Progress in Binary and Non-Binary Low Density Parity Check Codes

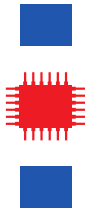
Philipp Schläfer from University of Kaiserslautern

Sponsored by NASA-GSFC

Creonic GmbH, Kaiserslautern, Germany

Introducing myself

TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN



MICROELECTRONIC
SYSTEMS DESIGN
RESEARCH GROUP

CREONIC
ip cores & system solutions

Qualification:

2011: Diploma in electrical engineering

Since 2011: Ph.D. student in a joint project of the university of Kaiserslautern at the chair of professor Wehn and Creonic GmbH.

Fields of research:

Very high throughput binary LDPC decoders

Efficient architectures for Non-Binary LDPC decoders

Outline

Ultra High Speed Binary Low Density Parity Check Decoders

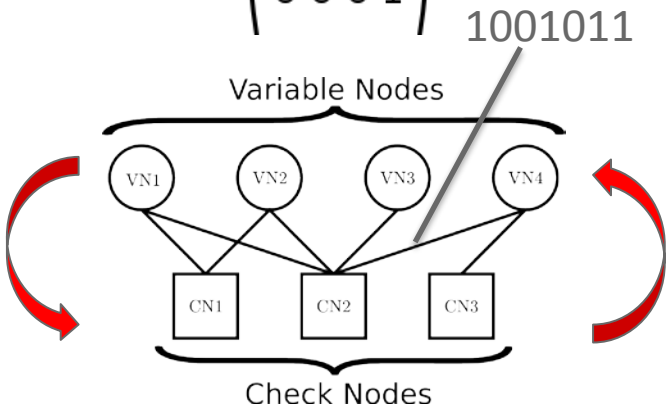
Progress in Non-Binary Low Density Parity Check Decoding

Maximum Likelihood and Soft-Reed-Solomon Decoding

Ultra High Speed Binary Low Density Parity Check Decoders

Low Density Parity Check Decoding

$$H = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



Iteratively exchange messages

Low Denisty Parity Check (LDPC) codes are **linear block codes**.

They are defined by a spares **parity check matrix H**.

Valid codewords have to satisfy $\mathbf{H} * \mathbf{x}^T = \mathbf{0}$.

Hardware architectures use Belief-Propagation (BP) algorithms for the decoding.

Soft messages are iteratively exchanged between Check Nodes (CN) and Variable Nodes (VN) until a valid codeword is detected.

Node complexity is rather low. E.g. CN of degree 5 needs to find the first two minima from 4 soft messages.

LDPC Min-Sum Decoding Algorithm

Set of CNs connected to VN n: $\mathcal{M}(n) = \{m | m \in \{0, ..., M-1\} \wedge H_{mn} \neq 0\}$

Set of VNs connected to CN m: $\mathcal{N}(m) = \{n | n \in \{0, ..., N-1\} \wedge H_{mn} \neq 0\}$

CN processing:

$$sgn\left(\epsilon_{m\to n}^{(i)}\right) = \prod_{n' \in \mathcal{N}(m)\setminus n} sgn\left(z_{n'\to m}^{(i)}\right)$$

$$\left|\epsilon_{m\to n}^{(i)}\right| = \gamma \times \min_{n' \in \mathcal{N}(m)\setminus n} \left(|z_{n'\to m}^{(i)}|\right)$$

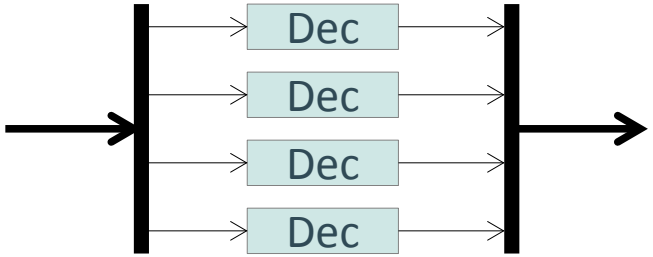
VN processing:

$$\Lambda_n^{(i)} = \lambda_n^{ch} + \sum_{m' \in \mathcal{M}(n)} \epsilon_{m'\to n}^{(i)}$$

$$z_{n\to m}^{(i)} = \lambda_n^{ch} + \sum_{m' \in \mathcal{N}(n)\setminus m} \epsilon_{m'\to n}^{(i-1)} = \Lambda_n^{(i-1)} - \epsilon_{m\to n}^{(i-1)}$$

How to Increase Throughput?

