chien search. Chien search [63] could be skipped for $t = 2$ which reduces the complexity of the decoder significantly. We refer [64] for details of the complexity of the high-speed RS decoders.

3.6.1 Hardware Complexity

Hardware complexity of the decoder is analyzed in this subsection. We have estimated the area of unrolled and pipelined polar decoder using the methodology described in Section 2.1.7.1.1. We have synthesized Polar(2048,1664) SC decoder in Genus and estimated 4.36 mm^2 area from the netlist. Table 3.4 shows the area estimates of the SC decoders by using the methodology described in Section 2.1.7.1.1.

Code (N,K)	# of Steps	Area(mm^2)
(2048,1704)	268	4.05
(2048,1664)	277	4.07
(2048,1776)	214	3.39
(2048,1832)	187	3.13

Table 3.4: Area estimates of various polar SC decoders.

Figure 3.13 shows the hierarchical floor-planning of the RS2-Polar decoder with Polar(2048,1664) inner code. We have synthesized the interleaver block in Cadence and estimated 3.27 mm^2 area from the netlist. We refer to [64] for the detailed analysis of high-speed product RS codes where area occupation of the product RS decoder is estimated as 6.63 mm^2. Total area of the decoder is 14.27 mm^2 in 28nm technology node. We have designed a core area with 1×2 ratio (2.73 mm$\times$5.234 mm). Input and output pins are placed to the top and bottom of the chip, respectively. Number of input pins are 10240 where 5 bits of LLR precision is used. Number of output pins are 3264. Area scaling methods are used to estimate the area occupation of the decoder circuitry in 16nm. By using Dennard's scaling law [36], area of the decoder is estimated as 4.65 mm^2 in 16nm.

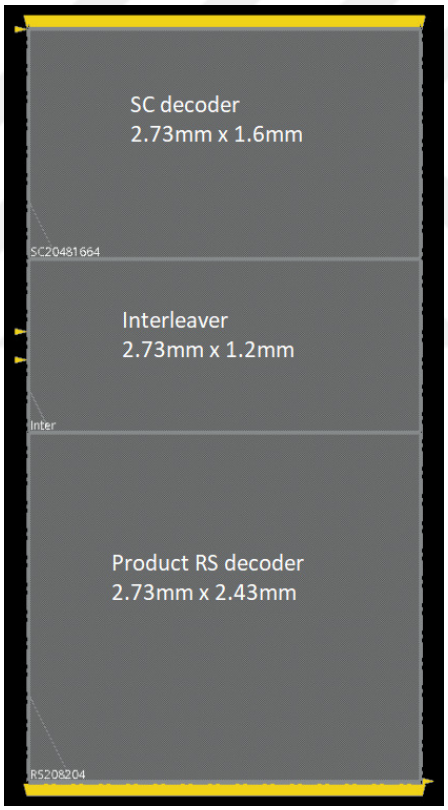

Figure 3.13: Hierarchical floor-planning of RS2-Polar decoder.

Required clock frequency $f_c = 626$ MHz could be calculated using Equation 3.4 where W is the bus-width, γ is the throughput, Q is the LLR precision, R is the code rate.

$$f_c = \frac{\gamma}{W}\frac{Q}{R} \tag{3.4}$$

In advanced technology nodes (7/16/28nm), clock rates up to 1 GHz are identified as feasible in [65]. Therefore, by sacrificing from W, we kept f_c within the feasible range. Moreover, in EPIC project, 1024 block-length polar decoder is synthesized successfully with 1.2 GHz clock frequency to support 1 Tb/s throughput. Interleaver block could keep pace with polar decoder since it is much simpler than the polar decoder. Simplicity of the $t = 2$ component RS decoders allow product RS decoder to operate in the desired clock speed as well [64].

3.7 Comparison with Other Codes

Developed FEC decoder is compared with other decoders in terms of decoding complexity and hardware complexity as much as possible. In [60], authors interleaved polar and RS codes where the polar block protects the first symbol of each RS codeword and so on. In this configuration, polar decoders work in parallel. After each polar decoder decodes a symbol, RS decoding takes place. After corrections are done by RS codes, polar decoders continue to decode the next symbol. Reusing soft information at the polar decoder is the major advantage of this algorithm. There are two drawbacks of this design; (i) throughput requirements may not be satisfied since unrolled and pipelined polar decoder architecture is not suitable for interlacing RS codes and (ii) high storage requirements due to reusing soft information.

