Universidad Nacional de Córdoba
Facultad de Ciencias Exactas, Físicas y Naturales

Tesis Doctoral

Codificación para Corrección de Errores con Aplicación en Sistemas de Transmisión y Almacenamiento de Información

Autor: Damián Alfonso Morero

Director: Dr. Ing. Mario Rafael Hueda

Diciembre de 2013

Codificación para Corrección de Errores con Aplicación en Sistemas de Transmisión y Almacenamiento de Información

por

Ing. Damián Alfonso Morero

Dr. Ing. Mario Rafael Hueda
Director

Comisión Asesora:

Dr. Ing. Victor Hugo Sauchelli
FCEFyN - UNC

Dr. Inga. Elizabeth Vera de Payer
FCEFyN - UNC

Esta Tesis fue enviada a la Facultad de Ciencias Exactas Físicas y Naturales de la Universidad Nacional de Córdoba para cumplimentar los requerimientos de obtención del grado académico de Doctor en Ciencias de la Ingeniería.

Córdoba, Argentina
Diciembre de 2013

UNIVERSIDAD NACIONAL DE CORDOBA
Facultad de Cs. Exactas, Fisicas y Naturales

ACTA DE EXAMENES

Libro: 00001 Acta: 02216 Hoja 01/01
LLAMADO: 1 11/12/2013
CATEDRA - MESA:

DI002 TESIS DOCTORADO EN CIENCIAS DE LA INGENIERIA

NUMERO	APELLIDO Y NOMBRE	DOCUMENTO	INGRESO	COND.	NOTA	FIRMA
29238027	MORERO, Damián Alfonso	DNI: 29238027	2006	T	APROBADO	

CASTIÑEIRA, Jorge - GALARZA, cECILIA - PODESTA, Ricardo - CORRAL BRIONES, gRACIELA - FINOCCHIET

Observaciones:

Córdoba, ____/_____/_____-.
Certifico que la/s firma/s que ha/n sido puesta/s en la presente Acta pertenece/n a:_____

1

Inscriptos	Ausentes	Examinados	Reprobados (0-3)	Aprobados (4-10)
09/12/2013 10:27:48				

Libro/Acta: 0000102216 Hoja: 01/ 01

A Mis Padres y Abuelos.

Agradecimientos

Este trabajo y la experiencia adquirida durante su desarrollo no hubieran sido posible sin la ayuda, motivación y confianza que me brindaron todas aquellas personas que me rodearon en este proceso. Soy deudor de todos ellos.

A mi Familia.

A mis Amigos.

A Mario Hueda y Oscar Agazzi quienes me ayudaron y guiaron en todo el proceso de mis estudios de doctorado.

A mis compañeros de trabajo en ClariPhy Argentina S.A.

A mis compañeros de estudio en el Laboratorio de Comunicaciones Digitales de la FCEFyN, UNC.

A aquellos que trabajan y comparten su conocimiento para enriquecernos a todos.

Resumen

El diseño de potentes *códigos de corrección de errores* (CCE) constituye uno de los desafíos más importantes para los futuros sistemas de comunicaciones y almacenamiento de información. La creciente demanda de mayores velocidades de transmisión y densidades de almacenamiento, hace necesario diseñar códigos con un desempeño próximo al límite teórico del canal (más conocido como *Límite de Shannon*), y con una tasa de error de bit muy pequeña (menor a 10^{-15}, esto es, un bit con error cada 10^{15} bits transmitidos!). Esta tarea resulta extremadamente difícil por dos razones: (*i*) la existencia de un fenómeno conocido como *piso de error*, y (*ii*) la elevada complejidad para la implementación en circuitos integrados. Esta Tesis Doctoral propone nuevos esquemas de codificación concatenados y no concatenados que permiten no sólo alcanzar las ganancias de códigos requeridas por las aplicaciones mencionadas, sino también reducir la complejidad de implementación de los mismos para hacer viable su incorporación en circuitos integrados para aplicaciones comerciales.

En esta Tesis se presenta una nueva técnica de diseño de códigos de chequeo de paridad de baja densidad (más conocido por sus siglas en inglés como LDPC) y un nuevo algoritmo de post-procesamiento para la reducción del piso de error. Uno de los aspectos más destacados de esta contribución es que permite la construcción de CCE *no-concatenados* con bajo piso de error y alta ganancia de codificación como los requeridos en los futuros sistemas de comunicaciones de alta velocidad por fibra óptica. Además, los códigos diseñados con esta nueva técnica poseen una estructura tal que permite simplificar sustancialmente la arquitectura del decodificador para su implementación en circuitos integrados de muy alta escala de integración. Como ejemplo de una aplicación práctica de la teoría aquí expuesta, se realiza el diseño de un código con una ganancia neta de codificación de 11.3 dB a una tasa de error de 10^{-15}. La ganancia de este código se verifica experimentalmente mediante emulación en *hardware* y constituye uno de los primeros esquemas de CCE no concatenados con este desempeño reportado en la literatura. Se destaca que el código aquí diseñado ha sido adoptado para su implementación en un transceptor comercial de 200Gb/s para comunicaciones por fibra óptica.

Como una segunda contribución fundamental de esta Tesis, se introduce un nuevo CCE *concatenado* serie de baja complejidad y alta eficiencia para combatir el problema del piso de error. Esta nueva técnica de concatenación es general y puede aplicarse con numerosos tipos de códigos (LDPC o turbo códigos) para obtener un mejor desempeño y reducir significativamente la complejidad de implementación. Además, el nuevo concepto de múltiple concatenación en serie que se propone en la Tesis permite el uso de numerosos códigos de corrección de errores de baja complejidad que hasta el momento no podían adoptarse en aplicaciones de alta velocidad debido al elevado piso de error que presentan. La técnica de concatenación en serie desarrollada en esta Tesis Doctoral constituye un nuevo paradigma general para combatir el problema de piso de error de manera eficiente.

Abstract

The design of powerful *error correction codes* (ECC) is one of the main challenges for future communication and storage systems. The growing demand for higher communication speeds and storage densities requires the design of codes with a performance close to the theoretical channel limit (better known as *Shannon Limit*) and a very low bit error rate (lower than 10^{-15}, i.e., one bit in error in 10^{15} transmitted bits!). This task is extremely difficult to achieve due to two problems: (*i*) the existence of a phenomenon known as *error-floor*, and (*ii*) the high implementation complexity in integrated circuits. This Ph.D. Thesis proposes new concatenated and non-concatenated coding schemes which not only can achieve the required coding gains for the above mentioned applications, but also can help reducing the implementation complexity, making feasible their use in commercial integrated circuits.

This Thesis introduces a new design technique for low-density parity-check codes (LDPC) and a new post-processing algorithm for error-floor reduction. One of the main aspects of this contribution is that it allows to create *non-concatenated* ECCs with low error-floor and high coding gain as it is required for the next generation of high-speed optical communication systems. Furthermore, the structure of the codes designed with this technique significantly simplifies the decoder implementation architecture for integrated circuits with high integration density. A code with a net coding gain of 11.3 dB at a bit error rate of 10^{-15} is designed as a practical application example. This gain is experimentally verified through *hardware* emulation and it represents one of the first reported non-concatenated ECC schemes with this performance. It is worth noting that the code designed in this Thesis has been selected to be implemented in a commercial fiber-optic transceiver for 200 Gb/s data rate.

As second main contribution of this Thesis, a new low-complexity serial concatenated ECC scheme with high-efficiency in reducing the error-floor problem is described. The new concatenation technique is general and it can be used with many different codes (such as LDPC and turbo codes) in order to improve performance and significantly reduce implementation complexity. Furthermore, the new concept of multiple serial concatenation allows the use of several low complexity error correction codes which were previously discarded for high-speed applications due to their high error-floor. The serial concatenation technique developed in this Ph.D. Thesis represents a new general paradigm to efficiently reduce the error-floor problem.

Zusammenfassung

Die Konstruktion von leistungsfähigen Vorwärtsfehlerkorrekturcodes (FEC) stellt eine der wichtigsten Herausforderungen für zukünftige Kommunikationssysteme und Datenspeicherung dar. Der zunehmende Bedarf an höheren Datenraten und Datendichten erfordert die Konstruktion von Kanalcodes, die nahe an der Grenze der theoretischen Kanalkapazität arbeiten können (auch bekannt als Shannon-Grenze), sowie über eine sehr niedrige Bitfehlerrate verfügen (geringer als 10^{-15}, d.h. ein Fehlerbit pro 10^{15} übertragene Bits!). Diese Aufgabe erweist sich aus zwei Gründen als besonders schwierig: (*i*) die Existenz eines Phänomens bekannt als Fehlerplateau und (*ii*) die hohe Komplexität für die Implementierung in integrierte Schaltungen. Diese Dissertation schlägt neue Schemata für verkettete und nicht verkettete Kanalcodierung vor, die nicht nur die erforderlichen Codegewinne für die erwähnten Anwendungen ermöglichen, sondern auch die Komplexität der Implementierung derselben reduzieren und damit die Voraussetzung für die Einbindung in integrierte Schaltungen für gewerbliche Anwendungen schaffen.

In der vorliegenden Dissertation wird eine neue Konstruktionstechnik für dünn besetzte Kanalcodes mit Paritätsprüfungen vorgestellt (bekannter durch ihr englisches Akronym LDPC) sowie ein neuer Algorithmus der Nach-Verarbeitung zur Absenkung des Fehlerplateaus. Einer der herausragendsten Aspekte von diesem Beitrag ist, dass die Konstruktion von nicht verketteten FEC mit niedrigem Fehlerplateau und hohem Codegewinn ermöglicht wird, welche für zukünftige Kommunikationssysteme über Lichtwellenleiter mit hoher Übertragungsgeschwindigkeit erforderlich werden. Des Weiteren verfügen die mit dieser neuen Technik entwickelten Kanalcodes über eine Struktur, die die Architektur der Dekodierung für ihre Implementierung in integrierte Schaltungen mit sehr hoher Integrationsskala wesentlich vereinfacht. Als Beispiel für eine praktische Anwendung der hier aufgeführten Theorie wird die Konstruktion eines Kanalcodes mit einem Netto-Codegewinn von 11.3 dB bei einer Fehlerrate von 10^{-15} vorgestellt. Dieser Codegewinn wird experimentell über Emulation in der Hardware überprüft und stellt eines der ersten Schemata von nicht verketteten FEC mit dieser Leistung dar, die in der Literatur erwähnt werden. Es wird betont, dass der hier konstruierte Kanalcode für die Implementierung in einem gewerblichen 200 Gb/s Transceiver für die Kommunikation über Lichtwellenleiter eingesetzt wird.

Als zweiten grundlegenden Beitrag dieser Dissertation wird eine neue verkettete FEC-Serie mit niedriger Komplexität und hoher Effizienz für die Eindämmung des Problems Fehlerplateau vorgestellt. Diese neue Technik der Verkettung ist umfassend und kann mit zahlreichen Arten von Kanalcodes (LDPC oder Turbo-Codes) für eine bessere Leistung und für eine bedeutende Reduzierung der Implentierungskomplexität eingesetzt werden. Außerdem ermöglicht dieses in der Dissertation vorgestellte, neue Konzept der multiplen Verkettungsserie die Verwendung von zahlreichen Fehlerkorrekturcodes mit niedriger Komplexität, die bis dato aufgrund ihres hohen Fehlerplateaus nicht in Hochgeschwindigkeits-Anwendungen eingesetzt werden konnten. Damit gründet die in dieser Dissertation entwickelte Technik der Verkettungsserie ein neues Gesamtparadigma für die effiziente Eindämmung des Problems Fehlerplateau.

Publicaciones en Conferencias con Referato:

2012 Damián A. Morero and Mario R. Hueda, "Efficient Concatenated Coding Schemes for Error Floor Reduction of LDPC and Turbo Product Codes", Global Communications Conference (Globecom), IEEE, Dec. 2012

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6503469

2012 Mario A. Castrillon, Damián A. Morero, and Mario R. Hueda, "A New Cycle Slip Compensation Technique for Ultra High Speed Coherent Optical Communications", Photonics Conference, IEEE, Sep. 2012

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6358547

2012 Fernando Gutierrez, Graciela Corral-Briones and Damián Morero, "*FPGA Implementation of the Parity Check Node for Min-Sum LDPC Decoders*", VIII Southern Programmable Logic Conference (SPL 2012), IEEE, March 2012

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6211802

2011 Damián A. Morero, M. Alejandro Castrillon, Facundo A. Ramos, Teodoro A. Goette, Oscar E. Agazzi, and Mario R. Hueda, "*Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks*", Global Communications Conference (Globecom), IEEE, Dec. 2011

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6133616

2011 Juan Manuel López Simao, Damián Morero and Cecilia Galarza, "*Simulador para códigos QC-LDPC no binarios*", RPIC 2011

2010 Damián A. Morero and Mario R. Hueda, "*Performance of Euclidean-Metric MLSD Receiver in the Presence of Channel Mismatch Caused by Nongaussian Noise*", International Conference on Communications, IEEE, May 2010

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5502001

2009 Martín I. del Barco, Gabriel N. Maggio, Damián A. Morero, Javier Fernandez, Facundo Ramos, Hugo S. Carrer, and Mario R. Hueda, "*FPGA Implementation of High-Speed Parallel Maximum a Posteriori (MAP) Decoders*", Argentine School of Micro-Nanoelectronics, Technology and Applications, IEEE, Sept 2009

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=5288890

2008 Damián Morero, Graciela Corral Briones and Mario Hueda, "*Parallel architecture for decoding LDPC codes on high speed communication systems*", Argentine School of Micro-Nanoelectronics, Technology and Applications, IEEE, Sept 2008

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4638987

Publicaciones en Revistas con Referato:

2013 Damián A. Morero and Mario R. Hueda, "Novel Serial Code Concatenation Strategies for Error Floor Mitigation of Low-Density Parity-Check and Turbo Product Codes", Canadian Journal of Electrical and Computer Engineering, Vol. 36, No. 2, IEEE, Spring 2013

Disponible en IEEEXPLORE:
ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6601080

Publicaciones sin Referato:

2012 Mario A. Castrillon, Damián A. Morero, and Mario R. Hueda, "Joint Demapping and Decoding for DQPSK Optical Coherent Receivers", arXiv, Jun. 2012

Disponible en arXiv:
arxiv.org/abs/1206.4914

2011 Damián A. Morero, Graciela Corral-Briones, Carmen Rodriguez, Mario R. Hueda, "*High-Rate Short-Block LDPC Codes for Iterative Decoding with Applications to High-Density Magnetic Recording Channels*", arXiv, Apr 2011

Disponible en arXiv:
arxiv.org/abs/1104.1457

Patentes:

2012 Damián Alfonso Morero et al., "Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks", U.S. Patent Application No. 13/406,452, February 27, 2012

Disponible en Google Patent:
www.google.com/patents/US20120221914

Lista de Acronismos:

AS	Absorbing Set Conjunto Absorbente
ASIC	Application-Specific Integrated Circuit Circuito Integrado para Aplicaciones Específicas
AWGN	Additive White Gaussian Noise Ruido Gaussiano Blanco Aditivo
BCH	Bose Chaudhuri Hocquenghem
BCJR	Bahl Cullum Frazer Jelinek
BEC	Binary Erasure Channel Canal Binario con Borrado
BER	Bit Error Rate Tasa de Error de Bits
BPSK	Binary Phase-shift keying Desplazamiento de Fase Binario
BSC	Binary Symmetric Channel Canal Binario Simétrico
CC	Convolutional Codes Códigos Convolucionales
CCE	Códigos de Corrección de Errores
CNPU	Check Node Processing Unit Unidad de Procesamiento del Nodo de Chequeo
FPGA	Field Programmable Gate Array Arreglo de Compuertas Programable
IS	Importance Sampling Muestreo por Importancia
LDPC	Low-Density Parity Check Baja Densidad de Chequeos de Paridad
LLR	Log Likelihood-Ratio Logaritmo de la Relación de Verosimilitud
MAP	Maximum A posteriori Probability Máxima Probabilidad a Posteriori
MIMO	Multiple-Input Multiple-Output Múltiple Entrada Múltiple Salida
MISO	Multiple-Input Single-Output Múltiple Entrada Simple Salida
MSA	Min-Sum Algorithm Algoritmo Min-Sum
NCG	Net Coding Gain Ganancia Neta de Codificación
OTN	Optical Transport Network Red de Transporte Óptico
PPA	Post-Processing Algorithm Algoritmo de Post-Procesamiento

RCP	Regular Column Partition Partición Regular por Columnas
RS	Reed Solomon
SCC	Serial Code Concatenation Concatenación en Serie de Códigos
SNR	Signal to Noise Ratio Relación Señal Ruido
SPA	Sum-Product Algorithm Algoritmo Suma-Producto
SPC	Single Parity Check Chequeo de Paridad Simple
TCM	Trellis Coded Modulation Modulación con Códificación Reticulada
TPC	Turbo Product Code Código Producto Turbo
TS	Trapping Set Conjunto de Captura
VNPU	Variable Node Processing Unit Unidad de Procesamiento del Nodo Variable

Lista de Abreviaturas:

i.e.	id est; esto es; es decir
et. al.	et alii; y otros
doc.	documento

Índice General

1. Introducción **1**
- 1.1. Reseña Histórica . . . 2
- 1.2. Códigos LDPC . . . 9
- 1.3. Objetivos y Resultados . . . 12
- 1.4. Organización . . . 13

2. El Algoritmo Suma-Producto y los Códigos LDPC **15**
- 2.1. El Algoritmo Suma-Producto . . . 16
- 2.2. Los Códigos LDPC . . . 25
- 2.3. Decodificación de Códigos LDPC Binarios . . . 27
- 2.4. Estimación del Desempeño . . . 34
- 2.5. Efecto de la Dependencia Entre Mensajes . . . 36
- 2.6. El Problema del Piso de Error . . . 41
- 2.7. Conclusión . . . 48

3. Código LDPC con Bajo Piso de Error **51**
- 3.1. Introducción . . . 52
- 3.2. Diseño de la Matriz de Paridad . . . 54
- 3.3. Desempeño . . . 60
- 3.4. Mejoras sobre el Algoritmo de Decodificación . . . 62
- 3.5. Conclusión . . . 66

4. Arquitectura de Implementación **69**
- 4.1. Introducción . . . 70
- 4.2. Circuitos Digitales Síncronos . . . 70
- 4.3. Nueva Arquitectura . . . 76
- 4.4. Implementación . . . 83
- 4.5. Conclusión . . . 83

5. Esquema de Concatenación de Códigos **85**
- 5.1. Introducción . . . 86
- 5.2. Contexto . . . 87
- 5.3. Nueva Estrategia de Concatenación . . . 91

5.4. Reducción del Piso de Error de Códigos TPC 100
5.5. Conclusión 105

6. Conclusiones **107**
6.1. Discusión Final 108
6.2. Caminos a Seguir 109

7. Publicaciones y Patentes **111**
7.1. Publicaciones y Patentes 112

Bibliografía **197**

Capítulo 1

Introducción

No desprecies el recuerdo del camino recorrido. Ello no retrasa vuestra carrera, sino que la dirige; el que olvida el punto de partida pierde fácilmente la meta.

Pablo VI (1897-1978).

Resumen

En el presente capítulo se introduce al lector en el contexto de los códigos de corrección de errores en el cual se encuadra la presente Tesis Doctoral. A tal efecto, se incluye una cronología de los avances producidos hasta la fecha en esta área. Dicha cronología permite entender tanto el estado actual del conocimiento científico sobre el tema en cuestión como así también sus tendencias. Entre estas últimas se encuentran los códigos de chequeo de paridad de baja densidad como los más importantes. En relación a dichos códigos, se describen los principales problemas que actualmente se encuentran abiertos, siendo el objetivo de esta Tesis aportar una solución a parte de ellos.

1.1. Reseña Histórica

En todo proceso de transmisión o almacenamiento de información existe siempre la posibilidad de que la información sea alterada. La única forma de atacar este problema es mediante la incorporación de información redundante que permita corregir los errores en la información recibida. Esta técnica se conoce con el nombre de *códigos de corrección de errores* (CCE). El uso de códigos de corrección de errores es muy común, a tal punto de que los humanos ya hacemos uso de los mismos en forma cotidiana y sin darnos cuenta. Por ejemplo, intente "descubrir" el mensaje en el siguiente texto "*oirntras lee egto, umjwd ewtz zorrisendo errodeg*"[1]. La razón por la que usted puede discernir el mensaje en la oración anterior se debe a que hay redundancia de información tanto a nivel de las palabras individuales como a nivel de la oración completa, i.e., no todo conjunto de caracteres forma una palabra válida y no todo conjunto de palabras forma una oración coherente.

A pesar del uso tan cotidiano de los CCE, la teoría detrás de los mismos es reciente y surge al intentar resolver el siguiente problema: ¿cómo enviar un mensaje a través de un canal ruidoso sin cometer errores? En base a este problema podríamos dividir el pensamiento científico en dos períodos a saber: el periodo anterior y el periodo posterior al trabajo de *Shannon* publicado en 1948. Antes de 1948, se tenía la idea de que la probabilidad de error se podía hacer tender a cero únicamente si la velocidad de transmisión también lo hacía [22, 110] (aumentando indefinidamente la redundancia). Un ejemplo ilustrativo del pensamiento de aquella época sería el siguiente:

> Suponga un canal binario simétrico [80] con probabilidad de error p y una velocidad de transmisión igual a ν bits por segundo. Una de las primeras técnicas usadas para reducir la probabilidad de error (actualmente conocida como código de repetición) es transmitir n veces cada bit de información y luego, en el receptor, decidir si se transmitió un 1 o un 0 como aquel que se recibió más veces. Está técnica reduce la probabilidad de error a $\sum_{i=r+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}$, la cual tiende a cero cuando n tiende a infinito. Sin embargo, como la velocidad de información es ν/n, esta última también tiende a cero.

Intentar generalizar la conclusión del ejemplo anterior a cualquier sistema de comunicación constituyó un error que fue esclarecido por Shannon en 1948 [139]. En particular, Shannon demostró que era teóricamente posible transmitir información por un canal sin cometer errores siempre y cuando la velocidad de transmisión sea menor a una cantidad finita que denominó *capacidad del canal*. Shannon no dio detalles de cómo implementar dicho sistema de transmisión. Sin embargo, dejó en claro que el mecanismo por el cual se lograría tal objetivo era una adecuada codificación de la información. Desde entonces, comenzó una muy ferviente investigación con la finalidad de

[1] Los errores en cada carácter fueron generados aleatoriamente con una probabilidad de 0.2.

diseñar e implementar técnicas de codificación para la corrección de errores que permitan alcanzar la máxima capacidad del canal.

A continuación se lista una reseña histórica de las principales contribuciones en el área de los códigos de corrección de errores[2]:

1948: **Teorema de codificación sobre un canal ruidoso de Shannon**[3] [139]. Shannon demuestra que para todo canal de comunicaciones existe una cota C denominada *capacidad del canal* tal que es teóricamente posible transmitir información con una probabilidad de error ϵ arbitrariamente pequeña siempre y cuando la velocidad de transmisión R sea menor a C.

1949: **Código de Golay** [58]. Se distinguen cuatro códigos, a saber: el código binario perfecto de golay con parámetros $[23, 12, 7]_2$, la versión extendida del mismo con parámetros $[24, 12, 8]_2$ y los códigos ternarios $[12, 6, 6]_3$ y $[12, 6, 5]_3$.

1950: **Códigos de Hamming** [67]. Los códigos de Hamming conforman una familia de códigos lineales perfectos que pueden corregir un error [103]. Junto a los códigos de Golay, los códigos de Hamming conforman los únicos códigos perfectos no triviales sobre un alfabeto de cardinal que sea potencia de un número primo.

1954: **Códigos de Reed-Muller** [131] [123]. Los códigos de Reed-Muller conforman una familia infinita de códigos binarios lineales que permiten su decodificación por lógica mayoritaria.

1954: **Códigos producto de Elias** [46]. Son códigos lineales de parámetros $[n_1 n_2, k_1 k_2, d_1 d_2]$ construidos a partir de la combinación de dos códigos de parámetros $[n_1, k_1, d_1]$ y $[n_2, k_2, d_2]$.

1955: **Códigos convolucionales (CC) de Elias** [47]. Son códigos lineales continuos (no en bloques) que pueden ser generados a partir de la secuencia de datos por una máquina de estados de memoria finita.

1957: **Códigos cíclicos de Prange** [128]. Son una familia de códigos de bloques lineales que pueden ser representados como ideales en el anillo $F_q[x]/(x^n - 1)$ donde n es la longitud del código, F_q el cuerpo finito de q elementos y $F_q[x]$ son los polinomios con coeficientes en F_q. Dicha

[2] Para más detalles sobre algunos temas de la historia de los códigos de corrección de errores se recomienda leer [36] y las referencia allí citadas

[3] Otro teorema publicado por Shannon en la misma fecha y de gran importancia para las comunicaciones (pero no estrictamente para los códigos de corrección de errores) es el *teorema de la codificación de la fuente*, el cual establece que el número de bits requeridos en promedio para representar los resultados de una fuente de datos aleatoria es igual a su entropía (detallando cómo se calcula esta última).

estructura simplifica significativamente el proceso de codificación y decodificación.

1957/69: **Decodificadores secuenciales de Wozencraft, Fano, Zigangirov y Jelinek**. Son decodificadores sub-óptimos originalmente propuestos para códigos convolucionales. Entre las principales variantes de decodificadores secuenciales se puede mencionar a: 1) el algoritmo de Wozencraft propuesto en 1957 [174], 2) el algoritmo de Fano propuesto en 1963 [52] y 3) el algoritmo "Stack" propuesto independientemente por Zigangirov en 1966 [189] y por Jelinek en 1969 [75]. Dichos algoritmos tienen un desempeño inferior al del algoritmo de Viterbi [166] pero la ventaja de ser menos complejos.

1958: **Códigos basados en residuos cuadráticos de Prange** [129]. Son una familia de códigos cíclicos cuyo polinomio generador es construido a partir de residuos cuadráticos.

1959/61: **Códigos de Bose, Ray-Chaudhuri and Hocquenghem (BCH)** [19] [70]. Son una familia de códigos lineales cíclicos con una gran flexibilidad en sus parámetros de diseño. Originalmente propuestos en su versión binaria en [19] [70] fueron luego generalizados en [63] por Gorenstein y Zierler para cualquier cuerpo finito.

1960: **Códigos de Reed-Solomon (RS)** [132]. Son una familia de códigos lineales cíclicos no binarios que cumplen la cota de *Singlenton* y por lo tanto son códigos con máxima distancia de separación. Los códigos RS y BCH están fuertemente relacionados, pudiéndose derivar uno a partir del otro. Dichos códigos fueron en realidad propuestos originalmente por Bush en 1952 [21] en el contexto de arreglos ortogonales pero su trabajo no tuvo impacto en el área de las comunicaciones.

1962: **Códigos LDPC de Gallager** [56]. Son una familia de códigos lineales definidos a partir de una matriz de paridad rala (i.e., con un baja densidad de elementos no nulos). Una de las principales ventajas de estos códigos es la existencia de un algoritmo eficiente para su decodificación, el cual es una aplicación particular del algoritmo *Suma-Producto* (SPA) [91].

1965: **Códigos concatenados de Forney** [54]. Son una combinación de códigos (en su versión original de dos códigos) con la finalidad de lograr una familia de esquemas de corrección de errores cuya complejidad crece polinómicamente con la longitud del código resultante y la probabilidad de error decrece exponencialmente.

1967: **El algoritmo de Viterbi** [166]. El algoritmo de Viterbi resuelve en forma eficiente el problema de estimar por máxima verosimilitud la

secuencia de estados por los que atraviesa un sistema dinámico de tiempo discreto con un número finito de estados al ser observado en presencia de ruido sin memoria (o ruido blanco). Como observó Omura [124], dicho algoritmo es una aplicación particular de *programación dinámica* [15] al problema mencionado.

1968/69: **El algoritmo de Berlekamp y Massey** [50] [113]. Es un algoritmo para calcular el polinomio del menor registro de desplazamiento con retroalimentación lineal que genera una secuencia de salida determinada. Dicho algoritmo proporciona un método eficiente para decodificar los códigos BCH y RS, siendo ésta la aplicación original propuesta por Berlekamp.

1970/72: **Códigos de Goppa Clásicos** [59] [60] [61]. Son una familia infinita de códigos lineales construidos a partir de polinomios. En general no son cíclicos. Forman parte de los llamados *códigos alternantes* los cuales son una generalización de los códigos BCH. Dichos códigos son los primeros capaces de alcanzar la cota de *Gilbert-Varshamov*.

1972: **Algoritmo BCJR de Bahl, Cocke, Jelinek y Raviv** [13]. Es un algoritmo originalmente propuesto para calcular la *probabilidad a posteriori* de símbolos generados por una fuente de Markov y transmitidos por un canal discreto sin memoria y cuya aplicación directa fue la decodificación de códigos convolucionales. Sin embargo, su rango de aplicación es mayor, pudiéndose utilizar para calcular la *probabilidad a posteriori* en cualquier cadena de Markov con un número finito de estados.

1975: **Algoritmo de Sugiyama et al.** [149]. Es una aplicación del algoritmo generalizado de Euclides que permite resolver en forma eficiente el mismo problema que el algoritmo de Berlekamp y Massey y por consiguiente puede ser usado para decodificar códigos BCH, códigos RS y códigos de Goppa clásicos.

1977: **Códigos multinivel de Imai e Hirakawa** [74]. Es un esquema de codificación que optimiza la distancia euclidiana asociada a una determinada modulación en lugar de optimizar la distancia de Hamming entre palabras código.

1981: **Códigos basados en grafos de Tanner** [151]. Son una generalización de los códigos LDPC y los códigos concatenados que se construyen en base a una representación gráfica de la matriz de paridad en donde las ecuaciones de chequeo pueden generalizarse a un sub-código.

1982: **Códigos de Goppa geométricos** [62]. Son una generalización de los códigos de Goppa clásicos construidos a partir de conceptos de

geometría algebraica. Están basados en evaluar funciones racionales de una curva en puntos racionales de dicha curva. Conforman la primer familia de códigos capaces de superar la cota de *Gilbert-Varshamov* [18] y alcanzar la cota de Tsfasman, Vladut y Zink (TVZ) [160].

1982: **Códigos TCM de Ungerboeck** [162][4]. Los códigos TCM combinan la codificación y la modulación para maximizar la mínima distancia euclidiana de entre dos secuencias posibles de símbolos transmitidos en lugar de la distancia de Hamming del código (en este sentido están relacionados a los códigos multinivel).

1993: **Códigos Turbo de Berrou, Glavieux y Thitimajshima** [17]. Constituyen el primer esquema de codificación que mostró ser capaz de alcanzar un desempeño cercano al límite de Shannon con una complejidad de implementación moderada.

1996/97: **Redescubrimiento de los códigos LDPC y de los códigos basados en grafos** por parte de Spielman [142] [143] [146], Wiberg [170] [171] [172], MacKay y Neal [109]. Mackay mostró que códigos LDPC de longitud n moderada podrían alcanzar un desempeño cercano a la capacidad de Shannon mientras que Spielman mostró que dichos código alcanzan el límite de Shannon cuando $n \to \infty$ con una complejidad lineal en n. Wiberg mostró que tanto los códigos LDPC como los código Turbo podían interpretarse como casos particulares de códigos sobre grafos cuyo algoritmo de decodificación es una aplicación particular del algoritmo *Suma-Producto* [91] [171].

1998/99 **Códigos Espacio-Temporales de Alamouti y Tarokh et al.** [5] [154] [153]. Es una técnica de codificación propuesta originalmente para canales inalámbricos tipo MISO o MIMO que permite obtener ganancia por *diversidad* al transmitir la "misma información" en diferentes instantes de tiempo y por diferentes antenas[5].

1999 **Decodificador por lista de Guruswami y Sudan** [66]. Es un algoritmo de decodificación sobre códigos no perfectos que permite corregir más de $\lfloor (d_{min} - 1)/2 \rfloor$ errores superando en desempeño a los decodificadores clásicos como el de Berlekamp-Massey. Posteriormente, en 2000/03, dicho algoritmo fue extendido por Koetter y Vardy para utilizar decodificación blanda [84] [85].

[4] Los códigos TCM fueron concebidos por Ungerboeck en 1970 [36] y publicados parcialmente en 1976 [161], sin embargo la publicación que le dio fama ocurrió recién en 1982 [162]

[5] Si bien podría considerarse que dicha técnica, en su forma más básica, se encuentra fuera el área de los códigos de corrección de errores, la misma debe ser considerada para poder alcanzar la capacidad de un canal MISO o MIMO y en este sentido se encuentra relacionada al tema que nos compete.

2007 **Códigos Polares de Arikan** [6] [8] [9]. Los códigos polares se pueden considerar como una generalización de los códigos de Reed-Muller [7] [10] que pueden alcanzar la capacidad de los canales binarios simétricos (BSC) y binario con borrado (BEC). Una generalización para un canal general discreto sin memoria puede verse en [138]. El desarrollo de dichos códigos aún se encuentra activo mostrando gran potencial para su aplicación en el corto plazo. Véase también [11] para más información.

1997:2013 **Otros avances relacionados a los códigos LDPC**

1997/98 **Códigos LDPC irregulares de Luby et al.** [106] [107] [108]. Originalmente, los códigos LDPC estaban basados en grafos regulares, i.e., los nodos variables y de chequeo tienen una cantidad de conexiones γ y ρ respectivamente que es fija. En los códigos LDPC irregulares no sólo no existe esta restricción sino que se optimiza la fracción de nodos para cada número de conexiones. Lo anterior permite alcanzar la capacidad de un canal de borrado.

1998 **Códigos LDPC basados en repetición acumulada de Divsalar, McEliece, et al.** [43]. Son códigos que se forman mediante la concatenación en serie de: 1) uno o varios códigos de repetición, 2) un entrelazador (interleaver) y 3) un acumulador con función de transferencia 1/(1+D) . Los mismos mostraron tener muy buen desempeño y ser relativamente fáciles de analizar. Dichos códigos fueron posteriormente generalizados en [1] [2] [79].

1998 **Códigos LDPC no binarios** [39]. Representan la generalización trivial de los códigos LDPC binarios al caso de códigos LDPC sobre un cuerpo no binario. Dichos códigos mostraron en muchos casos tener un desempeño superior al de sus códigos equivalentes en formato binario. La desventaja radica en una mayor complejidad en la decodificación. Sin embargo, avances posteriores permitieron reducir dicha complejidad, reavivando su interés práctico, véase [14] [28] [40] [182].

1998/99 **Códigos LDPC convoluciones de Felstrom y Zigangirov** [48] [49] [77] [78]. Los códigos convolucionales tipo LDPC forman una de las familias más atractivas de códigos en la actualidad que aún están siendo analizados debido a su gran desempeño [152].

2000/01 **Códigos LDPC basados en geometría finita**. En el 2000/01, Kou, Lin y Fossorier en [87] [88] [89] mostraron que es posible construir códigos LDPC con un excelente desempeño utilizando códigos clásicos basados en geometría finita. La ventaja de estos

códigos radica en que la estructura cíclica o cuasi-cíclica que poseen facilita una implementación de baja complejidad. Trabajos posteriores extendieron estos códigos a cuerpos no binarios y a matrices de paridad irregulares. Para más detalles ver [42] [72] [76] [81] [83] [102] [150] [177] [188].

2001 **Density Evolution de Richardson y Urbanke et al.** [134] [135]. Es un algoritmo que permite calcular el máximo desempeño posible que puede tener una determinada familia de códigos LDPC cuando la longitud del código y el número de iteraciones tienden a infinito. Esta técnica permite además optimizar la distribución del número de conexiones en códigos irregulares [134].

2001 **EXIT Chart de Ten Brink** o *gráfica de transferencia de información extrínseca*. Propuesta originalmente en [156] para códigos turbo y luego aplicado en códigos LDPC [157]: es un método que permite analizar y estimar el desempeño de los códigos iterativos. Al igual que density evolution, no es un método exacto ya que ignora la dependencia estadística de los mensajes.

2001/05 **Combinación de códigos basados en grafos con procesamiento digital de señales**. Se intensifica la idea mencionada por Wiberg en 1996 [170] de extender el algoritmo suma-producto para que no sólo realice la decodificación de códigos LDPC sino que al mismo tiempo realice tareas de ecualización [104], detección [173] y recuperación de sincronismo [35], entre otras.

2001/13 **Arquitecturas de implementación**. Se propusieron diversas arquitecturas para la implementación del algoritmo de decodificación donde cada una intenta aprovechar la estructura que posee cada familia de códigos en particular. Véase [168] [118] [119] y las referencias allí citadas.

2002/13 **Análisis de la causa del piso de error** [41] [111] [133] [158]. Para el canal binario con borrado (BEC), el piso de error es causado por sub-estructuras del grafo conocidas como *stopping set* [41] [126]. Por otro lado, en canales AWGN el piso de error está dominado por sub-estructuras llamadas *near-codewords* [111], *trapping sets* [133] y *absorbing sets* [45]. Además, en este último canal, también se observó que el piso de error es exacerbado por el uso de aritmética de precisión finita en la implementación del algoritmo de decodificación [118] [180] [181] [184] [186] [187]

2007/13 **Técnicas de reducción del piso de error**. Variadas técnicas han sido propuestas para atacar el problema del piso de error en los códigos LDPC. Entre ellas se pueden mencionar: (*i*)

la modificación del algoritmo de decodificación [165], (*ii*) el agregado de algoritmos de post-procesamiento [118] [185] y (*iii*) el agregado de un esquema de codificación externo (adicional al código LDPC) [30] [82] [121].

2010 **Códigos acoplados** [92] [93]. Es un método de construcción de códigos que permite acercar el desempeño del algoritmo suma-producto (SPA) al desempeño óptimo por máxima probabilidad *a posteriori* (MAP). Se puede interpretar como una generalización del mecanismo que opera en los códigos LDPC convolucionales y que le brindan a estos últimos su alto desempeño [69] [93] [94]. Actualmente conforman un área de mucho interés para la investigación.

Como se observa en la reseña histórica previa, a partir de la publicación del teorema de la capacidad del canal de Shannon, se inició una carrera científica por la búsqueda de códigos de corrección de errores que logren alcanzar dicha capacidad y también de algoritmos eficientes para la implementación de los mismos. En esta carrera, los canales binarios simétricos y los canales con ruido aditivo Gaussiano blanco (AWGN) fueron los que dominaron la escena debido a su simplicidad y generalidad. Muchos códigos fueron desarrollados desde entonces, siendo los códigos basados en grafos y en particular los códigos LDPC los que han dominado la escena en los últimos años.

1.2. Códigos LDPC

En relación a los códigos LDPC podemos distinguir dos años importantes, a saber: en 1962 se publica la tesis de Gallager proponiendo los códigos LDPC y en 1996 se redescubren por Spielman [142] [143] [146], Wiberg [170] [171], MacKay y Neal [109]. En el período comprendido entre 1962 y 1996 fueron muy pocas las referencias a los códigos LDPC y el interés en los mismos. En particular se puede citar el trabajo de Tanner [151] sobre códigos en grafos. Por el contrario, en el período comprendido entre su redescubrimiento en 1996 hasta la actualidad se ha desarrollado una extensiva investigación en torno a los mismos. Además, el excelente desempeño de los códigos LDPC los ha convertido en la mejor opción disponible para aplicaciones que requieran lograr un desempeño cercano al límite de Shannon. La Tabla 1.1 muestra las principales aplicaciones estandarizadas que actualmente utilizan códigos LDPC.

El éxito de los códigos LDPC se debe en gran parte a la existencia de un algoritmo eficiente para su decodificación. Este algoritmo es una aplicación particular de un algoritmo más general denominado algoritmo *Suma-Producto* (SPA) [3] [91] [127], el cual permite realizar inferencias en *Redes*

Año	Estándar	Aplicación / Documento
2005	DVB-S2	Comunicación satelital de alta velocidad. Doc.: ETSI EN 302 307.
2006	IEEE 802.3 (10 GBASE-T)	Redes cableadas de área local y metropolitana. Doc.: IEEE 802.3an-2006, IEEE 802.3-2008.
2006	CMMB	Radiodifusión móvil. Doc.: GY/T 220.1-2006.
2006	DTMB (DMB-T/H)	Radiodifusión digital terrestre. Doc.: GB 20600-2006.
2007	IEEE 802.11ad (WiGig)	Redes de área local y metropolitana. Doc.: IEEE P802.11ad.
2008	GMR-1	Telefonía satelital. Doc.: ETSI TS 101 376-5-3 V2.3.1 (GMPRS-1), ETSI TS 101 376-5-3 V3.1.1 (GMR-3G).
2009	IEEE 802.11 (WiFi)	Redes inalámbricas de área local y metropolitana. Doc.: IEEE 802.11n-2009, IEEE 802.11-2012.
2009	DVB-T2	Radiodifusión de televisión digital terrestre. Doc.: ETSI EN 302 755.
2009	IEEE 802.15.3c (60 GHz PHY)	Redes inalámbricas de área local y metropolitana. Doc.: IEEE 802.15.3c-2009.
2009	IEEE 802.16 (Mobile WiMAX)	Redes de área local y metropolitana. Doc.: IEEE 802.16e-2005, IEEE 802.16-2009.
2009	ITU-T G.hn (G.9960)	Redes alámbricas hogareñas por cables coaxiales, telefónicos y de energía. Doc.: ITU-T G.9960.
2009	WiMedia 1.5 UWB	Redes inalámbricas de área personal. Doc.: WiMedia PHY Specification 1.5.
2010	DVB-C2	Televisión digital por cable. Doc.: ETSI EN 302 769.
2011	IEEE 802.22 (WRAN)	Redes inalámbricas de área regional. Doc.: IEEE 802.22-2011.
2011	CCSDS	Comunicación cercana a Tierra y de espacio profundo. Doc.: CCSDS 131.1-O-2, CCSDS 131.0-B-2.
2012	DVB-T2-Lite	Televisión digital móvil. Doc.: ETSI EN 302 755, V1.3.1..
2012	DVB-NGH	Radiodifusión de televisión terrestre. Doc.: ETSI EN 303 105.
2013	IEEE 802.11ac (WiFi)	Redes de área local y metropolitana. Doc.: IEEE P802.11ac.

Cuadro 1.1: Aplicaciones estandarizadas de los códigos LDPC

Bayesianas y *Campos Aleatorios de Markov*. Además de su aplicación en el área de *Códigos de Corrección de Errores*, dicho algoritmo fue descubierto independientemente en otras áreas tales como *Inteligencia Artificial* e *Inferencia Probabilística*, en donde es conocido bajo los nombres en Inglés de *Belief Propagation* (BP) o *Message Passing* (MP). Una amplia variedad de algoritmos desarrollados en inteligencia artificial, procesamiento de señales y comunicaciones digitales se pueden obtener como casos particulares del SPA. En particular, el detector de Viterbi [166], el algoritmo BCJR [13], el filtro de Kalman [64] [155], la decodificación turbo o iterativa [17], y la transformada rápida de Fourier [34] pueden interpretarse como instancias particulares del algoritmo SPA.

1.2.1. Problemas Actuales Asociados a los Códigos LDPC

A pesar del excelente desempeño de los códigos LDPC, los mismos poseen problemas fundamentales que actualmente reducen su utilidad:

P1 Presentan una pérdida de desempeño que sólo es observable a tasas de errores muy bajas (típicamente menores a 10^{-10}). Este problema es denominado *piso de error*.

P2 No hay una arquitectura de implementación estándar que haya mostrado ser eficiente en general (como ocurre por ejemplo con el algoritmo de Berlekamp Massey para los códigos BCH y RS o el algoritmo de Viterbi para los códigos convolucionales).

P3 No hay métodos analíticos certeros para determinar el desempeño a una tasa de error muy baja[6] (por ejemplo 10^{-15}).

Los mencionados problemas no son propios de los códigos LDPC cuando estos últimos son entendidos como entidades algebraicas, es decir, como un sub-espacio particular de dimensión k sobre F_q^n, donde F_q^n es el espacio vectorial de dimensión n sobre el cuerpo de Galois de q elementos. Dichos problemas tienen lugar sólo cuando los códigos LDPC se decodifican mediante el algoritmo SPA (i.e., es a partir del algoritmo SPA de donde se derivan tanto las ventajas como las desventajas de los códigos LDPC). A pesar del excelente desempeño del algoritmo SPA, éste se desvía del desempeño óptimo de una forma que es difícil de analizar. Si existiera un algoritmo de decodificación óptimo para dichos códigos, el problema del piso de error y el de la estimación de desempeño se reducirían al cálculo de la distancia mínima (o en forma más precisa, al cálculo de la función de distribución de pesos). Desafortunadamente, no se conoce tal algoritmo y la posibilidad de su existencia parece muy remota ya que constituiría, para el

[6] Esta situación se ve agravada por el hecho de que la simulación de dichos códigos con las computadoras actuales no permiten medir BER menores a 10^{-10}

caso general, un problema perteneciente a NPC (véase [114] [115] y [167]). El camino más prometedor a seguir es entonces mejorar el desempeño del algoritmo SPA cuando se utiliza como decodificador para los códigos LDPC. Precisamente éste es el camino que recorre la presente Tesis Doctoral.

1.3. Objetivos y Resultados

Esta Tesis se enfoca en los problemas P1 y P2 previamente mencionados, esto es: (*i*) brindar una solución al piso de error y (*ii*) proponer una arquitectura de implementación para el algoritmo de decodificación. Para el problema del piso de error se proponen dos soluciones. La primera solución está basada en el diseño de la matriz de paridad y en un nuevo algoritmo de post-procesamiento que se añade al algoritmo de decodificación [118]. La segunda solución está basada en un nuevo método de concatenación de códigos [121] [122]. Finalmente, en lo referente a la implementación se propone una nueva arquitectura de implementación semi-paralela [118] [119].

1.3.1. Código LDPC con Bajo Piso de Error

Con la finalidad de reducir el piso de error en un código LDPC, en el Capítulo 3 se describe el diseño de la matriz de paridad y se propone un nuevo algoritmo de post-procesamiento para usar en combinación con el algoritmo SPA. Como aplicación de las ideas propuestas, se diseña un código LDPC con bajo piso de error, alta ganancia de codificación y una estructura tal que hace posible una arquitectura de implementación eficiente (la cual será analizada en el Capítulo 4). Finalmente, se analiza el desempeño del código propuesto mediante simulación en FPGA y estimación semi-analítica basada en la técnica *importance sampling*. Los aportes descriptos en el presente capítulo se encuentran en parte publicados en los artículos [119] y [121] y en la patente [51].

1.3.2. Arquitectura de Implementación

Se propone una arquitectura de implementación para el decodificador LDPC que evita los problemas de: (*i*) alta densidad de interconexiones y (*ii*) gran cantidad de memoria, que son típicos en aplicaciones de alta velocidad donde se requiere el uso de procesamiento en paralelo [44] [112]. La arquitectura propuesta toma ventaja de la estructura introducida en la matriz de paridad (ver Capítulo 3) con la finalidad de evitar ciclos inactivos e información en espera. Además, la misma posee un *overhead* programable que la hace atractiva para aplicaciones que requieran una codificación que se adapte a las condiciones del canal. Finalmente, la arquitectura propuesta es sintetizada en tecnología CMOS de 28mn en donde se determinó la velocidad de operación, la potencia de disipación y el área. Tal resultado representa

el primer código LDPC para comunicaciones ópticas cuya baja complejidad de implementación fue demostrada y cuyo desempeño verificado mediante emulación en FPGA, logrando una ganancia de codificación neta igual a 11.3 dB a un BER de 1e-15 sin piso de error.

1.3.3. Esquema de Concatenación de Códigos

El nuevo método de concatenación de códigos será descripto en el Capítulo 5. Los aportes relacionados a este nuevo método se encuentran en parte publicados en los artículos [121] y [122]. El mismo consiste en una nueva estrategia de codificación y decodificación basada en la combinación de dos códigos externos en adición al código LDPC interno. La misma reduce significativamente la complejidad de implementación en comparación con las técnicas existentes, llegando fácilmente a una reducción de complejidad de un orden de magnitud. Adicionalmente, en el Capítulo 5 se describe una generalización del esquema propuesto que permite reducir el piso de error causado por palabras código de bajo peso. Esto último permite combatir el piso de error no sólo en códigos LDPC sino también en códigos turbo como el TPC. Este esquema es aplicado a un sistema de codificación basado en un código TPC de baja complejidad que posee un piso de error a un BER de 10^{-7}, pero que combinado con el esquema propuesto alcanza una ganancia neta de $\sim$ 11,2 dB a un BER= 10^{-15} con un *overhead* total de $\sim$ 22 % y un piso de error a $\sim 7 \cdot 10^{-17}$. Tal esquema supera en 0,4 dB a los códigos propuestos en [4] [117] y [125].

1.4. Organización

La organización de la presente Tesis es la siguiente. El Capítulo 2 brinda una introducción a la decodificación de códigos LDPC basada en el algoritmo SPA y describe en detalle los problemas asociados al mismo y cuya solución son el objetivo de esta Tesis. El Capítulo 3 describe el diseño de la matriz de paridad y el nuevo esquema de post-procesamiento para reducir el piso de error. El Capítulo 4 describe la arquitectura de implementación propuesta. El Capítulo 5 describe el nuevo esquema de concatenación de códigos propuesto para reducir el piso de error. Finalmente el Capítulo 6 presenta las principales conclusiones.

Capítulo 2

El Algoritmo Suma-Producto y los Códigos LDPC

Las grandes ideas son aquellas de las que lo único que nos sorprende es que no se nos hayan ocurrido antes.

Noel Clarasó.

Resumen

En el presente capítulo se brinda una introducción al algoritmo Suma-Producto (SPA) y se describe su utilización como método aproximado para la decodificación por máxima probabilidad *a posteriori* (MAP) de los códigos LDPC binarios. Se analiza el problema de estimar el desempeño de dicho algoritmo y se describe el problema del piso de error.

2.1. El Algoritmo Suma-Producto

El algoritmo suma-producto tiene como objetivo calcular en forma eficiente las funciones marginales asociadas a una función multivariable global; donde eficiente se refiere a reducir la cantidad de operaciones aritméticas básicas como sumas, productos y evaluación de funciones. A tal efecto, dicho algoritmo hace uso de una generalización de la *propiedad distributiva* entre las operaciones suma y producto sobre un *semi-anillo*[1] — recordar que la propiedad distributiva nos dice que $a(b+c) = ab+ac$ donde a, b, c son elementos del semi-anillo — El hecho de que se trabaje sobre un semi-anillo le brinda al algoritmo SPA una generalidad muy grande permitiendo su uso en diversas aplicaciones. La idea básica es la siguiente, para calcular $ab + ac$ se necesita realizar dos multiplicaciones y una suma mientras que sacando factor común a obtenemos $a(b + c)$ y es posible realizar el mismo cálculo con sólo una multiplicación y una suma. Esta idea elemental puede generalizarse a cálculos más complejos trayendo aparejado en la mayoría de los casos una reducción de gran importancia en la complejidad de cálculo. Siguiendo la metodología usada en [91], veremos a continuación ejemplos progresivamente más complejos que nos iluminarán el camino hacia un algoritmo general[2]. Para más detalles véase [3] [91] [110].

Sea $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ una colección de variables tales que para cada i, x_i toma valores en el dominio o alfabeto A_i (usualmente finito) y sea $g(x_1, x_2, \ldots, x_n)$ una función a valores reales. El dominio de g es $S = A_1 \times A_2 \times \cdots \times A_n$ donde $\times$ denota el producto cartesiano de conjuntos. Asociada a cada función $g(x_1, x_2, \ldots, x_n)$ hay n funciones marginales $g_i(x_i)$. Para cada $a \in A_i$, el valor de $g_i(a)$ se obtiene de la suma de los valores de $g(x_1, \ldots, x_n)$ sobre todas las configuraciones de las variables que tienen

[1] Un *semi-anillo* $(A, +, \cdot)$ es una estructura algebraica similar a un *anillo* pero que no requiere la existencia de un elemento inverso para la suma. Esto es, un semi-anillo es un conjunto A sobre cuyos elementos existen dos operaciones binarias $+$ y $\cdot$ tales que $(A, +)$ es un *monoide conmutativo*; es decir que se cumple la propiedad asociativa $(a + b) + c = a + (b + c)$, la propiedad conmutativa $a + b = b + a$, y la existencia de un elemento neutro 0 tal que $0 + a = a + 0 = a$ para todos $a, b, c \in A$, por otro lado, $(A, \cdot)$ es también un *monoide* pero no necesariamente conmutativo y se cumple la ley distributiva $a \cdot (b + c) = a \cdot b + a \cdot c$ y $(b + c) \cdot a = b \cdot a + c \cdot a$ y la propiedad de elemento absorbente $0 \cdot a = a \cdot 0 = 0$.

[2] Vale la pena remarcar que muchos algoritmos conocidos por tener una alta eficiencia computacional se derivan en el fondo de la misma idea básica que acabamos de comentar, y por consiguiente son casos particulares del algoritmo SPA (es decir de un buen uso de la propiedad asociativa). Entre éstos podemos mencionar: el algoritmo de Baum-Welch, la transformada rápida de Fourier, el algoritmo de Viterbi, el algoritmo BCJR, el filtro de Kalman, el algoritmo de Shafer-Shenoy, la decodificación Turbo y algoritmos de propagación de mensajes en redes bayesianas tales como el algoritmo de propagación en árboles de Pearl [3] [91].

$x_i = a$; i.e.,

$$g_i(x_i) = \sum_{\mathcal{X}\setminus\{x_i\}} g(x_1, \ldots, x_n) = \sum_{\substack{x_j \in A_j \\ j \neq i}} g(x_1, \ldots, x_n). \tag{2.1}$$

donde $\mathcal{X}\setminus\{x\} = \{y \in \mathcal{X} \ : \ y \neq x\}$.

Calcular $g_i(x_i)$ en forma eficiente no es un problema trivial. En lo sucesivo supondremos que $g(x_1, \ldots, x_n)$ se puede factorizar en un producto de m *funciones locales* f_j. Al conjunto de dichas funciones lo denotaremos $\mathcal{F}$. Cada $f \in \mathcal{F}$ tiene el subconjunto $\mathcal{X}_{f_j} \subseteq \mathcal{X}$ de variables como argumento. En forma similar, al conjunto de funciones locales que tienen a x_i como argumento lo denotaremos $\mathcal{F}_{x_i} \subseteq \mathcal{F}$. Esto es,

$$g(\mathcal{X}) = g(x_1, \ldots, x_n) = \prod_{j=1}^{m} f_j(\mathcal{X}_{f_j}) = \prod_{f \in \mathcal{F}} f(\mathcal{X}_f) \tag{2.2}$$

Es posible aprovechar esta última propiedad para calcular $g_i(x_i)$ de una manera más eficiente que en el cálculo directo de la ecuación (2.1). Por ejemplo, sea $g(x_1, x_2, x_3, x_4, x_5)$ una función de cinco variables, donde x_i con $i = 1, \ldots, 5$ toman todas valores en el alfabeto A de $N = |A|$ elementos. Para calcular $g_1(x_1)$ con $x_1 \in A$, fijo, utilizando la definición de la ecuación (2.1) necesitaríamos evaluar y sumar $g(x_1, \ldots, x_n)$ en las N^4 combinaciones posibles de las variables $(x_2, x_3, x_4, x_5) \in A^4$. Es decir, necesitaríamos realizar $N^4 - 1$ sumas y N^4 evaluaciones de la función global $g(x_1, \ldots, x_n)$, donde cada evaluación de $g(x_1, \ldots, x_n)$ implica realizar m evaluaciones de las funciones locales $f \in \mathcal{F}$ y $m-1$ multiplicaciones. Supongamos a continuación que g se puede expresar como el siguiente producto:

$$g(x_1, \ldots, x_5) = f_A(x_1) f_B(x_2) f_C(x_3) f_D(x_4) f_E(x_5), \tag{2.3}$$

entonces podemos calcular $g_1(x_1)$ de la siguiente forma:

$$g_1(x_1) = f_A(x_1)\Big(\sum_{x_2} f_B(x_2)\Big)\Big(\sum_{x_3} f_C(x_3)\Big)\Big(\sum_{x_4} f_D(x_4)\Big)\Big(\sum_{x_5} f_E(x_5)\Big) \tag{2.4}$$

donde sólo es necesario realizar $4N$ evaluaciones de funciones, $4(N-1)$ sumas y 4 productos. Por lo tanto, gracias a esta factorización particular de g es posible reducir significativamente la complejidad cálculo de $O(N^4)$ a $O(N)$. La pregunta inmediata que surge es si se puede realizar algo similar aprovechando cualquier estructura de factorización de g. Afortunadamente la respuesta es afirmativa, aunque la reducción de complejidad dependerá de la estructura de factorización de cada g en particular.

Supongamos ahora que g se puede factorizar como

$$g(x_1, \ldots, x_5) = f_A(x_1) f_B(x_1, x_2, x_3) f_C(x_3) f_D(x_3, x_4, x_5) f_E(x_5), \tag{2.5}$$

Algorithm 1: Algoritmo para calcular $g_1(x_1)$ según la ecuación (2.6)

Entrada: x_1, f_A, f_B, f_C, f_D y f_E
Salida : $S = g_1(x_1)$

```
1.1  S_3 ← 0
1.2  foreach x_3 ∈ A do
1.3  |   S_2 ← 0
1.4  |   foreach x_2 ∈ A do
1.5  |   |   S_2 ← S_2 + f_B(x_1, x_2, x_3)
1.6  |   end
1.7  |   S_5 ← 0
1.8  |   foreach x_5 ∈ A do
1.9  |   |   S_4 ← 0
1.10 |   |   foreach x_4 ∈ A do
1.11 |   |   |   S_4 ← S_4 + f_D(x_3, x_4, x_5)
1.12 |   |   end
1.13 |   |   S_5 ← S_5 + f_E(x_5) · S_4
1.14 |   end
1.15 |   S_3 ← S_3 + f_C(x_3) · S_2 · S_5
1.16 end
1.17 S ← f_A(x_1) · S_3
```

entonces podríamos calcular $g_1(x_1)$ de la siguiente forma:

$$g_1(x_1) = f_A(x_1) \underbrace{\sum_{x_3} f_C(x_3) \underbrace{\sum_{x_2} f_B(x_1, x_2, x_3)}_{S_2(x_1,x_3)} \underbrace{\sum_{x_5} f_E(x_5) \underbrace{\sum_{x_4} f_D(x_3, x_4, x_5)}_{S_4(x_3,x_5)}}_{S_5(x_3)}}_{S_3(x_1)} \tag{2.6}$$

La ecuación (2.6) se puede resolver mediante el Algoritmo 1 el cual requiere $N^3 + 2N^2 + N$ sumas, $N^2 + 2N + 1$ multiplicaciones y $N^3 + 2N^2 + N + 1$ evaluaciones de funciones. Esto representa una reducción significativa en comparación con las $N^4 - 1$ sumas, $4N^4$ multiplicaciones y $5N^4$ evaluaciones de funciones requeridas en el cálculo directo de $g_1(x_1)$ sin distribuir las sumatorias. Esta reducción de los requerimiento de cálculo se torna particularmente importante cuando se incrementa la diferencia (o relación) entre el número de variables de la función global y el número de variables de sus funciones factores[3].

[3] En particular, esta relación es muy grande para las funciones de probabilidad conjunta de códigos LDPC. Esto último explica en gran parte la eficiencia del algoritmo SPA cuando

Algorithm 2: Algoritmo para calcular $g_3(x_3)$ según la ecuación (2.7)

Entrada: x_3, f_A, f_B, f_C, f_D y f_E
Salida : $S = g_3(x_3)$

2.1 $S_1 \leftarrow 0$
2.2 $S_5 \leftarrow 0$
2.3 **foreach** $x_1 \in A$ **do**
2.4 $\quad S_2 \leftarrow 0$
2.5 $\quad$ **foreach** $x_2 \in A$ **do**
2.6 $\quad\quad S_2 \leftarrow S_2 + f_B(x_1, x_2, x_3)$
2.7 $\quad$ **end**
2.8 $\quad S_1 \leftarrow S_1 + f_A(x_1) \cdot S_2$
2.9 $\quad S_4 \leftarrow 0$
2.10 $\quad$ **foreach** $x_4 \in A$ **do**
2.11 $\quad\quad S_4 \leftarrow S_4 + f_D(x_3, x_4, x_5)$
2.12 $\quad$ **end**
2.13 $\quad S_5 \leftarrow S_5 + f_E(x_5) \cdot S_4$
2.14 **end**
2.15 $S \leftarrow f_C(x_3) \cdot S_1 \cdot S_5$

Continuando con otro ejemplo ilustrativo, calculemos ahora $g_3(x_3)$ de la siguiente forma:

$$g_3(x_3) = f_C(x_3) \cdot \underbrace{\sum_{x_5} f_E(x_5) \cdot \underbrace{\sum_{x_4} f_D(x_3, x_4, x_5)}_{S_4(x_3, x_5)}}_{S_5(x_3)} \cdot \underbrace{\sum_{x_1} f_A(x_1) \cdot \underbrace{\sum_{x_2} f_B(x_1, x_2, x_3)}_{S_2(x_1, x_3)}}_{S_1(x_3)} \tag{2.7}$$

La ecuación (2.7) puede resolverse mediante el Algoritmo 2 y requiere un total de $2N^2 + 2N$ sumas, $2N + 2$ multiplicaciones y $2N^2 + 2N + 1$ evaluaciones de funciones. Nuevamente, la complejidad de cálculo es menor que la requerida al hacer el cálculo utilizando directamente la estrategia de la ecuación (2.1). Además, el lector atento notará que la complejidad de calcular conjuntamente $g_1(x_1)$ y $g_3(x_3)$ es menor que la suma de las complejidades de calcular cada una por separado. Esto último se debe a que es posible compartir algunos resultados parciales en el cálculo de ambas funciones marginales. En particular, se pueden compartir las sumatorias $S_2(x_1, x_3)$, $S_4(x_3, x_5)$ y $S_5(x_3)$. A continuación generalizaremos estas ideas utilizando una representación gráfica de la estructura de facturación de g.

es utilizado como algoritmo de decodificación.

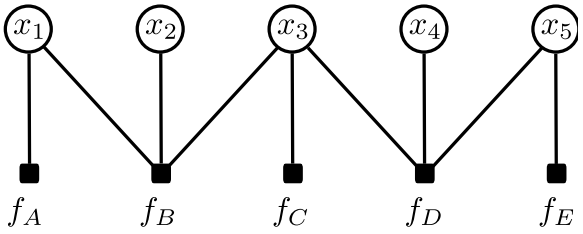


Figura 2.1: Grafo factorial de la función (2.5).

2.1.1. El Grafo Factorial

El análisis presentado previamente puede interpretarse como un algoritmo que opera sobre un grafo. Dicho grafo, denominado *grafo factorial* (GF), es un grafo bipartito que representa la estructura de factorización de una función global g. El mismo se compone de un nodo para cada variable x_i denominado *nodo-variable*, un nodo para cada función local f_j denominado *nodo-factor*, y lados o *ramas* que conectan el nodo-variable x_i con el nodo-factor f_j si y solo si x_i es un argumento de f_j. Tal grafo es denotado mediante la triada $(\mathcal{X}, \mathcal{F}, \mathcal{R})$ donde $\mathcal{X}$ y $\mathcal{F}$ son los conjuntos de nodos variables y nodos factores respectivamente y $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{F}$ es el conjunto de ramas, i.e., $\mathcal{R}$ es el conjunto de todos los pares ordenados (x, f) con $x \in \mathcal{X}$ y $f \in \mathcal{F}$ tal que x es una variable de f. A manera de ejemplo la Figura 2.1 muestra el grafo factorial de la función g según la ecuación (2.5).

Dentro del grafo factorial, un camino de longitud n entre dos nodos a y b es una secuencia $z_1, z_2, \ldots, z_n$ de nodos tales que $z_1 = a$, $z_n = b$ y existe una rama que conecta z_i con z_{i+1} para $i = 1, 2, \ldots, n-1$. El grafo se dice *conexo* si para cualquier par de nodos existe un camino entre ambos. Por otro lado, un ciclo de longitud n es un camino de longitud n que es cerrado, es decir que empieza y termina en el mismo nodo ($z_1 = z_n$). Si el grafo factorial no tiene ciclos y es conexo entonces se dice que es un grafo tipo árbol. Notar que el GF de g dado por la ecuación (2.5) es un árbol. A continuación sólo analizaremos grafos factoriales tipo árbol. Sin embargo, todo lo dicho será valido para grafos sin ciclos no necesariamente conexos, es decir para lo que se denomina un *grafo tipo bosque* (una unión disjunta de grafos tipo árbol). Posteriormente, en la Sección 2.1.4 consideraremos el caso más general de grafos factoriales con ciclos.

2.1.2. Cálculo de una Función Marginal

A la representación gráfica de un grafo de árbol se le puede asociar un nodo raíz a partir del cual se derivan todas las ramas y los nodos del grafo. Cualquier nodo de un grafo tipo árbol puede utilizarse como nodo raíz. Como veremos a continuación, el cálculo de la función marginal $g_i(x_i)$ de una función global g se reduce a un algoritmo que opera sobre el grafo de árbol

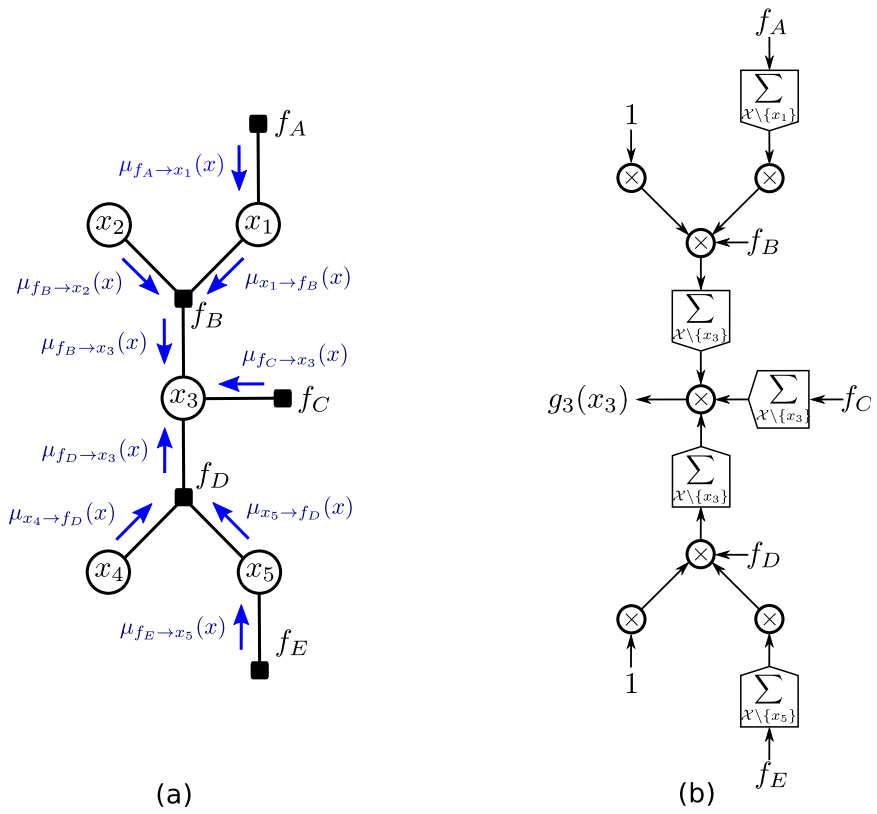


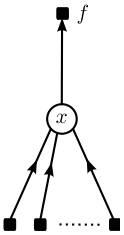
Figura 2.2: Grafo de árbol con raíz en x_3 asociado a la función (2.5).

de g con raíz en x_i. La Figuras 2.2 (a) y (b) muestran la secuencias de cálculos que hay que realizar sobre el GF con raíz en x_3 para obtener la función marginal $g(x_3)$ de la función global dada por la ecuación (2.5). La notación utilizada en la Figura 2.2 se clarifica en la ecuación (2.8):

$$
\begin{aligned}
g_3(x_3) \;=\; & \underbrace{f_C(x_3)}_{\mu_{f_C\to x_3}} \cdot \underbrace{\sum_{x_5}\sum_{x_4} f_D(x_3,x_4,x_5)\cdot \underbrace{1}_{\mu_{x_4\to f_D}} \cdot \underbrace{f_E(x_5)}_{\mu_{x_5\to f_D}=\mu_{f_E\to x_5}}}_{\mu_{f_D\to x_3}} \\
& \cdot \underbrace{\sum_{x_1}\sum_{x_2} f_B(x_1,x_2,x_3)\cdot \underbrace{1}_{\mu_{x_2\to f_B}} \cdot \underbrace{f_A(x_1)}_{\mu_{x_1\to f_B}=\mu_{f_A\to x_1}}}_{\mu_{f_B\to x_3}}
\end{aligned}
\tag{2.8}
$$

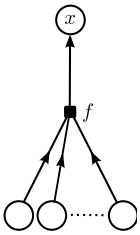
Del ejemplo anterior se observa que el proceso de cálculo realizado sobre el grafo factorial se reduce al cálculo de dos tipos de mensajes. Dichos

mensajes se envían desde los nodos-variable a los nodos-factores y viceversa. El mensaje que se envía desde el nodo-variable hacia el nodo-factor se calcula de la siguiente forma:



$$\mu_{x\to f}(x) = \prod_{h\in\mathcal{F}_x\setminus\{f\}} \mu_{h\to x}(x). \tag{2.9}$$

Por otro lado, el mensaje que se envía desde el nodo-factor hacia el nodo-variable se calcula como:



$$\mu_{f\to x}(x) = \sum_{\mathcal{X}\setminus\{x\}} \left(f(\mathcal{X}_f) \prod_{y\in\mathcal{X}_f\setminus\{x\}} \mu_{y\to f}(y) \right) \tag{2.10}$$

2.1.3. Cálculo Conjunto de Todas las Funciones Marginales

En muchas circunstancias, y en particular en la decodificación de códigos LDPC, nuestro interés es el cálculo de $g_i(x_i)$ para más de un valor de i. Dicho cálculo puede ser llevado a cabo aplicando el algoritmo anterior para cada i en forma separada. Sin embargo, este método no es eficiente debido a que muchos de los sub-cálculos realizados para diferentes valores de i se repiten. El objetivo será entonces reutilizar los sub-cálculos.

Cada función marginal $g_i(x_i)$ es el producto de los mensajes que llevan al nodo x_i. Luego, es necesario calcular todos los mensajes que entran a los nodos variables. Esto puede realizarse sobre grafos tipo árbol utilizando una estrategia tipo *programación dinámica* [15] [57] tanto en su forma "top-down" como "botton-up" a saber:

1. **Top-down:** Se utiliza un algoritmo recursivo con memoria. Esto es, partiendo de un orden cualquiera, se calculan las funciones marginales $g_i(x_i)$ mediante llamadas a un par de funciones conjuntamente recursivas para calcular los mensajes $\mu_{f\to x}(x)$ y $\mu_{x\to f}(x)$ según las ecuaciones (2.10) y (2.9) respectivamente. A dichas funciones se les agrega una memoria que guarda todos los mensajes que fueron calculados previamente evitando que se calculen mas de una vez.

2. **Botton-up:** En este método se realizan los cálculos por etapas. En cada etapa se calculan todos los mensajes (no calculados en una etapa

previa) para los cuales se dispone de la información necesaria para dicho cálculo. Luego de N etapas de cálculo, donde N es el diámetro del grafo[4], se habrá finalizado el cálculo de todos los mensajes.

Se destaca que la estrategia "botton-up" es el método clásico para implementar los decodificadores de códigos LDPC.

2.1.4. El SPA Sobre un Grafo con Ciclos

El algoritmo SPA permite calcular en forma eficiente las funciones marginales de una función global cuyo grafo factorial no contiene ciclos. Sin embargo, el procedimiento descripto en las Secciones 2.1.2 y 2.1.3 no puede aplicarse cuando dicho grafo tiene ciclos. Esto se debe a que un ciclo en el grafo factorial se traslada a una relación recursiva circular en las funciones $\mu_{f\to x}(x)$ y $\mu_{x\to f}(x)$ que no puede ser resuelta por etapas. Para resolver tal situación, se requiere la resolución conjunta de todas las funciones factores involucradas en dicho ciclo (lo que impide sacar provecho de la factorización). Tal resolución conjunta equivale a eliminar los ciclos mediante la agrupación de las variable involucradas. Esto genera una nueva *super-variable* cuyo dominio es el producto cartesiano de los dominios de las variables involucradas. A dichas *super-variables* hay que asociarles nuevas funciones factores que son generalmente más complejas que las funciones factores originales. Esta técnica se conoce como *agrupamiento* y se utiliza por ejemplo en [91] para derivar la transformada rápida de Fourier. Desafortunadamente, tal transformación no es siempre viable ya que usualmente conduce a *super-variables* con un dominio excesivamente grande.

Un método alternativo para operar en grafos con ciclos consiste en inicializar todos los mensajes $\mu_{f\to x}(x)$ y/o $\mu_{x\to f}(x)$ con un valor determinado y luego actualizar sus valores en forma iterativa según las ecuaciones (2.10) y (2.9). El algoritmo resultante no tiene una finalización natural como en el caso original, sino que los mensajes pueden pasar múltiples veces por la misma rama. La convergencia de dicho procedimiento no está garantizada como así tampoco la exactitud del resultado en caso de que exista convergencia. Sin embargo, para algunas de las aplicaciones interesantes como la decodificación de códigos turbo o LDPC, el resultado se aproxima bastante a la solución ideal. Tal procedimiento se detalla en el Algoritmo 3 en donde N es el número de variables de la función global, $\mathcal{R}$ es el conjunto de todas las ramas del grafo factorial, $\mathcal{R}_1(i) \subseteq \mathcal{R}$ y $\mathcal{R}_2(i) \subseteq \mathcal{R}$ son los subconjuntos ramas usados en la i-ésima iteración para calcular los mensajes $\mu_{x\to f}(x)$ y $\mu_{f\to x}(x)$ respectivamente; por último, I es el número de iteraciones. Si $\mathcal{R}_1 = \mathcal{R}_2 = \mathcal{R}$ entonces en cada iteración se actualizan

[4] El diámetro de un grafo conexo es la mayor distancia entre dos nodos cualesquiera, donde la distancia entre dos nodos es el mínimo número de ramas que hay que atravesar para llegar de un nodo al otro.

Algorithm 3: Algoritmo SPA sobre un grafo con ciclos

Entrada: I,$\mathcal{R}_1$,$\mathcal{R}_2$,α
Salida : $g_i(x)$ para $i = 1, 2, \ldots, N$

foreach $(x, f) \in R$ **do** `/* Inicialización de mensajes */`
  $\mu^{(0)}_{f\to x}(x) = \alpha$
end
for $i = 1, 2, \ldots, I$ **do** `/* Iteraciones */`
  foreach $(x, f) \in \mathcal{R}$ **do**
    if $(x, f) \in \mathcal{R}_1(i)$ **then**
      $\mu^{(i)}_{x\to f}(x) = \prod_{h\in\mathcal{F}_x\setminus\{f\}} \mu^{(i-1)}_{h\to x}(x)$
    else
      $\mu^{(i)}_{x\to f}(x) = \mu^{(i-1)}_{x\to f}(x)$
    end
  end
  foreach $(x, f) \in \mathcal{R}$ **do**
    if $(x, f) \in \mathcal{R}_2(i)$ **then**
      $\mu^{(i)}_{f\to x}(x) = \sum_{\mathcal{X}\setminus\{x\}} \left(f(X) \prod_{y\in\mathcal{X}_f\setminus\{x\}} \mu^{(i-1)}_{y\to f}(y) \right)$
    else
      $\mu^{(i)}_{f\to x}(x) = \mu^{(i-1)}_{f\to x}(x)$
    end
  end
end
for $k = 1, 2, \ldots, N$ **do** `/* Cálculo de marginales */`
  $g_k(x) = 1$
  foreach $f \in \mathcal{F}_{x_k}$ **do**
    $g_k(x) = g_k(x) \cdot \mu^{(I)}_{f\to x_k}(x)$
  end
end

todos los mensajes; a ésto se lo denomina *flooding schedule* [90] y constituye la metodología clásica para la implementación del algoritmo SPA. A lo largo de esta Tesis nos restringiremos a dicha metodología. Sin embargo, existe una amplia variedad de opciones para elegir los subconjuntos $\mathcal{R}_1$ y $\mathcal{R}_2$ en cada una de las cuales el algoritmo convergerá en forma diferente, mejorando el desempeño en algunos casos y empeorándolo en otros[5].

[5]Esto dará lugar a diferentes variantes del algoritmo de decodificación de códigos LDPC, véase [178] [140] [179] y las referencia allí citadas.

2.2. Los Códigos LDPC

Un código LDPC es un código lineal definido como el conjunto de elementos del espacio nulo (o núcleo) de una matriz de paridad H de tipo *rala*[6], esto es:

$$\mathfrak{C} \quad = \quad \{\mathbf{v} \ : \ H\mathbf{v} = 0\}. \tag{2.11}$$

Sea H una matriz con dimensiones $M \times N$. Dicha matriz define un código de longitud N y dimensión K con $K = N - \text{rango}(H)$. Asociada a la matriz H, se define un grafo bipartito denominado grafo de Tanner (GT) [151], que caracteriza completamente a dicho código (ver Figura 2.3). El grafo de Tanner posee dos tipos de nodos: los nodos de variable v_i y los nodos de chequeo c_i. Los nodos v_i y c_j representan al i-ésimo bit (o símbolo en el caso de un código no binario) codificado o columna de H y a la j-ésima ecuación de chequeo de paridad o fila de H, respectivamente. En GT, sólo existen conexiones entre un nodo variable y un nodo chequeo, en particular v_i se encuentra conectado a c_j sí y solo sí $H_{j,i} \neq 0$, donde los índices j e i indican la fila y columna de H respectivamente. El grado de un determinado nodo en el GT se define como el número de conexiones que posee dicho nodo. Equivalentemente, los grados de v_i y c_j representan la cantidad de elementos no nulos en la i-ésima columna y j-ésima fila de H, respectivamente. Si todos los nodos v_i poseen el mismo grado γ y todos los nodos c_i poseen el mismo grado ρ, el código se denomina *regular*. Similar a la notación usada en el algoritmo SPA, utilizaremos la siguiente notación para el código LDPC. Denotamos por $\mathcal{V}$ al conjunto de nodos variables y $\mathcal{C}$ es el conjunto de nodos de chequeo. Para $v_i \in \mathcal{V}$ y $c_j \in \mathcal{C}$, los conjuntos de vecinos de v_i y c_i son denotados $\mathcal{C}_{v_i}$ y $\mathcal{V}_{c_j}$ respectivamente, esto es

$$\mathcal{C}_{v_i} = \{c_j \in \mathcal{C} \ : \ H_{j,i} \neq 0\} \text{ y } \mathcal{V}_{c_j} = \{v_i \in \mathcal{V} \ : \ H_{j,i} \neq 0\}$$

Una restricción de diseño de H que se suele considerar en la bibliografía como parte de la definición de códigos LDPC es que el grafo de Tanner asociado a la matriz de paridad no debe tener ciclos de longitud 4. Esto significa que no debe existir en la matriz de paridad dos filas (y/o columnas) que tengan dos o más elementos no nulos en las mismas posiciones [137] (ver Figura 2.3). Lo anterior suele denominarse *restricción fila-columna*. Como se verá mas adelante, la razón para exigir tal restricción a la matriz

[6] Una matriz *rala* o *esparcida*, es una matriz que posee una reducida cantidad de elementos no nulos. Sin embargo, qué es considerado como "una reducida cantidad" no tiene una definición precisa. En el contexto de los códigos LDPC, se suele adoptar la noción de que si la relación entre la cantidad de elementos no-nulos y nulos es inferior a 0.01 entonces la matriz es considerada rala. Sin embargo, lo anterior sólo tiene un carácter orientativo. Cuan rala sea la matriz va a estar determinado en última instancia por el desempeño del código en cuestión cuando es decodificado por el algoritmo SPA y por la complejidad de implementación de este último.

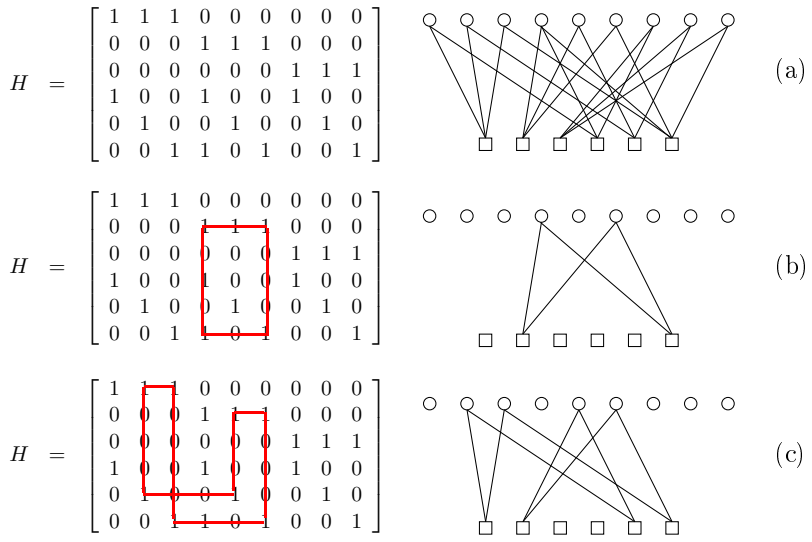


Figura 2.3: (a) Ejemplo de la matriz de paridad y el grafo de Tanner, (b) Ejemplo de un ciclo de longitud 4, (c) Ejemplo de un ciclo de longitud 8.

de paridad se debe a que tales ciclos de longitud 4 acarrean problemas de desempeño cuando los códigos LDPC se decodifican con el algoritmo SPA. Personalmente, prefiero no incluir tal restricción como parte de la definición de códigos LDPC debido a que: (*i*) está más asociada al algoritmo de decodificación que a los códigos propiamente dichos e (*ii*) incluso si no se cumple no indica necesariamente que el algoritmo SPA tenga un bajo desempeño. Lo anterior, en conjunto con la restricción de usar una matriz de paridad tipo rala nos indicaría que los códigos LDPC podrían ser considerados como un sub-producto del algoritmo SPA, es decir como al conjunto de códigos lineales que pueden ser decodificados eficientemente mediante algoritmo SPA. Además, este último conjunto podría ser generalizado para incluir códigos no lineales.

En relación a lo anterior, y a diferencia de lo que ocurre en la teoría clásica de códigos, es común que cuando nos referimos a los códigos LDPC no nos estamos refiriendo únicamente al sub-espacio definido por alguna matriz de paridad, pudiendo haber una infinidad de matrices diferentes que generan el mismo sub-espacio[7] (i.e., generan el mismo código). Por el contrario, es común incluir en el concepto de códigos LDPC al algoritmo de decodificación SPA y de esta forma se incluye indirectamente a la estructura de matriz de paridad. Esto se debe a que el desempeño del algoritmo SPA está íntimamente relacionado a la estructura del grafo de Tanner el cual está completamente determinado por la matriz de paridad. Es muy común

[7] Donde permitimos que la matriz no tenga rango completo.

que dos matrices que representan el mismo código en el sentido clásico tengan desempeños completamente diferentes cuando son decodificadas con el algoritmo SPA. Por esta razón los códigos LDPC se suelen definir mediante la descripción de su matriz de paridad. En el Capítulo 3 describiremos la construcción de dicha matriz, sin embargo, será de utilidad primero comprender el proceso de decodificación el cual describiremos a continuación.

2.3. Decodificación de Códigos LDPC Binarios

El esquema de la Figura 2.4 muestra un sistema de comunicación formado por la concatenación de un *codificador*, un *canal*, y un *decodificador* en donde:

1. El *canal* hace referencia al canal equivalente en tiempo discreto que incluye no sólo al medio físico a través del cual se realiza la transmisión, sino también a todas las etapas del transmisor y del receptor que se encuentran entre el codificador y el decodificador (esto es: filtros, conversión entre tiempo discreto y continuo, moduladores y demoduladores, etc).

2. El decodificador es dividido en dos sub-bloques denotados Deco:A y Deco:B en donde: el Deco:A se encarga de determinar la palabra código más probable en función de los símbolos recibidos y el Deco:B se encarga de recuperar los bits de información a partir de los bits de la palabra código decodificada.

En dicho esquema denotamos con $\mathbf{b} = \mathbf{b}_1^K = (b_1, b_2, \ldots, b_K) \in \{0,1\}^K$ al vector de bits de información que se quiere transmitir. Este vector es luego transformado por el codificador en el vector palabra código $\mathbf{v} = \mathbf{v}_1^N = (v_1, v_2, \ldots, v_N) \in \{0,1\}^N$ conformando la entrada al canal. La salida del canal es el vector $\mathbf{y} = \mathbf{y}_1^N = (y_1, y_2, \ldots, y_N) \in \mathbb{R}^N$. Por otro lado, los vectores $\hat{\mathbf{b}}$ y $\hat{\mathbf{v}}$ representan las estimaciones de los vectores $\mathbf{b}$ y $\mathbf{v}$, respectivamente, realizadas por el receptor. Los vectores $\mathbf{b}$, $\hat{\mathbf{b}}$, $\mathbf{v}$, $\hat{\mathbf{v}}$ e $\mathbf{y}$ son una realización particular de los vectores aleatorios $\mathbb{B}$, $\hat{\mathbb{B}}$, $\mathbb{V}$, $\hat{\mathbb{V}}$ e $\mathbb{Y}$, respectivamente. En este esquema, el canal queda completamente caracterizado por la función de densidad de probabilidad condicional $f_{\mathbb{Y}|\mathbb{V}}(\mathbf{y}|\mathbf{v})$. Sin embargo, la generalidad de dicha ecuación esconde una enorme complejidad que trasciende el objetivo de esta sección.

Para analizar el algoritmo de decodificación del código LDPC será suficiente restringirnos al caso particular en que la salida del canal en el tiempo k, y_k, sólo depende del valor de la entrada en el mismo instante de tiempo, v_k. Bajo esta última restricción, es posible factorizar la función de densidad conjunta del canal según se muestra en la ecuación (2.12):

$$f_{\mathbb{Y}|\mathbb{V}}(\mathbf{y}|\mathbf{v}) = \prod_{k=0}^{N} f_{\mathbb{Y}_k|\mathbb{V}_k}(y_k|v_k) \tag{2.12}$$

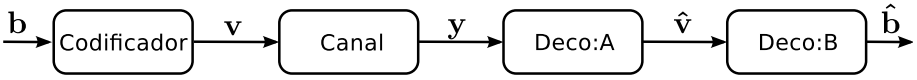


Figura 2.4: Diagrama en bloques del sistema de transmisión.

Gracias a la factorización de la ecuación (2.12) no existe interacción entre las diferentes palabras código; por lo que es posible restringirnos a la transmisión de una única palabra código. En estas condiciones, consideraremos a K y a N como la dimensión y la longitud del código $\mathfrak{C}$, respectivamente. El bloque *Deco:A* en la Figura 2.4 puede realizar dos tipos de decodificación, a saber: *decodificación por bloque* (o *decodificación por secuencia*) y *decodificación por bit*.

La decodificación por bloque tiene como objetivo determinar cuál es la palabra código más probable según la ecuación

$$\hat{\mathbf{v}} = \arg\max_{\mathbf{v}} P_{\mathbb{V}|\mathbb{Y}}(\mathbf{v}|\mathbf{y}) \tag{2.13}$$

donde

$$P_{\mathbb{V}|\mathbb{Y}}(\mathbf{v}|\mathbf{y}) = \frac{\chi_{\mathfrak{C}}(\mathbf{v}) \cdot f_{\mathbb{Y}|\mathbb{V}}(\mathbf{y}|\mathbf{v})}{\sum_{\mathbf{v}\in\{0,1\}^N} \chi_{\mathfrak{C}}(\mathbf{v}) \cdot f_{\mathbb{Y}|\mathbb{V}}(\mathbf{y}|\mathbf{v})}, \tag{2.14}$$

y $\chi_{\mathfrak{C}}(\mathbf{v})$ es la función característica del código, esto es $\chi_{\mathfrak{C}}(\mathbf{v}) = 1$ si $\mathbf{v} \in \mathfrak{C}$ mientras que $\chi_{\mathfrak{C}}(\mathbf{v}) = 0$ si $\mathbf{v} \notin \mathfrak{C}$.

Por otro lado, la decodificación por bit tiene por objetivo determinar cuál es el valor más probable para cada bit de la palabra código según la ecuación

$$\hat{v_i} = \begin{cases} 1 & \text{si} \quad P_{\mathbb{V}_i|\mathbb{Y}}(1|\mathbf{y}) > P_{\mathbb{V}_i|\mathbb{Y}}(0|\mathbf{y}) \\ 0 & \text{sino} \end{cases} \tag{2.15}$$

donde

$$P_{\mathbb{V}_i|\mathbb{Y}}(v|\mathbf{y}) = \sum_{\mathbf{v}:v_i=v} P_{\mathbb{V}|\mathbb{Y}}(\mathbf{v}|\mathbf{y}) \tag{2.16}$$

es la probabilidad marginal de v_i dado $\mathbf{y}$.

Notar que la secuencia decodificada en este último caso puede no ser una palabra código. Como veremos en lo sucesivo, la decodificación de códigos LDPC mediante el algoritmo SPA es una decodificación por bit:

El algoritmo SPA tendrá la finalidad de calcular dichas probabilidades marginales[8]. A tal efecto, construyamos la factorización de la función de

[8] Antes de proseguir, nótese que la transformación de $\mathbf{b}$ a $\mathbf{v}$, establecida por el *codificador*, es una correspondencia uno a uno. Además, si $\mathfrak{C}$ es sistemático, la transformación de $\hat{\mathbf{v}}$ a $\hat{\mathbf{b}}$ se reduce a la operación trivial de eliminar los bits de paridad. Ambas transformaciones son simples y de carácter secundario para el análisis del algoritmo SPA y por simplicidad no serán consideradas el proceso de decodificación. Es decir, restringiremos el problema a la estimación $\hat{\mathbf{v}}$ de la palabra código transmitida $\mathbf{v}$ realizada por el bloque *Deco:A*.

probabilidad conjunta $P_{\mathbb{V}|\mathbb{Y}}(\mathbf{v}|\mathbf{y})$ mediante la combinación de (2.14) y (2.12) obteniendo la ecuación

$$P_{\mathbb{V}|\mathbb{Y}}(\mathbf{v}|\mathbf{y}) \quad = \quad \Lambda \cdot \chi_{\mathfrak{C}}(\mathbf{v}) \cdot \prod_{k=0}^{N} f_{\mathbb{Y}_k|\mathbb{V}_k}(y_k|v_k), \tag{2.17}$$

donde

$$\Lambda = \Big(\sum_{\mathbf{v}\in\{0,1\}^N} f_{\mathbb{Y}|\mathbb{V}}(\mathbf{y}|\mathbf{v}) \cdot \chi_{\mathfrak{C}}(\mathbf{v}) \Big)^{-1}$$

es una constante de normalización. Por otro lado, siendo $\mathcal{V}_{c_j}$ el conjunto de variables involucradas en la j-ésima ecuación de paridad, denotamos por $\chi_{c_j}\left(\mathcal{V}_{c_j}\right)$ su respectiva función característica. Luego, es posible factorizar $\chi_{\mathfrak{C}}\left(\mathcal{V}\right)$ de la siguiente manera:

$$\chi_{\mathfrak{C}}\left(\mathcal{V}\right) = \prod_{j=1}^{M} \chi_{c_j}\left(\mathcal{V}_{c_j}\right) \tag{2.18}$$

Combinando las ecuaciones (2.17) y (2.18) obtenemos:

$$P_{\mathbb{V}|\mathbb{Y}}(\mathbf{v}|\mathbf{y}) \quad = \quad \Lambda \cdot \prod_{j=1}^{M} \chi_{c_j}\left(\mathcal{V}_{c_j}\right) \cdot \prod_{k=0}^{N} f_{\mathbb{Y}_k|\mathbb{V}_k}(y_k|v_k) \tag{2.19}$$

El cálculo de las funciones marginales se reduce entonces al cálculo de

$$P_{\mathbb{Y}_k|\mathbb{Y}}(v|\mathbf{y}) = \overbrace{\left[f_{\mathbb{Y}_k|\mathbb{V}_k}(y_k|v_k) \right]}^{P_k^{apr}(v_k)} \cdot \overbrace{\Lambda \cdot \sum_{\mathcal{V}\setminus\{v_k\}} \left[\prod_{j=1}^{M} \chi_{c_j}\left(\mathcal{V}_{c_j}\right) \cdot \prod_{i\neq k} f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|v_i) \right]}^{P_k^{ext}(v_k)} \tag{2.20}$$

para $k = 1, 2, \ldots, N$. Los términos $P_k^{apr}(v_k)$ y $P_k^{ext}(v_k)$ en la ecuación (2.20) se denominan *información a priori* e *información extrínseca*. Esto es, $P_k^{apr}(v_k)$ y $P_k^{ext}(v_k)$ representan la información que aporta el modelo del canal y la estructura del código, respectivamente, sobre la probabilidad $P_{\mathbb{V}_k|\mathbb{Y}}(v|\mathbf{y})$. El cálculo de la ecuación (2.20) para todo k puede ser eficientemente aproximado mediante el algoritmo SPA.