Use multiple slow decoders

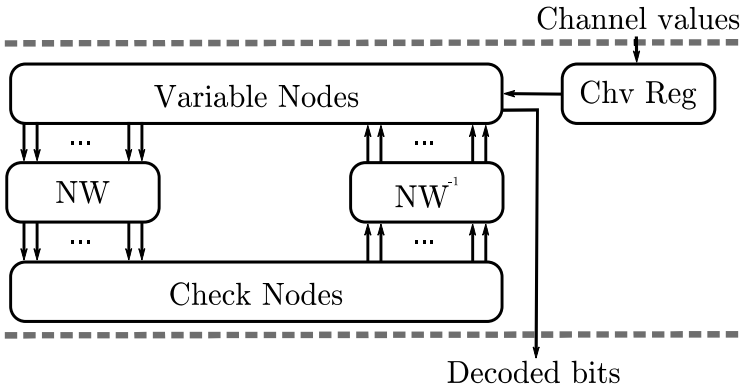


PRO
Easy to implement
CON
Low efficiency
Large memory
Large latency

Use monolithic high speed decoder

PRO
Higher efficiency
Lower latency
CON
Challenging architecture

Increasing the Parallelism



Fully parallel architecture:

High throughput, e.g. 10 GBASE-T standard.

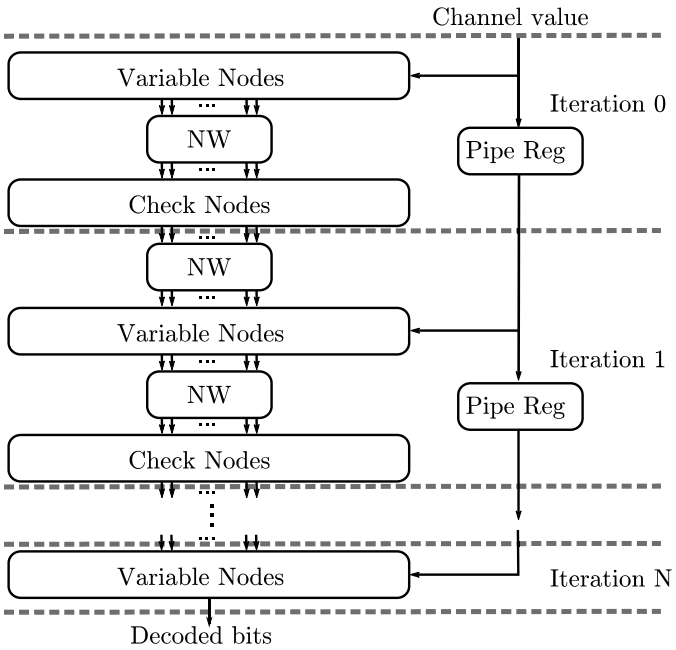
No flexibility in means of block length or code rate.

Routing congestion problems (>50% area) due to two overlapping networks.

Pipelining limited due to delay penalty in iterative loop.

Throughput limited by iterative data exchange and routing congestion.