In [66] and [67], author shows that thousand bits block-length LDPC codes with 3-4 iterations are feasible to support 1 Tb/s throughput. In [24], proposed LDPC-BCH code is decoded with 32 iterations and block length equals to 38400 bits. It will not be practical to implement 32 iterations at 1 Tb/s throughput due to hardware complexity limits.

In [27], authors developed a coding scheme that targets 1 Tb/s throughput. Decoder complexity is relatively simpler than the RS2-Polar decoder. Area occupied is in the same order as the RS2-Polar decoder. Utilizing a hard decision decoder limits the communications performance which could be a drawback of the algorithm. Communications performance of this algorithm falls behind the developed code in this thesis.

In [30], author designed polar codes with similar block-lengths to developed RS2-Polar code. However, hardware complexity of this decoder is not feasible to support the throughput requirements.

In [35], authors developed a turbo product code that can achieve 100 Gbps throughput. The proposed design occupies 4.49 mm^2 area with 1 GHz clock frequency in 28nm technology. However, proposed decoder architecture is not fully pipelined, thus increasing the throughput further may not be feasible due to clock frequency limits.

3.8 Summary of the Chapter

RS2-Polar coding scheme is explained in this chapter. A specific interleaver is developed to align inner and outer codes. Simulations have been carried out to verify the communications performance of the developed coding scheme. We have analyzed five code configurations which exhibit best performance among other candidates. Hardware complexity of the developed decoder is analyzed by projecting achieved results in the literature. Total area of the decoder is estimated as 14.27 mm^2 in 28nm technology. In 16nm technology, area of the decoder is expected to be 4.65 mm^2. Our estimations show that all the targets are satisfied in 16nm technology node. Comparisons with other works in the literature show that only the simplest algorithms could meet the complexity requirements while supporting 1 Tb/s throughput. Although the developed codes do not exhibit the best communication performance, it could support 1 Tb/s throughput under low complexity decoding algorithms.

Chapter 4

Conclusion

A FEC code and architecture is developed that supports 1 Tb/s throughput for optical communications. We have used combinations of simple codes to achieve better performance with limited complexity. Polar codes are used as inner codes and RS codes are used as outer codes in product form. Simulation results show that developed FEC code achieves 11.5 dB net coding gain with 28% overhead. Inner code is systematic Polar(512,416) under SC List 4 decoding. Outer code is a product RS code with RS8(208,204) component codes. An interleaver is developed to align the polar frames into product structure such that each byte of a polar frame is protected by a different component code. Iterations between inner and outer codes are not allowed to ensure a steady dataflow. We have demonstrated that 1 Tb/s throughput is feasible by using the existing codes without doing any modifications. Under these constraints, we have developed a coding scheme which combines best features of two codes.

In Chapter 1, a comprehensive survey of FEC codes is presented that has developed for optical communications. Most of the codes exploit reasonable communications performance while supporting 100 Gb/s or more. Three evaluation metrics are identified as error correction performance, decoding complexity and area occupation. Area occupation and throughput requirements are adopted from

EPIC project targets. We have extended these targets and included communications performance to evaluation metrics. 10 mm^2 area occupation, 1 Tb/s throughput with reasonable net coding gain (> 10.5 dB) at 10^{-15} post-FEC BER are selected as KPIs. Please note that these targets are only for decoder circuitry in advanced technology nodes (7/16/28nm). In addition, polar encoding is quite simple where only a series of XOR operations are involved.

In Chapter 2, a review of FEC codes are presented where polar, RS, product and concatenated codes are included. SC and SCL decoding algorithms are preferred for decoding of polar codes. Extreme throughput requirement has limited the code block-length and decoder complexity. Area occupation of the polar SC decoder is estimated by calculating the area of each component in the gate level. We have compared our area estimate with the codes that are implemented in the literature. We conclude that our estimates are accurate enough to get an idea without performing any synthesis.

