Reduction of complexity for Non-binary LDPC decoders with compressed messages

Jesús O. Lacruz, Francisco García-Herrero and Javier Valls, *Member IEEE*

Abstract

In this paper a method to compress the messages between the check nodes and the variable nodes is proposed. This method is named as compressed non-binary message-passing (CNBMP). The CNBMP reduces the number of messages exchanged between one check node and the connected variable nodes from $d_c \times q$ to $5 \times q$, and its application has a high impact in the performance of the decoder: the storage and routing area is reduced and the throughput is increased. Unlike other methods, the CNBMP does not introduce any approximation or modification in the information and the processed operations are exactly the same as the original decoders, hence, no performance degradation is introduced. To demonstrate its advantages, an architecture applying this CNBMP to the Trellis Min-max algorithm was derived showing that most of the storage resources were also reduced from $d_c \times q$ to $5 \times q$. This architecture was implemented for a (837,726) NB-LDPC code using a 90nm CMOS technology reaching a throughput of 981Mbps with an area of $10.67mm^2$, which is 3.9 more efficient than the best solution found in literature.

Index Terms

LDPC codes, decoding, non-binary, hardware implementation, high-throughput

I. Introduction

The two main bottlenecks of non-binary low-density parity-check (NB-LDPC) decoder architectures are the storage resources and the maximum throughput. Regardless their significant benefits, such as a better behaviour in the error floor region and a more robust correction for burst

F. García, and J. Valls are with the Instituto de Telecomunicaciones y Aplicaciones Multimedia, at Universitat Politècnica de València, 46730 Gandia, Spain (e-mail: fragarh2@epsg.upv.es, jvalls@eln.upv.es).

J. Lacruz is with the Electrical Engineering Department, Universidad de Los Andes, Mérida, 5101, Venezuela. (e-mail: jlacruz@ula.ve)

errors, NB-LDPC codes cannot compete with their binary counterparts in terms of complexity or throughput/area efficiency.

Several alternatives to the original Q-ary Sum-of-Product algorithm (QSPA) [1] were proposed during this last decade in order to keep the best correction performance possible and reduce complexity. The most remarkable ones are the Extended Min-Sum (EMS) [2] and Min-Max (MM) [3] algorithms, which reduced the complexity of the check node processor and the storage resources. However, a parallel implementation of these algorithms was prohibitive in terms of wiring between check node and variable node processors and arithmetic resources. For this reason all the architectures derived from these two algorithms applied the forward-backward metrics, which consist in a serial computation of the check node information. All the decoders based on the forward-backward suffer from a very large number of clock cycles per iteration, limiting the maximum throughput to a few Mbps [4].

In order to increase the degree of parallelism keeping the same error correction, a new version of the EMS algorithm named as Trellis-EMS (T-EMS) was proposed in [5]. This method allowed hardware designers to implement a fully parallel check node in a layered architecture [6]. This implementation did not sacrifice efficiency in terms of throughput/area compared to other serial implementations based on trellis [7] and increased throughput more than three times. Further improvements were introduced with the Trellis Min-max (TMM) in [8]. Despite this, the decoder from [8] required $14.7mm^2$ of area with a 90nm CMOS process and reached a throughput of 660Mbps, which is far from the results of modern binary LDPC decoders for the same technology ($9.6mm^2$, 45.42Gbps) [9]. While the binary architectures just exchange a number of messages equal to the degree of the check node (d_c) between check node and variable node, non-binary decoders require q times more wires/connections; and the same happens for the memories and registers, which are about the 80% of the decoder's area.

In this brief a method to reduce the number of messages exchanged in non-binary decoders between the check node and the variable node is introduced. This method does not vary the computation of the decoding algorithm nor reduces the information transferred between nodes, so it does not introduce any performance degradation. This proposal compresses the information transmitted in the message passing reducing the size of the messages from $d_c \times q$ to $5 \times q$. This has a great impact in both area and throughput specially for high rate codes. As an example, an implementation for the same code as in [8] and [7] achieves 981Mbps of throughput with an area of $10.6mm^2$ for a 90nm CMOS process.