2.3.1. El Algoritmo

La decodificación de códigos LDPC mediante el algoritmo SPA se reduce a aplicar el mismo sobre el grafo factorial asociado a la ecuación (2.20). La Figura 2.5 muestra el grafo factorial de un código LDPC general. Como se describió en el Algoritmo 3, una iteración del algoritmo SPA está compuesta de dos pasos. En el primer paso (A), se calculan y envían los mensajes $\mu_{v_i\to c_j}$ del nodo-variable v_i hasta el nodo-chequeo c_j para todo par $(v_i, c_j) \in \mathcal{R}$

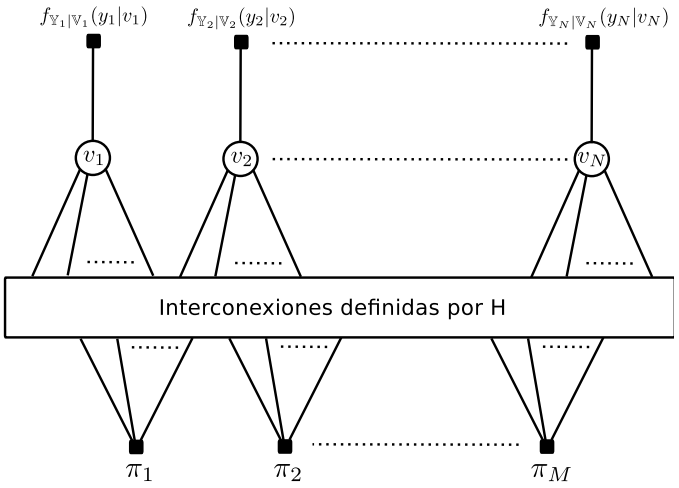


Figura 2.5: Grafo Factorial de $P_{\mathbb{Y}|\mathbb{Y}}(\mathbf{v}|\mathbf{y})$.

donde $\mathcal{R}$ es el conjunto de ramas del grafo factorial. En el segundo paso (B) se calculan y envían los mensajes $\mu_{c_j \to v_i}$ desde el nodo-chequeo c_j hasta el nodo-variable v_i para todo par $(v_i, c_j) \in \mathcal{R}$. Se repiten los pasos (A) y (B) tantas veces como iteraciones se requieran. Finalmente, en un tercer paso (C) se calcula la probabilidad *a posteriori* $P_{\mathbb{V}_k|\mathbb{Y}}(v|\mathbf{y})$. Las ecuaciones (2.21), (2.22) y (2.23) resumen los cálculos realizados en cada paso:

A. Mensaje de variable a chequeo:

$$\mu_{v_i \to c_j}(v_i) = f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|v_i) \cdot \prod_{c_m \in \mathcal{C}_{v_i} \setminus \{c_j\}} \mu_{c_m \to v_i}(v_i) \tag{2.21}$$

B. Mensajes de chequeo a variable:

$$\mu_{c_j \to v_i}(v_i) = \sum_{\mathcal{V}_{c_j} \setminus \{v_i\}} \left(\chi_{c_j}\left(\mathcal{V}_{c_j}\right) \prod_{v_m \in \mathcal{V}_{c_j} \setminus \{v_i\}} \mu_{v_m \to c_j}(v_m) \right) \tag{2.22}$$

C. Cálculo de la probabilidad a posteriori

$$P_{\mathbb{V}_i|\mathbb{Y}}(v|\mathbf{y}) = f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|v_i) \cdot \prod_{c_m \in \mathcal{C}_{v_i}} \mu_{c_m \to v_i}(v_i) \tag{2.23}$$

Los mensajes $\mu_{c_m \to v_i}(v_i)$ y $\mu_{v_i \to c_j}(v_i)$ son funciones cuyo argumento v_i sólo puede tomar los valores 0 y 1. Entonces es posible representar a dichos mensajes por los pares ordenados $(\mu_{c_m \to v_i}(0), \mu_{c_m \to v_i}(1))$ y $(\mu_{v_i \to c_j}(0), \mu_{v_i \to c_j}(1))$. Por otro lado, debido a que dichas funciones representan funciones de probabilidad, se verifica que $\mu_{c_m \to v_i}(0) +$

$\mu_{c_m \to v_i}(1) = \mu_{v_i \to c_j}(0) + \mu_{v_i \to c_j}(1) = 1$. En consecuencia, es posible reducir cada mensaje a un único valor. Existen diversas formas de definir el valor que representa a cada mensaje. Entre las más comúnes se pueden mencionar las siguientes:

(*i*) Asignar al mensaje el valor $\mu_{c_m \to v_i}(0)$ o $\mu_{c_m \to v_i}(1)$.

(*ii*) Asignar al mensaje la diferencia $\mu_{c_m \to v_i}(0) - \mu_{c_m \to v_i}(1)$ o su negación.

(*iii*) Asignar al mensaje la relación $\mu_{c_m \to v_i}(0)/\mu_{c_m \to v_i}(1)$ o su inversa.

(*iv*) Asignar al mensaje el logaritmo natural de la relación $\mu_{c_m \to v_i}(0)/\mu_{c_m \to v_i}(1)$ o su inversa.

Cada una de estas definiciones da lugar a diferentes estructuras de cálculo. Esta Tesis se enfocará en la opción (*iv*) la cual se denomina *relación logarítmica de verosimilitud*, y representa el estándar de facto para la implementación de decodificadores LDPC. Esto último se debe a que tal representación permite simplificar los operaciones del algoritmo SPA como veremos a continuación. Sean

$$L^e_{v_i \to c_j} = \ln\left[\frac{\mu_{v_i \to c_j}(0)}{\mu_{v_i \to c_j}(1)}\right] \tag{2.24}$$

$$L^e_{c_j \to v_i} = \ln\left[\frac{\mu_{c_j \to v_i}(0)}{\mu_{c_j \to v_i}(1)}\right] \tag{2.25}$$

$$L^a_{v_i} = ln\left[\frac{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|0)}{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|1)}\right] \tag{2.26}$$

donde la letra "L" en la notación anterior hace referencia a que estamos usando el logaritmo de la relación de probabilidades; los super-índices "e" y "a" hacen referencia a que los mensajes μ son la información extrínseca y $f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|v_i)$ es la información a priori. Combinando las ecuaciones (2.24) y (2.25) en el cálculo en la ecuación (2.21) es posible reescribir los mensajes de variable a chequeo de la siguiente manera:

$$\begin{aligned} L^e_{v_i \to c_j} &= ln\left[\frac{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|0) \cdot \prod\limits_{c_m \in \mathcal{C}_{v_i} \setminus \{c_j\}} \mu_{c_m \to v_i}(0)}{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|1) \cdot \prod\limits_{c_m \in \mathcal{C}_{v_i} \setminus \{c_j\}} \mu_{c_m \to v_i}(1)}\right] \\ &= ln\left[\frac{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|0)}{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|1)}\right] + \sum_{c_m \in \mathcal{C}_{v_i} \setminus \{c_j\}} ln\left[\frac{\mu_{c_m \to v_i}(0)}{\mu_{c_m \to v_i}(1)}\right] \\ &= L^a_{v_i} + \sum_{c_m \in \mathcal{C}_{v_i} \setminus \{c_j\}} L^e_{c_j \to v_i} \end{aligned} \tag{2.27}$$

Antes de proseguir con el cálculo de los mensajes de chequeo a variable $L^e_{c_j \to v_i}$ en el nuevo formato, haremos una conversión intermedia para la cual

definiremos los siguientes mensajes:

$$\begin{aligned}\Delta\mu_{v_i\to c_j} &= \mu_{v_i\to c_j}(0) - \mu_{v_i\to c_j}(1)\\ \Delta\mu_{c_j\to v_i} &= \mu_{c_j\to v_i}(0) - \mu_{c_j\to v_i}(1).\end{aligned} \tag{2.28}$$

Bajo dicha definición, es posible realizar la siguiente simplificación del mensaje de chequeo a variable:

$$\Delta\mu_{c_j\to v_i} = \prod_{v_m\in\mathcal{V}_{c_j}\setminus\{v_i\}} \Delta\mu_{v_m\to c_j}. \tag{2.29}$$

La demostración es directa a partir de la expansión de la productoria. Vale remarcar que dicha simplificación es de considerable importancia para el cálculo del algoritmo SPA sobre códigos binarios. Gracias a la misma, se reduce significativamente la cantidad de cálculos necesarios para el cómputo del mensaje de los nodos de chequeo a los nodos de variable. Reescribamos ahora la ecuación (2.29) en términos de $L^e_{v_i\to c_j}$ y $L^e_{c_j\to v_i}$. Definiendo

$$\begin{aligned}\mu_{v_i\to c_j}(0) &= \frac{e^{L^e_{v_i\to c_j}}}{e^{L^e_{v_i\to c_j}}+1}\\ \mu_{v_i\to c_j}(1) &= \frac{1}{e^{L^e_{v_i\to c_j}}+1},\end{aligned} \tag{2.30}$$

obtenemos que

$$\begin{aligned}\Delta\mu_{v_i\to c_j} &= \frac{e^{L^e_{v_i\to c_j}}}{e^{L^e_{v_i\to c_j}}+1} - \frac{1}{e^{L^e_{v_i\to c_j}}+1}\\ &= \frac{e^{L^e_{v_i\to c_j}}-1}{e^{L^e_{v_i\to c_j}}+1}\\ &= \frac{e^{\frac{1}{2}L^e_{v_i\to c_j}} - e^{-\frac{1}{2}L^e_{v_i\to c_j}}}{e^{\frac{1}{2}L^e_{v_i\to c_j}} + e^{-\frac{1}{2}L^e_{v_i\to c_j}}}\\ &= \tanh\left(\frac{L^e_{v_i\to c_j}}{2}\right)\end{aligned} \tag{2.31}$$

y de la misma manera

$$\Delta\mu_{c_j\to v_i} = \tanh\left(\frac{L^e_{c_j\to v_i}}{2}\right) \tag{2.32}$$

Finalmente, reemplazando (2.31) y (2.32) en (2.29) y resolviendo resulta:

$$\begin{aligned}\tanh\left(\frac{L^e_{c_j\to v_i}}{2}\right) &= \prod_{v_m\in\mathcal{V}_{c_j}\setminus\{v_i\}} \tanh\left(\frac{L^e_{v_m\to c_j}}{2}\right)\\ \ln\left[\tanh\left(\frac{L^e_{c_j\to v_i}}{2}\right)\right] &= \sum_{v_m\in\mathcal{V}_{c_j}\setminus\{v_i\}} \ln\left[\tanh\left(\frac{L^e_{v_m\to c_j}}{2}\right)\right]\end{aligned} \tag{2.33}$$

de aquí se obtiene

$$L^e_{c_j \to v_i} = \phi^{-1} \left\{ \sum_{v_m \in \mathcal{V}_{c_j} \setminus \{v_i\}} \phi\left[L^e_{v_m \to c_j} \right] \right\} \tag{2.34}$$

donde $\phi(x) = \ln\left[\tanh\left(x/2\right)\right]$ y $\phi^{-1}(x) = 2\tanh^{-1}\left(e^x\right)$.

Por último, el cálculo de la verosimilitud a posteriori que en el nuevo formato se denota como $L^o_{c_i}$, queda:

$$\begin{aligned} L^o_{v_i} &= \ln\left[\frac{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|0) \cdot \prod_{c_m \in \mathcal{C}_{v_i}} \mu_{c_m \to v_i}(0)}{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|1) \cdot \prod_{c_m \in \mathcal{C}_{v_i}} \mu_{c_m \to v_i}(1)} \right] \\ &= ln\left[\frac{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|0)}{f_{\mathbb{Y}_i|\mathbb{V}_i}(y_i|1)} \right] + \sum_{c_m \in \mathcal{C}_{v_i}} ln\left[\frac{\mu_{c_m \to v_i}(0)}{\mu_{c_m \to v_i}(1)} \right] \\ &= L^a_{v_i} + \sum_{c_m \in \mathcal{C}_{v_i}} L^e_{c_j \to v_i} \end{aligned} \tag{2.35}$$

Los tres pasos del algoritmo de decodificación en su versión basada en logaritmos de relaciones de probabilidad quedan entonces reducidos a las ecuaciones (2.36), (2.37) y (2.38) que se muestran a continuación:

A. Mensaje de variable a chequeo:

$$L^e_{v_i \to c_j} = L^a_{v_i} + \sum_{c_m \in \mathcal{C}_{v_i} \setminus \{c_j\}} L^e_{c_j \to v_i} \tag{2.36}$$

B. Mensajes de chequeo a variable:

$$L^e_{c_j \to v_i} = \phi^{-1} \left\{ \sum_{v_m \in \mathcal{V}_{c_j} \setminus \{v_i\}} \phi\left[L^e_{v_i \to c_j} \right] \right\} \tag{2.37}$$

C. Cálculo de la verosimilitud a posteriori:

$$L^o_{v_i} = L^a_{v_i} + \sum_{c_m \in \mathcal{C}_{v_i}} L^e_{c_j \to v_i} \tag{2.38}$$

2.3.2. El Algoritmo Mínimo-Suma

El cálculo del mensaje $L^e_{c_j \to v_i}$ es el más complejo de implementar ya sea en hardware o software. Esto último motiva el uso de diversas aproximaciones

para su simplificación [137]. Aquí presentaremos la aproximación más utilizada, la cual se basa en la siguiente propiedad de la función $\phi(x)$:

$$\begin{aligned}\phi^{-1}\Big(\phi(a)+\phi(b)\Big) &= 2\tanh^{-1}\left[\tanh\left(\frac{a}{2}\right)\cdot\tanh\left(\frac{b}{2}\right)\right]\\ &= sign(a\cdot b)\cdot \text{mín}\{|a|,|b|\}+\log\left(\frac{1+e^{-|a+b|}}{1+e^{-|a-b|}}\right)\\ &\approx sign(a\cdot b)\cdot \text{mín}\{|a|,|b|\}.\end{aligned}$$

Utilizando esta última aproximación de $\phi(x)$, el cálculo de los mensajes de chequeo a variable puede reescribirse así

$$\hat{L}^{e}_{c_j\to v_i} = \left(\min_{v_k\in\mathcal{V}_{c_j}\setminus\{v_i\}}\left\{\left|L^{e}_{v_k\to c_j}\right|\right\}\right)\cdot\left(\prod_{v_k\in\mathcal{V}_{c_j}\setminus\{v_i\}} sign(L^{e}_{v_k\to c_j})\right). \quad (2.39)$$

El mensaje $\hat{L}^{e}_{c_j\to v_i}$ no utiliza la función $\phi(x)$, reduciendo todo el cálculo a la búsqueda del mínimo valor absoluto del conjunto de mensajes $L^{e}_{v_k\to c_j}$ de entrada y al producto de sus signos. La versión del algoritmo SPA que utiliza esta aproximación se conoce con el nombre de algoritmo *mínimo-suma* [137] denotado como MSA por su siglas en inglés para *min-sum algorithm*. Mejoras adicionales a esta aproximación han sido propuestas en [29]. En particular, cabe mencionar el algoritmo *mínimo-suma re-escaldado* denotado como SMSA por su siglas en inglés para *scaled MSA*. En esta versión del algoritmo se introduce un factor de corrección en el mensaje $\hat{L}^{e}_{c_j\to v_i}$ obteniendo

$$\tilde{L}^{e}_{c_j\to v_i} = \alpha\cdot\hat{L}^{e}_{c_j\to v_i} \quad (2.40)$$

donde α es un número real menor que la unidad (típicamente $\alpha\approx 0{,}75$). Por su excelente relación entre desempeño y complejidad, a lo largo de esta Tesis se utilizará el algoritmo SMSA salvo que se aclare explícitamente el uso de otro algoritmo.

2.4. Estimación del Desempeño

La estimación del desempeño del algoritmo SPA o de sus simplificaciones (MSA, SMSA, etc) como métodos de decodificación de códigos LDPC presenta grandes dificultades. El grafo factorial de los códigos LDPC posee ciclos que influyen de forma compleja sobre dicho algoritmo. Esto último hace que los mensajes que se propagan en el grafo factorial no sean estadísticamente independientes, siendo necesario calcular una función de probabilidad conjunta que los englobe a todos para poder caracterizarlos. El problema se presenta al no poder calcular dicha función debido al gran

número de dimensiones que posee[9] (tantas dimensiones como bits tenga el código). Tal dependencia entre los mensajes genera dos efectos a saber:

1) una degradación general del desempeño que es producto de una mala estimación de las funciones marginales y

2) cambios bruscos en la pendiente de curva de BER vs SNR, conocidos como *pisos de error*, que son consecuencia de un cambio repentino en los patrones de error dominantes.

En las próximas dos secciones analizaremos ambos efectos en más detalle. A continuación mencionaremos brevemente los métodos clásicos utilizados para analizar el desempeño de códigos LDPC.

- **Density Evolution** (DE) propuesto en [135] es un algoritmo que permite calcular el máximo desempeño posible que puede tener una determinada familia de códigos LDPC cuando la longitud del código y el número de iteraciones tienden a infinito.

- **EXIT Chart** o *gráfica de transferencia de información extrínseca* propuesta originalmente en [156] para códigos turbo y luego aplicado en códigos LDPC [157], es un método que permite analizar y estimar el desempeño de los códigos iterativos. Al igual que density evolution, este método no es exacto ya que ignora la dependencia estadística de los mensajes.

- **Importance Sampling** (IS) es una técnica de estimación mediante simulación que permite reducir la varianza del estimador superando al método clásico de *Monte-Carlo* [145]. Lo anterior permite reducir el tiempo de simulación y de esta forma poder estimar desempeños a una tasa de error menor [24] [71] [176].

Las técnicas de *density evolution* y *EXIT chart* no proporcionan una estimación precisa sobre códigos realistas (i.e., longitud menor a 10^5 bits y menos de 50 iteraciones) al no ser válida la hipótesis de independencia entre los mensajes del algoritmo SPA. En particular, estas técnicas no pueden capturar los dos efectos previamente mencionados. Por otro lado, la técnica de *importance sampling* si bien puede capturar ambos efectos tampoco resuelve el problema por sí misma ya que para garantizar su buen desempeño se necesita información adicional sobre la estructura de los patrones de error dominantes, información que es difícil de obtener en la mayoría de los casos. En el Capítulo 3 usaremos esta última técnica para estimar el desempeño del

[9] Esta dificultad podría minimizarse si la función de probabilidad conjunta tuviera suficiente estructura como para utilizar una representación basada en una variedad posiblemente no euclidiana con un menor número de dimensiones. Tal camino no ha sido reportado en la literatura analizada.

código LDPC propuesto a un BER$\approx 10^{-15}$. Adicionalmente, otros métodos han sido propuestos como técnicas alternativas a los anteriores. Entre estos últimos se puede citar a [133] en donde se presenta una técnica para estimar el piso de error basándose en el conocimiento de sub-estructuras específicas del grafo factorial que dominan el desempeño a altos valores de SNR. Por otro lado, en [147] y [32] se propone un método basado en análisis de *instanton* también para estimar el desempeño a altos valores de SNR. Desafortunadamente, ninguno de los métodos anteriores brinda una solución completa al problema de la estimación del desempeño.

2.5. Efecto de la Dependencia Entre Mensajes

A continuación compararemos el desempeño real del algoritmo MSA (estimado con el método de Monte-Carlo) con el desempeño teórico que tendría dicho algoritmo si se cumpliera la hipótesis de independencia entre los mensajes que arriban a cada nodo variable y a cada nodo factor (es posible aproximarse asintóticamente a tal hipótesis cuando la longitud del código tiende a infinito). Para esto último se procederá en forma similar a *density evolution* en donde calcularemos la función de densidad de probabilidad de la información a posteriori $L^o_{c_i}$. Para obtener resultados concretos aplicaremos dicho análisis sobre un código simple. En particular, se utilizará un código producto basado en chequeo de paridad simple (TPC/SPC). La elección de dicho código radica en su simplicidad conceptual y la alta regularidad de su construcción.

2.5.1. El Código TPC/SPC

Un código producto (PC) se construye mediante un arreglo multi-dimensional de palabras código derivadas de otro u otros códigos previamente definidos. A los códigos que se utilizan para construir el arreglo multi-dimensional se los denomina códigos base. Los códigos base típicos son: códigos de Hamming, códigos BCH, y códigos de chequeo de paridad simple. Dentro de los códigos TPC, aquellos basados en arreglos bi-dimensionales (denotados como 2D-TPC) resultan de gran interés. Un código 2D-TPC compuesto por dos códigos componentes $\mathfrak{C}_1$ y $\mathfrak{C}_2$ con parámetros (N_1, K_1, D_1, G_1) y (N_2, K_2, D_2, G_2) respectivamente, tiene parámetros $(N, K, D, G) = (N_1N_2, K_1K_2, D_1D_2, G_1 \otimes G_2)$ donde N, K, D, G son la longitud, dimensión, distancia mínima y matriz generadora del código, y $\otimes$ representa producto de Kronecker. Dentro de los 2D-TPC, se centrará la atención en aquellos construidos en base a códigos de chequeo de paridad simple (SPC) de longitud n y dimensión $n-1$, denotados como 2D-TPC/SPC$(n,n-1)^2$ [100, 101, 99]. La Fig. 2.6 muestra el grafo factorial y la estructura de dicho grafo para un código 2D-TPC/SPC$(4,3)^2$. Los parámetros de un código 2D-TPC/SPC$(n,n-1)^2$ son:

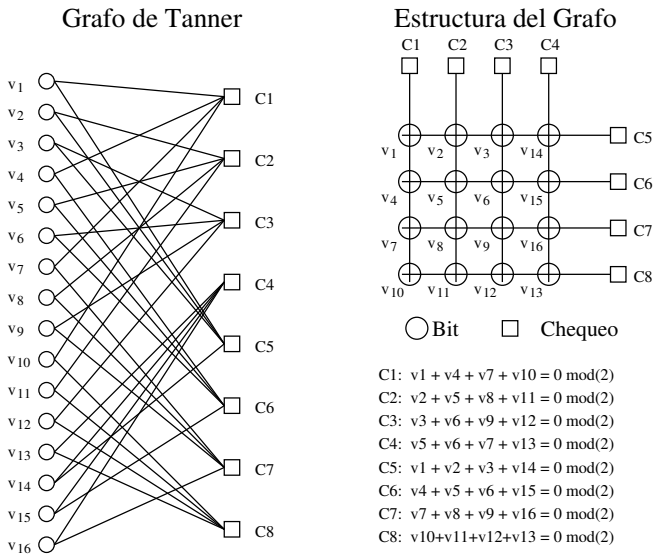


Figura 2.6: Código 2D-TPC/SPC$(4,3)^2$: el grafo de Tanner y su estructura.

Longitud del código	N	$=$	n^2
Dimensión del código	K	$=$	$(n-1)^2$
Tasa del código	R	$=$	$[(n-1)/n]^2$
Distancia mínima	D_{min}	$=$	4
Grado de los nodos de bits	γ	$=$	2
Grado de los nodos de chequeo	ρ	$=$	n
No. de nodos de chequeo		$=$	$2n$

2.5.2. Estimación del BER para una Iteración

Usando la aproximación min-sum del algoritmo SPA con una iteración, el logaritmo de la relación de verosimilitud a posteriori $L^o_{v_i}$ del i-ésimo bit se obtiene mediante la siguiente ecuación:

$$L^o_{v_i} = L^a_{v_i} + \sum_{c_j \in \mathcal{C}_{v_i}} \min_{\hat{v}_i \in \mathcal{V}_{c_j} \setminus \{v_i\}} \left\{ \left| L^a_{\hat{v}_i} \right| \right\} \cdot \prod_{\hat{v}_i \in \mathcal{V}_{c_j} \setminus \{v_i\}} sign(L^a_{\hat{v}_i}). \tag{2.41}$$

Si tomamos como ejemplo el código 2D-TPC/SPC$(4,3)^2$, la ecuación (2.41) con $v_i = v_6$ se reduce a

$$\begin{aligned} L^o_{v_6} \quad = L^a_{v_6} \quad & + \min \left\{ \left| L^a_{v_3} \right|, \left| L^a_{v_9} \right|, \left| L^a_{v_{12}} \right| \right\} \cdot sign(L^a_{v_3} \cdot L^a_{v_9} \cdot L^a_{v_{12}}) \\ & + \min \left\{ \left| L^a_{v_4} \right|, \left| L^a_{v_5} \right|, \left| L^a_{v_{16}} \right| \right\} \cdot sign(L^a_{v_4} \cdot L^a_{v_5} \cdot L^a_{v_{16}}) \end{aligned}$$

La Figura 2.7 muestra el flujo de cálculo correspondiente a dicha ecuación sobre el grafo factorial:

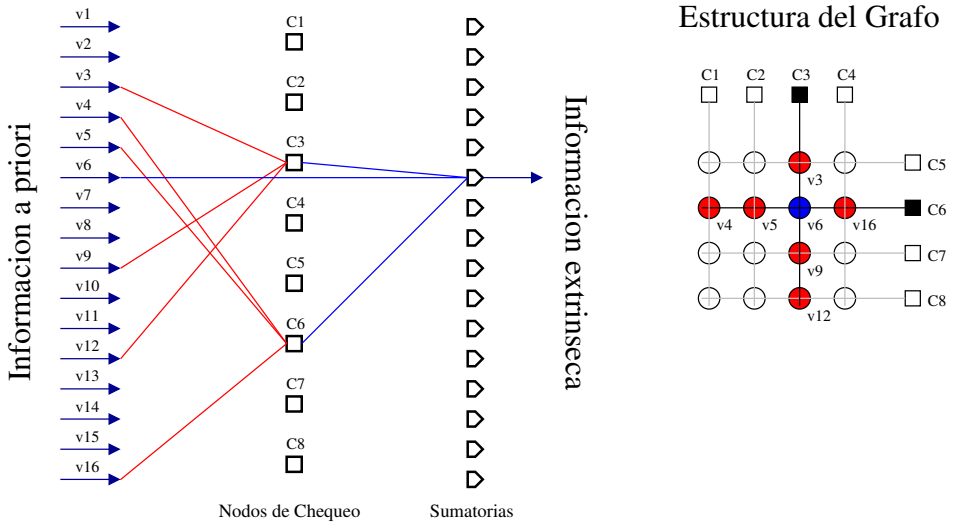


Figura 2.7: Flujo de cálculo para una iteración del algoritmo SPA correspondiente al código 2D-TPC/SPC(4,3)^2.

Si suponemos que el conjunto de información a priori $\{L^a_{v_i} : i = 1, \ldots, N\}$ son variables aleatorias independientes (VAI) (a lo cual denominaremos *hipótesis* H1) entonces para la primera iteración se verifican las propiedades:

P1: $\forall v_i \in \{v_1, v_2, \ldots, v_N\}$, la información extrínseca que arriba al nodo v_i es la suma de dos mensajes $\hat{L}^e_{c_j \to v_i}$ estadísticamente independientes.

P2: $\forall v_i \in \{v_1, v_2, \ldots, v_N\}$, la información extrínseca que arriba al nodo v_i es independiente de la información a priori $L^a_{v_i}$.

La hipótesis H1 (así como P1 y P2) se cumple para cualquier canal sin memoria con ruido blanco aditivo (AWN). La propiedades P1 y P2 permiten calcular la función de densidad de probabilidad (pdf) de la información a posteriori L^o a partir de la función de densidad de probabilidad de la información a priori L^a.

Para realizar dicho cálculo utilizaremos el siguiente hecho. Si

$$z = \text{mín}\{|x|, |y|\} \cdot sign\{x \cdot y\}$$

donde x, y son VAIs. Entonces

$$\begin{aligned} f_z(z) = f_x(z) \cdot [1 - F_y(|z|)] + f_x(-z) \cdot F_y(-|z|) \\ + \; f_y(z) \cdot [1 - F_x(|z|)] + f_y(-z) \cdot F_x(-|z|) \end{aligned} \tag{2.42}$$

Aplicando recursivamente la ecuación (2.42) podemos obtener la pdf $f_{L^e_{c_j \to v_i}}(x)$ de los mensajes $L^e_{c_j \to v_i}$ a partir de las pdfs $f_{L^a(x)}$ de la información

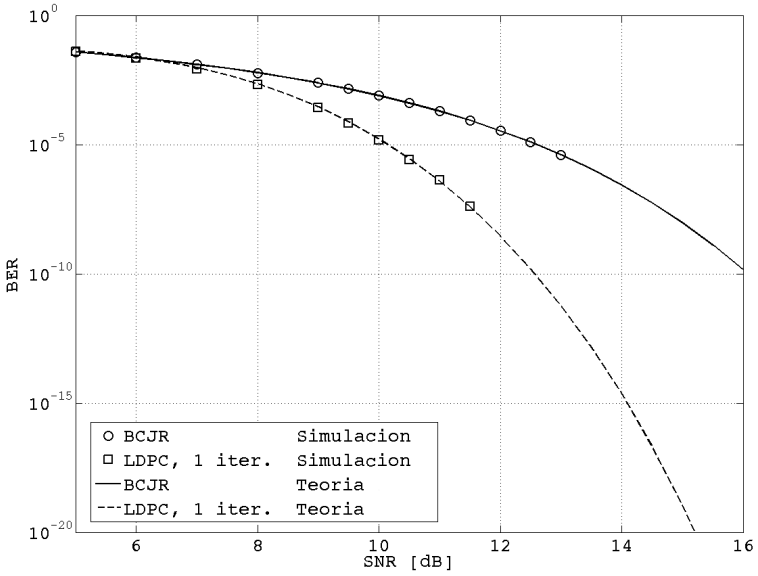


Figura 2.8: Estimación analítica y resultados de simulación del desempeño del código 2D-TPC/SPC(33,32)^2. Decodificación mediante el algoritmo SPA con una iteración. Canal sin memoria con ruido AWGN.

a priori. Luego podemos calcular la pdf $f_{L^o}(x)$ de la información a posteriori mediante la convolución

$$f_{L^o_{v_6}}(x) \quad = \quad f_{L^a_{v_6}}(x) * f_{L^e_{c_3 \to v_6}}(x) * f_{L^e_{c_6 \to v_6}}(x). \tag{2.43}$$

Finalmente, se puede estimar la tasa de error de bits (BER) integrando $f_{L^o_{v_i}}(x)$ en $x \in (-\infty, 0]$,

$$BER_{v_i} = \int_{-\infty}^{0} f_{L^o_{v_i}}(x) \cdot dx, \tag{2.44}$$

donde en esta última integral, se asume sin pérdida de generalidad que se transmitió la palabra nula (i.e., $v_i = 0$ para todo i). Esto último es posible gracias a la linealidad del código.

2.5.3. Resultados de Simulación para una Iteración

El procedimiento analítico descripto fue implementado en MatLab mediante la cuantización de las pdfs involucradas. Dicha representación permitió calcular numéricamente y de forma precisa la pdf de la información a posteriori. Finalmente, la integración numérica de esta última dio como resultado la estimación de desempeño que se buscaba. El código analizado

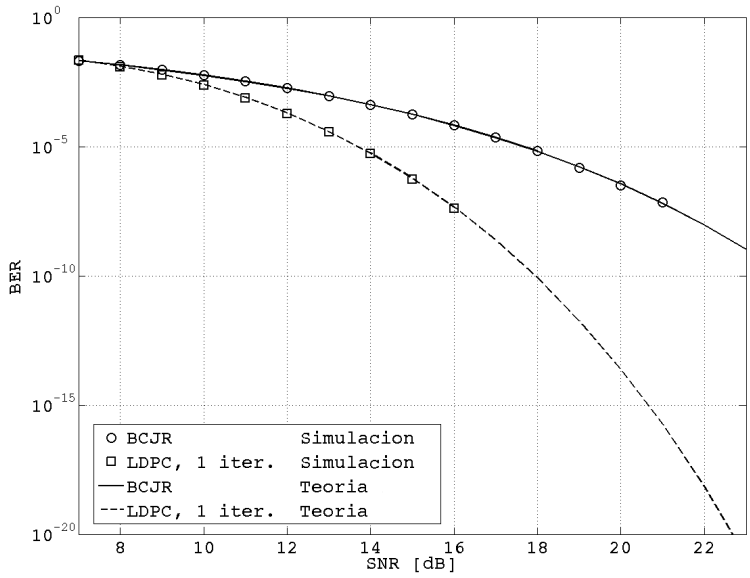


Figura 2.9: Estimación analítica y resultados de simulación del desempeño del código 2D-TPC/SPC(33,32)^2. Decodificación mediante el algoritmo SPA con una iteración. Canal sin memoria con ruido AWLN.

fue el 2D-TPC/SPC(33,32)^2. Se obtuvo además el desempeño del dicho código mediante simulación de Monte Carlo. La decodificación se realizó mediante el algoritmo MSA con una iteración. Se analizaron canales de transmisión sin memoria con ruido blanco aditivo de distribución Gaussiana (AWGN) y Laplaciana (AWLN). Las funciones de densidad de probabilidad Gaussiana $f_{N_G}(x|\mu,\sigma)$ y Laplaciana $f_{N_L}(x|\mu,\sigma)$ utilizadas se describen en las ecuaciones (2.45) y (2.46) respectivamente:

$$f_{N_G}(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.45}$$

$$f_{N_L}(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2}} e^{-\frac{|x-\mu|}{\sigma/\sqrt{2}}} \tag{2.46}$$

Los resultados de simulación y las estimaciones analíticas para el canal AWGN y AWLN se muestran en las Figuras 2.8 y 2.9 respectivamente. Una rápida inspección visual muestra la exacta correspondencia entre los resultados analíticos de simulación.

2.5.4. Estimación del BER para dos o más Iteraciones

El método analítico para el cálculo del BER utilizado en el caso de una iteración puede utilizase como método aproximado para el caso de dos o

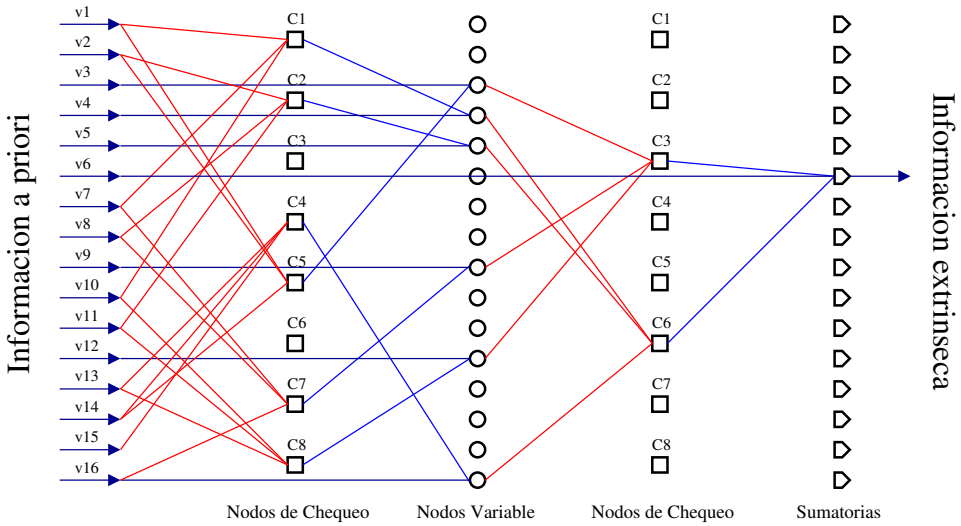


Figura 2.10: Flujo de cálculo para dos iteraciones del algoritmo MSA correspondiente al código 2D-TPC/SPC(4,3)^2.

más iteraciones. La Figura 2.10 muestra el flujo de cálculo sobre el grafo factorial del código 2D-TPC/SPC(4,3)^2 cuando se utilizan dos iteraciones. En dicha figura se observa que la propiedad P1 ya no es válida mientras que la propiedad P2 se mantiene. Sin embargo, esta última propiedad también pierde su validez para más iteraciones. Como veremos a continuación, la pérdida de estas propiedades trae aparejado una reducción en el desempeño.

2.5.5. Resultados de Simulación para dos o más Iteraciones

En la Figuras 2.11 y 2.12 se muestran los resultados numéricos obtenidos mediante simulación de Monte Carlo para el desempeño del código 2D-TPC/SPC(33,32)^2. En las mismas figuras se muestran además las estimaciones analíticas utilizando el método descripto para el caso de una iteración. Se analizan los canales AWGN y AWLN para los casos de 1, 2 y 3 iteraciones SPA. Los resultados muestran que la estimación analítica se aleja de los resultados de simulación a medida que la SNR aumenta cuando se utilizan 2 o más iteraciones. Dicha diferencia se hace más notoria cuando el número de iteraciones es mayor.

2.6. El Problema del Piso de Error

Además del efecto de degradación observado en la sección anterior, la dependencia en los mensajes del algoritmo SPA tiene otro efecto similar pero aún más nocivo denominado *piso de error*, el cual analizaremos a continuación. La curva de BER vs. SNR de códigos iterativos y en particular

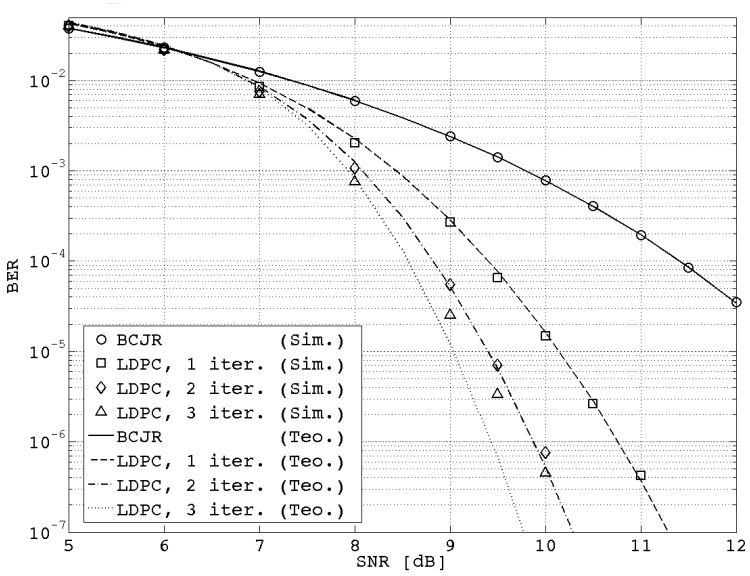


Figura 2.11: Estimación analítica y resultados de simulación del desempeño del código 2D-TPC/SPC(33,32)^2. Decodificación mediante el algoritmo MSA con 1, 2 y 3 iteraciones. Canal sin memoria con ruido AWGN.

de códigos LDPC se puede dividir en dos regiones denominadas *región de cascada* y *región de piso de error*[10]. La región de cascada ocurre a valores de SNR bajos o moderados en donde la curva cae rápidamente. Es común que en esta región la curva incremente monótonamente su pendiente. La región de piso de error se encuentra a continuación de la región de cascada extendiéndose hacia la zona de alta SNR. El punto de inicio de dicha región se encuentra alrededor de la zona en donde la curva reduce su pendiente. La Figura 2.13 muestra la curva de BER vs. SNR del código LDPC[2640,1320] de Margulis[11] utilizando diferentes resoluciones para la representación en aritmética finita de los mensajes del decodificador. En dicha figura se observa claramente las dos regiones mencionadas en el caso de usar una implementación de punto fijo para el decodificador (ver curvas marcadas con triángulos). En el caso de usar una implementación de punto flotante, la región de piso de error se encuentra a un BER inferior al que es posible llegar por simulación. Como una segunda observación, lo anterior

[10] Más conocidas por sus nombres en inglés como región de *waterfall* y de *error-floor*.

[11] Se utilizó como ejemplo el código de Margulis por dos razones: (*i*) dicho código presenta un piso de error a un BER alto, el cual es posible estimar mediante simulación y (*ii*) es un código ampliamente analizado en la bibliografía y comúnmente usado como referencia para el análisis del desempeño del algoritmo SPA.

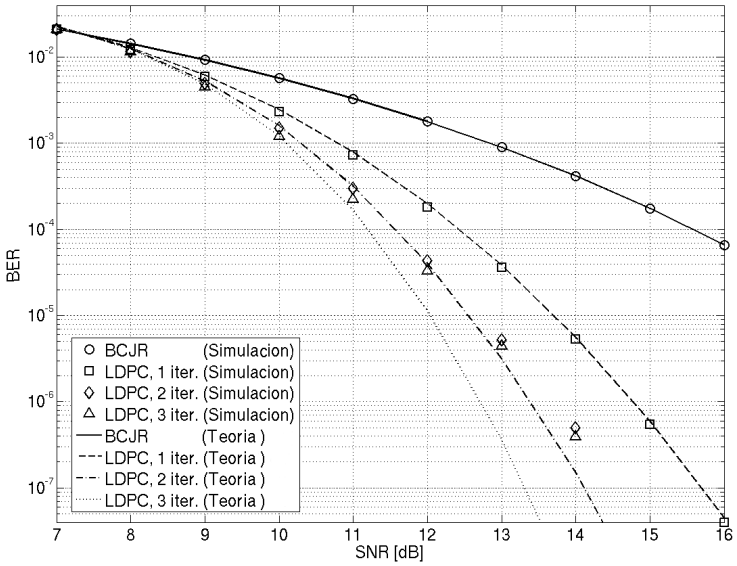


Figura 2.12: Estimación analítica y resultados de simulación del desempeño del código 2D-TPC/SPC(33,32)^2. Decodificación mediante el algoritmo MSA con 1, 2 y 3 iteraciones. Canal sin memoria con ruido AWLN.

nos adelanta el efecto que produce la aritmética finita sobre el piso de error, efecto que luego analizaremos en mayor detalle.

El piso de error representa un problema muy serio para las aplicaciones que requieran un bajo BER. Por ejemplo, un BER en el orden de 10^{-15} es un requerimiento típico en aplicaciones de transmisión por fibras ópticas y en dispositivos de almacenamiento de información. En estas aplicaciones no es posible estimar el desempeño mediante simulación por computadora ya que dicha simulación podría tomar años en las computadoras actuales. A ésto se le suma la ausencia de un método analítico para estimar el desempeño de códigos LDPC, particularmente para estimar el desempeño a BER bajos. Por estas razones, en las últimas dos décadas se ha realizado un gran esfuerzo para determinar la causa del piso de error e intentar eliminarlo. Estas investigaciones han mostrado que para el canal binario con borrado[12], el piso de error de un código LDPC se origina por pequeñas sub-estructuras del grafo factorial conocidas como *stopping set* (SS) [41]. Una caracterización de dicha estructuras que resulta útil para la estimación de desempeño se puede encontrar en [126]. Sin embargo, en esta Tesis nos enfocaremos en el canal AWGN en donde la situación es un poco más compleja que en el canal BEC. El piso de error en canales con ruido AWGN

[12] Más conocido por su nombre en inglés como *binary erasure channel* (BEC)

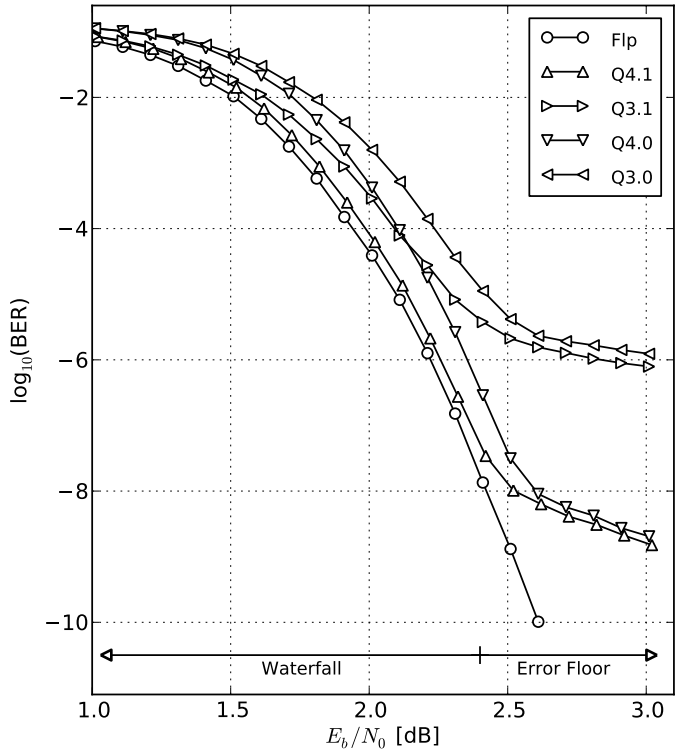


Figura 2.13: Curva de BER vs. SNR del código de Margulis cuando se utiliza el algoritmo de decodificación SMSA con resolución de punto flotante y punto fijo.

está dominado por diferentes estructuras denominadas: *near-codewords* [111], *trapping sets* [133], y *absorbing sets* [45]. Desafortunadamente, enumerar dichas estructuras y estimar su impacto en el desempeño ha mostrado ser un problema difícil de resolver. Aún para un canal tan simple como el AWGN, el problema de estimación del piso de error no está del todo resuelto.

2.6.1. Patrones de Error

Extendiendo la notación introducida en la Sección 2.2, para $V \subseteq \mathcal{V}$ y $C \subseteq \mathcal{C}$, los conjuntos de vecinos de V y C se denotan $\mathcal{C}_V = \bigcup_{v \in V} \mathcal{C}_v$ y $\mathcal{V}_C = \bigcup_{c \in C} \mathcal{V}_c$, respectivamente. Sean $\mathcal{V}^0 \subseteq \mathcal{V}$ y $\mathcal{V}^1 \subseteq \mathcal{V}$ los conjuntos de nodos variable sin errores y con errores respectivamente, donde $\mathcal{V}^0 \cup \mathcal{V}^1 = \mathcal{V}$ y $\mathcal{V}^0 \cap \mathcal{V}^1 = \emptyset$. Un nodo de chequeo $c \in \mathcal{C}$ se dice que está *satisfecho* si la decisión dura de los mensajes que llevan desde sus nodos variables vecinos $v \in \mathcal{V}_c$ satisfacen la ecuación de paridad asociada a c; de lo contrario, se dice que c esta *insatisfecho*. Luego, para $v \in \mathcal{V}$ y $x \in \{0,1\}$, sea $\mathcal{C}_v^x = \{c \in \mathcal{C}_v \ : \ |\mathcal{V}_c \cap \mathcal{V}^1| \equiv x \ mod(2)\}$; i.e., $\mathcal{C}_v^0$ y $\mathcal{C}_v^1$ son los conjuntos

de nodos de chequeo satisfechos e insatisfechos, respectivamente, conectados a v. Finalmente, para $V \subset \mathcal{V}$ y $x \in \{0,1\}$, $\mathcal{C}_V^x = \bigcup_{v \in V} \mathcal{C}_v^x$.

La estructura de los patrones de error sobre un canal AWGN fue capturada en los trabajos [45][97][111][133] bajo las nociones de *near-codewords*, *trapping-sets* y *absorbing-sets*, cuya definición se presenta a continuación.

Definición 2.6.1 (*near-codeword*)**.** Un (a,b) *near-codeword* es un conjunto de nodos variables $V \in \mathcal{V}$ tales que $|V| = a$ y $|\mathcal{C}_V^1| = b$ (i.e., b es el peso del síndrome). En particular, si $b = 0$ entonces se trata de una palabra código [111].

Definición 2.6.2 (*trapping-set*)**.** Un (a,b) *trapping-set* es un (a,b) *near-codeword* que es propenso a errores para un determinado algoritmo de decodificación. Por ejemplo, si se utiliza un decodificador por máxima verosimilitud, sólo los $(a,0)$ *near-codewords*, i.e., las palabras códigos, pueden ser *trapping-sets* [133].

Definición 2.6.3 (*elementary trapping-set*)**.** Un (a,b) *elementary trapping-set* es un (a,b) *trapping-set* en el cual $|\mathcal{V}_c \cap V| = 1$ para todo $c \in \mathcal{C}_V^1$ y $|\mathcal{V}_c \cap V| = 2$ para todo $c \in \mathcal{C}_V^0$ [97]. Los b nodos de chequeo $\mathcal{C}_V^1$ insatisfechos están conectados a exactamente uno de los a nodos variables V con error y el resto de los nodos de chequeo satisfechos $\mathcal{C}_V^0$ están conectados a exactamente 2 de los a nodos variables V con error.

Definición 2.6.4 (*absorbing-set*)**.** Un (a,b) *absorbing-set* es un (a,b) *near-codeword* tal que para cada nodo variable $v \in V$ se cumple que $|\mathcal{C}_v^1| < |\mathcal{C}_v^0|$ [45]. Esto es, cada nodo variable $v \in V$ tiene más nodos de chequeo satisfechos que nodos insatisfechos conectados a él.

Definición 2.6.5 (*fully absorbing-set*)**.** Un (a,b) *fully absorbing-set* es un (a,b) *absorbing-set* tal que la condición $|\mathcal{C}_v^1| < |\mathcal{C}_v^0|$ se cumple para todo nodo variable $v \in \mathcal{V}$ y no sólo para $v \in V$ [45].

De las definiciones anteriores se observa que el concepto de *near-codeword* es genérico y no implica necesariamente un patrón de error dominante, sin embargo es útil para definir a estos últimos. El concepto de *trapping-set* depende del decodificador en cuestión; en este sentido decimos que es una definición a posteriori (i.e., un *trapping-set* es un *near-codeword* para el cual sabemos que es un error dominante para un decodificador determinado). El concepto de *elementary trapping-set* impone mayor estructura al patrón de error. Tal concepto fue propuesto en [97] y estuvo motivado por la gran frecuencia con que se observó dicha estructura en los *trapping-set* capturados mediante simulación. Por último, los conceptos de *absorbing-set* y *fully absorbing-set* sólo dependen de la matriz de paridad y son independientes del decodificador en cuestión, por tal motivo decimos que son conceptos a priori.

Esto hace que el concepto de *absorbing-set* tenga una aplicación directa a la hora de caracterizar la matriz de paridad de un código.

La justificación detrás de las definiciones de los patrones de error anteriores se puede capturar utilizando un modelo simplificado para el comportamiento del SPA, el cual se aproxima a lo que ocurre en un decodificador de *bit-flipping* [103]. En dicho modelo se asume que los mensajes que llegan a un nodo variable $v \in \mathcal{V}$ desde nodos de chequeo satisfechos $\mathcal{C}_v^0$ van a reforzar el estado actual del nodo variable en cuestión, mientras que los mensajes que llegan desde nodos de chequeo insatisfechos $\mathcal{C}_v^1$ van a tener el efecto contrario[13]. Luego, son más propensos a no ser corregidos por el SPA todos aquellos patrones de error compuestos de nodos variables para los cuales los mensajes que le llegan desde nodos de chequeo falsamente satisfechos son más fuertes que los que llegan de los nodos de chequeo insatisfechos. Asumiendo que todos los mensajes tienen igual amplitud, lo anterior implica que un patrón de error va a ser problemático si cada uno de sus nodos variables está conectado a más nodos de chequeo falsamente satisfechos que a nodos de chequeo insatisfechos, es decir, que el patrón es un *absorbing-set*. Adicionalmente, si exigimos lo anterior para todos los nodos variables con el fin de evitar que un bit correcto se vuelva incorrecto con el correr de las iteraciones (cambiando de esta forma la estructura del patrón de error), se obtiene un *fully absorbing-set*[14]. Como corolario tenemos que si el patrón no es un *fully absorbing-set* entonces podría no ser estable. Tal inestabilidad se manifiesta como (i) una secuencia transitoria de mutaciones del patrón de error hasta converger a un patrón estable, (ii) una secuencia de mutaciones periódica o (iii) una secuencia de mutaciones no periódica y no convergente[15]. Afortunadamente, en la práctica los errores están usualmente dominados por patrones de error estables.

En un canal AWGN, mientras mayor es la cantidad de errores, menor es su probabilidad de ocurrencia. Consecuentemente, los (a, b) *near-codewords* dominantes serán aquellos que posean una cantidad de errores a pequeña. Otro factor que afecta la probabilidad de ocurrencia de un patrón de error es el número de ecuaciones de paridad insatisfechas b. Mientras menor sea este último, menor será la capacidad del decodificador de corregirlo. Finalmente, su probabilidad también está afectada por su multiplicidad. Un

[13] Notar que ésto es sólo una aproximación al comportamiento real del SPA ya que el estado de un nodo variable está determinado por la decisión dura de su correspondiente probabilidad a-posteriori, la cual puede no ser igual a la decisión dura de la información extrínseca enviada a sus nodos de chequeo vecinos.

[14] Notar que los *fully absorbing-set* son precisamente aquellos patrones de error que permanecen estables cuando se utiliza un decodificador de bit-flipping [103].

[15] Hay que aclarar que en general cualquier implementación del algoritmo SPA basada en aritmética finita va a tener un número finito aunque inmenso de estados, por consiguiente todo comportamiento va a ser período siendo el período más largo igual al número de estados del algoritmo SPA. Sin embargo, tal periodicidad escapa a los fines prácticos ya que el número de iteraciones es mucho menor que el número de estados.

caso límite ocurre cuando $b = 0$, es decir que el patrón de error es en sí mismo una palabra código. En esta última situación, independientemente del decodificador, no es posible corregir los errores, porque las palabras códigos de mínima distancia son los patrones de error dominantes. Lo anterior presenta la situación típica que tiene lugar en códigos clásicos y en códigos convolucionales turbo. Sin embargo, en códigos LDPC es usual que los *near-codewords* dominantes no sean palabras códigos. El hecho de que el SPA pueda quedar atrapado en (a, b) *near-codewords* con $b > 0$, abre la posibilidad a la existencia de patrones con un número de errores a menor a la distancia mínima del código (o con mayor multiplicidad) trayendo aparejado un degradación en el desempeño. Si la multiplicidad de dichos patrones de error es baja, entonces el efecto sólo va a ser apreciable a bajo BER generando lo que se conoce como piso de error.

Como es de esperar, la sub-estructura detrás de los patrones de error mencionados son los ciclos en el grafo. Por lo tanto los *near-codewords* dominantes serán aquellos que resultan de la combinación de múltiples ciclos cortos.

2.6.2. Efecto de la Aritmética Finita en el Piso de Error

En implementaciones prácticas del algoritmo de decodificación, valores reales no pueden ser utilizados para representar los mensajes y en su lugar se utiliza aritmética de precisión finita. Esto último genera efectos de saturación y cuantización que pueden afectar el comportamiento del algoritmo [118, 186, 181]. Tales representaciones numéricas pueden dividirse en *cuantización uniforme* (CU) y *cuantización no uniforme* (CNU). En CU el paso de cuantización es constante mientras que en CNU no lo es. Adicionalmente, la cuantización puede ser adaptiva o fija (no-adaptiva) dependiendo de si ésta cambia con el tiempo o no. La implementación clásica para el decodificador del códigos LDPC está basada en CU no adaptiva. En particular, se utiliza CU de punto fijo, es decir que se dispone de un número finito de dígitos antes y después de la coma. La ventaja de dicho formato es la baja complejidad de implementación de las operaciones aritméticas (suma, resta y multiplicación), de las operaciones lógicas (comparación, etc) y de las operaciones de redondeo y saturación. Por otro lado, el uso de CNU disminuye el efecto de saturación y cuantización a costa de una mayor complejidad de implementación. Para la cuantización de punto fijo utilizaremos en lo sucesivo la notación $\mathrm{Q}m.f$ para representar la resolución de un número signado de punto fijo. Si $[b_{m-1}, b_{m-2}, \ldots, b_0, b_{-1}, \ldots, b_{-f}] \in \{0,1\}^{m+f}$ es la representación binaria del número x con precisión $\mathrm{Q}m.f$ en lo que se denomina *formato modulo-2*, entonces el valor de x se puede calcular como $x = -b_{m-1} \cdot 2^{m-1} + \sum_{i=-f}^{m-2} b_i \cdot 2^i$, donde el paso de cuantización es 2^{-f} y el rango es $-2^{m-1} \leq x \leq 2^{m-1} - 2^{-f}$. En el Capítulo 4 se analizarán en más detalle este formato y una variante del mismo denotada *modulo-signo*

que será de mayor utilidad para los cálculos en el nodo de chequeo.

Los primeros estudios sobre el efecto de la aritmética finita fueron realizados para la región de cascada de la curva de BER vs SNR. Un gran esfuerzo fue llevado a cabo para optimizar la resolución usada para la cuantización uniforme de punto fijo [187][180][184]. Zhao et al. estudió el efecto de la saturación y la cuantización uniforme optimizando heurísticamente el número de bits y el paso de cuantización para algunos códigos LDPC. Por otro lado, en [180] se propone un esquema de doble cuantización uniforme para aproximar mejor la función $\log(\tanh(x))$ usada en el cálculo de los mensajes $L^e_{c_i \to v_i}$ del algoritmo SPA. Dicha cuantización consiste en usar cuantización $Qm.f$ con $f = m$ (i.e., puramente fraccionaria) para representar valores menores a la unidad y $f = 0$ (i.e., puramente entera) para representar valores mayores a la unidad. Finalmente, en [184] se propone una idea similar para incrementar la precisión en la cuantización de la función $\log(\tanh(x))$. En este último caso, también se utilizó aritmética de punto fijo. Sin embargo las resoluciones utilizadas en los mensajes desde los nodos variables a los nodos de chequeo y viceversa fueron optimizadas en forma independiente. Cabe mencionar que dichos métodos se enfocaron en analizar el efecto de cuantización y no el efecto de saturación. Además, estos métodos se enfocaron en los efectos sobre la región de cascada y no sobre la región de piso de error (pudiéndose observar además que las soluciones propuestas no tenían impacto significativo sobre el piso de error).

El efecto de la aritmética de punto fijo y en particular el efecto de la saturación sobre el piso de error se observa claramente en la Figura 2.13 para el código de Margulis. Tal dependencia con la saturación será analizada en más detalle en el Capítulo 3. En dicho capítulo se propone además el uso de un esquema de post-procesamiento basado en cuantización adaptiva para eliminar la degradación. Tal contribución dió lugar en parte a la publicación [118]. Posteriormente, se realizaron observaciones similares por parte de Zhang y Siegel en [181] confirmando los hallazgos publicados en [118] (véase el Capítulo 3).

2.7. Conclusión

El algoritmo suma-producto proporciona un método eficiente para calcular las funciones marginales de una función global al tomar ventaja de la estructura de factorización de esta última. Tal estructura de factorización puede caracterizarse por completo mediante un grafo bipartito denominado grafo factorial. Si dicho grafo no posee ciclos entonces el cálculo de las funciones marginales es exacto. Por el contrario, si el grafo posee ciclos la convergencia del algoritmo a la solución exacta no está garantiza. Sin embargo, reduciendo la cantidad de ciclos pequeños es posible obtener una excelente aproximación a la solución exacta. En particular, cuando

dicho algoritmo se aplica como método de decodificación de códigos LDPC, posee un excelente desempeño logrando resultados casi óptimos y con un requerimiento computacional moderado. Desafortunadamente, la sub-optimalidad del algoritmo en este último caso genera una pérdida de desempeño a tasas de errores muy bajas, usualmente menores a 10^{-10}. Tal efecto es conocido como piso de error y se manifiesta como un cambio repentino en la pendiente de la curva de BER vs. SNR. Este efecto se genera por la presencia de pequeñas sub-estructuras dentro del grafo factorial que son conocidas como *near-codewords*, *trapping-sets* y *absorbing-sets*. Aún no existe una solución completa a este problema, limitando el uso de los códigos LDPC en aplicaciones que requieren baja tasa de error como ocurre en transmisión por fibras ópticas y en dispositivos de almacenamiento. La presente Tesis intenta aportar una solución en este último sentido.

Capítulo 3

Código LDPC con Bajo Piso de Error

La verdadera grandeza de la ciencia acaba valorándose por su utilidad.

Gregorio Marañón.

Resumen

El presente capítulo describe el diseño de un código LDPC con bajo piso de error, alta ganancia de codificación y una estructura particular de la matriz de paridad que hace posible una arquitectura de implementación eficiente. Se describe el diseño de la matriz de paridad y se propone un nuevo algoritmo de post-procesamiento para reducir el piso de error. Además, se analiza el desempeño del código propuesto mediante simulación en FPGA y estimación semi-analítica basada en *importance sampling*. Los aportes descriptos en el presente capítulo se encuentran publicados en [119] y [118] y también en la patente [51].

3.1. Introducción

El presente Capítulo describe un nuevo código LDPC reconfigurable con alta ganancia de codificación y muy bajo piso de error. En particular, se describe el diseño de la matriz de paridad y un nuevo algoritmo de post-procesamiento. La estructura propuesta para la matriz de paridad no sólo permite reducir el piso de error sino que también posibilita el diseño de una arquitectura reconfigurable de baja complejidad para la implementación del decodificador. Dicha arquitectura será descripta en el Capítulo 4. Por otro lado, el esquema de post-procesamiento propuesto reduce el piso de error en más de tres ordenes de magnitud, superando al esquema de post-procesamiento clásico propuesto en [186] y [184]. Como caso particular, se construyó y analizó un código de 24576 bits de longitud y 20482 bits de dimensión que posee un desempeño a 1.3 dB del límite de Shannon a un BER de 10^{-15}. Cabe remarcar que dicho desempeño no se logra con un decodificador SPA ideal sino con un decodificador SMSA de 5 bits de precisión y un promedio de 8 iteraciones por palabra código. Este decodificador fue modificado con el algoritmo de post-procesamiento propuesto a fin de evitar un piso de error. El desempeño de dicho código fue determinado mediante la combinación de emulación en FPGA con la técnica de *importance sampling*.

3.1.1. Estado el Arte

El código propuesto tiene aplicación directa en las comunicaciones por fibras ópticas y en el almacenamiento de la información debido a que en ambas aplicaciones se requiere un piso de error muy bajo, una alta ganancia de codificación y una arquitectura de implementación para alta velocidad que sea eficiente en términos de área y potencia. Otra área de aplicación para el código propuesto se encuentra en las comunicaciones inalámbricas donde, a pesar de que la velocidad de transmisión es menor debido a las limitaciones del medio inalámbrico basado en ondas de radio[1], se requiere también una alta ganancia de codificación.

Particularmente, como validación de la utilidad del código propuesto en comunicaciones por fibras ópticas, cabe remarcar que dicho código será usado en la próxima generación de enlaces ópticos coherentes a velocidades de 100 Gb/s y 200 Gb/s — véase [96] — construidos por la empresa ClariPhy[2]. En tal implementación, los dos principales esquemas de codificación competidores fueron: (*i*) una triple concatenación de códigos

[1] Actualmente está bajo investigación el área de comunicaciones inalámbricas basadas en enlaces ópticos, en donde la transmisión podría alcanzar velocidades superiores a las logradas utilizando ondas de radio, particularmente para enlaces de corta distancia. En estas aplicaciones el código propuesto podría ser de gran interés.

[2] Véase www.clariphy.com.ar y www.clariphy.com

propuesta por Onohara et al. en el año 2010 en [125] y (*ii*) un código TPC propuesto en el año 2011 por Dave et al. en [38]. Ambos códigos, junto al aquí propuesto, son los únicos códigos publicados que demostraron poseer alta ganancia de codificación y fueron efectivamente implementados en un chip (no FPGA). El código propuesto en [125] posee un *overhead* fijo de ≈20.5 % logrando una ganancia neta de codificación de 10.8 dB a un BER=10^{-15}, la cual es 0.5 dB inferior a la ganancia del código aquí propuesto cuando éste es configurado con un *overhead* de 20 %. Por otro lado, el código propuesto en [38] posee un *overhead* fijo de ≈15 % y alcanza una ganancia neta de codificación de 11.1 dB, igualando a la del código aquí propuesto para el mismo *overhead*. Sin embargo, ambos códigos tienen la desventaja de que no pueden ser configurados para soportar diferentes *overheads*. Tal flexibilidad es necesaria para adaptar la codificación según el estado del canal, optimizando de esta forma el uso de este último[3] [23].

Otros códigos han sido presentados en los últimos dos años que prometen un desempeño similar o superior al del código propuesto. Sin embargo, ninguno de ellos ha demostrado hasta el momento poseer una eficiente arquitectura de implementación, limitándose al análisis de desempeño mediante simulación en computadora o emulación en FPGA. Entre dichos códigos los principales son: un código LDPC propuesto por Chang en el 2012 [26] que con un *overhead* de 20 % alcanza una ganancia neta de codificación de 11.5 dB. Este código fue emulado en FPGA pero su complejidad de implementación aún no ha sido evaluada. En particular, el algoritmo de decodificación utilizado en este último código no posee un *flooding schedule* [90] (véase Sección 2.1.4) lo que dificulta su paralelización. Por otro lado, en [82] se propuso un esquema de códigos concatenados que combina un LDPC con un RS logrando una ganancia neta de codificación de 11.30 dB con un *overhead* de 20.5 %. Este último LDPC es también un código cuasi-cíclico como el aquí propuesto pero cuya longitud (74844 bits) es más de tres veces superior al código aquí propuesto, además de requerir el uso de un código externo para eliminar el piso de error. Más recientemente, en el presente año en [148] se propone un esquema concatenado basado en un LDPC(38400,30832) y un BCH(30832,30592) con un *overhead* total de 25.5 % y una ganancia neta de codificación de 12 dB a un BER de 10^{-15} obtenida mediante extrapolación de resultados de simulación por computadora. Desafortunadamente, la factibilidad de implementación de dicho código aún no ha sido evaluada ni tampoco verificada la ausencia de piso de error. Cabe aclarar que tal ganancia se puede obtener utilizando el código aquí propuesto con el mismo *overhead*. En base a lo anterior, se pone en evidencia que el código aquí propuesto aún no ha sido efectivamente

[3] La adaptación del código al estado del canal ya es de uso estándar en el área de las comunicaciones inalámbricas y está comenzando a ser utilizada también en canales ópticos. Su aplicación en estos últimos estuvo limitada por cuestiones de complejidad de implementación debido a la alta velocidad.

superado ya que para tal fin se debe tener en cuenta no sólo la ganancia neta de codificación estimada sino también la ausencia de piso de error requerida para confirmar esta última. A ésto hay que sumarle la existencia de una arquitectura de implementación que haga viable su aplicación en equipos comerciales. En este sentido cabe destacar que este aspecto práctico constituye uno de los ejes principales de esta Tesis, ya que el desafío de los códigos LDPC no sólo se limita al desempeño sino también a su implementación.

3.2. Diseño de la Matriz de Paridad

Tanto el desempeño como la complejidad de implementación de un código LDPC están fuertemente ligados a la estructura de la matriz de paridad. Es entonces necesario realizar un diseño que tenga en cuenta ambos aspectos. A continuación describiremos la construcción del código combinando aspectos de implementación y de desempeño. La implementación está relacionada con lo que llamaremos estructura de *partición regular por columnas* (PRC) combinada con una estructura cuasi-cíclica (CC), mientras que el desempeño está relacionado con la metodología empleada para la reducción de ciclos de corta longitud.

3.2.1. Matriz con Partición Regular por Columnas

Sea $\mathbf{H}$ la matriz de paridad $(m \times n)$ de un código LDPC. Bajo la condición de que $n = \mu q$ con μ y q números enteros, la matriz $\mathbf{H}$ puede dividirse en μ sub-matrices de dimensión $(m \times q)$ según muestra la ecuación (3.1):

$$\mathbf{H} = \left[\mathbf{H}^{(1)} \cdots \mathbf{H}^{(l)} \cdots \mathbf{H}^{(\mu)}\right]. \tag{3.1}$$

A las sub-matrices $\mathbf{H}^{(l)}$ se les impondrá además la restricción de que los pesos de las filas y de las columnas no cambien en función de l. A tal restricción la denominaremos *Partición Regular por Columnas* PRC. La Figura 3.1 muestra un ejemplo en donde una matriz regular $\mathbf{H}$ de pesos $(4, 12)$ es dividida en $\mu = 6$ sub-matrices regulares $\mathbf{H}^{(l)}$ de pesos $(4, 2)$ (i.e., $q = 2$).

La restricción PRC posibilita la implementación de una arquitectura semiparalela eficiente basada en el algoritmo MSA o sus variantes [120]. En dicha arquitectura cada iteración es dividida en μ pasos. En el l-ésimo paso, sólo se calculan los mensajes relacionados con la sub-matriz $\mathbf{H}^{(l)}$. Lo anterior permite reutilizar en cada paso los mismos q nodos variables, reduciendo de esta forma el hardware asociado por un factor de μ. Por otro lado, la complejidad de interconexión es también reducida ya que la red de interconexión sólo está asociada a los elementos no nulos de la sub-matriz $\mathbf{H}^{(l)}$, la cual es μ veces menor que en la matriz original $\mathbf{H}$. Adicionalmente, el cálculo de los nodos de chequeo se simplifica al poder realizarlos en

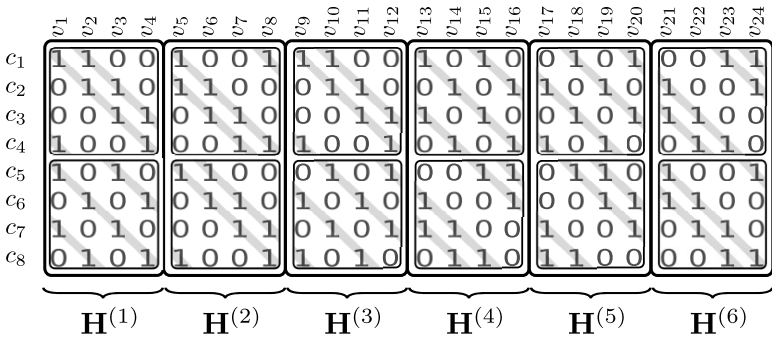


Figura 3.1: Ejemplo de la partición regular por columnas de una matriz de paridad cuasi-cíclica de pesos $(4, 12)$ en $\mu = 6$ sub-matrices regulares de pesos $(4, 2)$.

forma recursiva. Finalmente, gracias a la recursividad del nodo de chequeo, es posible realizar una importante simplificación en los requerimientos de memoria al sólo ser necesario almacenar dos mensajes [29]. Los detalles de tal arquitectura y sus ventajas serán analizados en el Capítulo 4. Véase también las publicaciones [120] y [118] y la patente [51].

3.2.2. Matriz Cuasi-Cíclica

Las matrices cuasi-cíclicas son aquellas construidas como un arreglo rectangular de sub-matrices cuadradas cíclicas del igual tamaño[4] (véase [103] para mayores detalles). Los códigos cuya matriz de paridad es una matriz cuasi-cíclica se denominan códigos cuasi-cíclicos, y representan una de las familias más importantes y generales de códigos [137]. Tal estructura en la matriz de paridad posee la ventaja de simplificar considerablemente la implementación de los algoritmos de codificación y decodificación. Nótese que los códigos cíclicos (entre los cuales se puede mencionar a los códigos BCH y RS) son un caso particular de códigos cuasi-cíclicos. Afortunadamente, las ventajas de la estructura cuasi-cíclica son independientes de la densidad de la matriz, siendo por lo tanto posible construir códigos LDPC cuasi-cíclicos, denotados CC-LDPC, con la única restricción adicional de que las sub-matrices cíclicas sean ralas. Dichos códigos han demostrado poder alcanzar desempeños muy cercanos al límite de Shannon, véase [86] y [137]. Además, la propiedad de ser cuasi-cíclicos ha mostrado ser una ventaja para su implementación [86][137]. Finalmente, la estructura cuasi-ciclica facilita el análisis de los parámetros del código y de la caracterización de sus patrones de error [151][175][137].

[4] Es posible generalizar esta definición a un arreglo de sub-matrices cuadradas cíclicas no necesariamente del mismo tamaño [141]. Dicha generalización tiene ventajas teóricas pero presenta problemas de implementación por lo que no será tratada aquí.

Los códigos LDPC cuasi-cíclicos pueden clasificarse en función del peso máximo δ de sus sub-matrices, i.e., δ es el número máximo de diagonales que tienen sus sub-matrices cíclicas. A dicho código se lo denomina "CC-LDPC tipo-δ". Los códigos cuasi-cíclicos más comunes son los CC-LDPC tipo-1 debido a su simplicidad de análisis. Esto códigos tienen una distancia mínima acotada superiormente por $(\gamma+1)!$ (donde γ es el peso de las columnas de la matriz de paridad) y un *girth* máximo de 12 [53]. Por otro lado, es posible obtener una distancia mínima mayor si se incrementa el peso δ de las sub-matrices [144]. Desafortunadamente, a medida que se incrementa δ disminuye el máximo *girth* del código. Para $\delta = 3$, el *girth* máximo se reduce a 6, lo que trae aparejado una degradación en el desempeño y un posible piso de error. La situación empeora para $\delta > 3$. Una buena relación entre la distancia mínima y el *girth* del código puede lograrse utilizando $\delta = 2$. Por tal motivo, en lo sucesivo se enfocará el diseño en un CC-LDPC tipo-2.

3.2.3. Código CC-PRC-LDPC

Denotaremos como código CC-PRC-LDPC tipo-δ a un código LDPC cuya matriz de paridad es CC tipo-δ y cumple la restricción PRC. La Figura 3.1 muestra un ejemplo de tal matriz para un código de longitud 24 bits. Notar que la regularidad introducida por la restricción PRC hace que todas las sub-matrices tengan peso δ (i.e., son matrices con δ diagonales). Tales códigos contienen siempre ciclos de orden 8. Dicha propiedad es enunciada en la siguiente proposición:

Proposición 3.2.1. *La matriz de paridad de un código CC-PRC-LDPC con $\mu \geq 2$ y $\gamma \geq 4$ contiene siempre ciclos de orden 8.*

Demostración. Sea

$$\mathbf{H} = \begin{bmatrix} H^{(1,1)} & \dots & H^{(1,l)} & \dots & H^{(1,\mu)} \\ H^{(2,1)} & \dots & H^{(2,l)} & \dots & H^{(2,\mu)} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ H^{(\eta,1)} & \dots & H^{(\eta,l)} & \dots & H^{(\eta,\mu)} \end{bmatrix}$$

la matriz de paridad de un código CC-PRC-LDPC donde $\mu \geq 2$, $\eta \geq \frac{1}{2}\gamma = 2$ y $H^{(i,j)}$ son matrices cíclicas no nulas. Sea

$$\mathbf{M} = \begin{bmatrix} H^{(i,j)} & H^{(i,j')} \\ H^{(i',j)} & H^{(i',j')} \end{bmatrix}$$

con $1 \leq i < i' \leq \eta$ y $1 \leq j < j' \leq \mu$. La existencia de un ciclo en $\mathbf{M}$ se traduce en la existencia de un ciclo en $\mathbf{H}$. Para analizar los ciclos de $\mathbf{H}$ haremos uso del siguiente teorema de Buckley y Harary [20] que dice:

Teorema 3.2.2. *Sea* $\mathbf{A}$ *la matriz de adyacencia del un grafo no dirigido, entonces: (i) el elemento (i,j) de* $\mathbf{A}^l$ *es igual al número de caminos de longitud* l *desde el nodo* i *al nodo* j *y (ii) el grafo asociado a* $\mathbf{A}$ *tiene girth* ξ *siendo los nodos* i *y* j *dos nodos opuestos dentro de un lazo cerrado de longitud* ξ *si y solo si* $\mathbf{A}_{i,j}^{\frac{1}{2}\xi} \geq 2$ *y* $\mathbf{A}_{i,j}^{\frac{1}{2}\xi-1} = 0$

Luego, $\mathbf{H}$ contiene ciclos de orden 8 si para algún $i, i', j, j' \in \mathbb{N}$ con $1 \leq i < i' \leq \eta$ y $1 \leq j < j' \leq \mu$, la matriz $\left(\mathbf{M}\mathbf{M}^{\mathrm{T}}\right)^2$ contiene algún elemento mayor a 1. Donde $\left(\mathbf{M}\mathbf{M}^{\mathrm{T}}\right)^2$ se puede descomponer como

$$\left(\mathbf{M}\mathbf{M}^{\mathrm{T}}\right)^2 = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

con

$$A = \left(H^{(i,j)}\right)^4 + \left(H^{(i,j')}\right)^4 + \left(H^{(i,j)}H^{(i',j)}\right)^2 + \left(H^{(i,j')}H^{(i',j')}\right)^2 + 2\left(H^{(i,j)}H^{(i,j')}\right)^2 + 2H^{(i,j)}H^{(i,j')}H^{(i',j)}H^{(i',j')}$$

donde B, C y D son ciertas matrices (explicitas, que no necesitamos). El factor 2 en la suma anterior hace que se cumpla la condición requerida y por consiguiente $\mathbf{H}$ contendrá siempre ciclos de orden 8. □

Otro parámetro de interés en el diseño de códigos LDPC regulares es γ (el peso de las columnas). Gallager demuestra en [55] que es necesario que $\gamma \geq 3$ para que el código pueda tener una distancia mínima que crezca linealmente con la longitud n, mientras que para $\gamma = 2$ la distancia mínima crece en el mejor de los casos en forma logarítmica con n. Por otro lado, no es conveniente utilizar un γ mucho mayor a 3 por limitaciones de complejidad ya que para n fijo esta última crece (al menos) linealmente con γ. Finalmente, notar que restricciones CC y PRC obligan a utilizar γ par, por lo que en lo sucesivo se opta por $\gamma = 4$ para la construcción del código CC-PRC-LDPC tipo-2. La matriz de paridad del tal código queda entonces compuesta de un arreglo de $2 \times \mu$ sub-matrices bi-cíclicas. Los parámetros del dicho código son:

- Longitud $n = h \cdot \mu$
- Dimensión $k = n - 2 \cdot (h-1) = h \cdot (\mu - 2) + 2$
- Tasa de codificación $r = \frac{k}{n} = \frac{(\mu-2)+2/h}{\mu}$
- Overhead $OH = \frac{n-k}{k} = \frac{2\cdot(h-1)}{h\cdot(\mu-2)+2}$

donde h representa la cantidad de filas o columnas de las sub-matrices cíclicas. El valor de h dependerá de la ganancia neta de codificación que se

desee. El objetivo planteado en el diseño aquí propuesto será una ganancia de 11.3 dB a un BER de 10^{-15} cuando el *overhead* es de 20 % con la finalidad de igualar el desempeño del código propuesto en [82]. Para lograr tal ganancia se deduce fácilmente mediante simulaciones de Monte-Carlo que es necesario un tamaño de sub-matriz h de 2048 bits. El código resultante tiene una longitud de $12 \cdot 2048 = 24576$ bits (con $\mu = 12$) para un *overhead* de 20 %. Notar que tal código es 3 veces más chico que el código propuesto en [82] y, como veremos más adelante, posee un piso de error muy inferior al de este último. Por otro lado, la dimensión del código será $k = n - rank(\mathbf{H}) = 20482$ bits que por razones de implementación será reducida en 2 bits (i.e., 20480).

3.2.4. Optimización de Ciclos

En base a lo anterior, la matriz de paridad del código resultante está construida como un arreglo de 2×12 sub-matrices bi-cíclicas de dimensiones 2048×2048. Sin embargo, para una completa descripción del código aún resta determinar la ubicación de las dos diagonales en cada sub-matriz. Lo anterior representa un amplio grado de libertad que será usado para reducir la cantidad de ciclos de corta longitud. A continuación se describe el método utilizado para reducir tales ciclos, el cual se compone de las siguiente etapas:

1. Generar una matriz CC-PRC $\mathbf{H}$ inicial. Dicha matriz posee la estructura CC-PRC mencionada en donde las posiciones de las diagonales han sido generadas aleatoriamente.

2. Calcular el vector $\phi = [\phi_4, \phi_6, \ldots, \phi_{g+2}]$ de enumeración de ciclos de $\mathbf{H}$ donde ϕ_i es la cantidad de ciclos de longitud i que tiene la matriz $\mathbf{H}$ y g es el máximo *grith* que puede tener $\mathbf{H}$.

3. Crear una copia de $\mathbf{H}$ denotada $\hat{\mathbf{H}}$.

4. Modificar $\hat{\mathbf{H}}$ manteniendo la estructura CC-PRC.

5. Calcular el vector $\hat{\phi} = [\hat{\phi}_4, \hat{\phi}_6, \ldots, \hat{\phi}_{g+2}]$ de enumeración de ciclos de $\hat{\mathbf{H}}$.

6. Calcular el vector $\rho = \phi - \hat{\phi}$ y reemplazar $\mathbf{H}$ con $\hat{\mathbf{H}}$ si el primer elemento distinto de cero es positivo[5]

7. Determinar si es necesario continuar el proceso de optimización. En caso afirmativo se retorna al paso 3, de lo contrario se obtiene como resultado la matriz $\mathbf{H}$.

El algoritmo anterior puede interpretarse como una caminata aleatoria dentro del espacio de parámetros que se pretenden optimizar, en donde cada

[5]Se realiza una comparación de la cantidad de ciclos en orden de longitud creciente

paso de la caminata se realiza sólo si se satisface el criterio de optimización. Las dos etapas principales de dicho proceso son entonces la generación del paso aleatorio y el criterio de optimización dado por el cálculo del vector de enumeración de ciclos. El paso aleatorio generado en la etapa 4 puede realizarse de la siguiente forma: se elige mediante un algoritmo pseudo-aleatorio (ASA) una sub-matriz cíclica y se cambia también pseudo-aleatoriamente la posición de una de sus diagonales. Este procedimiento puede repetirse varias veces como parte de la etapa 4. Por otro lado, se propone realizar el cálculo del vector de enumeración de ciclos en base al teorema 3.2.2 de la siguiente forma. Debido a que el GT de $\mathbf{H}$ es bipartito, éste contiene sólo ciclos de longitud par (i.e., ξ par). Luego, hay ξ pares de nodos opuestos en cada ciclo de longitud ξ, i.e., hay ξ entradas en la matriz $A_{i,j}^{\xi/2}$ que verifican en teorema 3.2.2 y corresponden al mismo ciclo. Por otro lado, cada subconjunto de dos de los $A_{i,j}^{\xi/2}$ caminos que conectan i con j corresponden a un ciclo diferente, i.e., hay $\Phi(i,j) = \frac{1}{2}A_{i,j}^{\xi/2}(A_{i,j}^{\xi/2} - 1)$ ciclos diferentes que contienen los nodos i y j como nodos opuestos. Entonces el número N de ciclos es

$$N = \frac{1}{\xi}\sum_{i,j} I(i,j)\cdot\Phi(i,j), \tag{3.2}$$

donde $I(i,j) \in \{0,1\}$ vale 1 si se verifica el teorema 3.2.2 para la entrada (i,j) de la matriz de adyacencia, y 0 en caso contrario. La velocidad de cálculo de la ecuación (3.2) puede acelerarse al tomar ventaja del isomorfismo entre las matrices cíclicas y el anillo de polinomios $\mathbb{Z}[x]/(x^L-1)$ [175]. De esta forma, tanto $\mathbf{H}$ como la matriz de incidencia A asociada al GT de $\mathbf{H}$ se pueden representar como un arreglo de polinomios sobre $\mathbb{Z}[x]/(x^L-1)$ en lugar de un arreglo de matrices cíclicas, reduciendo significativamente los cálculos.

Finalmente, se destaca que el algoritmo propuesto realiza una optimización a nivel de ciclos y no a nivel de *absorbing-sets* como sería preferible. Desafortunadamente, en la bibliografía consultada, no existe un método eficiente para calcular la cantidad de cada *absorbing-set* presente en la matriz $\mathbf{H}$. Sin embargo, con la finalidad de orientar la optimización a la reducción de *absorbing-sets*, una posible modificación se basa en aprovechar que estos últimos son estructuras creadas por la combinación de varios ciclos entrelazados. Luego es preferible eliminar ciclos entrelazados antes que los ciclos aislados, incluso si tal criterio no minimiza la cantidad de ciclos totales. Con tal fin se puede añadir un coeficiente exponencial a $\Phi(i,j)$ en la ecuación (3.2), i.e., $[\Phi(i,j)]^{\omega}$, con $\omega > 1$ o de forma más general es posible reemplazar $\Phi(i,j)$ por $f(\Phi(i,j))$ donde $f(\cdot)$ es una función no-decreciente.

3.2.5. Nuevos LDPC

Se construyeron dos códigos PRC-QC-LDPC basados en un arreglo de 2×12 sub-matrices bi-cíclicas de dimensiones 2048×2048. El primer código, denotado $\mathcal{C}_1$, se optimizó para eliminar únicamente los ciclos de longitud 4. El segundo código, denotado $\mathcal{C}_2$, se diseñó para (i) eliminar los ciclos de longitud 4 y 6, lo que equivale a alcanzar el máximo *girth* posible para este tipo de código, y (ii) minimizar el número de ciclos de longitud 8. El código $\mathcal{C}_1$ fue construido para disponer de un código con un alto piso de error que pueda observarse mediante simulación de Monte-Carlo. Esto último proporciona una referencia para verificar la metodología de estimación del BER basada en *importance sampling* (véase Figura 3.3) y evaluar los algoritmos de post-procesamiento (véase Figura 3.7). Por otro lado, el código $\mathcal{C}_2$ fue construido para lograr una alta ganancia y un piso de error por debajo de 10^{-15}. Cabe remarcar que tal logro parecía altamente improbable según el análisis realizado en el año 2010 por Onohara et al. en [125].

3.3. Desempeño

Se utilizó el algoritmo SMSA como decodificador del código PRC-QC-LDPC propuesto. El mismo fue implementado en dos placas de FPGAs a saber:

- Una placa con 8 FPGAs Virtex 5 en donde se logró una velocidad de 4.9 Gb/s, véase Figura 3.2 (izquierda)

- Una placa con 7 FPGAs Virtex 6 en donde se logró una velocidad de 10 Gb/s, véase Figura 3.2 (derecha)

Se utilizó modulación por desplazamiento de fase binario (binary phase-shift keying) BPSK, esto es: los bits 0 y 1 son mapeados a los símbolos +1 y -1 respectivamente para su transmisión. Se utilizó un canal sin memoria con ruido AWGN, el cual fue implementado usando un generador de números pseudo-aleatorios como el descripto en [98]. En las estimaciones de BER se contaron al menos 400 errores. El algoritmo de decodificación SMSA se implementó con una resolución $r_a = 5$ bits para la información a priori $L^a_{v_i}$ y de $r_e = 5$ bits para los mensajes extrínsecos $L^e_{v_i \to c_j}$ y $L^e_{c_j \to v_i}$.

La implementación realizada en la FPGA permite capturar los patrones de error para su posterior análisis. En particular, a partir del conjunto Φ de *absorbing-sets* capturados es posible estimar el desempeño a tasas de error $p_e(\Phi)$ menores de las simuladas en la FPGA como

$$p_e(\Phi) \approx \sum_{\alpha \in \Phi} m(\alpha) \cdot p(\alpha) \tag{3.3}$$

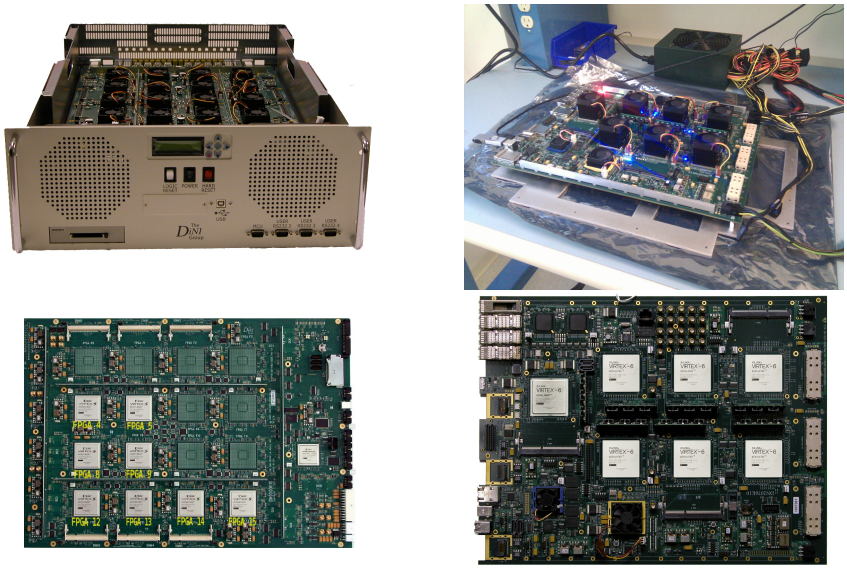


Figura 3.2: Placa de FPGAs Xilinix Virtex-V (izquierda) y Virtex-VI (derecha).

donde $m(\alpha)$ es la multiplicidad del *absorbing-set* α y $p(\alpha)$ es la tasa de error asociada al mismo. La multiplicidad $m(\alpha)$ se estima en base a los *absorbing-sets* capturados y al automorfismo de la estructura cuasi-cíclica del código. Esto es, cada *absorbing-set* pertenece a una familia de $h = 2048$ *absorbing-sets* isomorfos bajo rotaciones cuasi-cíclicas. Por lo tanto, capturando un único *absorbing-set* de cada familia es posible reconstruir la familia entera. Lo anterior permite una mejora en la estimación respecto al método de Monte-Carlo en $\log_{10}(2048) \approx 3{,}3$ ordenes de magnitud. Por último, la probabilidad $p(\alpha)$ se estima utilizando *importance sampling* [24] [71] [176].

La Figura 3.3 muestra la curva de BER vs. SNR del código $\mathcal{C}_1$ con 13 iteraciones. En la misma se observan los resultados obtenidos mediante la simulación en FPGA y la estimación semi-analítica en base a los *absorbing-sets* capturados. Se observa un piso de error a un BER de 10^{-10} que reduce la ganancia neta del código de 11.3 dB a 9.45 dB a BER=10^{-15}. Cabe remarcar también que la estimación semi-analítica basada en la ecuación (3.3) sigue correctamente a los resultados de la simulación por el método de Monte-Carlo. El análisis de los patrones de error capturados indica que este piso de error es causado por varios *absorbing-sets* creados por la combinación de ciclos de longitud 6; recordar que en este código tales ciclos no fueron eliminados con la intención justamente de exacerbar el piso de error.

