International Journal of Engineering & Technology, 3 (4) (2014) 451-456
©Science Publishing Corporation
www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v3i4.3185
Research Paper

Conception of a new LDPC decoder with hardware implementation on FPGA card

El Habti El Idrissi Anas [1]*, El Gouri Rachid [1, 2], Hlou Laamari [2]

[1] *Laboratory of Electrical Engineering and Energy System. Faculty of Sciences, University Ibn Tofail Kenitra, Morocco*
[2] *National School of Applied Sciences, University Ibn Tofail Kenitra, Morocco*
**Corresponding author E-mail: elhabtidrissi@gmail.com*

Copyright © 2014 El Habti El Idrissi Anas et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

Low Density Parity-Check codes are one of the hottest topics in coding theory nowadays. Equipped with very fast encoding and decoding algorithms, LDPC codes are very attractive both theoretically and practically. In this paper, A simplified algorithm for decoding Low-Density Parity-Check (LDPC) codes is proposed with a view to reduce the implementation complexity, this algorithm is based on a simple matrix equation which must be resolved in order to calculate all possible solutions of this equation, and then a simple circuit will be used to determine the errors produced during the transmission channel. First, we developed the design of the proposed algorithm second, we generated and simulated the hardware description language source code using Quartus software tools and finally we implemented the new algorithm of LDPC codes on FPGA card.

Keywords: *Bit-Flipping Algorithm, Error Detection, FPGA Card, LDPC Decoder, Matrix Equation*.

1. Introduction

This Low density parity-check code (LDPC) is an error correcting code used in noisy communication channel to reduce the probability of loss of information. With LDPC, this probability can be reduced to as small as desired, thus the data transmission rate can be as close to Shannon's limit as desired. LDPC was developed by Robert Gallager [1] in his doctoral dissertation at MIT in 1960 [2]. It was published 3 years later in MIT Press. Due to the limitation in computational effort in implementing the coder and decoder for such codes and the introduction of Reed-Solomon codes, LDPC was ignored for almost 30 years. During that long period, the only notable work done on the subject was due to R. Michael Tanner in 1981 where he generalized LDPC codes and introduced a graphical representation of the codes later called Tanner graph. Since 1993, with the invention of turbo codes, researchers switched their focus to finding low complexity code which can approach Shannon channel capacity. LDPC was reinvented with the work of Mackay, and Luby. Nowadays, LDPC have made its way into some modern applications such as 10GBase-T Ethernet, WiFi, WiMAX, Digital Video Broadcasting (DVB) [3], [4]. In this paper, a low complexity LDPC decoding algorithm is proposed to achieve a trade-off between implementation complexities compared to the bit-flipping algorithm (BFA) [5]. The algorithm is based on a simple matrix equation which must be resolved to calculate all possible solutions of this equation, and then a simple circuit which developed in VHDL will be used to select the correct code word that is found among the solutions of matrix equation. The rest of the paper is organized as follows: an overview of LDPC decoding is provided in section 2. Section 3 discusses the proposed algorithm and simulation. Finally, FPGA implementation details, hardware performance results are presented in section 4, followed by a conclusion.

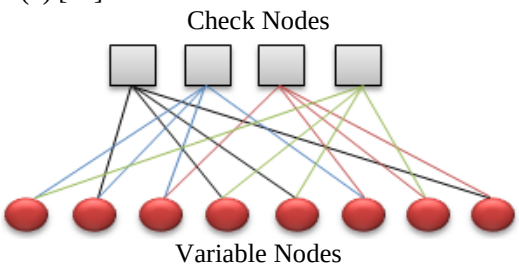
2. Overview of LDPC decoding

LDPC codes belong to a class of block codes. As their name suggests, its parity-check matrix (H) consists of very small number of non-zero elements. The sparseness of H determines the decoding complexity and the minimum distance of the code. Apart from the requirement that the LDPC matrix be sparse, there is no other difference between the LDPC

code and any other block code. An LDPC matrix is described by various parameters, which are briefly described here. An encoded message also known as code word consists of the useful information bits and the redundant bits. The code rate of information bits and the redundant bits. The code rate of a decoder is the ratio of the length of useful information bits to the length of code word bits. The number of nonzero entries in each of the rows and columns of H matrix is collectively known as degree distribution. An H matrix is said to be regular if the degree distribution of rows and columns are uniform, otherwise it is Irregular. The H matrix can be represented as a graph called Tanner graph [9]. A cycle in the graph is a sequence of connected nodes, which start and end at the same node. The girth or the smallest cycle in the parity-check graph, significantly contributes to the performance of the iterative decoding algorithms. A regular (2, 4) parity-check matrix with 8-bit code length is shown in Fig. 1 (a) and the corresponding Tanner graph representation of the parity check matrix is shown in Fig. 1 (b) [10].

