

Performance Study and Synthesis of the New LDPC Error Correcting Code Using the Bit-Flipping Algorithm

Lagrini Lakbir^{1,*}, El Habti Idrissi Anas², Moulay Brahim Sedra¹

¹Laboratory of Engineering Sciences and Modeling, Faculty of Sciences, University Ibn Tofail Kenitra, Morocco

²Laboratory of Electrical Engineering and Energy System, Faculty of Sciences, University Ibn Tofail Kenitra, Morocco

Received January 5, 2020; Revised March 9, 2020; Accepted March 19, 2020

Copyright©2020 by authors, all rights reserved. Authors agree that this article remains permanently open access under the terms of the Creative Commons Attribution License 4.0 International License

Abstract The discovery in 1993 of the turbo codes Berrou C., A. and P. Glavieux Thitimajshima at the International Conférence of Communication Florida has revolutionized communication. This innovation made it possible to rediscovery the error correcting codes invented by Robert Gallager in 1963 and rediscovered by Mackay in 1995, currently called Low Density Parity Check (LDPC). This code is one of the error correction codes introduced by the transmission channel that have the fastest and most effective level of protection and correction of the information transmitted during the communication. This code has been almost generalized in communication standards such as satellites. The objective of our article is to minimize the number of iterations and in some cases completely remove the iteration produced by the LDPC code bit flipping algorithm. The result of this algorithm, if we take for example the LDPC code (n, k) , the number of possible syndromes which we can find is 2^{k-1} and for each syndrome different from zero, we can do almost four iterations to correct the errors, which gives $4 \times (2^{k-1})$ iterations in totality. On the other hand, the proposed algorithm removes the iteration for k syndromes and reduces iterations for the rest of the cases from four to two iterations. This study of the new design gave more performance results and the results of the simulation below according to a hardware description language used in digital electronics using quartus software tools show the importance of this algorithm.

Keyword LDPC, VHDL, Bit-Flipping Algorithm

1. Introduction

In 1948 by information theory beginning with the founding article of Claude Shannon [1] in the Bell System

Technical Journal. Shannon's theory is based on the research bases that established in 1928 by H. Nyquist and R.V. Hartley. This article by Shannon showed the existence of a limit to the flow of information transmitted by the presence of disturbance noise, but Shannon did not explain the means of approaching so the theory remains inapplicable at this time. This information theory has now become indispensable in the design of any telecommunication system. The reason behind the motivation of the scientific community's research in the field correctors of errors, come mainly from the strong demand in the industrial telecommunication market and the strong use of electronic material like Mobile Phone, Tablet, GPS, Smart Grid, and different categories equipment, the overall health, etc. The different data exchange services require at all times reliability, performance and quality in the transmission of information. So the decoding coding tool and in particular the LDPC code (low density parity check) invented by Robber Gallager in 1962 [2] then rediscovered by Mackay in 1995 [3], currently used into some modern applications such as 10GBase-T Ethernet, Wi-Fi, WiMAX, Digital Video Broadcasting (DVB) [5]. The turbo-codes invented by C. Berrou, A. Glavieux, and P. Thitimajshima in 1993 [4]. To meet these needs, the two LDPC codes and Turbo code contribute in an essential way to the transmission level, the transmission channel and the final user. The objective of our article is to study the new decoding design proposed LDPC code and to make the simulation by the VHDL language, in order to decode the information quickly and without iteration compared to the basic bit flipping decoding algorithm used today in several communication standards. First, we entered the parity control equations in a table that gives all possible syndromes. Secondly, we can locate the wrong bit from these syndromes. Third, we will use the simulation of this proposed algorithm before and after error correction.