The rest of the paper has four sections. Section II includes a summary of the NB-LDPC message-passing of the decoding algorithms. Section III describes the proposal of this work. Section IV shows the impact of the new message-passing in a hardware implementation and compares the results to other existing architectures. Section V outlines the conclusions.

II. Non-binary LDPC message passing

Let $\mathbf{H}$ be the $M \times N$ parity check matrix with coefficients $h_{i,j} \in GF(q)$ that defines an (N,K) NB-LDPC code. $\mathcal{N}(m)$ and $\mathcal{M}(n)$ are described as the sets that consist of all the non-zero elements of a row m (check node) and a column n (variable node) respectively. The size of the sets $\mathcal{N}(m)$ and $\mathcal{M}(n)$ are the degree of check node (d_c) and the degree of variable node (d_v). The d_c and d_v degrees represent the number of messages that each check node and variable node receive respectively. The set of messages from check node to variable node are denoted as $\mathbf{R}$ and the set of messages from variable node to check node are $\mathbf{Q}$. Each of these messages consists of q elements, due to the fact of performing operations over GF(q). The method to compute each of these sets depends on the decoding algorithm applied. The algorithms that provide a better performance with lower complexity are T-EMS and T-MM, which have a different processing at the check node but share the same operations at the variable node. To a better understanding of the message-passing between check node and variable node, a short explanation of the basics operations performed in the check node is included next, for more details about the different decoding processes we refer to [5] and [8].

In addition, to perform a parallel processing of the check node we will assume delta domain [5], [6] messages as inputs and outputs at the check node.

Let $\triangle\mathbf{Q}$ be the set of d_c messages from the variable node in delta domain defined as:

$$\triangle\mathbf{Q} = \{\triangle\mathbf{Q}_{m,n}\} \ , \ n \in \mathcal{N}(m) \ , \ m \in \mathcal{M} \tag{1}$$

Each element $\triangle\mathbf{Q}_{m,n}$ includes the likelihood of being the symbol $\alpha^x \in GF(q)$, $x = \{-\infty, 0, 1, \ldots, q-2\}$:

$$\triangle\mathbf{Q}_{m,n} = \{\mathcal{Q}_{m,n}(\alpha^{-\infty}), \mathcal{Q}_{m,n}(\alpha^0), \ldots, \mathcal{Q}_{m,n}(\alpha^{q-2})\} \tag{2}$$

The output messages of the check node in the delta domain are also of length d_c:

$$\triangle\mathbf{R} = \{\triangle\mathbf{R}_{m,n}\} \ , \ n \in \mathcal{N}(m) \ , \ m \in \mathcal{M} \tag{3}$$

The likelihood of each symbol to accomplish the parity check equation of the check node is defined as:

$$\triangle \mathbf{R}_{m,n} = \{\triangle \mathcal{R}_{m,n}(\alpha^{-\infty}), \triangle \mathcal{R}_{m,n}(\alpha^{0}), \ldots, \triangle \mathcal{R}_{m,n}(\alpha^{q-2})\} \tag{4}$$

To compute the reliability of each one of the q symbols in a single message, the check node update equations consider the combinations of the most reliable input messages. If only the two most reliable messages per symbol are considered the update rules for the check node follow the next conditions:

i) If the input likelihood of the symbol α^x for the edge $\{m, n\}$ is not the most reliable for α^x nor is considered to compute other α^y output message, $\triangle \mathcal{R}_{m,n}(\alpha^x)$ is equal to the most reliable value $\mathcal{Q}_{m,n_0}(\alpha^x)$:

$$\begin{aligned}\triangle \mathcal{R}_{m,n}(\alpha^x) = \{min(\mathcal{Q}_{m,n_0}(\alpha^x), \mathcal{Q}_{m,n_0}(\alpha^y) + \mathcal{Q}_{m,n_0}(\alpha^z))\}, \\ \alpha^y + \alpha^z = \alpha^x, \forall \alpha^y, \alpha^z \in GF(q) \leftrightarrow \\ [\mathcal{Q}_{m,n}(\alpha^x) \neq \mathcal{Q}_{m,n_0}(\alpha^x)] \bigwedge [\mathcal{Q}_{m,n_0}(\alpha^x) + \mathcal{Q}_{m,n_0}(\alpha^z) \neq \\ \triangle \mathcal{R}_{m,n}(\alpha^y)], \alpha^x + \alpha^z = \alpha^y, \forall \alpha^y, \alpha^z \in GF(q)\end{aligned} \tag{5}$$