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

a) An LDPC Matrix



b) Tanner graph representation of LDPC Matrix

Fig. 1: Tanner Graph Representation of LDPC Matrix.

3. Decoding

Different authors come up independently with more or less the same iterative decoding algorithm. They call it different names: the sum-product algorithm, the belief propagation algorithm, and the message passing algorithm. There are two derivations of this algorithm: hard-decision and soft-decision schemes. In this paper, a Low complexity LDPC Decoding algorithm that can achieve very good performance compared to a low complexity Bit-Flip algorithm that suffers from poor performance is presented.

3.1. Bit-flipping algorithm

The Bit-Flip algorithm is based on hard-decision message passing technique [8]. A binary hard-decision is done on the received channel data and then passed to the decoder. The messages passed between the check node and variable nodes are also single-bit hard-decision binary values. The variable node (V) sends the bit information to the connected check nodes (C) over the edges. The check node performs a parity check operation on the bits received from the variable nodes. It sends the message back to the respective variable nodes with a suggestion of the expected bit value for the parity check to be satisfied [6].

The algorithm:

- **Step 1:** We use the equation s = r. Ht to calculate the syndrome with the received vector. If the elements in the set of s are all zeros, it's terminated with the correct vector, otherwise, go to the next step.
- **Step 2:** Calculate the set of $\{f_0, f_1, \cdots, f_{N-1}\}$ and find the largest fj .Then transfer the corresponding r_j to its opposite number (0 or 1), get a new vector r'.
- **Step 3:** Calculate the vector $S = r\ H^t$ with the new vector r'. If the elements of s are all zeros or the iterations reach the maximum number, the decoding is terminated with the current vector, otherwise, the decoding go back to step 2.

Clearly the BFA has simple check node and variable node operations, thus making it a very low complexity decoding algorithm compared to the other algorithms. But this advantage comes with a poor decoding performance.

3.2. Proposed algorithm

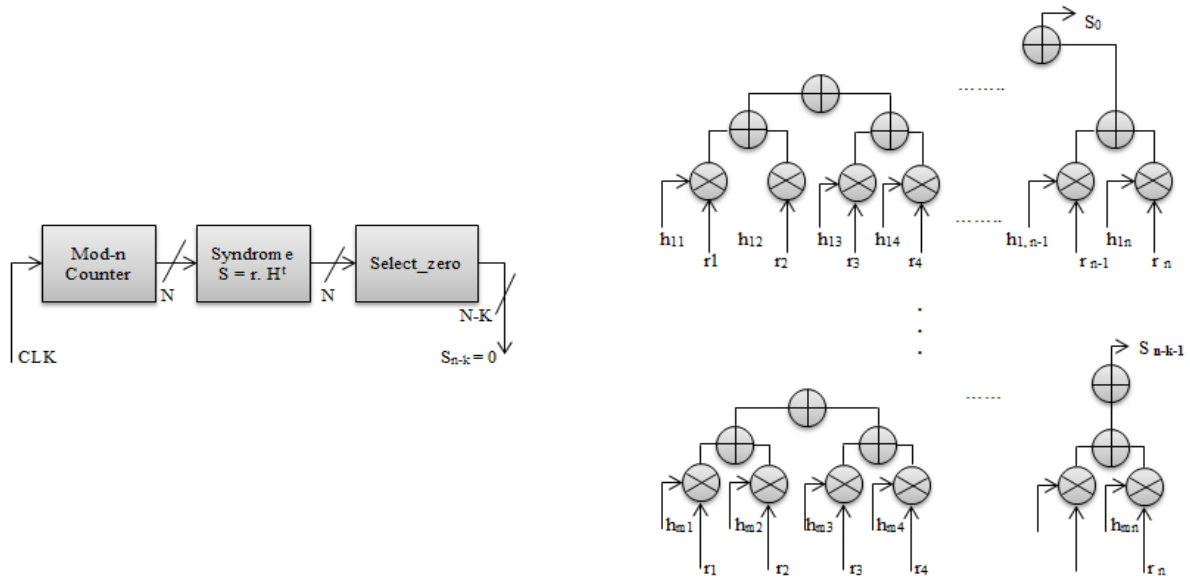
First this algorithm consists to solve the matrix equation $S = r.\ H^t = 0$ to calculate all possible solutions of this equation, and then a simple logic gate XOR will be used to determine the errors produced during the transmission channel [7]. The advantage of this algorithm is that the correct code word must be found among the solutions of the matrix equation $S = r.\ H^t = 0$, where S is the syndrome.

