SUM-SUBTRACT FIXED POINT LDPC DECODER

L. ARNONE[†], C. GAYOSO[†], C. GONZÁLEZ[†] and J. CASTIÑEIRA[‡]

[†]*Laboratorio de Componentes Electrónicos, Facultad de Ingeniería, UNMDP, Argentina.*
leoarn@fi.mdp.edu.ar
[‡] *Laboratorio de Comunicaciones, Facultad de Ingeniería, UNMDP, Argentina*
casti@intec.edu.ar

Abstract— **In this paper a low complexity logarithmic decoder for a LDPC code is presented. The performance of this decoding algorithm is similar to the original decoding algorithm´s, introduced by D. J. C. MacKay and R. M. Neal. It is a simplified algorithm that can be easily implemented on programmable logic technology such as FPGA devices because of its use of only additions and subtractions, avoiding the use of quotients and products, and of float point arithmetic. The algorithm yields a very low complexity programmable logic implementation of a LDPC decoder with an excellent BER performance.**

Keywords— **low density parity check codes, decoding, BER performance, look-up tables**

I. INTRODUCTION

A method for decoding Low Density Parity Check Codes (LDPC) is the sum-product algorithm proposed by Gallager (Gallager, 1962). Quotients and products involved in this algorithm make difficult the implementation of optimal LDPC decoders on low complexity programmable logic.

In this paper we propose a very low complexity sum-subtract fixed point decoding algorithm for LDPC codes. This algorithm also uses two look-up tables.

A comparison is done between the BER performance of the proposed decoding algorithm, and the BER performance of Gallager's sum-product decoding algorithm (Gallager, 1962) for a given LDPC code.

Results show that there is no significant difference in BER performance between the optimal and the proposed algorithm. The proposed algorithm is characterized by a very low complexity implementation, thus becoming a better alternative for its programmable logic implementation than the traditional sum-product algorithm.

This paper is organized as follows: Section II shows the main aspects of a LDPC decoder. Section III introduces the proposed algorithm. Section IV is related to the look-up tables utilized in the proposed algorithm. Section V is devoted to a comparative complexity analysis for these two LDPC decoding algorithms. Section VI shows BER performance results, and finally Section VII deals with the conclusions.

II. LDPC CODES

LDPC codes (Gallager, 1962) are a powerful class of linear block codes characterized by a parity check matrix $\mathbf{H}$, which fits the condition $\mathbf{H} \circ \mathbf{x} = \mathbf{0}$ for any codeword $\mathbf{x}$. A LDPC decoder is essentially a decoding algorithm (MacKay and Neal, 1997; MacKay, 1999) designed for finding a codeword $\hat{\mathbf{d}}$ (an estimate of the codeword $\mathbf{x}$), able to fit the condition:

$$\mathbf{H} \circ \hat{\mathbf{d}} = \mathbf{0} \tag{1}$$

The LDPC decoding algorithm is described over a bipartite graph depicted considering the relationship between the symbol nodes $d(j)$, which represent the bits or symbols of the code vector $\boldsymbol{x}$, and the parity check nodes $f(i)$, which represent the parity equations described in matrix $\mathbf{H}$. In this iterative process, each symbol node $d(j)$ sends to a parity check node $f(i)$ the estimation q_{ij}^{x} that this node generates with the information provided by all others parity check nodes connected to it, based on the fact that the parity check node j is in state x.

Then, each parity check node $f(i)$ sends the estimation r_{ij}^{x} to each symbol node $d(j)$ generated with the information provided by the other symbol nodes connected to it, based on the fact that the parity check node i condition is satisfied, if the symbol node $d(j)$ is in state x. This is an iterative process in which information is interchanged between these two types of nodes. This iterative process is stopped when the condition described by Eq. (1) is satisfied. In this case the corresponding decoded codeword is considered a valid codeword. Otherwise, the decoding algorithm stops after a given number of iterations are performed. In this case the decoded word may or may not be a codeword.