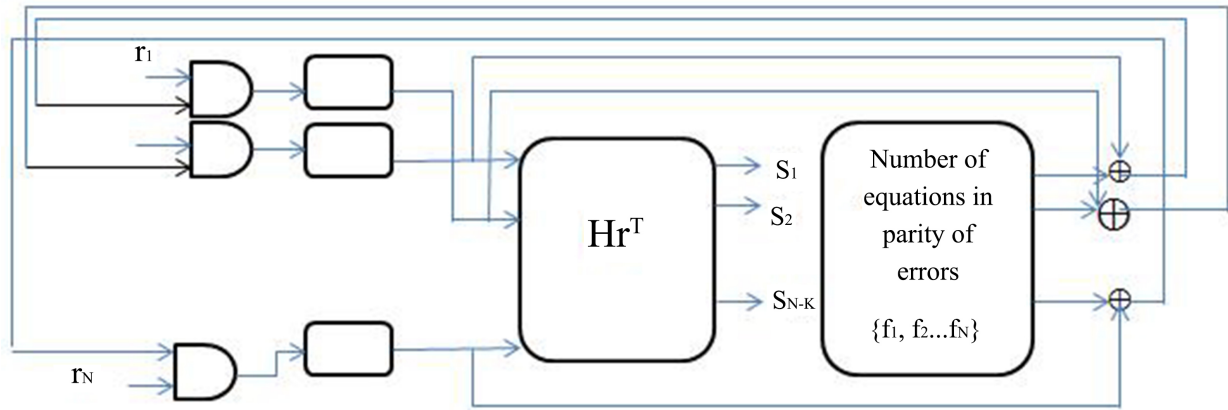


Figure 1. Diagramme of the bit flipping algorithm of LDPC code

2. Related Works

The Bit-Flipping algorithm currently used is based on the exchange of the number of parity failures when a bit of the received sequence is toggled. This is a control operation performed by the parity nodes on the received bits of the variable nodes, called the iterative decoding algorithm [2, 6, 7]. The decoder each time calculates all parity sums and then it changes each bit in the received sequence if it is part of the failed parity equations. From the modified sequence, the decoder recalculates the parity sums and the process is repeated until all parity sums are zero [8]. The number of iterations of the decoding is unstable and depends on the signal-to-noise ratio of the channel. This method requires a little more circuit complexity, but it gives better error performance than other methods. The bit flipping algorithm consists of four essential steps.

2.1. Four Various Steps

Step 1: calculation of the syndromes: $S = Hr^T$

If $S = 0$ the algorithm ends, so the received vector is a valid codeword, which can be decoded into a useful message, otherwise, go to the next step;

Step 2: determine for each of the N bits received the number of error parity equation: $\{f_1, f_2, \dots, f_N\}$;

Step 3: determine the set S of the received bits for which a maximum of error parity equations are obtained: $\max\{f_1, f_2, \dots, f_N\}$;

Step 4: to toggle the set of received bits in S and return to the first step (calculates of syndrome)

2.2. Explanatory Examples

By bit flipping decoding of an LDPC code:

The parity matrix H of an LDPC code of length $N = 12$ and $L = 6$, consisting of rows and columns such that the number of 1 in a row represents the weight $w_r = 6$ and the number of 1 in a column represent the weight $w_c = 3$, is illustrated below:

$$H = \begin{pmatrix} 110100001101 \\ 111110010000 \\ 101011000110 \\ 000111101010 \\ 011000110011 \\ 000001111101 \end{pmatrix}$$

The Tanner graph [9] of this LDPC code includes the variable nodes r_i with $1 \leq i \leq 12$ and the parity nodes S_j with $1 \leq j \leq 6$.

$$S_1 = r_1 \oplus r_2 \oplus r_4 \oplus r_9 \oplus r_{10} \oplus r_{12}$$

$$S_2 = r_1 \oplus r_2 \oplus r_3 \oplus r_4 \oplus r_5 \oplus r_8$$

$$S_3 = r_1 \oplus r_3 \oplus r_5 \oplus r_6 \oplus r_{10} \oplus r_{11}$$

$$S_4 = r_4 \oplus r_5 \oplus r_6 \oplus r_7 \oplus r_9 \oplus r_{11}$$

$$S_5 = r_2 \oplus r_3 \oplus r_7 \oplus r_8 \oplus r_{11} \oplus r_{12}$$

$$S_6 = r_6 \oplus r_7 \oplus r_8 \oplus r_9 \oplus r_{10} \oplus r_{12}$$

Suppose we receive the vector r in the form:

$$r = [100001101000]$$

According to the bit flipping decoding algorithm, he has to calculate the syndrome:

$$S = Hr^T = [010111]$$

This means that the received vector is in error. So, we will determine for each of the $N = 12$ bits the number of parity equations in error:

i.e. if we have $S = 010111 = S_1 S_2 S_3 S_4 S_5 S_6$ therefore the error parity equations are S_2, S_4, S_5 and S_6 .