In Chapter 3, RS2-Polar concatenation scheme is explained including inner codes, outer codes and interleaver block. In the design stage, we have selected rate of the inner code to be slightly above the channel capacity to collect soft information as much as possible. Table 4.1 summarizes the achieved code performances at 10^{-15} post-FEC BER where the first column denotes the corresponding section number. All the codes satisfy the net coding gain requirement. Number of iterations is set to six for all codes. Although there is a diminishing return in the number of iterations to code performance, at least four iterations are necessary to achieve a high coding gain. Simulation results show that inner code design affects the code performance dramatically where codes on the first and second row have different inner polar codes with the same overhead. On the other hand, code on the first row utilizes a more complex decoder than the code on the second row. Decoder complexity analysis shows that, a simple algorithm is preferred for polar codes and there are further algorithmic simplifications in the RS decoders. Instead of computation dominant architectures, we have preferred memory dominant architectures to support 1 Tb/s throughput.

An OTU frame contains 4×4080 bytes (130560 bits) where 92 bytes are reserved for headers and packet overheads. Remaining 16228 bytes (129824 bits) are allocated for data and FEC. Code on the third row in Table 4.1 will be highly compatible with a single OTU frame. One could sacrifice 0.5 dB coding gain for a fully compatible code design.

Code	Block Length (bits)	Overhead	Rate R	Net Coding Gain dB	Gap to Shannon Limit dB
3.3.1	425,984	28%	0.78	11.5	1.622
3.3.2	425,984	28%	0.78	11.3	1.811
3.3.3	129,024	24%	0.80	10.8	2.088
3.3.4	436,224	24%	0.80	11.25	1.638
3.3.5	454,656	20%	0.83	10.8	1.825
3.3.6	468,992	15%	0.87	10.7	1.512

Table 4.1: Performance results of the developed codes.

Hardware analysis shows that total area of the decoder is 14.27 mm^2 in 28nm technology. In 16nm technology, area of the decoder is expected to be 4.65 mm^2. Our estimations show that all the targets are satisfied in 16nm technology node. We have estimated that designed codes satisfy the coding gain requirement while supporting 1 Tb/s throughput.

We have compared our work with existing results in the literature. Comparisons show that only a few candidates could satisfy throughput requirements with existing architectural templates. We were able to perform direct comparisons with the codes targeting 1 Tb/s throughput and obtained similar decoding complexity but better communications performance.

Bibliography

[1] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, Jul 1948.

[2] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel Coding Rate in the Finite Blocklength Regime," *IEEE Transactions on Information Theory*, vol. 56, May 2010.

[3] M. C. Coskun, G. Durisi, T. Jerkovits, G. Liva, W. Ryan, B. Stein, and F. Steiner, "Efficient Error-Correcting Codes in the Short Blocklength Regime," *Physical Communication*, vol. 34, pp. 66–79, 2019.

[4] E. Agrell, M. Karlsson, A. R. Chraplyvy, D. J. Richardson, P. M. Krummrich, and P. Winzer, "Roadmap of optical communications," *Journal of Optics*, vol. 18, no. 6, 2016.

[5] ITU, *G.709 Interfaces for the optical transport network*. ITU, Nov 2019.

[6] I. C. Society, *IEEE Standard for Ethernet*, 2015.

[7] I. C. Society, *Physical Layers and Management Parameters for 50 Gb/s, 200 Gb/s and 400 Gb/s Operations over Single-Mode Fiber*, Nov 2019.

[8] S. Bjornstad, "Can OTN be replaced by Ethernet?," *22nd Conference on Optical Network Design and Modelling*, 2018.

[9] P. Larsson-Edefors, C. Foughstedt, and K. Cushon, "Implementation Challenges for Energy Efficient Error Correction in Optical Communication Systems," *Advanced Photonic Congress*, 2018.

[10] K. Szczerba, C. Fougstedt, P. Larsson-Edefors, P. Westbergh, A. G. i Amat, L. Svensson, M. Karlsson, A. Larsson, and P. A. Andrekson, "Impact of Forward Error Correction on Energy Consumption of VCSEL-Based Transmitters," *European Conference on Optical Communication (ECOC)*, 2015.

[11] ITU, *G.975 Forward Error Correction for Submarine Systems*, Oct 2000.