Por otro lado, la Figura 3.4 muestra el desempeño del código $\mathcal{C}_2$. En la misma no se observa piso de error hasta donde fue posible la simulación, esto es hasta 10^{-13}. Sin embargo, en base a los patrones de error capturados se estimó que a BER menores a 10^{-13} existe un piso de error que depende de

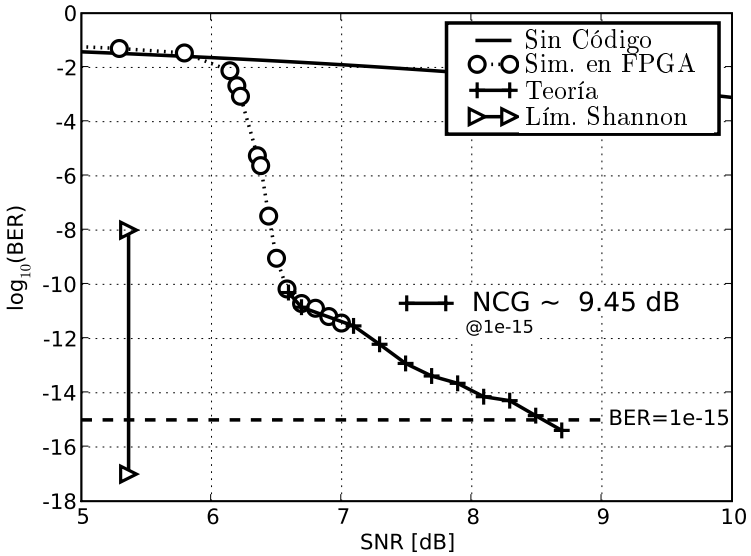


Figura 3.3: BER vs. SNR del código $\mathcal{C}_1$.

la resolución utilizada en la aritmética finita de los mensajes. Este piso de error está generado por combinación de un AS(12,8) y la precisión finita de los mensajes L^e — donde en la notación "AS(e,d)", e es el número de bits con error y d es la cantidad de nodos de chequeo insatisfechos (véase [185] para más detalles) — La Figura 3.4 muestra el piso de error estimado para resoluciones r_e = de 5, 6, y 7 bits con 13 iteraciones. La evolución de L^o para el AS(12,8) se muestra en la Figura 3.5. Se observa que el algoritmo SMSA con $r_e = 5$ bits no puede resolver el AS(12,8), independientemente del número de iteraciones. Por otro lado, si se incrementa la resolución r_e a 6 y 7 bits entonces el algoritmo SMSA puede resolver el AS(12,8) con 17 y 12 iteraciones, respectivamente.

3.4. Mejoras sobre el Algoritmo de Decodificación

Para reducir el piso de error en códigos LDPC se han propuesto varios algoritmos de post-procesamiento, véase [68], [183] y [185]. Los resultados de la simulación de dichos algoritmos mostraron que pueden reducir el piso de error entre 2 y 3 órdenes de magnitud. A continuación se describirá un nuevo algoritmo de post-procesamiento basado en cuantización adaptiva, y se analizará el desempeño de los códigos $\mathcal{C}_1$ y $\mathcal{C}_2$ en combinación con dicho algoritmo. Como referencia se comparará dicho algoritmo con el propuesto en [185].

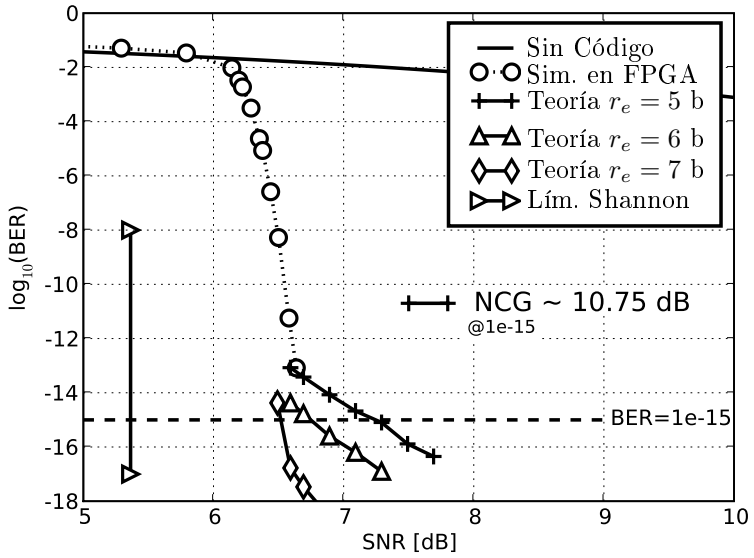


Figura 3.4: BER vs. SNR del código $\mathcal{C}_2$.

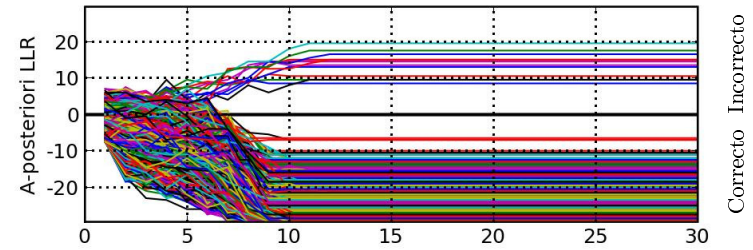
3.4.1. Algoritmo de Post-Procesamiento

En esta sección se describe un algoritmo de post-procesamiento de baja complejidad para combatir el piso de error causado por el uso de aritmética finita en el algoritmo MSA. En este algoritmo, la información a priori L^a y la información extrínseca de los mensajes L^e es ajustada en amplitud en cada iteración con la finalidad de incrementar el rango dinámico reduciendo de esta forma el efecto de saturación que ocurre en el nodo variable. Debido a que el número total de bits usados en L^a y L^e se mantiene constante, la expansión del rango dinámico tiene lugar a costa de un incremento en el paso de cuantización. El proceso de expansión del rango dinámico no ocurre en todas la iteraciones sino que tiene lugar sólo cuando el número de nodos de chequeo insatisfechos es menor a un determinado límite d_t. Esto ocurre después de varias iteraciones en funcionamiento normal (i.e., con la expansión de rango desactivada). El valor de d_t se determina como

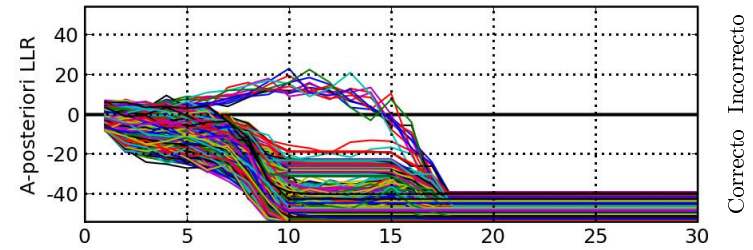
$$d_t = 1 + \max_{\alpha \in \Phi} \{\text{peso}(\text{sindrome}(\alpha))\}, \tag{3.4}$$

por lo tanto d_t es igual a uno más el máximo número de ecuaciones de chequeo insatisfechas generadas por los *absorbing-sets* Φ que producen el piso de error. En el caso particular del código $\mathcal{C}_2$, Φ se compone de un único *absorbing-set* tipo AS(12,8) y por este motivo se utiliza $d_t = 9$.

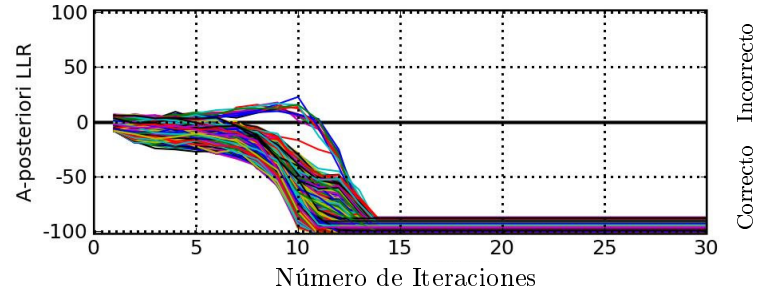
La expansión de rango puede implementarse eficientemente en el nodo



(a) $r_e = 5b$



(b) $r_e = 6b$



(c) $r_e = 7b$

Figura 3.5: Evolución de la información a posteriori L^o del código $\mathcal{C}_2$ con el AS(12,8) usando el decodificador SMSA con diferentes niveles de resolución.

variable de la siguiente manera

$$L^e_{v_i \to c_j} \quad = \quad (\kappa_1)^t \cdot L^a_i + \kappa_2 \cdot \left(\sum_{c_k \in \mathcal{C}_{v_i} \setminus \{c_j\}} L^e_{c_k \to v_i} \right), \tag{3.5}$$

donde $t = 1, 2, ...,$ denota el número de iteraciones *adicionales* requeridas como post-procesamiento. Los factores κ_1 y κ_2 son ganancias positivas menores que la unidad. Por simplicidad, utilizaremos en lo sucesivo $\kappa_1 = \kappa_2 = \kappa$. De esta forma, el algoritmo propuesto cambia la escala por un factor

κ en (i) la salida de los nodos variables según la ecuación

$$L^e_{v_i \to c_j} = \kappa \cdot \left(L^a_{v_i} + \sum_{c_k \in \mathcal{C}_{v_i} \setminus \{c_j\}} L^e_{c_k \to v_i} \right), \tag{3.6}$$

y (ii) en la información a priori según la ecuación

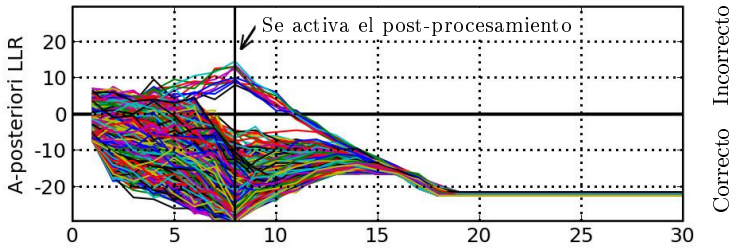
$$L^a_{v_i} \leftarrow \kappa \cdot L^a_{v_i}, \tag{3.7}$$

donde la flecha en la ecuación (3.7) indica que el valor de $L^a_{v_i}$ es reemplazado por $\kappa \cdot L^a_{v_i}$. Nótese que cuando el post-procesamiento está activo, la información a priori L^a se reduce gradualmente a cero, luego de lo cual el decodificador continúa operando sin información a priori. El funcionamiento bajo estas condiciones es posible debido a que los mensajes del algoritmo se encuentran aproximadamente estables al momento en que se activa el post-procesamiento.

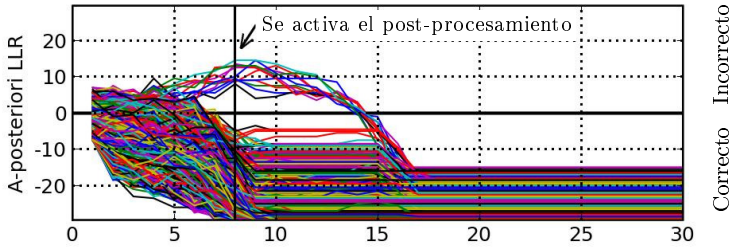
3.4.2. Resultados de Simulación

En la Figura 3.6 se muestra la evolución de la información a posteriori L^o al intentar corregir un AS(12,8) usando una resolución r_e =5 bits en combinación con el algoritmo de post-procesamiento propuesto en esta Tesis y en combinación con el algoritmo propuesto en [185]. Se observa que el AS puede corregirse con 5 iteraciones adicionales cuando se utiliza el esquema propuesto mientras que requiere 9 iteraciones adicionales cuando se utiliza el esquema de [185]. Tal reducción en el número de iteraciones tiene un importante impacto en la reducción de la complejidad de implementación, lo que posiciona al algoritmo desarrollado en esta Tesis como una excelente alternativa de post-procesamiento para atacar el piso de error. Por otro lado, la superioridad del algoritmo propuesto se ve confirmada en la Figura 3.7, en donde se compara el desempeño en la zona de piso de error del código $\mathcal{C}_1$ en combinación con ambos algoritmos de post-procesamiento. En dicha figura se observa que el esquema propuesto en [185] logra una reducción del piso de error apenas superior a un orden de magnitud, mientras que con el nuevo esquema no se llega a observar piso de error, logrando de esta forma una reducción de al menos tres ordenes de magnitud.

Finalmente, la Figura 3.8 muestra el desempeño del código $\mathcal{C}_2$ en combinación con el nuevo algoritmo utilizando r_e =5 bits. Tal esquema de codificación no muestra un piso de error detectable mediante simulación en FPGA hasta un BER de 10^{-15} y tampoco hay muestras de piso de error hasta al menos 10^{-16} en base a estimaciones basadas en los patrones de error capturados. Se concluye entonces que el código propuesto en combinación con el nuevo algoritmo de post-procesamiento alcanza un ganancia neta de codificación de 11.3 dB a un BER=10^{-15} utilizando el algoritmo SMSA con sólo 5 bits de precisión.



(a) PP propuesto en esta Tesis



(b) PP propuesto en [185]

Figura 3.6: Evolución de la información a posteriori L^o para el código $\mathcal{C}_2$ con el AS(12,8) usando una resolución $r_e = 5$ en combinación con (a) el algoritmo de post-procesamiento propuesto en esta Tesis y (b) el algoritmo de post-procesamiento propuesto en [185].

3.5. Conclusión

La partición regular por columnas en combinación con la estructura cuasi-cíclica posibilita la construcción de códigos LDPC de *overhead* configurable con (*i*) bajo piso de error (*ii*) alta ganancia de codificación y (*iii*) una alta regularidad en la matriz de paridad que posibilita la implementación de una arquitectura semiparalela eficiente (véase Capítulo 4). Por otro lado, el nuevo algoritmo de post-procesamiento posee un desempeño superior al del esquema propuesto recientemente en [185]. En particular, el algoritmo desarrollado en esta Tesis mostró una capacidad de reducción en el piso de error de al menos tres órdenes de magnitud mientras que el esquema de [185] logró una reducción apenas superior a un orden de magnitud. Otra ventaja es que dicho algoritmo puede ser fácilmente implementado como parte del nodo variable con muy poca complejidad adicional. A manera de ejemplo, se diseñó un código CC-PRC-LDPC tipo-2 con 20 % de *overhead* que posee una ganancia neta de codificación de 11.3 dB a un BER de 10^{-15} sin piso de error. Este desempeño se obtuvo usando el algoritmo SMSA con 5 bits de resolución y un promedio de 8 iteraciones en combinación con el nuevo algoritmo de post-procesamiento. Finalmente, es importante

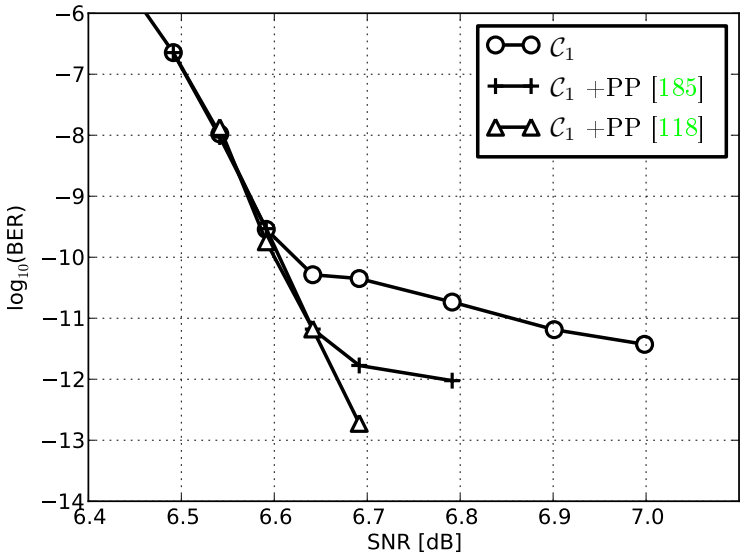


Figura 3.7: Emulación en FPGA del código $\mathcal{C}_1$ con 13 iteraciones. Se observa que sin PP hay un piso de error que comienza en $\approx 10^{-10}$. Con el PP propuesto en [185] hay un piso de error que comienza en $\approx 10^{-12}$. Finalmente, con el esquema propuesto en esta Tesis [118] no se observa piso de error.

mencionar que este código será usado en la próxima generación de enlaces ópticos coherentes a velocidades de 100 Gb/s y 200 Gb/s [96] superando en términos de desempeño y complejidad a sus dos principales competidores propuestos en [38] y [125].

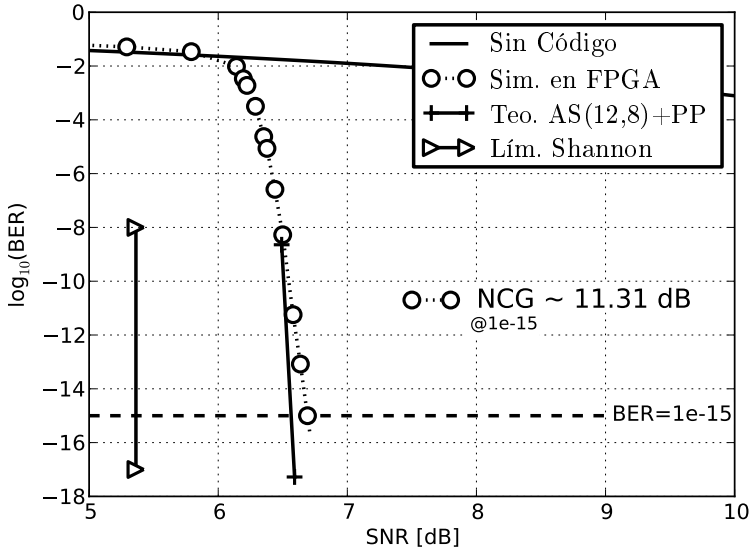


Figura 3.8: Desempeño del código $\mathcal{C}_2$ en combinación con el algoritmo de post-procesamiento propuesto operando sobre el algoritmo SMSA con una resolución de bits.

Capítulo 4

Arquitectura de Implementación

Hay ciertas cosas que para hacerlas bien no basta haberlas aprendido.

Lucio Anneo Séneca.

Resumen

El presente capítulo describe la arquitectura de implementación del decodificador del código LDPC descripto en el capítulo 3. Como primer paso se introducen los conceptos básicos que deben considerarse en el diseño de circuitos digitales síncronos para aplicaciones específicas. En base a dichas ideas se describe la arquitectura propuesta y sus ventajas. Finalmente se presentan los resultados de la implementación en tecnología CMOS de 28nm. Los aportes descriptos en el presente capítulo se encuentran parcialmente publicados en [118] y en la patente [51].

4.1. Introducción

El alto desempeño de los códigos LDPC los hace atractivos para numerosas aplicaciones. Sin embargo, su utilización se ve limitada por (*i*) el problema del piso de error y (*ii*) la alta complejidad de implementación de su algoritmo de decodificación. Soluciones al primer problema fueron descriptas en los Capítulos 3 y 5, aquí se describirá una solución al segundo. La dificultad asociada a este último problema radica en que la implementación directa del algoritmo SPA en un circuito integrado conlleva a una alta densidad de interconexiones y a una gran cantidad de memoria. Esto último incrementa significativamente el área del circuito y la disipación de energía como así también reduce la velocidad de procesamiento [37] [65]. Este problema se agrava aún más en aplicaciones de alta velocidad en donde es necesario el uso de procesamiento en paralelo [44] [112]. El objetivo de este capítulo es proponer una arquitectura que evite los problemas mencionados. En particular se propone una arquitectura parcialmente paralela que reduce la complejidad de las interconexiones y hace eficiente uso del hardware al evitar ciclos inactivos. Además, la misma posee un *overhead* programable que la hace atractiva para aplicaciones que requieran una codificación que pueda adaptarse a las condiciones del canal[1]. Como verificación de la factibilidad de implementación, la nueva arquitectura se sintetizó en tecnología CMOS de 28nm y se determinó la velocidad de operación, la potencia de disipación y el área. *Este trabajo representa el primer código LDPC no concatenado para comunicaciones ópticas que demostró ser implementable con tecnología CMOS de 28nm y con una ganancia de codificación neta igual a 11.3 dB a un BER de* 10^{-15} *sin piso de error verificado mediante emulación en FPGA.*

4.2. Circuitos Digitales Síncronos

La arquitectura de implementación analizada en este trabajo está orientada a un *circuito integrado de aplicación específica* (ASIC[2]). En particular, está enfocada a circuitos digitales síncronos. Estos últimos combinan tres elementos básicos a saber: (*i*) operaciones lógicas, (*ii*) unidades de memoria y (*iii*) conexiones. Las unidades de memoria se actualizan en forma simultánea y periódica. Tal cambio está usualmente controlado por una o más señales denominadas señales de reloj. El período de la señal de reloj debe ser tal que permita la propagación de las señales

[1] Actualmente, éste es un requerimiento típico en canales inalámbricos, por ejemplo, véanse los estándares de la Tabla 1.1. Como ejemplo particular, el DVB-S2 utiliza dos códigos LDPC, uno de longitud 16200 bits con tasas de codificación programables de 1/5, 1/3, 2/5, 4/9, 3/5, 2/3, 11/15, 7/9, 37/45, 8/9 y otro código de longitud 64800 bits con tasas de codificación programables de 1/4, 1/3, 2/5, 1/2, 3/5, 2/3, 3/4, 4/5, 5/6, 8/9, 9/10.

[2] Por sus siglas en inglés para *Application-Apecific Integrated Circuit.*

eléctricas a través de las operaciones lógicas y las interconexiones en el camino que va desde una unidad de memoria a otra. El diseño de ASICs síncronos se puede dividir según el nivel de abstracción. Una división típica involucra los siguientes niveles:

1. **Nivel de Sistemas.** En este nivel se *ataca* el principal problema mediante la división del mismo es sub-problemas más simples. El resultado es lo que se conoce como una arquitectura a nivel de bloques en donde cada bloque se encarga de un sub-problema particular. Se describe tanto la interacción entre bloques como la funcionalidad de cada uno a un nivel conceptual. En este nivel se ignoran los detalles algorítmicos y estructurales del hardware.

2. **Nivel de Algoritmos.** En este nivel se describen los algoritmos utilizados en cada bloque del sistema sin tener en cuenta consideraciones del hardware. Es común simular tales algoritmos mediante su programación en un lenguaje como C++, MatLab o Python con la finalidad de analizar el desempeño. En base a los resultados de simulación es frecuente realizar cambios en el diseño tanto en el nivel de algoritmos como en el nivel de sistemas.

3. **Nivel de Transferencia de Registros (RTL).** En este nivel se describe el algoritmo utilizando conceptos orientados a la implementación digital síncrona. Esta descripción combina tres elementos básicos a saber: (*i*) operaciones lógicas, (*ii*) unidades de memoria, y (*iii*) conexiones. En particular, se describe el flujo de datos a través de las unidades de memoria y los bloques combinacionales. El nivel de descripción de los bloques combinacionales puede ir desde operaciones lógicas simples hasta operaciones aritméticas complejas.

4. **Nivel de Celdas o Compuertas.** En este nivel se traducen los bloques combinacionales y de memoria del nivel RTL en términos de las celdas básicas disponibles en una determinada tecnología. Las celdas combinacionales pueden abarcar funcionalidades que van desde una simple compuerta lógica hasta decenas de ellas. Por otro lado, las celdas de memoria pueden ir desde un simple flip-flop hasta grandes memorias de acceso aleatorio (RAM).

5. **Nivel de Circuito.** En este nivel se traducen las celdas en términos de componentes electrónicos básicos como transistores, resistencias, capacitores e inductores. Esta descripción se utiliza para simular el circuito a nivel eléctrico. La descripción a nivel de circuito también puede extraerse a partir de la descripción física con la finalidad de modelar efectos de propagación de las señales a través de los cables y efectos de acoplamiento entre otros.

6. **Nivel Físico.** En este nivel se traducen las celdas en términos físicos. Los elementos básicos son polígonos que representan los diferentes materiales utilizados en la fabricación, tales como metal, óxidos, etc. La realidad física adquirida en este nivel hacen tangibles las propiedades espaciales del circuito. En particular, se manifiestan los problemas relacionados a la red de interconexiones[3] y al área.

A continuación se describirá el diseño de la arquitectura de implementación de un decodificador LDPC hasta un nivel intermedio entre sistemas y RTL. La descripción a nivel de compuertas o inferior es conocida como micro-arquitectura y se realiza en forma semi-automática por programas de *diseño electrónico automatizado* (EDA[4]) tomando como entrada la descripción a nivel de RTL. En particular, las herramientas de las empresas Cadence[5], Synopsys[6] y Mentor Graphics[7] traducen la descripción a nivel de RTL en una descripción a niveles inferiores optimizando los recursos de hardware disponibles según la tecnología de implementación particular que se utilice. Sin embargo, y a pesar de no ser tratadas las etapas de la micro-arquitectura en este capítulo, se cubrirán los resultados finales a nivel físico ya que éstos determinan la factibilidad real de la arquitectura propuesta.

4.2.1. Velocidad, Potencia y Área

Los objetivos que se persiguen en el diseño de una arquitectura eficiente son: (*i*) alta velocidad de procesamiento, (*ii*) baja potencia de disipación y (*iii*) área reducida. La alta velocidad de procesamiento está motivada directamente por la demanda de mayores velocidades en los sistemas de comunicaciones. La reducción en potencia está motivada por el incremento en la duración de las baterías de dispositivos móviles o el aumento en la densidad de integración, cuando esta última está limitada por cuestiones de refrigeración. Finalmente, la reducción en el área está motivada por el alto costo de fabricación en serie, el cual es proporcional al área del circuito. Estas tres dimensiones no son independientes, por ejemplo una reducción en el área puede reducir la potencia y el camino crítico posibilitando un aumento en la velocidad. Por el contrario, un aumento en la velocidad como consecuencia de un incremento en el paralelismo, trae aparejado un incremento tanto de la potencia como del área, mientras que si es llevado a cabo por un incremento

[3] Los códigos LDPC poseen en general una red de interconexiones muy compleja que dificulta enormemente la etapa final de implementación del decodificador. Por tal motivo, para demostrar la eficiencia de una determinada arquitectura de implementación es necesario llegar hasta el nivel físico.

[4] Por sus siglas en inglés para *Electronic Design Automation*.

[5] Cadence Design Systems, Inc. (NASDAQ: CDNS). URL: http://www.cadence.com.

[6] Synopsys Inc. (NASDAQ: SNPS). URL: http://www.synopsys.com.

[7] Mentor Graphics Inc. (NASDAQ: MENT). URL: http://www.mentor.com.

Figura 4.1: Circuito digital síncrono básico.

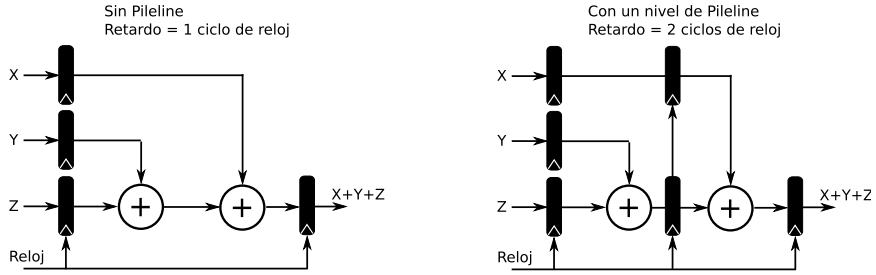


Figura 4.2: Ejemplo del uso de pipeline.

en la frecuencia de reloj traerá un impacto dominante en la potencia. Esta interrelación hace necesario que se deba tener una visión integral a la hora de optimizar la arquitectura. A continuación se describirán brevemente los conceptos básicos relacionados con cada uno de estos objetivos.

La velocidad de procesamiento está determinada principalmente por el factor de paralelismo y la frecuencia de reloj. El factor de paralelismo representa la cantidad de unidades de información que pueden procesarse simultáneamente. Incrementar el factor de paralelismo – manteniendo constante la frecuencia de reloj – incrementa linealmente la velocidad de procesamiento a expensas de una mayor cantidad de lógica y de memoria. Esto último se traduce en un incremento de la potencia y el área. Por otro lado, la frecuencia máxima de reloj está determinada principalmente por el *camino crítico*. El camino crítico es el camino eléctrico con el mayor tiempo de propagación de la señal eléctrica e incluye no sólo los tiempos de propagación a través de la lógica combinacional sino también el tiempo de propagación a través de los cables. El camino crítico nace y finaliza en unidades de memoria, como por ejemplo en flip-flops (véase Figura 4.1). La estrategia convencional para reducir el retardo del camino crítico, conocida bajo el nombre de *pipeline*, consiste en dividir la lógica combinacional en etapas separadas por registros de memoria como se muestra en la Figura 4.2. De esta forma, el camino crítico se divide en caminos más cortos permitiendo incrementar la velocidad del reloj y por consiguiente la velocidad de procesamiento (también se reduce la potencia disipada por el efecto de *glitching*).

La potencia depende de: la cantidad y tipo de celdas, la red de conexiones de datos, la red de conexiones del reloj y de las condiciones de operación

(voltaje, temperatura, etc). En tecnología CMOS la potencia promedio $\bar{P}$ se puede descomponer en tres componentes: (i) la potencia de conmutación o potencia dinámica P_d, (ii) la potencia de corto circuito P_c y (iii) la potencia de *leakage* P_l. Por lo tanto [25],

$$\bar{P} = P_d + P_c + P_l.$$

La potencia dinámica está relacionada con la energía requerida para cargar la capacidad de carga C_L a través del transistor PMOS cuando ocurre la transición de voltaje de 0 a V_{dd} (o 1 lógico). En la transición de voltaje desde V_{dd} a 0 no se consume energía de la fuente; sin embargo la energía $C_L \cdot V_{dd}^2/2$ almacenada en el capacitor es disipada. Luego, la energía dinámica promedio puede calcularse como $P_d = \alpha_{0\to1} \cdot C_L \cdot V_{dd}^2 \cdot f_{clk}$ donde $\alpha_{0\to1}$ es el factor de actividad o conmutación – i.e., la probabilidad de que se produzca una transición en un determinado ciclo de clock – y f_{clk} es la frecuencia del reloj. La anterior ecuación nos brinda varios grados de libertad para reducir la potencia. Desde un nivel de circuito o nivel físico podemos reducir la capacidad de carga y reducir el voltaje de alimentación. Por otro lado, desde un nivel de algoritmo o RTL podemos reducir la cantidad de conmutaciones o reducir la frecuencia de reloj. Respecto a la frecuencia de reloj, hay una relación de compromiso entre ésta y el factor de paralelismo (aumentando el paralelismo disminuye la frecuencia de reloj pero también aumenta el número de celdas y el área). Por otro lado, la cantidad de conmutaciones tiene varios caminos para su optimización que van desde una adecuada codificación de la información (por ejemplo mediante una codificación Grey) hasta eliminar las conmutaciones transitorias o espurias que se producen en los circuitos combinacionales por el efecto de *glitching* [31]. Además, se debe tener en cuenta que gran parte de la potencia dinámica es consumida por la red de interconexiones del reloj, por esta razón, cuando un bloque permanece inactivo es posible reducir su potencia al apagar el clock (técnica que se conoce como *clock gating*). Por último, las potencias de corto circuito y de *leakage* pueden calcularse como $P_c = I_c \cdot V_{dd}$ y $P_l = I_l \cdot V_{dd}$ donde I_c es la corriente de corto-circuito y I_l es la corriente de *leakage*. Estas últimas potencias pueden reducirse desde un nivel de circuito o físico pero no desde un nivel de algoritmos o RTL por lo que no serán consideradas en este capítulo. Además, P_c y P_l poseen una contribución secundaria en la potencia total en tecnología CMOS ya que la potencia dominante es usualmente la dinámica. Para más detalles, véase [25], [31], [169] y las referencias allí citadas.

Finalmente, el área depende tanto de la cantidad y tipo de celdas utilizadas para implementar la lógica combinacional y las unidades de memoria como de la red de interconexiones entre dichas celdas. La principal forma de reducir área desde el nivel de RTL es mediante serialización del procesamiento acompañada de un incremento en la frecuencia de reloj. Sin embargo, como se mencionó previamente, para aumentar la frecuencia de reloj es necesario reducir el camino crítico, siendo una posible estrategia el

uso de *pipeline*. Existe entonces desde el punto de vista del área un balance que involucra el factor de paralelismo y el nivel de *pipeline* utilizado. Por otro lado, se encuentra también el área de la red de interconexiones, la cual puede representar fácilmente un 50 % del área total. Esto último tiene lugar usualmente en arquitecturas basadas en bloques altamente interconectados en forma no-local, como ocurre en el algoritmo SPA. En este último caso, la mejor estrategia es atacar el problema en el más alto nivel posible para reducir la cantidad de interconexiones.

4.2.2. Divide y Vencerás

Para implementar un algoritmo complejo, una buena estrategia es dividirlo en sub-algoritmos más simples. Estos sub-algoritmos se implementan en *bloques de procesamiento* (BP) más fáciles de optimizar que al algoritmo completo. En nuestro caso, el algoritmo completo es el decodificador LDPC y los sub-algoritmos son los cálculos realizados en los nodos variables y de chequeo. Los bloques de procesamiento correspondientes a estos dos sub-algoritmos se denominan *bloque de procesamiento del nodo variable* (BP-NV) y *bloque de procesamiento del nodo de chequeo* (BP-NC) respectivamente. Dichos bloques de procesamiento están compuestos de una o varias *unidades de procesamiento* (UP), donde entendemos por UP a un BP que necesita esperar hasta tener todas sus entradas listas para poder procesarlas. Si en una UP no todas las entradas contienen información válida, dicha UP se ve forzada a permanecer en un estado inactivo y por consiguiente desperdiciando capacidad de cálculo. En esta última situación ocurre también que los mensajes válidos a la entrada de la UP tengan que ser almacenados generando lo que denominaremos *mensajes o información en espera*. Cabe notar que incluso si las UPs no sufren períodos de inactividad, es posible que haya mensajes en espera (particularmente si no hay suficientes UPs para la cantidad de información disponible). Una arquitectura eficiente debe evitar tanto la inactividad de las UPs como los mensajes en espera. Si una arquitectura evita estos dos problemas diremos que posee una generación y procesamiento de mensajes que operan bajo en método de *justo-a-tiempo*[8] (JaT). Aquí es importante remarcar que tal procesamiento JaT está relacionado únicamente con la implementación en hardware del algoritmo y no modifica de ningún modo el algoritmo en cuestión, en particular es independiente de la secuencia de cálculo de los mensajes analizada en la Sección 2.1.4. El objetivo de la arquitectura propuesta a continuación es lograr una arquitectura JaT que permita reducir la cantidad de lógica, memoria e interconexiones. Con tal objetivo, los sub-bloques BP-NV y BP-NC deben diseñarse en armonía con la arquitectura a nivel global.

[8] El concepto de "justo-a-tiempo" representa un sistema de organización de la producción usado originalmente en fábricas japonesas (véase [164]).

Las arquitecturas de implementación del algoritmo SPA pueden clasificarse de acuerdo al paralelismo en: *Completamente Paralelas* (CP), *Parcialmente Paralelas* (PP), y *Series*. En una arquitectura CP, cada nodo de chequeo y cada nodo variable tiene dedicado un hardware específico que opera en forma completamente paralela y los mensajes entre ambos nodos son transportados por una red de cables. El factor de paralelismo es por consiguiente igual a la longitud del código. A excepción que se trate de un código LDPC muy pequeño, tal paralelismo en general excede significativamente los requerimientos de velocidad trayendo aparejado un sobre-dimensionamiento del hardware. Por otro lado, en las arquitecturas serie suele ocurrir el fenómeno contrario: la frecuencia admisible de reloj no es suficiente para las aplicaciones de alta velocidad (mayores a 1 Gb/s). Es deseable entonces disponer de una solución intermedia utilizando una arquitectura PP. En esta última es usual no implementar todos los nodos como bloques de hardware independientes. En su lugar, un mismo bloque realiza en tiempos diferentes las operaciones correspondientes a varios nodos variables o nodos de chequeo. Lo anterior complica significativamente el flujo de mensajes, el cual si no se resuelve en forma adecuada, acarrea problemas de hardware inactivo y mensajes en espera.

Un ejemplo clásico al que se aspira usualmente en el diseño de una arquitectura JaT y eficaz en términos de conexiones es la denominada arquitectura *sistólica* propuesta por Kung y Leiserson en [95]. Como se muestra en la Figura 4.3, esta arquitectura posee una estructura regular y modular organizada en forma de red con un gran número de bloques de procesamiento idénticos y conectados localmente sólo con los bloques vecinos, y donde sólo a través de las celdas en la periferia de la red se pueden ingresar o extraer datos. Esta arquitectura posee grandes ventajas para su implementación en ASIC. Desafortunadamente, las propiedades ralas y pseudo-aleatorias de las matrices de paridad de los códigos LDPC impide realizar una intercomunicación local entre bloques de procesamiento por lo que será necesario resolver el problema con una estrategia diferente.

4.3. Nueva Arquitectura

La forma más directa de transformar una arquitectura CP en una PP es reducir el número de BP-NV y BP-NC reusándolos varias veces por iteración. Alternativamente, es posible mantener la misma cantidad de BP-NV y BP-NC – uno por nodo – pero en lugar de utilizar BP-NV y BP-NC completamente paralelos se utiliza sub-bloques parcialmente paralelos o series. La primer estrategia corresponde a una implementación parcialmente paralela a nivel global mientras que la segunda sólo lo es a nivel local. Una arquitectura eficiente debe en general combinar ambas estrategias, y el modo de hacerlo dependerá en general de las características de la matriz de paridad.

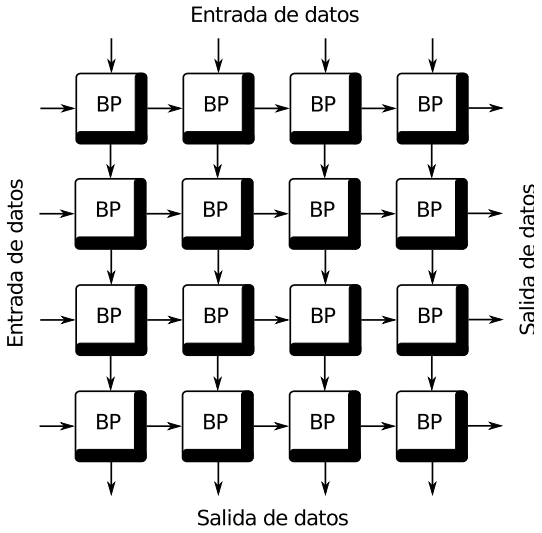


Figura 4.3: Ejemplo de un arreglo sistólico matricial.

En el código propuesto en el Capítulo 3, el peso de las filas es considerablemente mayor al peso de las columnas, por consiguiente el nodo de chequeo es significativamente más complejo que el nodo variable. Por otro lado, la cantidad de nodos variables es muy superior a la cantidad de nodos de chequeo. Una estrategia eficiente consiste en utilizar un procesamiento parcialmente paralelo a nivel local para los nodos de chequeo y un procesamiento parcialmente paralelo a nivel global para los nodos variables. Esto es, se implementa un bloque de hardware parcialmente paralelo por cada nodo de chequeo y se reusan bloques de hardware completamente paralelos para procesar varios nodos variables. Esto último permite además balancear el camino crítico sin utilizar *pipeline* ya que de lo contrario éste sería muy grande en el nodo de chequeo y pequeño en el nodo variable. Evitar el *pipeline* en el nodo de chequeo previene un incremento significativo en la complejidad. Además, como se verá en la próxima sección, tal estrategia en combinación con la partición regular por columnas descripta en la Sección 3.2.1 posibilitará una secuencia de procesamiento JaT.

4.3.1. Arquitectura PRC sin *Pipeline*

La Figura 4.4 muestra la arquitectura de implementación basada en la *partición regular por columnas* (PRC) analizada en el Capítulo 3. El ejemplo que ilustra dicha figura corresponde a un código LDPC con una matriz de paridad $(4, 12)$-regular como la de la Figura 3.1. La arquitectura se compone de $p = 4$ BP-NVs con procesamiento en paralelo, $m = 8$ BP-NCs con procesamiento parcialmente paralelo utilizando un factor de paralelismo $\delta = 2$ (véase Figura 4.5); dos redes de permutación reconfigurables Π y Π^{-1} y

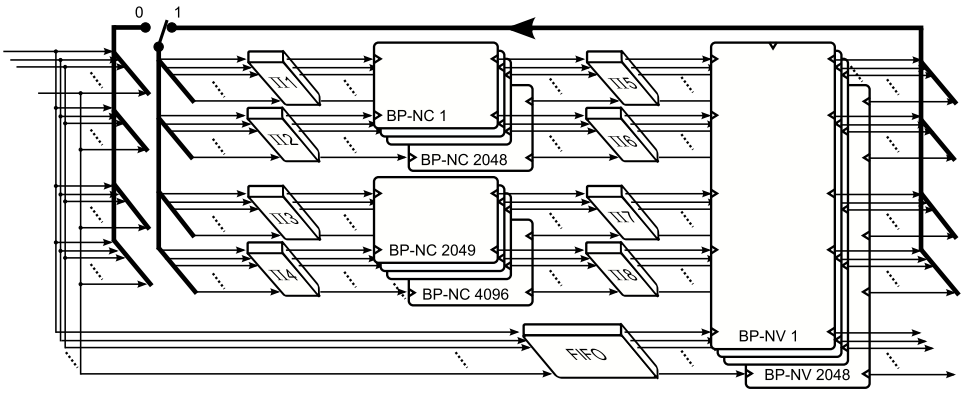


Figura 4.4: Arquitectura PRC

una memoria FIFO para almacenar información a priori L_i^a con $i = 1, \ldots, 24$. Cada iteración se divide en $\mu = 6$ pasos. En el q-ésimo paso, Π_q y Π_q^{-1} son configurados de acuerdo a la sub-matriz $H^{(q)}$. En la versión básica de esta arquitectura no hay registros dentro de los BP-NVs ni en las redes de permutación. El almacenamiento de información sólo tiene lugar dentro de los BP-NCs y en la FIFO. Cuando el interruptor está en la posición 0, una nueva palabra código ingresa al decodificador luego de lo cual el interruptor cambia a la posición 1, cerrando el lazo de los mensajes para comenzar el proceso iterativo. Los mensajes a la salida de los BP-NCs se procesan por los BP-NVs y se envían de vuelta a los BP-NCs a través de las redes de permutación. Notar que se realiza un procesamiento del tipo *justo a tiempo* que evita tiempos de espera en el cálculo de los mensajes. Por otro lado, debido a que no es necesario almacenar ninguno de los mensajes $L^e_{v_i \to c_i}$, los requerimientos de memoria se ven reducidos a la mitad. Además, cuando se utiliza el algoritmo MSA, sólo es necesario almacenar por cada BP-NCs (i) los valores del primer y segundo mínimo de los mensajes $L^e_{v_i \to c_i}$ y (ii) la posición del primer mínimo (véase la Figura 4.5). Tal requerimiento de memoria es independiente del grado de los nodos de chequeo y consecuentemente de la longitud del código, dependiendo solamente del número de nodos de chequeo. Esta propiedad permite una implementación que tenga una tasa de codificación (u *overhead*) reconfigurable sin necesidad de incrementar la complejidad. Dicha reconfiguración se logra variando la longitud n del código en pasos de p mientras se mantiene m constante, i.e., variando el número de columnas de sub-matrices y manteniendo constante el número de filas.

4.3.2. Arquitectura PRC con *Pipeline*

En la arquitectura PRC básica descripta en la sección anterior, no hay registros de *pipeline* en el camino que va desde la salida del BP-NC hasta la entrada a dicho bloque. En un único ciclo de reloj, las señales eléctricas

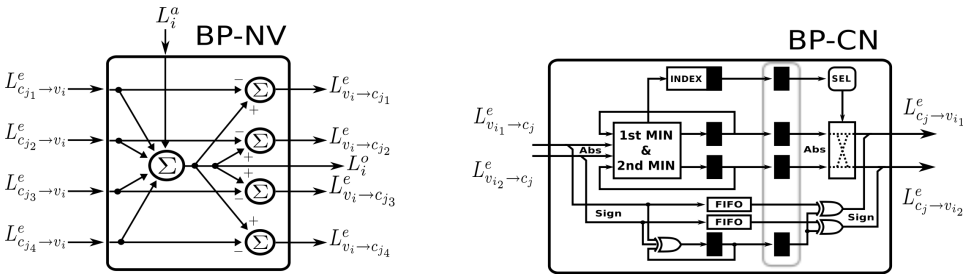


Figura 4.5: Arquitectura básica – sin pipeline – de los bloques de procesamiento del nodo variable (BP-NV) y del nodo de chequeo (BP-NC).

deben atravesar las redes de permutación, la lógica combinacional de los BP-NVs y los cables de interconexión. Esto último tiene la desventaja de limitar la frecuencia de reloj a la que puede operar el decodificador. Además, a pesar de que tal frecuencia sea aceptable de acuerdo a los requerimientos de velocidad, se estaría frente a una arquitectura sub-óptima como consecuencia de un desbalance entre el camino crítico interno del BP-NC y el externo. El BP-NC podría operar a una frecuencia muy superior a la que podría operar el sistema completo, desperdiciando de esta forma capacidad de cálculo en dicho bloque. Para solucionar este problema, es necesario reducir el camino crítico fuera del BP-NC mediante el uso de *pipeline*. Sin embargo, a pesar de que el *pipeline* permite balancear los caminos críticos, origina un nuevo problema. Los registros de *pipeline* incrementan el número de ciclos de reloj requeridos para realizar cada iteración, generando un período de tiempo muerto (igual al número de etapas de *pipeline*) durante el cual el hardware se encuentra inactivo. Esto se debe a que cuando el BP-CN genera la última salida de datos correspondiente a una iteración, no puede generar nuevas salidas hasta que toda la información enviada al BP-NV retorne al BP-NC y sea procesada por éste.

Para evitar el tiempo muerto debido al uso de *pipeline*, se propone procesar varias palabras código en forma entrelazada a nivel de iteraciones. Esto permite que el BP-NC pueda generar salidas mientras aún está procesando en sus entradas datos de una palabra código previa. El mínimo número de palabras código que es necesario procesar simultáneamente depende del nivel de *pipeline* utilizado. Procesar κ palabras códigos en forma entrelazada permite utilizar hasta $\mu(\kappa - 1)$ niveles de *pipeline* sin generar tiempos muertos. Para el caso particular del código propuesto en el Capítulo 3 (donde $\mu = 12$) con $\kappa = 2$ será suficiente ya que permite hasta 12 niveles de *pipeline*. Para soportar el procesamiento de dos palabras código en forma entrelazada es necesario (*i*) independizar la etapa de salida de la etapa de entrada en el BP-NC, (*ii*) duplicar la FIFO de información a priori y (*iii*) modificar la máquina de control. Tales modificaciones no acarrean una complejidad adicional significativa y permiten utilizar el hardware a su

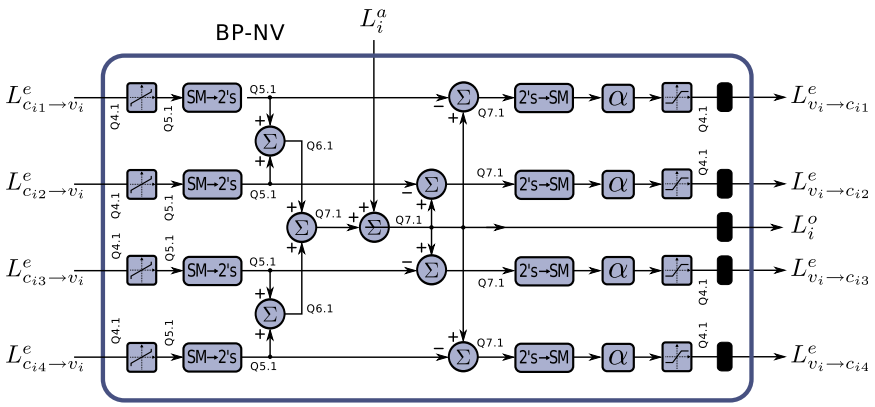


Figura 4.6: Detalle de la implementación del bloque de procesamiento del nodo variable (BP-NV).

máxima capacidad.

4.3.3. Arquitectura del Nodo Variable

La Figura 4.6 muestra la arquitectura del bloque BP-NV implementada en tecnología CMOS de 28nm a una frecuencia de reloj de 300MHz. En el bloque BP-NV se observa un nivel de *pipeline* ubicado a la salida del mismo. Los sub-bloques SM→2's y 2's→SM se encargan de cambiar el formato de la representación de punto fijo desde un esquema módulo-signo a un esquema módulo-2 y viceversa, respectivamente. Recordar del Capítulo 2 que si $Qm.f$ es la resolución de un número signado x de punto fijo en formato módulo-2 y $[b_{m-1}, b_{m-2}, \ldots, b_0, b_{-1}, \ldots, b_{-f}] \in \{0,1\}^{m+f}$ es su representación binaria, entonces el valor de x se obtiene a partir de

$$x \;=\; -b_{m-1}\cdot 2^{m-1} + \sum_{i=-f}^{m-2} b_i \cdot 2^i \quad \in [-2^{m-1}, 2^{m-1} - 2^{-f}] \qquad (4.1)$$

donde el paso de cuantización es 2^{-f}. Por otro lado, en el esquema módulo-signo el valor de x se obtiene a partir de

$$x \;=\; (1 - 2\cdot b_{m-1}) \cdot \sum_{i=-f}^{m-2} b_i \cdot 2^i \quad \in [-2^{m-1} + 2^{-f}, 2^{m-1} - 2^{-f}] \qquad (4.2)$$

donde el paso de cuantización es nuevamente 2^{-f}. Las principales diferencias entre un formato y otro son:

(i) en módulo-signo hay dos representaciones para el cero – una con signo positivo y otra con signo negativo – mientras que en módulo-2 hay una sola,

(*ii*) el mínimo valor en módulo-signo es $-(2^{m-1} - 2^{-f})$ mientras que en módulo-2 es -2^{m-1},

(*iii*) en módulo-signo, el valor absoluto queda completamente determinado por los bits $[b_{m-2}, \ldots, b_{-f}]$ independientemente del bit de signo b_{m-1} mientras que en módulo-2 depende además del signo.

Esta última propiedad es de particular interés para realizar los cálculos en el bloque BP-NC debido a que permite aislar en forma natural el procesamiento del signo y de la amplitud cuando se utiliza el algoritmo MSA o una de sus variantes. Por otro lado, en el bloque BP-NV es conveniente utilizar el formato módulo-2 debido a que simplifica las operaciones de suma y resta. Se propone entonces el uso de módulo-signo para el BP-NC y de módulo-2 para el BP-NV.

El factor de corrección α del algoritmo Minimo-Suma se aplica usualmente a la salida del nodo de chequeo (véase ecuación (2.40)). Sin embargo, en esta Tesis se propone aplicar esta corrección dentro del nodo variable. Esta modificación no afecta el desempeño del algoritmo MSA debido a la linealidad de las operaciones en el nodo de chequeo. Sin embargo, debido a que $\alpha < 1$, lo anterior tiene la ventaja de reducir el rango dinámico de los mensajes entre los nodos variables y los nodos de chequeo[9]. Esto último reduce la actividad de conmutación ayudando a reducir la potencia dinámica. Adicionalmente, reduce la degradación en caso de utilizar una resolución menor para la representación de punto fijo de los mensajes mencionados. En particular, esta modificación en combinación con la cuantización cuasi-uniforme que se describirá más adelante, permite utilizar 4 bits en lugar de 5 para los mensajes sin generar una degradación observable.

El último sub-bloque del BP-NV es el que centraliza los efectos de saturación reduciendo el número total de bits a la misma resolución Q4,1 utilizada en la entrada. Para lograr esto último, no es necesario que todas las operaciones anteriores incrementen la resolución. Como se observa en la Figura 4.6, sólo en las tres primeras operaciones de suma se incrementa la resolución hasta Q7,1. La operación de resta no requiere incremento de resolución ya que siempre se resta un número que previamente fue sumado. Además, la saturación que podría generar la suma de la información a priori no tiene efectos luego de la saturación final ya que la resta y el post-procesamiento combinados no pueden reducir la magnitud por un factor mayor a cuatro. Finalmente, el primer bloque se encarga de reducir el efecto de saturación utilizando una cuantización cuasi-uniforme. Esto se logra mediante la conversión del valor máximo a un valor aún mayor que represente mejor al conjunto de valores previos a la saturación. Esto es, en lugar de utilizar una cuantización uniforme dada por el conjunto

[9] Notar que en la versión original sólo se producía esta reducción en los mensajes desde los nodos de chequeo a los nodos variables pero no a la inversa.

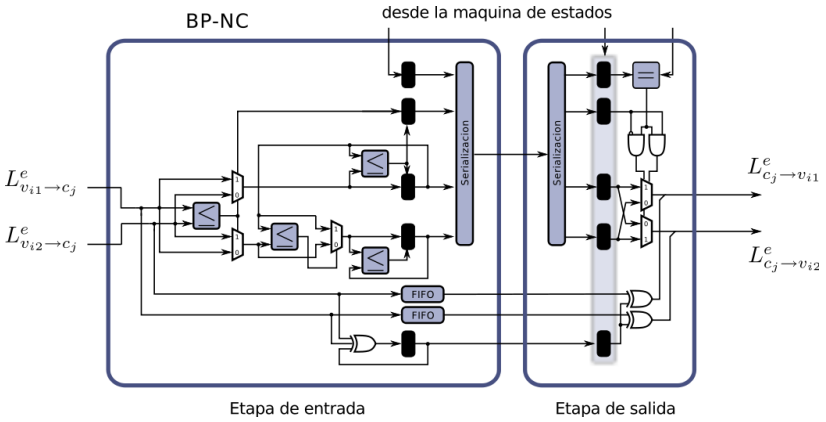


Figura 4.7: Detalle de la implementación del bloque de procesamiento del nodo de chequeo (BP-NC).

de valores $\{-\frac{4}{2}, -\frac{3}{2}, -\frac{2}{2}, -\frac{1}{2}, \frac{0}{2}, \frac{1}{2}, \frac{4}{2}, \frac{3}{2}\}$, correspondientes a la cuantización Q4.1 estándar, se utiliza el conjunto $\{-\frac{5}{2}, -\frac{3}{2}, -\frac{2}{2}, -\frac{1}{2}, \frac{0}{2}, \frac{1}{2}, \frac{4}{2}, \frac{11}{5}\}$ el cual representa estadísticamente mejor la señal previa a la saturación.

4.3.4. Arquitectura del Nodo de Chequeo

La Figura 4.7 muestra la arquitectura del bloque BP-NC la cual, al igual que el bloque BP-NV, fue implementada en tecnología CMOS de 28nm a una frecuencia de reloj de 300MHz. Como se observa, dicho bloque es dividido en dos etapas: (i) una etapa de entrada que realiza la búsqueda de los dos mínimos y la posición del primero como así también calcula el producto de los signos y (ii) una etapa de salida que reconstruye las señales de salida. La ventaja de tal separación radica en el hecho de que al procesar varias palabras código en forma entrelazada, no es necesario transferir los mínimos y sus posición desde la etapa de entrada a la etapa de salida apenas se disponga de dicha información. Por el contrario, se dispone de $\mu(\kappa - 1) - l_p$ ciclos de reloj para realizar tal transferencia donde l_p es el nivel de *pipeline* utilizado. Lo anterior permite serializar parcialmente dicha información reduciendo el número de interconexiones entre la etapa de entrada y la etapa de salida. Esta serialización convierte a las conexiones entre ambas etapas del NP-NC en el punto de menor densidad de interconexiones a lo largo del todo el lazo iterativo haciéndolo el punto ideal para realizar el lazo de retroalimentación como se muestra en la Figura 4.8. Alternativamente, si no se utiliza tal serialización, es posible relajar el camino crítico entre la etapa de entrada y de salida del BP-NC al utilizar un multi-ciclo (i.e., incrementando tiempo disponible para la propagación de la señal) o incluso es posible combinar la serialización con el uso de multi-ciclos.

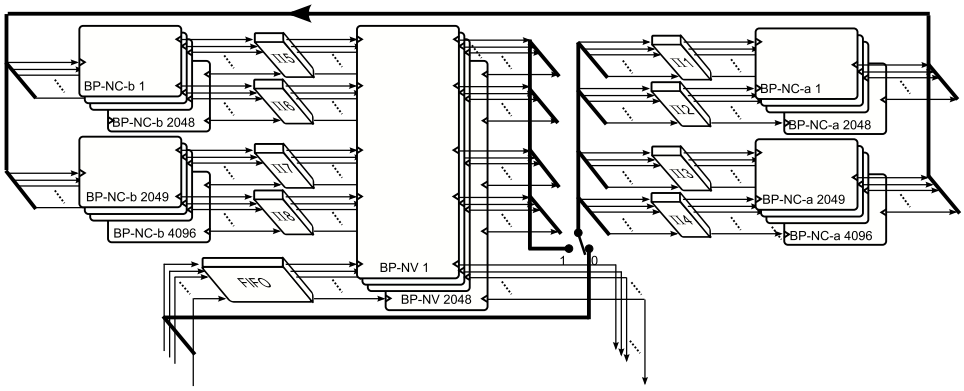


Figura 4.8: Detalle del lazo de realimentación luego de dividir y serializar el BP-NC.

4.4. Implementación

La Figura 4.9 muestra el *layout* final del decodificador en tecnología CMOS de 28nm. El bloque ocupa un área total de 19.6 mm^2 de los cuales 5 mm^2 corresponden únicamente a memoria RAM y los 14.6 mm^2 restantes a flip-flops, combinacionales y conexiones. La potencia de disipación es de 4.7 Watts operando en el punto crítico, esto es a una tasa de BER a la entrada del decodificador de 2.43e-2 (a cuya tasa el BER a la salida del decodificador es igual a 1e-15).

4.5. Conclusión

Se describió la arquitectura de implementación del decodificador del código LDPC descripto en el Capítulo 3. Esta arquitectura permite reducir el número de interconexiones ya que no trabaja con la matriz de paridad completa sino que utiliza una implementación parcialmente paralela basada en el esquema PRC. Cuando este último se utiliza con múltiples palabras código entrelazadas a nivel de iteraciones, permite utilizar *pipeline* y lograr un procesamiento JaT. La división y serialización del BP-NC reduce los problemas de interconexión en el lazo de realimentación. El uso de cuantización cuasi-uniforme en combinación con la aplicación del factor de corrección del algoritmo SMSA en el BP-NV en lugar del BP-NC permite reducir potencia y área sin degradar el desempeño. Finalmente, la arquitectura propuesta se implementó en tecnología CMOS de 28nm logrando una velocidad de procesamiento de 64 Gb/s sobre un área de 19.6 mm^2 y con una potencia de 4.7 Watts. *Este trabajo representa el primer código LDPC no concatenado para comunicaciones ópticas que demostró ser implementable con tecnología CMOS de 28nm y con una ganancia de codificación neta igual a 11.3 dB a un BER de* 10^{-15} *sin piso de error*

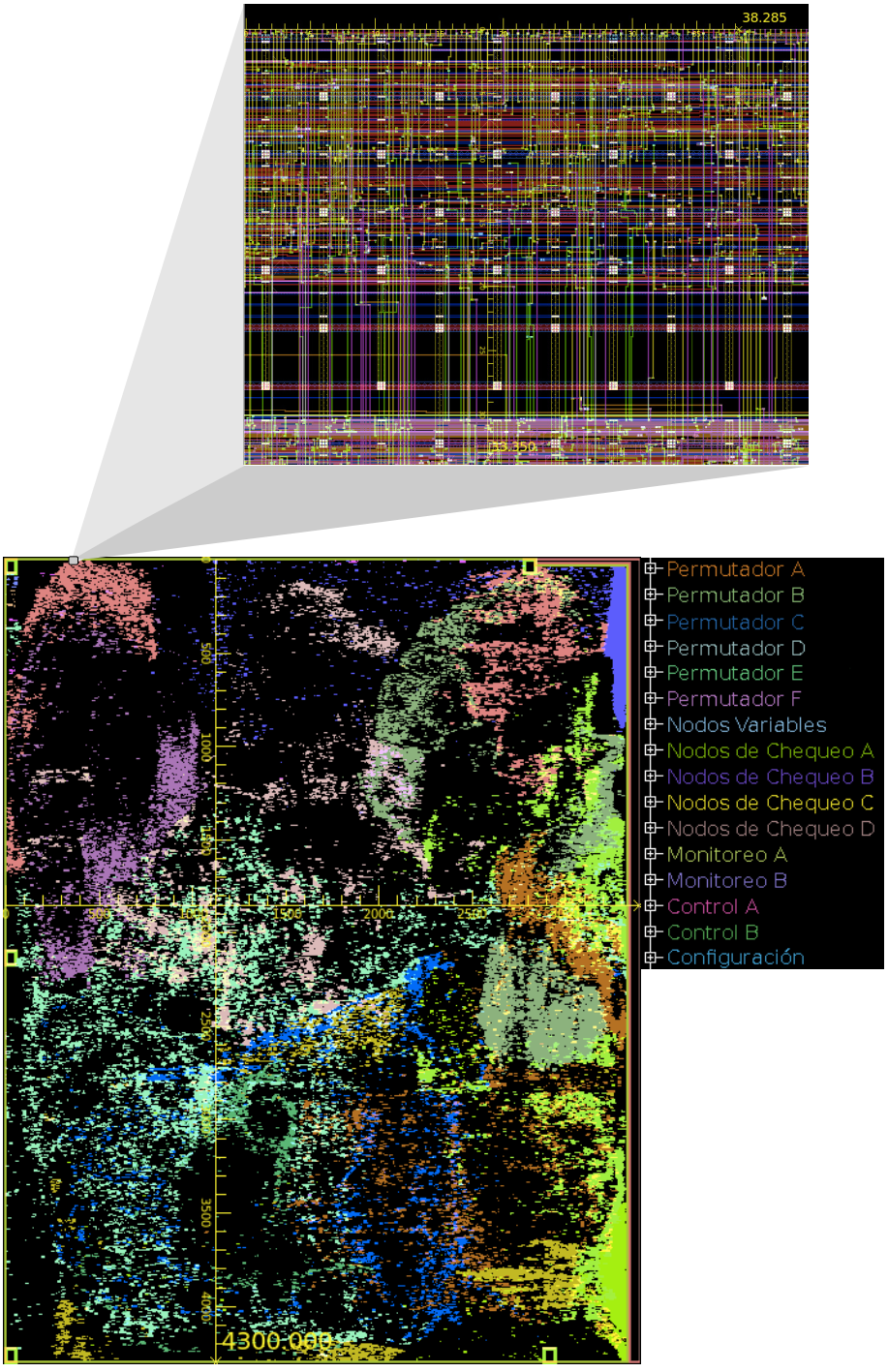


Figura 4.9: Layout de decodificador propuesto en esta Tesis.

verificado mediante emulación en FPGA.

Capítulo 5

Esquema de Concatenación de Códigos

No encuentres la falta, encuentra el remedio.

Henry Ford.

Resumen

El presente capítulo describe los esquemas clásicos de concatenación de códigos usados para combatir el piso de error. Luego se introduce un nuevo esquema de concatenación de códigos para combatir el piso de error y se lo compara con los esquemas clásicos poniendo en evidencia la superioridad del primero. Posteriormente se generaliza el esquema propuesto para corregir el piso de error debido a no sólo patrones de error con síndrome distinto de cero sino también a palabras código de bajo peso. Finalmente, se utiliza el esquema generalizado para reducir el piso de error de un código TPC basado en dos códigos de Hamming extendidos. Tales aportes se encuentran publicados en [121] y [122].

5.1. Introducción

Las técnicas para reducir el piso de error se pueden dividir en dos grupos. El primer grupo tiene como objetivo eliminar las debilidades del algoritmo de decodificación que causan el piso de error mientras que el segundo grupo tiene como objetivo corregir los errores residuales a la salida del decodificador. Dentro del primer grupo se encuentran los algoritmos de post-procesamiento que analizamos en el Capítulo 3 (véase [118] y [185]) y mejoras del algoritmo de decodificación [165]. Sin embargo, el diseño y análisis del piso de error residual de dichas técnicas es en general un problema difícil ya que requiere el conocimiento de la estructura de los patrones de error dominantes. Con tal fin se realizan complejas simulaciones de alta velocidad en FPGAs que permitan capturar dichos patrones. Cuando ésto no es posible o el algoritmo de post-procesamiento es incapaz de reducir el piso de error, la solución consiste en corregir patrones de error residuales a la salida del decodificador (el cual puede o no incluir un algoritmo de post-procesamiento). Con esta finalidad se agrega un código externo que corrija los errores residuales. Esto último es una solución simple y general al problema del piso de error. En particular, es posible estimar el piso de error residual conociendo únicamente el peso de los patrones de error y su probabilidad, sin necesidad de conocer su estructura. Estas ventajas han motivado varios esquemas de LDPC concatenados con un código externo para aplicaciones de 100 Gb/s (véase [82] [125] y las referencias allí citadas). En [125] se propone un esquema de concatenación con un *overhead* total de 20 % basado en la concatenación de un código LDPC y un código RS, el cual logra una ganancia neta de codificación de 9 dB a un BER de 10^{-15}. Otra alternativa propuesta en [125] es la concatenación de un código LDPC con un código externo formado a partir de dos códigos de decisiones duras. Esta triple concatenación de códigos tiene un *overhead* total de 20.5 % y una ganancia neta de codificación estimada en 10.8 dB a un BER de 10^{-15}. Por otro lado, en [82] se propone la concatenación de un código LDPC con un código de RS que logra una ganancia neta de codificación de 11.3 dB a un BER de 10^{-15} con un *overhead* total de 20.5 %.

La incorporación de un entrelazador de bits entre el código interno y el externo también ha sido considerada en el pasado para (*i*) reducir el *overhead* del código externo y/o (*ii*) reducir el piso de error residual [16]. En función de la estructura de los patrones de error dominantes, el entrelazador de bits puede ser capaz de reducir los requerimientos del código externo en términos de capacidad de corrección. Sin embargo, tal conocimiento de la estructura de los patrones de error es un problema complejo para la mayoría de los códigos de interés práctico como consecuencia del bajo piso de error y la alta ganancia de codificación requerida. Ignorando la estructura particular de los patrones de error es común utilizar entrelazadores de bits pseudo-aleatorios de gran tamaño [16]. Sin embargo, tales entrelazadores incrementan significativamente la complejidad de implementación y la

latencia del sistema.

En lo que sigue del presente capítulo se describirá una nueva estrategia de concatenación de códigos diseñada para reducir el problema del piso de error. En dicha técnica se reemplaza el uso de un entrelazador de gran tamaño en combinación con el código externo, por una nueva estrategia de codificación y decodificación basada en la combinación de dos códigos. Se muestra además que la técnica propuesta reduce significativamente la complejidad de implementación[1] en comparación con las técnicas existentes (llegando fácilmente a una reducción de un orden de magnitud). Se describe además una generalización del esquema propuesto que permite reducir el piso de error causado no sólo por patrones detectables, como son los (a, b) *near-codewords* con $b > 0$, sino también el piso de error causado por patrones no detectables como son los (a, b) *near-codewords* con $b = 0$ — i.e., palabras código — [111]. Esta generalización permite usar la técnica propuesta para combatir el piso de error no sólo en códigos LDPC donde el piso de error está dominado por patrones de error detectables, sino también en códigos turbo como el TPC en donde el piso de error suele estar dominado por palabras código de mínimo peso. Como segunda contribución y como muestra del potencial de la nueva técnica, se propone un esquema de codificación basado en un código TPC de baja complejidad compuesto en dos códigos de Hamming extendidos que por sí sólo tiene un piso de error a un BER de 10^{-7}, pero cuando se combina con el esquema propuesto alcanza una ganancia neta de codificación de $\sim$ 11,2 dB a un BER= 10^{-15} con un *overhead* total de $\sim 22\,\%$ y un piso de error a $\sim 7 \cdot 10^{-17}$. Cabe notar que dicho esquema de codificación supera en 0,4 dB la ganancia de los códigos propuestos en [4], [117] y [125].

5.2. Contexto

En esta sección se introducen los conceptos básicos y la notación que será usada a lo largo del presente capítulo. Sea Ω el conjunto de todos los posibles patrones de error a la salida del decodificador interno. Un *patrón de error* es, en su forma más general, definido como el conjunto de todos los bits con error que ocurren simultáneamente en una palabra código. Sea $p(\omega)$ la probabilidad de que ocurra el patrón de error $\omega \in \Omega$ a una determinada relación señal-ruido. La tasa de error de palabras código causada por Ω se denota $P_w(\Omega)$ y se calcula según la ecuación (5.1):

$$P_w(\Omega) = \sum_{\omega \in \Omega} p(\omega), \tag{5.1}$$

[1] Como medida de complejidad de implementación se utilizará el número de celdas lógicas (como por ejemplo, AND, XOR, etc).

mientras que tasa de error de transmisión $P_b(\Omega)$, y la tasa de error de información (i.e., de los datos no codificados), $\tilde{P}_b(\Omega)$, se calculan según las ecuaciones (5.2) y (5.3), respectivamente:

$$P_b(\Omega) = \frac{1}{n} \sum_{\omega \in \Omega} p(\omega) w(\omega), \tag{5.2}$$

$$\tilde{P}_b(\Omega) = \frac{1}{k} \sum_{\omega \in \Omega} p(\omega) \tilde{w}(\omega), \tag{5.3}$$

donde n y k son la longitud y la dimensión del código, respectivamente, mientras que $w(\omega)$ y $\tilde{w}(\omega)$ son el *peso de transmisión* (el cual incluye los bits de redundancia) y el *peso de información* (el cual no incluye los bits de redundancia) del patrón de error ω, respectivamente. El conjunto Ω puede dividirse en dos subconjuntos disjuntos $\mathfrak{Q}$ y $\mathfrak{Q}$ (i.e., $\Omega = \mathfrak{Q} \cup \mathfrak{Q}$ y $\mathfrak{Q} \cap \mathfrak{Q} = \emptyset$), donde $\mathfrak{Q}$ es el conjunto de todos los patrones de error que causan el piso de error y $\mathfrak{Q}$ es el conjunto de los patrones de error no-problemáticos. A partir de (5.1) y (5.3), se deriva la siguiente cota superior para la tasa de error de información causada por los patrones de error de $\mathfrak{Q}$:

$$\tilde{P}_b(\mathfrak{Q}) = \frac{1}{k} \sum_{\omega \in \mathfrak{Q}} p(\omega) \tilde{w}(\omega) \leq \frac{w_{max}}{k} P_w(\mathfrak{Q}), \tag{5.4}$$

donde $w_{max} = \max_{\omega \in \mathfrak{Q}} \{\tilde{w}(\omega)\}$ [130]. Los parámetros w_{max} y $P_w(\mathfrak{Q})$ serán el punto de entrada para el diseño de los esquemas de concatenación de códigos.

5.2.1. Concatenación de Códigos en Serie (CCS)

La concatenación de códigos es una técnica conocida que se basa en la combinación de un código interno y un código externo [54] [16] [137] [73]. Sean $\mathbb{C}_1$ y $\mathbb{C}_2$ el código interno y externo, respectivamente. Cada código está definido por el conjunto de parámetros $[n_j, k_j, d_j]$, donde n_j, k_j y d_j son el tamaño del bloque (o longitud), la dimensión, y la distancia mínima del código $\mathbb{C}_j$, respectivamente. El *overhead* del código se define como $\Theta_j = (n_j - k_j)/k_j$. Sea C_j^i la i-ésima palabra código de $\mathbb{C}_j$. Cada palabra código C_j^i está compuesta por el bloque de información o bloque de datos D_j^i de longitud k_j, y por el bloque de paridad P_j^i de longitud $r_j = n_j - k_j$. En las aplicaciones bajo análisis, el código interno $\mathbb{C}_1$ es generalmente un código LDPC o TPC mientras que el código externo $\mathbb{C}_2$ es un código bloque de decodificación dura con una capacidad de corrección $t_2 = \lfloor (d_2 - 1)/2 \rfloor$ diseñada para eliminar o al menos reducir el piso de error de $\mathbb{C}_1$.

Esquema I de Concatenación de Códigos en Serie (CCS-I)

La Figura 5.1 muestra un esquema de concatenación de códigos en serie clásico al cual denotaremos como CCS-I. El proceso de codificación se

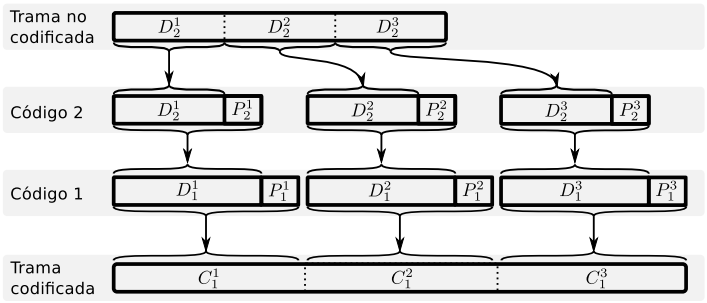


Figura 5.1: Proceso de codificación de la CCS-I.

compone de dos etapas. Primero, la *trama* sin codificar se divide en m bloques de k_2 bits denotados D_2^i para $i = 1, \ldots, m$ (ej., $m = 3$ en la Figura 5.1). Cada bloque D_2^i es codificado por $\mathbb{C}_2$ generando la palabra código C_2^i. En un segundo paso, las palabras códigos C_2^i se usan como palabras de datos del código $\mathbb{C}_1$ (i.e., $D_1^i = C_2^i$), las cuales son luego codificadas por $\mathbb{C}_1$ generando las palabras código C_1^i que serán transmitidas. Para eliminar el piso de error de $\mathbb{C}_1$ generado por el conjunto de patrones de error $\mathfrak{Q}$, $\mathbb{C}_2$ debe corregir al menos w_{max} bits. Nótese que $\mathbb{C}_2$ corregirá también los patrones de error $\omega \in \mathfrak{Q}$ con $\tilde{w}(\omega) \leq t_2$ pero podría introducir más errores sobre los patrones de error $\tilde{\mathfrak{Q}} = \{\omega \in \mathfrak{Q} : \tilde{w}(\omega) > t_2\}$. Debido a que el decodificador de $\mathbb{C}_2$ modifica a lo sumo t_2 bits, el máximo número de errores adicionales introducidos sobre los patrones de error $\tilde{\mathfrak{Q}}$ no es superior a t_2. Por lo tanto, en el peor caso, la tasa de error de información $\tilde{P}_b(\mathfrak{Q})$ se ve incrementada por un factor $(2t_2+1)/(t_2+1) < 2$. Esta última penalidad puede despreciarse a muy bajas tasas de error — por ejemplo 10^{-15} — donde la pendiente de la curva de BER vs. SNR es muy alta, lo que es típico en códigos con un desempeño cercano al límite de Shannon.

La Figura 5.2 muestra una variación del CCS-I en donde se incorpora un entrelazador entre el código $\mathbb{C}_1$ y el código $\mathbb{C}_2$. Dependiendo de la estructura de los patrones de error de $\mathbb{C}_1$, es posible diseñar un entrelazado tal que divida los patrones de error de $\mathbb{C}_1$ entre varias palabras código de $\mathbb{C}_2$. De esta forma, puede reducirse la capacidad de corrección de $\mathbb{C}_2$.

Esquema II de Concatenación de Códigos en Serie (CCS-II)

El esquema CCS-I puede generalizarse como se muestra en la Figura 5.3. Este esquema, denotado CCS-II, usa un código externo $\mathbb{C}_2$ de mayor tamaño que el usado en CCS-I con la finalidad de proteger una trama de m palabras de datos internas (ej., $m = 3$ en la Figura 5.3). Sea $t_2 = \tau \cdot w_{max}$ la capacidad de corrección de error de $\mathbb{C}_2$. Luego, el piso de error se elimina si $\tau = m$. Alternativamente, si $\tau < m$ el piso de error es reducido pero no eliminado. En este último caso, el piso de error residual $\tilde{P}_b^{(II)}(\mathfrak{Q})$ puede aproximarse con

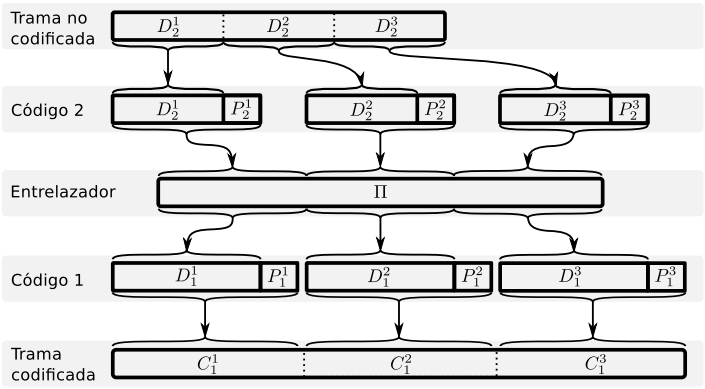


Figura 5.2: Proceso de codificación de la CCS-I con entrelazado.

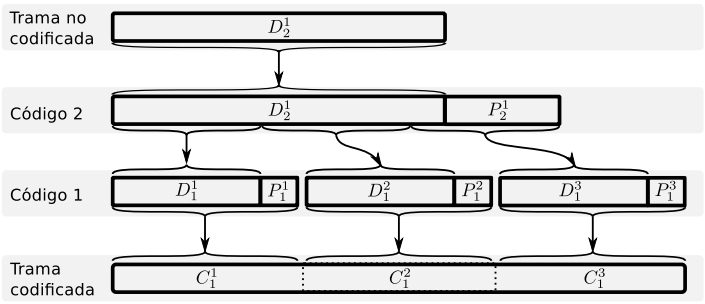


Figura 5.3: Proceso de codificación de la CCS-II.

la distribución binomial [130] según la ecuación (5.5):

$$\tilde{P}_b^{(II)}(\Omega) \approx \sum_{i=\tau+1}^{m} \frac{i \cdot w_{max}}{m \cdot k_1} \binom{m}{i} (P_w(\Omega))^i (1 - P_w(\Omega))^{m-i}. \tag{5.5}$$

Si bien los parámetros del esquema CCS-II pueden optimizarse según las características del código interno, en la mayoría de los casos de interés práctico — donde $P_w(\Omega) < 10^{-4}$, $w_{max} < 50$ y $k_1 > 500$ — es posible reducir el piso de error debajo de 10^{-17} usando $m = 20$ y $\tau = 4$; lo que muestra la generalidad y potencial del CCS-II. Además, debido a que el valor de τ necesario es significativamente menor que el valor de m, el *overhead* del código externo de SCC-II es mucho menor que en SCC-I. Sin embargo, esta última ventaja tiene como contrapartida la necesidad de implementar un código externo m-veces más largo con una capacidad de corrección τ-veces mayor que en CCS-I. Tal incremento de complejidad hace prohibitivo el uso del esquema SCC-II en la mayoría de las aplicaciones de alta velocidad.

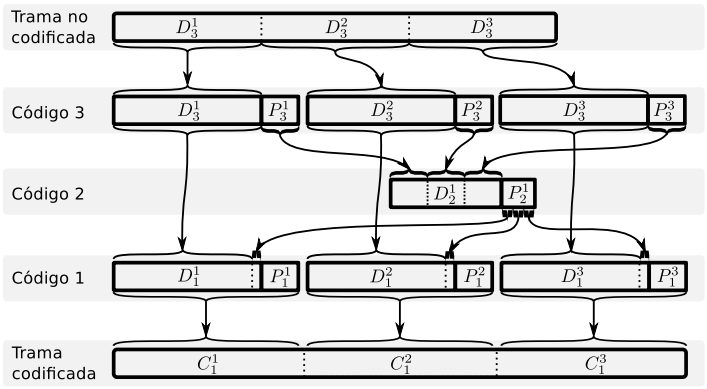


Figura 5.4: Proceso de codificación de la técnica de CCS propuesta.

5.3. Nueva Estrategia de Concatenación

Esta sección describe un nuevo esquema de CCS para combatir el piso de error. El nuevo esquema es capaz de lograr una reducción del piso de error similar a la obtenida por el esquema CCS-II. Sin embargo, el nuevo esquema está construido en base a códigos externos pequeños como en el CCS-I. De esta forma la complejidad de implementación se reduce significativamente en relación al esquema CSS-II.

En lo sucesivo, se considerará que el piso de error del código interno $\mathbb{C}_1$ es causado por el conjunto $\mathfrak{Q}$ de errores *detectables* como ocurre generalmente en los códigos LDPC (i.e., los patrones de error no son palabras código y por lo tanto su síndrome no es cero; esta restricción será eliminada luego en la Sección 5.4). La nueva CCS utiliza *dos* códigos externos pequeños, denotados como $\mathbb{C}_2$ y $\mathbb{C}_3$, para combatir el piso de error del código interno $\mathbb{C}_1$. El proceso de codificación se compone de tres pasos (véase Figura 5.4):

1. La trama no codificada se divide en m palabras de datos de k_3 bits cada una denotadas D_3^i para $i = 1, \ldots, m$ ($m = 3$ en el ejemplo de la Figura 5.4). Cada palabra de datos D_3^i es codificada por $\mathbb{C}_3$ generando los bits de paridad P_3^i.

2. Los m bloques de paridad P_3^i se agrupan en la palabra de datos D_2^1, la cual es codificada por $\mathbb{C}_2$ generando los bits de paridad P_2^1.

3. Los bits de paridad P_2^1 se dividen en m sub-bloques de igual, o casi igual, tamaño denotados como $P_2^{1,i}$ con $i = 1, \ldots, m$. Cada palabra de datos D_1^i es generada por la concatenación de la palabra de datos D_3^i y los bits de paridad $P_2^{1,i}$. Finalmente, cada palabra de datos D_1^i es codificada por $\mathbb{C}_1$ generando las palabras código C_1^i que serán transmitidas por el canal.

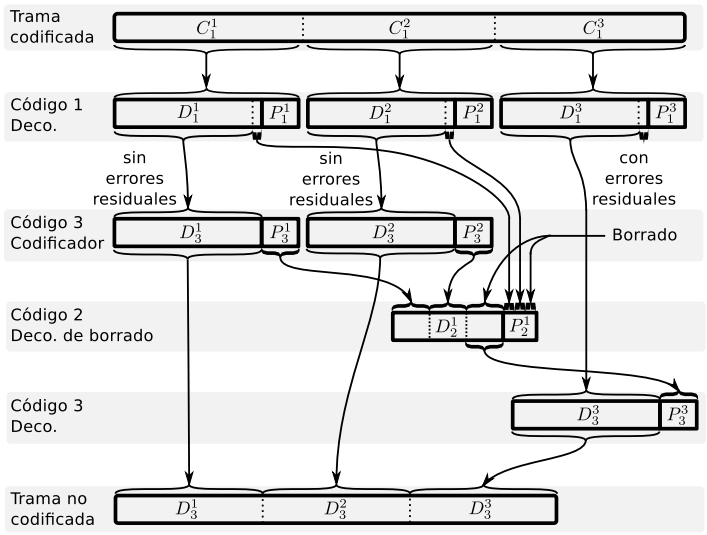


Figura 5.5: Proceso de decodificación de la técnica de CCS propuesta.

En la Figura 5.5 puede observarse un ejemplo del proceso de decodificación cuando el patrón de error tiene lugar en la tercera palabra código C_1^3. Dicho proceso se compone de los siguientes pasos:

1. Las palabras código C_1^i para $i = 1, \ldots, m$ son decodificadas y aquellas que contienen errores residuales son detectadas en base al valor del síndrome.

2. A partir de las palabras de datos D_1^i sin errores obtenidas en el paso 1, se extraen las palabras de datos D_3^i para luego codificarlas con la finalidad de recuperar los bits de paridad P_3^i.

3. Se reconstruye la palabra de datos D_2^1 a partir de los bits de paridad P_3^i generados en el paso 2, en donde los bits de paridad no disponibles — aquellos que corresponden a la palabras código con errores — son marcados como borrados (i.e., P_3^3 en el ejemplo de la Figura 5.5). Los bits de paridad P_2^1 son extraídos de las palabras de datos D_1^i que no contienen errores y aquellos bits relacionados a las palabras de datos corruptas son marcados como borrados. Luego se utiliza un decodificador de borrado en las palabras código C_2^1 para recuperar los bits de paridad P_3^i de las palabras código C_3^i con errores.

4. Finalmente, las palabras códigos C_3^i con errores residuales son corregidas.

Cell	Gate Count
Inverter (INVD0BWP35)	0,5
2-input AND (AN2D2BWP35)	2,0
2-input XOR (GXOR2D2BWP35)	4,0
2-input MUX (MUX2D2BWP35)	3,5
D Flip-Flop (DFD2BWP35)	7,5
One Bit of a RAM†	0,8

† estimado en base a la relación de área entre una 1024x16 SPSRAM y una NAND ND2D1BWP35

Cuadro 5.1: Número de compuertas equivalentes de las celdas básicas de tecnología de TSMC de 28nm.

5.3.1. Desempeño y *Overhead*

Si $\mathbb{C}_3$ corrige w_{max} bits y $\mathbb{C}_2$ recupera $\tau \cdot (n_3 - k_3) + (\tau/m) \cdot (n_2 - k_2)$ bits borrados, el esquema de CCS tiene un desempeño similar al obtenido por el esquema CCS-II. Nótese que $\tau \cdot (n_3 - k_3)$ es la cantidad de bits de paridad correspondientes a τ palabras código de $\mathbb{C}_3$ mientras que $(\tau/m) \cdot (n_2 - k_2)$ son los bits de paridad de $\mathbb{C}_2$ que pertenecen a τ palabras código de $\mathbb{C}_1$ con errores. Si $\mathbb{C}_2$ es un código con *máxima distancia de separación* (MDS) — ej. un código RS —, su capacidad de corrección de borrado es igual al tamaño de su redundancia [73], entonces se verifica:

$$n_2 - k_2 = \tau \cdot (n_3 - k_3) + \frac{\tau}{m}(n_2 - k_2), \tag{5.6}$$

que resolviendo para $n_2 - k_2$ queda

$$n_2 - k_2 = \frac{m \cdot \tau}{m - \tau} \cdot (n_3 - k_3). \tag{5.7}$$

Finalmente, el *overhead* del nuevo esquema de CCS es

$$\Theta(\tau) = \frac{n_2 - k_2}{m \cdot k_3} = \frac{\tau}{m - \tau} \left[\frac{n_3 - k_3}{k_3} \right]. \tag{5.8}$$

En base a evaluaciones numéricas de (5.8) se puede observar que para casi prácticamente todos los parámetros de interés práctico en aplicaciones de alta velocidad — $k_1 > 500$ $\tau \leq 4$, $w_{max} \leq \sqrt{k_1}/2 \leq 200$, y $m \geq 20$ — el *overhead* del esquema propuesto es menor al de los esquemas clásicos CCS-I y CCS-II. Este último resultado será nuevamente confirmado por los ejemplos de la Secciones 5.3.3 y 5.3.4 .

5.3.2. Complejidad

Como se observa en la Figura 5.6, el esquema de CCS propuesto está compuesto de tres codificadores en el lado de transmisión, uno para cada

	Block	Register	2-bits XOR	2-bits Mux	Const Mul	Mul	Gates Count
BCH	Encoder	$2t$	$2tp$	$2p$	0	0	$8tp + 15t + 7p$
	Syndrome	tq	tpq	tq	t	0	$4tpq + 2tq^2 + 7tq$
	Key Equation	$4(t+1)q$	$(2t+1)q$	$3t + 3 + tq$	0	$3t + 1$	$24tq^2 + 5{,}5tq + 22q + 28{,}5t + 26{,}5$
	Chien Search	tq	tpq	tq	tp	t	$2tpq^2 + 8tq^2 - tq + 6t$
	FIFO	$k + 5tp$	0	0	0	0	$0{,}8(k + 5tp)$
RS	Encoder	$2tq$	$2tpq$	$2pq$	$2tp$	0	$4tpq^2 + 15tq + 7pq$
	Syndrome	$2tq$	$2tpq$	$2tpq$	$2tp$	0	$4tpq^2 + 15tpq + 7pq$
	Forney Synd.	$2tq$	$2tq$	$2tq$	0	$2t$	$30tq + 2t(8q^2 - 12q + 6)$
	Erasure Locator	$2tq$	$2tq$	0	0	$2t - 1$	$23tq + (2t - 1)(8q^2 - 12q + 6)$
	Errata Evaluator	$(3t+1)pq$	$3tpq$	$2(3t-1)pq$	$2(3t-1)p$	0	$pq(31{,}5t + 12tq - 4q + 8{,}5)$
	FIFO	$(n + 4tp + 8p)q$	0	0	0	0	$0{,}8(n + 4tp + 8p)q$

donde p,q,t,n,k son el factor de paralelismo, el tamaño del cuerpo de Galois, la capacidad de corrección, la longitud del código y la dimensión del código respectivamente.

Cuadro 5.2: Complejidad de los códigos BCH binario y RS.

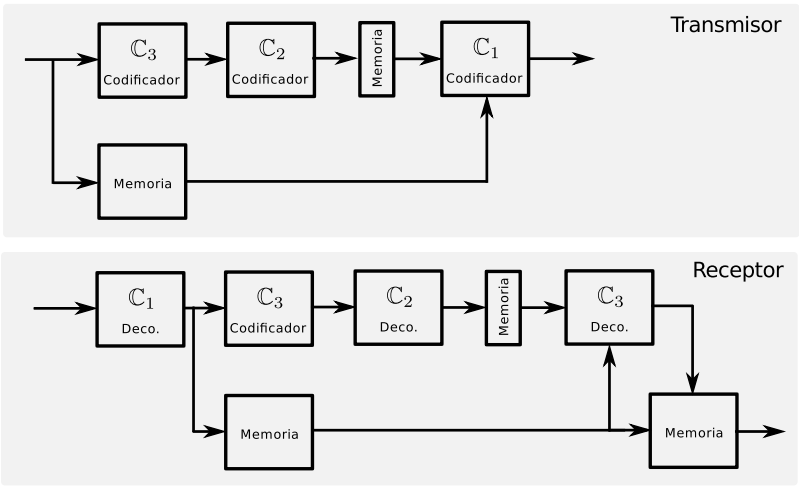


Figura 5.6: Diagrama en bloques del codificador y decodificador el esquema de CCS propuesto.

código, y del lado del receptor se compone de un codificador para el código $\mathbb{C}_3$ y tres decodificadores, nuevamente uno para cada código. Sea $\Phi(\mathbb{C}_x^e, \mathcal{T})$ y $\Phi(\mathbb{C}_x^d, \mathcal{T})$ la complejidad del codificador y decodificador del código $\mathbb{C}_x$ respectivamente cuando operan a una velocidad de $\mathcal{T}$ bits/s. En lo sucesivo, se utilizará el *número de compuertas equivalentes* de las diferentes celdas lógicas —ej. AND, XOR, etc.— como medida de la complejidad. Dicho número de compuertas equivalentes se calcula en base a las celdas de la tecnología de TSMC de 28nm descripta en la Tabla 5.1 (véase [105] para más detalles). El número estimado de compuertas lógicas de un código de RS sobre GF(2^q) y un código BCH para una implementación en paralelo basada en los algoritmos clásicos (véase [103] y [27]) se detalla en la Tabla 5.2.