The proposed algorithm:

- **Step 1:** We use the equation $S = r \cdot Ht = 0$ to calculate all possible solutions.
- **Step 2:** Calculate the errors produced during the transmission channel using a simple logic gate XOR.
- **Step 3:** Calculate the error using the equation S = e. Ht = 0.
- **Step 4:** Calculate the correct code word using the logic gate XOR.

3.2.1. Scheme of matrix equation

To calculate all solutions of matrix equation S = r. H^t = 0, we use the circuit represented in Fig.2 (a) and (b).



a) Scheme of Matrix equation b) Logic circuit of Matrix equation

Fig. 2: Logic Circuit Of Matrix Equation For LDPC (N, K) Code.

Where $\{H_{ij}\}$ i = 1, M and j = 1, N the elements of the parity check matrix H. $\{R_i\}$ i = 1, N represent all possible code words which are generated by a counter Mod-n. The circuit represented in Fig.2 allows calculating all possible solutions of the matrix equation S = r. H^t = 0, then one of these solutions is the correct code word. Fig.3 represented below shows all solutions of the LDPC (8, 4) code using Quartus.

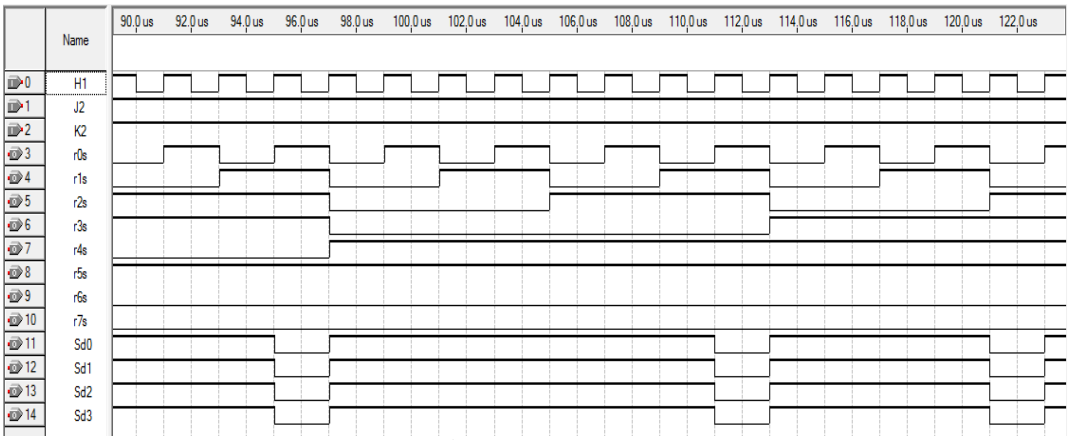


Fig. 3: Simulation of the Matrix Equation S=0 with Quartus.

For example in Figure 3 we have 3 solutions S = 0, so these code words 00101111, 00110111 and 00111100 are solutions of the matrix equation S = 0.

3.2.2. Proposed LDPC decoder

After we calculate all solutions of matrix equation, we are going to use theme to decode the received code word by determining the errors produced during the transmission channel. First we will use a multiplexer in order to add all solutions with the received code word R using a simple XOR, knowing that this code LDPC can correct t errors, so a simple circuit will be used to determinate the error that will be added to the received code word to calculate the correct code word. The LDPC decoder is shown in Fig.4.

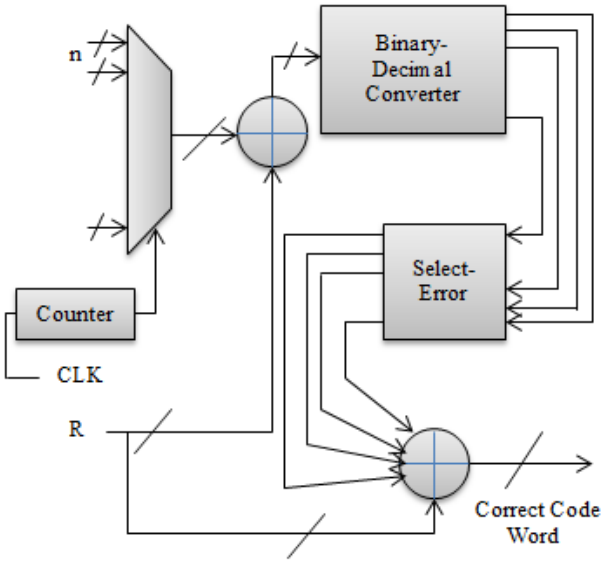


Fig. 4: Scheme of LDPC Decoder.