[12] C. Kestel, M. Herrmann, and N. Wehn, "When Channel Coding Hits the Implementation Wall," *2018 IEEE 10th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, Dec 2018.

[13] EPIC, "Enabling Practical Wireless Tb/s Communications with Next Generation Channel Coding." H2020, 2020.

[14] ITU, *G.975.1 Forward Error Correction for High Bit-Rate DWDM Submarine Systems*, Feb 2004.

[15] R. G. Gallager, "Low-Density Parity-Check Codes," *IRE Transactions on Information Theory*, vol. 8, pp. 21–28, Jan 1962.

[16] R. C. Bose and D. K. R. Chaudhuri, "On A Class of Error Correcting Binary Group Codes," *Information and Control*, vol. 3, pp. 68–79, 1960.

[17] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, "Staircase Codes: FEC for 100 Gb/s OTN," *Journal of Lightwave Technology*, vol. 30, Jan 2012.

[18] OIF, *100G Forward Error Correction White Paper*, May 2010.

[19] B. P. Smith and F. R. Kschischang, "Future Prospects for FEC in Fiber-Optic Communications," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 16, pp. 1245–1257, Sep 2010.

[20] A. Sheikh, A. G. i Amat, G. Liva, C. Hager, and H. D. Pfister, "On Low-Complexity Decoding of Product Codes for High-Throughput Fiber-Optic Systems," *2018 IEEE 10th International Symposium on Turbo Codes and Iterative Information Processing*, Dec 2018.

[21] A. Sheikhand, A. G. i Amat, and G. Liva, "Iterative Bounded Distance Decoding of Product Codeswith Scaled Reliability," *2018 European Conference on Optical Communication (ECOC)*, Nov 2018.

[22] T. Ahmad, "Polar Codes for Optical Communications," Master's thesis, Bilkent University, May 2016.

[23] G. Tzimpragos, C. Kachris, I. B. Djordjevic, M. Cvijetic, D. Soudris, and I. Tomkos, "A Survey on FEC Codes for 100 G and Beyond Optical Networks," *IEEE Communication Surveys and Tutorials*, vol. 18, first quarter 2016.

[24] K. Sugihara, Y. Miyata, T. Sugihara, K. Kubo, H. Yoshida, W. Matsumoto, and T. Mizuochi, "A Spatially-coupled Type LDPC Code with an NGC of 12 dB for Optical Transmission beyond 100Gb/s," *2013 Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC)*, June 2013.

[25] D. Chang, F. Yu, Z. Xiao, N. Stojanovic, F. N. Hauske, Y. Cai, C. Xie, L. Li, X. Xu, and Q. Xiong, "LDPC convolutional codes using layered decoding algorithm for high speed coherent optical transmission," *OFC/NFOEC*, May 2012.

[26] L. M. Zhang and F. R. Kschischang, "Low-Complexity Soft-Decision Concatenated LDGM-Staircase FEC for High-Bit-Rate Fiber-Optic Communication," *IEEE Journal of Lightwave Technology*, vol. 35, no. 18, pp. 3991–3999, 2017.

[27] C. Foughstedt and P. Larsson-Edefors, "Energy-Efficient High-Throughput VLSI Architectures for Product-Like Codes," *Journal of Lightwave Technology*, vol. 37, Jan 2019.

[28] C. Foughstedt, A. Sheikhand, A. G. i Amat, G. Liva, and P. Larsson-Edefors, "Energy-Efficient Soft-Assisted Product Decoders," *2019 Optical Fiber Communications Conference and Exhibition (OFC)*, Apr 2019.

[29] M. Barakatain and F. R. Kschischang, "Low-Complexity Concatenated LDPC-Staircase codes," *Journal of Lightwave Technology*, vol. 36, Jun 2018.

[30] T. Ahmad and F. Khan, "On the Coding Gain of Polar Codes for High Speed Communications over AWGN Channels," *17th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, Jan 2020.

[31] C. Condo, P. Giard, F. Leduc-Primeau, G. Sarkis, and W. J. Gross, "A 9.52 dB NCG FEC Scheme and 162 b/Cycle Low-Complexity Product Decoder Architecture," *IEEE Transactions on Circuits and Systems-I*, vol. 65, Apr 2018.