This node variable r_1 exists only in the parity node S_2 who is in error, so $f_1 = 1$. On the other hand, r_2 exists in the two nodes S_2 and S_5 which have different from zero, so $f_2 = 2$. in the same way for the other cases, we find all the f_i compatible with the r_i such that i integer between 1 and 12:

$$[f_1 = 1, f_2 = 2, f_3 = 2, f_4 = 2, f_5 = 2, f_6 = 2, f_7 = 3, f_8 = 3, f_9 = 2, f_{10} = 1, f_{11} = 2, f_{12} = 2]$$

Note that $\max\{f_i: 1 \leq i \leq 12\} = \{f_7, f_8\}$. The set S of the received bits leading to a maximum of erroneous parity equations is:

$$S = \{r_7, r_8\}$$

it is more likely that the received bits r_7 and r_8 have been changed on the transmission channel:

$$r = [100001011000]$$

The syndrome S is calculated another time and the number of parity equations in error is sought for each of the N bits received.

$$[f_1 = 0, f_2 = 1, f_3 = 1, f_4 = 0, f_5 = 0, f_6 = 1, f_7 = 2, f_8 = 2, f_9 = 1, f_{10} = 1, f_{11} = 1, f_{12} = 2].$$

And the maximum parity equation in error $S = \{r_7, r_8, r_{12}\}$.

So, I'm going to toggle the bits r_7, r_8 and r_{12} the vector r becomes: $r = [100001101001]$

And one calculates the syndrome $S = [110100]$: this one being always different from zero, one carries out a third iteration:

$$[f_1 = 2, f_2 = 2, f_3 = 1, f_4 = 3, f_5 = 2, f_6 = 1, f_7 = 1, f_8 = 1, f_9 = 2, f_{10} = 1, f_{11} = 1, f_{12} = 1],$$

$$\text{So } S = \{r_4\}.$$

We only switch the fourth bit of the received vector:

$$r = [100101101001]$$

And the calculation of the syndrome gives

$$S = [000011].$$

A S as a result, we did not achieve our goal because the syndrome is different from zero.

In this case, a fourth iteration of the bit flipping algorithm is performed. The errors in the parity equations for each received bit are this time:

the calculation of the syndrome is null = [000000]. So the vector is now valid, which can be decoded into a message.

3. Proposed Algorithm

3.1. The Steps of This Algorithm

The conception of our article is based in particular on the transfer of the equations of parity to a table, which will give directly the nodes of variable which one has to alter to find a null syndrome without doing several iterations. This table is characterized by three blocks and each of them has a specific function.

STEP 1: (P1) only consists of parity check equations.

STEP 2: (P2) includes a significant number of possibility of the syndrome that we can correct with 0 iteration. The column [pp.s] which represents a clean syndrome facilitates the search on the nodes of variables

which must be modified to find the null syndrome without doing several iterations.

Step 3: (P3) in this part, we introduce the notion of syndrome proper of the second part in order to find the rest of the probable syndromes; it suffices to add the syndrome proper of the second part. by using the following expression: $S_i \oplus S_j$ equals 0 and $S_i \oplus S_j$ different from 0.

3.2. The Proposed Table of a Dimension Code (7,3)

3.2.1. LDPC Code of the Dimension (7, 3)

We are working here on messages composed of binary digits. The transmitted messages therefore consist of strings of 0 and 1.

Parity check matrix of the dimension (7, 3)

$$H = \begin{pmatrix} 1001011 \\ 0101110 \\ 0010111 \end{pmatrix}$$

For the first step, we calculate the syndrome S of the received code word r such as $S = Hr^T$.

- If $S = Hr^T = 0$, i.e. syndrome is null then the received code word is correct, therefore terminate the algorithm, decoding it with the corresponding message.
- If $S = Hr^T \neq 0$, the syndrome is different from zero, so the code word received is incorrect.