La complejidad total Φ_T del esquema de CCS propuesto es

$$\begin{aligned} \Phi_T \;&\approx\; 2\Phi\big(\mathbb{C}_3^e, \mathcal{T}\big) + \Phi\big(\mathbb{C}_3^d, (\tau/m)\mathcal{T}\big) + \Phi\big(\text{Memoria}\big) \\ &+\; \Phi\big(\mathbb{C}_2^e, (r_3/k_3)\mathcal{T}\big) + \Phi\big(\mathbb{C}_2^d, (r_3/k_3)\mathcal{T}\big), \end{aligned} \tag{5.9}$$

donde el decodificador de $\mathbb{C}_3$ tiene que corregir no más de τ de las m palabras código por trama y por lo tanto su velocidad de procesamiento se reduce a $\approx (\tau/m)\mathcal{T}$. En forma similar, el decodificador de $\mathbb{C}_2$ tiene que corregir sólo una palabra código por trama lo que reduce su velocidad a $\approx (r_3/k_3)\mathcal{T}$. Finalmente, Φ(Memoria) es la complejidad de los $\sim 3mk_1$ bits de la memoria requerida para compensar la latencia de los diferentes códigos. Estas complejidades dependen de los códigos externos elegidos. Una buena relación entre desempeño y complejidad puede lograrse utilizando el esquema propuesto con un código de RS sobre GF(2^q) como $\mathbb{C}_2$ y un código BCH binario como $\mathbb{C}_3$. Como se demostrará en los siguientes ejemplos, la

Esquema	CCS-I	CCS-II	CCS Propuesto
Penalidad en SNR	0.3386 dB	0.0397 dB	0.0297 dB
Overhead	8.1081 %	0.9174 %	0.6865 %
Piso de error	None	$\approx 5 \cdot 10^{-19}$	$\approx 5 \cdot 10^{-19}$
Código Externo $\mathbb{C}_2$	BCH[1320, 1221, 19]	BCH[47520, 47088, 55]	RS[432, 396, 37]
Código Externo $\mathbb{C}_3$	None	None	BCH[1410, 1311, 19]
Complejidad Relativa	1.00	6.00	0.67

Cuadro 5.3: Comparación de desempeño y complejidad para los tres esquemas del ejemplo 1

complejidad del esquema de CCS propuesto es significativamente menor a la de los esquemas clásicos, lo cual es consecuencia del uso de códigos pequeños y de la reducción de velocidad en los decodificadores.

5.3.3. Ejemplo 1: Código Concatenado LDPC+RS+BCH

Sea $\mathbb{C}_1$ el código [2640, 1320] de Margulis [111]. Este código tiene un piso de error que comienza a $P_w(\wp) \approx 10^{-5}$. Este piso de error está dominado por *trapping-sets* (TS), en particular por TS(12,4) y TS(14,4) los cuales tienen 6 y 7 bits de información, respectivamente. Sin embargo, como se observó en [18], también hay TS de peso 15, 16, 17 y 18 bits. En lo sucesivo *atacaremos* también a estos últimos TSs mediante el diseño de un esquema de CCS para corregir patrones de error con $w_{max} = 9$ bits de información. La solución resultante de usar las diferentes técnicas de CCS analizadas se detalla a continuación:

- CCS-I: el código externo $\mathbb{C}_2$ tiene que ser un BCH[1320, 1221, 19]. Luego, la penalidad en la SNR y *overhead* adicional son $10 \cdot log_{10}(1320/1221) = 0{,}3386$ dB y $(1320 - 1221)/1221 = 8{,}11\,\%$ respectivamente.

- CCS-II: el código externo $\mathbb{C}_2$ tiene que ser un BCH[47520, 47088, 55] el cual puede corregir hasta $\tau = 3$ patrones de error de 9 bits. El piso de error no es completamente eliminado, pero se reduce a $\tilde{P}_b^{(II)}(\wp) \approx 5 \cdot 10^{-19}$ con una penalidad en la SNR y un *overhead* adicional de 0,0397 dB y 0,9174 % respectivamente.

- Nueva CCS: usando $m = 36$ y $\tau = 3$ como en la CCS-II, obteniendo de esta forma el mismo piso de error residual, $\mathbb{C}_3$ puede ser un BCH[1410, 1311, 19] binario sobre GF(2^{11}) y $\mathbb{C}_2$ puede ser un RS[432, 396, 37] sobre GF(2^9). El número de bits de paridad adicionales es $36 \cdot 9 = 324$. Estos bits de paridad introducen una penalidad en la SNR y un *overhead* adicional de 0,0297 dB y 0,6865 %, respectivamente.

La complejidad de los tres esquemas de CCS se estimó como se describe en la Sección 5.3.2. Un factor de paralelismo base $p = 160$ bits se utiliza como

referencia para lograr una velocidad de procesamiento de $\mathcal{T} \approx 100$Gb/s en tecnología CMOS de 28nm operando a una frecuencia de reloj de 625 MHz. La Tabla 5.3 muestra las complejidades relativas de los tres esquemas y sus respectivos desempeños. Nótese que:

- El desempeño del esquema CCS-II es mejor que en CCS-I a costa de una mayor complejidad de implementación como consecuencia de uso de un código BCH más grande.
- El nuevo esquema tiene un desempeño ∼0.3 dB superior al alcanzado por el esquema CCS-I con una complejidad incluso menor.
- El desempeño del esquema propuesto es similar al del esquema CCS-II. Sin embargo, la complejidad de implementación del primero es aproximadamente un orden de magnitud menor que en CCS-II.

5.3.4. Ejemplo 2: Código Concatenado LDPC+RS+RS

Como segundo ejemplo se utilizó la concatenación LDPC+RS propuesta en [116] para la próxima generación de sistemas de comunicaciones ópticas. Este esquema se compone de un código interno LDPC$[9252, 7967]$ y un código externo RS$[992, 956, 37]$. El *overhead* total es 20,5 % y la ganancia neta de codificación a un BER=10^{-15} es 10 dB. El código externo introduce una penalidad en la SNR de 0,1605 dB. A continuación se utilizará el esquema de CCS propuesto con el mismo código LDPC como código interno $\mathbb{C}_1$. Como códigos externos se utilizará al RS$[830, 794, 37]$ como el código $\mathbb{C}_3$ — i.e., una versión acortada del código de RS original —, y el RS$[1014, 936, 79]$ como código $\mathbb{C}_2$ donde $m = 26$ y $\tau = 2$. Nótese que la velocidad requerida para $\mathbb{C}_2$ es $k_3/r_3 \approx 22{,}14$ veces menor a la del RS original, luego la complejidad adicional introducida por el esquema de CCS propuesto es despreciable. De [116] se observa que la probabilidad de los patrones de error es $P_w(\mathfrak{Q}) \approx 5 \cdot 10^{-7}$. Luego, el piso de error residual del esquema propuesto es $\tilde{P}_b(\mathfrak{Q}) \approx 10^{-19}$. Por otro lado, la penalidad en la SNR y el *overhead* adicional son 0,0164 dB y 0,378 %, respectivamente. Lo anterior resulta en un incremento de la ganancia neta de codificación de 10 dB a 10,1441 dB y en una reducción del *overhead* total desde 20,5 % a 16,568 %. Estas mejoras no sólo reducen los requerimientos de SNR del sistema sino que también mejoran la eficiencia espectral y bajan la potencia de disipación[2].

5.3.5. Ejemplo 3: Código Concatenado LDPC+SPC+BCH

La mayoría de los códigos LDPC propuestos para aplicaciones de fibras ópticas tienen un piso de error bajo — usualmente menor a 10^{-10} — que

[2] Esto se debe a que la frecuencia de muestreo se reduce como consecuencia del mejor *overhead*

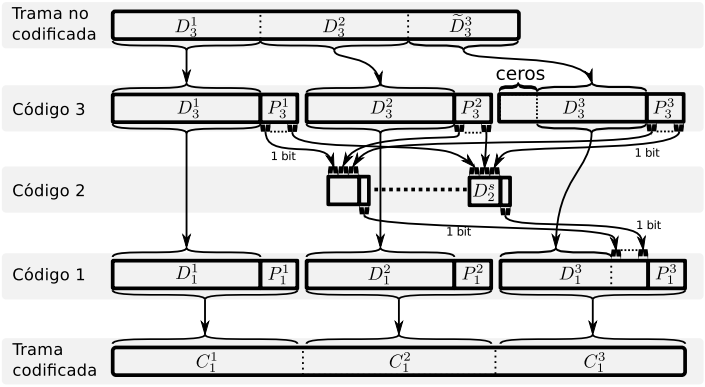


Figura 5.7: Proceso de codificación del esquema de CCS propuesto optimizado para $\tau = 1$.

puede reducirse a tasas de errores por debajo de 10^{-15} corrigiendo un sólo patrón de error por trama (i.e., con $\tau = 1$). En estos casos, es posible reducir el *overhead* y la complejidad de implementación del esquema propuesto mediante el uso de $n_3 - k_3$ *códigos de chequeo de paridad simple* (SPC[3]) como el código externo $\mathbb{C}_2$. El proceso de codificación resultante, el cual es descripto en la Figura 5.7, se compone de los siguientes pasos:

1. La trama no codificada se divide en m bloques. Las primeras $m-1$ palabras de datos D_3^i para $i = 1, \ldots, m-1$ se corresponden con los primeros $m-1$ bloques. La última palabra de datos D_3^m es la concatenación de $(n_3 - k_3)$ ceros y el último bloque $\widetilde{D}_3^m$, el cual contiene $k_3 - (n_3 - k_3)$ bits.

2. Cada palabra de datos D_3^i se codifica con $\mathbb{C}_3$ generando los bits de paridad P_3^i.

3. Para $i = 1, \ldots, n_3 - k_3$ las palabras de datos D_2^i de m bits se forman como la concatenación de los i-ésimos bits de paridad de P_3^j para $j = 1, \ldots, m$. Luego, cada D_2^i es codificado por el código SPC.

4. Las palabras de datos del código $\mathbb{C}_1$ son $D_1^i = D_3^i$ para $i = 1, \ldots, m-1$. La última palabra de datos D_1^m es la concatenación de $\widetilde{D}_3^m$ y los $n_3 - k_3$ bits de paridad generados en el paso 3. Finalmente, cada palabra de datos D_1^i es codificada por $\mathbb{C}_1$.

El proceso de decodificación es similar al del esquema original; véase la Figura 5.5. Los principales beneficios del esquema optimizado son:

- La complejidad de implementación del codificador y decodificador del código externo $\mathbb{C}_2$ es muy baja debido a que ambos se pueden

[3]Por sus siglas en Inglés para "single parity check".

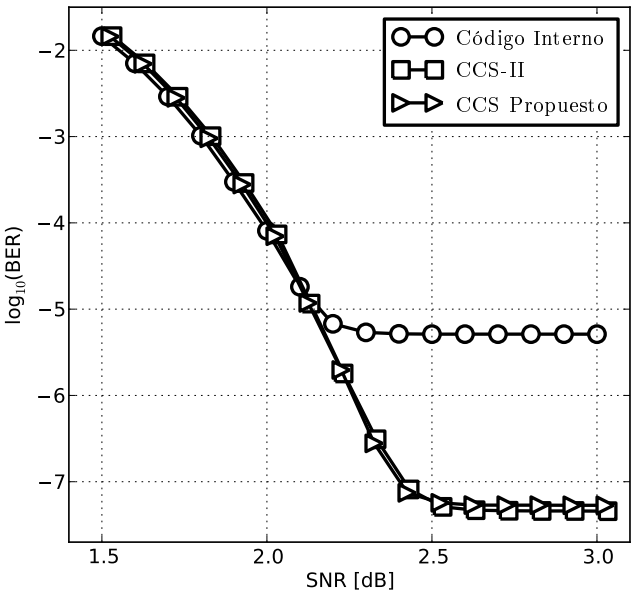


Figura 5.8: BER vs. SNR de la CCS-II y la CCS propuesta con $\tau = 1$.

implementar en forma recursiva utilizando $n_3 - k_3$ compuertas XOR y Flip-Flops. De esta forma $\Phi\big(\mathbb{C}_2^e\big) + \Phi\big(\mathbb{C}_2^d\big) = 2(n_3 - k_3)(\Phi(\text{XOR}) + \Phi(\text{FlipFlop})) = 23 \cdot (n_3 - k_3)$ compuestas.

- El *overhead* introducido por los códigos externos es reducido de $[(n_3 - k_3)/k_3]/(m-1)$ (ver ecuación (5.8) con $\tau = 1$) a $[(n_3 - k_3)/k_3]/m$ debido a que los bits de paridad de $\mathbb{C}_2$ que pueden tener errores no necesitan ser borrados.

- La latencia del codificador se reduce debido a que los bits de paridad P_2^i son transmitidos en la última palabra código de $\mathbb{C}_1$; i.e., dichos bits de paridad pueden calcularse mientras las palabras código de $\mathbb{C}_1$ son transmitidas. Lo anterior también reduce el tamaño de la memoria de $\approx 3mk_1$ a $\approx 2mk_1$ bits.

La Figura 5.8 muestra el desempeño del esquema de CCS optimizado para $\tau = 1$. Como código interno $\mathbb{C}_1$ se utiliza el código LDPC$[2640, 1320]$ de Margulis con $m = 10$. Por limitaciones en la velocidad de simulación, se inserta a la salida del decodificador un piso de error artificial con probabilidad $P_w(\mathfrak{Q}) = 10^{-3}$. Al igual que el piso de error real, los patrones de error insertados están formados por 7 bits de información y 7 bits de paridad con error. En dicha figura se observa el desempeño del código interno $\mathbb{C}_1$ solo (círculos), del esquema CCS-II (cuadrados) y del nuevo esquema de CCS (triángulos). Para el CCS-II, el código externo $\mathbb{C}_2$ es un BCH$[13200, 13102, 15]$. Por otro lado, para el esquema propuesto $\mathbb{C}_3$ es un BCH$[1397, 1320, 15]$ mientras que $\mathbb{C}_2$ es un SPC$[11, 10, 2]$. Nótese

que $\mathbb{C}_1$ tiene un piso de error a un BER $\approx (7/1320)P_w(\Omega) = 5{,}3 \cdot 10^{-6}$. Este piso de error es reducido por el código externo a un BER $\approx 4{,}73 \cdot 10^{-8}$. Este desempeño es similar al alcanzado por el esquema CCS-II. Sin embargo, cabe remarcar que la técnica propuesta utiliza códigos externos más pequeños reduciendo de esta forma la complejidad en circuitos integrados. Particularmente, usando un factor de paralelismo $p = 160$ para alcanzar una velocidad $\mathcal{T} \approx 100$Gb/s en tecnología CMOS de 28nm operando a una frecuencia de reloj de 625 MHz como en el Ejemplo 1 (véase Sección 5.3.3), la complejidad del esquema CCS-II es ≈ 575 K compuertas mientras que la complejidad del nuevo esquema es ≈ 107 K compuertas, i.e., 5,38 *veces menor*!

5.4. Reducción del Piso de Error de Códigos TPC

El nuevo esquema de CCS desarrollado en esta Tesis puede extenderse para mitigar el piso de error causado por palabras código de bajo peso; i.e., patrones de error no detectables. Tal propiedad es particularmente útil en la decodificación de códigos turbo tales como códigos turbo producto (TPC). En el nuevo esquema se utilizará un subconjunto de g bits de paridad de $\mathbb{C}_3$ con la finalidad de detectar las palabras código que tienen errores residuales después de ser decodificadas por el código interno[4].

5.4.1. Esquema CCS Generalizado

La Figura 5.9 describe el proceso de codificación del esquema de CCS generalizado. A diferencia del esquema anterior, en su versión generalizada un subconjunto de g bits de paridad de P_3^i, denotado $\hat{P}_3^i$, no es codificado por $\mathbb{C}_2$. En su lugar, este subconjunto de bits es transmitido como parte de la palabra de datos D_1^i del código interno $\mathbb{C}_1$. En el proceso de decodificación, $\hat{P}_3^i$ se utiliza para detectar las palabras código de $\mathbb{C}_1$ que están corruptas. El proceso de decodificación comienza con la decodificación de las m palabras código recibidas C_1^i. Luego, la palabra de datos D_1^i es extraída de C_1^i. Para cada palabra de datos D_1^i, la palabra de datos D_3^i es extraída y parcialmente codificada con $\mathbb{C}_3$ para regenerar sólo los bits de paridad $\hat{P}_3^i$. Si estos bits de paridad regenerados no son iguales a sus correspondientes bits dentro de D_1^i, la palabra de datos es marcada como corrupta. Una vez que todas las palabras de datos D_1^i corruptas son identificadas, el resto del proceso de decodificación continúa de la misma forma que en el esquema original.

[4]Es también posible realizar tal detección mediante el agregado de un *código de detección de errores* tal como un *código de redundancia cíclica* denotado CRC — por sus siglas en Inglés para *cyclic redundancy check*—. Otra opción consiste en reemplazar el decodificador de borrado de $\mathbb{C}_2$ por un decodificador para corrección de errores con el costo de incrementar la distancia mínima de $\mathbb{C}_2$.

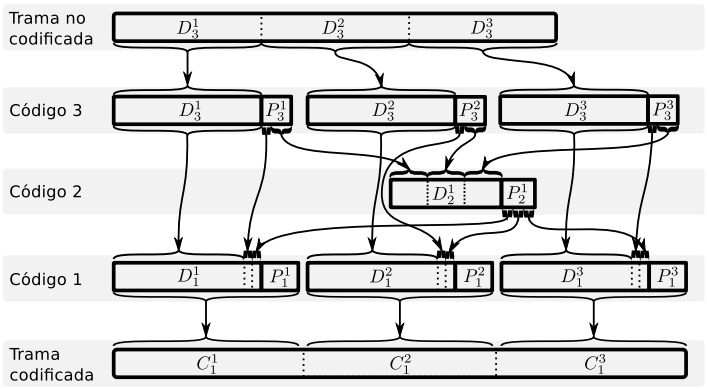


Figura 5.9: Proceso de codificación del esquema CCS generalizado.

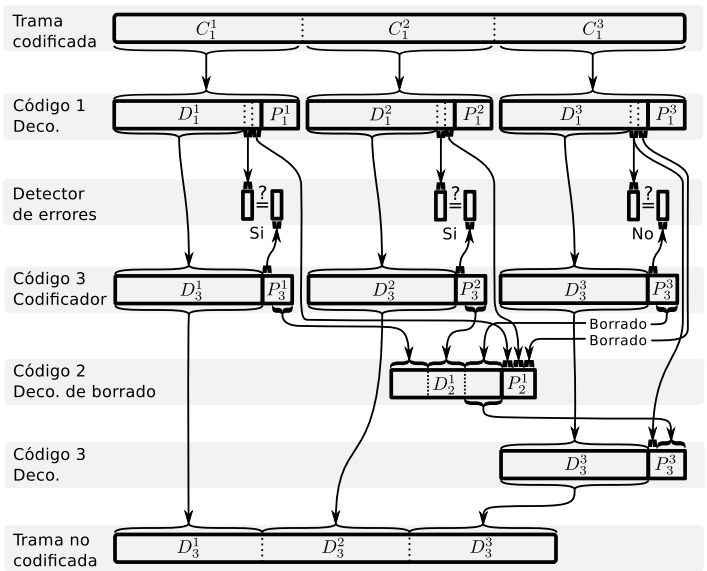


Figura 5.10: Proceso de decodificación del esquema CCS generalizado.

Este esquema generalizado puede tener un piso de error residual causado por la ocurrencia de más de t patrones de error pertenecientes a $\mathbb{Q}$ dentro de la misma trama. Este piso de error, denotado como $\tilde{P}_b^{(1)}(\mathbb{Q})$, puede estimarse con la ecuación (5.5) al igual que en el esquema original. Sin embargo, en el nuevo esquema también es posible tener un piso de error causado por patrones de error que no son detectados por los g bits de paridad $\hat{P}_3^i$. La probabilidad $P_w^{(2)}(\mathbb{Q})$ de que un patrón de error no detectable $\omega \in \mathbb{Q}$ ocurra en la palabra código C_1^i se puede calcular como:

$$P_w^{(2)}(\mathbb{Q}) = \sum_{\omega \in \mathbb{Q}} I_v \left(\hat{P}_3(\omega) = 0 \right) p(\omega) \tag{5.10}$$

donde $\hat{P}_3(\omega)$ son los primeros g bits de paridad de $\mathbb{C}_3$ asociados al patrón de error $\omega \in \mathbb{Q}$, mientras que $I_v(X)$ es el *operador de Iverson* el cual vale 1 si la sentencia X es verdadera y vale 0 si es falsa. Debido a que los patrones de error de diferentes palabras código son independientes, la probabilidad de al menos un patrón de error no detectable en la trama puede calcularse en base a la distribución binomial como indica la ecuación (5.11):

$$P_f^{(2)}(\mathbb{Q}) = \sum_{i=1}^{m} \binom{m}{i} (P_w^{(2)}(\mathbb{Q}))^i (1 - P_w^{(2)}(\mathbb{Q}))^{m-i}, \tag{5.11}$$

mientras que la tasa de error puede estimarse como indica la ecuación (5.12):

$$\tilde{P}_b^{(2)}(\mathbb{Q}) \approx \sum_{i=1}^{m} \binom{m}{i} \frac{i \cdot w_{max}}{m \cdot k_1} (P_w^{(2)}(\mathbb{Q}))^i (1 - P_w^{(2)}(\mathbb{Q}))^{m-i}. \tag{5.12}$$

5.4.2. Esquema CCS Generalizado Aplicado a un TPC

Como se mencionó en la introducción, potentes códigos de corrección de errores deben diseñarse para satisfacer los requerimientos de los futuros sistemas de transmisión de datos a alta velocidad. Por ejemplo, una ganancia de codificación neta >10 dB a un BER de 10^{-15} y un *overhead* de $\sim 20\,\%$ es necesario para la próxima generación de OTN [125]. Para alcanzar estos requerimientos, varios códigos LDPC y TPC han sido propuestos en la literatura (véase por ejemplo [125] y las referencia allí citadas). En particular, códigos TPC basados en dos BCH con una capacidad de corrección ≥ 2, denotados TPC-BCH, y una longitud total ≥ 32 Kbits han sido usados en sistemas de alta velocidad para proveer una relación de compromiso aceptable entre desempeño y complejidad [4] [38] [117].

A continuación, se utilizará el esquema de CCS generalizado para permitir el uso de códigos TPC de muy alto piso de error — los cuales tienen la ventaja de poseer baja complejidad — en aplicaciones que requieran una muy baja tasa de error. Se demostrará que un código TPC basado en

dos *códigos de Hamming extendidos*, denotado TPC-EH, con un tamaño de bloque de sólo 8192 bits y una distancia mínima de 16 bits, puede combinarse con el esquema propuesto alcanzando de esta forma una ganancia neta de codificación de $\sim 11{,}2$ dB a un BER$= 10^{-15}$ con un *overhead* total de $\sim 22\,\%$ y un piso de error a $\sim 7 \cdot 10^{-17}$. Cabe destacar que este desempeño es:

(*i*) 0,45 dB superior al alcanzado por el código TPC propuesto en [117], el cual está basado en el producto BCH(144,128,5)×BCH(256,239,6) con una longitud de 36864 bits y una distancia mínima de 30 bits,

(*ii*) 0,4 dB superior que el del TPC basado en el producto BCH(128,113,6)×BCH(256,239,6) propuesto en [4] el cual tiene una longitud de 32768 bits y una distancia mínima de 36 bits, y

(*iii*) 0,4 dB superior al desempeño alcanzado por el esquema de tres códigos concatenados propuesto en [125].

Además, se espera que la complejidad de implementación del TPC-EH con el esquema de CCS propuesto sea menor a la de los esquemas previos basados en un TPC no concatenado. Esto último se debe principalmente a que los códigos componentes del TPC-EH son mucho más simples que los utilizado en los esquemas TPC-BCH no concatenados. En particular, este último requiere códigos BCH grandes y con una capacidad de corrección mayor a la del código EH con la finalidad de evitar un problema de piso de error. Por tales motivos, el TPC-EH combinado con la técnica propuesta es una excelente opción para la nuevas generaciones del redes de comunicación por fibras ópticas [125].

5.4.3. Código Concatenado TPC-EH + BCH + RS

Sea el código interno $\mathbb{C}_1$ un TPC basado en dos códigos de Hamming extendidos con parámetros $[128, 120, 4]$ y $[64, 57, 4]$. Sea A_w el número de palabras código de $\mathbb{C}_1$ con peso w. Para $w < 28$, A_w se puede calcular según se describe en [159] obteniendo

$$A_w = \begin{cases} 1, & \text{si } w = 0 \\ 888943104, & \text{si } w = 16 \\ 7154214100992, & \text{si } w = 24 \\ 0, & \text{otro } w < 28 \end{cases} \tag{5.13}$$

La Figura 5.4.3 muestra la curva de BER vs SNR de este código cuando se decodifica con 8 iteraciones turbo entre los dos códigos componentes. El decodificador de *máxima probabilidad a posteriori* (MAP[5]) propuesto en [12] se utiliza para decodificar ambos códigos EH. Como se observa en la Figura 5.4.3, $\mathbb{C}_1$ tiene un piso de error a un BER $\leq 10^{-7}$ el cual es causado

[5] Por sus siglas en Inglés para *Maximum A posteriori Probability*

por las A_{16} palabras código de peso mínimo $w_{min} = 16$ [33]. La Figura 5.4.3 también muestra la estimación de BER basada en la cota de unión [33] para las palabras de peso 16 y 24.

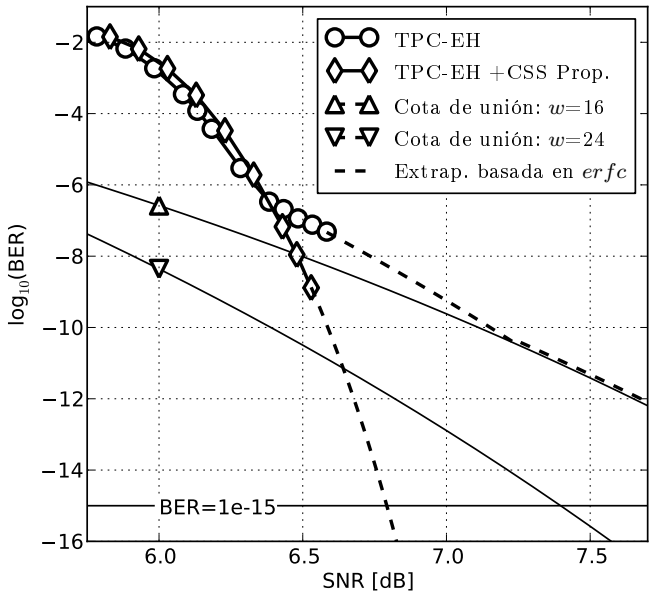
Para poder evitar un piso de error a un BER $= 10^{-15}$, de la Figura 5.4.3 se infiere que los errores debidos a palabras código de peso 24 o menor deben corregirse. Además, debido a que se utiliza un proceso iterativo sub-óptimo para la decodificación, se tienen que analizar también los errores debidos a *stopping-sets* que no sean palabras código. Esto último se puede calcular como se describe en [136] dando como resultado que el mínimo *stopping-set* que no sea palabra código tiene peso 24 y multiplicidad 81782765568, ~ 87 veces menor que A_{24}. Por otro lado, una trama compuesta de $m = 19$ palabras código acortadas TPC$(7859, 6507)$ se requiere para acomodar los 122368 bits de una *Unidad de datos del Canal Óptico* (ODU[6]) de la trama OTN G.709 [163] más los bits de paridad de los códigos externos. Para reducir el piso de error por debajo de 10^{-15} se requiere una capacidad de corrección de $\tau = 2$ patrones de error de peso ≤ 24 y una detección de palabras códigos corruptas basada en $g = 32$ bits de paridad. El BCH$[6753, 6441, 49]$ sobre GF(2^{13}) con una capacidad de corrección de 24 bits es usado como código $\mathbb{C}_3$, mientras que el código $\mathbb{C}_2$ es un código RS$[596, 532, 65]$ sobre GF(2^{10}). El *overhead* total introducido por $\mathbb{C}_2$ y $\mathbb{C}_3$ es $\approx 1{,}02\,\%$. Este último genera una penalidad en la SNR de 0,044 dB. La tasa de error residual causada por la ocurrencia de más de τ patrones de error en una misma trama es $\tilde{P}_b^{(1)}(\Omega) \approx 7 \cdot 10^{-17}$, mientras que la tasa de error causada por patrones de error no detectables es $\tilde{P}_b^{(2)}(\Omega) \approx 4{,}3 \cdot 10^{-18}$. Los valores previos han sido derivados en base a las ecuaciones (5.5) y (5.12), respectivamente, donde $P_w(\Omega) \approx 5 \cdot 10^{-6}$ fue obtenido a partir de la simulación por computadora a una SNR $= 6{,}7$ dB.

El piso de error del TPC-EH también puede reducirse utilizando los esquemas clásicos CCS-I y CCS-II. Sin embargo, como se muestra a continuación, el esquema propuesto provee mejor relación entre desempeño y complejidad que estos últimos:

- El esquema CCS-I requiere una trama de $m = 19$ palabras código TPC$(8105, 6753)$ protegidas por m palabras código de un BCH$[6753, 6441, 49]$. El *overhead* total es $25{,}8\,\%$, la ganancia neta de codificación es 11,05 dB y la complejidad es 2,02 veces mayor que la del esquema propuesto. Esto es, el esquema propuesto tiene mejor eficiencia espectral, un desempeño 0,15 dB mayor y menor complejidad que el esquema CCS-I.

- El esquema CCS-II requiere una trama de $m = 19$ palabras código TPC$(7836, 6484)$ combinadas con un BCH$[123196, 122380, 97]$. En forma similar al esquema propuesto, el *overhead* total es $21{,}7\,\%$ y la

[6] Por sus siglas en Inglés para *Optical Channel data Unit*



ganancia neta de codificación es 11,2 dB. Sin embargo, la complejidad es 6,73 veces mayor. Esto último representa una ventaja significativa en favor del esquema propuesto.

De lo anterior se evidencian las ventajas del esquema propuesto en la implementación de códigos de corrección de errores para aplicaciones de alta velocidad. En particular, el esquema de codificación TPC-EH+RS+BCH aquí descripto representa una alternativa de baja complejidad respecto al esquema que utiliza un TPC no concatenado basado en códigos BCH con capacidad de corrección ≥ 2 [4] [117] [125] debido a que este último tiene una mayor complejidad de implementación que el código EH.

5.5. Conclusión

La nueva estrategia de concatenación de códigos para reducir el piso de error posee ventajas tanto en términos de desempeño como de complejidad frente a los esquemas clásicos. Tal ventaja radica en el uso de dos códigos externos pequeños que aprovechan la independencia de los patrones de error entre diferentes palabras código para obtener un desempeño equivalente al de un código más grande. El esquema propuesto puede utilizarse para reducir el piso de error causado tanto por patrones detectables como no detectables. En particular, el nuevo esquema abre las puertas al uso de códigos de baja complejidad que actualmente no se utilizan por su alto piso de error como es el caso de códigos TPC basados en códigos de Hamming.

Capítulo 6

Conclusiones

El mundo exige resultados. No le cuentes a otros tus dolores del parto. Muéstrales al niño.

Indira Gandhi.

6.1. Discusión Final

A partir del trabajo de Shannon de 1948, la búsqueda de códigos de corrección de errores eficientes se ha convertido en una área importante de desarrollo científico e industrial. En los últimos años, tal desarrollo culminó en lo que hoy conocemos como códigos basados en grafos y en particular en los códigos LDPC. Estos códigos mostraron tener el potencial necesario para alcanzar el máximo desempeño teórico posible con una complejidad algorítmica moderada. Por tal motivo, han comenzado a utilizarse en forma estándar en las más diversas ramas de las comunicaciones digitales, entre las cuales cabe destacar a las comunicaciones inalámbricas. Sin embargo, existen problemas asociados a los mismos que dificultan su aplicación en aquellas áreas que requieren tasa de error muy baja y velocidad muy alta como ocurre en el caso de las comunicaciones ópticas y los dispositivos de almacenamiento de información. Estos problemas son: (*i*) una pérdida de desempeño observable sólo a tasas de errores muy bajas que es conocida como *piso de error* y (*ii*) la ausencia de una arquitectura de implementación estándar y eficiente. La presente Tesis propone soluciones a ambos problemas ayudando a abrir el camino para el uso de dichos códigos en las aplicaciones mencionadas.

En lo referente al problema del piso de error, se proponen dos soluciones independientes pero que pueden combinarse de ser necesario. La primera solución ataca el problema desde dentro del código en cuestión tanto a nivel de la matriz de paridad como del algoritmo de decodificación. Se propone un método de construcción de la matriz de paridad que reduce los ciclos de corta longitud. Este método se combina con un nuevo algoritmo de post-procesamiento basado en cuantización adaptiva. Este último algoritmo estuvo motivado por la observación (mediante la técnica de *importance sampling*) de que el nivel del piso de error es fuertemente afectado por la saturación que se produce en el nodo de variable. La comparación de dicho algoritmo con su rival, propuesto por Zhang en 2008, muestra la superioridad del primero. Estas soluciones permitieron la construcción de un código LDPC de alta ganancia y sin piso de error detectable hasta al menos 10^{-16}. Dicho código confirma los resultados que ofrece la metodología propuesta para el diseño de códigos LDPC para aplicaciones que requieren muy bajo piso de error.

La segunda solución al problema del piso de error ataca a este último en forma externa al código en cuestión, mediante la propuesta de un nuevo esquema de códigos concatenados. Este nuevo esquema reduce significativamente la complejidad de implementación en comparación con las técnicas existentes. Resultados numéricos indican una reducción de complejidad que puede alcanzar un orden de magnitud en comparación con las soluciones existentes. Este beneficio se obtiene como consecuencia de una nueva estrategia de codificación y decodificación basada en la combinación

de dos códigos externos que trabajan en forma conjunta. Se propone además una generalización de dicho esquema que permite reducir el piso de error causado por palabras código de bajo peso. Esto último permite combatir el piso de error no sólo en códigos LDPC sino también en códigos turbo como TPC. Como ejemplo se propone un sistema de codificación basado en un código TPC de baja complejidad que por si sólo tiene un piso de error a un BER de 10^{-7} pero cuando se combina con el esquema propuesto en esta Tesis alcanza una ganancia neta de $\sim$ 11,2 dB a un BER= 10^{-15} con un *overhead* total de $\sim 22\,\%$ y un piso de error a $\sim 7 \cdot 10^{-17}$. Este esquema supera en 0,4 dB a códigos equivalentes propuestos previamente.

En relación al segundo problema, se propone una arquitectura de implementación del decodificador LDPC para la familia de códigos que cumplen la *partición regular por columnas*. Dicha partición permite reducir el número de interconexiones al sólo tener que trabajar con una pequeña parte de la matriz de paridad. Además, combinar este esquema con múltiples palabras código entrelazadas a nivel de iteraciones hace posible utilizar pipeline y lograr al mismo tiempo un procesamiento JaT. Como optimizaciones adicionales se propone la división y serialización del nodo de chequeo y el uso de cuantización cuasi uniforme en combinación con la aplicación del factor de corrección del algoritmo SMSA en el nodo variable y no en el nodo de chequeo. Esto último permite reducir aún más el nivel de interconexiones y la potencia. Finalmente, como prueba de la eficiencia de la arquitectura propuesta, se implementó la misma en tecnología CMOS de 28nm logrando una velocidad de procesamiento de 64 Gb/s sobre un área de 19.6 mm^2 y con una potencia de 4.7 Watts. Dicho trabajo representa el primer código LDPC no concatenado para comunicaciones ópticas que demostró ser implementable con tecnología CMOS de 28nm y que posee una ganancia de codificación neta igual a 11.3 dB a un BER de 10^{-15} sin piso de error verificado mediante emulación en FPGA.

Las soluciones presentadas simplifican enormemente el uso de los códigos LDPC y similares en aplicaciones que requieran una muy baja tasa de error y/o alta velocidad. En particular, tales contribuciones muestran que es posible el uso de códigos LDPC en comunicaciones por fibra óptica.

6.2. Caminos a Seguir

El principal problema que aún queda abierto en relación a los códigos LDPC es la estimación de su desempeño a baja tasa de error, ya sea mediante un método puramente analítico o un método híbrido asistido por simulación en computadora. En particular, tal método debería ser capaz de predecir fácilmente el piso de error dada cualquier matriz de paridad y variante del algoritmo de decodificación. Las ventajas que proporcionaría disponer de una herramienta con estas cualidades serían enormes para el diseño de códigos

LDPC. Esto último sería de importantísimo valor incluso si sólo es aplicable a canales sin memoria con ruido AWGN.

Un segundo camino a explorar es la extensión de todo lo propuesto en esta Tesis para códigos LDPC no binarios y códigos LDPC generalizados. Es decir para aquellos códigos en los cuales las filas de la matriz de paridad no representan un código binario de chequeo de paridad simple, sino un código arbitrario binario o no-binario. Como punto de partida en relación a esto último se puede mencionar como caso particular al código TPC[1] basado en dos códigos de Hamming extendidos que se propuso en el Capítulo 5 como aplicación del nuevo esquema de concatenación de códigos.

Finalmente, otro camino para mencionar es la adaptación de la técnica de códigos concatenados propuesta en el Capítulo 5 a códigos internos continuos, i.e. a aquellos códigos que no trabajan en bloques, como por ejemplo los códigos convolucionales. Una opción es utilizar los códigos externos tal cual son propuestos en el Capítulo 5, y realizar un análisis de su desempeño bajo estas circunstancias. Otra opción es analizar la posibilidad de extender la idea propuesta utilizando códigos externos continuos.

[1] Notar que un código TPC puede ser interpretado como un caso particular de un código LDPC generalizado.

Capítulo 7

Publicaciones y Patentes

La lectura hace al hombre completo; la conversación, ágil, y el escribir, preciso.

Sir Francis Bacon.

7.1. Publicaciones y Patentes

A continuación se incluyen en orden cronológico copias de los papers y patentes derivados del presente trabajo de tesis. Dichas publicaciones se pueden dividir en tres grupos a saber:

1. Aquellas directamente relacionadas al trabajo descripto en el presente manuscrito y que son de mi autoría. Este grupo se compone de los papers y patentes:

   2013 Damián A. Morero and Mario R. Hueda, "Novel Serial Code Concatenation Strategies for Error Floor Mitigation of Low-Density Parity-Check and Turbo Product Codes", Can. J. Elect. Comput. Eng., Vol. 36, No. 2, IEEE, Spring 2013

   2012 Damián Alfonso Morero et al., "Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks", U.S. Patent Application No. 13/406,452, February 27, 2012

   2012 Damián A. Morero and Mario R. Hueda, "Efficient Concatenated Coding Schemes for Error Floor Reduction of LDPC and Turbo Product Codes", Global Communications Conference (Globecom), IEEE, Dec. 2012

   2011 Damián A. Morero, M. Alejandro Castrillon, Facundo A. Ramos, Teodoro A. Goette, Oscar E. Agazzi, and Mario R. Hueda, "*Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks*", Global Communications Conference (Globecom), IEEE, Dec. 2011

   2008 Damián Morero, Graciela Corral Briones and Mario Hueda, "*Parallel architecture for decoding LDPC codes on high speed communication systems*", Argentine School of Micro-Nanoelectronics, Technology and Applications, IEEE, Sept 2008

2. Aquellas que son de mi autoría y forman parte del cuerpo de investigación de mi trabajo de doctorado pero que no son detalladas en el presente manuscrito. El tema central de estas publicaciones está orientado a esquemas de detección y decodificación turbo sobre canales con interferencia intersímbolo como ocurre en aplicaciones de grabación magnética. Estos esquemas están estrechamente relacionados a los códigos LDPC y a los problemas analizados. Sin embargo, su descripción y análisis son omitidos ya que resultarían en un manuscrito demasiado extenso. Este grupo se compone de los papers:

   2011 Damián A. Morero, Graciela Corral-Briones, Carmen Rodriguez, Mario R. Hueda, "*High-Rate Short-Block LDPC Codes for Iterative Decoding with Applications to High-Density Magnetic Recording Channels*", arXiv, Apr 2011

2010 Damián A. Morero and Mario R. Hueda, "*Performance of Euclidean-Metric MLSD Receiver in the Presence of Channel Mismatch Caused by Nongaussian Noise*", International Conference on Communications, IEEE, May 2010

2009 Martín I. del Barco, Gabriel N. Maggio, Damián A. Morero, Javier Fernandez, Facundo Ramos, Hugo S. Carrer, and Mario R. Hueda, "*FPGA Implementation of High-Speed Parallel Maximum a Posteriori (MAP) Decoders*", Argentine School of Micro-Nanoelectronics, Technology and Applications, IEEE, Sept 2009

3. Aquellas en que no soy primer autor pero en las que colaboré y motivé su realización. Dichas publicaciones conforman ejemplos de la influencia que tuvo el presente trabajo de tesis tanto en la Universidad Nacional de Córdoba como en la Universidad de Buenos Aires y tanto a nivel de grado como de post-grado. Este grupo se compone de los papers:

2012 Mario A. Castrillon, Damián A. Morero, and Mario R. Hueda, "A New Cycle Slip Compensation Technique for Ultra High Speed Coherent Optical Communications", Photonics Conference, IEEE, Sep. 2012

2012 Mario A. Castrillon, Damián A. Morero, and Mario R. Hueda, "Joint Demapping and Decoding for DQPSK Optical Coherent Receivers", arXiv, Jun. 2012

2012 Fernando Gutierrez, Graciela Corral-Briones and Damián Morero, "*FPGA Implementation of the Parity Check Node for Min-Sum LDPC Decoders*", VIII Southern Programmable Logic Conference (SPL 2012), IEEE, March 2012

2011 Juan Manuel López Simao, Damián Morero and Cecilia Galarza, "*Simulador para códigos QC-LDPC no binarios*", RPIC 2011

[Copia del paper presentado en EAMTA-2008, pg.1/4]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2008

Parallel Architecture for Decoding LDPC Codes on High Speed Communication Systems

Damián A. Morero, Graciela Corral-Briones, and Mario R. Hueda

Digital Communications Research Laboratory - National University of Cordoba - CONICET
Av. Vélez Sarsfield 1611 - Córdoba (X5016GCA) - Argentina
Emails: dmorero, gcorral, mhueda@com.uncor.edu

***Abstract*—This paper presents a novel parallel architecture for decoding LDPC codes. The proposed architecture has low memory and interconnection requirements, becoming attractive for high speed applications such as fiber optic communications and high density magnetic recording. As an example, the implementation on an FPGA of a TPC/SPC code using the proposed architecture will also be described.**

I. Introduction

The use of iterative decoders based on Low Density Parity Check (LDPC) codes has allowed to reach information rates close to Shannon's channel capacity [1]. SISO (*Soft Input/Soft Output*) decoding of LDPC codes using the Sum-Product Algorithm (SPA) requires approximately one order of magnitude less calculations than equivalent Turbo Codes [2]. Nevertheless, the implementation complexity of SISO LDPC detectors is one of the main obstacles that conditions its viability on commercial integrated circuits. High interconnection complexity and important amounts of memory are required. This issue becomes more important on high-speed applications where parallel processing schemes are needed.

Different low-complexity architectures for implementing LDPC codes have been proposed. [3] describes one that has as drawback a degradation on the decoder performance due to a simplification on the SPA algorithm. Architectures described in [2] y [4] have high requirements of memory and interconnection. [5] y [6] show architectures focused on implementing LDPC codes with particular structural properties. The present paper introduces a new architecture for implementing a wide family of LDPC codes. It operates in a parallel fashion without requiring approximations on the SPA algorithm, and requires a reduced amount of memory and interconnection complexity. These features make the architecture desirable for implementing on integrated circuits.

The rest of the paper is organized as follows. Section II gives a brief introduction to LDPC codes. Section III explains the decoding algorithm. Section IV describes the proposed novel architecture that implements the algorithm explained on the previous section. Section V shows the results after implementing on an FPGA the architecture proposed on Section IV. Finally, section VI presents the conclusions.

II. Low-Density Parity Check Codes

An LDPC code is a linear block code defined by a sparse[1] parity check matrix H, of dimensions $(m \times n)$. This matrix defines a code of length n and dimension k, where $k = n - \text{rank}(H)$. A parity check matrix H has a bipartite graph associated with it, called Tannel graph (TG), which fully characterizes the code. The Tanner graph is composed of two types of nodes: variable nodes v_i and check nodes c_i. The nodes v_i and c_j represent the i-th coded bit or column of H, and the j-th check equation or row of H respectively. In a TG, connections exist only between a variable node and a check node. v_i is connected to c_j if and only if $H_{j,i} \neq 0$, where the indices j and i mean H's row and column respectively. The degree of a TG's node is determined by the number of connections the node has; i.e. the degree of v_i and c_j represent the number of non-zero elements on the i-th column and on the j-th row of H respectively. If all v_i nodes have the same degree γ and all c_i nodes the same degree ρ, then the code is said to be regular. Let V be a set of variable nodes and C a set of check nodes. If $v_i \in V$ and $c_j \in C$, the set of neighbors of v_i and c_i is denoted as $g(v_i) \in C$ and $g(c_j) \in V$ respectively; i.e. $g(v_i) = \{c_j \in C \ : \ H_{j,i} \neq 0\}$ and $g(c_j) = \{v_i \in V \ : \ H_{j,i} \neq 0\}$.

This paper focuses on regular LDPC codes that hold the following structural property. That is: $P_V = \{V_q \ : \ V_q \subset V \ \wedge \ V_q \cap V_{w \neq q} = \emptyset \ \wedge \ \cup_q V_q = V\}$ is a partition of V and with the following properties:

- P1: $\forall V_q \in P_V$ it is true that $g(V_q) = C$
- P2: $\forall V_q \in P_V$, let $v_i, v_j \in V_q$ where $i \neq j$; therefor it is true that $g(v_i) \cap g(v_j) = \emptyset$.

A partition P_V fulfilling the properties mentioned above is called a *Variable-node Partition with Full Check-node Connectivity* (VPFCC); and a code in which it is possible to create a partition of this nature is said to be a VPFCC-code. The architecture proposed in this paper to implement

■ The present paper has been supported by SeCyT-UNC and ClariPhy Argentina S.A.

[1] A sparse matrix is a matrix populated primarily with zeros.

[Copia del paper presentado en EAMTA-2008, pg.2/4]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2008

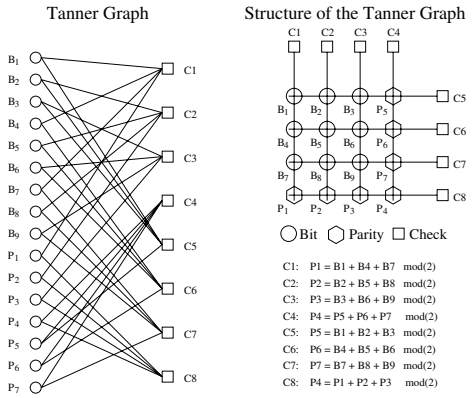


Figure 1. 2D-TPC/SPC(4,3)^2 code: the Tanner graph and its structure.

an LDPC decoder is grounded on these specific properties of a partition P_V.

There are several LDPC codes that satisfy the characteristic properties of VPFCC-codes. A set of these are the Turbo Product Codes based on Single Parity Check (TPC/SPC). [7], [8] y [9] suggest that these codes are suitable for iterative detection schemes for magnetic recording. In fact, the implementation of a TPC/SPC code is described in this paper to illustrate the proposed decoding architecture. The utility the VPFCC property of an LDPC code has on the new architecture will be analyzed in Section IV. Next, the main characteristics of TPC/SPC codes will be briefly described.

A. TPC/SPC Codes

A TPC/SPC code is built through a multi-dimensional array of code words derived from other previously defined code or codes. The codes used for building the multi-dimensional array are called base codes. Typical base codes are: Hamming, BCH, and simple parity check codes. Within TPC codes, those based on bi-dimensional arrays (denoted as 2D-TPC) are of special interest. A 2D-TPC code consisting of two component codes C_1 and C_2 with parameters (n_1, k_1, d_1, G_1) and (n_2, k_2, d_2, G_2) respectively, has parameters $(n_1 n_2, k_1 k_2, d_1 d_2, G_1 \otimes G_2)$; where n, k, d, G are the length, dimension, minimum distance, and generator matrix of the code respectively, and $\otimes$ represents the Kronecker product. Within the 2D-TPC codes, the attention is focused on those based on single parity check codes (SPC) of length N and dimension $N-1$, denoted as 2D-TPC/SPC(N,$N-1$)^2 [7], [8] y [9]. Fig. 1 shows the TG and its structure for a 2D-TPC/SPC(4,3)^2 code.

2D-TPC/SPC codes are all VPFCC-codes. This can be seen through the 2D-TPC/SPC(4,3)^2 example code of Fig. 1, from which the partition $P_V = \{V_1, V_2, V_3, V_4\}$ is built, where:

$$\begin{aligned} V_1 &= \{B_1, B_5, B_9, P_4\} \\ V_2 &= \{B_2, B_6, P_7, P_1\} \\ V_3 &= \{B_3, P_6, B_7, P_2\} \\ V_4 &= \{P_5, B_4, B_8, P_3\}. \end{aligned}$$

It can be verified that this partition holds properties (P1) and (P2), making 2D-TPC/SPC(4,3)^2 a VPFCC-code. This result can be generalized to a 2D-TPC/SPC(N,$N-1$)^2 code by creating a partition $P_V = \{V_i \ : \ i = 1, \ldots, N\}$, where V_i is the i-th modular-diagonal from the square array of $N \times N$ variable nodes. The modular-diagonal of a $N \times N$ square array of elements $x_{i,j}$ is defined as the sets

$$D_k = \{x_{i,j} \ : \ i = 1, \ldots, N \wedge j = (i + k - 1) mod(N)\}. \quad (1)$$

III. The Sum-Product Algorithm

Let b_i be the i-th bit of the code word. The Sum-Product Algorithm (SPA) [10] takes as input the a priori log-likelihood ratio of each bit:

$$AprLLR_i = ln\left(\frac{P(b_i = 1)}{P(b_i = 0)}\right). \quad (2)$$

It then iterates on the computation of the a posteriori log-likelihood ratio:

$$ApoLLR_i = ln\left(\frac{P_C(b_i = 1)}{P_C(b_i = 0)}\right), \quad (3)$$

where

$$P_C(b_i) = \sum_{b_j \ : \ j \neq i} P(b_1, \ldots, b_n \mid AprLLR_1, \ldots, AprLLR_n),$$

being $P(\cdot|\cdot)$ the joint probability function of the code word given the a priori information. One iteration of the SPA algorithm consists of two steps. In the first one (A), messages are calculated and sent from the variable nodes v_i to the check nodes c_j, $M(v_i \to c_j)$. In the second step (B), messages are calculated and sent from the check nodes c_j to the variable nodes v_i, $M(c_j \to v_i)$. Steps (A) and (B) are repeated as many times as required iterations. Finally, in a third step (C), the $ApoLLR$ is computed. Summarizing, the calculations performed on each step are the followings:

A. Messages from variable to check nodes:

$$M(v_i \to c_j) = AprLLR_i + \sum_{c_k \in g(v_i) \setminus c_j} M(c_k \to v_i) \quad (4)$$

B. Messages from check to variable nodes:

$$M(c_j \to v_i) = \phi^{-1}\left\{\sum_{v_k \in g(c_j) \setminus v_i} \phi\Big[M(v_k \to c_j)\Big]\right\} \quad (5)$$

where $\phi(x) = \ln\left[\tanh\left(x/2\right)\right]$, $\phi^{-1}(x) = 2\tanh^{-1}\left(e^x\right)$.

ISBN 978-987-655-003-1 EAMTA 2008 IEEE Catalog number CFP0854E

[Copia del paper presentado en EAMTA-2008, pg.3/4]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2008

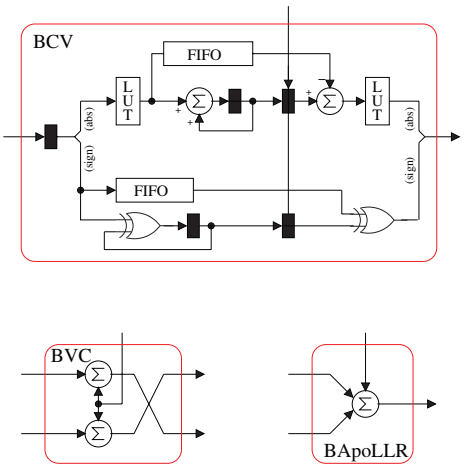


Figure 2. The BCV block computes recursively the messages $M(c_j \rightarrow v_i)$. The BVC block computes in parallel the messages $M(v_i \rightarrow c_j)$. The BApoLLR computes the a posterior information $ApoLLR_i$. All the blocks correspond to one code of the 2D-TPC/SPC family.

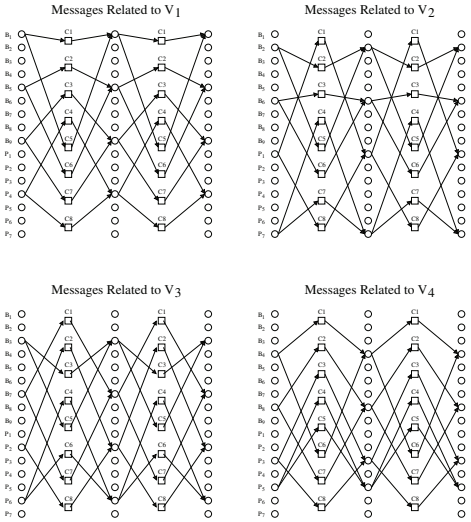


Figure 3. Partition of the computation flow of messages on a SPA algorithm using the VPFCC property of the 2D-TPC/SPC(4,3)^2 code.

C. A posteriori log-likelihood ratio (ApoLLR) computation:

$$ApoLLR_i = AprLLR_i + \sum_{c_k \in g(v_i)} M(c_k \rightarrow v_i) \qquad (6)$$

The computation in steps (A), (B), and (C) is performed in independent blocks named BVC, BCV, and BApoLLR respectively. As the BCV block, which computes messages $M(c_j \rightarrow v_i)$, consumes most of the computational requirements of the SPA algorithm, a modified version is usually implemented. This version is based on the following approximation:

$$\phi^{-1}\Big(\phi(a)+\phi(b)\Big) = 2\tanh^{-1}\left[\tanh\left(\frac{a}{2}\right)\cdot\tanh\left(\frac{b}{2}\right)\right]$$
$$= sign(a)\cdot sign(b)\cdot\min\{|a|,|b|\} + \log\left(\frac{1+e^{-|a+b|}}{1+e^{-|a-b|}}\right)$$
$$\approx sign(a)\cdot sign(b)\cdot\min\{|a|,|b|\}.$$

In this way, the computation of the check to variable messages can be rewritten in the following way:

$$M^*(c_j \rightarrow v_i) = \left[\min_{v_k \in g(c_j)\setminus v_i}\Big\{\Big|M(v_k \rightarrow c_j)\Big|\Big\}\right] \cdot \left[\prod_{v_k \in g(c_j)\setminus v_i} sign(M(v_k \rightarrow c_j))\right] \qquad (7)$$

IV. Proposed Architecture

The proposed implementation architecture is based on the order that messages $M(v_i \rightarrow c_j)$ and $M(c_j \rightarrow v_i)$ are computed and evaluated. Fig. 2 shows the BVC and BCV block architectures used to compute the messages $M(v_i \rightarrow c_j)$ and $M(c_j \rightarrow v_i)$, respectively. The BVC block works in a parallel way while the BCV blocks do it recursively, minimizing thus their complexity. Fig. 4 shows the complete architecture of a SPA decoder with two iterations for a 2D-TPC/SPC(4,3)^2 code. The system parallelism is 4, i.e., on each clock cycle 4 values of $AprLLR$ and $ApoLLR$ input and output respectively. The input and output sequence corresponds with the partition elements $P_V = \{V_i \ : \ i = 1,\ldots,4\}$ described in Section II. Fig. 3 shows how messages of the SPA algorithm are computed when the partition P_V is used. Property (P2) of P_V prevents a conflict between $M(v_i \rightarrow c_j)$ messages, i.e., on each clock cycle there will not be two $M(v_i \rightarrow c_j)$ messages arriving at the same check node. Property (P1) ensures that, on each clock cycle, all check nodes will have input information available. As Fig.3 shows, only $N = 4$ variables nodes are updated on each clock cycle. Due to this property, only 4 BVC blocks need to be implemented instead of the $N^2 = 16$ that a full parallel architecture would need. Multiplexers shown in Fig. 4 interconnect the 4 BVC blocks with the corresponding BCV blocks of each set V_i of the partition P_V.

For the case of the 2D-TPC/SPC(N,$N-1$)^2 family of codes, the proposed architecture presents the following char-

[Copia del paper presentado en EAMTA-2008, pg.4/4]

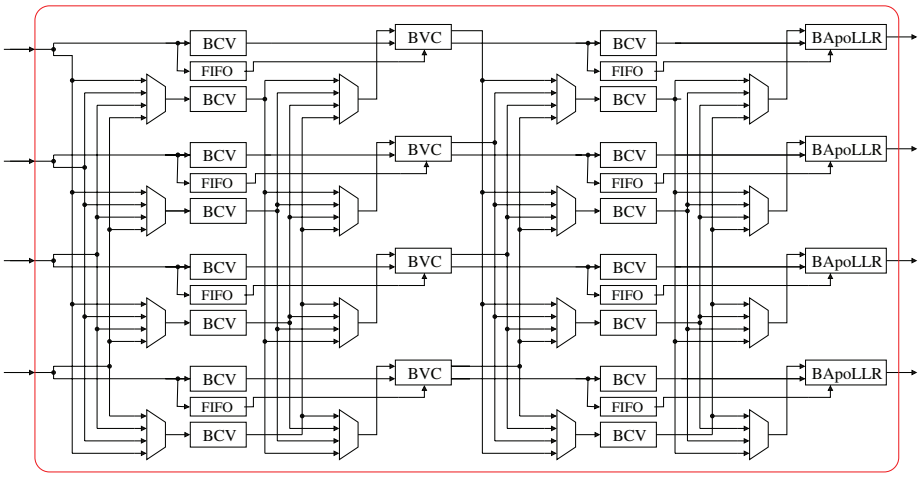


Figure 4. SISO decoder architecture with two SPA iterations for the 2D-TPC/SPC(4,3)^2 code.

acteristics:

- The partition of the computation flow in the SPA algorithm saves the need to store the $M(v_i \rightarrow c_j)$ and $M(c_j \rightarrow v_i)$ messages, which are computed and used on demand.
- The possibility to use a recursive architecture for the BCV block while maintaining a parallel processing of the whole decoder reduces the implementation complexity.
- Only N BVC blocks are implemented, instead of N^2.

The architecture of Fig. 4 can be easily generalized to any other LDPC code fulfilling the VPFCC property. Besides, the proposed technique can be implemented recursively without major modifications. This allows to do several SPA iterations by only *instantiating* the hardware that corresponds to an iteration, e.g. the first half of the architecture shown in Fig. 4.

V. FPGA Implementation

The synthesis of the proposed architecture was performed on an FPGA Virtex 5 (XC5VLX330) from Xilinx. The decoder was designed for the 2D-TPC/SPC(32,31)^2 code with the following parameters: ($n = 1024$, $k = 961$, $d = 4$) and rate $R = 0.9384$. All the signals present in the decoder were digitized using 8 bits precision variables. The resources used were: 16952 registers out of 297369 (8.18%) and 14402 *lookup tables* (LUTs) out of 207360 (6.95%).

VI. Conclusions

This paper proposed a novel parallel architecture for decoding VPFCC-type LDPC codes. This architecture is suitable for working on high speed applications, requiring low memory and interconnection complexity. These features make the architecture attractive for implementing iterative decoders on integrated circuits.

References

[1] D. J. C. Mackay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *IEE Electronics Letters, vol. 33, no. 66, pp 457-8*, March 1997.

[2] E. Yeo, P. Pakzad, B. Nikolic, and V. Anantharam, "VLSI architectures for iterative decoders in magnetic recording channels," *IEEE Trans. Magnetics, vol. 37, no. 2, pp. 748-55*, March 2001.

[3] ——, "High throughput low-density parity-check decoder architectures," *IEEE*, 2001.

[4] C. Howland and A. Blanksby, "A 220mw 1Gb/s 1024-bit rate-1/2 low density parity check code decoder," *IEEE Custom Integrated Circuits Conference*, 2001.

[5] E. Liao, E. Yeo, and B. Nikolic, "Low-density parity-check code constructions for hardware implementation," *IEEE Communications Society*, 2004.

[6] M. Karkooti, P. Radosavljevic, and J. R. Cavallaro, "Configurable, high throughput, irregular LDPC decoder architecture: Tradeoff analysis and implementation," *IEEE ASAP*, 2006.

[7] J. Li, K. Narayanan, and C. Georghiades, "Product accumulate codes: A class of codes with near-capacity performance and low decoding complexity," *IEEE Trans. on Inf. Theory. vol. 50, no. 1, pp. 31-46*, January 2004.

[8] J. Li, K. R. Narayanan, E. Kurtas, and C. N. Georghiades, "On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels," *IEEE Trans. on Communications, vol. 50, no. 5, pp. 723-734*, May 2002.

[9] J. Li, E. Kurtas, K. R. Narayanan, and C. N. Georghiades, "On the performance of turbo product codes over partial response channels," *IEEE Trans. on Communications, vol. 37, no. 4, pp. 1932-1934*, July 2001.

[10] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. on Inf. Theory, vol. 47, no. 2, pp. 498-519*, February 2001.

ISBN 978-987-655-003-1 EAMTA 2008 110 IEEE Catalog number CFP0854E

[Copia del paper presentado en EAMTA-2009, pg.1/5]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2009

FPGA Implementation of High-Speed Parallel Maximum *a Posteriori* (MAP) Decoders

Martín I. del Barco, Gabriel N. Maggio, Damián A. Morero,
Javier Fernández, Facundo Ramos, Hugo S. Carrer, and Mario R. Hueda

Digital Communications Research Laboratory - National University of Cordoba - CONICET
Av. Vélez Sarsfield 1611 - Córdoba (X5016GCA) - Argentina
Clariphy Argentina S.A - 12 de Octubre 1320 - Córdoba (X5000FSV) - Argentina
Emails: mdelbarco, gmaggio, dmorero, hcarrer,mhueda@com.uncor.edu

***Abstract*—This paper presents an efficient parallel architecture for high-speed maximum *a posteriori* (MAP) probability detectors. The parallel systolic scheme proposed here builds upon a sliding window approach, and is capable of providing very high throughput. The implementation of an 8-state MAP decoder on an off-the-shelf field programmable gate array (FPGA) achieves a throughput of 1.6 Gb/s.**

The MAP detector is well-known as the optimal solution to minimize the bit-error-rate (BER). Moreover, when used for equalization on iterative detectors (i.e., turbo equalizers), the MAP algorithm can achieve a performance near Shannon's channel capacity. Thus, the scheme described in this work results highly attractive to efficiently mitigate channel impairments found on high-speed optical fiber systems and other high speed communication applications.

***Keywords:* Iterative decoding, FPGA, MAP, BCJR, turbo equalization.**

I. Introduction

Turbo equalization has recently gained interest as a way of improving the performance of digital communication receivers. Turbo equalizers use the *turbo principle* [1] to perform joint equalization and decoding. It has been demonstrated that an important gain in performance can be obtained by iteratively exchanging extrinsic information between a soft-input/soft-output (SISO) equalizer and an SISO channel decoder. Numerous works show that receivers using turbo decoding can achieve a performance close to Shannon's theoretical limit [2].

Next generation optical fiber communications systems exacerbate the effects of channel impairments, especially intersymbol interference (ISI). Therefore, powerful decoding algorithms such as those based on turbo equalization are being considered for future designs.

This paper focuses on the implementation of a high-speed maximum-a-posteriori (MAP) decoder for a SISO equalizer. Although the MAP algorithm is the optimal solution to minimize bit-error-rate (BER) [3], its practical application on high-speed communication systems (e.g., 1Gb/s and beyond) has been limited as a result of its computational complexity and its implications on the hardware implementation (e.g., high power consumption). Thus, only suboptimal solutions, such as the Soft-Output Viterbi Algorithm (SOVA), have been used in practical systems [4]. However, recent advances in CMOS technology as well as increasingly higher channel dispersion on high speed applications, are pushing towards implementing better but at the same time more complex solutions.

This paper presents the design and analysis of the required resources of a *max-log*-MAP detector, which is a near optimal method derived from the Bahl, Cocke, Jelinek, and Raviv (BCJR) algorithm [3]. Although the hardware implementation of MAP detectors has been widely addressed in the literature (see [5] and references therein), this is the first published systolic parallel architecture implementation able to provide a throughput higher than 1.5Gb/s, as required by many next generation communication systems. The higher performance of this architecture has been achieved by implementing a highly concurrent systolic architecture for a max-log MAP equalizer for a generalized partial response class IV (GPR4) channel.This architecture could operate at a speed of 10 Gbps in a custom designed IC in 65nm or smaller CMOS technology by utilizing a higher parallelization factor.

The rest of this paper is organized as follows. In Section II, the MAP algorithm is presented. In Section III, the proposed systolic architecture is described. Implementation results are shown in Section IV. Finally, conclusions are drawn in Section V.

II. The Max-Log-MAP Algorithm

The goal of the MAP equalizer is to provide a soft output for the information bit, given the received samples. This soft output is called the *log-likelihood ratio* (LLR). Let $\mathbf{c} = \{c_k\}$ denote the transmitted symbol sequence ($c_k \in \{\pm 1\}$). The received samples at the input of the MAP equalizer can be written as

$$r_k = x_k + z_k, \tag{1}$$

■ This paper has been supported in part by SeCyT-UNC and ClariPhy Argentina SA.

ISBN 978-9-8725-1029-9 IEEE Catalog number CFP0954E

[Copia del paper presentado en EAMTA-2009, pg.2/5]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2009

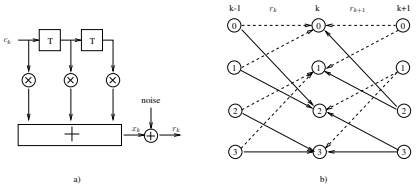


Figure 1. Example of a 4-state trellis ($\nu = 2$).

where $x_k = f(c_k, \ldots, c_{k-\nu})$ is the noise-free signal, which is in general a nonlinear function of $\nu+1$ consecutive symbols, and z_k is the noise sample (we assume the noise samples to be independent random variables). Then, the LLR provided by a MAP equalizer is given by

$$\Lambda(c_k) = \ln \frac{P(c_k = +1 \mid \mathbf{r})}{P(c_k = -1 \mid \mathbf{r})}, \quad (2)$$

where $\mathbf{r} = \{r_k\}$ is the received sequence, and $P(\cdot)$ denotes probability. In order to introduce the notation used in the BCJR algorithm [3], consider the channel and trellis diagram shown in Fig. 1. In this case, the memory of the channel is $\nu = 2$, and the number of states S_k is $2^\nu = 4$ ($S_k \in \{s_i : i = 0, 1, 2, 3\}$). The transitions caused by $c_k = 1$ ($c_k = -1$) are represented by solid (dashed) lines. Thus, the LLR (2) can be computed by using the BCJR algorithm as follows [3]:

$$\Lambda(c_k) = \ln \frac{\sum_{(s_i,s_j)} \overline{\alpha}_{k-1}(s_i)\overline{\gamma}_k^{(+1)}(s_i,s_j)\overline{\beta}_k(s_j)}{\sum_{(s_i,s_j)} \overline{\alpha}_{k-1}(s_i)\overline{\gamma}_k^{(-1)}(s_i,s_j)\overline{\beta}_k(s_j)}. \quad (3)$$

Above $\overline{\gamma}_k^{(a)}(s_i, s_j)$ is the probability of the transition $S_{k-1} = s_i \rightarrow S_k = s_j$ caused by the input symbol $c_k = a$, while $\overline{\alpha}_k(s_i)$ and $\overline{\beta}_k(s_i)$ are the probabilities of being in the state $S_k = s_i$ based on the past and future samples, respectively. As expression (3) would lead to an extremely complex hardware implementation, the near optimal approximation called *max-log MAP* algorithm is adopted:

$$\begin{aligned}\Lambda(c_k) \quad \approx \quad & \max_{s_j}\{\alpha_{k-1}(s_i) + \gamma_k^{(+1)}(s_i,s_j) + \beta_k(s_j)\} - \\ & \max_{s_j}\{\alpha_{k-1}(s_i) + \gamma_k^{(-1)}(s_i,s_j) + \beta_k(s_j)\}, \quad (4)\end{aligned}$$

where $\alpha_k(\cdot) = \ln(\overline{\alpha}_k(\cdot))$, $\gamma_k^{(a)}(\cdot) = \ln(\overline{\gamma}_k^{(a)}(\cdot))$, and $\beta_k(\cdot) = \ln(\overline{\beta}_k(\cdot))$. The computation of $\alpha_k(\cdot)$ and $\beta_k(\cdot)$ is equivalent to the computation of path metrics in the forward and backward recursions, respectively, in the Viterbi algorithm (VA) with a branch metric $\gamma_k^a(\cdot)$. That is:

$$\alpha_k(s_j) = \max_{s_i}\{\alpha_{k-1}(s_i) + \gamma_k^{(a)}(s_i,s_j)\}, \quad (5)$$

$$\beta_k(s_j) = \max_{s_i}\{\beta_{k+1}(s_i) + \gamma_{k+1}^{(a)}(s_j,s_i)\}, \quad (6)$$

where $a = \pm 1$. The operations involved in (5) and (6) are the same as the add-compare-select operations in the VA. In transmissions over additive white Gaussian noise (AWGN) channels:

$$\gamma_k^{(a)}(s_i,s_j) = \ln(P(c_k = a)) - \frac{(r_k - \hat{x}_k(s_i,s_j))^2}{2\sigma^2}, \quad (7)$$

where $\hat{x}_k(s_i, s_j)$ is the noise-free received signal associated to the transition $S_{k-1} = s_i \rightarrow S_k = s_j$ caused by $c_k = a$, while σ is the noise standard deviation.

III. Architecture

This section describes the parallel systolic architecture implementation of the max-log-MAP algorithm for a GPR4 channel using a simple example. A 4-state trellis ($\nu = 2$) is used, with $2^{\nu+1} = 8$ branch metrics for all possible state transitions. The parallelization factor is 4, and data blocks of 8 symbols are used. Each block consists of two consecutive input data blocks of 4 symbols each. The 4 symbols at the center of the block are used for detection, while the first and last 2 symbols are used for synchronization purposes [6]. The good regularity and symmetry of this approach make the implementation significantly simpler, in particular the place and route processes.

A. Description

Fig. 2 shows a data flow diagram for the architecture described above, where data flows from top to bottom. The architecture is fully pipelined in order to achieve the required unfolding and re-timing [7], which eventually allows to regularly divide the forward and backward recursions to reduce the critical path. On every clock cycle, a block of 4 samples and its corresponding *a priori* information, are acquired and allocated at the top of both sides of the input skew buffers. As shown in Fig. 2, the left arm of the input skew buffer has one delay line more than the right arm. This results in the left arm providing the entire 8 symbol block shifted by half a block.

The forward (5) and backward (6) recursions start respectively from both top corners of the input skew buffer. These recursions progress along the data flow chart up to point C in Fig.2, after which the last part of the recursive data processing is done (i.e., the opposite recursions and the LLR calculation). On every clock cycle, the recursions are carried out until they merge at the center of the data flow. Therefore, the input skew buffer delays more the samples of the central part than those of the corners.

The computed metrics are saved once the block synchronization symbols have been processed (Point A and B in Fig.2). It is at this point that the state metrics are assumed to be reliable, from here on both state and branch metrics are stored in the recursion skew buffers (see Fig. 2). After 4+1 clock cycles, forward and backward recursions merge at the center of the data flow chart, where the opposite recursions and LLR calculation (4) are carried out. Since the branch metrics have already been stored in the recursion skew buffers, branch metric calculation units (BMU) are not

ISBN 978-9-8725-1029-9 IEEE Catalog number CFP0954E

[Copia del paper presentado en EAMTA-2009, pg.3/5]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2009

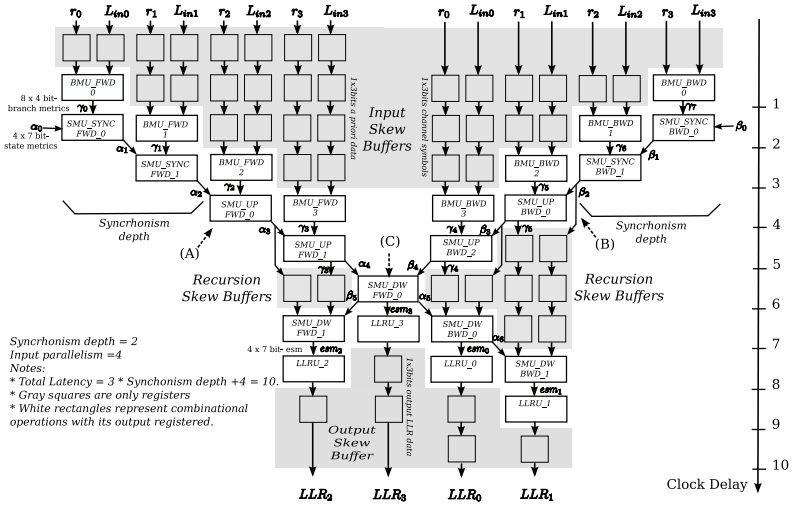


Figure 2. Example of the implemented systolic architecture for an 8-state max-log-MAP. Parallelization factor of 4. Synchronization depth of 2 samples.

required after the merge of forward and backward recursions. To evaluate the LLR (4), each state metric calculation unit (SMU) is followed by a log-likelihood computation unit (LLRU). Once the LLR has been computed, its output is stored in the output skew buffer, and reaches the bottom of the data flow chart after a total count of 10 clock cycles since data was first inserted in the input skew buffer (i.e., the total latency of this architecture).

B. Branch Metric Unit (BMU)

This unit computes the $2^{\nu+1}$ branch metrics for each state transition. The BMU is similar to the one used in the VA implementation reported in [8]. However in the MAP algorithm (7), unlike VA, the evaluation of the branch metrics also have to consider the noise power σ^2 and the *a-priori* information from the previous iteration.

Fig. 3-a shows a block diagram of the branch metrics computation stage (BM) used for evaluating (7). First, a comparator computes the difference $(r_k - \hat{x}_k(s_i, s_j))$, and then a look up table (LUT) is used to obtain the square and division by the factor $2\sigma^2$. The LLR2LOG block computes the term $\ln(P(c_k = a))$ by using the LLR provided by the external turbo channel decoder, L_{in}. Towards this end, the Jacob formula is used:

$$\ln(P(c_k=-1)) = -\ln\left(1+e^{L_{in}}\right) \approx -\max\{0, L_{in}\}$$
$$\ln(P(c_k=+1)) = L_{in} - \ln\left(1+e^{L_{in}}\right) \approx \min\{0, L_{in}\}$$

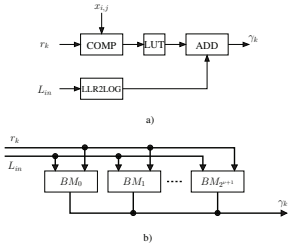


Figure 3. Block diagram of BMU. (a) BM units (b) BMU block

The BMU block shown in Fig. 3-b consists of $2^{\nu+1}$ BM units in a parallel arrangement.

C. State Metric Units (SMU)

These units calculate the next 2^{ν} state metrics, based on the previous (future) state metrics for the forward (backward) recursion. Towards this end, both, *(i)* the outputs of the SMU in the pipelined recursion (calculated at the previous clock), and *(ii)* the branch metrics provided by the corresponding BMU, are processed by add-compare-select (ACS) units, as shown in Fig. 4-a. SMU_SYNC and SMU_UP are used during synchronization and decoding recursions, respectively. They differ in that SMU_UP does not have branch metrics as output. On the other hand, SMU_DW units are used when

ISBN 978-9-8725-1029-9 IEEE Catalog number CFP0954E

[Copia del paper presentado en EAMTA-2009, pg.4/5]

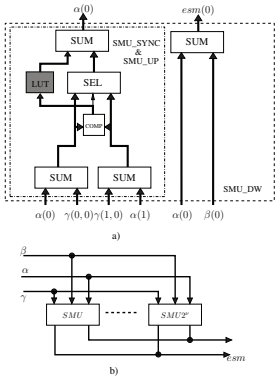


Figure 4. Block diagram of the SMU. a) SM units b) SMU Block.

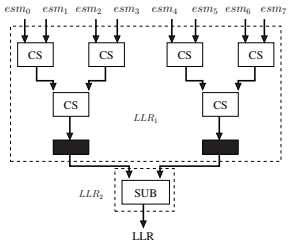


Figure 5. Block diagram of LLRU.

forward and backward recursions merge, at the center of the data flow. Unlike SMU_SYNC and SMU_UP, SMU_DW also provide extended state metrics (ESM), which are used by the log-likelihood ratio unit (LLRU). Finally, the SMU block consists of a parallel arrangement of 2^{ν} SM units shown in Fig.4-a.

D. Log Likelihood Ratio Unit (LLRU)

This unit computes the soft outputs according to (4) based on the ESM provided by SMU_DW (see Fig. 5). The first stage, referred as LLR_1, performs the compare-select tree computations (CS), while the second one, LLR_2, calculates the final subtraction.

E. Unsigned Saturated Metrics

Implementing BMU LUTs carries a considerably high implementation cost. Nevertheless, this cost can be reduced by a factor of 2 if the absolute value of the subtraction is used as input. Also, by using unsigned metrics (they have the same sign), LUTs outputs can be reduced in one bit. In the current implementation, the signs of the BM, BMU LUTs, and

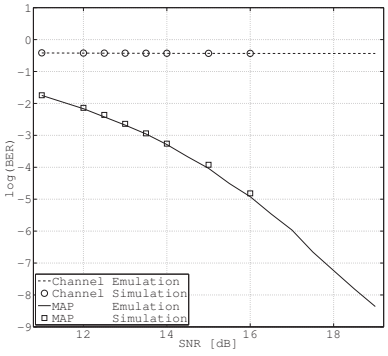


Figure 6. BER vs. SNR emulation and simulation results for a PR [1 3 3 1] channel.

SM were made positive. By doing this, one bit was saved in every operation inside the BMUs, SMUs, and recursion skew buffers. To produce these changes, minor tweaks are required in the algorithm implementation:

- LLR2LOG must provide $-\ln(P(\cdot))$ instead of $\ln(P(\cdot))$,
- $\min(\cdot)$ must be used instead of $\max(\cdot)$, in the SM units.

Saving this bit avoids the use of overflow for metric normalization. Then, saturation logic is implemented in the state metric computation [9], [10].

IV. Implementation

A. Description of emulation platform

A DN9000K10 FPGA emulation board was used, which consists of 8 interconnected Virtex 5 LX 330 FPGAs. The emulation hardware platform is configured using a PC-based GUI interface, connected to the platform through a USB interface. The emulation system contains a random sequence generator (RNG) which generates pseudo-random data sequences in periods of approximately 2^{80} ($\simeq 10^{24}$). These sequences are run through a GPR4 channel, where the convolution between the binary data and GPR the target is performed. Then, an AWGN generator based on the CDF (Cumulative Density Function), generates and adds noise. Finally, samples are fed into a first-in-first-out (FIFO) buffer in the soft channel detector. This FIFO is eventually used in the bit error rate (BER) calculation block.

Figure 6 shows the performance obtained through emulation and simulation of the system. Both methods match in results, and emulation is able to show the system behavior at a BER of 10^{-9}.

B. Resource utilization and maximum throughput comparisons

The present max-log-MAP implementation uses 8 states, a synchronization block of 8 symbols, and a decoding block of

ISBN 978-9-8725-1029-9 IEEE Catalog number CFP0954E

[Copia del paper presentado en EAMTA-2009, pg.5/5]

Proceedings of the Argentine School of Micro-Nanoelectronics, Technology and Applications 2009

Table I
RESOURCE USAGE BY HIERARCHY

Module	SliceReg per unit	LUTs per unit	LRAM per unit	BRAM per unit
CH_EST_LUT_0	448	92	0	0
CH_EST_LUT_1	0	3200	3136	0
LOG_MAP_0	10278	42575	0	98
1 x input_bwd_skw_buf	89	66	0	13
1 x input_fwd_skw_buf	101	70	0	7
16 x SMU_SYNC	112(*)	860(*)	0	0
16 x SMU_UP	112(*)	1050	0	0
1 x rec_bm_fwd_skw_buf	140	35	0	25
1 x rec_bm_bwd_skw_buf	136	30	0	21
1 x rec_sm_fwd_skw_buf	204	35	0	17
1 x rec_sm_bwd_skw_buf	200	30	0	12
2 x SMU_DW_0	0	599(*)	0	0
15 x SMU_DOWN	0	598(*)	0	0
2 x SMU_DOWN_END	0	176(*)	0	0
1 x State metric registers	5416			
16 x LLRU	22(*)	120(*)	0	0
1 x output_fwd_skw_buf	44	25	0	0
1 x output_bwd_skw_buf	44	25	0	3

(*) Average values over the total units.

16 symbols. Also, the fixed point resolutions were chosen so as to achieve a SNR degradation below 0.2 dB at the detector output.

BM calculations require two tables: one to save the estimated output of the channel (CH_EST_LUT_0), and a second one to store the calculated euclidean metric considering the channel noise power (CH_EST_LUT_1). The first table is not needed on fixed channels, though it is still used to change the channel characteristics.

Buffers requiring more than 6 taps were implemented as circular buffers, for which RAM blocks were used. Smaller buffers were implemented as shift registers using LUT registers. This approach reduces resource utilization and compilation time. Every block of the max-log-MAP architecture was implemented on a Xilinx Virtex 5 FPGA (v5lx330ff1760). Logic synthesis, place and route simulation, and timing analysis were performed using ISE Fundation 10.1 design tool. The usage of resources by hierarchy can be seen on Table I. In few words, the max-log-MAP implementation presented on this paper consumes the following amount of resources: 45,807 slice LUTs out of 207,360 (22%), 10726 slice registers out of 207,360 (5%), 3,136 slice register used as RAM out of 54,720 (6%), and 3,546 KB of memory out of 10,368 KB (34%).

The critical path is 10 ns and is located at the BMU. It results in a throughput of 1.6 Gbps if using a parallelism of 16. Table II shows a comparison between SISO detectors for high speed applications, it can be seen that the present implementation doubles the speed of the highest recorded throughput detector for high speed applications.

V. CONCLUSIONS

This paper proposed a systolic parallel architecture for max-log-MAP detectors based in the sliding window approach. The architecture is suitable for high-speed applications as required by next generation communication systems. A throughput of 1.6Gb/s has been achieved by implementing an 8-state max-log-MAP equalizer for a GPR4 channel on a commercial FPGA platform. Finally, when the proposed scheme is translated into a VLSI design, it would provide further improvements in speed.

Table II
THROUGHPUT COMPARISON BETWEEN SISO DETECTORS IMPLEMENTATIONS

Ref.	Tech.	Clock	Throughput 1 iteration	Algorithm
[11]	$180nm$[1]	$145MHz$	$144Mbps$	log-MAP
[12]	$Virtex5$[2]	$310MHz$	$139Mbps$	MAX Scale
[13]	$Virtex2$[2]	$56MHz$	$79.2Mbps$	MAP
[14]	$180nm$[1]	$500MHz$	$500Mbps$	SOVA
[5]	$130nm$[1]	$750MHz$	$750Mbps$	log-MAP
Our	$Virtex5$[2]	$100MHz$	$1.6Gbps$	max-log-MAP

[1]: *VLSI Technology.*
[2]: *FPGA Technology.*

REFERENCES

[1] C. Berrow, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-Codes," *IEEE Trans. Communications, Vol. 44, No. 10, pp. 1261-1271*, Oct 1996.

[2] R. Gallager, *Information theory and reliable communications*. John Wiley and Sons Inc, 1968.

[3] L. Bahl, J. Coocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory, Vol. 20, No. 2, pp. 284-287*, Mar 1974.

[4] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft decision outputs and its applications," *in Proc. IEEE GLOBECOM, IEEE, Vol. 3, pp. 1680-1686*, Nov 1989.

[5] R. Ratnayake, A. Kavcic, and G.-Y. We, "A high-throughput maximum a posteriori probability detector," *IEEE Journal of Solid-State Circuits, Vol. 43, No. 8, pp. 1846-1858*, Aug 2008.

[6] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE Journal in Selected Areas in Communications, Vol. 16, No. 2, pp. 260-264*, Feb 1998.

[7] K. Parhi, *VLSI digital signal processing systems. Design and implementation*. John Wiley and Sons Inc, 1999.

[8] P. Black and T. Meng, "A 1-Gb/s, four-state, sliding block Viterbi decoder," *IEEE Journal of Solid State Circuits, Vol. 32, No. 6, pp. 797-805*, June 1997.

[9] A. Hekstra, "An alternative to metric rescaling in Viterbi decoders," *IEEE Trans. Communication, Vol. 37, No. 11, pp. 1220-1222*, Nov 1989.

[10] B. Shim and H. Myung, "A novel metric representation for low-complexity log-map-decoder," *IEEE International Symposium on Circuits and Systems, Vol. 6, pp. 5830-5833*, May 2005.

[11] M. Bickerstaff, L. Davis, C. Thomas, D. Garret, and C. Nicol, "A 24 Mb/s radix-4 Log-MAP turbo decoder for 3GPP-HSDPA mobile wireless," in *Proceedings of IEEE International Solid-State Circuits Conference*, San Francisco, Calif, USA, Feb 2003, pp. 150–151.

[12] Xilinx, "3GPP turbo decoder v3.1," *DS318*, May 2007.

[13] D. Choi, M. Kim, J. Jeon, and et al., "An FPGA implementation of high speed flexible 27-Mbps 8-state turbo decoder," *ETRI Jornal, Vol. 29, No. 3, pp. 363-370*, 2007.

[14] E. Yeo, A. Ausgburger, W. R. Davis, and B. Nikolic, "A 500-Mb/s soft-output Viterbi decoder," in *IEEE Int. Solid-State Circuits Conf.(ISSCC) Dig. Tech. Papers*, Jul 2003, pp. 1234–1241.

ISBN 978-9-8725-1029-9 102 IEEE Catalog number CFP0954E

[Copia del paper presentado en ICC-2010, pg.1/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

Performance of Euclidean-Metric MLSD Receiver in the Presence of Channel Mismatch Caused by Nongaussian Noise

Damian A. Morero and Mario R. Hueda

Digital Communications Research Laboratory - National University of Cordoba - CONICET
Av. Vélez Sarsfield 1611 - Córdoba (X5016GCA) - Argentina
Emails: dmorero, mhueda@com.uncor.edu

***Abstract*—In this paper we present a theory of the bit error rate (BER) of Euclidean metric-based maximum likelihood sequence detectors (EM-MLSD) in the presence of channel mismatch caused by nongaussian noise. Although the theory is general, here we focus on the effects of quantization noise (QN) added by the front-end analog-to-digital converter (ADC) typically used in DSP based implementations of the receiver. Numerical results show a close agreement between the predictions of the theoretical analysis and computer simulations. As a practical application of the proposed theory, we investigate the performance of EM-MLSD in 10Gb/s Ethernet receivers for multimode optical fibers [1]. Since the BER required in this application is below 10^{-12}, which precludes the use of computer simulations to estimate BER, a theoretical study of the MLSD performance including the combined effects of the channel dispersion and QN, becomes necessary. We present numerical results for the three stressors specified by the 10GBASE-LRM standard. Our study shows that the impact of the QN added by the ADC on the performance depends strongly on the channel dispersion (i.e., the stressor).**

I. Introduction

Maximum likelihood sequence detection (MLSD) is widely used in digital communication receivers to combat the effects of the channel dispersion [2]. In particular, recent studies (see [3] and references therein) have demonstrated the superiority of MLSD for multimode fiber (MMF) links used in local area network (LAN) applications as specified by the 10GBASE-LRM standard [1]. In MMF links, thermal noise tends to be dominant. Thus, Gaussian noise is a reasonable approximation for these channels [4]. This fact motivated the use of Euclidean metric-based MLSD (EM-MLSD) in 10Gb/s Ethernet receivers over MMF links.

The bit error rate (BER) at the MLSD output required by several applications such as 10GBASE-LRM is $< 10^{-12}$ [1]. At this error rate, computer simulation alone is not a viable tool for performance evaluation; thus, some form of theoretical analysis is required. The error probability of EM-MLSD over dispersive channels has been widely investigated in the literature [2], [5], [6], [7], [8]. For example, [7] investigates the performance of MLSD for generalized nongaussian channels, while a theory of the error rate of MLSD in Gaussian channels is proposed in [5].

In real situations, there are some nongaussian noise components added by different stages of the communication system. One of the best known in DSP based implementations of the receiver is the quantization noise (QN) added by the analog-to-digital converter (ADC), which is well modeled as uniformly distributed noise [9]. Although the effect of QN can be neglected in numerous applications, it becomes relevant in some high-speed commercial systems such as 10GBASE-LRM, owing to practical limitations for developing low power, high-speed, and high resolution ADC. As we shall show later, assuming the Gaussian approximation for the QN leads to inaccurate and conservative BER estimates. On the other hand, although performance of MLSD designed for nongaussian noise has been investigated in the past (e.g., [7], [10]), analytical studies of MLSD with Euclidean metrics over nongaussian channels have not been reported so far. Therefore, theoretical tools should be developed in order to evaluate the performance of EM-MLSD in the presence of nongaussian noise at very low BERs.

In this paper we present a theory of the error probability of EM-MLSD over nongaussian dispersive channels. Although the theory is general, here we focus on the combined effects of thermal (Gaussian) noise and quantization (nongaussian) noise. Our analysis builds upon that reported in [5] for Gaussian channels. Computer simulations show a close agreement with the predictions of our theory. Furthermore, in this work we investigate the performance of EM-MLSD in 10Gb/s Ethernet receivers for MMF [1]. Numerical results for the three stressors (channels) specified by the 10GBASE-LRM standard show that the impact on performance of the QN added by the ADC depends strongly on the stressor.

The paper is organized as follows. Section II introduces the theory of the bit error rate of EM-MLSD in the presence of channel mismatch caused by nongaussian noise. In Section

■ This paper has been supported in part by SeCyT-UNC, PICT-2007-534, and MinCyT-CBA PID-2008.

978-1-4244-6404-3/10/$26.00 ©2010 IEEE

[Copia del paper presentado en ICC-2010, pg.2/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

III, we evaluate the peformance of EM-MLSD with QN. Numerical results are shown in Section IV. Finally, conclusions are drawn in Section V.

II. Analysis of EM-MLSD with Nongaussian Noise

The received signal at the input of the EM-MLSD can be expressed as

$$y_n = f_0(\mathbf{a}_{n-\rho+1}^{n}) + f_1(\mathbf{a}_{n-\lambda+1}^{n+\gamma}) + r_n + q_n, \quad (1)$$

where $\mathbf{a}_i^j$ represents the vector of transmitted bits $(a_i, a_{i+1}, \ldots, a_j)$. Term $f_0(\cdot)$ is the main nonlinear distortion of the channel. We assume that $f_0(\cdot)$ is known by the receiver and used for bit detection. $f_1(\cdot)$ represents a secondary nonlinear distortion of the channel which is not taken into account in the detection process. Terms r_n and q_n are the Gaussian noise and nongaussian noise, respectively. We assume that (*i*) both noise components are colored random processes, and (*ii*) y_n is an identically distributed random variable (RV).

A. Error Rate of EM-MLSD

In general, practical EM-MLSD receivers are designed assuming that the input samples are independent and identically distributed Gaussian variables. Thus, our objective is to investigate the impact on the performance of EM-MLSD of the channel mismatch caused by the presence of (correlated) nongaussian noise.