Being $\mathcal{Q}_{m,n_0}(\alpha^x)$ and $\mathcal{Q}_{m,n_1}(\alpha^x)$:

$$\mathcal{Q}_{m,n_0}(\alpha^x) \leq \mathcal{Q}_{m,n_1}(\alpha^x) \leq \mathcal{Q}_{m,n}(\alpha^x) \ , \ \forall n \in \mathcal{N}(m) \setminus \{n_0, n_1\} \tag{6}$$

ii) If the input likelihood of the symbol α^x for the edge $\{m, n\}$ is the most reliable for α^x, $\triangle \mathcal{R}_{m,n}(\alpha^x)$ takes the value of the second more reliable message:

$$\triangle \mathcal{R}_{m,n}(\alpha^x) = \{\mathcal{Q}_{m,n_1}(\alpha^x)\} \leftrightarrow [\mathcal{Q}_{m,n}(\alpha^x) = \mathcal{Q}_{m,n_0}(\alpha^x)] \tag{7}$$

iii) If the input likelihood of the symbol α^x for the edge $\{m, n\}$ is involved in the output reliability of α^y, $\triangle \mathcal{R}_{m,n}(\alpha^x)$ takes the value of the most reliable message $\mathcal{Q}_{m,n_0}(\alpha^x)$:

$$\begin{aligned}\triangle \mathcal{R}_{m,n}(\alpha^x) = \{\mathcal{Q}_{m,n_0}(\alpha^x)\} \leftrightarrow [\mathcal{Q}_{m,n_0}(\alpha^x) + \mathcal{Q}_{m,n_0}(\alpha^z) = \\ = \triangle \mathcal{R}_{m,n}(\alpha^y)], \alpha^x + \alpha^z = \alpha^y, \forall \alpha^y, \alpha^z \in GF(q)\end{aligned} \tag{8}$$

To reduce the number of operations at the check node and share results a set that includes common computation was proposed in [5], and defined as:

$$\mathbf{P}_m = \{\mathbf{P}_m(\alpha^{-\infty}), \mathbf{P}_m(\alpha^{0}), \ldots, \mathbf{P}_m(\alpha^{q-2})\} \ , \ m \in \mathcal{M} \tag{9}$$

Where each element from the set $\mathbf{P}_m$ includes the two most reliable input values from α^x:

$$\mathbf{P}_m(\alpha^x) = \{\mathcal{P}_{m_0}(\alpha^x) = \mathcal{Q}_{m,n_0}(\alpha^x), \mathcal{P}_{m_1}(\alpha^x) = \mathcal{Q}_{m,n_1}(\alpha^x)\} \tag{10}$$

Based on the set $\mathbf{P}_m$ an extra set is computed in [5]. This set includes the values from $\triangle\mathcal{R}_{m,n}(\alpha^x)$ in equation (5). The set is defined as follows:

$$\mathbf{E}_m = \{\mathcal{E}_m(\alpha^{-\infty}), \mathcal{E}_m(\alpha^0), \ldots, \mathcal{E}_m(\alpha^{q-2})\} \ , \ m \in \mathcal{M} \tag{11}$$

$$\begin{aligned}\mathcal{E}_m(\alpha^x) = \{min(\mathcal{Q}_{m,n_0}(\alpha^x), \mathcal{Q}_{m,n_0}(\alpha^y) + \mathcal{Q}_{m,n_0}(\alpha^z))\} \\ (\alpha^y + \alpha^z = \alpha^x \in GF(q)) \bigwedge (\mathcal{Q}_{m,n_0}(\alpha^y) + \mathcal{Q}_{m,n_0}(\alpha^z) < \\ < \mathcal{Q}_{m,n_0}(\alpha^a) + \mathcal{Q}_{m,n_0}(\alpha^b)),\ \alpha^a + \alpha^b = \alpha^x, \\ \forall \alpha^a, \alpha^b \in GF(q) \backslash \{\alpha^y, \alpha^z\}\end{aligned} \tag{12}$$

Regardless the definition of the extra set the output messages of the check node are $\triangle\mathbf{R}_{m,n}$, which is a set of size $q \times d_c$.