Then, our proposed algorithm can be determined the variable node r_i that we have to toggle to find a null syndrome without iteration. we consider the message received after the transmission channel is written in the form is $r = r_1 r_2 r_3 r_4 r_5 r_6 r_7$ Where each r_i is either 0 or 1.

With

$$S = Hr^T = s_1 s_2 s_3$$

$$\begin{pmatrix} 1001011 \\ 0101110 \\ 0010111 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \end{pmatrix} = s_1 s_2 s_3$$

The matrix product of Hr^T gives a parity check equation:

$$S_1 = r_1 \oplus r_4 \oplus r_6 \oplus r_7$$

$$S_2 = r_2 \oplus r_4 \oplus r_5 \oplus r_6$$

$$S_3 = r_3 \oplus r_5 \oplus r_6 \oplus r_7$$

In the P_1 part, the components of the syndromes S according to the nodes of variables.

In the P_2 part, there are $2^3 - 1$ possible syndromes.

The symbol x in the P_2 part represents the r_i that must be to toggle to find a null syndrome.

Table 1. Show all the probable syndromes of the LDPC code (7,3)

$s_1s_2s_3$	r_1	r_2	r_3	r_4	r_5	r_6	r_7	[pp.s]	
s_1	x			x		X	x		P ₁
s_2		X		x	x	x			
s_3			X		x	x	x		
100	x							s_1	P ₂
010		X						s_2	
001			X					s_3	
110				x				s_1s_2	
011					x			s_2s_3	
111						x		$s_1s_2s_3$	
101							x	s_1s_3	

[pp.s]: Proper syndrome.

3.2.2. Reading the Proposed Table

Example 1: Explanation of the proper syndrome: In column six (i.e. column r_5), the case between line s_1 and column r_5 (part 1) is empty, but between line s_2 and column r_5 (part 1), there is an X symbol and between line s_3 and column r_5 (part 1) there is also an X symbol, which gives the following expression s_2s_3 (proper syndrome in part 2), which is represented in binary by 011(column 1 part 2).

Example 2: Supposing after the calculation of Hr^T , we find the syndrome 011 which, according to the Proposed Algorithm, is equivalent to s_2s_3 . In this case we only need to toggle r_5 to get the null syndrome.

If $r_5 = 1$ will be changed to 0

If $r_5 = 0$ will be changed to 1.

More explanation regarding reading this table:

- **The first** part (P1) represents parity check equation s_1, s_2 and s_3 the X symbol represents the r_i forming the s_i
- **The second** part (P2) represents a large number of possible syndromes, from this block we can determine without calculation the variable node r_i (= 1 or 0) that must be returned to have the syndrome that corresponds to zero.
- **The third** part (P3) represents the rest of the possible syndromes, and is calculated by exploiting the new notion of own syndromes in Part 2 (P2).

3.3. The Proposed Table of a Dimension Code (9,4)

3.3.1. LDPC Code of dimension (9, 4) Parity Check Matrix

$$H = \begin{pmatrix} 011011000 \\ 101100100 \\ 111010010 \\ 000110001 \end{pmatrix}$$

We calculate the syndrome S of the received code word r such as $S = Hr^T$

- If $S = Hr^T = 0$, i.e. the syndrome of null then the received code word is correct, so the algorithm has finished, the received message is correct and we can determine the useful message by decoding it to the corresponding message.
- If $S = Hr^T \neq 0$, the non-zero syndrome, therefore the code word received is incorrect.

The proposed algorithm able to determine the variable nodes r_i which must be modified to find a null syndrome.

Suppose the received code word after the transmission channel is $r = r_1r_2r_3r_4r_5r_6r_7r_8r_9$

Where each r_i is either 0 or 1 and $S = Hr^T = s_1s_2s_3s_4$

$$(S = s_1s_2s_3s_4).$$

$$\begin{pmatrix} 011011000 \\ 101100100 \\ 111010010 \\ 000110001 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \end{pmatrix} = s_1s_2s_3s_4$$

Each row of H gives a parity check equation:

$$S_1 = r_2 \oplus r_3 \oplus r_5 \oplus r_6$$

$$S_2 = r_1 \oplus r_3 \oplus r_4 \oplus r_7$$

$$S_3 = r_1 \oplus r_2 \oplus r_3 \oplus r_5 \oplus r_8$$

$$S_4 = r_4 \oplus r_5 \oplus r_9$$

Table 2. Represents all possible syndromes for LDPC (9, 4) codes:

$s_1s_2s_3s_4$	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	$[pp.s]$	$[pi]$
s_1		x	x		x	x					p_1
s_2	x		x	x			x				
s_3	x	x	x		x			x			
s_4				x	x				x		
0110	x									s_2s_3	p_2
1010		x								s_1s_3	
1110			x							$s_1s_2s_3$	
0101				x						s_2s_4	
1011					x					$s_1s_3s_4$	
1000						x				s_1	
0100							x			s_2	
0010								x		s_3	
0001									x	s_4	
1111		x		X						$s_1s_2s_3s_4$	
0111	x								x	$s_2s_3s_4$	
1101				X		x				$s_1s_2s_4$	
1001						x			x	s_1s_4	
0011								x	x	s_3s_4	
1100	x	x								s_1s_2	

3.3.2. Illustrative Example

If the matrix product of Hr^T gave for example, the syndrome 0111, which is equivalent to the specific syndrome $s_2s_3s_4$ of the proposed algorithm, we will look in part (2) of the table on the compatible syndromes to add them up, to find the syndromes of parity (3), then alter the suitable variable nodes, to find the null syndrome.

The proposed algorithm uses the addition of proper syndrome as follows:

$s_2s_3 \oplus s_4 = s_2s_3s_4$ we are forced to toggle r_1 (case of s_2s_3) and r_9 (case of s_4) or $s_2 \oplus s_3 \oplus s_4 = s_2s_3s_4$ we are forced to toggle r_7 (case of s_2), r_8 (case of s_3) and r_9 (case of s_4) or $s_2s_4 \oplus s_3 = s_2s_3s_4$, we are forced to toggle r_4 (case of s_2s_4) and r_8 (case of s_3).

4. The Simulation of the Proposed Algorithm by the Quartus Software Tool

4.1. Algorithm Proposed of LDPC Code (7,3)

4.1.1. Simulation of LDPC (7, 3) Code after Correction

On reception, the decoder receives the different code words after the transmission channel. The hardware description language (VHDL) VHSIC which will calculate various possible cases of syndromes based on the equation $S = Hr^T$. Such as $S = s_1s_2s_3$ and

$$r = r_1 r_2 r_3 r_4 r_5 r_6 r_7$$

4.1.2. The Simulation Result after the Correction of the Errors of LDPC (7, 3)

Figure 2 shows that the syndrome has become zero, regardless of the received code word values, which implies the performance and reliability of this error correction algorithm.