Multi-Gigabit LDPC Decoder



Unrolled LDPC decoder:

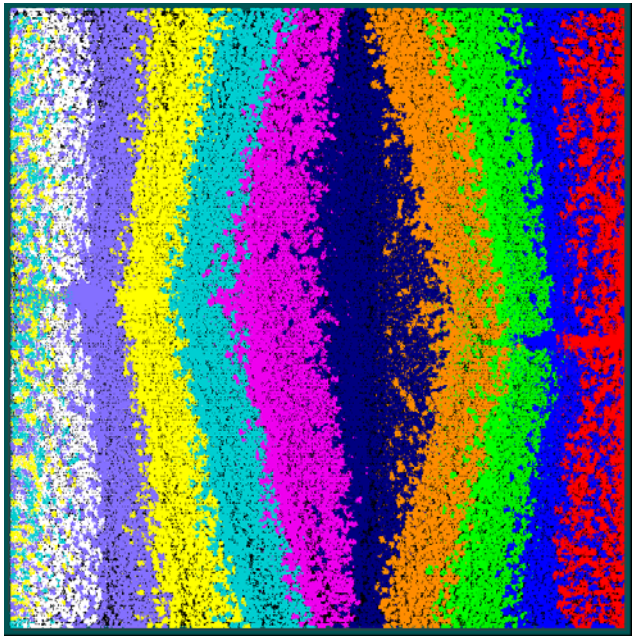
Unrolling the decoding iterations for increased throughput.

Deep pipelining possible for high clock frequency without penalty.

Largely reduced routing complexity as only one network between VNs and CNs is required.

High implementation efficiency (throughput / area) and very high throughput.

Multi-Gigabit LDPC Decoder Implementation



Unrolled LDPC decoder features: [1]

IEEE 802.11ad standard
672bit code word length
Code rate 13/16
One code word per clock cycle
9 iterations with two-phase schedule
4 bit quantization
105ns latency
12mm^2 core area (in 65nm technology)
160 Gbit/s throughput

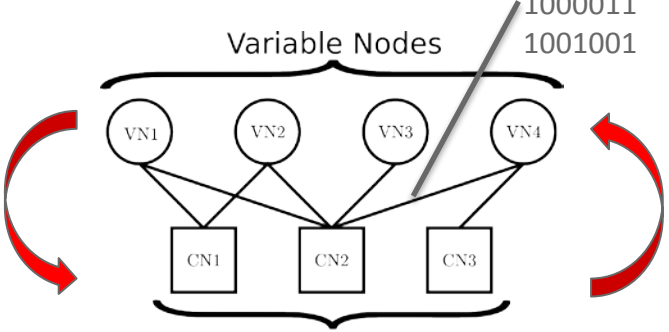
[1] Schläfer et al. “A New Dimension in Ultra High Throughput LDPC Decoding”, IEEE SIPS 2013

Progress in Non-Binary Low Density Parity Check Decoding

Non-Binary LDPC Codes

$$H = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 3 & 0 & 2 & 5 \\ 0 & 7 & 0 & 1 \end{pmatrix}$$

1001011
1011111
0100011
0001010
1001100
0100100
1000011
1001001



Iteratively exchange messages

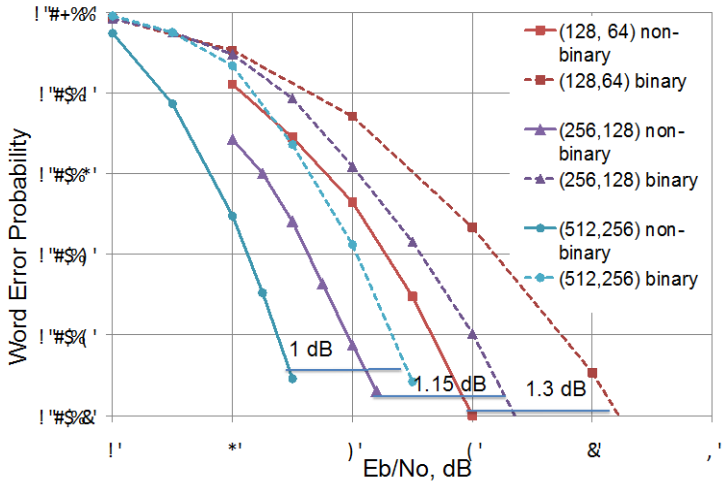
Non-Binary (NB) LDPC codes are processing Galois Field (GF(q)) symbols instead of single bits.

The basic concepts of binary LDPC stay the same for decoding of NB LDPC codes.