III. Compressed Non-Binary Message-Passing (CNBMP)

With the aim of reducing the size of the sets that conform the messages shared between check node and variable node we propose a new ordering of the information. With these new sets the number of information exchanged between check node and variable node is reduced considerably and the set $\triangle\mathbf{R}_{m,n}$ is easily derived at the variable node. We name this method Compressed Non-Binary Message-Passing (CNBMP).

First we define the set $\mathbf{C}_m$ as follows:

$$\mathbf{C}_m = \{\mathbf{C}_m(\alpha^{-\infty}), \mathbf{C}_m(\alpha^0), \ldots, \mathbf{C}_m(\alpha^{q-2})\} \ , \ m \in \mathcal{M} \tag{13}$$

$$\mathbf{C}_m(\alpha^x) = \{\mathbf{N}_{x'}(m)\} \tag{14}$$

Each $\mathbf{N}_{x'}(m)$ element contains the index n of the edge $\{m, n\}$ for the symbol α^x in which $\triangle\mathbf{R}_{m,n}$ is not updated following equation (5):

$$\begin{aligned}\mathbf{N}_{x'}(m) = \{n_0\} \leftrightarrow [(\alpha^x \in GF(q)) \bigwedge (\mathcal{Q}_{m,n_0}(\alpha^x) = \\ = \mathbf{E}_m(\alpha^x))] \bigvee [(\alpha^x + \alpha^z = \alpha^y, \forall \alpha^y, \alpha^z \in GF(q)) \bigwedge \\ \bigwedge (\mathcal{Q}_{m,n_0}(\alpha^x) + \mathcal{Q}_{m,n_0}(\alpha^z) = \mathbf{E}_m(\alpha^y))]\end{aligned} \tag{15}$$

Considering that the sets $\mathbf{E}_m$ and $\mathbf{P}_m$ are computed the message $\triangle\mathbf{R}_{m,n}$ can be recovered at the variable node following the next equations:

$$\triangle\mathcal{R}_{m,n}(\alpha^x) = \mathcal{E}_m(\alpha^x)\ ,\ n \in \mathcal{N}(m)\backslash\mathbf{N}_{x'}(m) \tag{16}$$

$$\triangle\mathcal{R}_{m,n}(\alpha^x) = \mathcal{P}_{m_1}(\alpha^x) \leftrightarrow \mathcal{P}_{m_0}(\alpha^x) = \mathcal{E}_m(\alpha^x)\ ,\ n \in \mathbf{N}_{x'}(m) \tag{17}$$

$$\triangle\mathcal{R}_{m,n}(\alpha^x) = \mathcal{P}_{m_0}(\alpha^x) \leftrightarrow \mathcal{P}_{m_0}(\alpha^x) \neq \mathcal{E}_m(\alpha^x)\ ,\ n \in \mathbf{N}_{x'}(m) \tag{18}$$

It is important to remark that: i) whether CNBMP is applied or not the sets $\mathbf{P}_m$ and $\mathbf{E}_m$ are computed because of computational efficiency [5], so we are not adding any extra operation; and ii) it can be demonstrated that the value of the messages $\triangle\mathbf{R}_{m,n}$ are exactly the same applying equations (5) to (8) or (16) to (18), so in terms of error correction performance we can claim that CNBMP is equivalent to the original T-EMS or T-MM algorithms as it does not include any approximation.

Note that applying the CNBMP the output information of the check node is conformed by the set $\mathbf{E}_m$ that contains q elements and the sets $\mathbf{C}_m$ and $\mathbf{P}_m$ that contain $2 \times q$ elements each one. So in total the cardinality of the output information is $5 \times q$, unlike previous proposals found in literature.

To sum up, the check node with the CNBMP does not compute equations (5) to (8), but equations (16) to (18). In addition, the message passing consists of the sets $\mathbf{C}_m$, $\mathbf{P}_m$ and $\mathbf{E}_m$, not of $\triangle\mathbf{R}_{m,n}$, which is of size $d_c \times q$, as shown in Fig.1.