Without loss of generality, in this work we assume that a symbol error corresponds to exactly one bit error. Then, the probability of bit error of the Viterbi decoder (VD) [2] is upper bounded by

$$P_b \leq \sum_{\Psi \neq \hat{\Psi}} W_H(\Psi, \hat{\Psi}) \Pr\{\hat{\Psi}|\Psi\} \Pr\{\Psi\}, \quad (2)$$

where $\Psi = \{a_n\}$ represents the transmitted sequence, $\hat{\Psi} = \{\hat{a}_n\}$ is an erroneous sequence, $\Pr\{\hat{\Psi}|\Psi\}$ is the probability of the error event $\Psi \to \hat{\Psi}$ (i.e., the error event that occurs when the Viterbi decoder chooses sequence $\hat{\Psi}$ instead of Ψ), and $W_H(\Psi, \hat{\Psi})$ is the Hamming weight of $\Psi \wedge \hat{\Psi}$ or, in other words, the number of bit errors in the error event ($\wedge$ is the exclusive OR operator). $\Pr\{\Psi\}$ is the probability that the transmitter sent sequence Ψ. It is well known that, in the high signal-to-noise ratio (SNR) regime, P_b is dominated by

$$P_b \approx \max_{\Psi \neq \hat{\Psi}} \left(W_H(\Psi, \hat{\Psi}) \Pr\{\hat{\Psi}|\Psi\} \Pr\{\Psi\} \right). \quad (3)$$

B. Probability of an Error Event

Considering that the erroneous sequence $\hat{\Psi}$ differs from the transmitted sequence Ψ only for $n_0 \leq n \leq n_1$, EM-MLSD will choose the erroneous path if

$$\sum_{k=n_0}^{n_1+\rho-1} \left(y_k - f_0\left(\mathbf{a}_{k-\rho+1}^{k}\right)\right)^2 > \sum_{k=n_0}^{n_1+\rho-1} \left(y_k - f_0\left(\hat{\mathbf{a}}_{k-\rho+1}^{k}\right)\right)^2.$$

Replacing (1) in the previous equation and computing the squares, we get

$$\sum_{k=n_0}^{n_1+\rho-1} \Delta_k \left(r_k + q_k\right) > \varrho, \quad (4)$$

where

$$\Delta_n = f_0(\hat{\mathbf{a}}_{n-\rho+1}^{n}) - f_0(\mathbf{a}_{n-\rho+1}^{n}), \quad (5)$$

$$\varrho = \frac{1}{2} \sum_{k=n_0}^{n_1+\rho-1} \left((\Delta_k)^2 - 2\Delta_k i_k\right), \quad (6)$$

$$i_n = f_1(\mathbf{a}_{n-\lambda+1}^{n+\gamma}). \quad (7)$$

Expression (4) can be rewritten as

$$\boldsymbol{\Delta}^T\mathbf{r} + \boldsymbol{\Delta}^T\mathbf{q} > \frac{1}{2}\left(\|\boldsymbol{\Delta}\|^2 - 2\boldsymbol{\Gamma}^T\boldsymbol{\Delta}\right), \quad (8)$$

where

$$\begin{aligned} \mathbf{r} &= (r_{n_0}, r_{n_0+1}, \ldots, r_{n_1+\rho-1})^T, \\ \mathbf{q} &= (q_{n_0}, q_{n_0+1}, \ldots, q_{n_1+\rho-1})^T, \\ \boldsymbol{\Delta} &= (\Delta_{n_0}, \Delta_{n_0+1}, \ldots, \Delta_{n_1+\rho-1})^T, \\ \boldsymbol{\Gamma} &= (i_{n_0}, i_{n_0+1}, \ldots, i_{n_1+\rho-1})^T. \end{aligned} \quad (9)$$

Let $\mathbf{R}$ be the normalized autocovariance matrix of the gaussian noise $\mathbf{r}$, i.e. $\mathbf{R}_{i,j} = \frac{1}{\sigma^2}\mathbb{E}\{\mathbf{r}_i^T\mathbf{r}_j\}$ ($\mathbb{E}\{\cdot\}$ denotes expectation). Since $\mathbf{R}$ is symmetric, its inverse $\mathbf{R}^{-1}$ is also a symmetric and diagonizable matrix. Therefore, $\mathbf{R}^{-1}$ can be expressed as $\mathbf{R}^{-1} = \mathbf{U}^T\mathbf{D}^2\mathbf{U}$, where $\mathbf{U}$ is an orthogonal matrix (i.e., $\mathbf{U}^{-1} = \mathbf{U}^T$), whose rows are the eigenvectors of $\mathbf{R}^{-1}$; $\mathbf{D}$ is a diagonal matrix whose diagonal elements are the square root of the eigenvalues of $\mathbf{R}^{-1}$. Let $\mathbf{x}$ be the noise vector computed by whitening $\mathbf{r}$ through the linear transformation $\mathbf{x} = \frac{1}{\sigma}\mathbf{D}\mathbf{U}\mathbf{r}$ (i.e. $\mathbf{r} = \sigma\mathbf{U}^T\mathbf{D}^{-1}\mathbf{x}$). Then, from (8) we can obtain

$$\sigma\boldsymbol{\Delta}^T\mathbf{U}^T\mathbf{D}^{-1}\mathbf{x} + \boldsymbol{\Delta}^T\mathbf{q} > \frac{1}{2}\left(\|\boldsymbol{\Delta}\|^2 - 2\boldsymbol{\Gamma}^T\boldsymbol{\Delta}\right). \quad (10)$$

Condition (10) is equivalent to

$$u + v > d, \quad (11)$$

where u and v are, respectively, a normalized zero-mean Gaussian RV and a nongaussian RV defined by

$$u = \left[\frac{\boldsymbol{\Delta}^T\mathbf{U}^T\mathbf{D}^{-1}}{(\boldsymbol{\Delta}^T\mathbf{R}\boldsymbol{\Delta})^{\frac{1}{2}}}\right]\mathbf{x}, \quad (12)$$

$$v = \left[\frac{\boldsymbol{\Delta}^T}{\sigma\,(\boldsymbol{\Delta}^T\mathbf{R}\boldsymbol{\Delta})^{\frac{1}{2}}}\right]\mathbf{q}, \quad (13)$$

while d is a constant given by

$$d = \frac{1}{2}\frac{\|\boldsymbol{\Delta}\|^2 - 2\boldsymbol{\Gamma}^T\boldsymbol{\Delta}}{\sigma\,(\boldsymbol{\Delta}^T\mathbf{R}\boldsymbol{\Delta})^{\frac{1}{2}}}. \quad (14)$$

[Copia del paper presentado en ICC-2010, pg.3/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

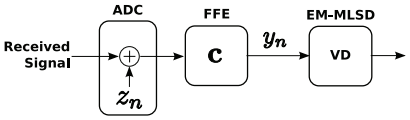


Figure 1. Block diagram of the receiver.

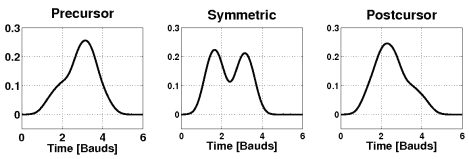


Figure 2. The LRM stressors.

Since u and v are independent RVs, the error event probability $\Pr\{\hat{\Psi}|\Psi\}$ can be computed as

$$\Pr\{\hat{\Psi}|\Psi\} = \Pr\ (u+v>d) = \int_d^\infty f(t)dt, \tag{15}$$

where

$$f(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f_v(\tau) e^{-\frac{1}{2}(t-\tau)^2} d\tau, \tag{16}$$

with $f_v(.)$ being the probability density funcion (pdf) of the nongaussian component v.

III. Performance of EM-MLSD with QN

In this section, the proposed theory is used to evaluate the performance of EM-MLSD in the presence of the nongaussian quantization noise added by the ADC. Although the effect of QN can be neglected in numerous applications, it may be important in commercial high-speed transmission systems over dispersive channels such as 10Gb/s Ethernet applications [1]. In these systems,

- the number of bit of the ADC is low/medium owing to power consumption restrictions (e.g.,$< 8b$), and
- the BER at the EM-MLSD output is very low (e.g., BER$< 10^{-12}$).

Note that the latter precludes the use of computer simulations for performance evaluation, therefore analytical approaches such the one proposed in this work should be adopted.

Figure 1 shows a block diagram of the receiver under consideration. The receiver signal samples are quantized by an ADC with effective number of bits (ENOB) [11] equals to N_b[1]. We assume that the QN samples, z_n, are independent and uniformly distributed (i.u.d) RVs with pdf given by [9]

$$f_z(z_n) = \begin{cases} \frac{1}{\delta} & \text{if } z_n \in \left[-\frac{\delta}{2}, \frac{\delta}{2}\right] \\ 0 & \text{otherwise} \end{cases}, \tag{17}$$

where $\delta = V_{pp} 2^{-N_b}$ with V_{pp} being the input range of the ADC. After ADC, the samples are processed first by a feedforward equalizer (FFE) with L taps, and then by a VD with $S = 2^{\rho-1}$ states (see [4] for a more detailed description of the receiver architecture).

[1]In general, N_b is not an integer number.

A. Bit Error Probability

The nongaussian noise component at the EM-MLSD input is given by

$$q_n = \sum_{i=0}^{L-1} z_{n-i} \cdot c_i, \tag{18}$$

where c_i is the i-th tap of the FFE. The vector $\mathbf{q}$ defined in (9) can be expressed as

$$\mathbf{q} = \mathbf{C}\mathbf{z}, \tag{19}$$

where $\mathbf{C}$ is the $(n_1-n_0+\rho)\times(n_1-n_0+\rho+L-1)$ convolution matrix defined by

$$\mathbf{C} = \begin{bmatrix} c_0 & c_1 & \cdots & c_{L-1} & \cdots & 0 & 0 \\ 0 & c_0 & \cdots & c_{L-2} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \cdots & c_{L-2} & c_{L-1} \end{bmatrix},$$

and $\mathbf{z}$ is the $(n_1 - n_0 + \rho + L - 1)$-dimensional vector

$$\mathbf{z} = (z_{n_0-L+1}, z_{n_0-L}, \ldots, z_{n_1+\rho-1})^T.$$

Replacing (19) in (13) we get

$$v = \mathbf{w}^T \mathbf{z}, \tag{20}$$

where

$$\mathbf{w} = \left[\frac{\mathbf{\Delta}^T\mathbf{C}}{\sigma\left(\mathbf{\Delta}^T\mathbf{R}\mathbf{\Delta}\right)^{\frac{1}{2}}}\right]^T. \tag{21}$$

Note that v is a linear combination of i.u.d. RVs, and its pdf $f_v(.)$ can be determined from the $(n_1 - n_0 + \rho + L - 1)$-dimensional vector $\mathbf{w}$ and the pdf of z_n. A generic closed-form expression for the pdf of v (20) is developed in Appendix.

Finally, the bit error probability can be estimated from (2) with

$$\Pr\{\hat{\Psi}|\Psi\} = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} [1 - F_v(\tau)] e^{-\frac{1}{2}(d-\tau)^2} d\tau, \tag{22}$$

where the $F_v(\cdot)$ is the cumulative distribution function (cdf) of v given by (27).

[Copia del paper presentado en ICC-2010, pg.4/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

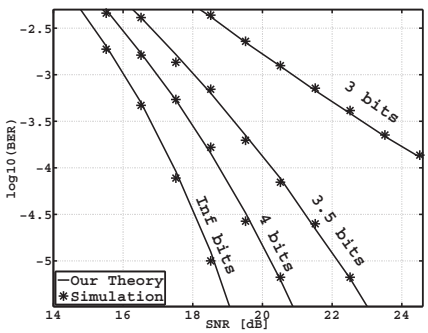


Figure 3. Simulation vs. theory for several values of ENOB (8-state VD and 31-tap FFE with precursor stressor).

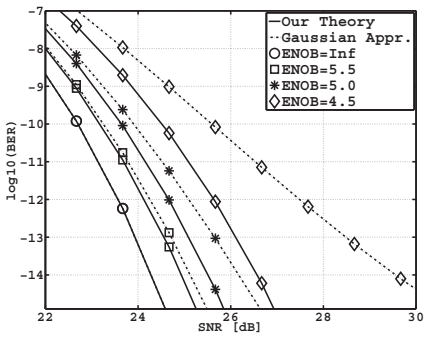


Figure 4. BER vs SNR for several values of ENOB (8-state VD and 31-tap FFE with postcursor stressor).

IV. Numerical Results

In this section we analyze the accuracy of the proposed theory and evaluate the peformance of EM-MLSD receivers in 10Gb/s Ethernet applications. We consider the three stressors specified by the 10GBASE-LRM standard (see Fig. 2) [1]. The sampling phase with the maximum energy pulse (stressor) is used. The optical modulation amplitude (OMA) of the received signal is adjusted to 1 (i.e., ± 0.5) [1]. The thermal Gaussian noise samples after the ADC are assumed independent with variance σ_0^2. The SNR is defined as $SNR = \frac{0.25}{\sigma_0^2}$. The ADC input range is $V_{pp} = 2$ (± 1). Theoretical values for BER are derived from (2) and (22)[2].

Figure 3 shows BER versus SNR for several values of ENOB obtained from computer simulation and our theory. The precursor stressor is used. In all cases, the excellent accuracy of the values derived from the proposed theory is verified. Figure 4 shows theoretical results derived from (2) and (22) at very low BER regimes for the postcursor stressor. Values derived from the Gaussian approximation (GA) for the QN are also presented. From this figure, we verify that BER derived from the GA for QN is highly inaccurate and conservative. This problem is exacerbed as the number of bits of the ADC is reduced.

In Fig. 5 we show the SNR penalty at BER$= 10^{-12}$ for the three LRM stressors, VD with $S = 2$ and 8 states, and several values of N_b. The SNR reference is the one required by a 31-tap FFE and 16-state EM-MLSD receiver. We observe that performance degrades significantly for the symmetric stressor when $S = 2$. Note that this degradation is caused by QN and the high residual intersymbol interference (see (7)-(14)). On the other hand, for $S = 8$ we verify higher degradation for the postcursor stressor. This is caused by the FFE taps that tend to *gaussianize* the QN, as it can be inferred from Fig. 6.

[2]The effect of error events with $W_H(\Psi, \hat{\Psi}) > 6$ is negligible for the channels considered in this paper.

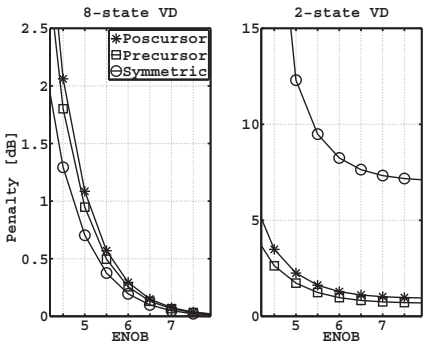


Figure 5. SNR penalty at BER$= 10^{-12}$ for LRM stressors.

V. Conclusions

A theory of the bit error rate of Euclidean metric-based MLSD in the presence of channel mismatch caused by nongaussian noise has been proposed in this paper. Although our theory is general, we focused on the effects of QN added by the ADC typically used in DSP based implementations of the receiver. Numerical results have shown a close agreement between the predictions of the theoretical analysis and computer simulations. Moreover, we have found that assuming the Gaussian approximation for the QN leads to highly inaccurate and conservative BER estimates. As an example of application of the proposed theory, we have evaluated the performance of FFE/EM-MLSD receivers in 10Gb/s Ethernet applications. Our results have shown that the performance degradation of FFE/EM-MLSD receivers caused by QN depends strongly on the dispersion of the channel.

[Copia del paper presentado en ICC-2010, pg.5/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

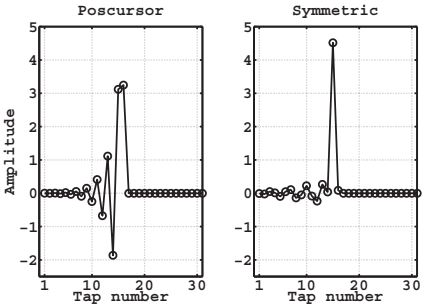


Figure 6. FFE taps at BER= 10^{-12} for an 8-state EM-MLSD with ENOB $N_b = 4.5b$.

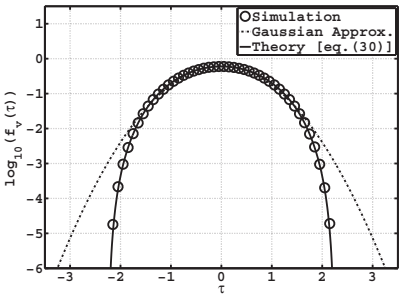


Figure 7. Numerical example for the pdf of the filtered QN noise.

APPENDIX

A. PDF and CDF of Filtered QN

In this Appendix we derive closed-form expressions for the pdf and cdf of the RV defined by

$$v = \mathbf{w}^T \mathbf{z}, \tag{23}$$

where $\mathbf{z} = (z_0, z_1, \ldots, z_{m-1})^T$ is a vector whose components are i.u.d RVs, while $\mathbf{w} = (w_0, w_1, \ldots, w_{m-1})^T$ is a given vector with constant components defined in (20). Then, the cdf $F_v(\tau)$ of v (23) can be calculated as

$$F_v(\tau) = \Pr\left(\mathbf{w}^T\mathbf{z} < \tau\right) = \int_{\mathbf{z}\in\mathcal{Z}(\tau)} \prod_{i=0}^{m-1} f_z(z_i) dz_i, \tag{24}$$

where $m = n_1 - n_0 + \rho + L - 1$ and $\mathcal{Z}(\tau) = \{\mathbf{z} \in \mathbb{R}^m : \sum_{i=0}^{m-1} w_i z_i \le \tau\}$. Expression (24) can be rewritten as

$$F_v(\tau) = 1 - \int_{\mathbf{z}\in\mathbb{R}^m} \prod_{i=0}^{m-1} f_z(z_i) \mu\left(-\tau + \sum_{i=0}^{m-1} w_i z_i\right) dz_i, \tag{25}$$

where $\mu(\cdot)$ is the unit step function. Replacing $f_z(\cdot)$ given by (17) in (25), we get

$$F_v(\tau) = 1 - \frac{1}{\delta^m} \int_{-\frac{\delta}{2}}^{\frac{\delta}{2}} dz_0 \cdots \int_{-\frac{\delta}{2}}^{\frac{\delta}{2}} dz_{m-1} \cdot \mu\left(-\tau + \sum_{i=0}^{m-1} w_i z_i\right). \tag{26}$$

Solving the integrals in (26), we obtain

$$F_v(\tau) = 1 - \frac{1}{\delta^m \prod |w_i|} \sum_{j=0}^{2^m-1} \varsigma_j \cdot \mu^{(m)}\left(\frac{\delta}{2}\mathbf{w}^T\mathbf{p}_j - \tau\right), \tag{27}$$

where $\mathbf{p}_j = (\pi_{j,0}, \pi_{j,1}, \ldots, \pi_{j,m-1})^T$ is a vector with components

$$\pi_{j,i} = -1 + 2 \cdot mod(\lfloor j/2^i \rfloor, 2), \tag{28}$$

$\varsigma_j \in \{+1, -1\}$ is the product of the elements of $\mathbf{p}_j$, and

$$\mu^{(m)}(x) = \int \mu^{(m-1)}(x) dx = \frac{x^m}{m!}\mu(x). \tag{29}$$

Finally, from (27) we can obtain the pdf of v as follows

$$f_v(\tau) = \frac{1}{\delta^m \prod |w_i|} \sum_{j=0}^{2^m-1} \varsigma_j \cdot \mu^{(m-1)}\left(\frac{\delta}{2}\mathbf{w}^T\mathbf{p}_j - \tau\right). \tag{30}$$

B. Numerical Example

Figure 7 compares numerical estimation of $f_v(\cdot)$ with analytical results derived from (30) for $\delta = \frac{1}{2}$ and $\mathbf{w} = [1, -2, 3, -2, 1]^T$. Note the close agreement between the theoretical values and the numerical estimates.

REFERENCES

[1] "IEEE Standard 802.3AQ-2006, Physical Layer and Management Parameters for 10Gb/s Operation, Type 10GBASE-LRM," Sep. 2006. [Online]. Available: http://standards.ieee.org/getieee802/802.3.html

[2] J. R. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital Communication*, 3rd ed. KAP, 2004.

[3] W. Rosenkranz and C. Xia, "Advanced electronic equalization for high-speed data transmission over multi-mode as well as single-mode optical fiber," *Invited Talk, ECOC 2005, Available: http://www.tf.uni-kiel.de/etit/NT/joint/publica.html*, Sept. 2005.

[4] D. E. Crivelli, H. S. Carrer, M. R. Hueda, and O. E. Agazzi, "A MIMO-MLSE receiver for electronic dispersion compensation of multimode optical fibers," *Globecom'06*, Nov. 2006, paper SPC03-3.

[5] O. Agazzi and N. Seshadri, "On the use of tentative decisions to cancel intersymbol interference and nonlinear distortion (with application to magnetic recording channels)," *IEEE Trans. Inf. Theory*, vol. 43, no. 2, pp. 394–408, Mar. 1997.

[6] L. C. Barbosa, "Maximum likelihood sequence estimators: A geometric view," *IEEE Trans. Inf. Theory*, vol. 35, no. 2, pp. 419–427, March 1989.

[7] O. E. Agazzi, M. R. Hueda, H. S. Carrer, and D. E. Crivelli, "Maximum likelihood sequence estimation in dispersive optical channels," *J. Lightw. Technol.*, vol. 23, no. 2, pp. 749–763, Feb. 2005.

[8] M. Franceschini, G. Ferrari, R. Raheli, F. Meli, and A. Castoldi, "State-complexity reduction in mlsd receivers for optical communications with direct photodetection," *J. Lightw. Technol.*, vol. 26, no. 21, pp. 3497–3508, November 2008.

[9] B. Widrow, I. Koll.r, and M. Liu, "Statistical theory of quantization," *IEEE Trans. Instrum. Meas.*, vol. 45, no. 2, pp. 353–361, Apr. 1996.

[10] M. R. Hueda, D. E. Crivelli, H. S. Carrer, and O. E. Agazzi, "Parametric estimation of IM/DD optical channels using new closed-form approximations of the signal PDF," *J. Lightw. Technol.*, vol. 25, no. 3, pp. 957–975, Mar. 2007.

[11] B. Razavi, *Principles of Data Conversion System Design*. IEEE Press, 1995.

[Copia del paper publicado en ArXiv-2011, pg.1/4]

High-Rate Short-Block LDPC Codes for Iterative Decoding with Applications to High-Density Magnetic Recording Channels

Damián A. Morero, Graciela Corral-Briones, Carmen Rodríguez, and Mario R. Hueda

Digital Communications Research Laboratory - National University of Cordoba - CONICET
Av. Vélez Sarsfield 1611 - Córdoba (X5016GCA) - Argentina
Emails: dmorero, gcorral, cer, mhueda@com.uncor.edu

arXiv:1104.1457v1 [cs.IT] 7 Apr 2011

***Abstract*—This paper investigates the *Triangle Single Parity Check* (T/SPC) code, a novel class of high-rate low-complexity LDPC codes. T/SPC is a regular, soft decodable, linear-time encodable/decodable code. Compared to previous high-rate and low-complexity LDPC codes, such as the well-known *Turbo Product Code / Single Parity Check* (TPC/SPC), T/SPC provides higher code rates, shorter code words, and lower complexity. This makes T/SPC very attractive for practical implementation on integrated circuits.**

In addition, we analyze the performance of iterative decoders based on a soft-input soft-output (SISO) equalizer using T/SPC over high-density perpendicular magnetic recording channels. Computer simulations show that the proposed scheme is able to achieve a gain of up to 0.3 dB over TPC/SPC codes with a significant reduction of implementation complexity.

I. Introduction

Iterative decoders based on soft-input soft-output equalizers with powerful error correction codes (ECC) -such as low density parity check (LDPC)- are considered in the literature to be suitable for coping with intersymbol interference and noise in high-speed transmission systems [1], [2], [3]. The potential of this architecture is high, but it requires additional improvements in order to be applicable to the next generation of magnetic recording systems. There are mainly two reasons for this: the high complexity of implementing high-speed iterative receivers in integrated circuits (e.g., 4 Gb/s or higher), and the potential *error floor* problem of fully iterative decoding solutions [4], [5], [6]. The latter problem is exacerbated by the difficulty of evaluating performance at very low bit error rates (BERs).

One interesting alternative that offers a good tradeoff between complexity and performance is the combination of *(i)* an iterative scheme based on a SISO equalizer with an *inner* high-rate low-complexity LDPC code and *(ii)* a powerful *outer* code (such as RS, BCH, or Goppa), as shown in Fig. 1 [7]. In this scheme, the inner LDPC code should achieve very high rates with low complexity as well as provide good error statistics to the outer ECC. It has been shown that the *Turbo Product Code / Single Parity Check* (TPC/SPC) is a suitable candidate to be used as inner code [8], [9]. The main features of the TPC/SPC codes are: minimum distance $d_H = 4$, no length-4 cycles, linear-time encodable and decodable (i.e., low complexity implementation), and high code rate for relatively short codewords [8], [9], [10]. In magnetic recording systems, where the sector size is fixed, the use of LDPC with shorter code words provides benefits in an iterative architecture. This is because the combination of interleaving with *various* code words per sector gives rise to interleaving gain. Therefore, the design of high rate short block LDPC codes holds great interest for iterative receivers in high-density magnetic recording systems.

In this paper we investigate the *Triangle Single Parity Check* (T/SPC) code, a novel LDPC code that can be derived from combinatorial design criteria [9], [11], [12], [13]. T/SPC is suitable as inner code in the iterative decoding scheme shown in Fig. 1. Compared to TPC/SPC, the proposed T/SPC exhibits:

- half code length for a given code rate;
- half parity check nodes for a given code rate;
- higher code rates for a given sector length;
- lower minimum distance ($d_H = 3$).

Note that T/SPC is able to provide a significant reduction of complexity but this comes at the expense of a lower minimum distance when compared to TPC/SPC. However, as it will be shown later, combining a lower code length together with an interleaver allows T/SPC to achieve a significant interleaving gain. Furthermore, numerical results show that T/SPC is able to provide a 0.3 dB gain over TPC/SPC in magnetic recording systems.

The rest of the paper is organized as follows. The system model is presented in Section II. T/SPC code is analyzed in Section III. Section IV evaluates the performance of T/SPC and TPC/SPC. Finally, conclusions are drawn in Section V.

II. System Model

The system model under study is shown in Fig. 1. Information bits are first encoded with an ECC such as RS or BCH code (i.e., the outer code), and the output of the ECC is then encoded with an LDPC code (i.e., the inner code). LDPC

[Copia del paper publicado en ArXiv-2011, pg.2/4]

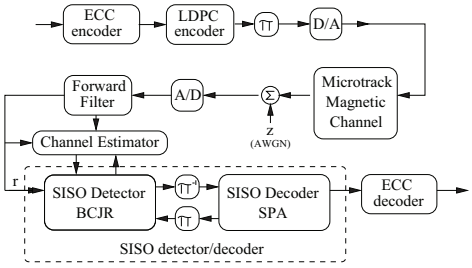


Figure 1. System Model

codes considered here are T/SPC and TPC/SPC. Codewords at the LDPC output are interleaved using a random-design interleaver. At the receiver side, samples are first processed by an adaptive linear feed forward filter (FFF) in order to adapt the channel response to the desired target response. Finally, the FFF output is used by the iterative detector to estimate the transmitted bit sequence. The SISO detector is implemented using the Max-Log-MAP approximation of the BCJR algorithm [14]. The SISO decoder runs with two iterations of the min-sum approximation of Sum-Product-Algorithm (SPA) [10].

A. Channel Model

The microtrack model [15] is used for the magnetic recording channel with signal dependent transition noise. The received signal is given by

$$s(t) = \frac{1}{N_t} \sum_{k} \sum_{n=1}^{N_t} b_k \cdot h(t - T_k - \tau_{k,n}) + z(t) \qquad (1)$$

where:

- $h(t)$ is the transition response of a simple microtrack. $h(t) = V \cdot erf\left(\frac{2\sqrt{ln(2)}\cdot t}{PW_{50}}\right)$, where V is the transition amplitude, PW_{50} is defined as the width of the derivative of $h(t)$ at half its peak amplitude.
- $b_k \in \{-1, +1\}$ represents the transitions direction.
- T_k is the ideal time in which the kth transition takes place.
- $\tau_{k,n}$ is the random shift of the kth transition on the nth microtrack.
- N_t is the total number of microtracks ($N_t = 2$ is used in this paper).
- $z(t)$ is electronic noise.

The density is defined as $D = \frac{PW_{50}}{T}$, where T is the bit period. The transition jitter noise $\tau_{k,n}$ is modeled as an i.i.d. white Gaussian random variable. The signal-to-noise ratio (SNR) is defined as $SNR = \frac{1}{\sigma_z^2 + \sigma_j^2}$ ($V = 1$ is assumed), where σ_z^2 is the power of the electronic noise samples, and σ_j^2 the power of the media noise component [9]. Media noise is defined by $s_j(t) = s_{sj}(t) - s_s(t)$, where $s_{sj}(t)$ is the microtrack channel output and $s_s(t)$ the jitter-free microtrack channel output.

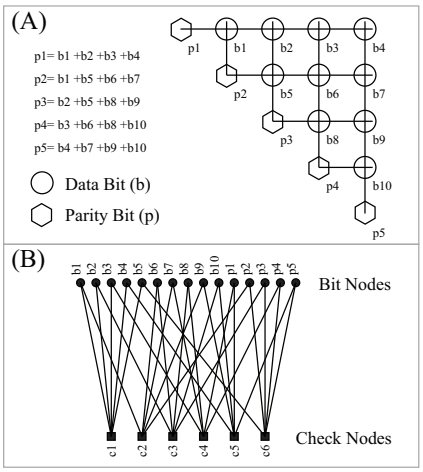


Figure 2. A) Parity bits structure of a T/SPC(5) code B) Tanner graph of a T/SPC(5) code.

III. The Triangle/SPC Code

T/SPC code is a subclass of LDPC codes, obtained from the combination of single parity check codes. A special case is the two-dimensional T/SPC, denoted as 2D-T/SPC, where the data and parity bits of the codeword are arranged in a 2-dimensional triangular array as shown in Fig. 2-A. Its Tanner graph is depicted in Fig. 2-B. Note that the 2D-T/SPC is a systematic code with the single parity bits as redundancy. This code can be completely characterized by the number of nodes in the edges of the triangle, N. The parameters of a 2D-T/SPC(N) code are:

Minimum dist.:	$d_{2,N}$	$= 3$
Code length:	$n_{2,N}$	$= N(N+1)/2$
Code dimension:	$k_{2,N}$	$= N(N-1)/2$
Code rate:	$R_{2,N}$	$= (N-1)/(N+1)$
Check equations:	$c_{2,N}$	$= N+1$

A 2D-T/SPC code can be extended to an M-dimensional T/SPC (MD-T/SPC) code by building an *M-dimensional triangular array*. The code parameters of the resulting MD-T/SPC(N) are:

Minimum dist.:	$d_{M,N}$	$= M+1$
Code length:	$n_{M,N}$	$= \sum_{i=1}^{N} n_{M-1,i}$
Code dimension:	$k_{M,N}$	$= \sum_{i=1}^{N} k_{M-1,i}$
Code rate:	$R_{M,N}$	$= k_{M,N}/n_{M,N}$
Check equations:	$c_{M,N}$	$= \sum_{i=1}^{N} c_{M-1,i}$

We focus on the 2D-T/SPC code that achieves a higher code

[Copia del paper publicado en ArXiv-2011, pg.3/4]

rate to code-length ratio than the 2D-TPC/SPC. We first show that the 2D-T/SPC code, as the 2D-TPC/SPC, is a special case of *Combinatorial Design Codes*, and later the lower complexity of the T/SPC compared with the TPC/SPC.

A. T/SPC and Combinatorial Design Codes

The T/SPC can be analyzed using *combinatorial design*. A combinatorial design is an arrangement of a set of m points into n subsets, called blocks, which satisfy certain regularity constraints [9], [11], [12], [13]. The *covalency* λ_{v_1,v_2} of two points v_1 and v_2 is the number of blocks that contain both of them. If λ_{v_1,v_2} is the same for all pairs of points, the design is said to be *balanced*. The number of points contained in each block and the number of blocks each point is incident with, are denoted by γ and ρ, respectively. If γ and ρ are the same for each block and point, respectively, the design is said to be *regular*. A regular and balanced design is denoted as a $(m, n, \rho, \gamma, \lambda)$-design. The *incidence* matrix M of the combinatorial design has dimension $n \times m$, where $M_{i,j} = 1$ if the point v_j is incident with the block B_i and $M_{i,j} = 0$ otherwise.

The transpose of the incidence matrix may be used as the parity check matrix H of an LDPC code. In this construction of an LDPC code, a point in a combinatorial design corresponds to a *check* node or a row in H, and a block corresponds to a *bit* node or a column in H. The row and column weights of H are ρ and γ respectively. A covalency $\lambda < 2$ guarantees the absence of length-4 cycles in the Tanner graph. For example, the 2D-TPC/SPC(N,N-1)^2 can be constructed from a $(2\rho, \rho^2, \rho, 2, \{0, 1\})$-design [13] where $\rho = N$.

One of the most interesting classes of combinatorial design are the *Steiner systems* [11], [16]. A (m, γ, t)-Steiner-system is a set $\mathcal{M}$ of m points, and a collection $\mathcal{N}$ of subsets of $\mathcal{M}$ of size γ, called blocks, such that any subset of t points of $\mathcal{M}$ is in exactly one of the blocks. The size of a Steiner system, n, is defined as the number of blocks:

$$n = |\mathcal{N}| = \binom{m}{t} \Big/ \binom{\gamma}{t} \tag{2}$$

In [11], [12], [16] the construction and performance of LDPC codes based on a $(m, 3, t)$-Steiner-system (called Steiner triple systems), and a particular case called Kirkman system, are analyzed. We will focus on the $(N-1, 2, 2)$-Steiner-system which produces a $(m, n, \rho, \gamma, \lambda)$-design with parameters: $n = N(N+1)/2$, $m = N+1$, $\rho = N$, $\gamma = 2$, $\lambda = 1$. The code obtained from the incidence matrix of this design is the 2D-T/SPC(N).

B. Complexity of T/SPC and TPC/SPC

The code rate of a 2D-TPC/SPC(N,$N-1$)^2 is given by

$$\begin{aligned} R_{TPC/SPC} &= \left(\frac{N-1}{N}\right)^2 \\ &= \frac{N-1}{N+1} - \frac{1}{N(N+1)} + \frac{1}{N^2(N+1)} \\ &\approx \frac{N-1}{N+1} = R_{T/SCP} \quad for\ N >> 1 \end{aligned} \tag{3}$$

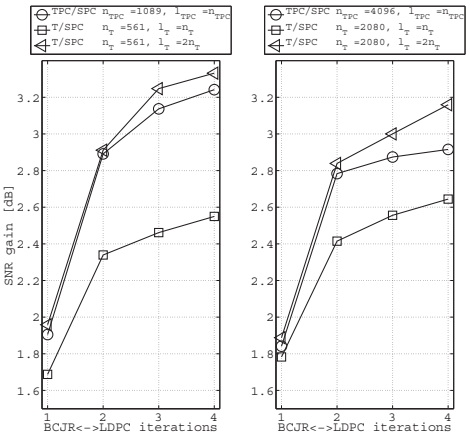


Figure 3. SNR gain vs. number of BCJR↔LDPC iterations at BER= 10^{-4}. Code rates 0.94 (left) and 0.969 (right).

Therefore, a simple complexity comparison between 2D-T/SPC and 2D-TPC/SPC can be done considering the same parameter N for both codes.

Given that complexity is mainly on the decoder, we will focus on the SPA-based decoder. T/SPC requires approximately the same bit-nodes and check-nodes computations as TPC/SPC. In general, the memory requirement and the interconnection complexity are proportional to the number of edges (N_e) of the code factor graph. Considering that the number of edges is $N_e = N(N+1)$ for T/SPC and $N_e = 2N^2$ for TPC/SPC, we conclude that T/SPC is able to provide a significant reduction of memory and interconnection complexity ($\sim 1/2$ for $N >> 1$).

IV. Numerical Results

The performance of T/SPC and TPC/SPC is analyzed by using computer simulations of the system described in Section II. Code rates of 0.94 and 0.969 are analyzed. We consider a perpendicular channel of density $D = 3$ and 80% / 20% jitter/electronic noise power (i.e., $\sigma_j^2/\sigma_z^2 = 0.8/0.2$). A generalized partial response target with 4 taps (GPR4) is utilized. The LDPC decoder performs two SPA iterations. Random-design interleavers are used.

Let l_T and l_{TPC} be the interleaver lengths used on the T/SPC and TPC/SPC codes respectively. Also, let n_T and n_{TPC} be the code-word lengths for the T/SPC and TPC/SPC codes respectively. For TPC/SPC we use $l_{TPC} = n_{TPC}$. On the other hand, we analyze T/SPC with two interleaver lengths: $l_T = n_T$ and $l_T = 2n_T \approx l_{TPC}$. Figure 3 shows the SNR gain vs. number of BCJR↔LDPC iterations at BER= 10^{-4}. The performance achieved by an uncoded BCJR equalizer is

[Copia del paper publicado en ArXiv-2011, pg.4/4]

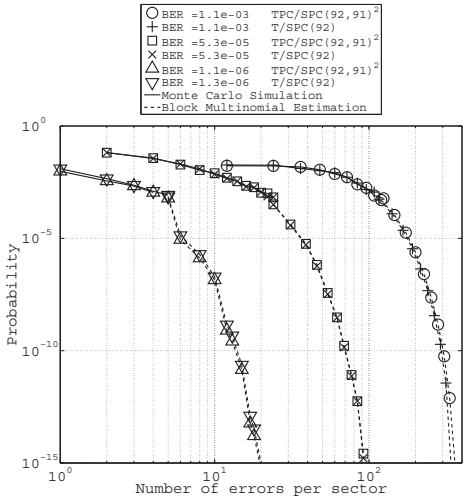


Figure 4. Probability of the number of errors per sector over a PR[1 3 3 1] with 100% electronic noise. Four TPC/SPC$(92, 91)^2$ codes and eight T/SPC(92) codes per sector.

taken as reference. The resulting performance of T/SPC for $l_T = n_T$ is worse than the one of TPC/SPC, mainly because of the T/SPC lower minimum distance. However, for $l_T = 2n_T \approx l_{TPC}$ the interleaver gain compensates the degradation caused by its lower minimum distance. Furthermore, in some cases note that T/SPC is able to provide a gain of around 0.27 dB over TPC/SPC owing to the interleaving gain. Note that this gain is achieved with lower complexity.

In the iterative-based receiver depicted in Fig. 1, the statistics of the bit errors at the output of the inner code strongly affect the performance of the outer code. Next we investigate the distribution of bit errors at the output of the T/SPC and TPC/SPC with code rate 0.978 and two BCJR↔LDPC iterations. Sector size is 4KB and the interleaving length is 1KB. Figure 4 shows the probability of the number of errors per sector for a partial response channel PR[1 3 3 1] with 100% electronic Gaussian noise. Solid lines correspond to values derived from simulations, while dashed lines represent estimates calculated by using the block-multinomial model [17]. From Fig. 4 we note that, at a given BER, the statistics of bit errors at the output of T/SPC are practically the same as those observed with TPC/SPC. Therefore, we infer that the behavior of T/SPC and TPC/SPC in an iterative-based receiver as shown in Fig. 1, should be similar. This topic, and other related issues, will be addressed in a future work.

V. Conclusions

We have proposed and investigated T/SPC, a novel high-rate LDPC code. T/SPC provides higher code rates, shorter code words, and lower complexity than TPC/SPC. Computer simulations of T/SPC on high-density perpendicular magnetic recording channels have shown that T/SPC is able to achieve a significant interleaving gain, outperforming TPC/SPC by almost 0.3 dB. Our results suggest that the design of LDPC with high-rate, low complexity and short code word is a promising research area for next generation developments of magnetic recording devices.

References

[1] H. Song, R. M. Todd, and J. R. Cruz, "Low density parity check codes for magnetic recording channels," *IEEE Trans. on Magnetics, vol. 36, no. 5, pp. 2183-2186*, September 2000.

[2] T. Morita, Y. Sato, and T. Sugawara, "ECC-less LDPC coding for magnetic recording channels," *IEEE Trans. on Magnetics, vol. 38, no. 5, pp. 2304-2306*, September 2002.

[3] X. Hu and B. V. K. V. Kumar, "Evaluation of low-density parity-check codes on perpendicular magnetic recording model," *IEEE Trans. on Magnetics, vol. 43, no. 2, pp. 727-732*, February 2007.

[4] L. Sun, H. Song, B. V. K. V. Kumar, and Z. Keirn, "Field-programmable gate-array-based investigation of the error floor of low-density parity check codes for magnetic recording channels," *IEEE Trans. on Magnetics, vol. 41, no. 10, pp. 2983-2985*, October 2005.

[5] C. A. Cole and E. K. Hall, "A general method for finding low error rates of LDPC codes," *IEEE*, May 2006.

[6] M. Stepanov and M. Chertkov, "Instanton analysis of low-density parity-check codes in the error-floor regime," *IEEE ISIT*, July 2006.

[7] T. Morita, M. Ohta, and T. Sugawara, "Efficiency of short LDPC codes combined with long Reed-Solomon codes for magnetic recording channels," *IEEE Trans. on Magnetics, vol. 40, no. 4, pp. 3078-3080*, July 2004.

[8] J. Li, K. R. Narayanan, E. Kurtas, and C. N. Georghiades, "On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels," *IEEE Trans. on Communications, vol. 50, no. 5, pp. 723-734*, May 2002.

[9] B. Vasic and E. M. Kurtas, "Coding and signal processing for magnetic recording systems," *CRC PRESS Book*, 2005.

[10] J. Li, K. Narayanan, and C. Georghiades, "Product accumulate codes: A class of codes with near-capacity performance and low decoding complexity," *IEEE Trans. on Inf. Theory. vol. 50, no. 1, pp. 31-46*, January 2004.

[11] B. Vasic, "Structured iteratively decodable codes based on Steiner systems and their application in magnetic recording," *Proc. GLOBECOM, San Antonio, TX, vol. 5, pp. 2954-2960*, November 2001.

[12] B. Vasic, E. M. Kurtas, and A. V. Kuznetsov, "Kirkman systems and their application in perpendicular magnetic recording," *IEEE Trans. on Magnetics, vol. 38, no. 4, pp. 1705-1710*, July 2002.

[13] J. Li and E. Kurtas, "A class of high-rate, low complexity, well-structured LDPC codes from combinatorial design and their applications on ISI channels," *Proc. Intl. Conf. on Commun., Internet anf Inform. Tech. (CIIT), pp. 418-425*, November 2002.

[14] P. Robertson, P. Hoeher, and E. Villebrun, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding," *ETT, 8(2):119-125*, March-April 1997.

[15] M. N. Marrow, M. K. Cheng, P. H. Siegel, and J. K. Wolf, "A fast microtrack simulator for high-density perpendicular recording," *IEEE Trans. on Magnetics. vol. 40, no. 4, pp. 3117-3119*, July 2004.

[16] D. J. MacKay and M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," *Proc. of the IMA Workshop on Codes, System and Graphical Models*, 1999.

[17] Z. A. Keirn, V. Y. Krachkovsky, E. F. Haratsch, and H. Burger, "Use of redundant bits for magnetic recording: Single-parity codes and Reed-Solomon error-correcting code," *IEEE Trans. on Magnetics, vol. 40, no. 1*, January 2004.

[Copia del paper publicado en RPIC-2011, pg.1/6]

RPIC2011 Estudiantil

Simulador para códigos QC-LDPC no binarios

Juan Manuel López Simão

Supervisores: Dra. Ing. Cecilia Galarza‡

Ing. Damián Morero†

‡ *LPSC, Departamento de Electrónica, Facultad de Ingeniería Buenos Aires*
Universidad de Buenos Aires, Buenos Aires, Argentina
cgalar@fi.uba.ar

† *Facultad de Ciencias Exactas Físicas y Naturales*
Universidad Nacional de Córdoba
dmorero@efn.uncor.edu

***Resumen*— En este trabajo se presenta una herramienta de simulación por computadora diseñada para códigos de paridad de baja densidad (LDPC) no binarios (NB-LDPC), programada en lenguaje C++. El desarrollo de la misma se vio motivado por la idea de poder predecir el desempeño de códigos NB-LDPC cuasi-cíclicos (NB-QC-LDPC), como potenciales candidatos a ser implementados en las etapas de corrección de errores para discos de grabación magnética. El simulador diseñado permitió evaluar el desempeño de códigos NB-QC-LDPC bajo esquemas de decodificación iterativa FFT-QSPA. Los resultados obtenidos impulsan el interés sobre la aplicación de códigos NB-QC-LDPC con estructuras afines desde el punto de vista del hardware, alcanzando mayores densidades de grabación que las ofrecidas por los códigos RS, actualmente utilizados como estándares de la industria** [1]**.**

***Keywords*— NB-QC-LDPC, SPA, FFT**

1. INTRODUCCIÓN

El aumento en la capacidad de almacenamiento de los discos rígidos, se debió en gran parte a la incorporación de técnicas propias del campo de las telecomunicaciones en el diseño de nuevos discos. Los procesos de escritura-lectura en un medio magnético y el de la transmisión-recepción en un sistema de comunicaciones tienen muchos aspectos en común. Estas similitudes han permitido aplicar técnicas avanzadas de procesamiento de señales y codificación sobre canales de grabación magnética. Particularmente, una de las principales limitaciones en el aumento de la densidad de datos en los discos, se debe a la superposición entre los pulsos que representan los bits. Utilizando la analogía con un canal de comunicaciones, este impedimento es eficazmente atacado abordándolo como un problema de interferencia intersimbólica (ISI) en un canal selectivo en frecuencia. La presencia de ISI es el principal factor de degradación de performance en los canales de grabación magnética. Diversas técnicas de ecualización combinadas con esquemas correctores de errores proveen la base para mitigar los efectos de ISI en este ámbito. A principios de la década de 1990, los esquemas de detección y corrección PRML (*Partial Response Maximum Likelihood*) aportaron un aumento de la densidad lineal de almacenamiento en discos magnéticos cercano al 50% [2]. La búsqueda de mayores densidades de grabación condujo a las llamadas arquitecturas NPML (*Noise-Predictive Maximum Likelihood*) combinadas con códigos de multiparidad hacia finales de la década del '90, mejorando aún más la tasa de error de bit del canal interno [3]. Dados los niveles de eficiencia que se han alcanzado en el proceso de decodificación y capacidad de corrección de códigos Reed-Solomon (RS), la mayoría de los sistemas actuales de almacenamiento magnético incorpora este tipo de esquema corrector. A medida que se incrementa la densidad de almacenamiento en este tipo de medios, los efectos de ISI son cada vez más severos, y los códigos RS no logran alcanzar la performance requerida. Actualmente es sabido que los códigos LDPC superan en performance a códigos RS tanto sobre canales AWGN como en canales de ruido por ráfagas [4]. Uno de los desafíos actuales conlleva a evaluar la performance de estos códigos sobre canales reales de escritura/lectura en material magnético [5]. A su vez, el carácter altamente estructurado de los códigos LDPC cuasi-cíclicos despierta el interés de implementar esta clase de códigos de manera sistemática y eficiente.

La sección **2** presenta las nociones elementales que contextualizan a los códigos LDPC en esquemas de detección y corrección de errores. La sección **3** está dedicada a la descripción del simulador desarrollado para códigos LDPC no binarios. En la sección **4** se incluyen los resultados de desempeño de los códigos NB-QC-LDPC simulados sobre canales AWGN. En la sección **5** se exponen las conclusiones obtenidas a partir de los resultados, junto con el desarrollo e implementa-

[Copia del paper publicado en RPIC-2011, pg.2/6]

RPIC2011 Estudiantil

ción a futuro que tiene el presente trabajo, en el ámbito de los sistemas de grabación magnética.

2. CÓDIGOS EN BLOQUE

2.1. Esquemas de Comunicación para la Detección y Corrección de Errores

A diferencia de los sistemas ARQ (*Automatic Repeat reQuest*), donde la corrección de errores puede efectuarse pidiendo una repetición de la transmisión, los esquemas de transmisión FEC (*Forward Error correction*) no contemplan la posibilidad de solicitar una retransmisión de datos cuando un error es detectado. En casos como estos, se requerirá de la implementación de algún código corrector de errores, a fines de efectuar una transmisión confiable de información. En un esquema de transmisión FEC, la fuente de información discreta le envía un mensaje integrado por k bits al dispositivo codificador. Este último producirá una palabra-código de longitud $n > k$, mediante el agregado conveniente de redundancia. El mensaje originalmente producido por la fuente podrá o no, aparecer literalmente en la palabra-código resultante. Los códigos correctores de errores que incluyan al mensaje sin modificar en la palabra-código son llamados sistemáticos, caso contrario se tratará de un código no sistemático. La relación k/n es la denominada tasa R_c del código y da una medida de la cantidad de redundancia añadida para lograr la corrección del error.

2. 2. Códigos en Bloque Sistemáticos Lineales

En un esquema de codificación por bloque, la secuencia de información binaria es segmentada como bloques de mensaje de una longitud fija igual a k bits. De este modo, se tendrá un total de 2^k mensajes diferentes. De acuerdo a ciertas reglas, el codificador convertirá unívocamente cada mensaje en una secuencia binaria o palabra-código de longitud $n > k$. El juego de 2^k palabras-código es referido como código en bloque. Usualmente, suelen utilizarse códigos en bloque cuyos procesos de codificación puedan efectuarse de manera práctica y simplificada. Una de estas características se basa en disponer de linealidad en las operaciones que hacen a la codificación de cada mensaje. A su vez, una propiedad deseable para los códigos en bloque lineales es poder hacer uso de estructuras sistemáticas en sus palabras-código, donde cada una de ellas puede verse como la unión de dos segmentos: el que contiene al mensaje como secuencia inalterada de k dígitos, y la parte que abarca los $(n–k)$ dígitos de redundancia. La fig. 1 muestra esta clase de estructura.

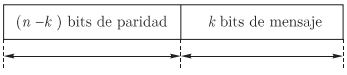


Figura 1: Forma sistemática de una palabra-código perteneciente a un código en bloque lineal sistemático

2.3 Descripción Vectorial de un Código en Bloque Lineal

Un código en bloque lineal binario $C_b(n, k)$ de longitud n es convenientemente descripto sobre el espacio vectorial n-dimensional, definido en el campo binario o Campo de Galois GF(2). El código C_b queda definido unívocamente a partir de k vectores linealmente independientes $\{\boldsymbol{g}_0, \boldsymbol{g}_1, ..., \boldsymbol{g}_{k-1}\}$ tales que cada palabra-código de C_b sea obtenida como combinación lineal de los mismos. Estos vectores son usualmente dispuestos como los vectores fila de la matriz generadora $\mathbf{G}$ de C_b. Cada palabra-código será obtenida a través del producto interno entre el mensaje $\boldsymbol{m} = (m_0, m_1, ..., m_{k-1})$ y la matriz generadora:

$$\boldsymbol{c} = \boldsymbol{m} \circ \mathbf{G} = m_0 \bullet \boldsymbol{g}_0 \oplus m_1 \bullet \boldsymbol{g}_1 \oplus ..... \oplus m_{k-1} \bullet \boldsymbol{g}_{k-1} . \quad (1)$$

La contrapartida dual del juego de vectores $\{\boldsymbol{g}_0, \boldsymbol{g}_1, ..., \boldsymbol{g}_{k-1}\}$ que definen el espacio fila de $\mathbf{G}$, es el conjunto de $n–k$ vectores linealmente independientes $\{\boldsymbol{h}_0, \boldsymbol{h}_1, ..., \boldsymbol{h}_{n-k-1}\}$, donde todo vector $\boldsymbol{h}_j$. resulta ortogonal a todo vector $\boldsymbol{g}_i$ ($\boldsymbol{g}_i \circ \boldsymbol{h}_j = 0$) y viceversa. Este conjunto de $n–k$ vectores genera el espacio fila de la matriz de paridad $\mathbf{H}$ que también define unívocamente al código. Si $C_b(n,k)$ es además un código sistemático, ambas matrices podrán obtenerse en sus respectivas formas sistemáticas como:

$$\mathbf{G} = [\, \mathbf{P} \quad \mathbf{I}_k \,], \quad (2)$$

$$\mathbf{H} = [\, \mathbf{I}_{n-k} \quad \mathbf{P}^{\mathrm{T}} \,]. \quad (3)$$

La estructura sistemática de $\mathbf{G}$ en la ec. (2) conlleva a la forma sistemática de palabra-código de la fig. 1. En casos como estos, los bits de paridad pueden obtenerse en función de los bits de mensaje a través de las siguientes ecuaciones:

$$\begin{aligned} & c_{n-k+i} = m_i \,, \\ & c_j = m_0 \bullet p_{0j} \oplus m_1 \bullet p_{1j} \oplus ..... \oplus m_{k-1} \bullet p_{k-1,j} \,, \\ & 0 \le j \le n-k-1 \,. \end{aligned} \quad (4)$$

Estas $(n–k)$ ecuaciones son las llamadas ecuaciones de paridad del código lineal $C_b(n, k)$ y son descriptas equivalentemente tanto por la matriz generadora como por la matriz de paridad del código. La ortogonalidad entre las filas de $\mathbf{H}$ y las filas de $\mathbf{G}$, provoca que toda palabra-código $\boldsymbol{r}$ perteneciente a C_b verifique:

$$\boldsymbol{r} \circ \mathbf{H}^{\mathrm{T}} = \mathbf{0} \,. \quad (5)$$

2.4. Códigos LDPC

Los códigos LDPC presentados por Robert G. Gallager en su tesis doctoral de 1962 [6], son códigos en bloque lineales, construidos a partir del diseño de una matriz de paridad rala, esto es, una matriz cuya cantidad de elementos no nulos es reducida. En la versión binaria de estos códigos, esta característica denota una matriz $\mathbf{H}$ con un número acotado de ‘1’s, distribuidos sobre un porcentaje mayoritario de ‘0’s. Gallager definió origi-

[Copia del paper publicado en RPIC-2011, pg.3/6]

RPIC2011 Estudiantil

nalmente los códigos LDPC binarios (n, γ, ρ) para presentar una longitud de bloque igual a n, y una matriz de paridad con exactamente γ '1's por columna y ρ '1's por fila, donde $\rho \geq 3$. El valor de γ es referido como el peso (esto es, la cantidad de elementos no nulos) de cada columna de **H**, mientras que ρ indicará el peso por fila de tal matriz. En casos como éste, el código LDPC definido es habitualmente referido como "(γ,ρ)-regular". Si las filas o columnas de la matriz de paridad presentan diferentes valores de peso, se tratará de un código LDPC irregular. Diversas construcciones para códigos LDPC binarios, fueron elaboradas y presentadas por Gallager en su trabajo original, junto con un método de decodificación suave (probabilística) para los mismos. La complejidad de cómputo del mismo trascendía la capacidad de los procesadores electrónicos disponibles en aquel entonces. Los códigos LDPC fueron relegados hasta el año 1995, cuando fueron redescubiertos por Mackay y Neal [7], proponiendo un algoritmo de decodificación que resultó idéntico al desarrollado por Gallager. Este algoritmo, actualmente conocido como Algoritmo Suma-Producto (SPA, *Sum-Product Algorithm*) obtiene la palabra-código $\hat{\boldsymbol{c}}$ que maximiza las probabilidades a posteriori de cada dígito o símbolo de la palabra. La probabilidad a posteriori de cada palabra depende de las características del canal, de las ecuaciones de paridad que definen al código, y de la palabra-código recibida a la salida del canal. Michael Tanner [8] generalizó las representaciones gráficas utilizadas para la descripción de los códigos de Gallager, a partir de grafos bipartitos sobre los cuales el SPA es convenientemente descripto. Cada matriz de paridad presenta un grafo bipartito de Tanner definido a través de dos clases disjuntas de nodos: los nodos de variable c_j (representando al j-ésimo dígito codificado o columna de **H**) y los nodos de paridad h_i (representando a la i-ésima ecuación de paridad o fila de **H**). En cada grafo de Tanner, sólo existirán conexiones entre nodos de clases distintas, donde h_i y c_j serán conectados por una arista sí y sólo sí $\mathbf{H}\{i, j\} \neq 0$. Bajo este contexto, el SPA puede entenderse como un intercambio de mensajes probabilísticos que viajan sobre las aristas del grafo. Cuando el grafo presenta una estructura de árbol, el pasaje de mensajes conducirá exactamente a los valores de probabilidades MAP para cada dígito codificado. Cuando el grafo presente uno o más ciclos (esto es, una secuencia de 4 ó más aristas que comienza y termina en el mismo nodo), los mensajes pasan múltiples veces por determinadas aristas, conduciendo a un algoritmo iterativo sin una finalización exacta hacia los valores de probabilidades MAP. En casos como estos, el algoritmo es usualmente detenido en la iteración donde $\hat{\boldsymbol{c}}$ o $\mathbf{H}^{\mathrm{T}}$ valga **0**, ó bien cuando se haya excedido un número de iteraciones predefinido sin alcanzar tal condición. Si bien la aparición de ciclos en aplicaciones prácticas de códigos LDPC eficientes es inevitable [9], un buen diseño de tales códigos permite obtener resultados bastante próximos a los ideales. Davey y Mackay generalizaron el algoritmo SPA para códigos NB-LDPC [10] definidos sobre extensiones del campo binario con $Q=2^m$ elementos. La complejidad de cómputo de esta versión del SPA, habitualmente referida como QSPA, fue posteriormente reducida por los mismos autores mediante la incorporación de un esquema basado en transformadas rápidas de Fourier (FFT) [11]. Este algoritmo FFT-QSPA sirvió como punto de partida para diseños de decodificadores LDPC no binarios con una complejidad de implementación manejable [12].

El grafo factorial de un código NB-LDPC permite describir gráficamente la operación del algoritmo suma producto para este tipo de códigos. Para la siguiente matriz de paridad de un código NB-LDPC construido sobre $GF(2^2)$:

$$\mathbf{H} = \begin{bmatrix} \alpha & 0 & 1 & \alpha & 0 & 1 \\ \alpha^2 & \alpha & 0 & 1 & 1 & 0 \\ 0 & \alpha & \alpha^2 & 0 & \alpha^2 & 1 \end{bmatrix} \tag{6}$$

El grafo factorial presenta la estructura de la fig.2:

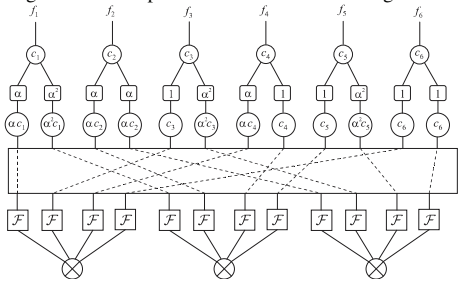


Figura 2: Grafo factorial del código NB-LDPC definido por la matriz de la ec. (6)

Es sabido que los códigos LDPC no binarios adecuadamente construidos, no sólo superan en performance a sus contrapartidas binarias, sino que también muestran desempeños muy buenos sobre varios tipos de canales [4]. Diversos tipos de construcciones, tanto aleatorias como determinísticas, pueden utilizarse para definir códigos LDPC no binarios sobre campos finitos de la forma $GF(2^m)$. Estos campos contienen el conjunto finito de elementos $\{0, 1, \alpha, \alpha^2, \ldots\ldots, \alpha^{2^m-2}\}$ (siendo α el llamado elemento primitivo de tal campo), y pueden construirse a partir de representaciones por bases de polinomios irreducibles [13]. Las construcciones de códigos LDPC basadas en campos finitos muestran un buen desempeño sobre canales AWGN a través de decodificación iterativa basada en el SPA [14], además de exhibir bajos pisos de error, volviéndose aptos para diversas aplicaciones en sistemas de comunicaciones y de almacenamiento de datos. A su vez, la mayoría de las construcciones de códigos LDPC sobre bases de campos finitos son cuasi-cíclicas, permitiendo implementar estructuras de codificación a través de registros

[Copia del paper publicado en RPIC-2011, pg.4/6]

RPIC2011 Estudiantil

de desplazamiento simples con complejidad lineal. El desempeño de dos clases de códigos NB-QC-LDPC basados en este tipo de construcciones [14] fue cuantificado por el simulador descripto enseguida.

3. DESCRIPCIÓN DEL SIMULADOR

Esta herramienta cuantifica la performance FER (*Frame Error Rate*) del código LDPC no binario, evaluada sobre **t** relaciones señal-ruido, y transmitiendo un total de L palabras-código por cada una de ellas. Esta cantidad es calculada como el cociente entre el total de bits transmitidos por cada SNR y la longitud ($n \cdot m$) en bits de cada palabra-código. Se ha hecho uso a su vez, de señalizaciones BPSK para cada palabra-código, bajo simulaciones de canal AWGN registradas mediante histogramas. La fig. 3 detalla el esquema completo del simulador, enumerando el orden en que el mismo efectúa cada operación:

(**1**) Construcción de la matriz de paridad **H**, en función del campo GF(2^m) y del método de construcción elegido [14]. Cada elemento no nulo de **H** podrá tomar uno de los valores $\{1, \alpha, \alpha^2, ..., \alpha^{2^m-2}\}$.Generalmente, **H** presentará varias filas linealmente dependientes. (**2**) Deducción de la matriz generadora del código a partir de **H**. A los fines del simulador, la matriz **G** es obtenida mediante el proceso de eliminación de Gauss-Jordan efectuado sobre una matriz **H´**, réplica de **H**. Luego de la ejecución del mismo, se removerán las filas linealmente dependientes en **H´**, resultando en una matriz de rango completo. Si **H´** se encuentra en su forma sistemática $[\mathbf{I}_{n-k} \quad \mathbf{P}^{\mathrm{T}}]$, **G** podrá ser obtenida como $[\mathbf{P} \quad \mathbf{I}_k]$. Caso contrario, deberán aplicarse las conmutaciones de columnas en **H´** tales que permitan llevarla a la forma $[\mathbf{I}_{n-k} \quad \mathbf{P'}^{\mathrm{T}}]$, luego **G** podrá será obtenida a partir de una matriz $[\mathbf{P'} \quad \mathbf{I}_k]$ a la que se le aplicarán las desconmutaciones de columnas, correspondientes a las conmutaciones efectuadas sobre **H´**.

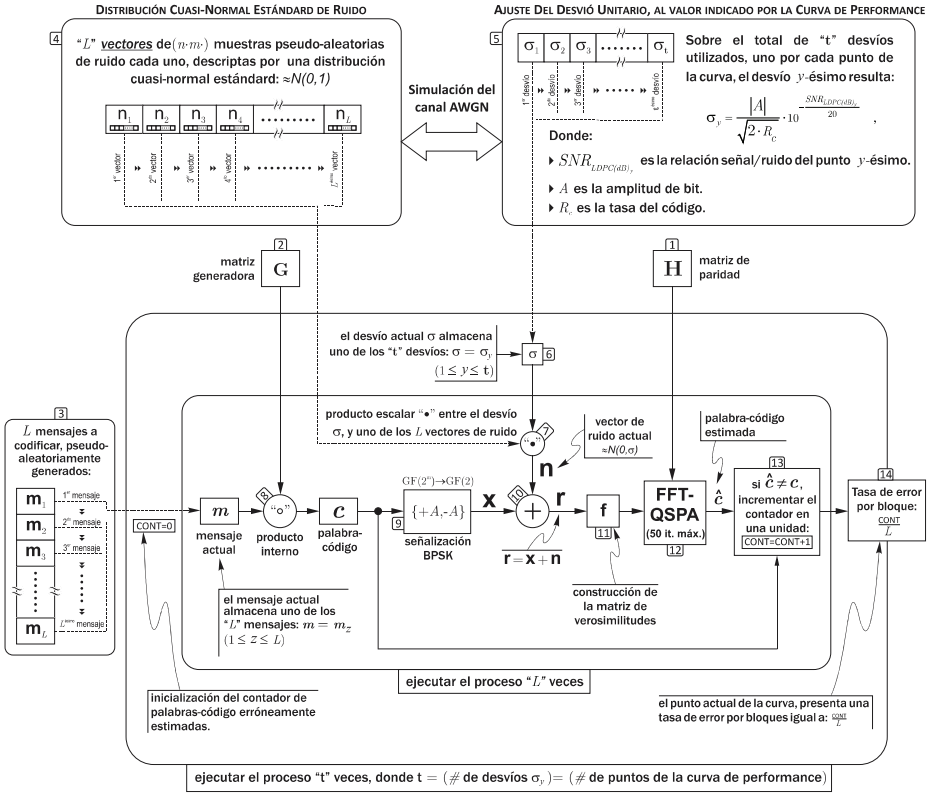


Figura 3: Diagrama en bloques del simulador

[Copia del paper publicado en RPIC-2011, pg.5/6]

RPIC2011 Estudiantil

(**3**) Generación pseudo-aleatoria de L mensajes a codificar, a partir de $(L \cdot k)$ valores pertenecientes a una distribución cuasi-uniforme que asigna elementos del conjunto $\{0, 1, \alpha, \alpha^2, ..., \alpha^{2^m-2}\}$.(**4**) Generación pseudo-aleatoria de L vectores $\mathbf{n}_z$ de $(n \cdot m)$ muestras de ruido cada uno, pertenecientes a una distribución cuasi-normal estándar ~N(0,1). Lo anterior es conseguido a través de la transformación de Box-Muller:

$$\mathbf{n}_z = \sqrt{-2\,ln(\mathbf{u}_z)} \cdot \cos(2\pi \mathbf{v}_z), \qquad (7)$$

donde los vectores $\mathbf{u}_z$ y $\mathbf{v}_z$ son de longitud $(n \cdot m)$, con valores pseudo-aleatorios, pertenecientes a una distribución cuasi-uniforme en el intervalo [0,1]. (**5**) Cómputo de los **t** desvíos (ver fig. 3) necesarios, que ajustan el desvío unitario de la distribución ~N(0,1) al valor necesario para cada punto de la curva desempeño. Para el punto y-ésimo, las $(L \cdot n \cdot m)$ muestras de ruido pertenecerán a una distribución cuasi-normal ~N(0,σ_i). (**6**) Asignación del valor σ_y para el desvío σ actual, junto con la inicialización de la variable CONT en cero. (**7**) Cómputo del vector de ruido actual **n**, a través del producto escalar entre σ y el z-ésimo vector de ruido $\mathbf{n}_z$. (**8**) Codificación del mensaje z-ésimo a través del producto interno $\boldsymbol{m}_z$ o **G**. El mismo es computado como lo describe la ec. (1) respetando las sumas y productos efectuadas entre elementos del campo GF(2^m) elegido. (**9**) Construcción del vector **x**, como la señalización BPSK de la palabra-código $\boldsymbol{c}$, en base a la representación vectorial sobre GF(2) de cada elemento de GF(2^m). Una componente igual a **1** será transmitida con una amplitud de bit $+A$, mientras que un **0** será representado con un valor igual a $-A$. (**10**) Contaminación aditiva del vector **x** con el vector de ruido **n**. (**11**) Construcción de la matriz de verosimilitudes **f**, a partir del vector recibido **r**. Esta matriz, de dimensiones $2^m \times n$ almacena las estimaciones a priori de los n elementos de $\boldsymbol{c}$ como:

$$\mathbf{f}_{\{i\,j\}} = \begin{cases} P(\boldsymbol{c}_{\{j\}} = 0) & \text{si} \quad i = 1 \quad (1 \le j \le n) \\ P(\boldsymbol{c}_{\{j\}} = \alpha^{i-2}) & \text{si} \quad 2 \le i \le 2^m \ (1 \le j \le n) \end{cases}. \qquad (8)$$

En función de la representación vectorial sobre GF(2) de x, la probabilidad $P(\boldsymbol{c}_{\{j\}} = x)$ será calculada a través de la productoria adecuada de valores de probabilidades gaussianas:

$$g_j^{(0)} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(\mathbf{r}_{\{j\}}+A)^2}{2\sigma^2}}, \; g_j^{(1)} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(\mathbf{r}_{\{j\}}-A)^2}{2\sigma^2}}. \qquad (9)$$

(**12**) Inicialización y ejecución del FFT-QSPA a partir de los valores obtenidos en **f**. (**13**) Si la palabra-código estimada es distinta a $\boldsymbol{c}$, CONT será incrementado en una unidad. (**14**) Una vez transmitidas las L palabras-código, el valor de FER para el punto y-ésimo de la curva es obtenido como el cociente entre CONT y L.

4. RESULTADOS

Se ha evaluado el desempeño FER de dos clases de códigos QC-LDPC no binarios que figuran en [14] sobre campos de 16 y 32 elementos. El número máximo de iteraciones para el FFT-QSPA ha sido prefijado en un valor igual a 50, tal como lo especifica el trabajo citado. Se han simulado cantidades del orden de 200~600 millones de muestras de ruido, en función de las SNRs que requieren las mayores cantidades de datos, a fines de poder computar las menores tasas de error de bloque. Los valores obtenidos para tales desempeños se presentan en la tabla 1. Los mismos son superpuestos a las curvas de desempeño de [14] en las figs. 4 y 5.

QC-LDPC(120,71) sobre GF(2^4)		
$SNR_{LDPC(dB)}$	L	CONT
1dB	416.666	373.874
1,4dB	416.666	268.921
1,8dB	416.666	128.661
2,2dB	416.666	36.882
2,6dB	416.666	6.508
3,0dB	416.666	708
3,4dB	416.666	46
3,8dB	1.250.000	5

QC-LDPC(248,136) sobre GF(2^5)		
$SNR_{LDPC(dB)}$	L	CONT
1,00dB	80.645	80.559
1,25dB	80.645	79.368
1,50dB	80.645	72.352
1,75dB	80.645	53.261
2,00dB	112.903	38.164
2,25dB	112.903	12.477
2,50dB	112.903	2.605
2,75dB	112.903	358
3,00dB	161.290	61

Tabla 1: resultados de performance FER para los códigos NB-QC-LDPC evaluados

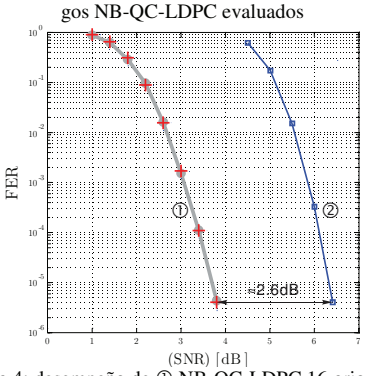


Figura 4: desempeño de ① NB-QC-LDPC 16-ario (4,8) regular, ② RS (120,71,50) sobre GF(2^7), alg. BM.

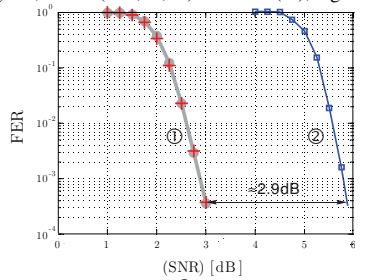


Figura 5: desempeño de ① NB-QC-LDPC 32-ario, ② RS (248,136,113) sobre GF(2^8), alg. BM.

[Copia del paper publicado en RPIC-2011, pg.6/6]

5. CONCLUSIONES

La herramienta de simulación presentada permitió cuantificar el desempeño de las construcciones de códigos QC-LDPC no binarios basadas en campos finitos que figuran en [14]. La verificación numérica de lo anterior ciertamente extiende el alcance del simulador desarrollado hacia cualquier otro método de construcción para códigos LDPC no binarios, factibles de ser implementados. En particular, las figs. 4 y 5 denotan que la performance de los códigos QC-LDPC evaluados sobre canales AWGN alcanza ganancias significativas sobre los desempeños conocidos de códigos RS a iguales tasas y longitudes. Comparaciones de esta índole impulsan el interés sobre la aplicación de códigos LDPC no binarios en canales de grabación magnética, donde los códigos RS son utilizados actualmente como estándares de la industria [1]. Dados los requerimientos de confiabilidad en los sistemas de almacenamiento actuales, resulta imperioso evaluar la performance de códigos LDPC para valores de BER por debajo de 10^{-12}, y valores de FER inferiores a 10^{-8}. Tales cantidades resultan impracticables de alcanzar mediante simulaciones por computadora. Esto motiva la simulación de canales de grabación magnética sobre hardware programable, a fines de poder estimar fidedignamente las tasas de error de bits más bajas. Lo anterior permitirá percibir el desempeño de la corrección de errores del código LDPC sobre modelos realísticos de canal, y también dará una idea del piso de error que presente el código por debajo de un cierto umbral de BER. Esto sugiere el uso de códigos LDPC con estructuras afines desde el punto de vista del hardware. En este aspecto, el carácter notablemente estructurado de los códigos QC-LDPC permite diseñar sistemática y eficientemente las etapas de codificación, mediante registros de desplazamiento simples con una complejidad lineal de implementación. A su vez, los esquemas de decodificación propuestos en trabajos como [12] resultan en considerables candidatos para la implementación de decodificadores LDPC no binarios con complejidad razonable, convirtiendo a esta clase de códigos en un serio contendiente para los códigos actualmente utilizados en sistemas de almacenamiento y comunicación de datos.

REFERENCIAS

[1] X.W.Shan y M.T.Alexander, Magnetic Information Storage Technology IBM. San Jose, CA: Academic, 1998.

[2] R.D.Cidecyan, F.Dolvio, R. Hermann, W.Hirt, "A PRML System for Digital Magnetic Recording" - IEEE Journal on Selected Areas in Comm., - 1992, vol.10, No.1, pp.38 -56.

[3] R. D. Cideciyan et al., "NPML Detection Combined with Parity-Based Post-Processing," IEEE Trans. Mag., vol. 37, Mar. 2001, pp. 714–20.

[4] J. Chen, L. Wang, Y. Li "Performance Comparison between Non-Binary LDPC Codes and Reed-Solomon Codes over Noise Bursts Channels", IEEE Int. Conf. on Comm. Circuits and Systems, 2005.

[5] H. Zhong, T. Zhong and E. F. Haratsch "Quasi-Cyclic LDPC Codes for the Magnetic Recording Channel: Code Design and VLSI Implementation", IEEE Trans. Mag., Vol. 43, no. 3, Marzo 2007

[6] R. G. Gallager, "Low Density Parity Check Codes," Sc.D. thesis, Mass. Inst. Tech., Cambridge; Sept. 1960.

[7] Mackay, D.J. y Neal, R.M. (1995) "Good codes based on very sparse matrices". Cryptography and Coding, 5th IMA Conference, Vol. 1025, pp. 100–11.

[8] Tanner, L. M., "A recursive approach to low complexity codes," IEEE Trans. Inf. Theory, vol. 27, no. 5, pp. 533–547, 1981.

[9] Etzion, T., Trachtenberg, A. andVardy, A., "Which codes have cycle-free Tanner graphs?" IEEE Trans. Inf. Theory, vol. 45, no. 6, pp. 2173–2180, Sept. 1999.

[10] M. Davey y D.J.C. MacKay, "Low Density Parity Check Codes over GF(q)," IEEE Commun. Lett., vol. 2, pp. 165-167, Junio 1998.

[11] D. J .C. Mackay y M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," in Codes, Systems and Graphical Models, ser. IMA Volumes in Mathematics and its Applications, B. Marcus and J. Rothenthal, eds. New York: Springer, 2000, vol. 123, pp. 113-130.

[12] D. Declercq y M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(q)", IEEE Trans. Comm., vol. 55, no. 4, pp. 633–643, Abr. 2007.

[13] D. Hankerson, A. Menezes, S. Vanstone "Guide to Elliptic Curve Cryptography", 2004 Springer-Verlag New York, Inc.

[14] S. Lin, S. Song, B. Zhou, et al. (2007) "Algebraic constructions of non-binary quasi-cyclic LDPC codes: array masking and dispersion". 9th Int. Symposium on Communication Theory and Applications (ISCTA), Ambleside, Lake District, UK.

[Copia del paper presentado en GlobeCom-2011, pg.1/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks

Damian A. Morero*†, M. Alejandro Castrillon*, Facundo A. Ramos†, Teodoro A. Goette†,
Oscar E. Agazzi‡ and Mario R. Hueda*

*Laboratorio de Comunicaciones Digitales - Universidad Nacional de Cordoba - CONICET
Av. Velez Sarsfield 1611 - Cordoba (X5016GCA) - Argentina
†ClariPhy Argentina S.A. - 12 de Octubre 1320 - Cordoba (5000) - Argentina
‡ClariPhy Communications, Inc. - 7585 Irvine Center Drive, Suite 100 Irvine, CA 92618, USA
Emails: dmorero, mhueda@efn.uncor.edu

***Abstract*—This paper presents a non-concatenated forward error correction (FEC) code suitable for applications in 100Gb/s optical transport networks (OTN). A typical requirement in this application is a net coding gain (NCG) > 10dB at a bit error rate (BER) of 10^{-15} with an overhead (OH) of $\sim 20\%$. As discussed in [1], non-concatenated codes are the ultimate frontier in terms of performance for OTN applications, because of their superior performance, lower latency, and lower overhead than concatenated codes. However, a major stumbling block for the use of these codes has been the existence of BER floors at levels significantly higher than the required 10^{-15} (typically 10^{-10}). In this paper we present a new coding scheme based on a low density parity check (LDPC) code with an expected net coding gain of 11.30dB at 10^{-15}, 20% OH, and a codeword length of 24576 bits. This represents a significant improvement over the previous state of the art [2], based on a concatenated code with a codeword length of 74844 bits and 20.5% OH. The code is designed to minimize the BER floor while simultaneously reducing the memory requirements and the interconnection complexity of the iterative decoder [3]. Experimental results obtained with an FPGA-based hardware emulator demonstrate an NCG of 10.70dB at a BER of 10^{-13} and no error floors. These experimental results are extrapolated to 10^{-15} using importance sampling techniques, resulting in the expected performance stated above. Moreover, we find that fixed-point implementation is the main cause of error floors below 10^{-13}. Based on this finding, we introduce a new low complexity postprocessing technique to push BER floors down to 10^{-15}.**

I. Introduction

In next generation coherent optical communication systems, powerful forward error correction (FEC) codes are required to achieve high net coding gain (NCG) (e.g., $\geq$ 10dB) at a bit error rate (BER) of 10^{-15}. Given their superior performance and suitability for parallel processing, large codeword length low density parity check (LDPC) codes are a promising solution for ultra-high speed optical fiber communication systems. Because of the large codeword length required to achieve high NCG, the use of low complexity soft decoding techniques such as the min-sum algorithm (MSA) is required when aiming at an efficient very large scale integration (VLSI). The main stumbling block for the application of this coding approach in optical transport networks (OTN) has been the fact that most LDPC codes suffer from BER floors above 10^{-15}. To reduce these error floors in MSA decoders, postprocessing algorithms have been reported in past work [4] [5] [6]. It has been shown that these techniques can reduce the error floor around two or three orders of magnitude. Since BER floors of reported LDPC codes are typically $\gtrsim 10^{-10}$, previous literature concluded that postprocessing is not sufficient for OTN applications. As a result, in previous work the error floor problem has been mitigated by concatenating an LDPC code with a hard-decision-based block code.

Several concatenated FEC schemes for 100 Gigabits per second (Gb/s) OTN applications have been proposed (see [1], [2] and references therein). In [1] it is experimentally shown that a 20.5% concatenated code based on an inner LDPC and an outer Reed-Solomon (RS) code achieves an NCG of 9 dB at a BER=10^{-13}. The concatenation of two hard-decision block codes with an LDPC is other alternative proposed in [1]. The total overhead of this triple-concatenated approach is 20% and the expected NCG is 10.80 dB at a BER=10^{-15}. A concatenated LDPC+RS coding scheme with 20.5% OH and 11.30 dB expected NCG at a BER=10^{-15} was proposed in [2]. This approach is based on a long quasi-cyclic (QC) LDPC (74844,63552) code with a 4-bit, 15-iteration, MSA decoder.

On the other hand, it is generally believed that LDPC-only codes may outperform concatenated codes with significantly reduced complexity. However, as mentioned before, their practical application to multigigabit optical systems has been precluded so far as a result of the error floor problem above 10^{-15}. In addition, the complex interconnection patterns inherent in these codes have been another major challenge for their VLSI implementation. Techniques to overcome these obstacles are presented in this paper, and a code suitable for application in 100Gb/s optical communications is proposed.

This work introduces a new high-gain, very low error floor, long LDPC-only based code for ultra-high speed fiber optical communication systems. The key ingredient of our code is a minimization of the BER floor to a level about three orders of magnitude below that reported in previous literature [7] [8]. This is achieved by combining semianalytic techniques with an FPGA-based study of dominant error events in order to optimize the parity check matrix. Although BER floors are not

978-1-4244-9268-8/11/$26.00 ©2011 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

completely eliminated, they are pushed to a sufficiently low level that postprocessing techniques can effectively eliminate them. Great attention is also paid to the optimization of the integrated circuit implementation architecture. The LDPC code and the hardware architecture are jointly designed in order to (*i*) minimize the error floor and (*ii*) reduce the amount of memory and interconnection complexity [3]. We develop a (24576,20482) (i.e., 20% OH) QC-LDPC-only code with a 5-bit MSA decoder for 100 Gb/s optical systems. An FPGA-based simulator is used to experimentally demonstrate that the proposed LDPC-only code achieves an NCG of 10.70dB at BER of 10^{-13} with a 13-iteration MSA decoder, and that the code is free of error floors down to 10^{-13}.

As mentioned before, a key problem addressed in this work is the estimation and reduction of very low error floors (i.e., $< 10^{-13}$). To accurately evaluate BER floors, we use a combination of analytical tools with FPGA simulation [9]. Our study shows that error floors below 10^{-13} are mainly caused by the fixed-point implementation of MSA. This finding was also reported in [8] for decoders based on the sum-product algorithm (SPA). We use this result to develop a postprocessing technique to combat the error floor in MSA decoders. Numerical results show that error floors can be drastically lowered, resulting in an expected NCG $\geq$ 11.30dB at a BER of 10^{-15}. This result agrees very well with that recently reported in [10], which has been derived by using pure FPGA simulations. Although this performance is similar to that achieved by the concatenated scheme reported in [2], in our approach not only a hard-decision-based block outer code is avoided, but also the codeword length is reduced around three times. This reduction of complexity and the concomitant reduction of latency becomes a key factor for commercial applications.

The rest of the paper is organized as follows. Section II introduces the LDPC code, the decoding algorithms, and the performance evaluation techniques. The new long LDPC is designed and investigated in Section III. A new postprocessing technique to reduce low BER floor is introduced in Section IV, while conclusions are drawn in Section V.

II. Background

A. LDPC Codes

An LDPC code is a linear block code defined by a sparse $(m \times n)$ parity check matrix $\mathbf{H}$, where n represents the number of bits in the block and m denotes the number of parity checks [11]. Matrix $\mathbf{H}$ can be graphically represented using a Tanner graph (TG) [12]. TG is composed of two types of nodes: the variable v_i and the check c_i nodes. A connection between nodes v_i and c_j exists if $\mathbf{H}_{j,i} = 1$. For a given variable or check node, the number of connections determines its degree. If all v_i nodes have the same degree γ and all c_i nodes the same degree ρ, then the code is said to be a (γ, ρ)-regular LDPC.

An important family of LDPC codes are quasi-cyclic (QC) LDPC codes [11]. In these codes, $\mathbf{H}$ is an array of sparse square cyclic matrices of the same size (see [13] for more details). QC-LDPC codes can perform very close to the Shannon limit [14], [11]. Moreover, their cyclic properties reduce the implementation complexity [15], [16] and allow the use of efficient algebraic techniques to compute code parameters and optimize the performance [12], [17], [11].

B. Iterative Decoding Algorithms

Typically, LDPC codes are iteratively decoded using the sum-product algorithm (SPA) [18]. Let b_i and x_i be the i-th coded bit and the corresponding channel output, respectively. The input to the SPA decoder is the prior log-likelihood ratio L_i^a defined by

$$L_i^a = ln\left(\frac{P_r\{b_i = 0|x_i\}}{P_r\{b_i = 1|x_i\}}\right). \quad (1)$$

Then, an iterative decoding procedure between variable and check nodes is carried out as follows:

$$L^e_{v_i \to c_j} = L_i^a + \sum_{c_k \in C^{(v_i)} \setminus c_j} L^e_{c_k \to v_i}, \quad (2)$$

$$L^e_{c_j \to v_i} = \phi^{-1}\left\{\sum_{v_k \in V^{(c_j)} \setminus v_i} \phi\left[L^e_{v_k \to c_j}\right]\right\}, \quad (3)$$

where $C^{(v_i)} = \{c_j : \mathbf{H}_{j,i} \neq 0\}$, $V^{(c_j)} = \{v_i : \mathbf{H}_{j,i} \neq 0\}$, $\phi(x) = \ln[\tanh(x/2)]$, and $\phi^{-1}(x) = 2\tanh^{-1}(e^x)$. The posterior LLR is computed in each iteration by

$$L_i^o = L_i^a + \sum_{c_k \in C^{(v_i)}} L^e_{c_k \to v_i}. \quad (4)$$

Hard decisions are derived from (4). The iterative decoding process is carried out until hard decisions satisfy all the parity check equations or when an upper limit on the iteration number is reached.

Computation of (2), (3), and (4) are performed by independent blocks called variable-node processing-unit (VNPU), check-node processing-unit (CNPU), and *a-posteriori* processing unit (APU), respectively. Since the CNPU consumes most of the computational requirements of SPA, a simplified expression of (3) is usually implemented:

$$\hat{L}^e_{c_j \to v_i} = \min_{v_k \in V^{(c_j)} \setminus v_i} \left|L^e_{v_k \to c_j}\right| \cdot \prod_{v_k \in V^{(c_j)} \setminus v_i} sign(L^e_{v_k \to c_j}). \quad (5)$$

This approach is known as the min-sum algorithm (MSA) [11]. To reduce the approximation error of (5), some modifications to MSA have been proposed in [19]. In this paper we use the scaled min-sum algorithm (SMSA) [19]. The check node computation in SMSA is given by

$$\hat{L}^e_{c_j \to v_i} = \alpha \cdot \min_{v_k \in V^{(c_j)} \setminus v_i} \left|L^e_{v_k \to c_j}\right| \cdot \prod_{v_k \in V^{(c_j)} \setminus v_i} sign(L^e_{v_k \to c_j}) \quad (6)$$

with α being a factor smaller than unity (typically $\alpha \approx 0.75$).

C. Performance Evaluation of Long LDPC at Very Low BER

Bit error rates as low as 10^{-15} are required in OTN applications. Unfortunately, no analytical tool is available to evaluate the performance of LDPC codes. Furthermore, traditional *Monte Carlo* simulation cannot be performed at very low BER regimes due to prohibitive simulation times. To estimate the error floor of LDPC in magnetic recording, a combination of analytical tools and FPGA simulation has been proposed in [9]. A similar approach is adopted in this paper. Specifically, FPGA

[Copia del paper presentado en GlobeCom-2011, pg.3/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

simulations in the proximity of the low BER region of interest (e.g., $\lesssim 10^{-13}$) are used to obtain the dominant *trapping sets* (see [9] for more details). Based on these trapping sets, BER is estimated by using importance sampling technique [20]. The excellent accuracy of this method in our application shall be shown in the next section.

III. Design of Long LDPC Codes for Optical Transport Network

Because of the large codeword length required to achieve high NCG, and as a consequence of the complex interconnection patterns inherent in these codes, efficient VLSI implementation is another major challenge. Towards this end, we use an implementation oriented parity check matrix constraint called *regular column-partition* (RCP) [3][21]. This technique is combined with the quasi-cyclic constraint to generate RCP QC-LDPC codes.

A. QC-LDPC Codes with RCP

Let $\mathbf{H}$ be the $(m \times n)$ parity check matrix of an LDPC code. Assuming that $n = \mu q$ with μ and q integers, the matrix $\mathbf{H}$ can be partitioned into μ $(m \times q)$ sub-matrices:

$$\mathbf{H} = \left[\mathbf{H}^{(1)} \cdots \mathbf{H}^{(l)} \cdots \mathbf{H}^{(\mu)}\right]. \quad (7)$$

The code is designed so that the weights of the rows and columns of $\mathbf{H}^{(l)}$ do not change with l. This technique is called here RCP. Fig. 1 shows an example where a $(4, 12)$-regular matrix $\mathbf{H}$ is partitioned into $\mu = 6$ $(4, 2)$-regular sub-matrices $\mathbf{H}^{(l)}$ (i.e., $q = 2$).

The RCP constraint allows an efficient partial parallel implementation of the MSA [3]. Each iteration is divided into μ steps. At the l-th step, only the messages related to $\mathbf{H}^{(l)}$ are computed. This allows to reuse the same q VNPUs at each step, reducing μ times the associated hardware. The interconnection complexity is also reduced because the interconnection network is associated with the non-zeros entries of $\mathbf{H}^{(l)}$, which is μ times smaller than that of the original $\mathbf{H}$. The CNPU blocks are simplified since the recursive computation of (5) has significantly lower complexity than a full-parallel implementation. Furthermore, the recursive CNPU needs to store only two messages [19], therefore it is not necessary to store all L^e messages. The latter reduces the memory requirements of the decoder. The reader is referred to [3] and [21] for a more detailed complexity analysis of RCP.

An RCP-QC-LDPC code is a QC-LDPC code that meets the RCP constraint (e.g., see Fig. 1). We focus here on regular RCP-QC-LDPC codes based on cyclic sub-matrices with the same rows weight p (i.e. the number of nonzero diagonals [11]). We denote this coding scheme as a *type-p* RCP-QC-LDPC code. For VLSI implementation, a small value of p is preferred since the complexity increases (at least) linearly with p. Notice that a high value of p reduces the maximum girth of the code, which increases the error floor probability.

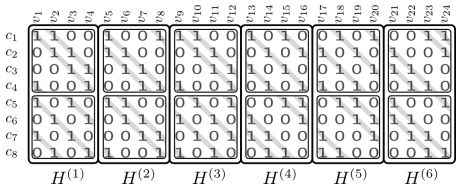


Figure 1. Example of the RCP partition of a $(4, 12)$-regular parity check matrix into $\mu = 6$ $(4, 2)$-regular sub-matrixes

B. Code Parameters

We design a 20% OH LDPC-only coding scheme based on a type-2, column-weight 4, RCP-QC-LDPC code[1]. The code is composed by a 2×12 array of bi-cyclic sub-matrices. The maximum girth of the code is 8. From computer simulations, we found that the size of the bi-cyclic sub-matrices must be 2048 to achieve the expected NCG (i.e., $>$ 10dB). Thus, the code parameters result $n = 12 \cdot 2048 = 24576$ and $k = n - rank(\mathbf{H}) = 20482$ bits. Two diferent RCP-QC-LDPC codes are generated with these parametes. The first code, $\mathcal{C}_1$, is designed to avoid cycles of length 4. The second one, $\mathcal{C}_2$, is designed to (*i*) avoid cycles of length 6 (i.e. achieve the maximum girth), and (*ii*) minimize the number of cycles of length 8. We use the technique proposed in [17] to speed up the cycle optimization process of the RCP-QC-LDPC codes.

C. Performance Evaluation

An SMSA decoder for the proposed (24576,20482) RCP-QC-LDPC codes has been designed using the Xilinix Virtex-V FPGA. Let r_a and r_e be the number of bits used to represent the prior LLRs and the messages from both check and variables nodes, respectively. The prior LLRs are quantized using $r_a = 5$ bits. The decoder is implemented using $r_e = 5$ bits. In all experiments, an all-zeros codeword is transmitted. We use binary phase-shift keying (BPSK) modulation (i.e., bits $\{0, 1\}$ are mapped into symbols $\{+1, -1\}$ for transmission, respectively). An additive white Gaussian noise (AWGN) channel is implemented by using a random number generator as described in [22], which is adequate for the application considered here. In all cases, 400 errors are counted for BER computation.