III. THE PROPOSED ALGORITHM

The bases of the LDPC decoding algorithm are described in (MacKay and Neal, 1997). The proposed simplification makes this algorithm operate using only additions and subtractions. This simplification makes use of a logarithmic version of the calculations involved in the original algorithm. The proposed algorithm is procedure based on the fact that a given number z,

Table 1. Summarises the add-subtract decoding

Sum-Product Algorithm	Add-Sub Expressions
Initialization $q_{ij}^{x} = p_{j}^{x}$	$\lvert wq_{ij}^{0}\rvert = \lvert wp_{j}^{0}\rvert$, $\lvert wq_{ij}^{1}\rvert = \lvert wp_{j}^{1}\rvert$
Horizontal Step. Calculate: $r_{ij}^{x} = e^{-\lvert wr_{ij}^{x}\rvert}$ with $x = 0,1$	
Horizontal Step 1	$\lvert w\delta q_{ij}\rvert = \min\left(\lvert wq_{ij}^{0}\rvert,\lvert wq_{ij}^{1}\rvert\right) + f_{-}\left(\lvert wq_{ij}^{0}\rvert,\lvert wq_{ij}^{1}\rvert\right)$ and $s_{ij} = 0$ *if* $\lvert wq_{ij}^{0}\rvert \le \lvert wq_{ij}^{1}\rvert$ else $s_{ij} = 1$
Horizontal Step 2	$\lvert w\delta r_{ij}\rvert = \sum_{j' \in N(i)\setminus j} \lvert w\delta q_{ij'}\rvert$ and $s\delta r_{ij} = \sum_{j' \in N(i)\setminus j} s_{ij'}$
Horizontal Step 3	if $s\delta r_{ij}$ is even $\lvert wr_{ij}^{0}\rvert = \ln(2) - f_{+}\left(\lvert w\delta r_{ij}\rvert,0\right)$ $\lvert wr_{ij}^{1}\rvert = \ln(2) + f_{-}\left(\lvert w\delta r_{ij}\rvert,0\right)$ if $s\delta r_{ij}$ is odd: $\lvert wr_{ij}^{0}\rvert = \ln(2) + f_{-}\left(\lvert w\delta r_{ij}\rvert,0\right)$ $\lvert wr_{ij}^{1}\rvert = \ln(2) - f_{+}\left(\lvert w\delta r_{ij}\rvert,0\right)$
Vertical Step Calculate: $q_{ij}^{x} = e^{-\lvert wq_{ij}^{x}\rvert}$ with $x = 0,1$	$\lvert wc_{ij}^{x}\rvert = \lvert w\,p_{j}^{x}\rvert + \sum_{i' \in M(j)\setminus i} \lvert wr_{i'j}^{x}\rvert$ $\lvert wq_{ij}^{0}\rvert = \lvert wc_{ij}^{0}\rvert - \min\left(\lvert wc_{ij}^{0}\rvert,\lvert wc_{ij}^{1}\rvert\right) + f_{+}\left(\lvert wc_{ij}^{0}\rvert,\lvert wc_{ij}^{1}\rvert\right)$ $\lvert wq_{ij}^{1}\rvert = \lvert wc_{ij}^{1}\rvert - \min\left(\lvert wc_{ij}^{0}\rvert,\lvert wc_{ij}^{1}\rvert\right) + f_{+}\left(\lvert wc_{ij}^{0}\rvert,\lvert wc_{ij}^{1}\rvert\right)$
Vertical Step 2. A posteriori estimation: $q_{j}^{x} = e^{-\lvert wq_{j}^{x}\rvert}$	$\lvert wc_{j}^{x}\rvert = \lvert wc_{ij}^{x}\rvert + \lvert wr_{ij}^{x}\rvert$ $\lvert wq_{j}^{0}\rvert = \lvert wc_{j}^{0}\rvert - \min\left(\lvert wc_{j}^{0}\rvert,\lvert wc_{j}^{1}\rvert\right) + f_{+}\left(\lvert wc_{j}^{0}\rvert,\lvert wc_{j}^{1}\rvert\right)$ $\lvert wq_{j}^{1}\rvert = \lvert wc_{j}^{1}\rvert - \min\left(\lvert wc_{j}^{0}\rvert,\lvert wc_{j}^{1}\rvert\right) + f_{+}\left(\lvert wc_{j}^{0}\rvert,\lvert wc_{j}^{1}\rvert\right)$
Estimation of Decoded Symbol $\hat{d}_{j}$ $d_{j} = \max\left(q_{j}^{x}\right) = \max\left(e^{-\lvert wq_{j}^{x}\rvert}\right)$	$\hat{d}_{j} = 0$ if $\lvert wq_{j}^{0}\rvert < \lvert wq_{j}^{1}\rvert$, else $\hat{d}_{j} = 1$

such that $z \le 1$, can be represented as:

$$z = e^{-|wz|} \; ; \; |wz| = |ln(z)| \qquad (2)$$

The algorithm has two part in wich quantities $|wq_{ij}|$ and $|wr_{ij}|$ associated with q_{ij} and r_{ij} are iteratively updated. Table 1 summarises the add-subtract decoding procedure. Where $f_{+}(|a|,|b|)$ and $f_{-}(|a|,|b|)$ are obtained by using a look-up table (Woodard and Hanzo, 2000; Bhatt *et al.*, 2000). The set of indexes of all the symbol nodes $d(j)$ related to the parity check node $f(i)$ will be denoted as $N(i)$, and $N(i)\setminus j$ will be the same set but excluding the index j . $M(j)$ is the set of sub indexes of all parity check nodes $f(i)$ related to the symbol node $d(j)$, and $M(j)\setminus i$ will be the same set but excluding the index i . The LDPC decoding algorithm involves the evaluation of the following steps:

A. Initialization

The initialization process is done by setting the values of estimations q_{ij}^{x} to the A Priori Probabilities of the symbols p_{j}^{x} . The A Priori Probability p_{j}^{x} is the probability of the j_{th}-symbol node adopting the value of x, with $x = \{0,1\}$. Since $p_j^x = e^{-\left|wp_j^x\right|}$ is always less than or equal to one, then $q_{ij}^x = e^{-\left|wq_{ij}^x\right|}$ also fits such condition, thus being only necessary to initialize variables $\left|wq_{ij}^0\right|$ and $\left|wq_{ij}^1\right|$ with the values $\left|wp_j^0\right|$ and $\left|wp_j^1\right|$ respectively.

B. Horizontal Step

Compute $\left|wr_{ij}^0\right|$ and $\left|wr_{ij}^1\right|$ for each i, j.

C. Vertical Step

For each i, j the quantities $\left|wq_{ij}^0\right|$ and $\left|wq_{ij}^1\right|$ are evaluated. Then A Posteriori Estimation $\left|wq_j^0\right|$ and $\left|wq_j^1\right|$ are updated.

D. Estimation of Decoded Symbol

Finally, an estimation of the decode bit $\hat{d}_j$ can be done using A Posteriori Estimation $\left|wq_j^0\right|$ and $\left|wq_j^1\right|$. If $\mathbf{H} \circ \hat{\mathbf{d}} = \mathbf{0}$ then the decoding algorithm halts. Otherwise the algorithm repeats from the horizontal step.

IV. LOOK-UP TABLES IMPLEMENTATION

The performance of the proposed decoding algorithm is set by the characteristics of the look-up tables $f_+(|a|,|b|)$ and $f_-(|a|,|b|)$. Assuming that the maximum number of bits used to construct these tables is c, the maximum number of entries of these tables is of size $N = 2^c$.

V. COMPLEXITY ANALYSIS

If N is the number of columns of the matrix $\mathbf{H}$, and t is the average number of ones per column of that matrix, the sum-product decoding algorithm (MacKay and Neal, 1997; Ping and Leung, 2000) involves the calculation of $6 \cdot N \cdot t$ products and $5 \cdot N \cdot t$ sums (average).

The proposed algorithm requires $14 \cdot N \cdot t$ sums, $3 \cdot N \cdot t$ subtracts, and $4 \cdot N \cdot t$ accesses to the look-up tables. In spite of requiring more sums than the traditional decoding algorithm, the complexity of the proposed algorithm implementation is highly reduced due to the fact of operating with neither quotients nor products.

VI. RESULTS

The proposed decoder for a LDPC code has been implemented for two LDPC codes, one with a parity check matrix $\mathbf{H_1}$ of 60 rows and 30 columns and another one with a parity check matrix $\mathbf{H_2}$ of 1008 rows and 504 columns. Look-up tables make use of $c = 16$, so that the number of entries can be as high as $N = 2^c = 65536$, and the maximum value in these tables is 65535. Matrix $\mathbf{H_1}$ has been randomly generated and tested in order to get the best one in terms of BER performance using the traditional algorithm (MacKay and Neal, 1997).

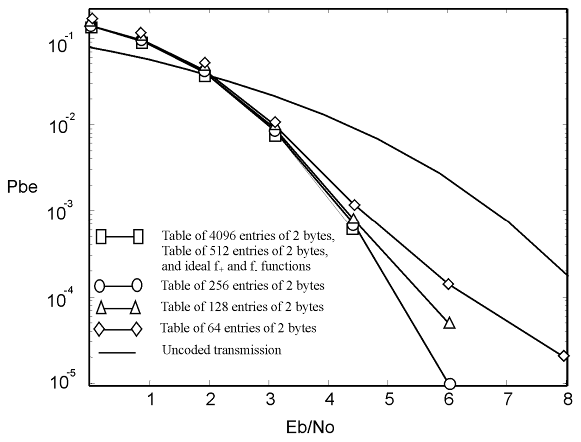

Fig. 1. BER performance of a LDPC code with a parity check matrix $\boldsymbol{H_1}$ of size $60x30$ for different sizes of the look-up tables