IV. Hardware impact of CNBMP

The first improvement for the hardware architectures of NB-LDPC decoders is the reduction of the wiring. According to the implementation reports, the maximum frequency of the decoder is not limited by the depth of the logic gates, but for the length of the wiring and the routing congestion. So, if we apply CNBMP, the wires between both check node and variable node processors will be reduced and hence, routing congestion will be mitigated. The reduction is $\lambda = (d_c \times q \times Q_b)/(3 \times q \times Q_b + 2 \times q \times \lceil log_2(d_c) \rceil)$ (Fig.2), assuming that the messages at the check node are quantized with Q_b bits and that the set $\mathbf{C}_m$ requires $\lceil log_2(d_c) \rceil$ bits to represent the indexes n. As it is shown next with this reduction of the routing there is an improvement in the maximum frequency.

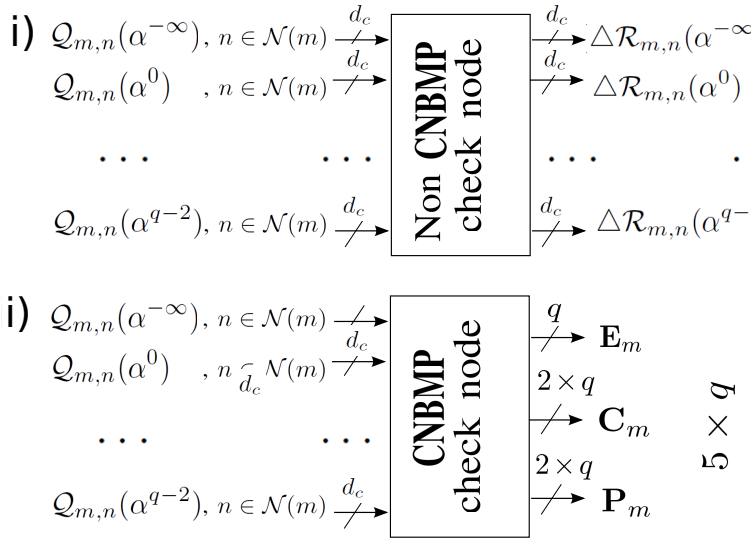

Fig. 1. i) Check node without CNBMP ii) Check node with CNBMP

TABLE I

COMPARISON OF THE PROPOSED NB-LDPC LAYERED DECODER WITH OTHER WORKS FROM LITERATURE

Algorithm	MS [10]	T-QSPA [11]	MM [12]	MM [7]	T-EMS [6]	T-MM [8]	T-MM CNBMP
Report (nm)	Syn. (180)	Layout (90)	Syn. (130)	Syn. (180)	Syn. (90)	Layout (90)	Syn/Layout (90)
Quantization (Q_b)	5 bits	7 bits	5 bits	5 bits	7 bits	6 bits	6 bits
Gate Count (NAND)	1.29M	8.51M	2.1M	871K	2.75M	3.28M	0.9M / 1.25M
f_{clk} (MHz)	200	250	500	200	250	238	333 / 300
Throughput (Mbps)	64	223	64	66	484	660	1089 / 981
Throughput (Mbps) 90 nm	149	223	107	154	484	660	1089 / 981
Efficiency 90 nm (Mbps/M-gates)	115.5	26.2	50.9	176.8	176	201	1210 / 784.8
Area (mm^2)	-	46.18	-	-	19	14.75	10.4 / 10.6