Figure 2 depicts the BER versus the signal-to-noise ratio (SNR) for code $\mathcal{C}_1$ with 13 iterations. We present results from FPGA simulations and theoretical estimates derived from the method described in Section II-C. The plot shows an error floor at BER= 10^{-11} with an NCG of 9.45dB at BER=10^{-15}. Notice the excellent agreement between simulation and theory. Analysis of experimental data obtained from the FPGA simulator showed that this error floor is caused by the presence of several *absorbing sets* (AS) [4] created by the combination of cycles of length 6.

Figure 3 shows the performance of $\mathcal{C}_2$. No error floor is observed up to 10^{-13}. The measured NCG at this BER is 10.70dB. However, from importance sampling analysis it has

[1]Numerical results not included here have shown that a type-2 performs similarly to an equivalent type-1 RCP-QC-LDPC.

[Copia del paper presentado en GlobeCom-2011, pg.4/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

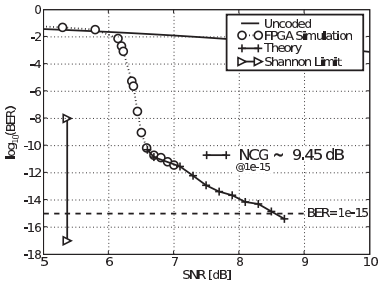


Figure 2. BER vs. SNR of code $\mathcal{C}_1$.

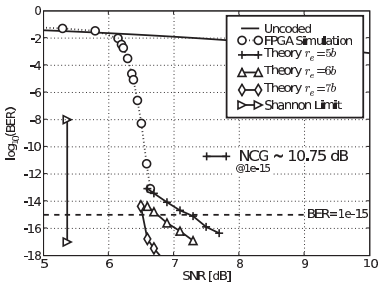


Figure 3. BER vs. SNR of code $\mathcal{C}_2$.

been estimated a quantization sensitive error floor below 10^{-13}. This error floor is caused by the combination of a (12,8) AS[2] and the fixed-point representation used for the extrinsic messages L^e. Fig. 3 shows the estimated error floor for $r_e = 5$, 6, and 7 bits with 13 iterations. The *a-posteriori* LLR evolution of the (12,8) AS is shown in Fig 4. Note that the SMSA decoder with $r_e = 5$ bits cannot resolve the (12,8) AS, independently of the number of iterations. On the other hand, the SMSA decoder requires 17 and 12 iterations with $r_e = 6$ and 7 bits, respectively.

IV. Min-Sum Algorithm with Adaptive Quantization

Postprocessing algorithms (PPAs) have been proposed to reduce error floor of LDPC codes [4] [5] [6]. Numerical results showed that PPA can reduce error floor around 2-3 orders of magnitude. Thus, PPA could be combined with code $\mathcal{C}_2$ to achieve an NCG$>$ 11dB at BER=10^{-15} with free error floor (see Fig. 3). Although several PPA could be considered for our application, in this work we propose a new low complexity technique designed to combat error floor caused by the fixed-point implementation of the MSA.

A. Adaptive Quantization (AQ) Algorithm

We present a new low complexity postprocessing scheme to combat the error floor caused by the fixed-point implementation

[2]In the notation "(e, d) AS", e is the number of wrong bits and d is the number of unsatisfied check nodes (see [4] for more details).

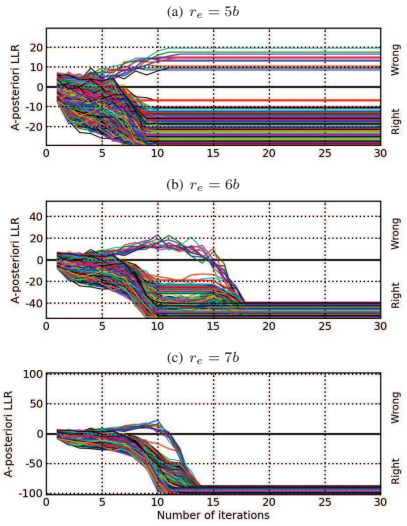


Figure 4. *A-posteriori* LLR evolution of code $\mathcal{C}_2$ for the (12,8) AS.

of the MSA. In this approach, the LLRs and messages are scaled at every iteration in order to increase the range of representation, and thus, reduce the saturation effects generated in the MSA operations. Since the total number of bits is maintained constant, this wider range is obtained at expense of an increase of the quantization. This adaptive quantization (AQ) process is started when the number of unsatisfied check nodes is smaller than a given threshold d_t (e.g., $d_t = 9$ for a (12,8) AS), which occurs after some *normal* iterations (i.e., without AQ). The proposed postprocessing algorithm can be efficiently implemented in the VNPU as follows:

$$L^e_{v_i \to c_j} = (\kappa_1)^t \cdot L^a_i + \kappa_2 \cdot \left(\sum_{c_k \in C^{(v_i)} \setminus c_j} L^e_{c_k \to v_i} \right), \quad (8)$$

where $t = 1, 2, ...,$ denotes the number of the *extra* iteration. Factors κ_1 and κ_2 are positive gains smaller than unity. To simplify implementation, $\kappa_1 = \kappa_2 = \kappa$ can be used. Then, the AQ algorithm reduces to scale by κ (*i*) the output of the variable-node equation (2), and (*ii*) the prior LLR, that is,

$$L^e_{v_i \to c_j} = \kappa \cdot \left(L^a_i + \sum_{c_k \in C^{(v_i)} \setminus c_j} L^e_{c_k \to v_i} \right), \quad (9)$$

$$L^a_i \leftarrow \kappa \cdot L^a_i. \quad (10)$$

As the AQ-based postprocessing evolves, from (9) and (10) note that the contribution of the prior information to the variable node equation output is gradually reduced. Notice also that after a given number of iterations, the MSA operates without prior information. We found that $\kappa = \frac{1}{2}$ provides a good tradeoff between performance and implementation complexity. In this case, our approach reduces to that proposed in [23].

[Copia del paper presentado en GlobeCom-2011, pg.5/5]

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

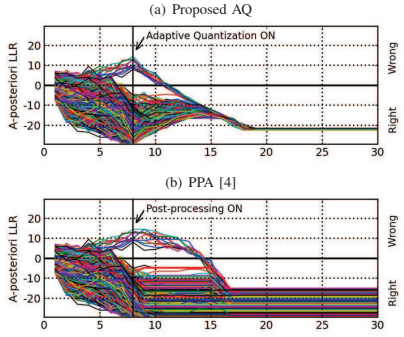


Figure 5. *A-posteriori* LLR evolution of code $\mathcal{C}_2$ for the (12,8) AS and $r_e = 5$ with (a) our AQ scheme and (b) the PPA proposed in [4].

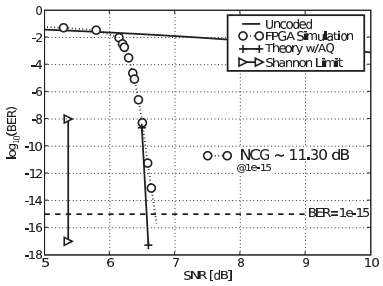


Figure 6. Performance of code $\mathcal{C}_2$ with AQ and r_e =5 bits.

B. Numerical Results and Discussion

The AQ algorithm and the PPA reported in [4] are analyzed in Fig. 5. We show the *a-posteriori* LLR evolution of the SMSA decoder with r_e =5 bits over the (12,8) AS. Numerical results show that the AS can be resolved with 5 extra iterations when AQ is used. On the other hand, we observe that the PPA introduced in [4] can resolve the (12,8) AS with 9 extra iterations. Since a low complexity is crucial for practical implementation of long LDPC codes, it is concluded that AQ becomes a promising postprocessing alternative for the application considered in this work.

Figure 6 shows the estimated BER with the AQ algorithm and r_e =5 bits. We note that the error floor observed in Fig. 3 is successfully corrected, while the expected NCG results 11.30dB at BER=10^{-15}.

V. Conclusions

In this paper we have presented a non-concatenated code suitable for applications in 100Gb/s OTN. The design was based on a new class of long LDPC codes designed to jointly minimize BER floors and implementation complexity. We also have introduced a postprocessing technique to reduce the already low BER floors to a level well below the specifications. Importance sampling analysis has shown an NCG $\geq$ 11.30dB at a BER of 10^{-15}.

References

[1] K. Onohara *et. al.*, "Soft-decision-based forward error correction for 100 Gb/s transport systems," *IEEE J. Sel. Topics Quantum Electron.*, vol. 16, no. 5, pp. 1258 –1267, Sept.-Oct. 2010.

[2] N. Kamiya and S. Shioiri, "Concatenated QC-LDPC and SPC codes for 100 Gbps ultra long-haul optical transmission systems," March 2010, pp. 1–3.

[3] D. Morero, G. Corral-Briones, and M. Hueda, "Parallel architecture for decoding LDPC codes on high speed communication systems," in *EAMTA 2008 (available in IEEE Xplorer)*, Sept. 2008, pp. 107 –110.

[4] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. Wainwright, "Lowering LDPC error floors by postprocessing," in *IEEE Globecom*, Dec. 2008, pp. 1 –6.

[5] Y. Han and W. Ryan, "Low-floor decoders for LDPC codes," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1663–1673, June 2009.

[6] Y. Zhang and W. Ryan, "Toward low LDPC-code floors: a case study," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1566–1573, June 2009.

[7] Z. Zhang, L. Dolecek, M. Wainwright, V. Anantharam, and B. Nikolic, "Quantization effects in low-density parity-check decoders," June 2007, pp. 6231 –6237.

[8] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. Wainwright, "Design of LDPC decoders for improved low error rate performance: quantization and algorithm choices," *IEEE Trans. Commun.*, vol. 57, no. 11, pp. 3258 –3268, Nov. 2009.

[9] X. Hu, Z. Li, B. Vijaya Kumar, and R. Barndt, "Error floor estimation of long LDPC codes on magnetic recording channels," *IEEE Trans. Magn.*, vol. 46, no. 6, pp. 1836 –1839, June 2010.

[10] D. Chang, F. Yu, Z. Xiao, Y. Li, N. Stojanovic, C. Xie, X. Shi, X. Xu, and Q. Xiong, "FPGA verification of a single QC-LDPC code for 100 Gb/s optical systems without error floor down to BER of 10^{-15}," in *OFC*, 2011.

[11] W. Ryan and S. Lin, "Channel codes: Classical and modern," *Cambridge University Press*, 2009.

[12] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533 – 547, Sep. 1981.

[13] S. Lin and D. Costello, "Error control coding, fundamental and applications," *Pearson Prentice Hall, Second Edition*, 2004.

[14] Y. Kou, S. Lin, and M. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Trans. Inf. Theory*, vol. 47, no. 7, pp. 2711 –2736, Nov. 2001.

[15] Z. Li, L. Chen, L. Zeng, S. Lin, and W. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Trans. Commun.*, vol. 53, no. 11, p. 1973, Nov. 2005.

[16] Z. Wang and Z. Cui, "Low-complexity high-speed decoder design for quasi-cyclic LDPC codes," *IEEE Trans. VLSI Syst.*, vol. 15, no. 1, pp. 104 –114, Jan. 2007.

[17] X. Wu, X. You, and C. Zhao, "An efficient girth-locating algorithm for quasi-cyclic LDPC codes," in *Inform. Theory, International Symposium on*, July 2006, pp. 817 –820.

[18] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498 –519, Feb. 2001.

[19] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288 – 1299, Aug. 2005.

[20] D. B. Cavus E., Haymes C.L., "Low BER performance estimation of LDPC codes via application of importance sampling to trapping sets," *IEEE Trans. Commun.*, vol. 57, no. 7, pp. 1886 –1888, July 2009.

[21] L. Liu and C.-J. Shi, "Sliced message passing: High throughput overlapped decoding of high-rate low-density parity-check codes," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 11, pp. 3697 –3710, Dec. 2008.

[22] D.-U. Lee, R. C. Cheung, J. D. Villasenor, and W. Luk, "Inversion-based hardware gaussian random number generator: A case study of function evaluation via hierarchical segmentation," in *FPT 2006. IEEE International Conference on*, Dec. 2006, pp. 33 –40.

[23] C.-H. Chung, Y.-L. Ueng, M.-C. Lu, and M.-C. Lin, "Adaptive quantization for low-density-parity-check decoders," in *Information Theory and its Applications (ISITA), 2010 International Symposium on*, Oct. 2010, pp. 13 –18.

FPGA Implementation of the Parity Check Node for Min-Sum LDPC Decoders

Fernando Gutierrez, Graciela Corral-Briones
and Damián Morero
Digital Communication Research Laboratory
National University of Córdoba
Av. Vélez Sarsfield 1611 - Córdoba, Argentina
Email: {fgutierrez, gcorral, dmorero}@efn.uncor.edu

Teodoro Goette and Facundo Ramos
ClariPhy Argentina S.A.
Humberto Primo 680
Córdoba (5000) - Argentina
Email: {teodoro.goette, facundo.ramos}@clariphy.com.ar

***Abstract*—A typical high-speed decoder implementation for an LDPC may require hundreds or even thousands of variable and check node processors. Since check node processing unit (CNPU) is far more complex than variable processing unit, hardware requirements of CNPU has a big impact on the final decoder complexity. Here, an FPGA implementation of the soft parity check node for Min-Sum LDPC Decoders is analyzed. The hardware cost and speed of the main block of CNPU, which finds the two smallest input values, is thoroughly studied for different numbers of input values with different bit-widths. Experiments for an FPGA implementation demonstrate that hardware cost and speed vary with the number of input values in the same way as they do for an ASIC implementation. Furthermore, it is shown that more than 60% of the hardware resources of the CNPU is used for finding the two smallest input values.**

I. Introduction

Low Density Parity Check (LDPC) codes [1] have become one of the best option for the applications that require to achieve near Shannon capacity in the communication channel. LDPC codes have been adopted for 10GBase-T [2], DVB-S2 [3], and Mobile WiMax (802.16e) [4]. The decodification process is based on a Sum-Product (SP) algorithm composed of two main processing units (PU). These PUs are the check-node PU (CNPU) and the variable-node PU (VNPU) [5] [6]. A typical high-speed decoder implementation may require hundreds or even thousands of VNPUs and CNPUs. Therefore, any optimization of this PUs has a big impact on the final decoder complexity.

The SP which is based on soft-decision decoding achieves the best decoding performance but has very high complexity [7], particularly at the CNPU. Many modifications have been proposed to simplify the node computations in SP. Among them, the min-sum (MS) algorithm has significantly simplified the CNPU and it is more suited for practical implementation [8]. That algorithm reduces memory usage by computing only the two smallest values and the index of the first minimum value. In [9] an algorithm based on a tree search (TS) approach is proposed for finding the two smallest values. That paper shows that TS approach achieves better performance in both area and speed than sorting based algorithm that requires less number of comparisons. Also a theoretical complexity analysis is presented in that paper and numerical results for a cell library in a 0.18 μm digital CMOS process is shown.

This paper has been supported in part by the ANPCyT (PICT2008-1256).

In [10] a reduced complexity LDPC decoder is designed and implemented on FPGA. The overall hardware resource utilization is shown without specifying the hardware cost required for implementing the VNPU and CNPU.

In this paper, we analyze the speed performance and hardware cost for an FPGA implementation of the soft check node processing units based on the TS approach algorithm for finding the two smallest values from a set of n values with i bit-width. Experimental results show that the hardware cost tends to increase linearly with n, while the hardware speed tends to decrease linearly with $log(n)$, in agreement with the analytical results reported in [9] for the area and speed performance of an ASIC implementation. The increment (decrement) slope in hardware resources (hardware speed) with bit-width is more pronounced for higher n.

The algorithm and a detailed digital architecture implementation is presented in Section II. The FPGA implementation is explained in Section III, and experimental results are discussed in Section IV. Finally, conclusions are drawn in Section V.

II. Algorithm and Digital Architecture for the TS Approach

The LDPC codes can be efficiently decoded with Belief Propagation (BP) algorithm using an iterative message-passing method. The general BP algorithm consists of two phases of message passing from variable nodes to check nodes and vice versa. Denoting the messages from variable nodes to check nodes with $l_{i\to a}$ and the messages from the check nodes to variable nodes by $r_{a\to i}$, in the log-likelihood ratio (LLR) domain version of BP algorithm, the following computations are performed at the check nodes:

$$S_{a\to i} = \prod_{j\in\partial a\setminus i} \text{sgn}(l_{j\to a}) \tag{1}$$

978-1-4673-0186-2/12/$31.00 ©2012 IEEE

[Copia del paper publicado en SPL-2012, pg.2/6]

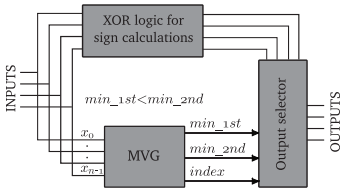


Fig. 1. Check node architecture

$$r_{a\to i} = S_{a\to i} \cdot 2\tanh^{-1}\Big(\prod_{j\in\partial a\setminus i} \tanh\Big(\frac{|l_{j\to a}|}{2}\Big)\Big) \quad (2)$$

Although LDPC codes can be decoded very well with a class of BP algorithms, the MS decoding algorithm greatly reduces the decoding complexity. The MS algorithm simply replaces the check node calculation (2) with its approximation:

$$r^{MS}_{a\to i} = S_{a\to i} \cdot \min_{j\in\partial a\setminus i}(|l_{j\to a}|) \quad (3)$$

The tradeoff in this simplification is a significant degradation in performance. It can easily be shown that the magnitude of the messages calculated by (3) are larger than the magnitude of the messages calculated by (2).

Since check node processing is fundamentally more complicated, in this work we focus on implementing an check node performing the MS algorithm developed in [11]. In the literature, there have been many designs and ideas for check node in MS form, and some of them are developed in [12] [13]. A block diagram of a check node implemented is shown in Fig. 1. This design basically finds the two smallest inputs and outputs them to the appropriate output (Note that the magnitudes on the check node outputs take on only two values). From Fig. 1, it can be seen that the check node has two separate parts, one calculating signs and the other calculating magnitudes. Since sign calculation is a one-bit operation and is simple, the only way to reduce the complexity of a check node is to reduce complexity of magnitude calculations.

This section presents a high-speed low-complexity design of the Minimum Value Generator (MVG) block which is the main building block of the check node, see Fig. 1. The MVG block computes the first and second minimum values, min_1st and min_2nd, and the $index$ of the first minimum value in the n input values $x_0, \cdots, x_{n-1}$. The MVG block implements a TS algorithm for an $n = 2^k$ and $n = 2^k + 2^r$ number of inputs, n-MVG. The algorithm and its digital architecture are discussed in this section.

A. Algorithm

Consider the basic 2-MVG building block shown in Fig. 2(a), in which the input values x_0 and x_1 are compared, and the minimum value min_1st, its index $index$ (i.e., 0 or 1), and the other value defined as min_2nd are determined. The

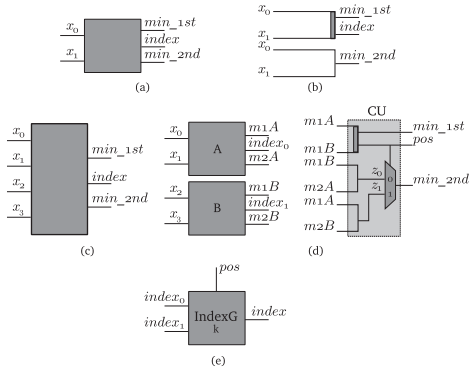


Fig. 2. TS Approach. (a) and (b) 2-MVG. (c) and (d) 4-MVG. (e) Index Generator

symbols shown in 2(b) are used to represent two basic operations: one of them is the operation of getting the minimum of two inputs and its corresponded index, and the other one the operation of getting only the minimum. Fig. 2(c) is a 4-MVG building block constructed using two 2-MVG blocks, A and B, and one Connection Unit (CU), as it is shown in Fig. 2(d). Let the inputs of building block A be x_0 and x_1 and the min_1st, $index$ and min_2nd be the outputs $m1A$, $index_0$, and $m2A$, respectively. Similarly, the inputs of building block B are x_2 and x_3 and its outputs $m1B$, $index_1$ and $m2B$. Obviously, the minimum value of $m1A$ and $m1B$ is the minimum value of the four inputs, i.e., min_1st. However, the second minimum min_2nd can be either the min$\{m1B, m2A\}$ or the min$\{m1A, m2B\}$, depending on whether $m1A$ or $m1B$ is the smaller. In either case, we do not need to compare $m2A$ with $m2B$.

Similarly, a building block of 8-MVG is constructed using two 4-MVG building blocks with a CU, or four 2-MVG blocks with three CUs. In general, a 2^k-MVG is constructed by two 2^{k-1}-MVGs with a CU, in a TS fashion way; or by 2^{k-1} 2-MVGs with $2^{k-1}-1$ CUs. Therefore, a 2^k-MVG requires 2^{k-1} comparisons for the 2-MVG blocks and $3 \cdot 2^{k-1}$ comparisons for the CUs, and each CU requires three comparisons.

The index of min_1st for a 4-MVG is computed at the block called IndexG, which is shown in Fig. 2(e). Its inputs, $index_0$ and $index_1$, originated from two 2-MVG building blocks, A and B, take values from a set $\{0, \ldots, 2^{k-1} - 1\}$. Depending on the control signal $pos \in \{0, 1\}$, the input from block A or B is selected and the output index is build as

$$index = pos \cdot 2^{k-1} + index_{pos}, \quad (4)$$

though, output $index \in \{0, \ldots, 2^k - 1\}$. The control signal pos comes from a CU and indicates which block, A or B, has the minimum value, see Fig. 2.

[Copia del paper publicado en SPL-2012, pg.3/6]

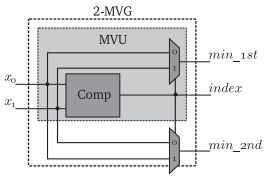


Fig. 3. Building Block of 2-MVG

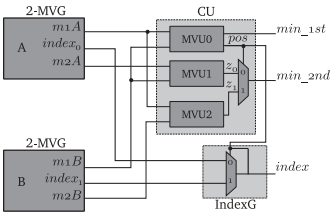


Fig. 4. A 4-MVG Using the TS approach

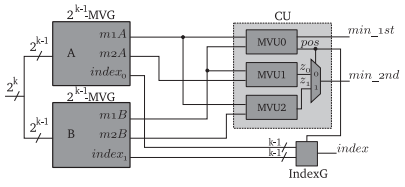


Fig. 5. 2^k-MVG Using the TS approach

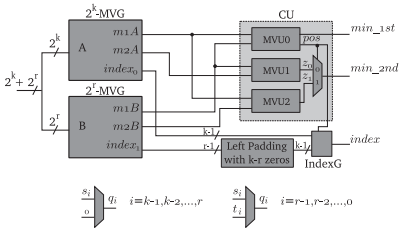


Fig. 6. $2^k + 2^r$-MVG Using the TS approach

*B. Digital Architecture for n-*MVG with n=2^k

The basic 2-MVG building block in Fig. 2 can be implemented by a comparator and two 2-input i-bit multiplexers (MUXs). The comparator output $index$ selects the input signals of the multiplexers, as shown in Fig. 3, where

$$\begin{aligned} &index = 0;\ min_1st = x_0;\ min_2nd = x_1, && \text{if } x_0 \le x_1 \\ &index = 1;\ min_1st = x_1;\ min_2nd = x_0, && \text{if } x_1 < x_0 \end{aligned}$$

In Fig. 3, we define a basic block, called Minimum Value Unit (MVU), that consists of a comparator and a multiplexer.

Fig. 4 shows a 4-MVG containing two 2-MVGs and a CU. Each 2-MVG includes one comparator and two 2-input i-bit MUXs, while CU has three MVUs. The IndexG block requires a 2-input 1-bit MUX. The 2-bit output $index$ is built with pos as the most significant bit and MUX output. In total, the 4-MVG is implemented by five comparators, eight 2-input i-bit MUXs, and one 2-input 1-bit MUX. Both $m1A$ and $m1B$ determine min_1st, while min_2nd is determined by z_0 and z_1. If $pos = 0$, i.e., $min_1st = m1A$, then $min_2nd = z_0$. On the other hand, if $pos = 1$, i.e., $min_1st = m1B$, then $min_2nd = z_1$.

The digital architecture of a 2^k-MVG can be constructed using the TS approach as shown in Fig. 5, where A and B are two 2^{k-1}-MVGs with minimum values $m1A, m2A, m1B, m2B$, and indexes $index_0$ and $index_1$. Let min_1st, z_0, and z_1 be the outputs of MVU0, MVU1, and MVU2, respectively, given by

$$\begin{aligned} min_1st &= min\{m1A, m1B\} \\ min_2nd &= MUX(z_0, z_1; pos) \\ z_0 &= min\{m1B, m2A\} \\ z_1 &= min\{m1A, m2B\} \end{aligned}$$

Note that $index_0$ and $index_1$ are represented by $k-1$ bits, and the output of IndexG is built concatenating pos as the most significant bit with the $k-1$ bits selected by pos from the inputs $index_0$ and $index_1$.

*C. Digital Architecture for n-*MVG with n=$2^k + 2^r$ for $k > r$

The design for n-MVG with any positive integer $n = 2^k + 2^r$ is shown in Fig. 6, where two n-MVG blocks with $n = 2^k$ and $n = 2^r$ inputs are used. For computing the index of the first minimum, $index_1$ is left-padded with $k - r$ zeros, before entering the IndexG block. In conclusion, from Fig. 6 the outputs of the $2^k + 2^r$-MVG are

1) $min_1st = \min\{m1A, m1B\}$.
2) $min_2nd = \text{MUX}(z_0, z_1; pos)$, where $z_0 = \min\{m2A, m1B\}$, $z_1 = \min\{m1A, m2B\}$, and pos is the comparison signal of mVU0.
3) $q_k = pos$; for $i = k-1, k-2, \ldots, r$, $q_i = \text{MUX}(s_i, 0; pos)$; for $i = r-1, r-2, \ldots, 0$, $q_i = \text{MUX}(s_i, t_i, pos)$.

III. FPGA Implementation

In this section we presents a brief description of the FPGA architecture used for the n-MVG implementation and the design methodology followed which starts with an executable specification written in SystemC. These information is important to analyze the results presented in Section IV.

A. Stratix II GX Architecture

The general architecture of the EP2SGX90 device consists of 71 Logic Array Block (LAB) columns and 68 LAB rows. Each LAB consists of eight adaptive logic modules (ALMs), carry chains, shared arithmetic chains, LAB control signals,

[Copia del paper publicado en SPL-2012, pg.4/6]

local interconnects, and register chain connection lines. The basic building block of logic in the Stratix II GX architecture is the ALM. Each ALM can be divided between two Adaptive Look Up Tables (ALUTs). With up to eight inputs to the two ALUTs, one ALM can implement various combinations of two functions. In addition to the adaptive LUT-based resources, each ALM contains two programmable registers, two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain.

The Stratix II GX ALM can operate in Normal mode, Extended LUT mode, Arithmetic mode, Shared arithmetic mode and, therefore, each mode uses ALM resources differently. The ALMs are routed with the MultiTrack interconnect architecture, enabling a Stratix series FPGA to implement high-speed logic, arithmetic, and register functions. The MultiTrack interconnect consists of continuous, performance-optimized routing lines of different lengths and speeds used for inter- and intra-design block connectivity. For more details see [14].

B. Methodology

The algorithm was implemented in System Description Language SystemC, and tested for various sets of parameters. The SystemC Class Library has been developed to support system level design, and also provides a methodology for describing algorithms and hardware architecture. The design methodology was to start with a high-level model written in C++ and apply an iterative process of transforming the code to use only elements that have an equivalent in Hardware Description Language (HDL).

Then, the algorithm description was coded in Verilog and tested using the ModelSim Altera Starter Edition 6.4a tool. Later on, the FPGA design was synthesized using Quartus II version 9.0 into an Stratix II GX FPGA (EP2SGX90FF1508C3) from Altera. In order to illustrate the modularity of the architecture, we have tested the design with different sets of parameters. Simulation and Synthesis was performed with several number of inputs (from 4 to 64) and input widths (from 4 to 12 bits).

The architecture has a combinational design which required the addition of input and output registers at the top-level module for the appropriate timing analysis. To perform timing analysis we used the TimeQuest timing analyzer tool, and to create timing constraints we used the TimeQuest graphical user interface tools. Finally, we add a random-number generator that produces n items of data with a specified length.

IV. Analysis of the Results

This section presents the analysis of the results in terms of speed performance and hardware cost. All circuits have been physically implemented, tested, and the results have been extracted after place and route. A brief analysis for 6 bit-width will be explained later.

A. Speed performance

Fig. 7 shows the critical path delay vs the number of inputs (n) with different bit-widths. A logarithmic scale is used in the horizontal axis to show that critical path delay tends to increase linear with the $\log n$, in the same fashion as the TS architecture implemented by multiplexer and comparator [9]. From Fig. 5 and 6, it seems that the increase of bit-width increases the size of multiplexers and comparators, and therefore the delay. Fig. 8 shows the relationship between delay and bit-width for the FPGA implementation. Note that the critical path delay increases slower with bit-width than with the number of inputs, which is consistent since the more number of inputs required more TS stage. Note also that the delay increment slope with bit-width, in Fig. 8, is higher for a larger number of inputs.

B. Hardware cost

First we analyze the area requirements in ALUTs of several MVGs using the TS approach. Fig. 9 shows the number of ALUTs vs. number of inputs. We can see that increasing the number of inputs basically doubles the required area. The number of basic building block of logic shown in the figure, involves all sizes of ALUTs as possible. Similarly, Fig. 10 shows the relationship between number of ALUTs and input width. Increasing the number of bits, i.e. from 4 to 8, the hardware cost is doubled and if we consider the boundary points, i.e. from 4 to 12, the hardware cost is tripled. Note also that as bits increases, the slope increases for higher numbers of inputs.

Since the final implementation area depends on the total number of ALMs used in the device, not on the number of ALUTs generated, it is very important to generate a distribution of LUTs that is amenable to efficient packing [15]. Table I summarizes the results of experiments related to the number of ALMs partially or completely used by the MVG block. We used balanced technology mapping to maintain optimal critical path depth while producing a more packable LUT distribution.

Fig. 11 shows the distribution of LUT sizes for 6 bits in percentage of total ALUTs. The graph clearly illustrates that Stratix II GX implementation of the design approximately uses the same LUT distribution for all number of inputs. On the other hand, Fig. 12 shows the percentage of delay of critical path used for logic, wiring, and register. As it is expected, the percentage of delay used for registers are insignificant. For all input numbers, the percentage of delay used for logic and wiring remains almost the same. Because there are no abrupt changes in the distribution of LUTs and the percentage of delay for logic/wiring, shown in Figs 11 and 12, we can conclude that hardware resources and critical path delay in the FPGA implementation of the TS based algorithm depend only on the number of inputs (for a given bit-width).

Finally, the check node architecture shown in Fig 1 was implemented with 5 bit-width and the number of ALMs used by block is shown in Table II. For table entries, the numbers listed indicate the total number of ALMs partially or completely used by the specific block and all of its sub-blocks in the hierarchy. The number of ALMs used by the MVG block with 5 bit-width, is not the same in Table I and Table II. The reason for this difference is because to calculate the values of Table I, we have considered directly the absolute value of the

[Copia del paper publicado en SPL-2012, pg.5/6]

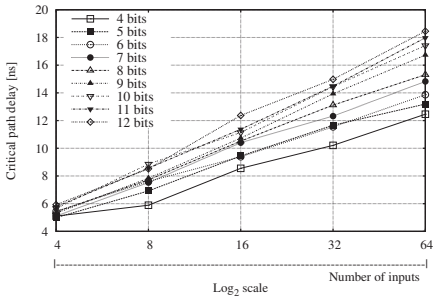


Fig. 7. Critical path delay vs. Number of inputs

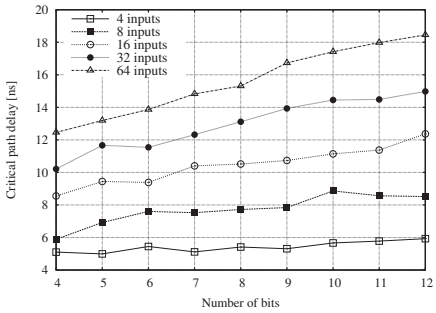


Fig. 8. Critical path delay vs. Number of bits

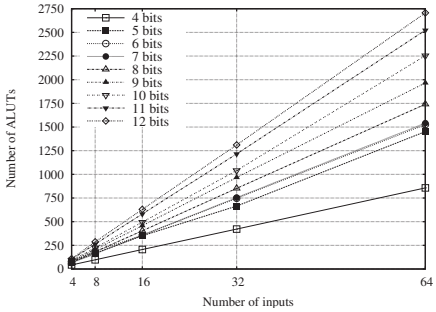


Fig. 9. Number of ALUTs vs. Number of inputs

inputs. On the other hand, to calculate the values of Table II, an additional processing on the MVG block is performed to obtain the absolute value and the two's complement of inputs. Then, the minimum values and the index are calculates. Table II shows clearly that the complexity of the check node is

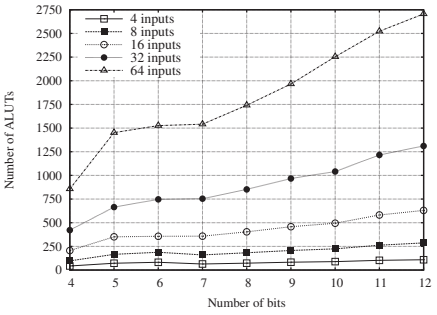


Fig. 10. Number of ALUTs vs. Number of bits

TABLE I
NUMBER OF ALMS USED BY THE MVG BLOCK

#bits	*#inputs*				
	n=4	*n=8*	*n=16*	*n=32*	*n=64*
4	30	65	135	275	557
5	46	103	216	413	886
6	51	108	227	470	955
7	54	118	235	489	995
8	58	124	266	554	1127
9	60	140	301	628	1271
10	65	152	328	681	1312
11	74	176	379	784	1528
12	79	192	411	848	1719

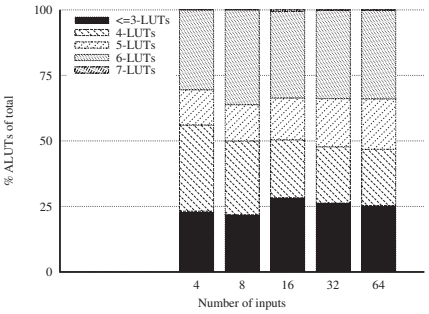


Fig. 11. LUT-size distribution (6 bits)

TABLE II
NUMBER OF ALMS USED BY BLOCK

blocks	*#inputs*				
	n=4	*n=8*	*n=16*	*n=32*	*n=64*
sign calculations	4	10	22	47	91
output selector	24	48	96	301	640
MVG	52	121	262	547	1109
Total	80	179	380	895	1840

dominated by the MVG block when the design is implemented in an FPGA.

[Copia del paper publicado en SPL-2012, pg.6/6]

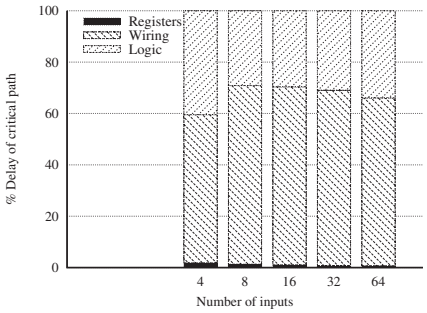


Fig. 12. Routing (6 bits)

V. Conclusion

This paper presented an FPGA implementation of a soft check node processing unit based on the tree structure approach for finding the two smallest values from a set of n input values and the index of the first minimum. The speed and area performance has been analyze for different number of inputs and bit-widths. As the number of inputs increases, the required area tends to increase in a linear way while the speed tends to decrease in a logarithm way. More than 60% of the total hardware cost is used for finding the two smallest input values.

Acknowledgment

The authors would like to thank Carlos Zerbini for very useful comments and suggestions.

References

[1] R. G. Gallager. *Low Density Parity-Check Codes*. MIT Press, Cambridge, MA, 1963.
[2] IEEE. *802.3an Task Force*, June 2004.
[3] European Telecommunication Standards Institute. *EN 302 307 V1.1.1*, June 2004.
[4] IEEE 802.16e. *IEEE standard for local and metropolitan area networks,*, February 2006.
[5] Chien-Ching Lin, Kai-Li Lin, Hsie-Chia Chang, and Chen-Yi Lee. A 3.33gb/s (1200,720) low-density parity check code decoder. In *Solid-State Circuits Conference, 2005. ESSCIRC 2005. Proceedings of the 31st European*, pages 211 – 214, sept. 2005.
[6] Se-Hyeon Kang and In-Cheol Park. Loosely coupled memory-based decoding architecture for low density parity check codes. In *Custom Integrated Circuits Conference, 2005. Proceedings of the IEEE 2005*, pages 703 – 706, sept. 2005.
[7] A. Anastasopoulos. A comparison between the sum-product and the min-sum iterative detection algorithms based on density evolution. In *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, volume 2, pages 1021 –1025 vol.2, 2001.
[8] J. Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and X.-Y. Hu. Reduced-complexity decoding of ldpc codes. *Communications, IEEE Transactions on*, 53(8):1288 – 1299, aug. 2005.
[9] Chin-Long Wey, Ming-Der Shieh, and Shin-Yo Lin. Algorithms of finding the first two minimum values and their hardware implementation. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 55(11):3430 –3437, dec. 2008.
[10] V.A. Chandrasetty and S.M. Aziz. Fpga implementation of high performance ldpc decoder using modified 2-bit min-sum algorithm. In *Computer Research and Development, 2010 Second International Conference on*, pages 881 –885, may 2010.
[11] K.K. Gunnam, G.S. Choi, and M.B. Yeary. A parallel vlsi architecture for layered decoding for array ldpc codes. In *VLSI Design, 2007. Held jointly with 6th International Conference on Embedded Systems., 20th International Conference on*, pages 738 –743, jan. 2007.
[12] A. Darabiha, A.C. Carusone, and F.R. Kschischang. A bit-serial approximate min-sum ldpc decoder and fpga implementation. In *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, page 4 pp., may 2006.
[13] R. Zarubica, S.G. Wilson, and E. Hall. Multi-gbps fpga-based low density parity check (ldpc) decoder design. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 548 –552, nov. 2007.
[14] Altera Corporation. *Stratix II GX Device Handbook*, may. 2007.
[15] Mike Hutton, Jay Schleicher, David Lewis, Bruce Pedersen, Richard Yuan, Sinan Kaptanoglu, Gregg Baeckler, Boris Ratchev, Ketan Padalia, and Mark Bourgeault. Improving FPGA Performance and Area Using an Adaptive Logic Module. In *Proceedings Field-Programmable Logic and Applications*, pages 135–144. Springer, 2004.

[Copia del paper publicado en ArXiv-2012, pg.1/3]

arXiv:1206.4914v1 [cs.IT] 21 Jun 2012

Joint Demapping and Decoding for DQPSK Optical Coherent Receivers

Mario A. Castrillon, Damian A. Morero, and Mario R. Hueda
Digital Communications Research Laboratory - National University of Cordoba - CONICET
Av. Velez Sarsfield 1611 - Cordoba (X5016GCA) - Argentina
acastrillon, dmorero, mhueda@efn.uncor.edu

Abstract: We present a low-complexity joint demapper-decoder scheme for coherent optical receivers with DQPSK modulation. The new technique reduces to 0.7dB the gap between QPSK and DQPSK in 100Gb/s coherent optical systems.

1. Introduction

Coherent detection based receivers with electronic dispersion compensation (EDC) are being considered for next generation optical transport networks (OTN) [1]. Quadrature phase shift keying (QPSK) modulation is the leading candidate for 40Gb/s and 100Gb/s OTN. However, QPSK may suffer from $\pm\pi/2$ phase jumps or *cycle slips* (CS's), which are induced by phase noise. These CS's lead to catastrophic bit errors that cannot be corrected with forward error correction (FEC) codes. The use differential QPSK (DQPSK) modulation avoids the CS problem at the expense of some performance degradation.

Powerful FEC codes with iterative soft-decoding are required for $\geq$ 100 Gb/s coherent optical transmission systems. In DQPSK receivers, a *demapper* block is used to provide soft information to the iterative channel decoder. Traditional low complexity demappers are based on a *serial concatenated architecture* (SCA), as depicted in Fig.1 [2]. However, as we shall show later, the performance of this suboptimal approach with DPQSK modulation is around 1.5 dB worse than that achieved with QPSK [3]. This performance degradation can be combated by using a *turbo concatenated architecture* (TCA) or a *joint architecture* (JA) (see Fig.1). The use of these iterative decoding techniques based on the Bahl, Cocke, Jelinek, and Raviv (BCJR) algorithm, has been reported in past literature [4]. Unfortunately, the high complexity of BCJR-based iterative decoders makes prohibitive their implementation in multigigabit per second commercial optical receivers.

This paper presents a low complexity *joint demapper and decoder* (JDD) architecture for coherent DQPSK modulation. In order to reduce complexity, the proposed JDD is built upon the *sum product algorithm* (SPA) [5]. Although our JDD is general, we consider here its application with low density parity check (LDPC) FEC codes. The new JDD extends the *Factor Graph* (FG) of the LDPC code by including the minimum set of additional factor and variable nodes to represent the statistical relationship between *blocks* of coded bits and received signals. This way, the accuracy of the decoding process of bit blocks can be improved at every iteration over a joint factor graph that includes both the demapper and channel decoding functions. We analyze the JDD using DQPSK modulation with Gray mapping and an LDPC code with 20% overhead (OH) with net-effective coding gain (NCG) of 11.3 dB at a bit-error-rate of 10^{-15} [6]. Simulation results show that the new JDD improves significantly the performance of DQPSK, providing a 0.6 dB gain over the traditional SCA [2].

2. System Model

Figure 2-(A) shows the transmit system. The information bits b_k are grouped into blocks of K-bits, i.e., $\mathbf{b} \in \{0,1\}^K$. Each data block $\mathbf{b}$ is encoded with an LDPC code obtaining an N-bit block $\mathbf{c} \in \{0,1\}^N$ where N is even. Codewords $\mathbf{c}$ are mapped into a sequence of $N/2$ QPSK complex symbols $\mathbf{s}$, with $\mathbf{s} \in \{1, i, -1, -i\}^{N/2}$. Finally, the components of the transmitted symbol block $\mathbf{d}$ is computed using differential modulation, i.e, $d_k = d_{k-1} \cdot s_k$, where $k = 1, \ldots, N/2$ and $d_0 = 1$. The discrete-time baseband receiver signal is given by

$$r_k = d_k + n_k \tag{1}$$

where $k = 1, \ldots, N/2$ and n_k are independent identically distributed (iid) complex Gaussian random variables with zero mean and variance N_0.

[Copia del paper publicado en ArXiv-2012, pg.2/3]

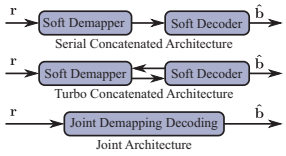


Fig. 1. Decoder architectures

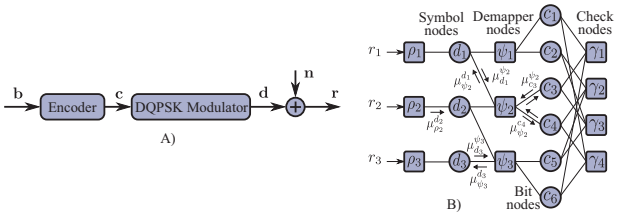


Fig. 2. A) Transmit system model. - B) Proposed JDD Factor Graph

3. Joint Demapper Decoder

We derive a soft-input soft-output (SISO) joint demapper decoder by applying the SPA over the FG of the joint a posteriori probability (APP) mass function $P(\mathbf{c} \mid \mathbf{r})$ of the coded bits given the received symbols. Based on the Bayes rule, we get $P(\mathbf{c} \mid \mathbf{r}) = f(\mathbf{r} \mid \mathbf{c})P(\mathbf{c})/f(\mathbf{r})$. Term $f(\mathbf{r})$ can be neglected for detection, thus the soft decision can be performed over $f(\mathbf{r} \mid \mathbf{c})P(\mathbf{c})$. Since $\mathbf{c} \to \mathbf{s} \to \mathbf{d} \to \mathbf{r}$ is a Markov chain, the probability density function $f(\mathbf{r} \mid \mathbf{c})P(\mathbf{c})$ can be expressed as $f(\mathbf{r} \mid \mathbf{c}) = f(\mathbf{r} \mid \mathbf{d}) \cdot P(\mathbf{d} \mid \mathbf{s}) \cdot P(\mathbf{s} \mid \mathbf{c}) \cdot P(\mathbf{c})$. Terms $P(\mathbf{d} \mid \mathbf{s}) \cdot P(\mathbf{s} \mid \mathbf{c})$ are grouped into one term $P(\mathbf{d} \mid \mathbf{c})$. This simplification avoids the implementation of the variable nodes related the symbols $\mathbf{s}$ and it reduces the implementation complexity of the SPA with no performance degradation. Finally, the symbol/bit level factorization of $f(\mathbf{r} \mid \mathbf{c})P(\mathbf{c})$ is

$$f(\mathbf{r} \mid \mathbf{c})P(\mathbf{c}) = \prod_{k=1}^{N/2} f(r_k \mid d_k) \prod_{k=1}^{N/2} I(d_k \mid c_{2k-1}, c_{2k}) \prod_{i=1}^{M} Q(\mathbf{c}_{(i)}), \tag{2}$$

where $\mathbf{c}_{(i)} = \{c_j \in \mathbf{c} \mid H_{i,j} = 1\}$ are the coded bits related to the i-th parity equation, $Q(\mathbf{c}_{(i)}) = 1$ if the bits in $\mathbf{c}_{(i)}$ have even parity or zero else where and H is the parity check matrix of the code whose dimensions are $M \times N$, and $I(\cdot)$ is an indicator function.

Figure 2-(B) shows the FG of $f(\mathbf{r} \mid \mathbf{c})P(\mathbf{c})$ according to eq. 2. The messages from node x to node y are denoted by the vector μ_x^y. The messages between nodes c and γ are computed using the standard SPA for LDPC codes. The messages between nodes c and ψ are 2 dimensional vectors whose components $\mu_x^y(b)$ for $b \in \{0,1\}$ represent bit probabilities. The messages between nodes ψ, d and ρ are 4 dimensional vectors whose components $\mu_x^y(e^{j\phi})$ for $\phi \in \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$ represent symbol probabilities. The notation $x\langle y \rangle$ represents the two cases of equation, one case using x and other case using y. These messages can be computed as:

$$\mu_{d_k}^{\psi_k \langle \psi_{k+1} \rangle}(e^{j\phi}) = \frac{\mu_{\rho_k}^{d_k}(e^{j\phi}) \cdot \mu_{\psi_{k+1}\langle \psi_k \rangle}^{d_k}(e^{j\phi})}{\sum_{n=0}^{3} \mu_{\rho_k}^{d_k}\left(e^{j\frac{n\pi}{2}}\right) \cdot \mu_{\psi_{k+1}\langle \psi_k \rangle}^{d_k}\left(e^{j\frac{n\pi}{2}}\right)} \tag{3}$$

$$\mu_{\psi_k}^{d_k \langle d_{k-1} \rangle}(e^{j\phi}) = \sum_{n=0}^{1} \sum_{m=0}^{1} \mu_{c_{2k}}^{\psi_k}(n) \cdot \mu_{c_{2k-1}}^{\psi_k}(m) \cdot \mu_{d_{k-1}\langle d_k \rangle}^{\psi_k}\left(e^{j[\phi - \langle + \rangle \Delta(n,m)]}\right) \tag{4}$$

$$\mu_{\psi_k}^{c_{2k} \langle c_{2k-1} \rangle}(n\langle m \rangle) = \sum_{m\langle n \rangle = 0}^{1} \sum_{l=0}^{3} \mu_{d_{k-1}}^{\psi_k}\left(e^{j\frac{\pi}{2}l}\right) \cdot \mu_{d_k}^{\psi_k}\left(e^{j\left[\Delta(n,m) - \frac{\pi}{2}l\right]}\right) \cdot \mu_{c_{2k-1}\langle c_{2k} \rangle}^{\psi_k}(m\langle n \rangle) \tag{5}$$

$$\mu_{\rho_k}^{d_k}(e^{j\phi}) = \frac{1}{\pi N_0} \exp\left(\frac{-\|r_k - e^{j\phi}\|^2}{N_0}\right) \tag{6}$$

where $\Delta(n,m) = \frac{(m-n+2mn)\pi}{2}$.

[Copia del paper publicado en ArXiv-2012, pg.3/3]

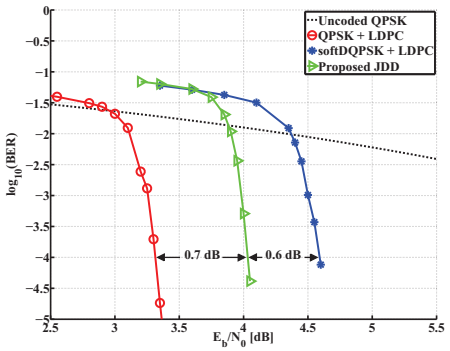


Fig. 3. Performance of DQPSK and QPSK schemes.

4. Numerical Results

Figure 3 shows the BER versus the signal-to-noise ratio per bit (E_b/N_0) for the proposed JDD with DQPSK modulation. An LDPC code of length 24576 bits and rate 0.8334 is used. The number of iterations in the FG is set to 20. Perfect knowledge of the noise power N_0 is assumed at the receiver size. The performance of an SCA based on the soft DQPSK demapper defined by eqs. (4) and (5) in [2], as well as the performance with QPSK modulation and optimal demapping, are also presented for comparison proposes. At least, 1000 errors were counted at each simulation point. Note that JDD provides a 0.6 dB gain at BER=10^{-4} respect to SCA. Furthermore, note that the gap between QPSK and DQPSK is reduced to 0.7 dB.

5. Conclusions

A new low complexity JDD scheme for coherent DQPSK modulation has been presented. An SPA-based demapper has been developed to reduce implementation complexity. Numerical results have shown that JDD outperforms traditional SCA. This good tradeoff between complexity and performance makes the proposed JDD an excellent alternative to improve the performance of DQPSK modulation in next generation optical transport networks.

References

1. D. Crivelli, H. Carter, and M. Hueda, "Adaptive digital equalization in the presence of chromatic dispersion, PMD, and phase noise in coherent fiber optic systems," in "Global Telecommunications Conference (GLOBECOM), IEEE," (2004).
2. M. Kuschnerov, S. Calabro, K. Piyawanno, B. Spinnler, M. Alfiad, A. Napoli, and B. Lankl, "Low complexity soft differential decoding of QPSK for forward error correction in coherent optic receivers," in "Optical Communication (ECOC), 36th European Conference and Exhibition on," (2010).
3. T. Mizuochi, Y. Miyata, K. Kubo, T. Sugihara, K. Onohara, and H. Yoshida, "Progress in soft-decision FEC," in "Optical Fiber Communication Conference and Exposition (OFC/NFOEC), and the National Fiber Optic Engineers Conference," (2011).
4. P. Hoeher and J. Lodge, ""Turbo DPSK": iterative differential PSK demodulation and channel decoding," Communications, IEEE Transactions on **47**, 837–843 (1999).
5. A. Worthen and W. Stark, "Unified design of iterative receivers using factor graphs," Information Theory, IEEE Transactions on **47**, 843–849 (2001).
6. D. Morero, M. Castrillon, F. Ramos, T. Goette, O. Agazzi, and M. Hueda, "Non-concatenated FEC codes for ultra-high speed optical transport networks," in "Global Telecommunications Conference (GLOBECOM), IEEE," (2011).

[Copia del paper publicado en PC-2012, pg.1/2]

A New Cycle Slip Compensation Technique for Ultra High Speed Coherent Optical Communications

Mario A. Castrillon , Damian A. Morero and Mario R. Hueda
Laboratorio de Comunicaciones Digitales - Universidad Nacional de Cordoba - CONICET
Av. Velez Sarsfield 1611 - Cordoba (X5016GCA) - Argentina
ClariPhy Argentina S.A. - Humberto Primo 680 - Cordoba (5000) - Argentina.
Email: acastrillon, dmorero, mhueda @efn.uncor.edu

***Abstract*—We propose a novel cycle slip mitigation algorithm suitable for next generation OTNs. Simulation results of a 100 Gb/s DP-QPSK optical system show almost no degradation at a post-FEC BER of .**

I. Introduction

Coherent detection based receivers with electronic dispersion compensation (EDC) are being used on next generation optical transport networks (OTN) [1]. These receivers allow using phase encoded modulation formats such as phase shift keying (PSK) or quadrature PSK (QPSK).

Carrier phase recovery (CPR) is a key function of coherent optical receivers [1], [2]. In these devices, CPR algorithms are required to compensate for effects such as laser phase noise and carrier frequency fluctuations [3]. However, since QPSK modulation has rotational symmetry, errors in the carrier phase estimation may cause cycle slips (CS). After a CS occurs, all detected symbols are erroneous and they cannot be corrected by forward error correction (FEC) codes [2]. To combat this catastrophic effect, *differential modulation* is typically used [2]. In differential modulation schemes the information is transmitted as the phase difference between two consecutive symbols. Therefore, the effects of a CS do not translate into catastrophic bit errors. While this option provides a solution to the CS problem, it is prone to introduce signal-to-noise ratio (SNR) penalty compared to non-differential schemes. For instance, a penalty of 1.2 dB (post soft-FEC) has been reported for differential QPSK (DQPSK) modulation [4].

To avoid this penalty produced by differential modulation formats, the use of pilot symbols to prevent error propagation has been proposed in previous works [5]–[7]. Although they provide better performance than differential modulation, their implementation in high-speed applications such as 100 Gigabits per second (Gb/s) OTN, is not straightforward. In particular, the need for feedback or iterative precessing requires a careful design of parallel architectures for the implementation of high speed receivers.

In this paper we present a novel pilot-assisted algorithm to detect and correct errors caused by cycle slips. The proposed scheme has a low-complexity forward architecture suitable for high speed parallel implementation. In particular, it can be implemented as a simple post-processing stage after a Viterbi

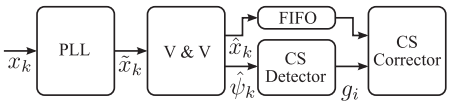


Figure 1. Proposed CS compensation scheme.

Viterbi (VV) based carrier phase estimator [3]. The low density parity check (LDPC) code proposed in [8] is used to investigate the effectiveness of the new CS compensation algorithm. Simulation results of a 100 Gb/s DP-QPSK optical system will demonstrate that the new pilot symbol-based scheme with only of overhead can eliminate the performance degradation caused by cycle slips at a post-FEC BER of .

II. System Model

Figure 1 depicts the application of the proposed CS detector and corrector in combination with a carrier recovery scheme as proposed in [3]. The latter is built upon a low latency phase locked loop (PLL) stage combined with a traditional VV carrier phase estimator. Let the discrete time PLL input signal be modeled as

(1)

where is the complex-valued transmitted symbol at the -th time instant, is the cumulative phase effect due to carrier offset, frequency fluctuations, and laser noise phase; are independent identically-distributed (iid) complex Gaussian random variables with zero mean and variance per dimension. The PLL output is given by

(2)

where is the PLL phase correction, is the residual phase error, and . Finally, the output of the VV is

(3)

where is the error phase estimated by the VV algorithm, and .

[Copia del paper publicado en PC-2012, pg.2/2]

III. Proposed CS Compensation Algorithm

The proposed CS mitigation algorithm is composed of 3 stages. In the first one, a coarse CS detection based on pilot symbols is carried out. Then, a fine estimation of the CS position is done based on the phase estimated by VV. Finally, the errors due to the CS are corrected.

The transmitted information is divided into consecutive blocks of complex symbols, where the first symbols are pilots and the last symbols information. The overhead of pilot symbols is defined as . These pilot symbols are used to estimate the phase based on the minimum euclidean distance criteria as

| | (4)

where is the possible offset produced by a CS, is the pilot symbol value, and indicates the estimated phase offset corresponding to the block. When , a CS in block is detected.

Once a CS is detected, its exact position inside the block is estimated. This is done by comparing the VV phase estimation and a delayed version of itself, . The position of the CS is estimated as

| – | (5)

where , . Parameter is optimized by computer simulations. The idea behind equation 5 is based on the following observations. A CS caused by additive noise generates a *fast* jump of the phase estimated by the VV in the direction of CS. The phase difference between the end and the start of the jump largely cancels the CS jump estimated by pilot symbols. On the other hand, a CS caused by a fast change in the laser phase noise (which cannot be tracked by the VV) generates a peak value in the direction of CS. This peak value partially cancels the CS phase jump estimated by pilot symbols generating a minimum value at the CS position. Finally, after the CS position is estimated, the CS effect is canceled.

IV. Simulation Results

Numerical results derived from computer simulations are presented and analyzed. Dual polarization (DP) QPSK modulation with a transmission rate of 100 Gb/s is analyzed. We use 500 KHz of laser linewidth at Tx and Rx in a non-dispersive optical channel. The adopted carrier recovery scheme is able to efficiently compensate carrier frequency offset, jitter, and laser phase noise [3]. Fig. 2 shows the BER versus the SNR for (*i*) proposed algorithm, (*ii*) an ideal QPSK (i.e., without CS), and (*iii*) DQPSK modulation. The overhead of pilot symbols is 1.01 (this overhead has been considered in the definition of the SNR). An interleaver of 20 codewords has been also incorporated. In Fig. 2 a gain of 1.16 dB compared to DQPSK at post FEC BER of can be observed. Furthermore, it is important to note that performance is only 0.07 dB from the ideal QPSK ones.

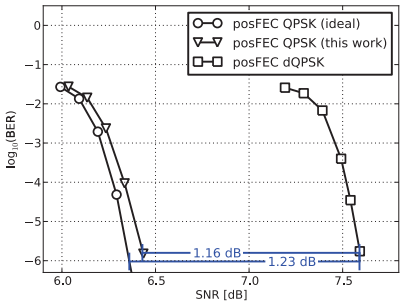


Figure 2. BER vs SNR with laser linewidth of 500 KHz at Tx and Rx

V. Conclusions

In this paper we have introduced a novel algorithm to combat CS in optical coherent communication systems. The proposed scheme can be easily combined with traditional carrier phase estimation algorithms in order to virtually eliminate the impact of the CS on the performance of QPSK receivers. In particular, the new low complexity approach is able to operate at very low OSNR regimes with almost no degradation. Furthermore, the described CS compensation architecture is suitable for parallel implementation in high speed receivers. These features make the proposed technique a very attractive alternative to be used with powerful FECs as required in next-generation optical networks.

References

[1] D. Crivelli et al., "Adaptive digital equalization in the presence of chromatic dispersion, PMD, and phase noise in coherent fiber optic systems," in *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, vol. 4, Dec. 2004, pp. 2545–2551.

[2] M. Taylor, "Phase estimation methods for optical coherent detection using digital signal processing," *Lightwave Technology, Journal of*, vol. 27, no. 7, pp. 901–914, Apr. 2009.

[3] P. Gianni et al., "A new parallel carrier recovery architecture for intradyne coherent optical receivers in the presence of laser frequency fluctuations," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, Dec. 2011, pp. 1–6.

[4] A. Bisplinghoff et al., "Soft decision metrics for differentially encoded QPSK," in *Optical Communication (ECOC), 2011 37th European Conference and Exhibition on*, Sept. 2011, pp. 1–3.

[5] S. Zhang et al., "Pilot-assisted decision-aided maximum-likelihood phase estimation in coherent optical phase-modulated systems with nonlinear phase noise," *Photonics Technology Letters, IEEE*, vol. 22, no. 6, pp. 380–382, Mar. 2010.

[6] X. Wu et al., "Iterative carrier recovery in turbo receivers with distributed pilots," in *Consumer Electronics, Communications and Networks (CECNet), 2011 International Conference on*, Apr. 2011, pp. 5024–5026.

[7] H. Zhang et al., "Cycle slip mitigation in POLMUX-QPSK modulation," in *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference*, Mar. 2011, pp. 1–3.

[8] D. Morero et al., "Non-concatenated FEC codes for ultra-high speed optical transport networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, Dec. 2011, pp. 1–5.

[Copia del paper publicado en Globecom-2012, pg.1/6]

Efficient Concatenated Coding Schemes for Error Floor Reduction of LDPC and Turbo Product Codes

Damian A. Morero*† and Mario R. Hueda*
*Laboratorio de Comunicaciones Digitales - Universidad Nacional de Córdoba - CONICET
Av. Vélez Sarsfield 1611 - Córdoba (X5016GCA) - Argentina
†ClariPhy Argentina S.A. - Humberto Primo 680 - Córdoba (5000) - Argentina

***Abstract*—This work introduces a novel serial code concatenation (SCC) scheme to combat the error floor problem experienced in iterated sparse graph-based error correcting codes such as turbo product (TP) codes and low density parity check (LDPC) codes. SCC has been widely used in the past to reduce the error floor in iterative decoders. However, the main stumbling block for its practical application in high speed communication systems has been the need for long and complex outer codes. The use of short outer block codes with interleaving has been shown to provide a good tradeoff between complexity and performance. Nevertheless, its application to next-generation ultra high-speed communication systems is still a major challenge as a result of the careful design of long complex interleavers needed to meet the requirements of these applications (e.g., a net coding gain >10 dB at a bit error rate of 10^{-15} with an overhead of $\sim$ 20% for 100 Gb/s optical transport networks [1]). In this paper we present a new SCC scheme built from short outer block codes. Unlike previous proposals, the long interleaver is replaced by a simple block code combined with a novel encoding/decoding strategy. Based on this finding, we show that complexity and latency can be drastically reduced with negligible penalty. The SCC technique introduced here provides a new general framework for solving the error floor problem induced by low-weight error patterns of any coding scheme.**

I. Introduction

Powerful forward error correction (FEC) codes are needed to satisfy the requirements imposed by future high speed communication systems. For example, net coding gains (NCGs) $\geq$ 10dB at a bit error rate (BER) of 10^{-15} with an overhead (OH) as low as possible (e.g., $\sim$ 20%) are mandatory for next generation optical transport networks (OTN) [1], [2], [3]. Given their superior performance and suitability for parallel processing, *large* block size low density parity check (LDPC) codes and turbo product (TP) codes have been considered as FEC coding schemes for ultra-high speed transmission systems.

Because of the high implementation complexity of the large block size needed to achieve high NCGs, the use of serial concatenated codes (SCC) is required when aiming at an efficient very large scale integration (VLSI). SCC schemes based on an inner LDPC or TP code with a hard-decision-based outer block code have shown to be a reasonable FEC solution to (*i*) provide NCG$\geq$ 10dB, (*ii*) mitigate the well-known *error floor* problem of LDPC and TP codes[1], and (*iii*) reduce complexity. For instance, several SCC FEC schemes for 100 Gigabits per second (Gb/s) OTN applications have been proposed (see [2], [3] and references therein). In [2] it is experimentally shown that a 20.5% concatenated code based on an inner LDPC and an outer Reed-Solomon (RS) code achieves an NCG of 9 dB at a BER=10^{-13}. The concatenation of two hard-decision block codes with an LDPC is other alternative proposed in [2]. The total overhead of this triple-concatenated approach is 20% and the expected NCG is 10.80 dB at a BER=10^{-15}. A concatenated LDPC+RS coding scheme with 20.5% OH and NCG=11.30 dB at a BER=10^{-15} is proposed in [3].

Practical applications of large block size outer codes to multigigabit systems have been precluded so far as a result of their prohibitive complexity for implementation in integrated circuits. To mitigate this problem, the use of short outer block codes with interleaving has been considered [5]. Although this solution has been shown to provide a good tradeoff between complexity and performance, their application in future high-speed transmission systems is still a major challenge. In particular, a careful design of long interleavers shall be needed to (*i*) mitigate the error floor problem and (*ii*) achieve the expected (high) NCG. Additionally, the use of long interleavers will increase both implementation complexity and latency, two critical factors for commercial applications of SCC schemes.

This paper introduces a novel SCC-based FEC architecture designed for very high-speed applications. The key ingredients of our technique are (*i*) the use of a short outer block code and (*ii*) the replacement of the long interleaver by a simple block code in combination with a novel encoding/decoding strategy. Based on this finding, we show that complexity and latency can be drastically reduced with negligible penalty in comparison with existing SCC solutions. We also demonstrate that our approach is able to combat the error floor caused by both the suboptimal decoding operation of an inner LDPC code, and the minimum distance of an inner TP code. The SCC approach presented in this work provides a new general framework for solving the error floor problem induced by low-weight error patterns of any coding scheme.

The rest of this paper is organized as follows. Section II introduces the error floor problem and describes the classical SCC schemes proposed for ultra high speed transmission systems. The new SCC technique is described and analyzed in Section III. Section IV presents two improved schemes of the

[1]Post-processing algorithms have also been proposed to combat the error floor problem [1] [4]. However, the design of these algorithms for practical high gain coding schemes may be difficult since the knowledge of both the weight and *structure* of the dominant error patterns is required.

proposed SCC for practical applications with inner LDPC and turbo codes (TCs). Finally, conclusions are drawn in Section V.

II. Background

As mentioned before, a high NCG at very low BER (e.g. 10^{-15}) with OH as low as possible will be required in future high speed communication systems. Although it is well known that LDPC and TP codes provide a performance very close to the Shannon limit, their practical application to multigigabit systems has been precluded so far as a result of the error floor problem and the high implementation complexity. To mitigate these drawbacks, SCC-based FEC techniques have been adopted in several high-speed applications such as 100 Gb/s OTN [2], [3], [6]. This section reviews basic concepts related to the error floor problem, and describes the SCC schemes adopted in ultra high-speed devices.

A. Error Floor on LDPC and Turbo Codes

Error floors are usually caused by low-weight error patterns such as *low-weight codewords* or *near codewords* [7], [8]. Low-weight codewords are the main cause of error floor in parallel concatenated TC. This error floor cannot be corrected nor detected. On the other hand, error floor in LDPC codes are caused by low-weight near codewords [8]. In this case, the LDPC decoder is able the *detect* the errors. As we shall show in Section III, this fact will be used to derive the new reduced complexity SCC scheme.

Let Ω be the set of all error-patterns that causes an error floor. An *error-pattern* is defined as the set of all bits in error that jointly take place in one received codeword. Let $p(\omega)$ be the probability of a certain error pattern $\omega \in \Omega$ at a given signal-to-noise ratio (SNR). The *word error rate* (WER) due to Ω is defined by

$$P_w(\Omega) = \sum_{\omega \in \Omega} p(\omega), \tag{1}$$

while the BER is given by

$$P_b(\Omega) = \frac{1}{n} \sum_{\omega \in \Omega} p(\omega) w(\omega), \tag{2}$$

where n is the codeword length and $w(\omega)$ is the weight of the error pattern ω. From (1) and (2), it is simple to derive the following upper bound

$$P_b(\Omega) \lessapprox \frac{w_{max}}{n} P_w(\Omega), \tag{3}$$

where $w_{max} = \max_{\omega \in \Omega} \{w(\omega)\}$. Parameters w_{max} and $P_w(\Omega)$ are used here for designing different code concatenation schemes.

B. Serial Code Concatenation (SCC)

Code concatenation is a known FEC technique based on the combination of an inner code and an outer code [5], [7], [9], [10]. Let $\mathbb{C}_1$ and $\mathbb{C}_2$ denote the inner and outer code, respectively. Each code is defined by the set of parameters $[n_j, k_j, d_j]$, where n_j, k_j, and d_j are the block size, the dimension, and the minimum distance of the code $\mathbb{C}_j$ respectively. The overhead of the code is defined as $\Theta_j = (n_j - k_j)/k_j$. Let C_j^i denote the i-th codeword of $\mathbb{C}_j$. A codeword C_j^i is composed by the information data block D_j^i of length k_j, and the parity block P_j^i of length $r_j = n_j - k_j$. Next we describe the SCC schemes used in high-speed communication systems. In these applications, the inner code $\mathbb{C}_1$ is an LDPC or TP code, while the outer code $\mathbb{C}_2$ is a block code with error capability $t_2 = \lfloor \frac{d_2 - 1}{2} \rfloor$ designed to eliminate or reduce the error floor of $\mathbb{C}_1$.

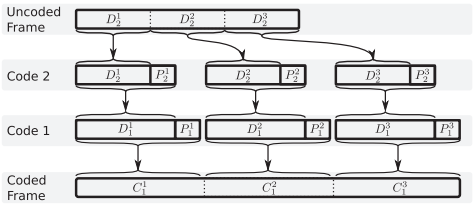


Figure 1. Encoding of SCC-I.

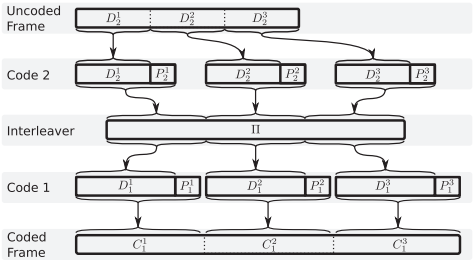


Figure 2. Encoding of SCC-I with interleaving.

SCC Scheme I (SCC-I)

Figure 1 shows a classical serial code concatenation scheme denoted here as *SCC-I*. The encoding process is composed of two steps. First, the uncoded frame is divided into m blocks of k_2 bits denoted D_2^i for $i = 1, \ldots, m$ (e.g., $m = 3$ in Fig. 1). Each D_2^i block is encoded by $\mathbb{C}_2$ generating the codeword C_2^i. In the second step, the codewords C_2^i are used as the dataword of code $\mathbb{C}_1$ (i.e., $D_1^i = C_2^i$) and they are encoded by $\mathbb{C}_1$ generating the codewords C_1^i that will be transmitted. In order to eliminate the error floor of $\mathbb{C}_1$, $\mathbb{C}_2$ must correct at least $t_2 = w_{max}$ bits.

Figure 2 shows a variation of SCC-I where an interleaver is introduced between $\mathbb{C}_1$ and $\mathbb{C}_2$. Based on the structure of the error pattern of $\mathbb{C}_1$, it is possible to design a proper interleaver to divide each error pattern into several codewords of $\mathbb{C}_2$. This way, it is possible to show that not only the correction capability required for $\mathbb{C}_2$ can be *relaxed*, but also the bandwidth overhead penalty can be reduced. Note that these benefits are obtained at the expense of a higher complexity and latency required by the interleaver.

[Copia del paper publicado en Globecom-2012, pg.3/6]

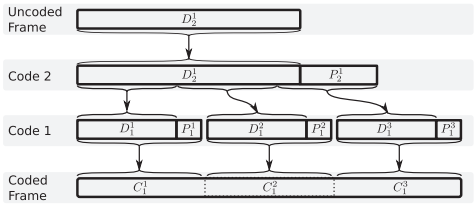


Figure 3. Encoding of SCC-II.

SCC Scheme II (SCC-II)

The second classical code concatenation scheme is depicted in Fig. 3. Compared to the previous scheme, SCC-II uses a longer outer code $\mathbb{C}_2$ to protect a frame of m inner datawords (e.g., $m = 3$ in Fig. 3). Let $t_2 = \tau \cdot w_{max}$ be the error correction capability of $\mathbb{C}_2$. Then, we can conclude that the error floor is eliminated with SCC-II if $\tau = m$. On the other hand, note that the error floor can be reduced but not eliminated if $\tau < m$. This feature of SCC-II can be used to provide a good tradeoff between performance and complexity.

C. Discussion

The overhead of the outer code $\mathbb{C}_2$ given by $\Theta_2 = (n_2 - k_2)/k_2$, is a key factor that defines the *goodness* of SCC-I and SCC-II. This OH must be as low as possible in order to reduce the SNR penalty of the SCC scheme, which is defined as $10 \cdot log_{10}(1 + \Theta_2)$ dB.

Next we assume that $\mathbb{C}_2$ is a narrow-sense binary BCH code [11] with redundancy $\Phi(n_2, t_2) = n_2 - k_2$. Assuming that $t_2 = w_{max}$ (i.e., the error floor is eliminated) and $n_2 = k_1$ with k_1 being the dimension of $\mathbb{C}_1$, the overhead of the SCC-I-based solution can be expressed as

$$\Theta^{(I)} = \Theta_2 = \frac{\Phi(k_1, w_{max})}{k_1 - \Phi(k_1, w_{max})}. \quad (4)$$

As we shall show later, the SNR penalty of SCC-I may be large as a result of a high correction capability required to eliminate the error floor of $\mathbb{C}_1$ (i.e., $t_2 = w_{max}$). This penalty can be reduced by adding an interleaver between $\mathbb{C}_1$ and $\mathbb{C}_2$ (see Fig. 2) [5]. The optimum interleaver depends strongly on the error patterns of the inner code $\mathbb{C}_1$. Unfortunately, this information is not always available for most of the graph based codes. Alternatively, suboptimal random interleaving can be used in SCC-I. Although the design of this suboptimal solution does not require information of the error patterns, its application to high speed communication systems may be difficult due to the need for very large frame sizes.

Next we evaluate the overhead of SCC-II. The error correction capability of $\mathbb{C}_2$ can be expressed as $t_2 = \tau \cdot w_{max}$ with $\tau < m$. In this case, note that the error floor is reduced but not eliminated as in SCC-I. The overhead penalty of SCC-II can be computed as follows:

$$\Theta^{(II)}(\tau) = \frac{\Phi(m \cdot k_1, \tau \cdot w_{max})}{m \cdot k_1 - \Phi(m \cdot k_1, \tau \cdot w_{max})}. \quad (5)$$

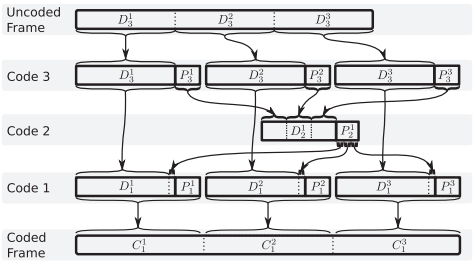


Figure 4. Encoding of the new SCC technique.

In most practical high-speed applications, the following conditions can be verified: $\tau \leq 3$, $w_{max} \leq \frac{1}{2}\sqrt{k_1} \leq 200$, $m \geq 20$, and $k_1 \geq 500$. Then, based on numerical evaluations of (4) and (5), we have found that the ratio $\Theta^{(II)}(\tau)/\Theta^{(I)}$ can be bounded by

$$\frac{\tau}{m} \leq \frac{\Theta^{(II)}(\tau)}{\Theta^{(I)}} \leq 1.75 \cdot \frac{\tau}{m}. \quad (6)$$

On the other hand, the residual error floor of SCC-II can be approximated by

$$P_b^{(II)}(\Omega) \approx \sum_{i=\tau+1}^{m} \frac{i \cdot w_{max}}{m \cdot k_1} \binom{m}{i} [P_w(\Omega)]^i [1 - P_w(\Omega)]^{m-i}. \quad (7)$$

Assuming that $P_b^{(II)}(\Omega)$ satisfies the requirement of the application (e.g., $P_b^{(II)}(\Omega) < 10^{-15}$), and taking into account that $\tau \ll m$, from (6) we conclude that the overhead of an SCC-II-based FEC solution is lower than that required by SCC-I. Unfortunately, the application of SCC-II to future multigigabit systems (e.g., [2], [12]) may be difficult owing to the high NCG required at very low BER. The latter imposes the use of a large block size for the outer code $\mathbb{C}_2$, which increases significantly its implementation complexity.

III. A New Code Concatenation Scheme

We introduce a novel code concatenation scheme to combat the error floor problem experienced in iterated sparse graph-based error correcting codes. As we shall show here, the new approach is able to achieve an error floor reduction similar to that accomplished by the SCC-II. However, our approach builds from short outer block codes as in SCC-I, therefore the implementation complexity can be drastically reduced.

Next we assume that the error floor of the inner code $\mathbb{C}_1$ is caused by a set of detectable error-patterns Ω. Note that this assumption is satisfied by LDPC codes. The new SCC approach uses two short outer block codes (denoted as $\mathbb{C}_2$ and $\mathbb{C}_3$) to combat the error floor of the inner code $\mathbb{C}_1$. The encoding process comprises three steps (see Fig. 4):

1. The uncoded frame is divided into m datawords of k_3 bits denoted D_3^i for $i = 1, \ldots, m$ (e.g., $m = 3$ in the example of Fig. 4). Each dataword D_3^i is encoded by $\mathbb{C}_3$ generating the parity bits P_3^i.

[Copia del paper publicado en Globecom-2012, pg.4/6]

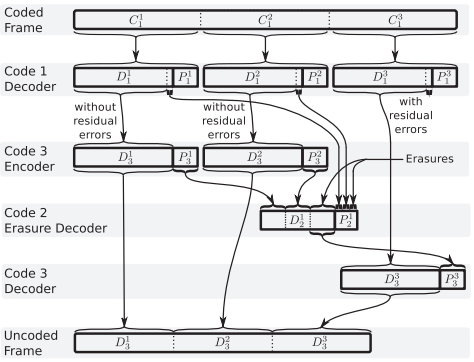


Figure 5. Decoding process of the new SCC.

2. The m parity bits P_3^i are grouped together into the dataword D_2^1 which is encoded by $\mathbb{C}_2$, generating the parity bits P_2^1
3. The parity bits P_2^1 are divided into m sub-blocks of equal (or almost equal) size denoted as $P_2^{1,i}$ with $i = 1, \ldots, m$. Each dataword D_1^i is generated by the concatenation of the dataword D_3^i and the parity bits $P_2^{1,i}$. Finally, each dataword D_1^i is encoded by code $\mathbb{C}_1$ generating the codeword C_1^i to be transmitted over the channel.

Figure 5 presents an example of the decoding process when the error pattern occurs in the third codeword, C_1^3. The process comprises the following steps:

1. Codewords C_1^i for $i = 1, \ldots, m$ are decoded and those containing uncorrectable errors are detected.
2. From the error-free datawords D_1^i obtained at Step 1, the datawords D_3^i are extracted and encoded in order to recover the parity bits P_3^i.
3. The dataword D_2^1 is reconstructed from the parity bits P_3^i generated at Step 2, while the unavailable parity bits (those that belong to the corrupted codewords) are marked as erasures (i.e., P_3^3 in the example of Fig. 5). The parity bits P_2^1 are extracted from the error-free datawords D_1^i and those bits related to the corrupted datawords are marked as erasures. An erasure decoding is carried out over the codeword C_2^1, where the parity bits P_3^i of the corrupted codewords C_3^i are regenerated.
4. Finally, the codewords C_3^i with residual errors are decoded.

A. Overhead of the New SCC Scheme

The proposed SCC achieves the same performance than SCC-II provided that $\mathbb{C}_3$ corrects w_{max} bits and $\mathbb{C}_2$ recovers $\tau \cdot (n_3 - k_3) + \frac{\tau}{m}(n_2 - k_2)$ erased bits. Note that $\tau \cdot (n_3 - k_3)$ are the parity bits of τ codewords of $\mathbb{C}_3$, while $\frac{\tau}{m}(n_2 - k_2)$ are the parity bits of $\mathbb{C}_2$ that belong to τ corrupted codewords of $\mathbb{C}_1$. Assuming that $\mathbb{C}_2$ is a *maximum distance separable* (MDS) code (e.g., RS codes), it is verified that its erasure correction capability is equal to its redundancy [10], therefore

$$n_2 - k_2 = \tau \cdot (n_3 - k_3) + \frac{\tau}{m}(n_2 - k_2), \tag{8}$$

then,

$$n_2 - k_2 = \frac{m \cdot \tau}{m - \tau} \cdot (n_3 - k_3) \tag{9}$$

Finally, the overhead of the new SCC is

$$\Theta^{(III)}(\tau) = \frac{n_2 - k_2}{m \cdot k_3} = \frac{\tau}{m - \tau}\left[\frac{n_3 - k_3}{k_3}\right] \tag{10}$$

Assuming that the dimension of the inner code $\mathbb{C}_1$ is $k_1 > 500$, we have verified numerically that the condition

$$\Theta^{(III)}(\tau) < \Theta^{(II)}(\tau) \tag{11}$$

is satisfied in most practical high-speed applications[2]. From (11) note that the proposed SCC performs better than the classical SCC-II. Moreover, we realize that the implementation complexity is lower than in SCC-II. In particular, note that the new approach builds from two *short* block size outer codes ($\mathbb{C}_2$ and $\mathbb{C}_3$). Furthermore, notice that $\mathbb{C}_2$ requires only one decoding process per frame. Therefore, the complexity of the new SCC is slightly higher than that required by the SCC-I *without interleaving*. From the above, we can conclude that the new SCC scheme offers a better tradeoff between performance and complexity than previous SCC proposals.

B. Example 1

Let $\mathbb{C}_1$ be the Margulis $[2640, 1320]$ LDPC code [13]. This code has an error floor starting at $P_w(\Omega) \approx 10^{-5}$. This error floor is caused by *trapping-sets* (TS), in particular (12,4) TS and (14,4) TS[3] which have 6 and 7 systematic bits, respectively. However, as noticed in [14], there are also trapping sets of weight 15, 16, 17 and 18 bits. We take into account these additional trapping sets by designing an SCC scheme to correct error patterns with $w_{max} = 9$ systematic bits. The solutions for the different SCC schemes are described in the following.

- SCC-I: the outer code $\mathbb{C}_2$ must be the BCH$[1320, 1221, 19]$. Therefore, the SNR penalty and bandwidth overhead are $10 \cdot log_{10}(1320/1221) = 0.3386$ dB and $(1320 - 1221)/1221 = 8.11\%$, respectively.
- SCC-II: the outer code $\mathbb{C}_2$ must be a binary BCH$[47520, 47088, 55]$ code, which corrects up to $\tau = 3$ error-patterns of 9 bits. The error floor is not entirely eliminated, but it is reduced to $P_b^{(II)}(\Omega) \approx 5 \cdot 10^{-19}$ with an SNR penalty and bandwidth overhead of 0.0397 dB and 0.9174%, respectively.
- New SCC: setting $m = 36$ and $\tau = 3$ as in SCC-II (this way, the same residual error floor is achieved), $\mathbb{C}_3$ can be a BCH$[1410, 1311, 19]$ over GF(2^{11}), and $\mathbb{C}_2$ can be an RS$[432, 396, 37]$ over GF(2^9). The number of additional parity bits is $36 \cdot 9 = 324$. This OH introduces

[2]In these cases, $\tau \leq 3$, $w_{max} \leq \frac{1}{2}\sqrt{k_1} \leq 200$, and $m \geq 20$.

[3]In the notation "(e, d) TS", e is the number of wrong bits and d is the number of unsatisfied check nodes (see [13], [14] for more details)

[Copia del paper publicado en Globecom-2012, pg.5/6]

Table I
PERFORMANCE AND COMPLEXITY COMPARISON OF EXAMPLE 1.

Scheme	SNR Penalty	Overhead	Error Floor	Outer Code $\mathbb{C}_2$	Outer Code $\mathbb{C}_3$	Performance	Complexity
SCC-I	0.3386 dB	8.1081 %	None	BCH$[1320, 1221, 19]$	None	Bad	Good
SCC-II	0.0397 dB	0.9174 %	$\approx 5 \cdot 10^{-19}$	BCH$[47520, 47088, 55]$	None	Good	Bad
This work	0.0297 dB	0.6865 %	$\approx 5 \cdot 10^{-19}$	RS$[432, 396, 37]$	BCH$[1410, 1311, 19]$	Good	Good

an SNR penalty and a bandwidth overhead of 0.0297 dB and 0.6865%, respectively.

Table I presents a comparison of the different SCC solutions. Note that

- the performance of SCC-II is better than that of SCC-I at the expense of a higher implementation complexity (due to the longer outer BCH code);
- the new approach achieves better performance than SCC-II with a complexity similar to that of SCC-I[4].

C. Example 2

We use the proposed SCC to improve the performance of the LDPC+RS concatenation scheme designed for next generation optical communication systems [6]. This scheme comprises an inner LDPC$[9252, 7967]$ code and an outer RS$[992, 956, 37]$ code. The total overhead is 20.5% and the NCG at BER=10^{-15} is 10 dB. The outer RS code introduces an SNR penalty of 0.1605 dB.

Next we use the new SCC approach with the same inner LDPC code $\mathbb{C}_1$. For the outer codes, we consider the RS$[833, 797, 37]$ as $\mathbb{C}_3$ (a shortened version of the original RS code), and the RS$[1014, 936, 79]$ code as $\mathbb{C}_2$ where $m = 26$ and $\tau = 2$. From [6], the error pattern probability is $P_w(\Omega) \approx 5 \cdot 10^{-7}$. Then, the residual error floor is $P_b^{(III)}(\Omega) \approx 10^{-19}$. The SNR penalty and bandwidth overhead are 0.0164 dB and 0.378% respectively. Finally, we get NCG$= 10.1441$ dB (vs. 10 dB) with a total overhead of 16.568% (vs. 20.5%).

IV. IMPROVEMENTS OF THE NEW SCC SCHEME

We introduce two improvements of the new SCC suitable for many practical applications based on LDPC codes and TCs. The first one is an optimization for $\tau = 1$ motivated by the fact that most LDPC codes have a low error floor (BER$<10^{-10}$) that can be reduced below 10^{-15} by correcting only one error pattern per frame (i.e., $\tau = 1$). The second approach is a generalization of the SCC technique designed to reduce the error floor caused by low-weight codewords (i.e., undetectable error patterns). This is particularly useful to lower the error floor of TC such as TPC.

A. New SCC with $\tau = 1$

When $\tau = 1$ it is possible to lower the overhead penalty, implementation complexity, and encoder latency by using $(n_3 - k_3)$ *single parity check* (SPC) codes as the outer code $\mathbb{C}_2$. The new encoding process, as depicted in Fig. 6, comprises the following steps:

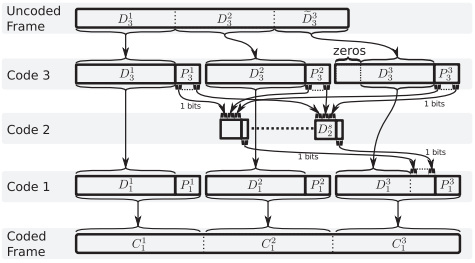


Figure 6. Encoding process of the new SCC optimized for $\tau = 1$.

1. The uncoded frame is divided into m blocks. The first $m-1$ datawords D_3^i for $i = 1, \ldots, m-1$ correspond to the first $m-1$ blocks. The last dataword D_3^m is the concatenation of $(n_3 - k_3)$ zeros and the last block $\tilde{D}_3^m$ of $k_3 - (n_3 - k_3)$ bits.
2. Each dataword D_3^i is encoded by $\mathbb{C}_3$ generating the parity bits P_3^i.
3. For $i = 1, \ldots, n_3 - k_3$ the m-bit dataword D_2^i is the concatenation of the i-th parity bit of P_3^j for $j = 1, \ldots, m$. Each D_2^i is encoded by an SPC code.
4. The dataword for $\mathbb{C}_1$ is $D_1^i = D_3^i$ for $i = 1, \ldots, m-1$. The last dataword D_1^m is the concatenation of $\tilde{D}_3^m$ and the $n_3 - k_3$ parity bits generated in Step 3. Finally, each dataword D_1^i is encoded by $\mathbb{C}_1$.

The decoding process is similar to that of the original scheme (e.g., see Fig. 5). The main benefits of this optimized scheme are:

- The implementation complexity of the encoder and decoder of the outer code $\mathbb{C}_2$ is very low since they can be implemented with $n_3 - k_3$ XOR gates of m input bits.
- It is possible to show that the overhead penalty is reduced to $\Theta^{(III)}(1) = \frac{1}{m}\left[\frac{n_3-k_3}{k_3}\right]$ because the corrupted parity-bits of $\mathbb{C}_2$ do not have to be erased.
- The latency is reduced because the P_2^i parity bits are transmitted in the last codeword of $\mathbb{C}_1$ (i.e., the parity bits can be computed while the codewords of $\mathbb{C}_1$ are being transmitted).

The performance of the improved SCC approach with $m = 10$ is verified by using computer simulations in Fig. 7. The Margulis $[2640, 1320]$ code is used as inner code, $\mathbb{C}_1$. Owing to time constraints, an artificial high error floor with probability $P_w(\Omega) = 10^{-3}$ is inserted after the decoding process. Similarly

[4]In particular, note that the outer BCH code of the new SCC and the outer BCH code of SCC-I belong to the same primitive BCH code over GF(2^{11}) (they have practically the same complexity). Also notice that the second outer RS code of the new SCC is simple and requires only one decoding process per frame.

[Copia del paper publicado en Globecom-2012, pg.6/6]

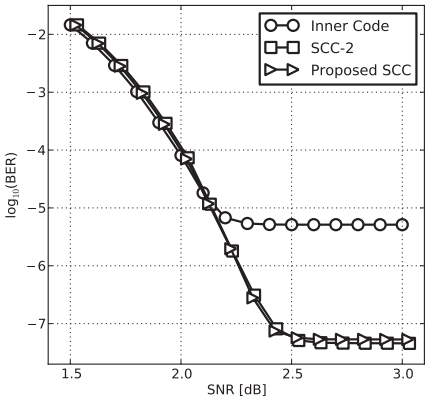


Figure 7. BER vs SNR for SCC-II and the new SCC with $\tau = 1$.

to the *real* error floor, the artifical BER is generated from error patterns with 7 information bits and 7 parity bits. Figure 7 shows the BER of the inner code $\mathbb{C}_1$ (circles), SCC-II (squares) and the proposed SCC (triangles). For SCC-II, the outer code $\mathbb{C}_2$ is a BCH$[13200, 13102, 15]$. On the other hand, $\mathbb{C}_3$ is a BCH$[1397, 1320, 15]$ while $\mathbb{C}_2$ is an SPC$[11, 10, 2]$ code for the new SCC. From Fig. 7 note that $\mathbb{C}_1$ has an error floor at BER $\approx \frac{7}{1320} P_w(\Omega) = 5.3 \cdot 10^{-6}$. This error floor is reduced by the outer codes to BER $\approx 4.73 \cdot 10^{-8}$. From this figure we see that the new SCC is able to achieve a similar performance than SCC-II. It is important to realize that our technique achieves this performance by using short outer block codes. Compared with SCC-II, this fact reduces significantly the implementation complexity in integrated circuits.

B. Generalization to Correct Low-Weight Codewords

The proposed SCC strategy can be generalized to reduce also the error floor due to low-weight codewords (i.e. undetectable error patterns), which is particularly useful for turbo codes. Figure 8 shows the encoding process of the generalized SCC. Unlike the previous case, a subset of the parity bits P_3^i of $\mathbb{C}_3$, denoted as $\hat{P}_3^i$, is not encoded by $\mathbb{C}_2$. Instead, this subset is transmitted as a part of the dataword D_1^i of the inner code $\mathbb{C}_1$. In the decoding process, $\hat{P}_3^i$ is used to detect the corrupted $\mathbb{C}_1$-codewords. The decoding process starts by applying the decoder of code $\mathbb{C}_1$ to the m received codewords C_1^i. After that, the dataword D_1^i is extracted from C_1^i. For each dataword D_1^i, the dataword D_3^i is extracted and *partially* encoded with $\mathbb{C}_3$ in order to regenerate only the parity bits $\hat{P}_3^i$. If these regenerated parity bits are not equal to the corresponding bits in D_1^i, this dataword is marked as corrupted. Once all corrupted datawords D_1^i are identified, the rest of the decoding process continues as in the original SCC.