The BER performance has been evaluated using the proposed algorithm for different sizes of the look-up tables, assuming that each entry is an integer number

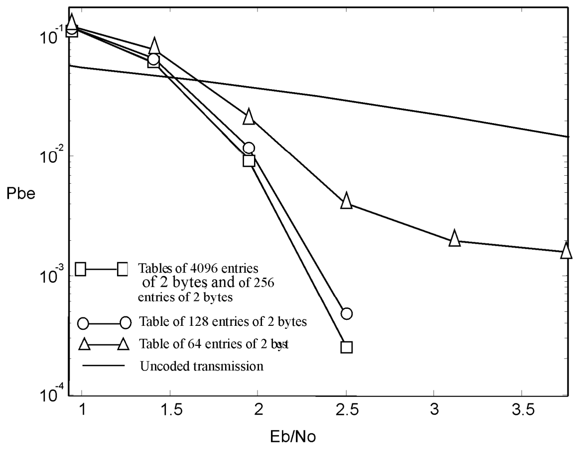

Fig. 2. BER performance of a LDPC code with a parity check matrix $\boldsymbol{H_2}$ of size $1008x504$ for different sizes of the look-up tables

represented in binary format using 2 bytes. As seen in Fig. 1, for the LDPC code using a small parity check matrix $\mathbf{H_1}$ there is no significant loss in BER performance using the proposed algorithm, if the size of each of the two tables is of 256 entries of 2 bytes, or larger.

The use of tables of size 512 or larger do not show differences with respect to the use of the ideal functions. The BER performance of a more practical LDPC code that uses a parity check matrix $\mathbf{H_2}$ is shown in Fig. 2.

As it may be noticed there is no significant loss in BER performance using the proposed algorithm, if the size of each of the two tables is of 128 entries of 2 bytes, or larger. The use of tables of size 256 or larger do not show differences with respect to the use of the ideal functions. Therefore, it is possible to implement a low complexity decoding algorithm without significant BER performance loss, by using the proposed logarithmic decoder with two look-up tables of reasonable size.

In comparison with other simplified decoding algorithms, we can say that the proposed sum-subtract fixed point decoding algorithm performs slightly better than similar ones presented by Fossorier (Fossorier *et. al.*, 1999), called UMP APP-based decoder and UMP BP-based decoder, where a simulation is done over a similar LDPC code, (a (1008,504) LDPC code) making use of 50 iterations. In our case we use 16 iterations, and we also take into account the channel information.

VII. CONCLUSION

In this paper a low complexity decoding algorithm for LDPC codes is presented. A comparison is done with respect to the traditional decoding algorithm (MacKay and Neal, 1997) to show that the BER performance of the proposed algorithm is close to the traditional decoding algorithm's, if the involved look-up tables are constructed appropriately.

The proposed decoding algorithm shows very low complexity, being thus suitable for programmable logic implementations. It makes use only of sums and substractions over fixed point arithmetic, thus avoiding the use products or quotients, and of float point arithmetic.

REFERENCES

Bhatt T., K. Narayanan, and N. Kehtarnavaz. "Fixed Point DSP Implementation of Low-Density Parity Check Codes." *Proc 9th IEEE DSP2000 Workshop*, (2000).

Gallager R. G. "Low Density Density Parity Check Codes." *IRE Trans. Information Theory*, **IT-8**, 21-28, (1962).

MacKay D. J. C. and M. Neal, R. "Near Shannon limit performance of low density parity check codes." *Electronics Letters*, **33**, 457-458, (1997).

MacKay D.J.C. "Good Error-Correcting Codes based on Very Sparse Matrices." *Transactions on Information Theory*, **45**, 399-431, (1999).

Ping L. and W. K. Leung. "Decoding Low Density Parity Check Codes with Finite Quantization Bits." *IEEE Communications Letters*, **4**, 62-64 (2000).

Woodard J.P. and L. Hanzo. "Comparative Study of Turbo Decoding Techniques." *IEEE Transaction on Vehicular Technology*, **49**, (2000).

Fassorier P. C., M. Mihaljević, and H. Imai. "Reduced Complexity Iterative Decoding of Low-Density Parity Check Codes Based on Belief Propagation." *IEEE Transaction on communications,* **47**, 673-680, (1999)

Received: April 14, 2006.
Accepted: September 8, 2006.
Recommended by Special Issue Editors Hilda Larrondo, Gustavo Sutter.