[32] Y. Miyata, K. Kubo, K. Sugihara, T. Ichikawa, W. Matsumoto, H. Yoshida, and T. Mizuochi, "Performance Improvement of a Triple-Concatenated FEC by a UEP-BCH Product Code for 100 Gb/s Optical Transport Networks," *18th OptoElectronics and Communications Conference (OECC)*, 2013.

[33] S. I. Han, J. P. Fonseka, and E. M. Dowling, "Constrained Turbo Block Convolutional Codes for 100G and Beyond Optical Transmissions," *IEEE Photonics Technology Letters*, vol. 26, May 2014.

[34] V. Inc., *ECC66100 Series SD-FEC Encoder/Decoder Cores*, 2017.

[35] Y. Wang, J. Lin, and Z. Wang, "A 100 Gbps Turbo Product Code Decoder for Optical Communications," *2019 IEEE 5th International Conference on Computer and Communications*, pp. 721–725, Dec 2019.

[36] R. H. Dennard, F. H. Gaensslen, H.-N. Yu, V. L. Rideout, E. Bassous, and A. L. Leblanc, "Design of Ion-Implanted MOSFET's with Very Small Physical Dimensions," *IEEE Journal of Solid-State Circuits*, vol. 9, Oct 1974.

[37] E. Arikan, "Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels," *IEEE Transactions on Information Theory*, vol. 55, pp. 3051–3073, Jul 2009.

[38] 3GPP, "5G NR Channel Coding and Multiplexing," tech. rep., ETSI, 2018.

[39] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[40] P. Trifonov, "Efficient Design and Decoding of Polar Codes," *IEEE Transactions on Communications*, vol. 60, pp. 3221–3227, Nov 2012.

[41] I. Tal and A. Vardy, "How to Construct Polar Codes," *IEEE Transactions on Information Theory*, vol. 59, pp. 6562–6582, Oct 2013.

[42] E. Arikan, "Systematic Polar Coding," *IEEE Communications Letters*, vol. 15, pp. 860–862, Aug 2011.

[43] W. E. Ryan and S. Lin, *Channel Codes: Classical and Modern*. Cambridge University Press, 2009.

[44] A. Alamdar-Yazdi and F. R. Kschischang, "A Simplified Successive-Cancellation Decoder for Polar Codes," *IEEE Communications Letters*, vol. 15, pp. 1378–1380, Dec 2011.

[45] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, "Fast Polar Decoders: Algorithm and Implementation," *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 946–957, May 2014.

[46] C. Zhang, B. Yuan, and K. K. Parhi, "Reduced-latency SC Polar Decoder Architectures," *2012 IEEE International Conference on Communications*, Jun 2012.

[47] P. Giard, C. Thibeault, and W. J. Gross, *High-Speed Decoders for Polar Codes*. Springer, 2017.

[48] I. Tal and A. Vardy, "List Decoding of Polar Codes," *IEEE Transactions on Information Theory*, vol. 61, pp. 2213–2226, May 2015.

[49] J. B. Anderson and S. Mohan, "Sequential Coding Algorithms: A Survey and Cost Analysis," *IEEE Transactions on Communications*, vol. 32, pp. 169–176, Feb 1984.

[50] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, "LLR-Based Successive Cancellation List Decoding of Polar Codes," *IEEE Transactions on Signal Processing*, vol. 63, pp. 5165–5179, Oct 2015.

[51] S. A. Hashemi, C. Condo, and W. J. Gross, "Fast Simplified Successive-Cancellation List Decoding of Polar Codes," *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, May 2017.

[52] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields," *J. Soc. Indust. Appl. Math.*, vol. 8, Jun 1960.

[53] K.-M. Cheung and R. J. McEliece, "The Undetected Error Probability for Reed-Solomon Codes," *Military Communications Conference*, vol. 1, pp. 163–167, Oct 1988.

[54] T. C. Nolan and W. E. Stark, "A Recursive Method for Calculating Error Probabilities for a Reed-Solomon Codeword," *VTC'98. 48th IEEE Vehicular Technology Conference. Pathway to Global Wireless Revolution*, pp. 1613–1617, May 1998.