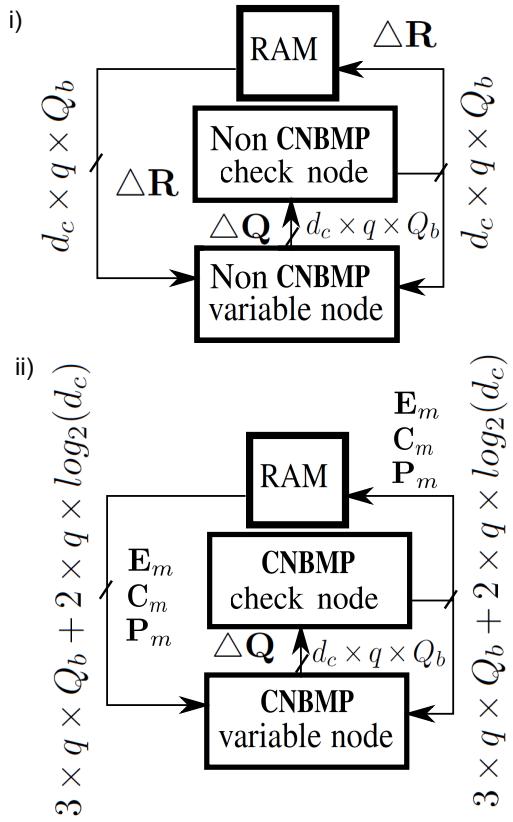

Fig. 2. i) Layered architecture of a NB-LDPC decoder without CNBMP. RAM memory from this architecture has M addresses of size $d_c \times q \times Q_b$ ii) Layered architecture of a NB-LDPC decoder with CNBMP. RAM memory from this architecture has M addresses of size $3 \times q \times Q_b + 2 \times q \times log_2(d_c)$

The second improvement is in terms of storage resources. To perform the layered schedule the decoder requires the storage, in registers or memories, of the information from the check node in the previous iteration, in order to compute the extrinsic information. Therefore, M addresses of depth equal to the size of the output messages from the check node are required. As it is previously explained, the number of the output messages without CNBMP is $d_c \times q \times Q_b$ and the number with CNBMP is equal to $3 \times q \times Q_b + 2 \times q \times \lceil log_2(d_c) \rceil$, so the reduction in storage resources is also λ (Fig.2). Note that applying CNBMP will be specially advantageous for high rate codes, where d_c is very large. However, even with low and medium rate codes there will be significant improvements, as far as the only requirement to get some complexity reduction is that $d_c > 5$. To de-compress the messages at the variable node comparators and multiplexors implement the conditions from equations (16) to (18) to select whether $\mathcal{E}_m(\alpha^x)$ or $\mathcal{P}_{m_0}(\alpha^x)$ and

$\mathcal{P}_{m_1}(\alpha^x)$ is applied to update $\triangle\mathcal{R}_{m,n}(\alpha^x)$.

In Table I we include the hardware results of the best architectures for NB-LDPC decoding and the results of our layered T-MM decoder with CNBMP. The code under test is for all the decoders the (N=837,K=726) NB-LDPC code over $GF(32)$, with $d_c = 27$ and $d_v = 4$ [13]. Cadence RTL Compiler was used for the synthesis and SOC encounter for place and route of the design employing a 90nm CMOS process of nine layers with standard cells and operating conditions of 25^oC and 1.2V. Compared with a conventional implementation of T-MM algorithm, CNBMP decoder improves the requirements of area due to the reduction of storage resources in the check-node, in a layered schedule. On the other hand, the clock frequency is increased owing to the reduction of the wiring congestion and the core area in general. Additionally, we eliminate some pipeline stages in the decoder thanks to the reduction in the complexity of the check-node processor and hence the critical path is also reduced. These facts contribute to increment the overall throughput of the decoder.

If we compare this work to the most efficient architectures found in literature [7] and [8], we can see that the maximum frequency is increased in 50% and 26% respectively due to the reduction of the routing congestion. On the other hand, area is about 43% larger than the decoder from [7] and 3 times smaller than the one in [8]. After applying the CNBMP the area of storage resources (RAM memories and registers) is reduce from 80% (2.2×10^6 NAND gates) of the total area in [8] to 50% (0.62×10^6 NAND gates). About the throughput, the CNBMP proposal is 1.48 times faster than the T-MM decoder in [8] and 14.8 times faster than the Min-max from [7]. In terms of efficiency Throughput/Area the decoder with CNBMP is 3.9 times more efficient than [7] and [8]. For the gate count, we consider the equivalence of one bit of RAM equals to 1.5 NAND gates and one register equals to 4.5 NAND gates.