BP decoding can be applied but instead of one soft message a vector of q soft messages is exchanged between the nodes.

The decoding complexity of VNs and CNs increases significantly. E.g. CN of degree 5 and GF(64) must find the first 64 minima from $64^4 = 16777216$ possibilities.

Benefits of Non-Binary LDPC (1)



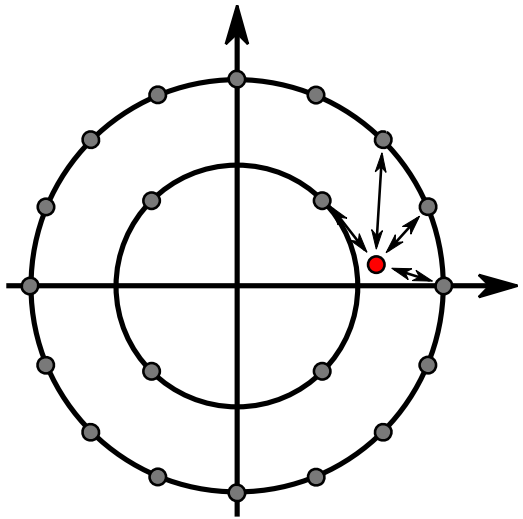
NB LDPC codes show an enhanced communications performance.

Compared to state-of-the-art binary LDPC codes, the communications performance can be enhanced by 0.5 up to several dB by NB LDPC codes [1][2][3].

Especially for short blocks the gain is significant which makes NB LDPC codes interesting for latency critical applications.

[1] Davey and MacKay “Low-Density Parity Check Codes over GF(q)”, IEEE Communications Letters 1998
[2] Zhou et al. “Non-Binary LDPC Codes vs. Reed-Solomon Codes”, IEEE Information Theory and Applications Workshop, 2008
[3] Divsalar and Dolecek “Progress in Non-binary Protograph LDPC Codes with Short Blocklengths”, CCSDS Spring Meeting 2013

Benefits of Non-Binary LDPC (2)



Modulation symbols represent a number of bits.

By mapping modulation symbols to independent bits for binary decoding, the information concerning their correlation is lost.

NB-LDPC decoding avoids bit interpolation by direct processing of symbols.

Dependent on the modulation scheme additional performance gains up to several dB have been observed [1][2][3].

At the same time the complexity of the demapper is reduced [2].

[1] Boutillon et al. “Non-Binary Low-Density Parity-Check coded Cyclic Code-Shift Keying”, IEEE WCNC 2013

[2] Declercq et al. “Getting closer to MIMO capacity with Non-Binary Codes and Spatial Multiplexing”, IEEE GLOBECOM 2010

[3] Scholl et al. “ML vs. BP Decoding of Binary and Non-Binary LDPC Codes”, IEEE ISTC 2012

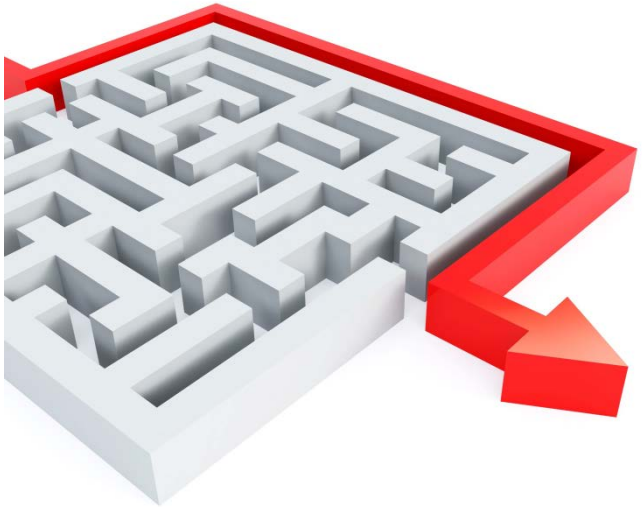
Decoding Algorithms for NB LDPC

Exact:

Sum Product Algorithm (SPA) [1]