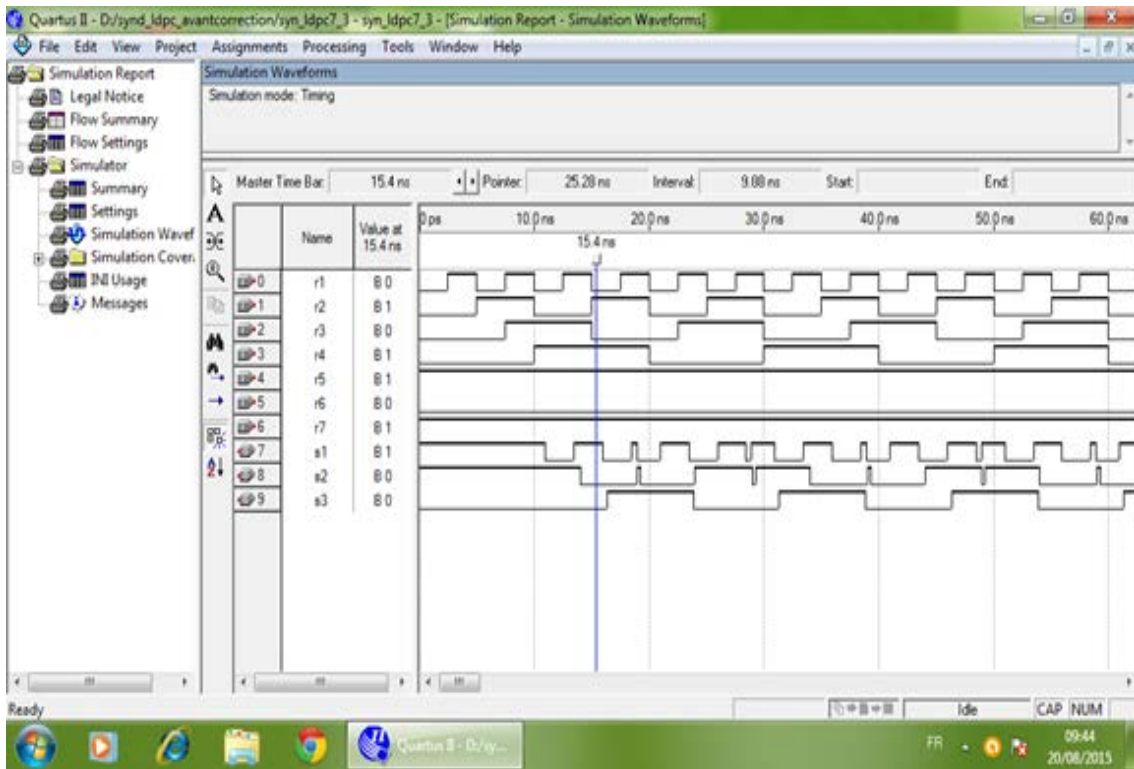


Figure 2. Different possible values of the syndrome for an LDPC code (7, 3)

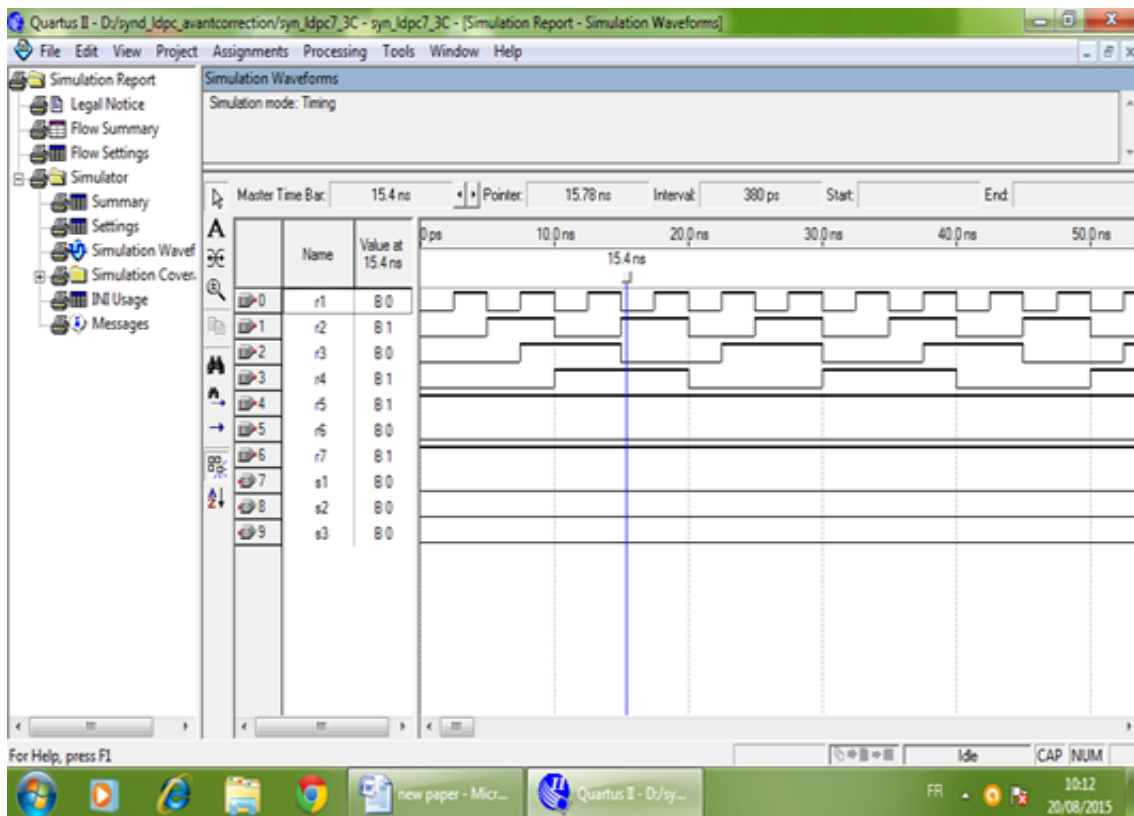


Figure 3. Different values of the syndrome become null.