Finally, if we compare CNBMP to the binary LDPC decoder from [9], which has a gate count of 3.4 millions of equivalent NAND gates and a throughput of 45.42Gbps for a code with a similar rate and half codeword length in terms of bits ((2048, 1723) LDPC code), CNBMP has 2.72 times less gates and reaches 17.46 times less throughput[1]. So, in terms of Throughput/Area efficiency, our non-binary decoder is 6.32 times less efficient than the binary one. Even not reaching the efficiency of a binary decoder, with CNBMP we reduce the difference to less than q, which is a good step forward compared to solutions like the one in [8] that has $2 \times q$ times lower efficiency..

V. Conclusions

In this paper a new message-passing definition is proposed for NB-LDPC decoders. This method reduces the number of the messages exchanged between check node and variable node, simplifying the routing of the derived hardware architectures and saving a big percentage of storage resources. Moreover, the new message passing does not modify the processing of the information at the decoder, keeping the same error correction performance as the original message-passing.

VI. Acknowledgement

This research was supported by the Spanish Ministerio de Ciencia e Innovación, under Grant No. TEC2011-27916.

F. García-Herrero has a FPU grant sponsored by the Spanish Ministerio de Educación (Grant No. AP2010-5178).

References

[1] M. Davey and D. MacKay, "Low-density parity check codes over GF(q)," *IEEE Communications Letters*, vol. 2, no. 6, pp. 165–167, 1998.

[2] D. Declercq and M. Fossorier, "Decoding Algorithms for Nonbinary LDPC Codes Over GF(q)," *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 633–643, 2007.

[3] V. Savin, "Min-Max decoding for non binary LDPC codes," in *IEEE International Symposium on Information Theory*, 2008, pp. 960–964.

[4] J. Lin, J. Sha, Z. Wang, and L. Li, "Efficient Decoder Design for Nonbinary Quasicyclic LDPC Codes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 5, pp. 1071–1082, May 2010.

[5] E. Li, D. Declercq, and K. Gunnam, "Trellis-Based Extended Min-Sum Algorithm for Non-Binary LDPC Codes and its Hardware Structure," *IEEE Transactions on Communications*, vol. 61, no. 7, pp. 2600–2611, 2013.

[6] E. Li, D. Declercq, K. Gunnam, F. García-Herrero, J. Lacruz, and J. Valls, "Low Latency T-EMS Decoder for NB-LDPC Codes," in *Conference Record of the Forty Seventh Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, 2013.

[7] F. Cai and X. Zhang, "Relaxed Min-Max Decoder Architectures for Nonbinary Low-Density Parity-Check Codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. PP, no. 99, pp. 1–1, 2012.

[8] J. Lacruz, F. Garcia-Herrero, D. Declercq, and J. Valls, "Simplified Trellis Min-Max Decoder Architecture for Nonbinary Low-Density Parity-Check Codes," vol. PP, no. 99, 2014, pp. 1–1.

[9] C.-C. Cheng, J.-D. Yang, H.-C. Lee, C.-H. Yang, and Y.-L. Ueng, "A Fully Parallel LDPC Decoder Architecture Using Probabilistic Min-Sum Algorithm for High-Throughput Applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 9, pp. 2738–2746, Sept 2014.

[10] X. Chen and C.-L. Wang, "High-Throughput Efficient Non-Binary LDPC Decoder Based on the Simplified Min-Sum Algorithm," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 11, pp. 2784 –2794, nov. 2012.

[11] Y.-L. Ueng, K.-H. Liao, H.-C. Chou, and C.-J. Yang, "A High-Throughput Trellis-Based Layered Decoding Architecture for Non-Binary LDPC Codes Using Max-Log-QSPA," *IEEE Transactions on Signal Processing*, vol. 61, no. 11, pp. 2940–2951, 2013.

[12] J. Lin and Z. Yan, "Efficient Shuffled Decoder Architecture for Nonbinary Quasi-Cyclic LDPC Codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 9, pp. 1756–1761, 2013.

[13] B. Zhou, J. Kang, S. Song, S. Lin, K. Abdel-Ghaffar, and M. Xu, "Construction of non-binary quasi-cyclic LDPC codes by arrays and array dispersions - [transactions papers]," *IEEE Transactions on Communications*, vol. 57, no. 6, pp. 1652–1662, 2009.