FFT-Belief Propagation [2]

Integer Linear Programming [3]



Approximate:

Extended Min-Sum (EMS) [4]

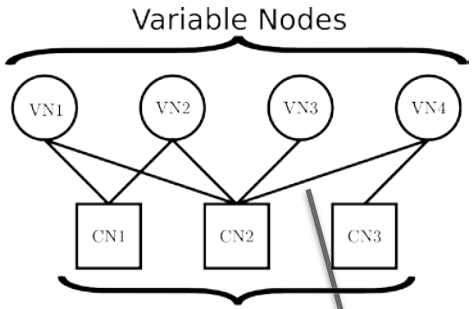
Min-Max [5]

Symbol Flipping [6]

Stochastic Decoding [7]

[1] Davey and MacKay “Low-Density Parity Check Codes oder GF(q)”, IEEE Communications Letters 1998
[2] Barnault and Declercq “Fast Decoding Algorithm for LDPC over GF(q)”, IEEE ITW 2003
[3] Scholl et al. “ML vs. BP Decoding of Binary and Non-Binary LDPC Codes”, IEEE ISTC 2012
[4] Voicila et al. “Low-Complexity and Low-Memory EMS Algorithm for Non-Binary LDPC Codes”, IEEE ICC 2007
[5] Savin “Min-Max decoding for non binary LDPC codes”, IEEE ISIT 2008
[6] Chen et al. “Two low-complexity reliability-based message-passing algorithms for decoding non-binary LDPC codes”, IEEE Transactions on Communications 2010
[7] Sarkis et al. “Stochastic Decoding of LDPC Codes over GF(q)”, IEEE ICC 2009

The EMS Decoding Algorithm



1001011
1011111
0100011
0001010
1001100
0100100
1000011
1001001

CN calculation in GF(q):

$Symbol$:

$$\forall i,j,k,l \in q : out_i = in_j^1 \oplus in_k^2 \oplus in_l^3 = i$$

LLR:

$$\forall i,j,k,l \in valid(q) : out_i = \min\left(in_j^1 + in_k^2 + in_l^3\right)$$

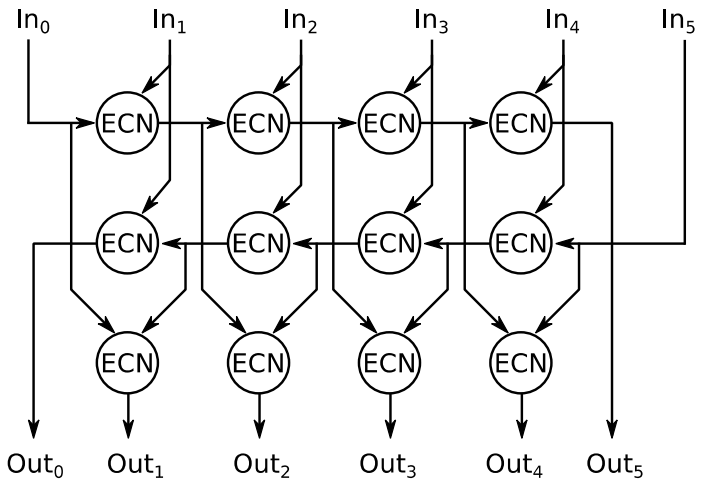
At high level EMS is like binary Min-Sum.

Messages are iteratively passed between VNs and CNs until a valid codeword is found.

Instead of binary messages, a GF(q) vector is transferred on each edge. EMS reduces the size of the GF(q) vector from q messages to nm messages with negligible loss in communications performance. E.g. only the most reliable messages are exchanged between VNs and CNs.

The most complex task in EMS decoding is the Check Node processing.