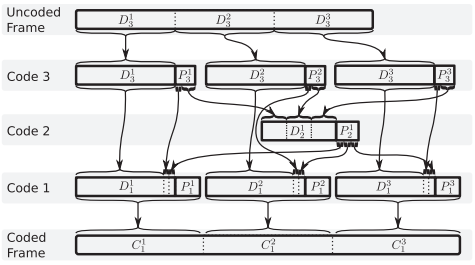


Figure 8. Encoding process of the generalized SCC.

V. Conclusions

We have introduced a novel SCC scheme to combat the error floor problem experienced in iterated sparse graph-based error correcting codes. This SCC scheme is built upon two short outer codes combined with a novel encoding/decoding strategy. We have shown that the new approach reduces significantly the complexity with negligible penalty. The proposed SCC can be efficiently used to combat the error floor experienced in both LDPC and TP codes. In particular, the new SCC approach can be used to improve the performance in high-speed optical communications, where high coding gain and very low BER are required.

References

[1] D. Morero, M. Castrillon, F. Ramos, T. Goette, O. Agazzi, and M. Hueda, "Non-concatenated FEC codes for ultra-high speed optical transport networks," *GLOBECOM. IEEE*, pp. 1 –5, Dec. 2011.

[2] K. Onohara *et. al.*, "Soft-decision-based forward error correction for 100 Gb/s transport systems," *IEEE J. Sel. Topics Quantum Electron.*, vol. 16, no. 5, pp. 1258 –1267, Sept.-Oct. 2010.

[3] N. Kamiya and S. Shioiri, "Concatenated QC-LDPC and SPC codes for 100 Gbps ultra long-haul optical transmission systems," *OFC/NFOEC*, pp. 1–3, March 2010.

[4] Z. Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. Wainwright, "Lowering LDPC error floors by postprocessing," *GLOBECOM. IEEE*, pp. 1 –6, Dec. 2008.

[5] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding," *Information Theory, IEEE Transactions on*, vol. 44, no. 3, pp. 909 –926, May 1998.

[6] Y. Miyata, W. Matsumoto, H. Yoshida, and T. Mizuochi, "Efficient FEC for optical communications using concatenated codes to combat error-floor," *OFC/NFOEC*, pp. 1 –3, Feb. 2008.

[7] W. Ryan and S. Lin, "Channel codes: Classical and modern," *Cambridge University Press*, 2009.

[8] T. Richardson, "Error floors of LDPC codes," *Proc. of the 41st Allerton Conf.*, Oct. 2003.

[9] G. D. Forney, "Concatenated codes," *Cambridge, MA: MIT Press*, 1966.

[10] W. C. Huffman and V. Pless, "Fundamentals of error-correcting codes," *Cambridge University Press*, 2003.

[11] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information Control*, vol. 3, pp. 68–79, March 1960.

[12] B. Vasic and E. M. Kurtas, "Coding and signal processing for magnetic recording systems," *CRC Press*, 2005.

[13] D. J. MacKay and M. S. Postol, "Weaknesses of Margulis and Ramanujan-Margulis low-density parity-check codes," *Electronic Notes in Theoretical Computer Science*, 2003.

[14] Y. Han and W. Ryan, "LDPC decoder strategies for achieving low error floors," *Information Theory and Applications Workshop*, pp. 277 –286, Feb. 2008.

[Copia del paper publicado en CJECE, pg.1/8]

Novel serial code concatenation strategies for error floor mitigation of low-density parity-check and turbo product codes

Nouvelles stratégies de concaténation de codes séries pour la réduction du seuil d'erreur dans le contrôle de parité à faible densité et dans les turbo codes produits

Damian A. Morero and Mario R. Hueda*

This paper presents a novel multiple serial code concatenation (SCC) strategy to combat the error-floor problem in iterated sparse graph-based error correcting codes such as turbo product-codes (TPC) and low-density parity-check (LDPC) codes. Although SCC has been widely used in the past to reduce the error-floor in iterative decoders, the main stumbling block for its practical application in high-speed communication systems has been the need for long and complex outer codes. Alternative, short outer block codes with interleaving have been shown to provide a good tradeoff between complexity and performance. Nevertheless, their application to next-generation high-speed communication systems is still a major challenge as a result of the careful design of long complex interleavers needed to meet the requirements of these applications. The SCC scheme proposed in this work is based on the use of short outer block codes. Departing from techniques used in previous proposals, the long outer code and interleaver are replaced by a simple block code combined with a novel encoding/decoding strategy. This allows the proposed SCC to provide a better tradeoff between performance and complexity than previous techniques. Several application examples showing the benefits of the proposed SCC are described. Particularly, a new coding scheme suitable for high-speed optical communication is introduced.

Cet article présente une nouvelle stratégie de concaténation de codes séries (SCC) multiples permettant de s'affranchir du problème de seuil d'erreur dans les codes correcteurs d'erreurs tels que les Turbo codes produits (TPC) et les codes contrôle de parité à faibles densité (LDPC). Bien que dans le passé, la SCC ait été largement utilisée pour réduire le seuil d'erreur dans les décodeurs itératifs, le principal obstacle pour son implémentation pratique dans les systèmes de communications haut débit a été le recours à des codes extérieurs longs et complexes. De courts codes en blocs extérieurs avec entrelacement ont montré qu'ils peuvent fournir un bon compromis entre la complexité et la performance. Cependant, leur implémentation dans la prochaine génération de systèmes de communications haut débit reste un défit majeur vu que le résultat d'une conception minutieuse des entrelaceurs longs et complexes doit remplir les exigences de ces systèmes. Le schéma SCC proposé dans ce travail repose sur l'utilisation de courts codes en blocs extérieurs. Partant des techniques existantes, le code extérieur long et l'entrelaceur sont remplacés par un simple code en bloc combiné à une nouvelle stratégie de codage/ décodage. Ainsi, par comparaison aux techniques courantes, la stratégie SCC proposée fournit un meilleur compromis entre la performance et la complexité. Plusieurs exemples d'applications sont donnés et montrent les avantages de la stratégie¸SCC proposée. En particulier, nous présentons un nouveau schéma de codage adapté aux communications optiques haut débit.

Keywords: concatenated codes; error correction codes; high-speed optical communication; product codes; turbo codes.

I Introduction

In future high speed communication systems (e.g., next generation optical transport networks (OTN)), forward error correction (FEC) codes with net coding gains (NCGs) ≥ 10 dB at a bit error rate (BER) of 10^{-15} with an overhead (OH) as low as possible (e.g., $\sim$ 20%) are mandatory [1]–[3]. Given their superior performance and suitability for parallel processing, *large* block size low density parity check (LDPC) codes and turbo product (TP) codes have been considered as FEC coding schemes of choice for ultra-high speed transmission systems. Unfortunately, these iterative coding schemes usually have error floor problems which significantly degrade their performance at low BER.

Manuscript received January 27, 2013; accepted March 7, 2013
* D. A. Morero and M. R. Hueda are with Laboratorio de Comunicaciones Digitales - Universidad Nacional de Cordoba - CONICET. Av. Velez Sarsfield 1611 - Cordoba (X5016GCA) - Argentina; Email: dmorero@gmail.com, mhueda@gmail.com
This work was supported in part by Fundación Fulgor.
Associate Editor managing this paper's review: S. Yousefi

Numerous techniques have been proposed in the literature to lower the error floor [1]–[5]. These techniques can be divided into two categories. The first one aims at eliminating all weaknesses in the decoder algorithm that create the error floor, while the second one aims at correcting the residual error pattern by adding an outer code. The first category comprises several post-processing [1], [4] and improved decoding algorithms [5] which are mainly proposed for LDPC codes. The design and performance evaluation of these algorithms may be difficult since the knowledge of both the weight and *structure* of the dominant error patterns is required. On the other hand, the addition of an outer code provides a simple and more general solution to the error floor problem. In particular, its performance can be estimated based on the knowledge of the weights and the probabilities of the error patterns. For these reasons, several SCC FEC schemes for 100 Gigabits per second (Gb/s) OTN applications have been proposed (see [2]–[3] and references therein). In [2], it is experimentally shown that a 20.5% concatenated code based on an inner LDPC and an outer Reed-Solomon (RS) code achieves an NCG of 9 dB at a BER= 10^{-13}. The concatenation of two hard-decision block codes with an LDPC is another alternative proposed in [2]. The total overhead of

[Copia del paper publicado en CJECE, pg.2/8]

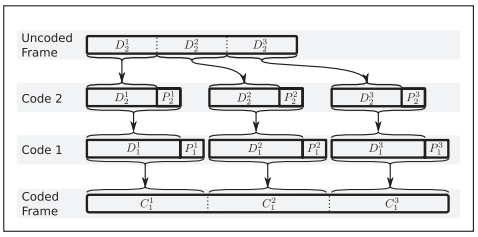


Figure 1: *Encoding of SCC-I.*

this triple-concatenated approach is 20% and the expected NCG is 10.80 dB at a BER=10^{-15}. In [3], the authors present a concatenated LDPC+RS coding scheme with 20.5% OH and NCG=11.3 dB at a BER=10^{-15}.

The use of short outer block codes with interleaving has also been considered in the past to (*i*) improve the performance and (*ii*) reduce the error floor [6]. Based on the structure of the dominant error patterns, the interleaving-based SCC solution is able to achieve a good tradeoff between performance and complexity. However, the evaluation of the dominant error patterns is highly complex in numerous codes of practical interest as a result of the very low BER and the high NCGs required in high-speed applications such as OTN. To avoid the evaluation of the structure of the error patterns, long pseudo-random interleavers can be used [6]. Unfortunately, long interleavers significantly increase not only the implementation complexity but also the latency. Therefore, the use of SCC with interleaving in future high-speed transmission systems is still a major challenge.

The present paper describes a novel SCC strategy designed to reduce the error floor problem in very high-speed communication systems. The key ingredient of our technique is the replacement of the long outer code and interleaver by simple block codes in combination with a novel encoding/decoding strategy [7]. Based on this finding, we demonstrate that an error floor reduction of LDPC codes in multigigabit applications can be achieved with a drastic reduction of complexity[1] (e.g., one order of magnitude) in comparison with existing SCC solutions. As a second contribution of this work, we extend the new SCC strategy to combat the error floor caused by both (*i*) the near-codewords[2] with non-zero syndrome [8] and (*ii*) the minimum distance codewords. The latter allows the new SCC technique to be efficiently used with TP codes (TPC). In particular, we show that a TPC composed by two extended Hamming (EH) codes, in combination with the proposed SCC algorithm, can achieve an NCG of $\sim$11.2 dB at BER $=10^{-15}$ with $\sim$22 % total overhead and error floor at $\sim 7\cdot 10^{-17}$. It is important to highlight that this NCG is approximately 0.4 dB higher than that achieved by TPC schemes reported in past literature [2],[9]–[10].

The rest of this paper is organized as follows. Section 2 introduces the basic notation used along the paper and describes the classical SCC schemes proposed for ultra high speed transmission systems. The new SCC technique is described and analyzed in Section 3. An example of the use of the proposed SCC for error floor reduction of TPC with application in high-speed optical communication is presented in Section 4. Finally, Section 5 reviews the main conclusions of the paper.

II Background

This section introduces basic concepts and notation used in the paper. Let Ω be the set of all possible error-patterns at the output of the inner decoder. An *error-pattern* is defined as the set of all bits in error that jointly take place in one received codeword. Let $p(\omega)$ be the probability of a certain error pattern $\omega \in \Omega$ at a given signal-to-noise ratio (SNR). The *exact* word error rate (WER) due to Ω is defined by

$$P_w(\Omega) = \sum_{\omega\in\Omega} p(\omega), \tag{1}$$

while the *transmission* BER, $P_b(\Omega)$, and the *information* BER, $\tilde{P}_b(\Omega)$, are given by

$$P_b(\Omega) = \frac{1}{n}\sum_{\omega\in\Omega} p(\omega)w(\omega), \tag{2}$$

$$\tilde{P}_b(\Omega) = \frac{1}{k}\sum_{\omega\in\Omega} p(\omega)\tilde{w}(\omega), \tag{3}$$

where n and k are the length and the dimension of the code, respectively, while $w(\omega)$ and $\tilde{w}(\omega)$ are the weight (including the redundant bits) and the *information*-weight (which does not include the redundant bits) of the error pattern ω, respectively. Set Ω can be divided into two disjoint subsets $\mathfrak{Q}$ and Ω (i.e., $\Omega = \mathfrak{Q} \cup \Omega$ and $\mathfrak{Q} \cap \Omega = \varnothing$), where $\mathfrak{Q}$ is the set of all error-patterns that causes the error floor and Ω are the non-problematic error patterns. From (1) and (3), it is simple to derive the following upper bound for the information bit error rate due to $\mathfrak{Q}$:

$$\tilde{P}_b(\mathfrak{Q}) = \frac{1}{k}\sum_{\omega\in\mathfrak{Q}} p(\omega)\tilde{w}(\omega) \le \frac{w_{max}}{k} P_w(\mathfrak{Q}), \tag{4}$$

where $w_{max} = \max_{\omega\in\mathfrak{Q}}\{\tilde{w}(\omega)\}$ [11]. Parameters w_{max} and $P_w(\mathfrak{Q})$ will be used throughout this paper to design various code concatenation schemes.

II.A Serial Code Concatenation (SCC)

Code concatenation is a known FEC technique based on the combination of an inner code and an outer code [6], [12]–[14]. Let $\mathbb{C}_1$ and $\mathbb{C}_2$ denote the inner and outer code, respectively. Each code is defined by the set of parameters $[n_j, k_j, d_j]$, where n_j k_j and d_j are the block size, the dimension, and the minimum distance of the code $\mathbb{C}_j$ respectively. The overhead of the code is defined as $\Theta_j = (n_j - k_j)/k_j$. Let C_j^i denote the i th codeword of $\mathbb{C}_j$. A codeword C_j^i is composed by the information data block D_j^i of length k_j, and the parity block P_j^i of length $r_j = n_j - k_j$. In high-speed communication systems, the inner code $\mathbb{C}_1$ is generally an LDPC or TP code, while the outer code $\mathbb{C}_2$ is a block code with error correction capability $t_2 = \lfloor (d_2 - 1)/2 \rfloor$ designed to eliminate or reduce the error floor of $\mathbb{C}_1$.

SCC Scheme I (SCC-I)

Figure 1 shows a classical serial code concatenation scheme denoted here by *SCC-I*. The encoding process is composed of two steps. First, the uncoded frame is divided into m blocks of k_2 bits denoted D_2^i for $i = 1,\ldots,m$ (e.g., $m = 3$ in Fig. 1). Each D_2^i block is encoded by $\mathbb{C}_2$ generating the codeword C_2^i. In the second step, the codewords C_2^i are used as the dataword of code $\mathbb{C}_1$ (i.e., $D_1^i = C_2^i$) and they are encoded by $\mathbb{C}_1$ generating the codewords C_1^i that will be transmitted. In order to eliminate the error floor of $\mathbb{C}_1$ generated by the error patterns $\mathfrak{Q}$, $\mathbb{C}_2$ must correct at least w_{max} bits. Note that $\mathbb{C}_2$ will also correct the error patterns $\omega \in \Omega$ with $\tilde{w}(\omega) \le t_2$ but it may introduce additional errors in the error patterns $\tilde{\Omega} = \{\omega \in \Omega : \tilde{w}(\omega) > t_2\}$. Since the decoder for $\mathbb{C}_2$ modifies a maximum of t_2 bits, the maximum number of additional errors introduced over the error patterns $\tilde{\Omega}$ is no larger than t_2. Therefore, as a worst case scenario, the information BER $\tilde{P}_b(\Omega)$ is increased by a factor $(2t_2+1)/(t_2+1) < 2$. This penalty can be neglected at very low BER (such as 10^{-15}) where the

[1] In this paper we use the equivalent number of gates of different logic cells (e.g., AND, XOR, etc.) to compute the complexity of a given SCC approach.

[2] Let C be a binary linear code of length *n*. An (*a, b*) near-codeword is a binary vector of length *n* and Hamming weight *a* whose syndrome has weight *b* [8]. Particularly, if $b = 0$, (*a, b*) is a codeword. The error-floor in LDPC codes is typically dominated by (*a, b*) near-codewords where *b* is small but higher than zero, and *a* is lower than the minimum distance of the code.

[Copia del paper publicado en CJECE, pg.3/8]

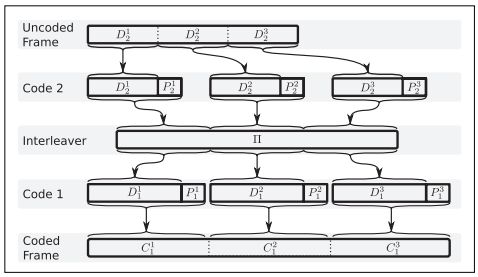


Figure 2: *Encoding of SCC-I with interleaving.*

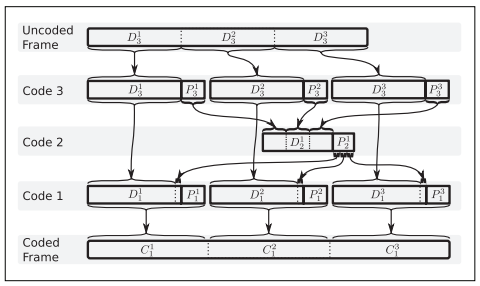


Figure 4: *Encoding of the proposed SCC technique.*

slope of the BER vs. SNR curve is very high (in particular for codes with performance close to the Shannon limit).

Figure 2 shows a variation of SCC-I where an interleaver is introduced between $\mathbb{C}_1$ and $\mathbb{C}_2$. Depending on the structure of the error pattern of $\mathbb{C}_1$, it may be possible to design a proper interleaver to divide each error pattern into several codewords of $\mathbb{C}_2$. In this way, the correction capability required for $\mathbb{C}_2$ can be relaxed.

SCC Scheme II (SCC-II)

The SCC-I scheme can be generalized as depicted in Fig. 3. This scheme, denoted SCC-II, uses a longer outer code $\mathbb{C}_2$ to protect a frame of m inner datawords (e.g., $m=3$ in Fig. 3). Let $t_2 = \tau \cdot w_{max}$ be the error correction capability of $\mathbb{C}_2$. Then, the error floor is eliminated if $\tau = m$. On the other hand, if $\tau < m$ the error floor is reduced but not eliminated. Based on the binomial distribution [11], the residual error floor $\tilde{P}_b^{(II)}(\Omega)$ of SCC-II can be approximated by

$$\tilde{P}_b^{(II)}(\Omega) \approx \sum_{i=\tau+1}^{m} \frac{i \cdot w_{max}}{m \cdot k_1} \binom{m}{i} [P_w(\Omega)]^i [1 - P_w(\Omega)]^{m-i}. \qquad (5)$$

In virtually all applications $P_w(\Omega) < 10^{-4}$, $w_{max} < 50$ and $k_1 > 500$. Therefore, by choosing $m = 20$ and $\tau = 4$ it is possible to reduce the error floor below 10^{-17}. Furthermore, because the required value of τ is significantly lower than m, the outer code overhead of SCC-II is much lower than that of SCC-I. However, this advantage comes at the expense of implementing an outer code m-times longer with a correction capability τ-times higher. Unfortunately, this complexity increase makes the use of SCC-II prohibitive in most high speed applications.

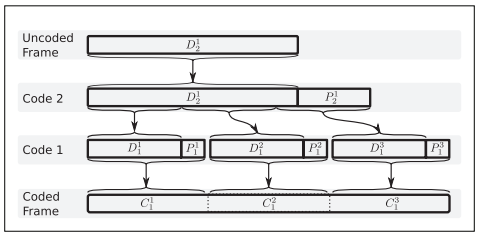


Figure 3: *Encoding of SCC-II.*

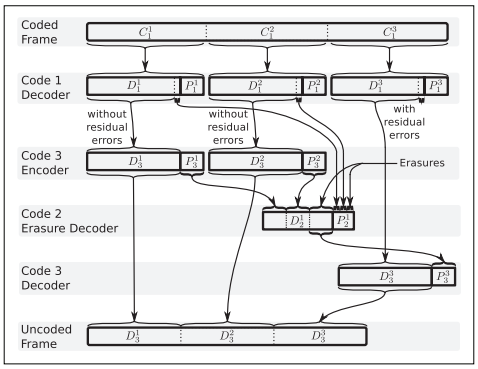


Figure 5: *Decoding process of the proposed SCC.*

III New serial code concatenation strategy

This section describes a novel SCC scheme to combat the error floor. The new technique is able to achieve an error floor reduction similar to that accomplished by the SCC-II. However, the new approach builds on short outer block codes as in SCC-I; therefore the implementation complexity can be significantly reduced.

Next, we assume that the error floor of the inner code $\mathbb{C}_1$ is caused by a set of *detectable* error-patterns Ω as experienced in most LDPC codes (i.e. the error-patterns are not codewords). The new SCC approach uses two short outer block codes (denoted as $\mathbb{C}_2$ and $\mathbb{C}_3$) to combat the error floor of the inner code $\mathbb{C}_1$. The encoding process comprises three steps (see Fig. 4):

1) The uncoded frame is divided into m datawords of k_3 bits denoted D_3^i for $i = 1,\ldots,m$ (e.g., $m=3$ in the example of Fig. 4). Each dataword D_3^i is encoded by $\mathbb{C}_3$ generating the parity bits P_3^i.
2) The m parity blocks P_3^i are grouped together into the dataword D_2^1 which is encoded by $\mathbb{C}_2$, generating the parity bits P_2^1
3) The parity bits P_2^1 are divided into m sub-blocks of equal (or almost equal) size denoted as $P_2^{1,i}$ with $i = 1,\ldots,m$. Each dataword D_1^i is generated by the concatenation of the dataword D_3^i and the parity bits $P_2^{1,i}$. Finally, each dataword D_1^i is encoded by code $\mathbb{C}_1$ generating the codeword C_1^i to be transmitted over the channel.

Figure 5 presents an example of the decoding process when the error pattern occurs in the third codeword, C_1^3. The process comprises the following steps:

[Copia del paper publicado en CJECE, pg.4/8]

Table 2
Binary BCH and RS code complexities

	Block	Register	2-bits XOR	2-bits Mux	Const Mul	Mul	Gates Count
BCH	Encoder	$2t$	$2tp$	$2p$	0	0	$8tp+15t+7p$
	Syndrome	tq	tpq	tq	t	0	$4tpq+2tq^2+7tq$
	Key Equation	$4(t+1)q$	$(2t+1)q$	$3t+3+tq$	0	$3t+1$	$24tq^2+5.5tq+22q+28.5t+26.5$
	Chien Search	tq	tpq	tq	tp	t	$2tpq^2+8tq^2-tq+6t$
	FIFO	$k+5tp$	0	0	0	0	$0.8(k+5tp)$
RS	Encoder	$2tq$	$2tpq$	$2pq$	$2tp$	0	$4tpq^2+15tq+7pq$
	Syndrome	$2tq$	$2tpq$	$2tpq$	$2tp$	0	$4tpq^2+15tpq+7pq$
	Forney Synd.	$2tq$	$2tq$	$2tq$	0	$2t$	$30tq+2t(8q^2-12q+6)$
	Erasure Locator	$2tq$	$2tq$	0	0	$2t-1$	$23tq+(2t-1)(8q^2-12q+6)$
	Errata Evaluator	$(3t+1)pq$	$3tpq$	$2(3t-1)pq$	$2(3t-1)p$	0	$pq(31.5t+12tq-4q+8.5)$
	FIFO	$(n+4tp+8p)q$	0	0	0	0	$0.8(n+4tp+8p)q$

where p,q,t,n,k are the parallelism factor, Galois field size, correction capability, code length, and code dimension respectively.

Table 1
TSMC gate count of basic cells

Cell	Gate Count
Inverter (INVD0BWP35)	0.5
2-input AND (AN2D2BWP35)	2.0
2-input XOR (GXOR2D2BWP35)	4.0
2-input MUX (MUX2D2BWP35)	3.5
D Flip-Flop (DFD2BWP35)	7.5
One Bit of a RAM†	0.8

† estimated based on the area relation between a 1024x16 SPSRAM and a NAND ND2D1BWP35

1) Codewords C_1^i for $i=1,\ldots,m$ are decoded and those containing uncorrectable errors are detected.

2) From the error-free datawords D_1^i obtained at Step 1, the datwords D_3^i are extracted and encoded in order to recover the parity bits P_3^i.

3) The dataword D_2^1 is reconstructed from the parity bits P_3^i generated at Step 2, while the unavailable parity bits (those that belong to the corrupted codewords) are marked as erasures (e.g., P_3^3 in the example of Fig. 5). The parity bits P_2^1 are extracted from the error-free datawords D_1^i and those bits related to the corrupted datawords are marked as erasures. An erasure decoding is carried out over the codeword C_2^1, where the parity bits P_3^i of the corrupted codewords C_3^i are regenerated.

4) Finally, the codewords C_3^i with residual errors are decoded.

III.A Performance and overhead

The proposed SCC achieves a performance similar to the one derived from SCC-II if $\mathbb{C}_3$ corrects w_{max} bits and $\mathbb{C}_2$ recovers $\tau\cdot(n_3-k_3)+(\tau/m)\cdot(n_2-k_2)$ erased bits. Note that $\tau\cdot(n_3-k_3)$ are the parity bits of τ codewords of $\mathbb{C}_3$, while $(\tau/m)\cdot(n_2-k_2)$ are the parity bits of $\mathbb{C}_2$ that belong to τ corrupted codewords of $\mathbb{C}_1$. Assuming that $\mathbb{C}_2$ is a *maximum distance separable* (MDS) code (e.g., RS codes), it is verified that its erasure correction capability is equal to its redundancy [14], therefore

$$n_2-k_2=\tau\cdot(n_3-k_3)+\frac{\tau}{m}(n_2-k_2), \tag{6}$$

then,

$$n_2-k_2=\frac{m\cdot\tau}{m-\tau}\cdot(n_3-k_3). \tag{7}$$

Finally, the overhead of the new SCC is

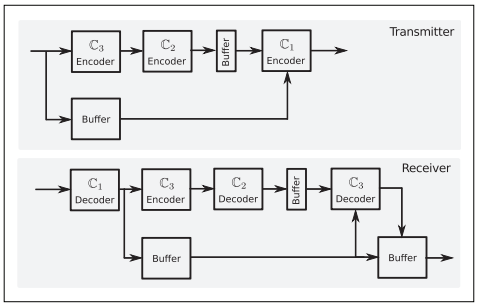


Figure 6: *Encoding and Decoding Block Diagram of the Proposed SCC Scheme.*

$$\Theta(\tau)=\frac{n_2-k_2}{m\cdot k_3}=\frac{\tau}{m-\tau}\left[\frac{n_3-k_3}{k_3}\right]. \tag{8}$$

Numerical evaluation of (8) shows that the overhead of the proposed SCC scheme is lower than that required by the classical SCC-I and SCC-II schemes in practical high-speed applications where $k_1>500$ $\tau\le 4$, $w_{max}\le\sqrt{k_1}/2\le 200$, and $m\ge 20$ [7].

III.B Complexity

The proposed SCC scheme is composed of three encoders at the transmitter side (one for each code) and one $\mathbb{C}_3$ encoder and three decoders at the receiver side (one decoder for each code), as observed in Fig. 6. Let $\Phi(\mathbb{C}_x^e,\mathcal{T})$ and $\Phi(\mathbb{C}_x^d,\mathcal{T})$ be the complexity of the encoder and decoder respectively of code $\mathbb{C}_x$ operating at throughput $\mathcal{T}$ bits/s. In this work, we use the *equivalent number of gates* of different logic cells (e.g., AND, XOR, etc.) as a complexity measure of a given SCC approach. The equivalent gate count is computed based on the TSMC cells described in Table 1 [15]. The estimated number of logic gates of RS over GF (2^q) and BCH codes for a classic p-parallel implementation is summarized in Table 2 [16]–[17].

The total complexity Φ_T of the proposed SCC scheme is

$$\begin{aligned}\Phi_T &\approx 2\Phi(\mathbb{C}_3^e,\mathcal{T})+\Phi(\mathbb{C}_3^d,(\tau/m)\mathcal{T})+\Phi(\text{Buffer})\\ &+ \Phi(\mathbb{C}_2^e,(r_3/k_3)\mathcal{T})+\Phi(\mathbb{C}_2^d,(r_3/k_3)\mathcal{T}),\end{aligned} \tag{9}$$

where the decoder of $\mathbb{C}_3$ has to correct $\le\tau$ of the m codewords per frame and therefore its throughput is reduced to $\approx(\tau/m)\mathcal{T}$. Similarly, the decoder of $\mathbb{C}_2$ has to correct one codeword per frame

[Copia del paper publicado en CJECE, pg.5/8]

Table 3
Performance and complexity comparison of example 1

Scheme	SNR Penalty	Overhead	Error Floor	Outer Code $\mathbb{C}_2$	Outer Code $\mathbb{C}_3$	Relative Complexity
SCC-I	0.3386 dB	8.1081 %	None	BCH[1320, 1221, 19]	None	1.00
SCC-II	0.0397 dB	0.9174 %	$\approx 5 \cdot 10^{-19}$	BCH[47520, 47088, 55]	None	6.00
This work	0.0297 dB	0.6865 %	$\approx 5 \cdot 10^{-19}$	RS[432, 396, 37]	BCH[1410, 1311, 19]	0.67

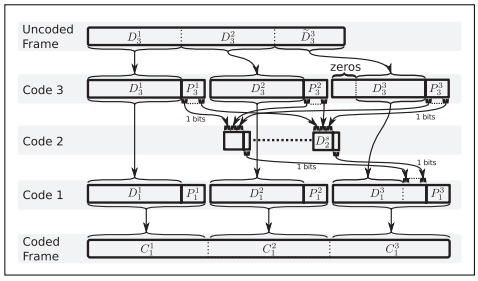


Figure 7: Encoding process of the proposed SCC optimized for $\tau = 1 \cdot$

which reduces its throughput to $\approx (r_3 / k_3)\mathcal{T}$. Finally, Φ(Buffer) is the complexity of approximately $3mk_1$ bits of buffering required to compensate for the latencies of the different codes. These complexities depend on the adopted outer codes. A good tradeoff between performance and complexity can be achieved by using the new SCC with an RS code over GF (2^q) as code $\mathbb{C}_2$ and a binary BCH code as code $\mathbb{C}_3$. As we shall show in the following examples, the complexity of the proposed SCC scheme is significantly lower than that of the existing SCC as a result of the throughput reduction of the decoders.

III.C Example 1: Concatenated LDPC+RS+BCH Code

Let $\mathbb{C}_1$ be the [2640,1320] Margulis LDPC code [8]. This code has an error floor starting at $P_w(\Omega) \approx 10^{-5}$. This error floor is caused by *trapping-sets* (TS), in particular (12,4) TS and (14,4) TS[3] which have 6 and 7 information bits, respectively. However, as noticed in [18], there are also trapping sets of weight 15, 16, 17 and 18 bits. We take into account these additional trapping sets by designing an SCC scheme to correct error patterns with $w_{max} = 9$ information bits. The solutions for the different SCC schemes are described in the following items:

- SCC-I: the outer code $\mathbb{C}_2$ must be the BCH [1320,1221,19] . Therefore, the SNR penalty and bandwidth overhead are $10 \cdot log_{10}(1320/1221) = 0.3386$ dB and $(1320 - 1221)/1221 = 8.11\%$, respectively.

- SCC-II: the outer code $\mathbb{C}_2$ must be a binary BCH [47520,47088,55] code, which corrects up to $\tau = 3$ error-patterns of 9 bits. The error floor is not entirely eliminated, but it is reduced to $\tilde{P}_b^{(II)}(\Omega) \approx 5 \cdot 10^{-19}$ with an SNR penalty and bandwidth overhead of 0.0397 dB and 0.9174% , respectively.

- Proposed SCC: setting $m = 36$ and $\tau = 3$ as in SCC-II (in this way, the same residual error floor is achieved), $\mathbb{C}_3$ can be a BCH [1410,1311,19] over GF (2^{11}) , and $\mathbb{C}_2$ can be an RS [432,396,37] over GF(2^9). The number of additional parity bits is $36 \cdot 9 = 324$. This OH introduces an SNR penalty and a bandwidth overhead of 0.0297 dB and 0.6865% , respectively.

The complexity of the three SCC schemes was estimated as described in Section 3.2. A base parallelism factor $p = 160$ bits was used as a reference to achieve a throughput of $\mathcal{T} \approx 100$ Gb/s in 28nm CMOS technology operating at a clock frequency of 625 MHz. Table 3 summarizes the relative complexity of the three schemes and their performances. Note that:

- The performance of SCC-II is better than that of SCC-I at the expense of a higher implementation complexity (because of the longer outer BCH code).

- The new SCC achieves an NCG ~ 0.3 dB higher than that provided by the SCC-I with a similar complexity.

- The performances of SCC-II and the new SCC algorithm are similar. However, the implementation complexity of our technique is approximately one order of magnitude lower with respect to the SCC-II.

III.D Example 2: Concatenated LDPC+RS+RS Code

As a second example, we consider the LDPC+RS concatenation scheme proposed in [19], which has been designed for next generation optical communication systems. This scheme comprises an inner LDPC [9252,7967] code and an outer RS [992,956,37] code. The total overhead is 20.5% and the NCG at BER=10^{-15} is 10 dB. The outer RS code introduces an SNR penalty of 0.1605 dB. Next, we use the new SCC approach with the same inner LDPC code $\mathbb{C}_1$ defined before (i.e., LDPC [9252,7967]). For the outer codes, we consider the RS [830,794,37] as $\mathbb{C}_3$ (a shortened version of the original RS code), and the RS [1014,936,79] code as $\mathbb{C}_2$ where $m = 26$ and $\tau = 2$. Note that the throughput of $\mathbb{C}_2$ is $k_3 / r_3 \approx 22.14$ times lower than the original RS (i.e., RS [992,956,37]), therefore the extra complexity required by the new SCC is negligible. From [19], the error pattern probability is $P_w(\Omega) \approx 5 \cdot 10^{-7}$. Then, the residual error floor is $\tilde{P}_b(\Omega) \approx 10^{-19}$. The SNR penalty and bandwidth overhead are 0.0164 dB and 0.378% respectively. Therefore, the NCG is increased from 10 dB to 10.1441 dB and the total overhead is reduced from 20.5% to 16.568% . This not only reduces the SNR requirement of the system but also increases the spectral efficiency and reduces the power dissipation (since the sampling rate can be reduced because of the lower overhead).

III.E Example 3: Concatenated LDPC+SPC+BCH Code

Most LDPC codes proposed for optical applications have a low error floor (BER $< 10^{-10}$) that can be reduced below 10^{-15} by correcting only one error pattern per frame (i.e., $\tau = 1$). For these cases, it is possible to lower the overhead penalty and implementation complexity by using $(n_3 - k_3)$ *single parity check* (SPC) codes as the outer code $\mathbb{C}_2$. The new encoding process, as depicted in Fig. 7, comprises the following steps:

1) The uncoded frame is divided into m blocks. The first $m-1$ datawords D_3^i for $i = 1,\ldots,m-1$ correspond to the first $m-1$ blocks. The last dataword D_3^m is the concatenation of $(n_3 - k_3)$ zeros and the last block $\tilde{D}_3^m$ of $k_3 - (n_3 - k_3)$ bits.

2) Each dataword D_3^i is encoded by $\mathbb{C}_3$ generating the parity bits P_3^i .

[3] In the notation "(e, d) TS", e is the number of wrong bits and d is the number of unsatisfied check nodes (see [8] and [18] for more details)

[Copia del paper publicado en CJECE, pg.6/8]

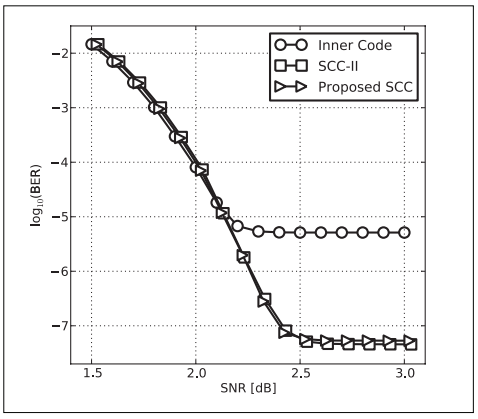


Figure 8: *BER vs SNR for SCC-II and the proposed SCC with* $\tau = 1$.

3) For $i = 1,\ldots,n_3 - k_3$ the m-bit dataword D_2^i is the concatenation of the i-th parity bit of P_3^j for $j = 1,\ldots,m$. Each D_2^i is encoded by an SPC code.

4) The dataword for $\mathbb{C}_1$ is $D_1^i = D_3^i$ for $i = 1,\ldots,m-1$. The last dataword D_1^m is the concatenation of D_3^m and the $n_3 - k_3$ parity bits generated in Step 3. Finally, each dataword D_1^i is encoded by $\mathbb{C}_1$.

The decoding process is similar to that of the original scheme (e.g., see Fig. 5). The main benefits of this optimized scheme are:

- The implementation complexity of the encoder and decoder of the outer code $\mathbb{C}_2$ is very low since they can be implemented recursively with $n_3 - k_3$ XOR gates and Flip-Flops, i.e. $\Phi(\mathbb{C}_2^e) + \Phi(\mathbb{C}_2^d) = 2(n_3 - k_3)(\Phi(\text{XOR}) + \Phi(\text{FlipFlop})) = 23 \cdot (n_3 - k_3)$ gates.

- The overhead penalty is reduced from $[1/(m-1)] \cdot [(n_3 - k_3)/k_3]$ (see eq. (8) with $\tau = 1$) to $(1/m) \cdot [(n_3 - k_3)/k_3]$ because the corrupted parity-bits of $\mathbb{C}_2$ do not have to be erased.

- The encoder latency is reduced because the P_2^i parity bits are transmitted in the last codeword of $\mathbb{C}_1$ (i.e., the parity bits can be computed while the codewords of $\mathbb{C}_1$ are being transmitted). This also reduces the buffer length from $\approx 3mk_1$ to $\approx 2mk_1$ bits.

The performance of the optimized SCC approach for $\tau = 1$ is evaluated by using computer simulations in Fig. 8. The [2640,1320] Margulis LDPC code with $m = 10$ is used as inner code, $\mathbb{C}_1$. Owing to time constraints, an artificially high error floor with probability $P_w(\Omega) = 10^{-3}$ is inserted after the decoding process. As the *real* error floor, the artificial BER is generated from error patterns with 7 information bits and 7 parity bits. Fig. 8 shows the BER of the inner code $\mathbb{C}_1$ (circles), SCC-II (squares) and the proposed SCC (triangles). For SCC-II, the outer code $\mathbb{C}_2$ is a BCH[13200,13102,15] . On the other hand, $\mathbb{C}_3$ is a BCH[1397,1320,15] while $\mathbb{C}_2$ is an SPC[11,10,2] code for the new SCC. From Fig. 8 note that $\mathbb{C}_1$ has an error floor at BER $\approx (7/1320)P_w(\Omega) = 5.3 \cdot 10^{-6}$. This error floor is reduced by the outer codes to BER $\approx 4.73 \cdot 10^{-8}$. From this figure we see that the new SCC is able to achieve a similar performance to SCC-II. It is important to realize that our technique achieves this performance by using short outer block codes. Compared with SCC-II, this fact reduces significantly the implementation complexity in integrated circuits. Particularly, using a parallelism factor $p = 160$ to achieve a throughput $\mathcal{T} \approx 100$ Gb/s in 28nm CMOS technology operating at a clock frequency of 625 MHz as in example 1 (see section 3.3), the complexity of SCC-II is ≈ 575 Kgates while the complexity of the proposed scheme is ≈ 107 Kgates, i.e. 5.38 times lower.

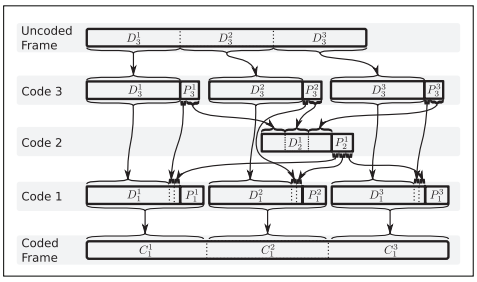


Figure 9: *Encoding process of the generalized SCC.*

IV Error Floor Reduction in TPC

The SCC strategy introduced previously can be extended to mitigate the error floor caused by low-weight codewords (i.e., undetectable error patterns). Note that this feature is particularly useful for decoding of turbo codes such as turbo product codes (TPC). This approach uses a subset of g parity check bits of $\mathbb{C}_3$ to detect those inner codewords with residual errors after the inner code decoder[4].

IV.A Generalized SCC Scheme

Figure 9 depicts the encoding process of the generalized SCC. Unlike in the previous strategy, in the generalized SCC a subset of g parity bits of P_3^i , denoted as $\hat{P}_3^i$, is not encoded by $\mathbb{C}_2$. Instead, this subset is transmitted as a part of the dataword D_1^i of the inner code $\mathbb{C}_1$. In the decoding process, (should it say Figure 10 here?) $\hat{P}_3^i$ is used to detect the corrupted $\mathbb{C}_1$-codewords. The decoding process starts by applying the decoder of code $\mathbb{C}_1$ to the m received codewords C_1^i . After that, the dataword D_1^i is extracted from C_1^i . For each dataword D_1^i , the dataword D_3^i is extracted and *partially* encoded with $\mathbb{C}_3$ in order to regenerate only the parity bits $\hat{P}_3^i$. If these regenerated parity bits are not equal to the corresponding bits in D_1^i , this dataword is marked as corrupted. Once all corrupted datawords D_1^i are identified, the rest of the decoding process continues as in the original proposed SCC.

The generalized SCC scheme may have a residual error floor caused by the occurrence of more than t error patterns $\in \Omega$ in the same frame. This residual error floor, denoted as $\tilde{P}_b^{(1)}(\Omega)$, can be estimated from (5). Additionally, a residual error floor caused by an error pattern that cannot be detected by the g parity bits $\hat{P}_3^i$ is also possible. The probability $P_w^{(2)}(\Omega)$ that an undetectable error pattern $\omega \in \Omega$ takes place in the inner codeword C_1^i can be computed as

$$P_w^{(2)}(\Omega) = \sum_{\omega \in \Omega} I_v\left(\hat{P}_3(\omega) = 0\right) p(\omega) \tag{10}$$

where $\hat{P}_3(\omega)$ are the first g parity bits of $\mathbb{C}_3$ associated with the error pattern $\omega \in \Omega$ and $I_v(X)$ is the *Iverson operator* which is equal to 1 if the statement X is true, and 0 otherwise. Because the error

[4] It is also possible to use an additional *error-detecting code,* such as a *cyclic redundancy check* (CRC) code, as part of the proposed SCC scheme to detect those inner codewords with residual errors. Furthermore, in a different approach, it is also possible to replace the erasure decoder of code C_2 by an error-correcting decoder at the expense of increasing the minimum distance of C_2.

[Copia del paper publicado en CJECE, pg.7/8]

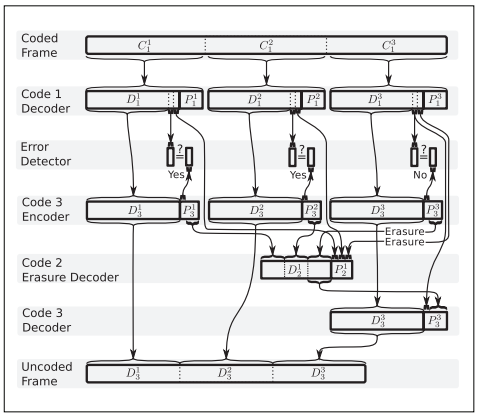


Figure 10: *Description of the decoding process of the generalized SCC.*

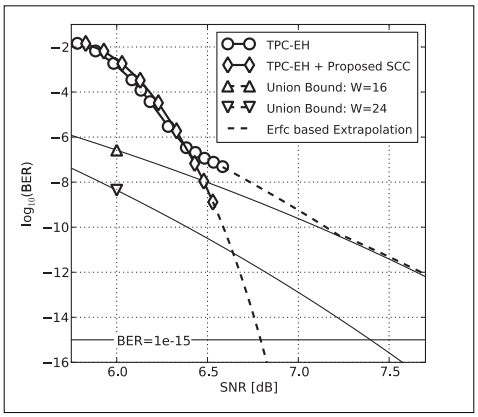


Figure 11: *BER vs SNR simulations of the TPC-EH and the TPC-EH combined with the proposed SCC*

patterns of different inner codewords are independent, the probability of at least one undetectable error pattern in the frame can be computed based on the binomial distribution as

$$P_f^{(2)}(\Omega) = \sum_{i=1}^{m} \binom{m}{i} [P_w^{(2)}(\Omega)]^i [1 - P_w^{(2)}(\Omega)]^{m-i}, \tag{11}$$

while the BER can be estimated as

$$\tilde{P}_b^{(2)}(\Omega) \approx \sum_{i=1}^{m} \binom{m}{i} \frac{i \cdot w_{max}}{m \cdot k_1} [P_w^{(2)}(\Omega)]^i [1 - P_w^{(2)}(\Omega)]^{m-i}. \tag{12}$$

IV.B Generalized SCC with Turbo Product Codes (TPC)

As mentioned before, powerful FEC codes must be designed to satisfy the need of future multigigabit transmission systems. For instance, net coding gains >10 dB at a BER of 10^{-15} and overhead of $\sim$ 20% are mandatory for next generation OTN [2]. In order to meet these requirements, numerous LDPC and TP codes have been reported in the literature (e.g., see [2] and references therein). In particular, TPC based on ≥ 2-error-correcting BCH codes (or TPC-BCH) with block sizes ≥ 32 Kbits have been used in high-speed systems to provide an acceptable tradeoff between performance and complexity [9]–[10], [20]. The feasibility of TPC-BCH for commercial applications at 100 Gbps with NCG of $\sim$11.4 dB at BER $= 10^{-15}$ and a total overhead of $\sim$ 20 % has been demonstrated in [20].

In the following, we consider the use of the proposed generalized SCC technique to improve the behavior of TPC in high-speed applications. We demonstrate that a TPC based on simple *extended Hamming codes* (TPC-EH) with a block size of 8192 bits and minimum distance of 16, can be combined with the new SCC strategy in order to achieve an NCG of $\sim$11.2 dB at BER $= 10^{-15}$ with $\sim$22 % total overhead and error floor at $\sim 7 \cdot 10^{-17}$. Notice that the this performance is: (*i*) 0.45 dB better than the one achieved by the BCH(144,128,5) $\times$ BCH(256,239,6) TPC with block size of 36864 bits and minimum distance 30 proposed in [9]; (*ii*) 0.4 dB better than the one accomplished by the BCH(128,113,6) $\times$ BCH(256,239,6) TPC with block size of 32768 bits and minimum distance 36 proposed in [10], and (*iii*) 0.4 dB better than achieved by the triple concatenated-codes proposed in [2]. Furthermore, the implementation complexity of the TPC-based proposed SCC technique is expected to be lower than that of non-concatenated TPC-BCH schemes. This is mainly because the component codes in the TPC-EH with the proposed SCC are much simpler than those required in the non-concatenated TPC-BCH codes. In particular, the latter requires longer BCH codes with an error correction capability higher than that of the EH code in order to reduce the error-floor. These features make the TPC-EH with the proposed SCC, introduced here, a suitable option for next-generation optical fiber communication networks [2].

IV.C Concatenated TPC-EH + BCH + RS

Let the inner code $\mathbb{C}_1$ be a TPC based on two extended Hamming codes with parameters [128,120,4] and [64,57,4]. Let A_w be the number of codewords with weight w in $\mathbb{C}_1$. For $w < 28$, A_w can be computed as described in [21] obtaining

$$A_w = \begin{cases} 1, & \text{if } w = 0 \\ 888943104, & \text{if } w = 16 \\ 7154214100992, & \text{if } w = 24 \\ 0, & \text{other } w < 28 \end{cases} \tag{13}$$

Figure 11 shows the BER vs SNR of this code when it is decoded with 8 turbo iterations between the two component codes. The optimal *maximum a-posteriori probability* (MAP) decoder proposed in [22] is used to decode both EH codes. As observed in Fig. 11, $\mathbb{C}_1$ has a error floor at a BER $\leq 10^{-7}$ which is caused by the A_{16} codewords of minimum weight $w_{min} = 16$ [23]. Fig. 11 also reports the BER estimations based on the union bound [23] for codewords of weight 16 and 24.

In order to avoid an error floor at BER $= 10^{-15}$, from Fig. 11 we infer that codewords with weight 24 must be corrected. Furthermore, since a suboptimal iterative decoder algorithm is used, non-codeword stopping sets also have to be analyzed. The latter can be computed as described in [24], given that the minimum non-codeword stopping sets have weight 24 and multiplicity 81782765568 (i.e. $\approx$87 times lower A_{24}). A frame composed of $m = 19$ shorted TPC (7859,6507) codewords is required to accommodate the 122368 bits of the *Optical Channel data Unit* (ODU) of the G.709 OTN frame [25] and the parity bits of the outer codes. To reduce the error floor below 10^{-15} a correction capability of $\tau = 2$ error patterns of weight ≤ 24 and a corrupted codeword detection based on $g = 32$ bits are used. The shortened BCH [6753,6441,49] over GF (2^{13}) with a correction capability of 24 bits is used as code $\mathbb{C}_3$. Code $\mathbb{C}_2$ is the shorted RS [596,532,65] code over GF (2^{10}). Therefore, the total additional overhead due to $\mathbb{C}_2$ and $\mathbb{C}_3$ is $\approx$1.02% which introduces an SNR penalty of 0.044 dB. The residual error rate caused by the occurrence of more than τ error patterns is $\tilde{P}_b^{(1)}(\Omega) \approx 7 \cdot 10^{-17}$, while the residual error rate due to undetectable error patterns is $\tilde{P}_b^{(2)}(\Omega) \approx 4.3 \cdot 10^{-18}$. These values have

[Copia del paper publicado en CJECE, pg.8/8]

been derived from (5) and (12), respectively, with $P_w(\Omega) \approx 5 \cdot 10^{-6}$, which has been obtained from computer simulation with SNR = 6.7 dB.

The error floor problem of the TPC-EH can also be solved with schemes SCC-I and SCC-II. However, as it will be shown below, the proposed SCC scheme provides a better performance vs. complexity tradeoff:

- SCC-I requires a frame of $m = 19$ shorted TPC $(8105, 6753)$ combined with m BCH $[6753, 6441, 49]$. The total overhead is 25.8%, the NCG is 11.05 dB and the complexity is 2.02 times higher than that of the proposed scheme. Therefore, the proposed SCC scheme has better spectral efficiency, 0.15 dB higher NCG and lower complexity than SCC-I.

- SCC-II requires a frame of $m = 19$ shorted TPC $(7836, 6484)$ combined with one BCH $[123196, 122380, 97]$. Similarly to the proposed scheme, the total overhead is 21.7% and the NCG is 11.2 dB. However, the complexity is 6.73 times higher, representing a significant complexity advantage in favor of the proposed scheme.

The above illustrates the advantages of the proposed scheme for implementing forward error correction codes for high speed applications. Particularly, the proposed TPC-EH + RS + BCH scheme here proposed represents a low complexity alternative to the non-concatenated TPC based on ≥ 2 -error-correcting BCH codes [2],[9]–[10] since the later has higher implementation complexity than EH codes.

V Conclusions

We have introduced a novel SCC scheme to combat the error floor problem experienced in iterated sparse graph-based error correcting codes. This SCC scheme is based on the use of two short outer codes combined with a novel encoding/decoding strategy. We have shown that the new approach significantly reduces the complexity with negligible penalty. The proposed SCC can be efficiently used with both LDPC and TP codes. In particular, the new SCC approach can be used to improve the performance of high-speed optical communication systems, where high coding gain and very low BER are required. The SCC technique introduced in this work provides a new general framework for solving the error floor problem induced by low-weight error patterns of any coding scheme.

References

[1] D.A. Morero, et al., ``Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks,'' *IEEE Global Telecomm. Conf.,* pp.1–5, Dec. 2011

[2] K. Onohara, et al., ``Soft-Decision-Based Forward Error Correction for 100 Gb/s Transport Systems,'' *IEEE J. Sel. Topics Quantum Electron.,* vol.16, no.5, pp.1258–1267, Sept.–Oct. 2010

[3] N. Kamiya and S. Shioiri, ``Concatenated QC-LDPC and SPC codes for 100 Gbps ultra long-haul optical transmission systems,'' *Optical Fiber Comm. (OFC), collocated National Fiber Optic Eng. Conf. (OFC/NFOEC),* pp.1–3, March 2010

[4] Z. Zhengya, et al., ``Lowering LDPC Error Floors by Postprocessing,'' *IEEE Global Telecomm. Conf.,* pp.1–6, Nov.-Dec. 2008

[5] N. Varnica, M. Fossorier, A. Kavcic, ``Augmented Belief-Propagation Decoding of Low-Density Parity-Check Codes,'' *IEEE Trans. Comm.* vol.54, no.10, pp.1896, Oct. 2006

[6] S. Benedetto, et al.; , ``Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding,'' *IEEE Trans. Inf. Theory,* vol.44, no.3, pp.909–926, May 1998

[7] D.A. Morero and M.R. Hueda, ``Efficient concatenated coding schemes for error floor reduction of LDPC and turbo product codes,'' *IEEE Global Telecomm. Conf.,* pp.1–5, Dec. 2012

[8] D.J. MacKay and M.S. Postol, ``Weaknesses of Margulis and Ramanujan-Margulis low-density parity-check codes,'' *Elect. Notes in Theoretical Computer Science,* 2003.

[9] T. Mizuochi, et al., ``Forward error correction based on block turbo code with 3-bit soft decision for 10-Gb/s optical communication systems,'' *IEEE J. Sel. Topics Quantum Electron.,* vol.10, no.2, pp.376–386, March-April 2004

[10] M. Akita, et al., ``Third generation FEC employing turbo product code for long-haul DWDM transmission systems,'' *Optical Fiber Comm. (OFC),* pp.289–290, Mar. 2002

[11] J. G. Proakis, ``Digital Communications,'' *McGraw-Hill Higher Education, Third Edition,* 1996.

[12] W. Ryan and S. Lin, ``Channel codes: Classical and modern,'' *Cambridge University Press,* 2009

[13] G.D. Forney, ``Concatenated codes,'' Cambridge, MA: MIT Press, 1966

[14] W.C. Huffman and V. Pless, ``Fundamentals of error-correcting codes,'' *Cambridge University Press,* 2003.

[15] Taiwan Semiconductor Manufacturing Company Ltd, ``N28HP standard cell library,'' *Datasheet TCBN28HPBWP35,* Nov. 2010.

[16] S. Lin and D. Costello, ``Error control coding, fundamental and applications,'' *Pearson Prentice Hall, Second Edition,* 2004.

[17] Hsie-Chia Chang, et al., ``A Universal VLSI Architecture for Reed-Solomon Error-and-Erasure Decoders,'' *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol.56, no.9, pp.1960–1967, Sept. 2009

[18] H. Yang; W.E. Ryan, ``LDPC decoder strategies for achieving low error floors,'' *Inf. Theory and Applications Workshop,* pp.277–286, Jan.-Feb. 2008

[19] Y. Miyata, et al., ``Efficient FEC for Optical Communications using Concatenated Codes to Combat Error-floor,'' *Optical Fiber Comm. (OFC), collocated National Fiber Optic Eng. Conf. (OFC/NFOEC),* pp.1–3, Feb. 2008

[20] S. Dave, et al., ``Soft-decision forward error correction in a 40-nm ASIC for 100-Gbps OTN applications,'' *Optical Fiber Comm. (OFC), collocated National Fiber Optic Eng. Conf. (OFC/NFOEC),* pp.1–3, Mar, 2011

[21] L.M.G.M. Tolhuizen, ``More results on the weight enumerator of product codes,'' *IEEE Trans. Inf. Theory,* vol.48, no.9, pp.2573–2577, Sep. 2002

[22] A. Ashikhmin and S. Litsyn, ``Simple MAP decoding of first-order Reed-Muller and Hamming codes,'' *IEEE Trans. Inf. Theory,* vol.50, no.8, pp.1812–1818, Aug. 2004

[23] F. Chiaraluce and R. Garello, ``Extended Hamming product codes analytical performance evaluation for low error rate applications,'' *IEEE Trans. Wireless Comm.,* vol.3, no.6, pp.2353–2361, Nov. 2004

[24] E. Rosnes, ``Stopping Set Analysis of Iterative Row-Column Decoding of Product Codes,'' *IEEE Trans. Inf. Theory,* vol.54, no.4, pp.1551–1560, Apr. 2008

[25] Int. Telecomm. Union, ``Interfaces for the optical transport network,'' *ITU-T G.709,* Feb. 2010.

Damian A. Morero received with honors the degree in electronic engineering from the National University of Cordoba (UNC), Cordoba, Argentina where he is currently working toward the Ph.D. degree in Engineering Science. In 2003 and 2005, he received the Academic Excellence Award from the Engineers Association of Cordoba Argentina and the UNC respectively. From 2006 to 2009, he received a Ph.D. Fellowships from the Secretary of Science and Technology (SeCyT), Argentina. He is currently with ClariPhy Argentina S.A. where he has been engaged in the research and development of error correction coding schemes for high speed optical communications. His research interests include coding, information theory and signal processing.

Mario R. Hueda received the degree in electrical and electronic engineering and the Ph.D. degree from the National University of Cordoba, Cordoba, Argentina, in 1994 and 2002, respectively. From March 1994 to 1996, he received a fellowship from the Scientific and Technological Research Council of Cordoba to carry out research and development in the area of voiceband-data transmission. During the summer of 1996, he was a Visiting Scholar with Lucent Technologies-Bell Laboratories, Murray Hill, NJ, where he worked on code-division multiple-access receivers. Since 1997, he has been with the Digital Communications Research Laboratory, Department of Electronic Engineering, National University of Córdoba. He is currently with the National Scientific and Technological Research Council (CONICET), Cordoba. His research interests include digital communications and performance analysis of communication systems.

[Copia de la patente de USA, pg.1/28]

US 20120221914A1

(19) **United States**

(12) **Patent Application Publication**

Morero et al.

(10) **Pub. No.: US 2012/0221914 A1**

(43) **Pub. Date: Aug. 30, 2012**

(54) **NON-CONCATENATED FEC CODES FOR ULTRA-HIGH SPEED OPTICAL TRANSPORT NETWORKS**

(76) Inventors: **Damian Alfonso Morero**, Cordoba (AR); **Mario Alejandro Castrillon**, Cordoba (AR); **Teodoro Ariel Goette**, Cordoba (AR); **Matias German Schnidrig**, Cordoba (AR); **Facundo Abel Alcides Ramos**, Cordoba (AR); **Mario Rafael Hueda**, Cordoba (AR)

(21) Appl. No.: **13/406,452**

(22) Filed: **Feb. 27, 2012**

Related U.S. Application Data

(60) Provisional application No. 61/447,620, filed on Feb. 28, 2011.

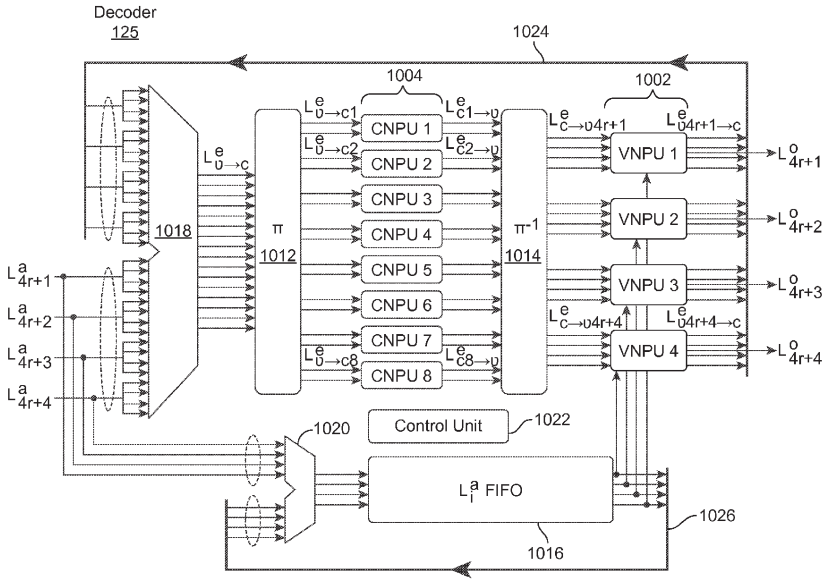
Publication Classification

(51) **Int. Cl.**
H03M 13/05 (2006.01)
G06F 11/10 (2006.01)

(52) **U.S. Cl.** **714/752**; 714/E11.032

(57) **ABSTRACT**

A decoder performs forward error correction based on quasi-cyclic regular column-partition low density parity check codes. A method for designing the parity check matrix reduces the number of short-cycles of the matrix to increase performance. An adaptive quantization post-processing technique further improves performance by eliminating error floors associated with the decoding. A parallel decoder architecture performs iterative decoding using a parallel pipelined architecture.



[Copia de la patente de USA, pg.2/28]

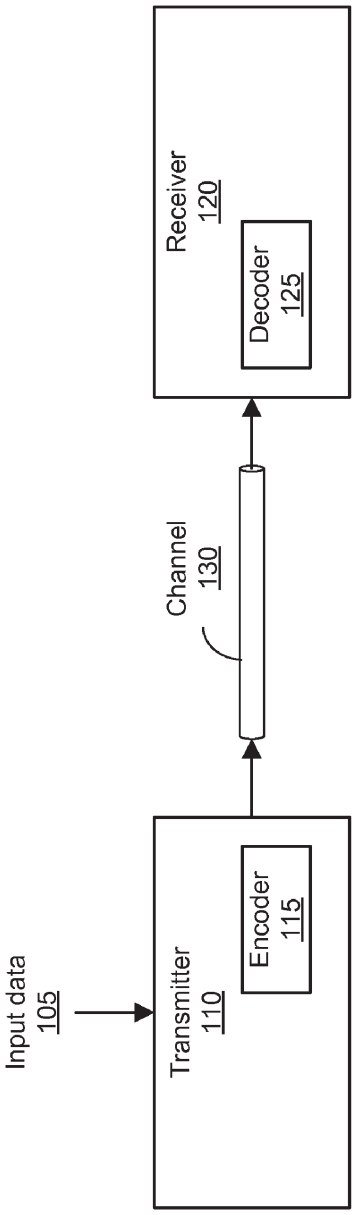


FIG. 1

[Copia de la patente de USA, pg.3/28]

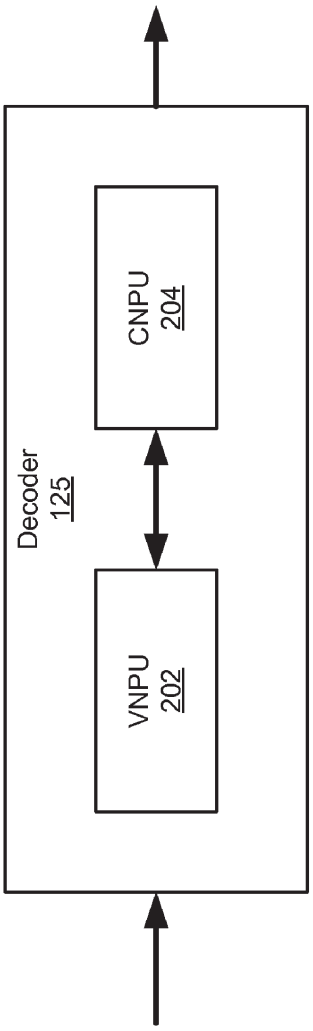
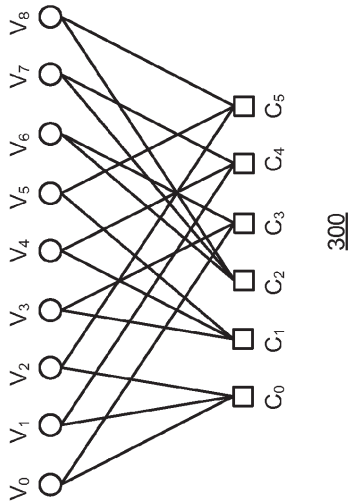


FIG. 2

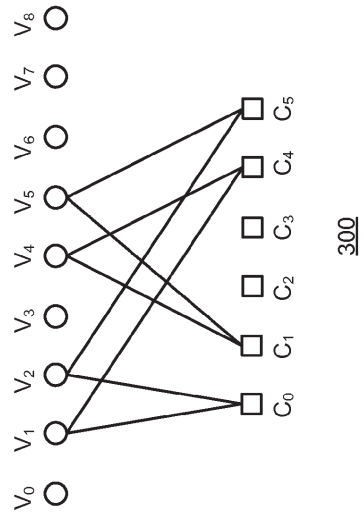
[Copia de la patente de USA, pg.4/28]



$$H = \begin{array}{c} \\ V_8 \\ V_7 \\ V_6 \\ V_5 \\ V_4 \\ V_3 \\ V_2 \\ V_1 \\ V_0 \end{array} \begin{array}{c} \begin{array}{cccccc} C_0 & C_1 & C_2 & C_3 & C_4 & C_5 \end{array} \\ \left[\begin{array}{cccccc} 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{array}\right] \end{array}$$

FIG. 3

[Copia de la patente de USA, pg.5/28]



$$H = \begin{array}{c} \\ V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \\ V_8 \end{array} \begin{bmatrix} C_0 & C_1 & C_2 & C_3 & C_4 & C_5 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

FIG. 4

[Copia de la patente de USA, pg.6/28]

$$H = \left[\begin{array}{ccc|ccc|ccc|ccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

$$H_g = \left[\begin{array}{ccc|ccc|ccc|ccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{array}\right]$$

$$H_i = \begin{bmatrix} \{0,1\} & \{0\} & \{\} & \{1\} \\ \{1\} & \{1\} & \{0\} & \{2\} \\ \{\} & \{0\} & \{0, 2\} & \{0\} \end{bmatrix}$$

FIG. 5

[Copia de la patente de USA, pg.7/28]

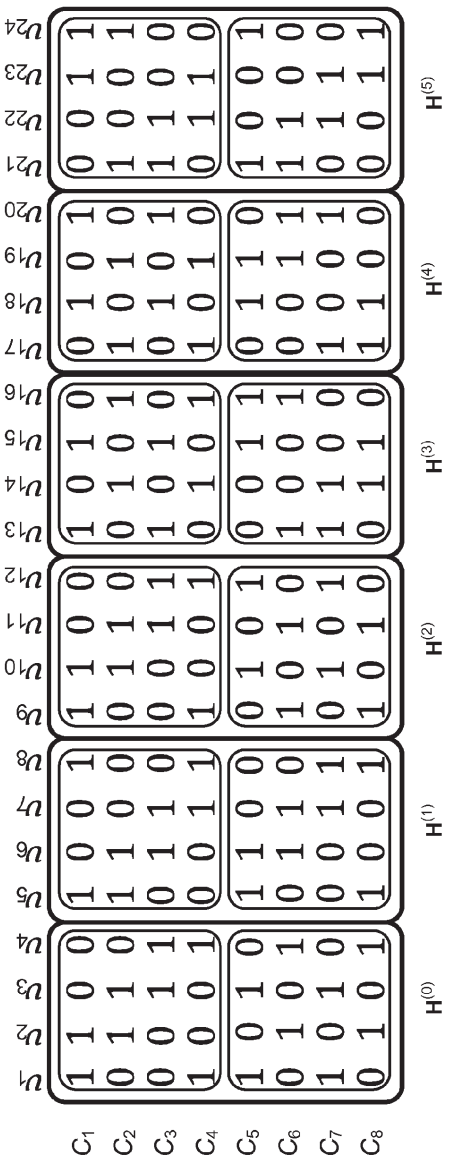


FIG. 6

[Copia de la patente de USA, pg.8/28]

$$H_i^T = \begin{bmatrix} \{0, 229\} & \{236, 1584\} \\ \{0, 764\} & \{971, 1613\} \\ \{0, 1462\} & \{402, 727\} \\ \{0, 1443\} & \{1464, 1721\} \\ \{0, 83\} & \{892, 1862\} \\ \{0, 1203\} & \{915, 1258\} \\ \{0\ 1921\} & \{970, 1340\} \\ \{0, 1686\} & \{1152, 1931\} \\ \{0, 1841\} & \{819, 1190\} \\ \{0, 642\} & \{625, 1466\} \\ \{0, 1779\} & \{0, 372\} \\ \{0, 1195\} & \{511, 580\} \end{bmatrix}$$

FIG. 7A

$$H_i^T = \begin{bmatrix} \{0, 272\} & \{1813, 506\} \\ \{0, 242\} & \{1222, 1598\} \\ \{0, 154\} & \{585, 947\} \\ \{0, 669\} & \{1646, 92\} \\ \{0, 558\} & \{966, 1085\} \\ \{0, 285\} & \{1322, 1904\} \\ \{0\ 644\} & \{535, 1894\} \\ \{0, 1714\} & \{198, 1782\} \\ \{0, 1887\} & \{1615, 2042\} \\ \{0, 20\} & \{358, 834\} \\ \{0, 176\} & \{951, 1682\} \\ \{0, 479\} & \{84, 557\} \end{bmatrix}$$

FIG. 7B

[Copia de la patente de USA, pg.9/28]

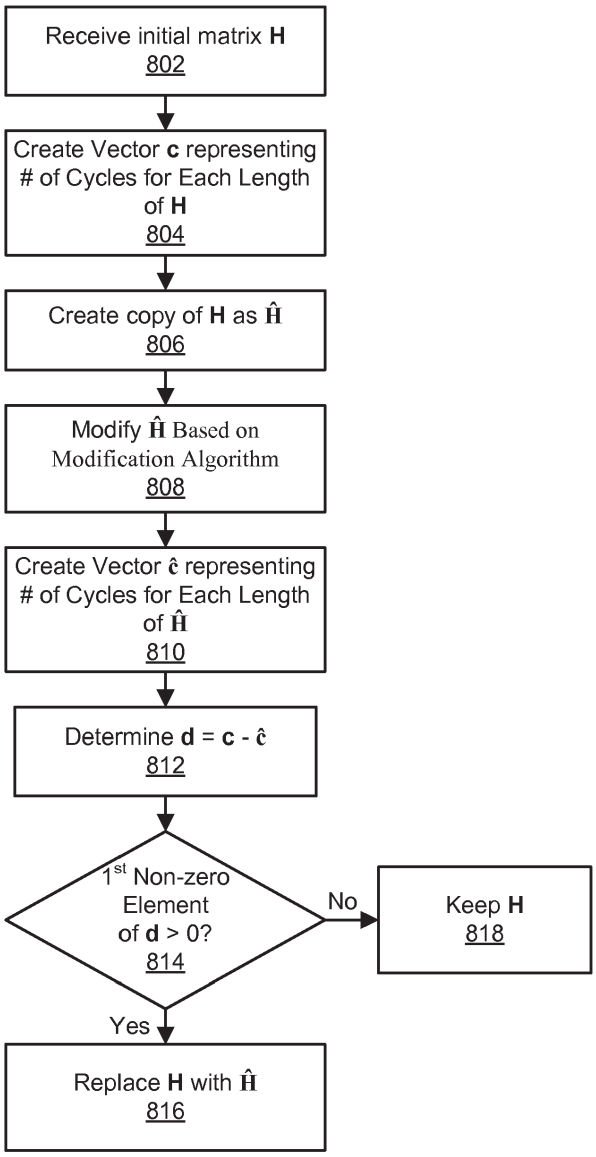


FIG. 8

[Copia de la patente de USA, pg.10/28]

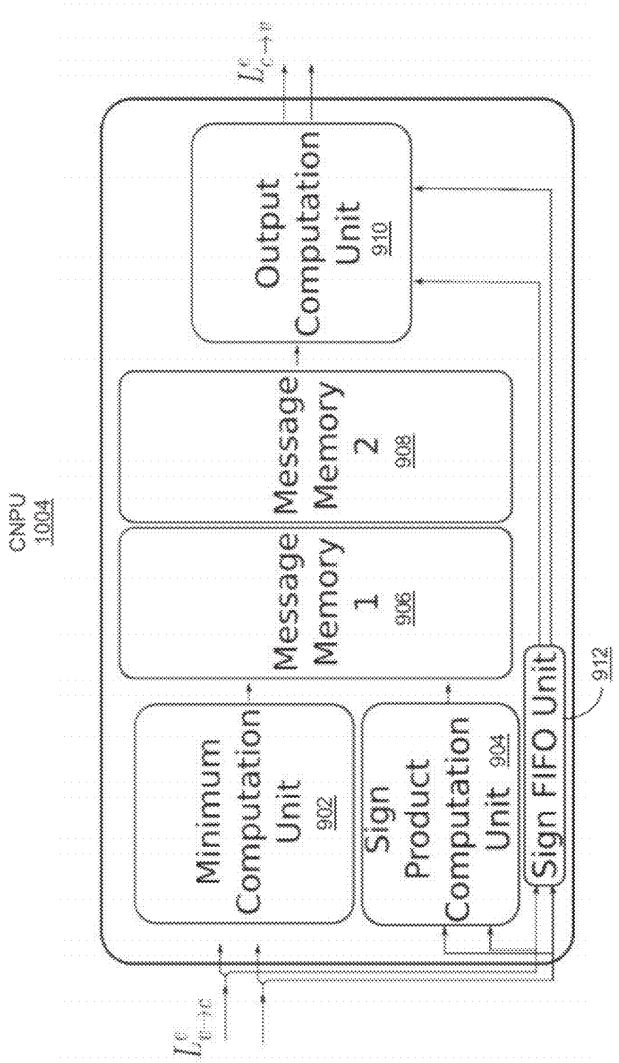


FIG. 9

[Copia de la patente de USA, pg.11/28]

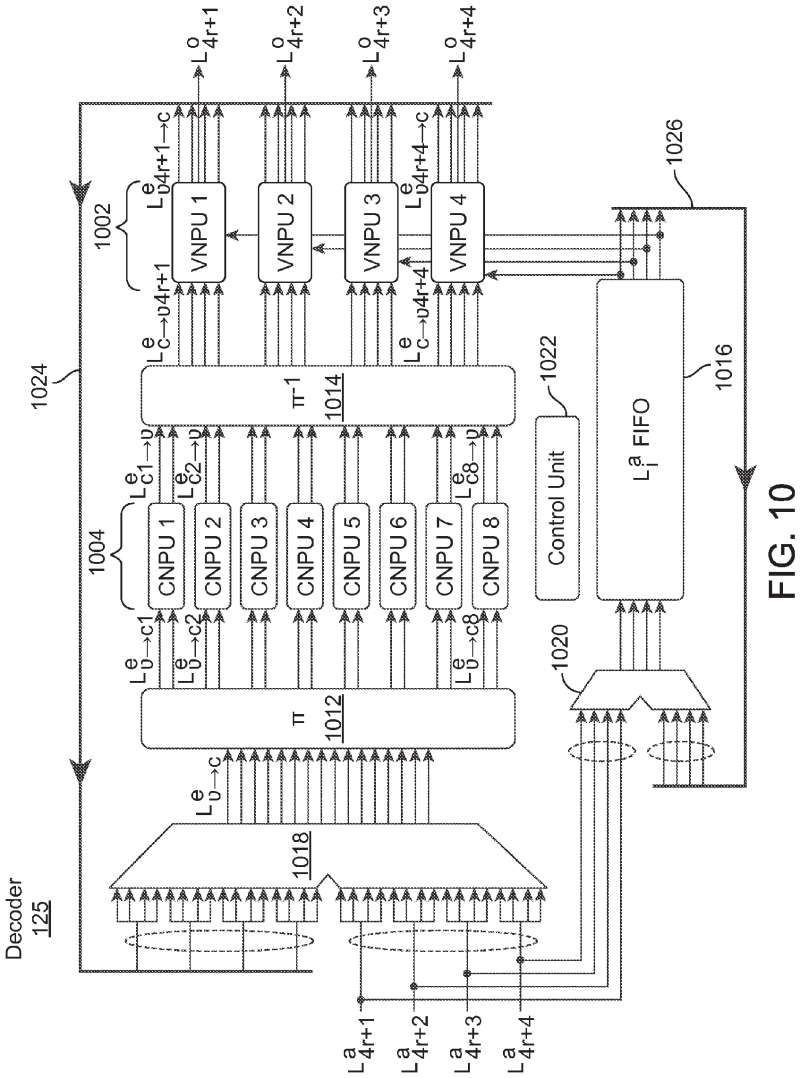


FIG. 10

[Copia de la patente de USA, pg.12/28]

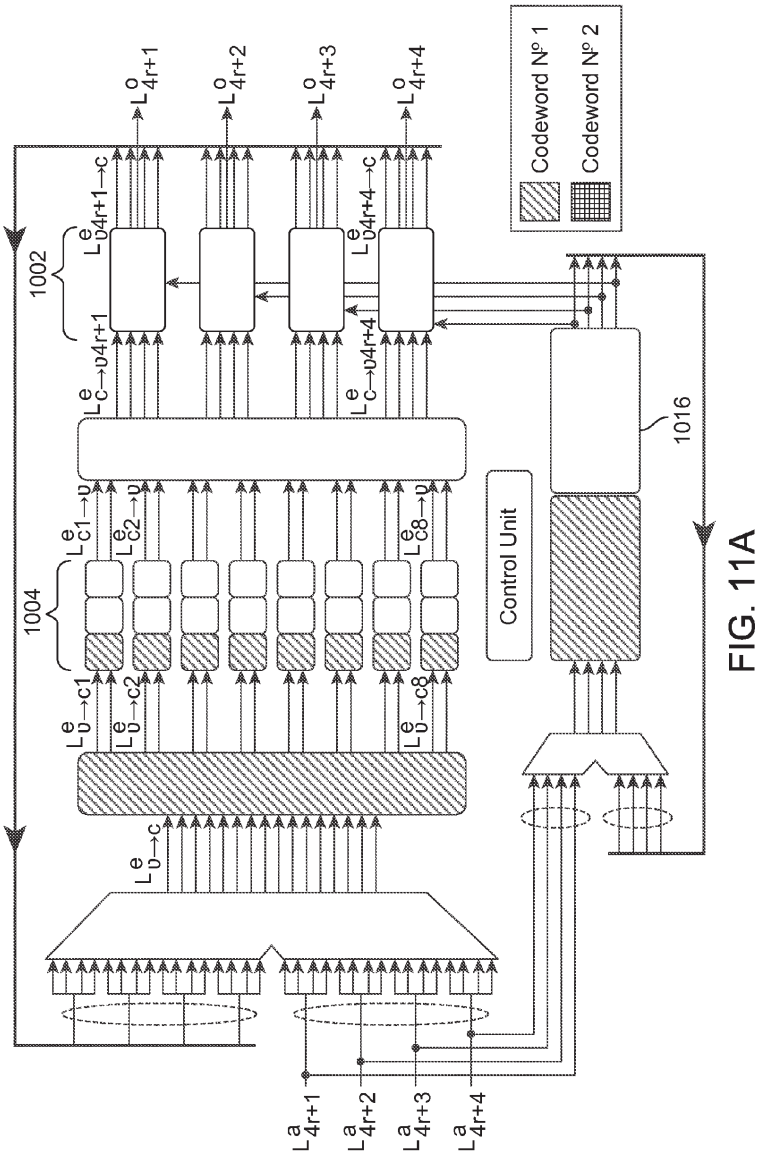


FIG. 11A

[Copia de la patente de USA, pg.13/28]

Patent Application Publication Aug. 30, 2012 Sheet 12 of 17 US 2012/0221914 A1

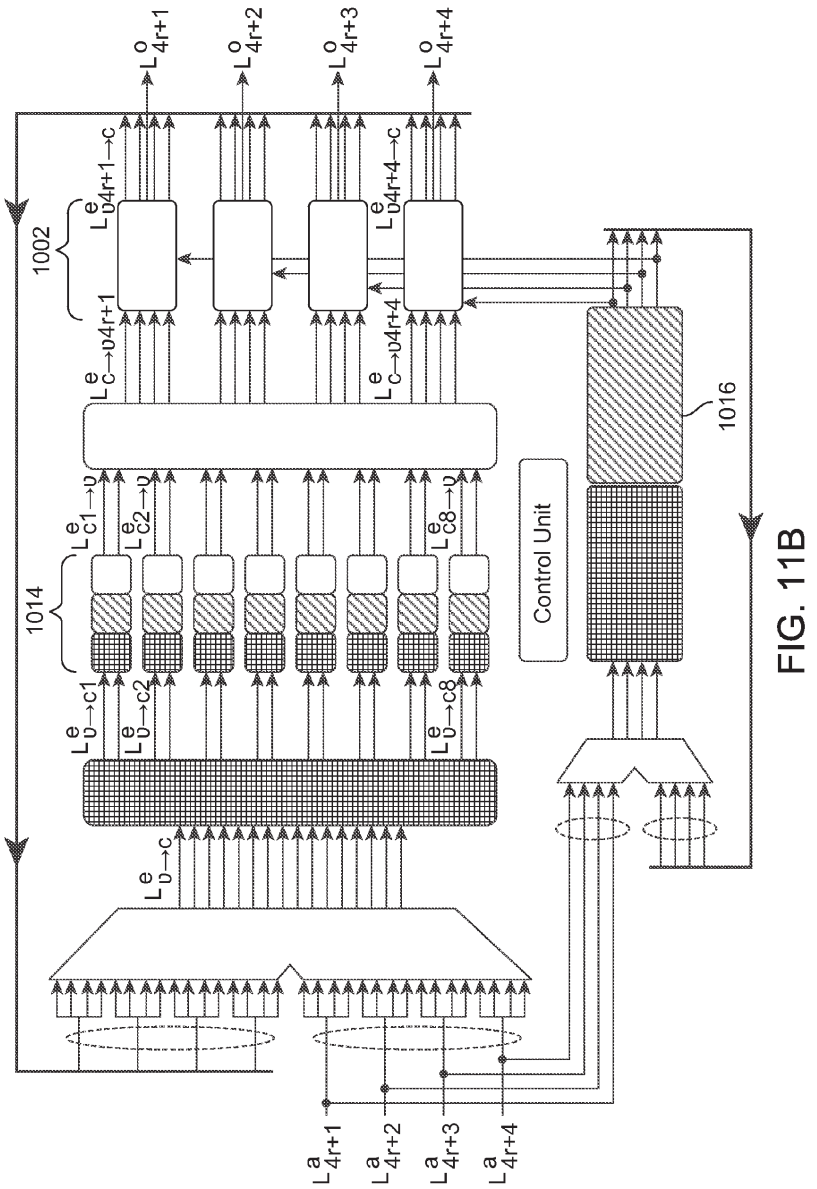


FIG. 11B

[Copia de la patente de USA, pg.14/28]

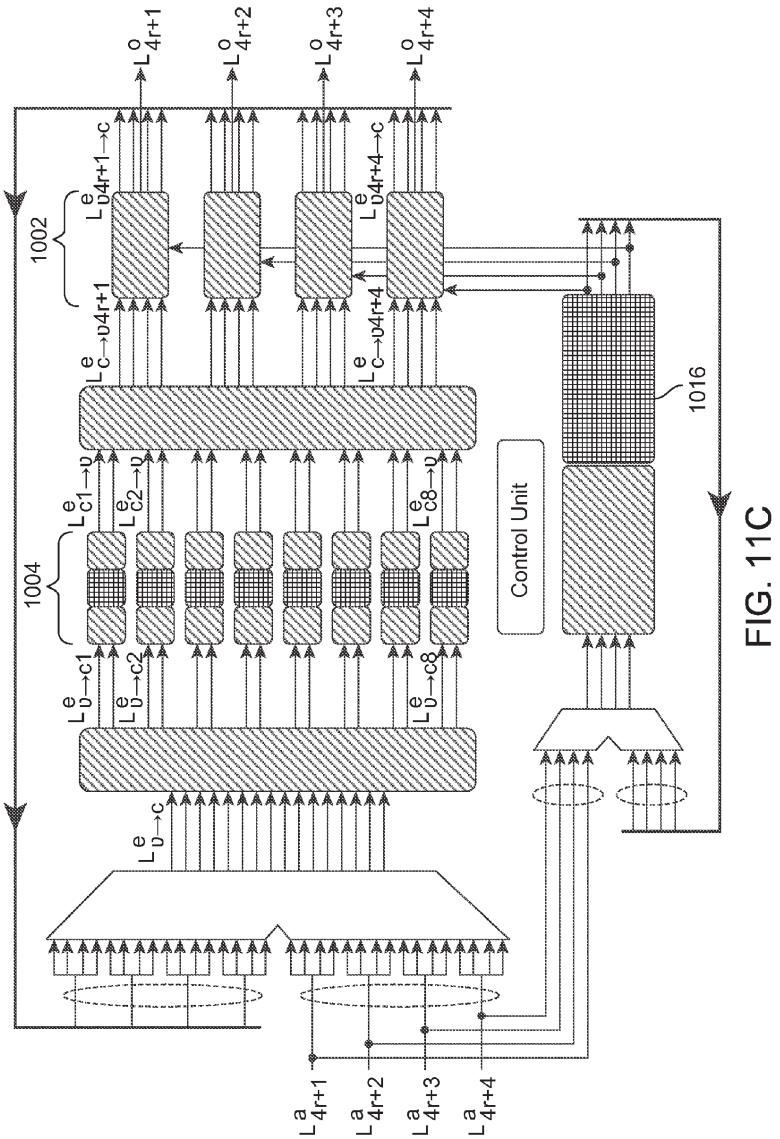


FIG. 11C

[Copia de la patente de USA, pg.15/28]

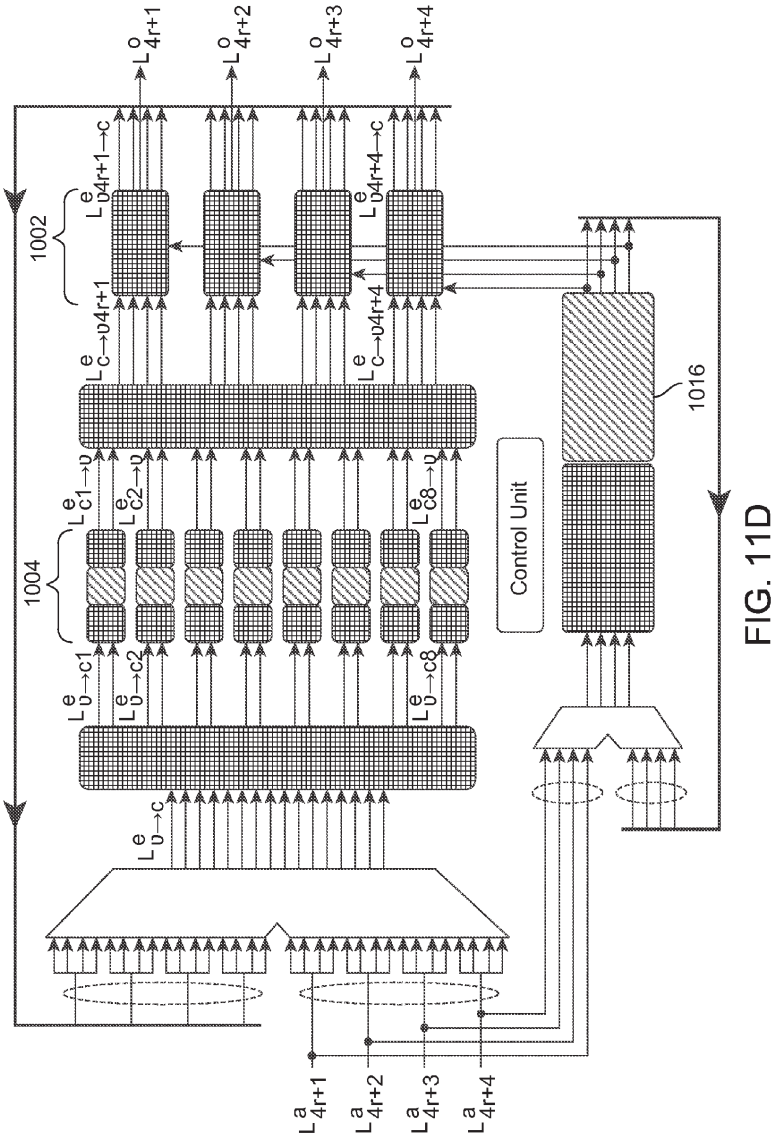


FIG. 11D

[Copia de la patente de USA, pg.16/28]

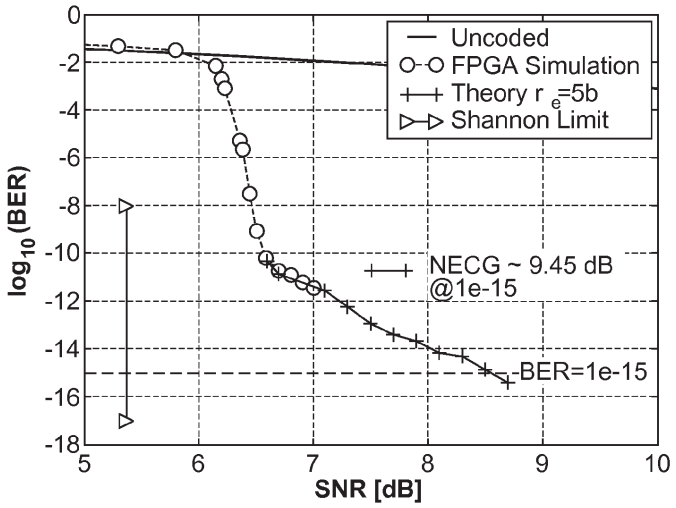


FIG. 12A

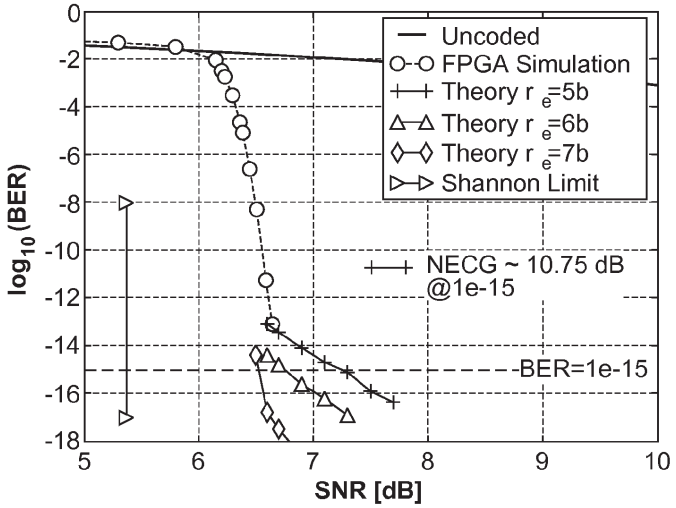


FIG. 12B

[Copia de la patente de USA, pg.17/28]

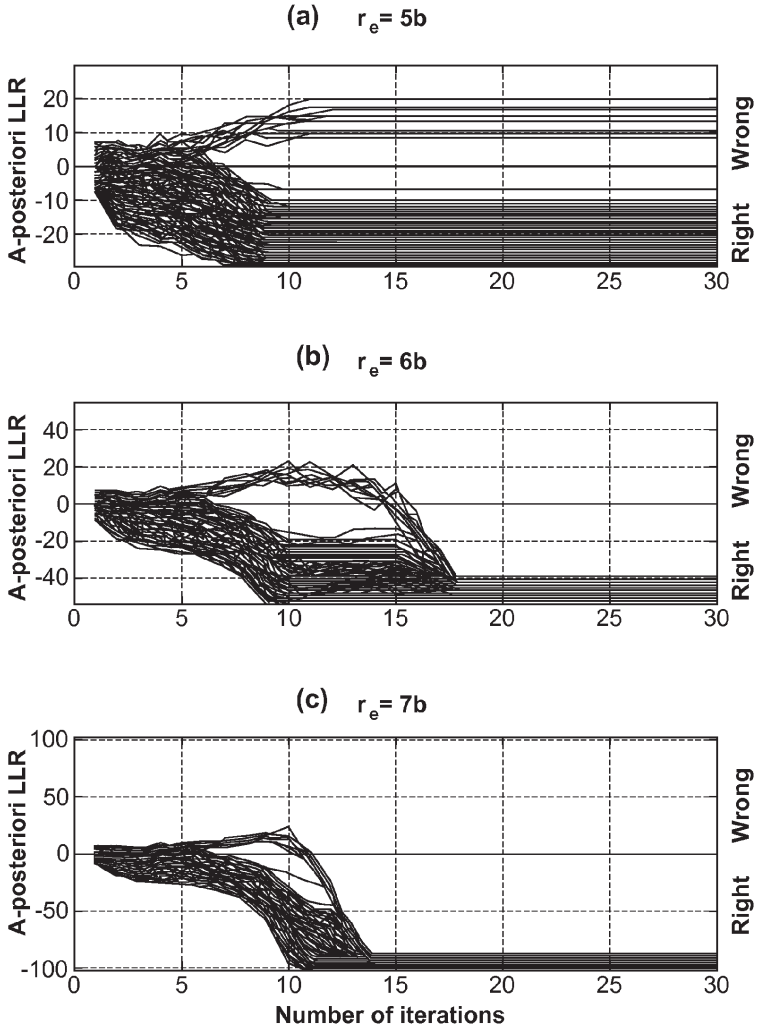


FIG. 12C

[Copia de la patente de USA, pg.18/28]

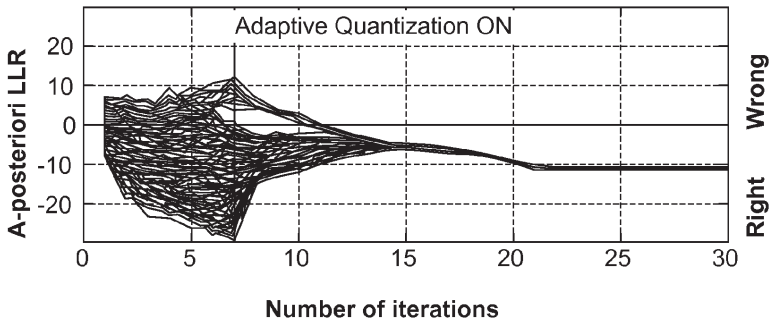


FIG. 13A

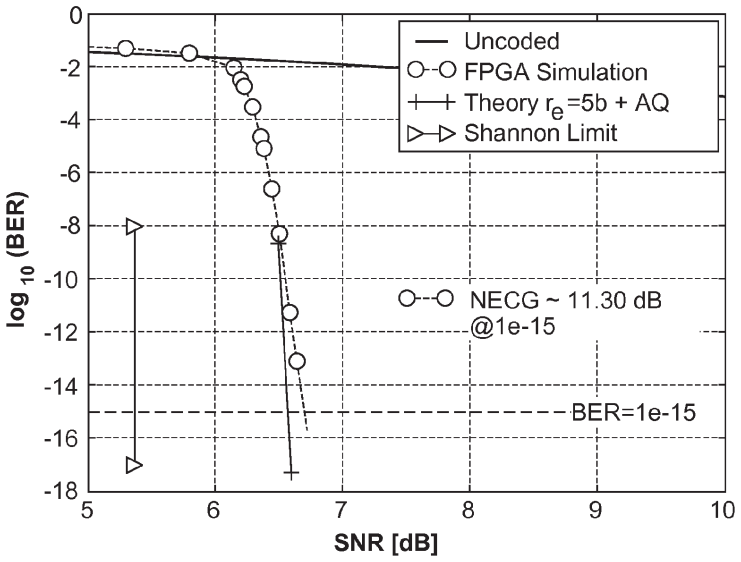


FIG. 13B

[Copia de la patente de USA, pg.19/28]

NON-CONCATENATED FEC CODES FOR ULTRA-HIGH SPEED OPTICAL TRANSPORT NETWORKS

RELATED APPLICATIONS

[0001] This application claims priority under 35 U.S.C. §119(e) to U.S. Provisional Patent Application Ser. No. 61/447,620 entitled "Non-Concatenated FEC Codes for Ultra-High Speed Optical Transport Networks," filed Feb. 28, 2011 by Damian Morero, et al., the content of which is incorporated by reference herein.