[55] P. Elias, "Error-Free Coding," *IRE Transactions Information Theory*, 1954.

[56] N. Abramson, "Cascade Decoding of Cyclic Product Codes," *IEEE Transactions on Communications Technology*, vol. 16, pp. 398–402, June 1968.

[57] S. M. Reddy and J. P. Robinson, "Random Error and Burst Correction by Iterated Codes," *IEEE Transactions on Information Theory*, vol. IT-18, January 1972.

[58] F. Blomqvist, "On Hard-Decision Decoding of Product Codes." https://arxiv.org/abs/2001.04715, 2020.

[59] G. D. Forney, "Concatenated Codes," *MIT Technical Report*, December 1965.

[60] H. Mahdavifar, M. El-Khamy, and I. Kang, "Performance Limits and Practical Decoding of Interleaved Reed-Solomon Polar Concatenated Codes," *IEEE Transactions on Communications*, vol. 62, May 2014.

[61] A. Sural, E. G. Sezer, Y. Ertugrul, O. Arikan, and E. Arikan, "Terabits-per-Second Throughput for Polar Codes," *2019 IEEE 30th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC Workshops)*, Oct 2019.

[62] C. Paar, P. Fleischmann, and P. Roelse, "Efficient Multiplier Architectures for Galois Fields gf(2sup4n)," *IEEE Transactions on Computers*, vol. 47, Feb 1998.

[63] R. Chien, "Cyclic Decoding Procedures for Bose- Chaudhuri-Hocquenghem Codes," *IEEE Transactions on Information Theory*, vol. 10, pp. 357–363, Oct 1964.

[64] E. G. Sezer, "ASIC Implementation Study of Reed-Solomon Codes." private communication, 2020.

[65] EPIC, "D1.2 B5G Wireless Tb/s FEC KPI Requirement and Technology Gap Analysis," tech. rep., H2020, 2018.

[66] N. Wehn, "High Throughput Coding: An Implementation Centric View," oct 2019.

[67] *Channel Coding for Tbit/s Communications: An Implementation Centric View*, Zenodo, oct 2019.

Appendix A

Interleaver

Interleaver block is analyzed in this chapter. Interleaver is designed in such a way that a component RS code protects distinct m bits of distinct polar blocks. This property could be satisfied by using component codes with odd block lengths. We will prove this property for 0^{th} m bit collection of polar block since the rest will be trivial.

Lemma. Let x and y to be two distinct even numbers in $[\alpha, 2\alpha)$ interval. Let $x' = mod(\alpha)$ and $y' = mod(\alpha)$ where α is an odd number. Then x' and y' are distinct odd numbers.

Proof. Use Euclidean division to represent x and y in terms of α, x' and y'. $x = c\alpha + x'$ and $y = c\alpha + y'$ where $x' \neq y'$ since $x \neq y$. Since α is odd, x' and y' must be odd.

Let component code block length equal to 2n+1, (i, j) denote an element of interleaver matrix and T_0 denote the set of locations of 0^{th} m bit collection of each polar block. First m bit collection of the polar blocks will occupy the following indexes.

$$T_0 = \{(0,0), (1,2), (2,4), ..., (2n-2, 4n-4), (2n-1, 4n-2), (2n, 4n)\} \quad \text{(A.1)}$$

More precisely mod(2n+1) will be applied to each element of T_0 to find the

location in $(2n+1) \times (2n+1)$ matrix. Here index i of each element remain the same since all of them are less than 2n+1. Index j becomes the following.

$$T_0 = \{(0,0),(1,2),(2,4),...,(2n-2,2n-5),(2n-1,2n-3),(2n,2n-1)\} \quad \text{(A.2)}$$

One can observe that index j of the each element in T_0 is a distinct number in $[0, 2n-1)$ interval since in A.1 index j's less than 2n+1 are all even numbers and index j's greater than 2n+1 will become distinct odd numbers since Lemma. Therefore, 0^{th} m bit collection of each polar block will be on different rows and columns. This property could be generalized to other elements in the interleaver matrix.

For component codes with even block lengths this property could not be satisfied in vertical axis since mod(2n) of even numbers are also even numbers.