Forward Backward Check Node Processing

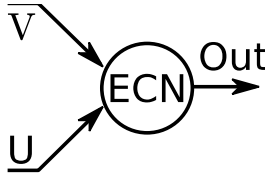


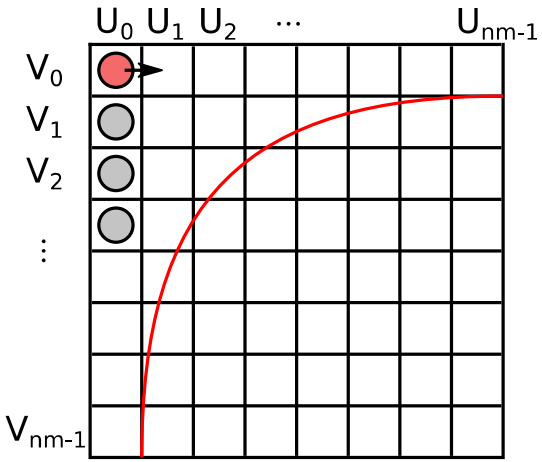
Forward Backward (FWBW) processing is the state-of-the-art check node algorithm.

With a divide and conquer approach using Elementary Check Nodes (ECN) the most reliable messages for each outgoing edge are computed. Each ECN considers two GF(q) vectors. The intermediate results are combined in a smart way to generate the output vectors.

The FWBW scheme allows for small hardware implementations but suffers from low throughput and high latency.

Elementary Check Node





The ECN transforms two input GF(q) vectors U and V into one output GF(q) vector consisting of the most reliable symbol combinations. For EMS the vector size is reduced from q elements to only nm < q elements.

The search space (nm^2 possibilities) can be represented by a matrix showing all possible combinations of the input symbols. Each field in the matrix can be computed as the sum of the according input reliabilities and GF values.

Under the assumption that the input vectors are sorted according to their reliability (e.g. U_0 and V_0 have the highest reliability), the most reliable values in the matrix are located left from and above the red line. Bubble Check [1] is an efficient implementation for the ECN. It executes a smart exploration of the matrix.

[1] E. Boutillon and Conde-Canencia “Bubble check: a simplified algorithm for elementary check node processing in extended min-sum non-binary LDPC decoders”, IEEE Electronic Letters 2010

New EMS Decoding Approaches

Several new directions are investigated to achieve more efficient architectures:

Symbol parallel FWBW check node

Same divide and conquer approach like FWBW.

Process N symbols per clock cycle, decoding speedup of factor N.

Minimal hardware increase in CN and VN to support approximately sorted messages.

Multi edge check node

Direct processing of multiple input vectors instead of FWBW approach.

Apply the Bubble Check algorithm in more than two dimensions.

Significantly reduced CN delay but high hardware cost for high CN degrees.

Truncated T-EMS check node

Efficient algorithmic transformation of the EMS algorithm for high CN degrees. The original algorithm requires full GF(q) vectors. We are currently investigating what is necessary to reduce the vector size.

Maximum Likelihood and Soft-Reed Solomon Decoding

Integer Programming Decoding [1]

Decoding as optimization problem:

variables **x**: bitvector (code word)

Objective function:

$$\min \sum_{i=0}^{N} \lambda_i x_i \qquad x_i \in \{0,1\}$$

coefficients: received LLRs

Constraints on **x:**
ensures that bitvector **x** is a code word

$$s.t. \quad H \vec{x} - 2 \vec{z} = 0$$

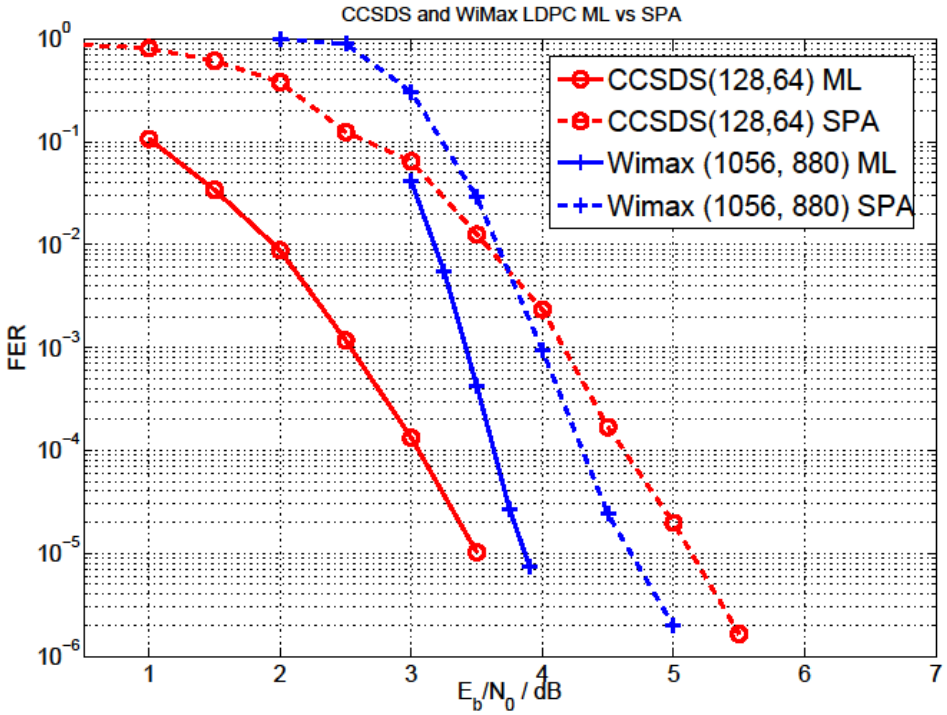
Parity check matrix H:

assistance variables **z** (>0, integer): account for mod 2 arithmetic

Variables **x** are bits -> integer program, IP (hard to solve!)
Real ML Decoder, not a heuristic
ML Monte Carlo Simulation of channel codes possible

[1] Scholl et al. “Efficient Maximum-Likelihood Decoding of Linear Block Codes on Binary Memoryless Channels”, IEEE ISIT 2014

Efficient Maximum Likelihood Decoding



Efficient Linear Programming Decoding

Integer Program (IP Formulation)

Our optimized solver algorithm:
10x faster than CPLEX

Solver uses: branch & bound

Database for Simulation Results:
www.uni-kl.de/channel-codes

Soft Decision Decoding for Reed-Solomon Codes (1)

Algorithm: Information Set Decoding

Variant: Ordered-Statistics Decoding

We use: Low Complexity Version [1]

Hard Decision Dec. added for better FER

1) ***Sorting:***
determine the M least reliable bit positions (LRP)

2) ***Gaussian Elimination:***
diagonalize $\mathbf{H}$ *at the LRP to obtain* $\hat{\mathbf{H}}$

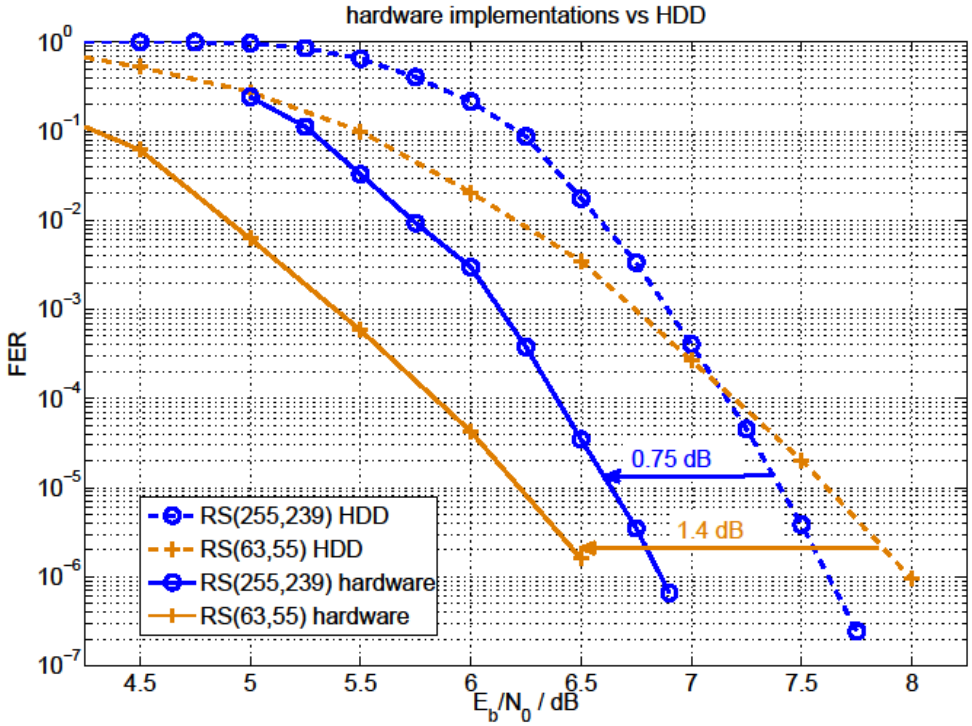
3) ***calculate the syndrome*** $\hat{\mathbf{s}} = \hat{\mathbf{H}}\bar{\mathbf{y}}^{\mathbf{T}}$
and its binary weight $wgt\,(\hat{\mathbf{s}})$