BACKGROUND

[0002] 1. Field of the Art

[0003] The disclosure relates generally to communication systems, and more specifically, to forward error correction codes.

[0004] 2. Description of the Related Art

[0005] Error correction is important component of applications such as optical transport networks and magnetic recording devices. For example, in the next generation of coherent optical communication systems, powerful forward error correction (FEC) codes are desirable to achieve high net effective coding gain (NECG) (e.g., ≧10 dB at a bit error rate (BER) of 10^{15}). Given their performance and suitability for parallel processing, large block size low density parity check (LDPC) codes are a promising solution for ultra-high speed optical fiber communication systems. Because of the large block size to achieve high NECG, the use of low complexity soft decoding techniques such as the min-sum algorithm (MSA) is often used when aiming at an efficient very large scale integration (VLSI) implementation. The main stumbling block for the application of this coding approach has been the fact that traditional LDPC codes suffer from BER error floors that are undesirably high.

[0006] The error floor is a phenomenon encountered in traditional implementations of iterated sparse graph-based error correcting codes like LDPC codes and Turbo Codes (TC). When the bit error ratio (BER) curve is plotted for conventional codes like Reed Solomon codes under algebraic decoding or for convolutional codes under Viterbi decoding, the curve steadily decreases as the Signal to Noise Ratio (SNR) condition becomes better. For LDPC codes and TC, however, there is a point after which the curve does not fall as quickly as before. In other words, there is a region in which performance flattens. This region is called the error floor region.

[0007] To reduce these error floors, some decoders concatenate an LDPC code with a hard-decision-based block code. However, this approach increases the overhead and reduces the performance and the spectral efficiency.

SUMMARY

[0008] In a first embodiment, a decoder is provided for forward error correction using an LDPC code based on a parity check matrix comprising a plurality of sub-matrices. The decoder comprises a plurality of check node processing units (CNPU) and a plurality of variable node processing units (VNPU). Each check node processing unit performs a check node computation corresponding to a different row of the parity check matrix. Each variable node processing unit determines variable node update computations corresponding to a different columns belonging to a same sub-matrix of the parity matrix. The plurality of check node processing units and the plurality of variable node processing units operate on only one sub-matrix of the parity check matrix at each step of an iterative decoding process. The decoder processes two or more codewords in parallel such that decoder begins decoding a subsequently received codeword prior to completing decoding for a prior received codeword. The decoder architecture beneficially improves the decoding process by, for example, avoiding the penalty introduced by latency of the different blocks inside the decoder

[0009] In a second embodiment, a method for forward error correction is provided. A decoder receives a stream of low density parity check codewords. The decoder iteratively decodes the low density parity check codewords based on a parity check matrix. For each iteration of the decoding, it is determined if an activation criteria is met. Responsive to the activation criteria being met for an iteration of the decoding, the decoder is configured to adaptively quantize messages processed by the decoder based on a scaling factor. This post-processing method based on adaptive quantization beneficially improves the performance of low density parity check decoding algorithms by contributing to reduction or elimination of the error floor.

[0010] In a third embodiment, a method is provided for generating a quasi-cyclic regular column-partition parity check matrix for forward error correction. An initial matrix H is received. A count of cycles of the initial matrix H is determined for each of a plurality of different cycle lengths. A matrix $\hat{H}$ is created as copy of the initial matrix H. The matrix $\hat{H}$ is modified based on a modification algorithm. A count of cycles of the modified matrix $\hat{H}$ is determined for each of the plurality of different cycle lengths. A lowest of the plurality of different cycle lengths is determined for which the initial matrix H and the modified matrix $\hat{H}$ have different counts. The initial matrix H is replaced with the modified matrix $\hat{H}$ responsive to the initial matrix H having a higher count for the lowest of the plurality of different cycle lengths for which the initial matrix H and the modified matrix $\hat{H}$ have different counts.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The invention has other advantages and features which will be more readily apparent from the following detailed description of the invention and the appended claims, when taken in conjunction with the accompanying drawings, in which:

[0012] Figure (FIG. **1**) is a high level block diagram of an embodiment of a communication system.

[0013] FIG. **2** is a high level block diagram of an embodiment of a decoder.

[0014] FIG. **3** is an example embodiment of a parity check matrix and Tanner graph for forward error correction.

[0015] FIG. **4** is an example embodiment of a cycle of a parity check matrix and Tanner graph for forward error correction.

[0016] FIG. **5** is an example embodiment of various notations for a parity check matrix for forward error correction.

[0017] FIG. **6** is an example embodiment of a parity check matrix for forward error correction.

[0018] FIG. **7**A is a first example embodiment of a parity check matrix for forward error correction using a short notation.

[Copia de la patente de USA, pg.20/28]

[0019] FIG. **7**B is a second example embodiment of a parity check matrix for forward error correction using a short notation.

[0020] FIG. **8** is an example embodiment of process for determining reducing cycle length in a parity check matrix for forward error correction.

[0021] FIG. **9** is an example embodiment of a check node processing unit for decoding LDPC codes.

[0022] FIG. **10** is an example embodiment of a decoder architecture for decoding LDPC codes.

[0023] FIG. **11**A is an example embodiment of a first step of information flow through a decoder architecture for decoding LDPC codes.

[0024] FIG. **11**B is an example embodiment of a second step of information flow through a decoder architecture for decoding LDPC codes.

[0025] FIG. **11**C is an example embodiment of a third step of information flow through a decoder architecture for decoding LDPC codes.

[0026] FIG. **11**D is an example embodiment of a fourth step of information flow through a decoder architecture for decoding LDPC codes.

[0027] FIG. **12**A is a first performance graph for a set of example LDPC codes.

[0028] FIG. **12**B is a second performance graph for a set of example LDPC codes.

[0029] FIG. **12**C is a third performance graph for a set of example LDPC codes.

[0030] FIG. **13**A is a first performance graph for a set of example LDPC codes using adaptive quantization decoding.

[0031] FIG. **13**B is a second performance graph for a set of example LDPC codes using adaptive quantization decoding.

DETAILED DESCRIPTION

Overview

[0032] A system operates using non-concatenated forward error correction (FEC) codes suitable for applications such as 100 Gb/s optical transport networks (OTN) or magnetic recording (MR) apparatuses. The system operates using a high-gain, very low error floor, long LDPC-only based code suitable for ultra-high speed fiber optical communication systems. The described system can, in some embodiments, achieve a net effective coding gain (NECG)>10 dB or better at a bit error rate (BER) of 10^{15} with an overhead (OH) of ~20% or better. Relative to prior systems that use concatenated codes, the non-concatenated codes described herein achieve superior performance, lower latency, and lower overhead than the concatenated codes.

[0033] To overcome potential performance issues due to BER floors, a low density parity check (LDPC) code is described. Semi-analytic techniques are combined with a study of dominant error events in order to determine a parity check matrix having high performance characteristics. A post-processing method is also described (e.g., an adaptive quantization (AQ) post-processing method) that can effectively eliminate the BER floors once they are pushed to sufficiently low levels in the parity check matrix design. An implementation of a decoder is also described for use with the LDPC codes and post-processing method.

[0034] The LDPC code and the hardware architecture of the decoder are jointly designed in order to (i) minimize (or sufficiently lower) the error floor and (ii) reduce the amount of memory and interconnection complexity. For example, in one embodiment, a (24576, 20482) (i.e., 20% OH) QC-LDPC-only code with a 5-bit MSA decoder is used for 100 Gb/s optical systems. Under certain conditions, the described LDPC-only code can achieve an NECG of, for example, 10.70 dB at BER of 10^{-13} with a 13-iteration MSA decoder, while being free of error floors down

System Architecture

[0035] FIG. **1** is a block diagram of a communication system **100**. The communication system **100** comprises a transmitter **110** for transmitting data to a receiver **120** via a communication channel **130**. The transmitter **110**, receiver **120**, and communication channel **130** may be of various types, depending on the end application for the communications system **100**. For example, in one embodiment, the communication system **100** comprises an ultra-high speed (e.g., 100 Gb/s or faster) optical fiber communication system. In alternative embodiments, the communication system **100** may comprise, for example, a microwave, radio frequency (RF), cable, or other type of communication system.

[0036] The communication channel **130** may be unreliable or noisy. Thus, the data received by the receiver **120** often contains errors (e.g., bit flips) relative to the transmitted data. The transmitter **110** and receiver **120** therefore utilize an error correction technique that enables the receiver **120** to detect, and in many cases, correct errors in the data received over the channel **130** from the transmitter **110**.

[0037] The transmitter **110** receives input data **105** for transmission to the receiver **120** via the communication channel **130**. The transmitter **110** includes an encoder **115** that encodes the data using forward error-correcting (FEC) codes. In one embodiment, a block coding scheme is used in which each block of binary input data is mapped to an FEC codeword. Generally, the FEC code provides some redundancy in the data by incorporating extra data symbols. For example, in one embodiment, the encoder applies a transform function to an input data block having k symbols to generate an FEC code having n symbols, where n>k. This redundancy allows the receiver **120** to detect a limited number of errors that may occur in the transmitted data and in many cases to correct such errors. More specific details about the FEC codes are provided below.

[0038] In addition to the encoder **115**, the transmitter **110** may comprise other conventional features of a transmitter **110** which are omitted from FIG. **1** for clarity of description. For example, the transmitter may include components such as a modulator, a serial or parallel/serial converter, a driver or amplifier circuit, a laser source, etc.

[0039] The receiver **120** receives the data encoded as FEC codes from the transmitter **110** via the communication channel **120**. The receiver **120** includes a decoder **125** that decodes the FEC codes data to attempt to recover the original data blocks. For example, in one embodiment, the decoder **125** applies a parity check matrix H to a received FEC codeword having n symbols to recover a data block having k symbols where n>k. More specific details on the decoding technique is provided below.

[0040] In addition to the decoder **125**, the receiver **120** may comprise other conventional features of a receiver **120** which are omitted from FIG.**1** for clarity of description. For example, the receiver **120** may include components such as a demodulator, an analog-digital converter, amplifier circuits, timing recovery circuits, an equalizer, various filters, etc.

[Copia de la patente de USA, pg.21/28]

[0041] Components of the transmitter **110** and the receiver **120** described herein may be implemented, for example, as an integrated circuit (e.g., an Application-Specific Integrated Circuit (ASIC) or using a field-programmable gate array (FPGA), in software (e.g., loading program instructions to a processor from a computer-readable storage medium and executing the instructions by the processor), or by a combination of hardware and software.

[0042] FIG. **2** illustrates an example embodiment of a decoder **125**. In this embodiment, the decoder **125** iteratively decodes the received codewords using a decoding algorithm such as, for example, the sum-product algorithm (SPA), the min-sum algorithm (MSA), or the scaled min-sum algorithm (SMSA). In one embodiment, the decoder **125** comprises a variable-node processing unit (VNPU) **202**, and a check-node processing unit (CNPU) **204**. The VNPU **202**, and/or CNPU **204** may each comprise a plurality of parallel processing units (e.g., q processing units). This allows for an efficient parallel decoding process as will be described in further detail below. More specific examples of architectures for the decoder **125** are described in FIGS. **9-11**.

General LDPC Codes

[0043] In one embodiment, the communications system **100** uses low density parity check (LDPC) codes for forward error correction. An LDPC code is a linear block code defined as the null space of a sparse (m×n) parity check matrix H, where n represents the number of bits in the block and m denotes the number of parity checks. The matrix H is considered "sparse" because the number of 1s is small compared with the number of 0s. Using the above definition, the set of LDPC codes $\mathbb{C}$ is defined as:

$$\mathbb{C}=\{c:Hc=0\} \quad (1)$$

where c is an LDPC codeword in the set $\mathbb{C}$. Note that each row of H provides a parity check on the codewords. Particularly, each row indicates that a linear combination of certain bits (specified by the 1s in the row) will add to zero for a valid codeword. Furthermore, an invalid codeword can often be corrected by comparing results of multiple parity checks and determining a most likely location of the error(s).

[0044] Matrix H can be graphically represented using a Tanner graph **300** as illustrated in FIG. **3** for an example matrix H. The Tanner graph **300** is a bipartite graph composed of two types of nodes: (1) variable or bit nodes v_i which represent the columns of H; and (2) the check nodes c_j which represent the rows of H. A connection between nodes v_i and c_j exists in the Tanner graph **300** if and only if $H_{j,i}=1$. Note that there are not connections between two check nodes or between two bit nodes.

[0045] LDPC codes can be classified as "regular" or "irregular" based on characteristics of the matrix H. A matrix H is regular if it is both row-regular and column-regular. Otherwise, the matrix H is irregular. Matrix H is row-regular if $\rho_i=\rho$ for all i where ρ_i is the number of 1s in the i^{th} row of H. In other words, the matrix H is row-regular if all rows have the same number of 1s. Similarly, Matrix H is column-regular if $\gamma_i=\gamma$ for all i where γ_i is the number of 1s in the i^{th} column of H. In other words, the matrix H is column-regular if all columns have the same number of 1s.

[0046] For a given variable node v_i or check node c_j, the number of connections to it determines its degree. If all v_i nodes have the same degree γ and all c_j nodes the same degree ρ, then the LDPC code is said to be a (γ, ρ)-regular LDPC.

[0047] A "cycle" in the Tanner graph **300** for a matrix H is a closed sequence (e.g., a loop) of connected nodes. FIG. **4** illustrates an example of a cycle having a length 8. As can be seen, the series of connections forms a closed loop with a total of 8 links. Note that a Tanner graph for a matrix H may have multiple cycles. The "girth" of the Tanner graph for a matrix H is the length of the shortest cycle.

Quasi-Cyclic LDPC Codes

[0048] A cyclic matrix or "circulant" is a square matrix in which each row is the cyclic shift of the row above it (i.e., the symbols in the row are right-shifted by one relative to the row immediately above it with the last symbol in the row shifted to the first position), and the first row is the cyclic shift of the last row. Furthermore, each column is the downward cyclic shift of the column on its left (i.e., the symbols in the column are down-shifted by one relative to the column immediate to the left of it, with the last symbol in the column shifted to the first position), and the first column is the cyclic shift of the last column.

[0049] A characteristic of a circulant is that the row and column weights w are the same, where the weight w of a row or column represents the number of 1s in the row or column. Note that due to the characteristics of the circulant, the row and column weights also give the number of non-zero diagonals in the matrix. The weights w of the rows and columns of the circulant can also generally be referred to as the weight of the circulant. Note that if w=1, then the circulant is a permutation matrix, referred to as a circulant permutation matrix.

[0050] Another characteristic of a circulant is that the circulant can be completely characterized by its first row (or first column). In other words, if the first row (or column) of the matrix is known, the rest of the matrix can be generated by applying appropriate shifts to this vector based on the characteristics of the circulant defined above. Therefore, the first row (or column) is referred to herein as the "generator of the circulant."

[0051] In quasi-cyclic LDPC codes, the parity check matrix H is an array of sparse square circulant matrices of the same size. Observing that the LDPC code is given by the null-space of H, a set of quasi-cyclic LDPC codes can be defined by the null space of an array of sparse square circulant matrices of the same size. Quasi-cyclic codes represent a generalization of cyclic codes whereby a cyclic shift of a codeword by p positions results in another codeword. Therefore, cyclic codes are simply QC codes with p=1. QC-LDPC codes can beneficially perform very close to the Shannon limit and their cyclic properties reduce the implementation complexity, and allow the use of efficient algebraic techniques to compute the code parameters and optimize the performance.

[0052] An example of a parity check matrix H for QC-LDPC codes is illustrated in FIG. **5** according to various notations. In this example, matrix H is a 3×4 array of 3×3 circulants having varying weights of 0, 1, and 2. As explained above, the circulants are completely defined by the generator (first row) of each circulant. Therefore, the same matrix H can be represented in a short notation given by H_g based on the generator. An even more compact notation for representing the matrix H is shown in matrix H_t. In this notation, each circulant is represented by a vector defining the non-zero

[Copia de la patente de USA, pg.22/28]

column positions of the generator of each circulant. As can be seen, the compact notation in H_t completely defines the matrix H.

Regular Column Partition (RCP) QC-LDPC Codes

[0053] A regular column partition QC-LDPC (RCP-QC-LDPC) code is an LDPC code that meets both the column-regular constraint and the quasi-cyclic constraint described above. Let H be the (m×n) parity check matrix of an LDPC code. Assuming that n=μq with μ and q integers, the matrix H can be partitioned into μ(m×q) sub-matrices:

$$H=[H^{(0)}\ldots H^{(r)}\ldots H^{(\mu-1)}]. \quad (2)$$

[0054] The parity check matrix H has the characteristic that the weights of the rows and columns of each of the sub-matrices $H^{(r)}$ do not change with r. Thus, each sub-matrix of H is regular and the matrix H itself is regular.

[0055] Furthermore, in one embodiment, a type-p RCP-QC-LDPC code is used. In this type of code, each of the circulant sub-matrices $H^{(r)}$ has the same rows weight p (i.e., the number of non-zero diagonals). For VLSI implementation, a small value of p is often desirable since the complexity increases (at least) linearly with p. A high value of p reduces the maximum girth of the code, which increases the error floor probability.

[0056] FIG. **6** illustrates an example (4, 12)-regular matrix H that is partitioned into μ=6 (4, 2)-regular sub-matrices ($H^{(1)}\ldots H^{(6)}$), where q=4. While the example in FIG. **6** is illustrative of the properties of a general parity check matrix for an for RCP-QC-LDPC codes, a parity check matrix H may in practice be composed of significantly larger sub-matrices $H^{(r)}$ and may have a larger number of sub-matrices (higher value of μ). For example, in one embodiment, a parity check matrix H for RCP-QC-LDPC codes comprises a 2×12 array of circulants each having a size 2048×2048 and a weight of 2. Thus, in this embodiment, the parity check matrix H is configured for type-2 RCP-QC-LDPC codes and has a column weight of 4. In one embodiment, the maximum girth of the parity check matrix H is eight.

[0057] FIG. **7**A illustrates an example of a transposed version of H_t representing the compact notation for matrix H corresponding to a first example set of codes C_1. This first code set, C_1, is designed to avoid cycles of length 4. FIG. **7**B illustrates an example of a transposed version of H_t representing the compact notation for matrix H corresponding to a second example set of codes C_2. This second code set, C_2, is designed to (i) avoid cycles of length 4 and 6 (i.e. it achieves the maximum girth) and (ii) minimize number of cycles of length 8. A parity check matrix H having the characteristics described above will result in a code length of 24576 symbols (e.g., bits) and k=n−rank(H)=20,482 symbols where k is the size of a decoded codeword. In one embodiment, the RCP-QC-LDPC codes having the characteristics above is able to achieve a net effective coding gain (NECG) of 10 dB or higher at a bit error rate (BER) of 10^{-15}. Furthermore, the RCP-QC-LDPC codes have an overhead of about 20%.

[0058] In one embodiment, a variable-node partition with full check-node connectivity (VPFCC) constraint in also imposed on the parity check matrix H. In this embodiment, all of the sub-matrices $H^{(r)}$ have only a single 1 in each of its rows.

Reducing the Number of Short-Cycles of the Parity Check Matrix

[0059] In order to choose a specific parity check matrix H and an associated set of RCP-QC-LDPC codes C, a technique may be used to find a quasi-cyclic parity check matrix having a low number of short-cycles. An example embodiment of a process for reducing the number of short-cycles is illustrated in FIG. **8**. The process can begin by receiving **802** an initial parity check matrix H meeting RCP and quasi-cyclic constraints described above. A vector c is created **804** representing the number of cycles of H of different lengths in order of increasing cycle length. For example, the number of cycles of H with lengths 4, 6, 8, . . . and so on are computed to create a histogram with each bin corresponding to a different cycle length. This computation can be implemented in several ways. For example, in one embodiment, a matrix A is denoted as the adjacency matrix of the Tanner Graph of H. It can be shown that the (i, j)-entry of A^l equals the number of paths of length l from node-i to node-j (Theorem 1). Furthermore, it can be shown that in a graph with girth δ, the nodes i and j are directly opposite each other in a δ-loop if and only if $A_{i,j}^{\delta/2}\geqq 2$ and $A_{i,j}^{\delta/2-2}=0$ (Theorem 2). Then, because the Tanner Graph of H is bipartite it contains only even-length (δ even) cycles. Therefore, there are δ ordered pairs of opposite nodes in a δ-cycle, i.e. there are entries on $A_{i,j}^{\delta/2}$ that verified the constraints above and represent the same cycle. Note also that each non-ordered pair of the $A_{i,j}^{\delta/2}$ paths connecting i and j create a different loop, i.e. there are $\Phi(i,j)=A_{i,j}^{\delta/2}(A_{i,j}^{\delta/2}-1)/2$ different loops that contain nodes i and j as opposed nodes. Therefore, the number N of minimum length cycles is:

$$N=\frac{1}{\delta}\Sigma_{i,j}I(i,j)\cdot\Phi(i,j) \quad (3)$$

[0060] where $I(i,j)\in\{0,1\}$ is the indicator function which takes the value 1 if Theorem 2 is verified for that particular entry of the adjacency matrix or 0 elsewhere. In one embodiment, to speed up the computation of N in Eq. (3), the polynomial representation of H over the ring $Z[x]/(x^L-1)$ may be used. It also possible to modify Eq. (3) in order to increase the penalty of the Φ(i,j) interconnected cycles. This may be done, for example, by adding an exponential coefficient as $[\Phi(i,j)]^w$ with w>1 or in general by replacing (i, j) by $f(\Phi(i,j))$ for some non-decreasing function $f(\cdot)$. Since absorbing sets are usually created by the interconnection of several short length cycles, this variation may help to further reduce the probability of an error floor.

[0061] A copy of H is then created **806** and denoted $\hat{H}$. $\hat{H}$ is then modified **808** according to a modification algorithm while maintaining the same quasi-cyclic constraint. For example, in one embodiment, one of the cyclic sub-matrices of $\hat{H}$ is chosen based on a pseudo-random algorithm and the position of one of its diagonals is changed to a different location which is also chosen based on a pseudo-random algorithm. This step **808** may be repeated several times before continuing to the following step. This technique results in a random walk over the parameters of a quasi-cyclic parity check matrix. In alternative embodiments, a different technique could be used to modify $\hat{H}$.

[0062] A vector $\hat{c}$ is then created **810** representing the number of cycles of $\hat{H}$ of each length in order of increasing cycle length. For example, the number of cycles of $\hat{H}$ with lengths 4, 6, 8, . . . and so on are computed to create a histogram with each bin corresponding to a different cycle length. This computation can performed using a similar algorithm as described above. A vector is computed **812** as $d=c-\hat{c}$. If at decision block **814**, the first non-zero element in d is positive, then H is

[Copia de la patente de USA, pg.23/28]

replaced **816** with $\hat{H}$. Otherwise, the matrix H is kept **818**. Note that the comparison of the number of cycles between c and ĉ occur in increasing order of cycle length. Thus, for example, the cycles of length 4 are compared first; if they are equal the cycles of length 6 are compared and so on. Optionally, if further optimization is desired, the process may return to step **802** and repeat for any number of iterations (e.g., a fixed number of iterations or until a stopping criterion is met).

Iterative Decoding Algorithms

[0063] As data blocks are received by the decoder **125**, the decoder decodes the data blocks and applies the parity check matrix H to recover the transmitted data. In one embodiment, the decoder **125** may apply, for example, a sum-product algorithm (SPA), a min-sum algorithm (MSA), or a scaled min-sum algorithm (SMSA) to decode the received data blocks.

[0064] Let b_i and x_i be the i-th bit of the codeword and the corresponding channel output respectively. The input to the decoder **125** is the prior log-likelihood ratio (LLR) L^a_i defined by:

$$L_i^a = \ln\left(\frac{P_a(b_i = 0 \mid x_i)}{P_a(b_i = 1 \mid x_i)}\right), \tag{4}$$

where $P_a(\cdot)$ denotes the a-priori probability of the bit b_i. Thus, L^a_i represents an initial likelihood of the input bit i being a 0 or a 1. Then, an iterative decoding procedure between variable and check nodes is carried out as follows:

$$L^e_{v_i \to c_j} = L_i^a + \sum_{c_k \in C^{(v_i)} \setminus c_j} L^e_{c_k \to v_i}, \tag{5}$$

$$L^e_{c_j \to v_i} = \phi^{-1}\left\{ \sum_{v_k \in V^{(c_j)} \setminus v_i} \phi\left[L^e_{v_k \to c_j}\right] \right\}, \tag{6}$$

[0065] where $C^{(v_i)}=\{c_j : H_{j,i}\neq 0\}$, $V^{(c_j)}=\{v_i : H_{j,i}\neq 0\}$, $\phi(x)=\ln[\tanh(x/2)]$, and $\phi^{-1}(x)=2\tanh^{-1}(e^x)$. The posterior LLR is computed in each iteration by

$$L_i^o = L_i^a + \sum_{c_k \in C^{(v_i)}} L^e_{c_k \to v_i} \tag{7}$$

[0066] Hard decisions are derived from (7). The iterative decoding process is carried out until hard decisions satisfy all the parity check equations or when an upper limit on the iteration number is reached.

[0067] The decoding algorithm can be understood in view of the Tanner Graph (see e.g., FIG. **3**). Here, the algorithm can be represented as the passing messages between the variable nodes and the check nodes of the Tanner Graph as described in the equations above. In the equations (4)-(7), $L^e_{v_i \to c_j}$ is the extrinsic information sent by the variable node 'i' to the check node 'j'. It represents an estimation of the probability of the bit 'i' being a '0' or '1' given the a priori information L^a_i and the information coming from all other check nodes connected to the variable node 'i' except that coming from the check node 'j'. $L^e_{c_j \to v_i}$ is the extrinsic information sent by the check node 'j' to the variable node 'i'. It represents an estimation of the probability of the bit 'i' being a '0' or '1' given the information coming from all the other variable nodes connected to the check node 'j' except that coming from the variable node 'i'.

[0068] The computation of (5) and (7) are performed by the VNPU **202** of the decoder **125** and the computation of (6) is performed by the CNPU **204** of the decoder **125**. Since the CNPU **204** consumes most of the computational requirements of the above-described decoding algorithm, a simplified expression of (6) may be implemented:

$$\hat{L}^e_{c_j \to v_i} = \min_{v_k \in V^{(c_j)} \setminus v_i} \left|L^e_{v_k \to c_j}\right| \cdot \prod_{v_k \in V^{(c_j)} \setminus v_i} \operatorname{sign}\left(L^e_{v_k \to c_j}\right). \tag{8}$$

[0069] This approach is called the min-sum algorithm (MSA). To reduce the approximation error of (8), another modification can optionally be employed called the scaled min-sum algorithm (SMSA). The check node computation performed by the CNPU **204** in SMSA is given by:

$$\hat{L}^e_{c_j \to v_i} = \alpha \cdot \min_{v_k \in V^{(c_j)} \setminus v_i} \left|L^e_{v_k \to c_j}\right| \cdot \prod_{v_k \in V^{(c_j)} \setminus v_i} \operatorname{sign}\left(L^e_{v_k \to c_j}\right) \tag{9}$$

with α being a factor smaller than unity (e.g., $\alpha \approx 0.75$).

[0070] To further reduce the implementation complexity, computation of Eq. (9) can divided into a series of steps represented by equations (10A) to (10E) which are implemented by the CNPU **204**:

$$\hat{L}^e_{c_j \to v_i} = \begin{cases} \alpha \cdot M^{(1)}_{j,i} \cdot S_{j,i} \cdot \operatorname{sign}\left(L^e_{v_i \to c_j}\right) & \text{if } v_i \neq v^{(1)}_{j,i} \\ \alpha \cdot M^{(2)}_{j,i} \cdot S_{j,i} \cdot \operatorname{sign}\left(L^e_{v_i \to c_j}\right) & \text{if } v_i = v^{(1)}_{j,i} \end{cases} \tag{10A}$$

$$M^{(1)}_{j,i} = \min_{v_k \in V^{(c_j)}} \left|L^e_{v_k \to c_j}\right| \tag{10B}$$

$$M^{(2)}_{j,i} = \min_{v_k \in V^{(c_j)} \setminus v^{(1)}_{j,i}} \left|L^e_{v_k \to c_j}\right| \tag{10C}$$

$$v^{(1)}_{j,i} = \arg\left\{ \min_{v_k \in V^{(c_j)}} \left|L^e_{v_k \to c_j}\right| \right\} \tag{10D}$$

$$S_{j,i} = \prod_{v_k \in V^{(c_j)}} \operatorname{sign}\left(L^e_{v_k \to c_j}\right) \tag{10E}$$

[0071] FIG. **9** illustrates an embodiment of a CNPU **1004** for processing two codewords at the same time according to Eqs. (10A)-(10E) above. In this architecture, Eq. (10A) is computed by the output computation unit **910**, Eqs. (10B), (10C), and (10D) are computed by the Minimum Computation unit **902**, and Eq. (10E) is computed by the Sign Product Computation Unit **904**. The Message Memory **1 906** and Message Memory **2 908** save the results of equations (10B)-(10E) as described below.

[0072] The minimum computation unit **902** computes the minimum value (called the first minimum value) of the absolute value of $L_{v \to c}{}^e$ as indicated in Eq. (10B). The minimum computation unit **902** also determines which variable node corresponds to this minimum value as described in Eq. (10D).

[Copia de la patente de USA, pg.24/28]

Furthermore, the minimum computation unit **902** computes the minimum value (called the second minimum value) of the absolute values $L_{v\to c}^{e}$ but without taking into account the message coming from the variable node which corresponds to the first minimum value as described in Eq. (10C). In other words, the minimum computation unit **902** determines the two lowest absolute values of the input messages from the set of variable nodes and the variable nodes that these messages came from. The sign product computation unit **904** determines the product of the signs of $L_{v\to c}^{e}$ as indicated in Eq. (10E) above. The outputs of the minimum computation unit **902** and the sign product computation unit **904** are stored to the pipelined message memory **1 906** and message memory **2 908**. A sign FIFO unit **912** stores the signs of the input messages $L_{v\to c}^{e}$ to be used later by the output computation unit **910**. The output computation unit **910** combines the values stored in the sign FIFO unit **912** and the memory message **908** according to Eq. (10A) above and outputs the result $L_{c\to v}^{e}$. Operation of the CNPU **1004** in conjunction with a parallel decoder implementation is described in further detail below.

Parallel Implementation of Iterative Decoding Algorithm

[0073] The constraint imposed by RCP allows an efficient partial parallel implementation of the decoding algorithm. An example embodiment of a parallel pipelined decoding architecture is illustrated in FIG. **10** for the example case where q=4 as in the matrix H of FIG. **6**. The decoder **125** includes a first-in-first-out (FIFO) memory **1016** that stores the a-priori LLRs, permutation blocks Π^{-1} **1012** and Π^{-1} **1014** , parallel VNPUs **1002**, serial, parallel, or semi-parallel CNPUs **1004**, a control unit **1022**, and multiplexers **1018**, **1020**. The permutation blocks **1012**, **1014** can be implemented with multiplexers (if the permutation is not constant, i.e. the sub-matrices $H^{(r)}$ are not equal) or they can be implemented as wires. The control unit **1022** generates control signals utilized by the other blocks of the decoder **125**. In particular, the control unit **1022** controls the permutation blocks **1012**, **1014**, and turns on and off post-processing algorithms (which are implemented by the VNPUs **1002** or the CNPUs **1004**) that will be described in further detail below. The control unit **1022** also controls the computations and memories inside the CNPUs **1004** and controls the select lines of the multiplexers **1018**, **1020**.

[0074] Each iteration of the iterative decoding algorithm is divided into μ steps with each step corresponding to one of the sub-matrices of H. At the r-th step, only the messages related to the sub-matrix $H^{(r)}$ are computed. Thus, for example, at a first step (r=0), the decoder **125** receives LLRs from Eq. (4) corresponding to the first q bits (e.g., q=4) of a codeword (e.g., bits corresponding to v_1, v_2, v_3, v_4 of the first sub-matrix $H^{(0)}$). The multiplexer **1018** and permutation block **1012** operate to select the appropriate inputs to each of the CNPUs **1004** to perform the computation of Eq. (8), (9) or (10A)-(10E) (depending on the particular implementation used). In one embodiment, the permutation block **1012** comprises a barrel shifter. The CNPUs **1004** perform the check node computation of Eqs. (8), (9), or (10A)-(10E) with each CNPU **1004** corresponding to a different parity check (row of $H^{(r)}$ for the sub-matrix r being processed). In this embodiment, eight CNPUs **1004** operate in parallel corresponding to each of the rows (check nodes) of H. In one embodiment, the number of input messages $L_{v_i\to cj}^{e}$ and output messages $L_{cj\to v_i}^{e}$ that each CNPU **1004** can compute per clock cycle is equal to the number of '1s' in the corresponding row of the sub-matrix being processed. If the CNPU **1004** computes only one input and one output messages per clock cycle it is called a serial CNPU. If it computes more than one (but lower than the total number of 1s in the corresponding row of H) input and output messages per clock cycle it is called a semi-parallel CNPU. Furthermore, in one embodiment, each CNPU **1004** can operate on two different received codewords at a time using, for example, the CNPU architecture of FIG. **9** described above. For example, in one embodiment, the minimum computation unit **902** and the sign product computation unit **904** of FIG. **9** can operate on one codeword while the output computation unit **910** operates on a different codeword. The CNPU supports two different codewords because the minimum computation unit **902** and the output computation unit **910** are isolated by the implementation of the two message memories **906** and **908**.

[0075] Inverse permutation block **1014** (e.g., a barrel shifter) receives the outputs of the CNPUs **904** and provides appropriate inputs to the VNPUs **1002** for carrying out the computation of Eq. (5) and Eq. (7). In one embodiment, the decoder **125** has q parallel VNPUs **902** (e.g., q=4) corresponding to the q columns (variable nodes) of each sub-matrix of H. In one embodiment, the complexity is reduced because only q (and not n) parallel VNPUs **1002** are implemented, i.e., it is not necessary to implement one VNPU per variable node. Multiplexer **1020** provides LLR values to FIFO register **1016** which outputs these to the VNPUs **1002** at the appropriate time to compute Eq. (5) and Eq. (7). Feedback paths **1024**, **1026** provide intermediate values to the beginning of the pipeline to perform additional iterations of the iterative decoding process.

[0076] The decoder architecture of FIG. **10** beneficially allows the decoder **125** to reuse the same q VNPUs **1002** at each step, reducing μ times the associated hardware. Furthermore, the interconnection complexity is also reduced because the interconnection network is associated with the non-zeros entries of $H^{(r)}$, which is μ times smaller than that of the original H. The locks of the CNPU **904** are simplified since the recursive computation of the check node equation (8) has significant lower complexity than a full-parallel implementation. Furthermore, the recursive CNPU **1004** stores only two minimum values which are the outputs of equations (10B) and (10C), in one embodiment. Therefore, it is not necessary to store all L^{e} messages. This reduces the memory requirements of the decoder **125**.

[0077] In one embodiment, the decoder architecture of FIG. **10** efficiently performs the iterative decoding process by processing multiple codewords. For example, rather than processing all of the iterations of one codeword and then going to the next codeword, the decoder instead processes one iteration of a first codeword, then one iteration of a second codeword and so on up to an N^{th} codeword. Then, the decoder processes the next iteration of the first codeword, and so on. Note that two codewords can be processed at the same time by different blocks of the decoder (for instance, the minimum computation unit **902** and the output computation unit **910** can process different codewords at the same time). This modification can be combined with early termination (i.e., a variable number of iterations is performed on each codeword depending on the outcome of the parity check). In this embodiment, when the decoding process of one of the N codewords is completed, a new codeword can replace it while the other codewords continue the decoding process. Thus, the

[Copia de la patente de USA, pg.25/28]

decoder need not necessarily wait until all the N codewords are decoded in order to introduce new codewords to the decoder.

[0078] For example, when the multiplexers **1018**, **1020** close the decoder loop, there may be two codewords stored in the decoder: e.g., codeword A and codeword B. The output computation unit **910** of the CNPU **1004** read the information of the codeword A from the message memory **908** (see FIG. **9**) and computes the messages $L_{C_j \to v_i}^{e}$ of codeword A. These messages are passed to the VNPU **1002** through the permutation block **1014**. The VNPU **1002** computes the messages $L_{v_i \to C_j}^{e}$ of codeword A. These messages return to the CNPU **1004** through the multiplexer **1018** and the permutation block **1012**. All these blocks (**1014**, **1002**, **1018**, **1012**) may introduce a latency (for example, due to their pipeline implementation). Because of this latency, the minimum computation unit **902** does not start processing until the new $L_{v_i \to C_j}^{e}$ arrived. Therefore, if the decoder **125** supported only one codeword, the output computation unit **910** may finish before the minimum computation unit **902** has finished and the output computation unit **910** would have to wait for the minimum computation unit **902** to finish its process. This waiting time would reduce the computation speed of the decoder **125**. In order to avoid this penalty, two (or more) codewords are stored in the decoder loop. As soon as the output computation unit **910** finishes the computation process of one codeword, for example codeword A, it can start with the computation of the other codeword, for example codeword B, which was stored in the message memory **906**. This is done by copying the contents of memory **906** to memory **908**. Later, when the minimum computation unit **902** finishes its computation process of the codeword A, it stores the results in the message memory **906** and it can immediately starts the computation process of codeword B. If the total latency of blocks **1014**, **1002**, **1018**, **1012**, and **904** is higher than the number of sub-matrices, more than two codewords may be decoded and stored at the same time in order to avoid the above described waiting time. This can be done by increasing the number of message memories (in a serial FIFO concatenation) inside the CNPU **1004**.

[0079] FIG. **11**A-D illustrate flow of information through the pipelined decoder architecture of FIG. **10** in which the decoder processes two codewords in parallel. The CNPUs **1004** showed in FIG. **11**A-D are divided in 3 sub-blocks. The first (left) sub-block corresponds to the minimum computation unit **902**, the sign product computation unit **904**, and part of the FIFO unit **912** showed in FIG. **9**. The second (center) sub-block corresponds to the message memory **1 906** showed in FIG. **9**. The third (right) sub-block corresponds to the message memory **2 908**, the output computation unit **910** and part of the FIFO **912**. In FIG. **11**A, a first iteration of a first codeword (e.g., q LLR) is passed in μ clock cycles to the CNPUs **1004** and enter the FIFO register **1016**. After that, in FIG. **11**B, the first iteration of the first codeword moves forward in the internal pipelines of the CNPUs **1004** and a first iteration of a second codeword is passed to the CNPUs **1004**. Furthermore, the first iteration of the second codeword enters the FIFO register **1016** and the first iteration of the first codeword moves forward in the FIFO register **1016**. After that, in FIG. **11**C, the first iteration of the first codeword is passed from the CNPUs **1004** to the VNPUs **1002**. The first iteration of the second codeword moves forward in the CNPU **1004** pipelines and in the FIFO register **1016**. A second iteration of the first codeword enters the CNPU **1004** and the FIFO register **1016**. After μ clock cycles, in FIG. **11**D, the first iteration of the second codeword is passed from the CNPUs **1004** to the VNPUs **1002**. The second iteration of the first codeword moves forward in the CNPU **1004** pipeline and FIFO register **1016**. A second iteration of the second codeword enters the CNPU **1004** and FIFO register **1016**. As will be apparent, the process described above can be extended to N codewords for any integer N.

EXAMPLE PERFORMANCE MEASUREMENTS

[0080] In one embodiment, performance of the LDPC codewords can be evaluated using a combination of analytical tools and simulation (e.g., using a field-programmable gate array or other device). For example, in one embodiment, simulations in the proximity of the low BER region of interest (e.g., $>10^{-13}$) could be used to obtain dominant trapping sets. Based on these trapping sets, BER can be estimated by using importance sampling technique.

[0081] Let r_a and r_e be the number of bits used to represent the prior LLRs and the messages from both check and variables nodes, respectively. In one embodiment, the prior LLRs are quantized (e.g., using r_a=5 bits). Furthermore, in one embodiment, the decoder is implemented using r_e=5 bits. To obtain the perform measures, an all-zeros codeword can be transmitted using binary phase-shift keying (BPSK) modulation (i.e., bits {0, 1} are mapped into symbols {+1, −1} for transmission). An additive white Gaussian noise (AWGN) channel can also be implemented to model channel noise by using a random number generator.

[0082] FIGS. **12**A-**12**C illustrate performance results for an example implementation of the decoder **125**. FIG. **12**A depicts the BER versus the signal-to-noise ratio (SNR) for code C_1 described above with 13 iterations of the decoding algorithm. The curve shows an error floor at BER=10^{-11} with an NECG of 9.45 dB at BER=10^{-15}. This error floor is caused by the presence of several absorbing sets (AS) created by the combination of cycles of length 6.

[0083] FIG. **12**B shows the performance of code C_2 described above. No error floor is observed up to 10^{-13} and the expected NEGC is 11.30 dB. However, from importance sampling analysis, a quantization sensitive error floor below 10^{-13} can be estimated. This error floor is caused by the combination of a (12,8) absorbing set and the quantization of the L^e messages in the SMSA.

[0084] FIG. **12**B shows the estimated error floor for r_e=5, 6, and 7 bits with 13 iterations. The a-posteriori LLR evolution of the (12, 8) absorbing set is shown in FIG. **12**C. (In the notation, "(e, d) absorbing set, e is the number of wrong bits and d is the number of unsatisfied check nodes). Note that the SMSA decoder with r_e=5 bits does not resolve the (12,8) absorbing set independently of the number of iterations. On the other hand, the SMSA decoder takes 17 and 12 iterations with r_e=6 and 7 bits, respectively.

Min-Sum Algorithm With Adaptive Quantization

[0085] A common problem with decoders based on SPA, MSA or its variations is that error floors tend to arise. These error floors can be challenging to estimate and reduce particularly at very low levels (e.g., below 10^{-13}). As shown above, very low error floors (e.g., below 10^{-13}) may be caused by quantization effects. In order to effectively combat these low error floors, a post-processing technique may be applied.

[Copia de la patente de USA, pg.26/28]

[0086] In one embodiment, the performance limitations described above can be improved using a real-time adaptive quantization scheme. The real-time adaptive quantization scheme combats error floor exacerbation caused by a low precision implementation of the decoder **125**. The decoder **125** applies real-time adaptation of the fractional point position in the fixed point representation of the internal MSA messages, keeping constant the total number of bits.

[0087] The adaptive quantization algorithm applies a scaling to the log-likelihood ratios (LLRs) and messages in order to increase the range of representation, and therefore reduce the saturation effects. In one embodiment, this scaling step is activated only when a predefined activation condition is met. For example, in one embodiment, the scaling is applied when the number of unsatisfied check nodes do not exceed a minimum value d_t (e.g., d=8 or d=9 for a (12, 8) absorbing set). A check node is unsatisfied if its corresponding parity equation (i.e. a row of the parity check matrix H) is unsatisfied according to the sign value of the a posteriori output of the decoder at that state. This activation condition usually occurs only after some normal iterations without scaling. Note that since the total number of bits is maintained constant, this wider range is obtained at the expense of an increase in quantization.

[0088] The fixed-point modification increases the dynamical range of the decoder messages (by increasing the quantization step). In one embodiment, the quantization change is implemented in the VNPU **1002** after summation because here messages achieve their highest value, and the saturations have the stronger distortion effect.

[0089] The scaling step can be generalized and implemented inside the VNPU **1002** as:

$$L^{e}_{v_i \to c_j} = \kappa_1^t \cdot L_i^a + \kappa_2 \cdot \left(\sum_{c_k \in c^{(v_i)} \setminus c_j} L^{e}_{c_k \to v_i} \right), \tag{11}$$

where t=1, 2, . . . , denotes the number of the extra iteration used for post-processing. Factors κ_1 and κ_2 are positive gains smaller than unity.

[0090] In one embodiment, to simplify the implementation, $\kappa_1=\kappa_2=\kappa$ can be used. Thus, the algorithm reduces to scale by κ both the output of the variable-node equation (Eq. (5)) and the prior LLR. This is shown as:

$$L^{e}_{v_i \to c_j} = \kappa \cdot \left(L_i^a + \sum_{c_k \in c^{(v_i)} \setminus c_j} L^{e}_{c_k \to v_i} \right), \tag{12}$$

$$L_i^a \leftarrow \kappa \cdot L_i^a. \tag{13}$$

[0091] Note that the prior information is gradually reduced to zero as the adaptive quantization process evolves. After a given number of iterations, the MSA operates without prior information. In one embodiment, $\kappa=1/2$ provides a good tradeoff between performance and implementation complexity.

[0092] FIGS. **13**A-**13**B illustrate performance of the decoder using the adaptive quantization algorithm. In FIG. **13**A, the a-posteriori LLR evolution of the SMSA decoder for code C_2 is shown with r_c=5 bits over the (12,8) absorbing set. Note that the absorbing set can be resolved with 5-6 extra iterations. FIG. **13**B shows the estimated BER versus SNR derived with r_e=5 bits and the adaptive quantization algorithm. As can be seen, the error floor observed in FIG. **12**B is corrected by the adaptive quantization algorithm.

[0093] Using the RCP-QC-LDPC codes and the adaptive quantization technique described above, the complexity of the decoding can be substantially reduced (e.g., to about 5 extra iterations under some conditions. Furthermore, the error floors can be drastically lowered, resulting in an expected NECG≧11.30 dB or better at a BER of 10^{-15} in some embodiments. Furthermore, the described approach beneficially avoids a hard-decision-based block outer code and reduces the block size significantly relative to prior techniques. This reduction of complexity and the concomitant reduction of latency can be an important factor for commercial applications, thereby enabling applications such as 100 Gb/s optical transport networks.

[0094] In some embodiments, these codes can achieve an expected coding gain of, for example, 11.30 dB at 10^{-15}, 20% OH, and a block size of 24576 bits. The described code beneficially can minimize the BER floor while simultaneously reducing the memory requirements and the interconnection complexity of the iterative decoder. Under certain conditions, the described codes can achieve NECG of 10.70 dB at a BER of 10^{-13} and no error floors.

[0095] Although the detailed description contains many specifics, these should not be construed as limiting the scope of the invention but merely as illustrating different examples and aspects of the invention. It should be appreciated that the scope of the invention includes other embodiments not discussed in detail above. For example, the functionality has been described above as implemented primarily in electronic circuitry. This is not required, various functions can be performed by hardware, firmware, software, and/or combinations thereof. Depending on the form of the implementation, the "coupling" between different blocks may also take different forms. Dedicated circuitry can be coupled to each other by hardwiring or by accessing a common register or memory location, for example. Software "coupling" can occur by any number of ways to pass information between software components (or between software and hardware, if that is the case). The term "coupling" is meant to include all of these and is not meant to be limited to a hardwired permanent connection between two components. In addition, there may be intervening elements. For example, when two elements are described as being coupled to each other, this does not imply that the elements are directly coupled to each other nor does it preclude the use of other elements between the two. Various other modifications, changes and variations which will be apparent to those skilled in the art may be made in the arrangement, operation and details of the method and apparatus of the present invention disclosed herein without departing from the spirit and scope of the invention as defined in the appended claims. Therefore, the scope of the invention should be determined by the appended claims and their legal equivalents.

1. A decoder for decoding forward error correcting codewords using a parity check matrix comprising a plurality of sub-matrices, the decoder comprising:

a plurality of check node processing units, each check node processing performing a check node computation corresponding to a different row of the parity check matrix; and

a plurality of variable node processing units, each variable node processing unit determining variable node update

[Copia de la patente de USA, pg.27/28]

computations corresponding to different columns belonging to a same sub-matrix of the parity check matrix;

wherein the plurality of check node processing units and the plurality of variable node processing units operate on only one sub-matrix of the parity check matrix at each step of an iterative decoding process; and

wherein the decoder processes two or more codewords in parallel such that the decoder begins decoding a subsequently received codeword prior to completing decoding of a prior received codeword.

2. The decoder of claim **1**, wherein the decoder is configured to apply an iterative decoding algorithm and wherein the decoder is configured to process a first iteration of the prior received codeword, and process a first iteration of the subsequently received codeword prior to processing a second iteration of the prior received codeword.

3. The decoder of claim **1**, wherein each of the check node processing units comprises:

a parallel pipelined processing architecture for processing two or more codewords at a time.

4. The decoder of claim **1**, wherein each of the check node processing units comprises:

a minimum computation unit determining first and second minimum values of input messages received from a plurality of variable nodes and determining a variable node corresponding to the first minimum value;

a sign product computation unit determining a product of signs of the input messages received from the plurality of variable nodes;

a plurality of pipelined message memories storing the determined first and second minimum values, an identifier of the determined variable node corresponding to the first minimum value, and the product of the signs;

a sign FIFO unit storing the signs of the input messages from the plurality of variable nodes; and

an output computation unit determining an output message based on the signs of the input messages from the sign FIFO unit and values stored in the plurality of message memories.

5. The decoder of claim **1**, wherein the parity check matrix is row-regular and column-regular such that the parity check matrix has a first same number of 1s in each row and a second same number of 1s in each column.

6. The decoder of claim **1**, wherein the parity check matrix is quasi-cyclic such that a circular shift of a valid codeword by an integer amount results in another valid codeword.

7. The decoder of claim **6**, wherein the parity check matrix comprises an array of circulant sub-matrices.

8. The decoder of claim **1**, wherein the parity check matrix comprises a 2×12 array of circulants of size 2048×2048, each circulant having two non-zero diagonals.

9. The decoder of claim **1**, wherein the decoder is configured to iteratively decode the low density parity check codewords based on one of: a sum- product algorithm, a min-sum algorithm, and a scaled min-sum algorithm.

10. The decoder of claim **9**, wherein the decoder further comprises a control unit for:

determining if an activation criteria is met during an iteration of the decoding; and

responsive to the activation criteria being met, configuring the decoder to adaptively quantize messages processed by the decoder.

11. The decoder of claim **10**, wherein adaptively quantizing the input values to the decoder comprises scaling a log-likelihood ratios and the messages by scaling factors to increase a representation range given a fixed number of bits.

12. The decoder of claim **10**, wherein determining if the activation criteria is met comprises determining if a number of unsatisfied check nodes is smaller than a predetermined threshold.

13. A method for forward error correction comprising:

receiving, by a decoder, a stream of low density parity check codewords;

iteratively decoding the low density parity check codewords based on a parity check matrix;

for each iteration of the decoding, determining if an activation criteria is met; and

responsive to the activation criteria being met for an iteration of the decoding, configuring the decoder to adaptively quantize messages processed by the decoder based on a scaling factor.

14. The method of claim **13**, wherein iteratively decoding the low density parity check codewords comprises decoding based on one of: a sum- product algorithm, a min-sum algorithm, and a scaled min-sum algorithm.

15. The method of claim **13**, wherein determining if the activation criteria is met comprises determining if a number of unsatisfied parity checks of the parity checks matrix is smaller than a predetermined threshold.

16. The method of claim **13**, where adaptively quantizing the input to the decoder comprises:

determining a log-likelihood ratio of received bits of the low density parity check codewords;

receiving a message representing output of a prior decoding iteration;

scaling the log-likelihood ratio and the message by a scaling factor to increase a representation range given a fixed number of bits; and

iteratively decoding the low density parity check codewords using the scaled log-likelihood ratio and scaled message.

17. A method for generating a quasi-cyclic regular column-partition parity check matrix for forward error correction, the method comprising:

receiving an initial matrix H;

determining a count of cycles of the initial matrix H for each of a plurality of different cycle lengths;

creating a matrix $\hat{H}$ as copy of the initial matrix H;

modifying the matrix $\hat{H}$ based on a modification algorithm;

determining a count of cycles of the modified matrix $\hat{H}$ for each of the plurality of different cycle lengths;

determining a lowest of the plurality of different cycle lengths for which the initial matrix H and the modified matrix $\hat{H}$ have different counts; and

replacing the initial matrix H with the modified matrix $\hat{H}$ responsive to the initial matrix H having a higher count for the lowest of the plurality of different cycle lengths for which the initial matrix H and the modified matrix $\hat{H}$ have different counts.

[Copia de la patente de USA, pg.28/28]

18. The method of claim **17**, wherein modifying the matrix $\hat{H}$ based on a modification algorithm comprises:

pseudo-randomly selecting a sub-matrix of the matrix $\hat{H}$;

pseudo-randomly selecting a diagonal of the selected sub-matrix of the matrix $\hat{H}$;

changing a position of the selected diagonal of the selected sub-matrix of the matrix $\hat{H}$ to a different pseudo-randomly selected position.

19. The method of claim **17**, wherein determining the count of cycles of the initial matrix H for each of the plurality of different cycle lengths comprises:

determining an adjacency matrix of a Tanner Graph of matrix H;

determining the count based on the adjacency matrix.

* * * * *

Bibliografía

[1] A. Abbasfar, D. Divsalar, and K. Yao. Accumulate repeat accumulate codes. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, 2004.

[2] A. Abbasfar, D. Divsalar, and Kung Yao. Accumulate repeat accumulate codes. In *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, volume 1, pages 509–513 Vol.1, 2004.

[3] Srinivas M. Aji and Robert J. McEliece. The generalized distributive law. *IEEE Trans. Info. Theory*, 46(2):325–343, March 2000.

[4] M. Akita, H. Fujita, T. Mizuochi, K. Kubo, H. Yoshida, K. Kuno, and S. Kurahashi. Third generation fec employing turbo product code for long-haul dwdm transmission systems. In *Optical Fiber Communication Conference and Exhibit, 2002. OFC 2002*, pages 289–290, 2002.

[5] S. Alamouti. A simple transmit diversity technique for wireless communications. *Selected Areas in Communications, IEEE Journal on*, 16(8):1451–1458, 1998.

[6] E. Arikan. Channel polarization: A method for constructing capacity-achieving codes. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 1173–1177, 2008.

[7] E. Arikan. A performance comparison of polar codes and reed-muller codes. *Communications Letters, IEEE*, 12(6):447–449, 2008.

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

[9] E. Arikan. Source polarization. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 899–903, 2010.

[10] E. Arikan. A survey of reed-muller codes from polar coding perspective. In *Information Theory Workshop (ITW), 2010 IEEE*, pages 1–5, 2010.

[11] E. Arikan. Systematic polar coding. *Communications Letters, IEEE*, 15(8):860–862, 2011.

[12] A. Ashikhmin and S. Litsyn. Simple map decoding of first-order reed-muller and hamming codes. *Information Theory, IEEE Transactions on*, 50(8):1812–1818, 2004.

[13] L. Bahl, C. Cullum, W. Frazer, and F. Jelinek. An efficient algorithm for computing free distance (corresp.). *Information Theory, IEEE Transactions on*, 18(3):437–439, 1972.

[14] L. Barnault and D. Declercq. Fast decoding algorithm for ldpc over gf(2q). In *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, pages 70–73, 2003.

[15] R. Bellman. Dynamic programming. *Princeton University Press*, 1957.

[16] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara. Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding. *Information Theory, IEEE Transactions on*, 44(3):909–926, 1998.

[17] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Communications, 1993. ICC '93 Geneva. Technical Program, Conference Record, IEEE International Conference on*, volume 2, pages 1064–1070, 1993.

[18] I. Blake, C. Heegard, T. Hoholdt, and V. Wei. Algebraic-geometry codes. *Information Theory, IEEE Transactions on*, 44(6):2596–2618, 1998.

[19] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3:68–79, March 1960.

[20] F Buckley and E Harary. *Distance in Graphs*. Addison-Wesley, Redwood, Califomia, 1990.

[21] K. A. Bush. Orthogonal arrays of index unity. *Ann. Math. Stat*, 23:426–434, 1952.

[22] A.R. Calderbank. The art of signaling: fifty years of coding theory. *Information Theory, IEEE Transactions on*, 44(6):2561 –2595, oct 1998.

[23] M. Carroll, J. Roese, and Takuya Ohara. The operator's view of otn evolution. *Communications Magazine, IEEE*, 48(9):46–52, 2010.

[24] E. Cavus, C.L. Haymes, and B. Daneshrad. Low ber performance estimation of ldpc codes via application of importance sampling to trapping sets. *Communications, IEEE Transactions on*, 57(7):1886–1888, 2009.

[25] A.P. Chandrakasan and R.W. Brodersen. Minimizing power consumption in digital cmos circuits. *Proceedings of the IEEE*, 83(4):498–523, 1995.

[26] Deyuan Chang, Fan Yu, Zhiyu Xiao, N. Stojanovic, F.N. Hauske, Yi Cai, Changsong Xie, Liangchuan Li, Xiaogeng Xu, and QianJin Xiong. Ldpc convolutional codes using layered decoding algorithm for high speed coherent optical transmission. In *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2012 and the National Fiber Optic Engineers Conference*, pages 1–3, 2012.

[27] Hsie-Chia Chang, Chien-Ching Lin, Fu-Ke Chang, and Chen-Yi Lee. A universal vlsi architecture for reed–solomon error-and-erasure decoders. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 56(9):1960–1967, 2009.

[28] Chao-Yu Chen, Qin Huang, Chi chao Chao, and Shu Lin. Two low-complexity reliability-based message-passing algorithms for decoding non-binary ldpc codes. *Communications, IEEE Transactions on*, 58(11):3140–3147, 2010.

[29] J. Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and X.-Y. Hu. Reduced-complexity decoding of LDPC codes. *IEEE Trans. Commun.*, 53(8):1288 – 1299, Aug. 2005.

[30] Pin-Han Chen, Jian-Jia Weng, Chung-Hsuan Wang, and Po-Ning Chen. Bch code selection and iterative decoding for bch and ldpc concatenated coding system. *Communications Letters, IEEE*, 17(5):980–983, 2013.

[31] Zhanping Chen, Liqiong Wei, and Kaushik Roy. Reducing glitching and leakage power in low voltage CMOS circuits. *ECE Technical Reports, Purdue University*, (85), 1997.

[32] V. Chernyak, M. Chertkov, M. G. Stepanov, and B. Vasic. Error correction on a tree: An instanton approach. *Physical Review Letters*, 93(19), November 2004.

[33] F. Chiaraluce and R. Garello. Extended hamming product codes analytical performance evaluation for low error rate applications. *Wireless Communications, IEEE Transactions on*, 3(6):2353–2361, 2004.

[34] W.T. Cochran, James W. Cooley, D.L. Favin, H.D. Helms, R. Kaenel, W.W. Lang, Jr. Maling, G.C., D.E. Nelson, C.M. Rader, and Peter D. Welch. What is the fast fourier transform? *Audio and Electroacoustics, IEEE Transactions on*, 15(2):45–55, 1967.

[35] G. Colavolpe, A. Barbieri, and G. Caire. Algorithms for iterative decoding in the presence of strong phase noise. *Selected Areas in Communications, IEEE Journal on*, 23(9):1748–1757, 2005.

[36] D.J. Costello and Jr. Forney, G.D. Channel coding: The road to channel capacity. *Proceedings of the IEEE*, 95(6):1150–1177, 2007.

[37] Ahmad Darabiha, Anthony Chan Carusone, and Frank R. Kschischang. Block-interlaced ldpc decoders with reduced interconnect complexity. *IEEE Trans. on Circuits and Systems, vol. 55, no. 1, pp. 74-78*, January 2008.

[38] S. Dave, L. Esker, Fan Mo, W. Thesling, J. Keszenheimer, and R. Fuerst. Soft-decision forward error correction in a 40-nm asic for 100-gbps otn applications. In *Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference*, pages 1–3, 2011.

[39] M. C. Davey and D. J. MacKay. Low density parity check codes over $gf(q)$. *IEEE Com. Letters*, 2(6), 1998.

[40] D. Declercq and M. Fossorier. Decoding algorithms for nonbinary ldpc codes over gf (q). *Communications, IEEE Transactions on*, 55(4):633–643, 2007.

[41] Changyan Di, D. Proietti, I.E. Telatar, T.J. Richardson, and R.L. Urbanke. Finite-length analysis of low-density parity-check codes on the binary erasure channel. *Information Theory, IEEE Transactions on*, 48(6):1570 –1579, jun 2002.

[42] Qiuju Diao, Wei Zhou, Shu Lin, and K. Abdel-Ghaffar. A transform approach for constructing quasi-cyclic euclidean geometry ldpc codes. In *Information Theory and Applications Workshop (ITA), 2012*, pages 204–211, 2012.

[43] D. Divsalar, H. Jin, and R. J. McEliece. Coding theorems for turbo-like's codes. In *in Proc. 1998 Allerton Conf., Allerton, IL*, pages 201–210, Sep. 1998.

[44] Ivan Djordjevic, Murat Arabaci, and Lyubomir Minkov. Next generation fec for high-capacity communication in optical transport networks. *Journal of Lightwave Technology, vol. 27, no. 16, pp. 3518-3530*, Jun 2009.

[45] L. Dolecek, Zhengya Zhang, V. Anantharam, M.J. Wainwright, and B. Nikolic. Analysis of absorbing sets and fully absorbing sets of array-based ldpc codes. *Information Theory, IEEE Transactions on*, 56(1):181 –201, jan. 2010.

[46] P. Elias. Error-free coding. *IRE Trans. Inform. Theory.*, IT-4:29–37, Sep. 1954.

[47] P. Elias. Coding for noisy channels. *IRE Conv. Rec.*, pages 37–46, Mar. 1955.

[48] K. Engdahl, M. Lentmaier, and K. S. Zigangirov. On the theory of low density convolutional codes. *in AAECC-13: Proceedings of the 13th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. London, UK: Springer-Verlag*, pages 77–86, 1999.

[49] K. Engdahl and K. S. Zigangirov. On the theory of low density convolutional codes i. *Problemy Peredachi Informatsii*, 35(4):295–310, 1999.

[50] E.R.Berlekamp. Algebraic coding theory. *McGraw-Hill, New York*, 1968.

[51] Morero D.A. et al. Non-concatenated fec codes for ultra-high speed optical transport networks. In *U.S. Patent Application No. 13/406,452*, Feb. 2012.

[52] R. Fano. A heuristic discussion of probabilistic decoding. *Information Theory, IEEE Transactions on*, 9(2):64–74, 1963.

[53] M.P.C. Fossorier. Quasicyclic low-density parity-check codes from circulant permutation matrices. *Information Theory, IEEE Transactions on*, 50(8):1788–1793, 2004.

[54] Jr. G. D. Forney. Concatenated codes. *Technical Report 440, MIT*, Dec. 1965.

[55] R. G. Gallager. Low-density parity-check codes. *Cambridge, MA: MIT press*, 1963.

[56] Robert G. Gallager. Low-density parity-check codes. *IR Trans. on Info. Theory*, IT-8:21–28, January 1962.

[57] David Gamarnik, Devavrat Shah, and Yehua Wei. Belief propagation for min-cost network flow: Convergence and correctness. *arXiv:1004.1586v4 [cs.DM]*, (4), Jul. 2012.

[58] M. J. E. Golay. Notes on digital coding. *Proc. IRE*, 37(6):657, 1949.

[59] V. D. Goppa. A new class of linear error-correcting codes. *Probl. Inform. Transm.*, 6:207–212, Sept. 1970.

[60] V. D. Goppa. Rational representation of codes and (L,g) codes. *Probl. Inform. Transm.*, 7:41–49, Sept. 1971.

[61] V. D. Goppa. Some codes constructed on the basis of (L,g) codes. *Probl. Inform. Transm.*, 8:107–109, June 1972.

[62] V. D. Goppa. Codes that are associated with divisors (russian). *Problemy Peredaci Informacii*, (1):33–39, 1977.

[63] D. C. Gorenstein and N. Zierler. A class of error-correcting codes in p^m symbols. *J. SIAM*, 9:207–214, 1961.

[64] M.S. Grewal and A.P. Andrews. *Kalman filtering: theory and practice.* Prentice-Hall information and system sciences series. Prentice-Hall, 1993.

[65] Frederic Guilloud, Emmanuel Boutillon, Jacky Tousch, and Jean-Luc Danger. Generic description and synthesis of ldpc decoders. *Transactions on Communications, vol. 55, no. 11, pp. 2084-2091*, November 2007.

[66] V. Guruswami and M. Sudan. Improved decoding of reed-solomon and algebraic-geometry codes. *Information Theory, IEEE Transactions on*, 45(6):1757–1767, 1999.

[67] R. W. Hamming. Error detecting and error correcting codes. *Bell Syst. Tech. J.*, 2(26), April 1950.

[68] Yang Han and W. Ryan. Low-floor decoders for LDPC codes. *IEEE Trans. Commun.*, 57(6):1663–1673, June 2009.

[69] S.H. Hassani, N. Macris, and R. Urbanke. Coupled graphical models and their thresholds. In *Information Theory Workshop (ITW), 2010 IEEE*, pages 1–5, 2010.

[70] A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres (Paris)*, (2):147–156, Sept. 1959.

[71] R. Holzlohner, A. Mahadevan, C. R. Menyuk, J. M. Morris, and J. Zweck. Evaluation of the very low ber of fec codes using dual adaptive importance sampling. *IEEE Communications Letters*, 9(2), February 2005.

[72] Qin Huang, Qiuju Diao, Shu Lin, and K. Abdel-Ghaffar. Cyclic and quasi-cyclic ldpc codes: New developments. In *Information Theory and Applications Workshop (ITA), 2011*, pages 1–10, 2011.

[73] W.C. Huffman and V. Pless. Fundamentals of error-correcting codes. *Cambridge University Press*, 2003.

[74] H. Imai and S. Hirakawa. A new multilevel coding method using error-correcting codes. *Information Theory, IEEE Transactions on*, 23(3):371–377, May. 1977.

[75] F. Jelinek. A fast sequential decoding algorithm using a stack. *IBM J. Res. and Dev.*, (13):675–685, 1969.

[76] Xueqin Jiang and Moon ho Lee. Large girth non-binary ldpc codes based on finite fields and euclidean geometries. *Signal Processing Letters, IEEE*, 16(6):521–524, 2009.

[77] A. Jimenez and K.S. Zigangirov. Periodic time-varying convolutional codes with low-density parity-check matrices. In *Information Theory, 1998. Proceedings. 1998 IEEE International Symposium on*, pages 305– , 1998.

[78] A. Jimenez Felstrom and K.S. Zigangirov. Time-varying periodic convolutional codes with low-density parity-check matrix. *Information Theory, IEEE Transactions on*, 45(6):2181–2191, 1999.

[79] H. Jin, A. Khandekar, and R. McEliece. Irregular repeat-accumulate codes. *Proc. 2nd. Int. Symp. on Turbo Codes and Related Topics, Brest, France*, pages 1–8, Sep. 2000.

[80] David G. Messerschmitt John R. Barry, Edward A. Lee. *Digital Communication, Third Edition*. Kluwer Academic, 2004.

[81] N. Kamiya. High-rate quasi-cyclic low-density parity-check codes derived from finite affine planes. In *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pages 2295–2299, 2005.

[82] N. Kamiya and S. Shioiri. Concatenated QC-LDPC and SPC codes for 100 Gbps ultra long-haul optical transmission systems. *OFC/NFOEC*, pages 1–3, March 2010.

[83] Jingyu Kang, Qin Huang, Li Zhang, Bo Zhou, and Shu Lin. Quasi-cyclic ldpc codes: an algebraic construction. *Communications, IEEE Transactions on*, 58(5):1383–1396, 2010.

[84] R. Koetter and A. Vardy. Algebraic soft-decision decoding of reed-solomon codes. *Information Theory, IEEE Transactions on*, 49(11):2809–2825, 2003.

[85] R. Kotter and A. Vardy. Algebraic soft-decision decoding of reed-solomon codes. In *Information Theory, 2000. Proceedings. IEEE International Symposium on*, pages 61–, 2000.

[86] Y. Kou, S. Lin, and M.P.C. Fossorier. Low-density parity-check codes based on finite geometries: a rediscovery and new results. *IEEE Trans. Inform. Theory*, 47(7):2711 –2736, Nov. 2001.

[87] Yu Kou, Shu Lin, and M.P.C. Fossorier. Low density parity check codes based on finite geometries: a rediscovery. In *Information Theory, 2000. Proceedings. IEEE International Symposium on*, pages 200–, 2000.

[88] Yu Kou, Shu Lin, and M.P.C. Fossorier. Low density parity check codes: construction based on finite geometries. In *Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE*, volume 2, pages 825–829 vol.2, 2000.

[89] Yu Kou, Shu Lin, and M.P.C. Fossorier. Low-density parity-check codes based on finite geometries: a rediscovery and new results. *Information Theory, IEEE Transactions on*, 47(7):2711–2736, 2001.

[90] F.R. Kschischang and B.J. Frey. Iterative decoding of compound codes by probability propagation in graphical models. *Selected Areas in Communications, IEEE Journal on*, 16(2):219–230, 1998.

[91] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Info. Theory*, 47(2):498–519, February 2001.

[92] S. Kudekar, T. Richardson, and R. Urbanke. Threshold saturation via spatial coupling: Why convolutional ldpc ensembles perform so well over the bec. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 684–688, 2010.

[93] S. Kudekar, T.J. Richardson, and R.L. Urbanke. Threshold saturation via spatial coupling: Why convolutional ldpc ensembles perform so well over the bec. *Information Theory, IEEE Transactions on*, 57(2):803–834, 2011.

[94] S. Kudekar, T.J. Richardson, and R.L. Urbanke. Spatially coupled ensembles universally achieve capacity under belief propagation. *e-print: http://arxiv.org/abs/1201.2999*, Jan. 2012.

[95] H.T. Kung, C.E. Leiserson, and CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE. *Systolic Arrays for (VLSI)*. CMU-CS. Carnegie-Mellon University, Department of Computer Science, 1978.

[96] Maxim Kuschnerov, O Agazzi, V Veljanovski, J Slovak, M Herrmann, C Hofer, U Bauer, T Rieger, S Camatel, and P a Voois. Recent advances in signal processing for real–time implementation–40gb/s, 100gb/s and beyond. In *Signal Processing in Photonic Communications*. Optical Society of America, 2012.

[97] S. Landner and O. Milenkovic. Algorithmic and combinatorial analysis of trapping sets in structured ldpc codes. In *Wireless Networks, Communications and Mobile Computing, 2005 International Conference on*, volume 1, pages 630–635 vol.1, 2005.

[98] Dong-U Lee, Ray C.C. Cheung, John D. Villasenor, and Wayne Luk. Inversion-based hardware gaussian random number generator: A case study of function evaluation via hierarchical segmentation. In *FPT 2006. IEEE International Conference on*, pages 33 –40, Dec. 2006.

[99] Jing Li, Erozan Kurtas, Krishna R. Narayanan, and Costas N. Georghiades. On the performance of turbo product codes over partial response channels. *IEEE Trans. on Communications*, 37(4):1932–1934, July 2001.

[100] Jing Li, Krishna Narayanan, and Costas Georghiades. Product accumulate codes: A class of codes with near-capacity performance and low decoding complexity. *IEEE Trans. Info. Theory.*, 50(1):31–46, January 2004.

[101] Jing Li, Krishna R. Narayanan, Erozan Kurtas, and Costas N. Georghiades. On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels. *IEEE Trans. on Communications*, 50(5):723–734, May 2002.

[102] Shu Lin, Lei Chen, Jun Xu, and I. Djurdjevic. Near shannon limit quasi-cyclic low-density parity-check codes. In *Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE*, volume 4, pages 2030–2035 vol.4, 2003.

[103] Shu Lin and Daniel Costello. Error control coding, fundamental and applications. *Pearson Prentice Hall, Second Edition*, 2004.

[104] H.-A. Loeliger. On hybrid factor graphs and adaptive equalization. In *Information Theory, 2001. Proceedings. 2001 IEEE International Symposium on*, pages 268–, 2001.

[105] Taiwan Semiconductor Manufacturing Company Ltd. N28hp standard cell library, *Datasheet TCBN28HPBWP35*. Nov. 2010.

[106] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann. Practical loss-resilient codes. *in Proc. 29th Symp. Theory Computing*, pages 150–159, 1997.

[107] M.G. Luby, M. Amin Shokrolloahi, M. Mizenmacher, and D.A. Spielman. Improved low-density parity-check codes using irregular graphs and belief propagation. In *Information Theory, 1998. Proceedings. 1998 IEEE International Symposium on*, 1998.

[108] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman. Improved low-density parity-check codes using irregular graphs. *Information Theory, IEEE Transactions on*, 47(2):585–598, 2001.

[109] D. J. C. Mackay and R. M. Neal. Near Shannon limit performance of low density parity check codes. *IEE Electronics Letters, vol. 33, no. 66, pp 457-8*, March 1997.

[110] David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2005.

[111] David J.C. MacKay and Michael S. Postol. Weaknesses of Margulis and Ramanujan-Margulis low-density parity-check codes. *Electronic Notes in Theoretical Computer Science*, 2003.

[112] M. M. Mansour and N. R. Shanbhag. High-throughput ldpc decoders. *IEEE Trans. Very Large Scale Intergr. (VLSI) Syst., vol. 11, no. 6, pp. 976-996*, December 2003.

[113] J.L. Massey. Shift-register synthesis and bch decoding. *IEEE Trans. on Inform. Theory*, IT-15(1):122–127, Jan. 1969.

[114] A. McGregor and O. Milenkovic. On the hardness of approximating stopping and trapping sets in ldpc codes. In *Information Theory Workshop, 2007. ITW '07. IEEE*, pages 248–253, 2007.

[115] A. McGregor and O. Milenkovic. On the hardness of approximating stopping and trapping sets. *Information Theory, IEEE Transactions on*, 56(4):1640–1650, 2010.

[116] Y. Miyata, W. Matsumoto, H. Yoshida, and T. Mizuochi. Efficient fec for optical communications using concatenated codes to combat error-floor. In *Optical Fiber communication/National Fiber Optic Engineers Conference, 2008. OFC/NFOEC 2008. Conference on*, pages 1–3, 2008.

[117] T. Mizuochi, Y. Miyata, T. Kobayashi, Kazuhide Ouchi, K. Kuno, K. Kubo, K. Shimizu, H. Tagami, H. Yoshida, H. Fujita, M. Akita, and K. Motoshima. Forward error correction based on block turbo code with 3-bit soft decision for 10-gb/s optical communication systems.

Selected Topics in Quantum Electronics, IEEE Journal of, 10(2):376–386, 2004.

[118] D.A. Morero, M.A. Castrillon, F.A. Ramos, T.A. Goette, O.E. Agazzi, and M.R. Hueda. Non-concatenated fec codes for ultra-high speed optical transport networks. In *IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1 –5, dec. 2011.

[119] D.A. Morero, G. Corral-Briones, and M.R. Hueda. Parallel architecture for decoding ldpc codes on high speed communication systems. In *Argentine School of Micro-Nanoelectronics, Technology and Applications (EAMTA)*, pages 107 –110, sept. 2008.

[120] D.A. Morero, G. Corral-Briones, and M.R. Hueda. Parallel architecture for decoding LDPC codes on high speed communication systems. In *EAMTA 2008 (available in IEEE Xplorer)*, pages 107 –110, Sept. 2008.

[121] D.A. Morero and M.R. Hueda. Efficient concatenated coding schemes for error floor reduction of ldpc and turbo product codes. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 2361–2366, 2012.

[122] D.A. Morero and M.R. Hueda. Novel serial code concatenation strategies for error floor mitigation of low-density parity-check and turbo product codes. In *Can. J. Elect. Comput. Eng., IEEE*, volume 36, Spring 2013.

[123] D.E. Muller. Application of boolean algebra to switching circuit design and to error detection. *Electronic Computers, Transactions of the I.R.E. Professional Group on*, EC-3(3):6–12, 1954.

[124] J. Omura. On the viterbi decoding algorithm. *Information Theory, IEEE Transactions on*, 15(1):177–179, 1969.

[125] K. Onohara *et. al.* Soft-decision-based forward error correction for 100 Gb/s transport systems. *IEEE J. Select. Topics Quantum Electron.*, 16(5):1258 –1267, Sept.-Oct. 2010.

[126] A. Orlitsky, K. Viswanathan, and J. Zhang. Stopping set distribution of ldpc code ensembles. *Information Theory, IEEE Transactions on*, 51(3):929 –953, march 2005.

[127] Payam Pakzad and Venkat Anantharam. A new look at the generalized distributive law. *IEEE Trans. Info. Theory*, 50(6):1132–1155, June 2004.

[128] E. Prange. Cyclic error-correcting codes in two symbols. *Air Force Cambridge Research Center, Cambridge, MA. Tech. Note AFCRC-TN-57-103*, Sept. 1957.

[129] E. Prange. Some cyclic error-correcting codes with simple decoding algorithms. *Tech. Note AFCRC-TN-156, Air Force Cambridge Research Center*, 1958.

[130] John G. Proakis. *Digital Communications, Third Edition.* McGraw-Hill, 1995.

[131] I. Reed. A class of multiple-error-correcting codes and the decoding scheme. *Information Theory, Transactions of the IRE Professional Group on*, 4(4):38–49, 1954.

[132] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *J Soc. Indust. Appl. Math*, 8(2):300–304, June 1960.

[133] T. Richardson. Error floors of LDPC codes. *Proc. of the 41st Allerton Conf.*, Oct. 2003.

[134] T.J. Richardson, M.A. Shokrollahi, and R.L. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *Information Theory, IEEE Transactions on*, 47(2):619–637, 2001.

[135] T.J. Richardson and R.L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *Information Theory, IEEE Transactions on*, 47(2):599–618, 2001.

[136] E. Rosnes. Stopping set analysis of iterative row-column decoding of product codes. *Information Theory, IEEE Transactions on*, 54(4):1551–1560, 2008.

[137] William Ryan and Shu Lin. Channel codes: Classical and modern. *Cambridge University Press*, 2009.

[138] E. Sasoglu, I.E. Telatar, and E. Arikan. Polarization for arbitrary discrete memoryless channels. In *Information Theory Workshop, 2009. ITW 2009. IEEE*, pages 144–148, 2009.

[139] Claude E. Shannon. A mathematical theory of communications. *Bell System Technical Journal*, 27:379–423, oct 1948.

[140] E. Sharon, S. Litsyn, and J. Goldberger. An efficient message-passing schedule for ldpc decoding. In *Electrical and Electronics Engineers in Israel, 2004. Proceedings. 2004 23rd IEEE Convention of*, pages 223–226, 2004.

[141] Irfan Siap and Nilgun Kulhan. The structure of generalized quasi cyclic codes. *Applied Mathematics E-Notes*, 5(ISSN 1607-2510):24–30, 2005.

[142] M. Sipser and D.A. Spielman. Expander codes. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 566–576, 1994.

[143] M. Sipser and D.A. Spielman. Expander codes. *Information Theory, IEEE Transactions on*, 42(6):1710–1722, 1996.

[144] R. Smarandache and P. O. Vontobel. On regular quasi-cyclic ldpc codes from binomials. *in Proc. 2004 IEEE International Symposium on Information Theory, p. 274*, Jun 2004.

[145] P.J. Smith, M. Shafi, and Hongsheng Gao. Quick simulation: a review of importance sampling techniques in communications systems. *Selected Areas in Communications, IEEE Journal on*, 15(4):597–613, 1997.

[146] D.A. Spielman. Linear-time encodable and decodable error-correcting codes. *Information Theory, IEEE Transactions on*, 42(6):1723–1731, 1996.

[147] Mikhail Stepanov and Michael Chertkov. Instanton analysis of low-density parity-check codes in the error-floor regime. *IEEE ISIT*, July 2006.

[148] K. Sugihara, Y. Miyata, T. Sugihara, K. Kubo, H. Yoshida, W. Matsumoto, and T. Mizuochi. A spatially-coupled type ldpc code with an ncg of 12 db for optical transmission beyond 100 gb/s. In *Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC), 2013*, pages 1–3, 2013.

[149] Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa. A method for solving key equation for goppa codes. *Inf. and Control*, 27:87–89, 1975.

[150] H. Tang, Jun Xu, Shu Lin, and K. A S Abdel-Ghaffar. Codes on finite geometries. *Information Theory, IEEE Transactions on*, 51(2):572–596, 2005.

[151] R. Tanner. A recursive approach to low complexity codes. *IEEE Trans. Info. Theory*, 27(5):533–547, September 1981.

[152] R.M. Tanner, D. Sridhara, A. Sridharan, T.E. Fuja, and D.J. Costello. Ldpc block and convolutional codes based on circulant matrices. *Information Theory, IEEE Transactions on*, 50(12):2966–2984, 2004.

[153] Vahid Tarokh, Hamid Jafarkhani, and A.R. Calderbank. Space-time block codes from orthogonal designs. *Information Theory, IEEE Transactions on*, 45(5):1456–1467, 1999.

[154] Vahid Tarokh, N. Seshadri, and A.R. Calderbank. Space-time codes for high data rate wireless communication: performance criterion and code construction. *Information Theory, IEEE Transactions on*, 44(2):744–765, 1998.

[155] B. Teixeira. Kalman filters [ask the experts]. *Control Systems, IEEE*, 28(2):16–18, 2008.

[156] S. Ten Brink. Convergence behavior of iteratively decoded parallel concatenated codes. *Communications, IEEE Transactions on*, 49(10):1727–1737, 2001.

[157] S. ten Brink, G. Kramer, and A. Ashikhmin. Design of low-density parity-check codes for modulation and detection. *Communications, IEEE Transactions on*, 52(4):670–678, 2004.

[158] Tao Tian, C. Jones, J.D. Villasenor, and R.D. Wesel. Construction of irregular ldpc codes with low error floors. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 5, pages 3125–3129 vol.5, 2003.

[159] L. M G M Tolhuizen. More results on the weight enumerator of product codes. *Information Theory, IEEE Transactions on*, 48(9):2573–2577, 2002.

[160] M. A. Tsfasman, S. G. Vladut, and T. Zink. Modular curves, shimura codes and goppa codes better than the varshamov-gilbert bound. *Math. Machr., 109*, pages 21–28, 1982.

[161] G. Ungerboeck. On improving data-link performance by increasing the channel alphabet and introducing sequence coding. *Int. Symp. Inform. Theory, Ronneby, Sweden*, June 1976.

[162] G. Ungerboeck. Channel coding with multilevel/phase signals. *Information Theory, IEEE Transactions on*, IT-28(1):55–67, Jan. 1982.

[163] Int. Telecomm. Union. Interfaces for the optical transport network. *ITU-T G.709*, Feb. 2010.

[164] K. Van Rompaey, I. Bolsens, and H. De Man. Just in time scheduling. In *Computer Design: VLSI in Computers and Processors, 1992. ICCD '92. Proceedings, IEEE 1992 International Conference on*, pages 295–300, 1992.

[165] N. Varnica, M.P.C. Fossorier, and A. Kavcic. Augmented belief propagation decoding of low-density parity check codes. *Communications, IEEE Transactions on*, 55(7):1308–1317, 2007.

[166] A.J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, IT-13(2):260–269, Apr. 1967.

[167] Chih-Chun Wang, S.R. Kulkarni, and H.V. Poor. Finding all small error-prone substructures in ldpc codes. *Information Theory, IEEE Transactions on*, 55(5):1976–1999, 2009.

[168] Zhongfeng Wang, Zhiqiang Cui, and Jin Sha. Vlsi design for low-density parity-check code decoding. *Circuits and Systems Magazine, IEEE*, 11(1):52–69, 2011.

[169] N. Wehn and M. Munch. Minimizing power consumption in digital circuits and systems: an overview. *Kleinheubacher Berichte, University of Kaiserslautern, Germany*, 1999.

[170] N. Wiberg. Codes and decoding on general graph. *Ph.D. Dissertation, Linkoping Univ., Linkoping Sweden*, 1996.

[171] N. Wiberg, H.-A. Loeliger, and R. Kotter. Codes and iterative decoding on general graphs. *Euro. Trans. Telecommun*, 6:513–525, 1995.

[172] N. Wiberg, H.-A. Loeliger, and R. Kotter. Codes and iterative decoding on general graphs. In *Information Theory, 1995. Proceedings., 1995 IEEE International Symposium on*, pages 468–, 1995.

[173] Andrew P. Worthen and W.E. Stark. Unified design of iterative receivers using factor graphs. *Information Theory, IEEE Transactions on*, 47(2):843–849, 2001.

[174] J. M. Wozencraft. Sequential decoding for reliable communications. *IRE Nat. Conv. Rec.*, 5:11–25, 1957.

[175] Xiaofu Wu, Xiaohu You, and Chunming Zhao. An efficient girth-locating algorithm for quasi-cyclic LDPC codes. In *Inform. Theory, International Symposium on*, pages 817 –820, July 2006.

[176] Bo Xia and W.E. Ryan. Estimating ldpc codeword error rates via importance sampling. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, pages 473–, 2004.

[177] Lingqi Zeng, Lan Lan, Ying Yu Tai, Bo Zhou, Shu Lin, and K.A.S. Abdel-Ghaffar. Construction of nonbinary cyclic, quasi-cyclic and regular ldpc codes: a finite geometry approach. *Communications, IEEE Transactions on*, 56(3):378–387, 2008.

[178] Juntan Zhang and M. Fossorier. Shuffled belief propagation decoding. In *Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on*, volume 1, pages 8–15 vol.1, 2002.

[179] Juntan Zhang and M.P.C. Fossorier. Shuffled iterative decoding. *Communications, IEEE Transactions on*, 53(2):209–213, 2005.

[180] T. Zhang, Z. Wang, and K.K. Parhi. On finite precision implementation of low density parity check codes decoder. In *Circuits and Systems, 2001. ISCAS 2001. The 2001 IEEE International Symposium on*, volume 4, pages 202 –205 vol. 4, may 2001.

[181] Xiaojie Zhang and P.H. Siegel. Quantized min-sum decoders with low error floor for ldpc codes. In *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, pages 2871 –2875, july 2012.

[182] Xinmiao Zhang, Fang Cai, and Shu Lin. Low-complexity reliability-based message-passing decoder architectures for non-binary ldpc codes. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 20(11):1938–1950, 2012.

[183] Yifei Zhang and W. Ryan. Toward low LDPC-code floors: a case study. *IEEE Trans. Commun.*, 57(6):1566–1573, June 2009.

[184] Zhengya Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M. Wainwright. Design of LDPC decoders for improved low error rate performance: quantization and algorithm choices. *IEEE Trans. Commun.*, 57(11):3258 –3268, Nov. 2009.

[185] Zhengya Zhang, L. Dolecek, B. Nikolic, V. Anantharam, and M.J. Wainwright. Lowering LDPC error floors by postprocessing. In *IEEE Globecom*, pages 1 –6, Dec. 2008.

[186] Zhengya Zhang, L. Dolecek, M. Wainwright, V. Anantharam, and B. Nikolic. Quantization effects in low-density parity-check decoders. In *ICC*, pages 6231 –6237, June 2007.

[187] Jianguang Zhao, F. Zarkeshvari, and A.H. Banihashemi. On implementation of min-sum algorithm and its modifications for decoding low-density parity-check (ldpc) codes. *Communications, IEEE Transactions on*, 53(4):549 – 554, april 2005.

[188] Bo Zhou, Jingyu Kang, Ying Yu Tai, Shu Lin, and Zhi Ding. High performance non-binary quasi-cyclic ldpc codes on euclidean geometries ldpc codes on euclidean geometries. *Communications, IEEE Transactions on*, 57(5):1298–1311, 2009.

[189] K. Sh. Zigangirov. Some sequential decoding procedures. *Probl. Peredachi Inf.*, (2):13–25, 1966.