- Multiplexer

In this circuit we use a multiplexer that selects all possible solutions of the matrix equation S = r. Ht = 0.

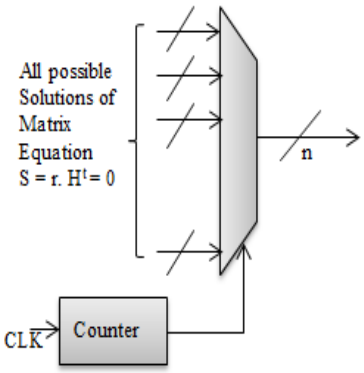


Fig. 5: Multiplexer of All Possible Solutions.

- Logic Gate XOR

The solutions of matrix equation will be added to the received code word in order to calculate all possible errors.

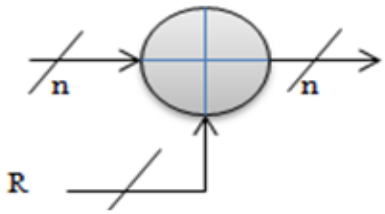


Fig. 6: Logic Gate Xor.

- Error selector

It's a simple circuit which developed in VHDL and determines the error of all errors previously calculated.

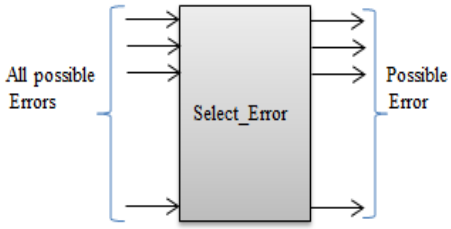


Fig. 7: Circuit of Error Selector.

3.2.3. Simulation using Quartus

The figure 8 shown below presents a VHDL simulation of ½ rate (2, 4) 8-bit LDPC code using Quartus. Where HORL represents the CLOCK and Se00, Se10… Se70, represent the outputs of the decoder (2, 4) 8-bit. For the case of LDPC (2, 4) 8-bit knowing that the received code word R is 11010101, the error produced during the transmission channel is 01000000, so according to the simulation represented in Fig.8 the correct code word is 10010101. Where Se70 =’1’, Se60 =’0’, Se50 =’0’, Se40 =’1’, Se30 =’0’, Se20 =’1’, Se10 =’0’, Se00 =’1’.

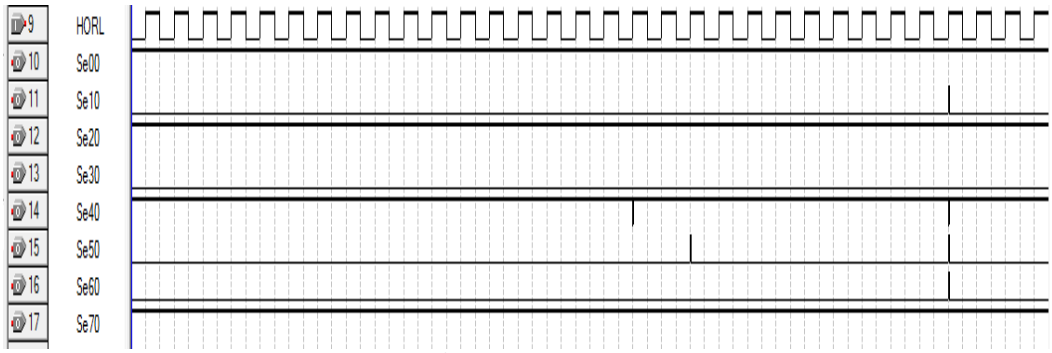


Fig. 8: VHDL Simulation of ½ Rates (2, 4) 8-Bit LDPC Code with Quartus.

4. FPGA implementation

Implementation of decoders for large LDPC codes has been problematic due to very large amount of resources required We seek to implement an LDPC decoder on FPGA based on the proposed algorithm to judge the savings in hardware resources. In this paper a parameterized hardware model of the decoder was developed using the Hardware Description Language (VHDL) and synthesized using Xilinx Synthesis Tool. The block diagram of the LDPC decoder as implemented is shown in Fig.9.

4.1. Design procedure

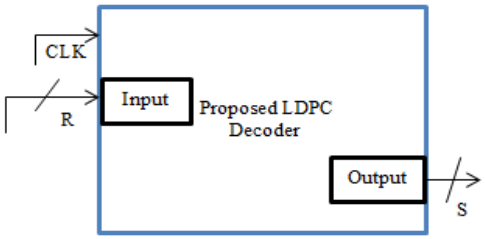


Fig. 9: Block Diagram of the Designed LDPC Decoder.