4) **If** $wgt\,(\hat{\mathbf{s}}) > \Theta$: /* MRP errors */
flip the received bit at position
$j = \underset{i=0,\ldots,N-1}{argmin}\ wgt\left(\hat{\mathbf{s}} \oplus \hat{\mathbf{h}}_{\mathbf{i}}\right)$
update the syndrome $\hat{\mathbf{s}} = \hat{\mathbf{s}} \oplus \hat{\mathbf{h}}_{\mathbf{j}}$ *and* $wgt\,(\hat{\mathbf{s}})$, goto 5

5) **If** $wgt\,(\hat{\mathbf{s}}) \leq \Theta$: /* only *LRP* errors remaining */
For all $\hat{s}_i = 1$, *flip the LRP* l, *for which* $\hat{h}_{il} = 1$
output OSD result, terminate
else
perform HDD on $\bar{\mathbf{y}}$ *and output HDD result,* terminate

[1] Shanbhag et al. “Performance analysis of the adaptive parity check matrix based soft-decision decoding algorithm”, IEEE 2004.

Soft Decision Decoding for Reed-Solomon Codes (2)



New Hardware Architecture

Information Set Decoding

Outstanding gain in FER over HDD
RS(255,239): 0.75 dB gain
RS(63,55): 1.4 dB gain

Hardware complexity lower than comparable state-of-the art soft decoders

FPGA implementations

State-of-the-art soft decoder RS(255,239)

	[1]	[2]	2014 [Scholl et al., 2014]
Algorithm	Adaptive BP	Stochastic Chase	Information Set
Communications Gain	0.75 dB	0.7 dB	0.75 dB
FPGA	Stratix II	Virtex 5	Virtex 5
Size in FPGA LUT	43,700	117,000	**13,700**
Throughput	4 Mbit	50 Mbit/s	**805 Mbit/s**

Our new architecture

Currently developed: Improved Architecture with a **gain of 1.3 dB**

[1]: Kan et al. “Hardware implementation of soft-decision decoding for Reed-Solomon code”, IntTurbo Codes and Related Topics Symp 2008
[2]: Gross et al. ”Stochastic chase decoder for reed-solomon codes”, IEEE NEWCAS 2012

Backup

NB-LDPC Implementation Efficiency

	[1]	Our work
GF(q)	64	2(binary)
Algorithm	EMS	Min-Sum
Throughput	1150 Mbit/s	1560 Mbit/s
Area	7 mm^2	1.8 mm^2
Energy	277 pJ/bit/Iter.	7.3 pJ/bit/Iter.

The area and energy efficiency (throughput / area and energy/bit) of state-of-the-art NB LDPC decoders is still orders of magnitude less than their binary counterparts.

Ongoing research is significantly reducing this difference.

The gain in communications performance is up to several dB.

[1]: Zhang et al. “A 1.15Gb/s Fully Parallel Nonbinary LDPC Decoder with Fine-Grained Dynamic Clock Gating”, IEEE ISSCC 2013