The decoder consists of a global ‘Clk’ and the decoding process is initiated by an‘Input’ which is latched. In parallel, the ‘Decoded Data’ can be obtained immediately after entering input.

4.2. Test procedure

The LDPC decoder was implemented on a Xilinx Spartan 3E-500 FG 320 FPGA (xc3s500e-5fg320). A comprehensive testing environment was developed to test the implemented decoder. The test setup is shown in Fig.10.



a) FPGA card Spartan 3E



b) FPGA Implementation of LDPC (2, 4) 8-bit

Fig. 10: An FPGA Implementation of ½ Rate (2, 4) 8-Bit LDPC Code.

4.3. Comparison of algorithms

According to the table 1 which shows the number of the used logic gates using the two algorithms, we can conclude that: on the one hand, the proposed algorithm presents a low complexity and a very good performance compared to the bit-flipping algorithm on the other hand, this algorithm does not use the iterations method which allows detecting easily and quickly the errors introduced by the transmission channel.

Table 1: FPGA Device Utilization Summary

Resources	Proposed algorithm	BFA
Device	Xilinx Spartan 3E (xc3s500e-5fg320)	Xilinx Spartan 3E (xc3s500e-5fg320)
Total Number Slice Registers	14	16
Number of occupied Slices	54	56
Number used as Flip Flops	8	8
Number of 4 input LUTs	107	107

5. Conclusion

A simplified algorithm for LDPC decoding has been presented in this paper. The advantage of this algorithm is that the correct code word must be found among the solutions of the matrix equation $S = r.\ H^t = 0$. The proposed algorithm also does not use the iterations method compared to BFA which allows us to detect easily and quickly the errors. The algorithm has been verified through FPGA implementation of a LDPC decoder. The results show that the decoder requires significantly reduced hardware resources compared to the bit-flipping algorithm (BFA), the hardware resource utilization results obtained from the fully parallel implementation of the decoder presented in this paper can be used to guide the design of partially parallel architectures for large codes to reduce the hardware resource requirement even further.

Acknowledgements

This work was supported by the Laboratory of Electrical Engineering and Energy System, Faculty of Science, University Ibn Tofail Kenitra, the National Society of Radio and Television (SNRT), Rabat and the National School of Applied Sciences, University Ibn Tofail Kenitra, Morocco.

References

[1] D. Han, H. Chen and L. Xi, "PMD mitigation through interleaving LDPC codes with polarization scramblers", *Optics & Laser Technology*, Vol.50, (2013), pp: 141-144, http://dx.doi.org/10.1016/j.optlastec.2013.02.010.

[2] B. Shin, H. Park, S. Hong, J. Seon and SH. Kim "Quasi-cyclic LDPC codes using overlapping matrices and their layered decoders", *International Journal of Electronics and Communications* VOL.68(5), (2014),pp: 379–383, http://dx.doi.org/10.1016/j.aeue.2013.10.004.

[3] AH. El-Maleh, MA. Landolsi and EA. Alghoneim, "Window-constrained interconnect-efficient progressive edge growth LDPC codes", i*nternational journal of electronics and communications* VOL. 67, (2013), pp: 588-594.

[4] W. Zhenbang, W. Zhenyon, G. Xuemai and G. Qing, "Cross-layer design of LT codes and LDPC codes for satellite multimedia broadcast / multicast services", *china journal of aeronautics* (2013),pp: 1–7.

[5] Y. Jianguo, X. Liang, J. Yuexing, T. Qingzhen and W. Yong, "A novel construction algorithm of the LDPC code for high-speed long-haul optical transmission systems", *Optik* VOL.124,(2013),pp:3181-3186, http://dx.doi.org/10.1016/j.ijleo.2012.09.047.

[6] NP. Bhavsar, B. Vala , "Design of Hard and Soft Decision Decoding Algorithms of LDPC", I*nternational Journal of Computer Applications* VOL. 90(4),(2014),pp:10–15, http://dx.doi.org/10.5120/15803-4653.

[7] H. Ge, *Investigation of LDPC code in DVB-S2*, Institutionen för systemteknik Department of Electrical Engineering, 2012, pp: 9-12.

[8] V.A. Chandrasetty, and S. M. Aziz, "FPGA Implementation of a LDPC Decoder using a Reduced Complexity Message Passing Algorithm," *Journal of Networks University of South Australia* VOL. 6, NO. 1, (2011) pp: 36–45.

[9] CC. Wong, *Low Density Parity-Check Codes*, National Chiao-Tung University, 2012, pp: 3-4.

[10] T. Tuan, *Tutorial on Low Density Parity-Check Codes*, university of Texas at Austin, pp: 1-9.