4.2. Algorithm Proposed of LDPC Code (9,4)

4.2.1. Simulation of LDPC (9, 4) Code after Correction

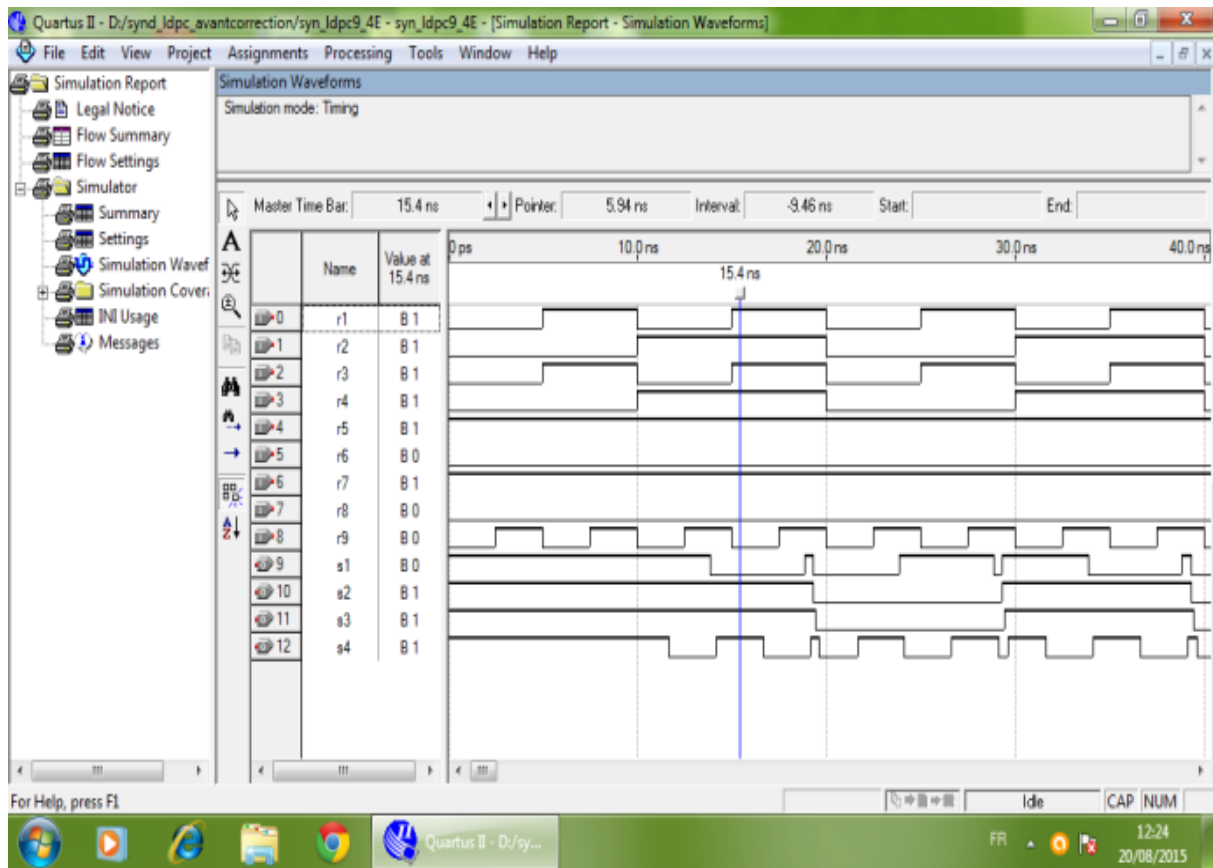


Figure 4. Different possible values of the syndrome for an LDPC code (9, 4).

4.2.2. The Simulation Result after the Correction of the Errors of LDPC (9, 4)

The figure below shows for the different values of r_i with $1 \leq i \leq 9$, we always find a null syndrome (i.e. $S_1S_2S_3S_4 = 0000$).

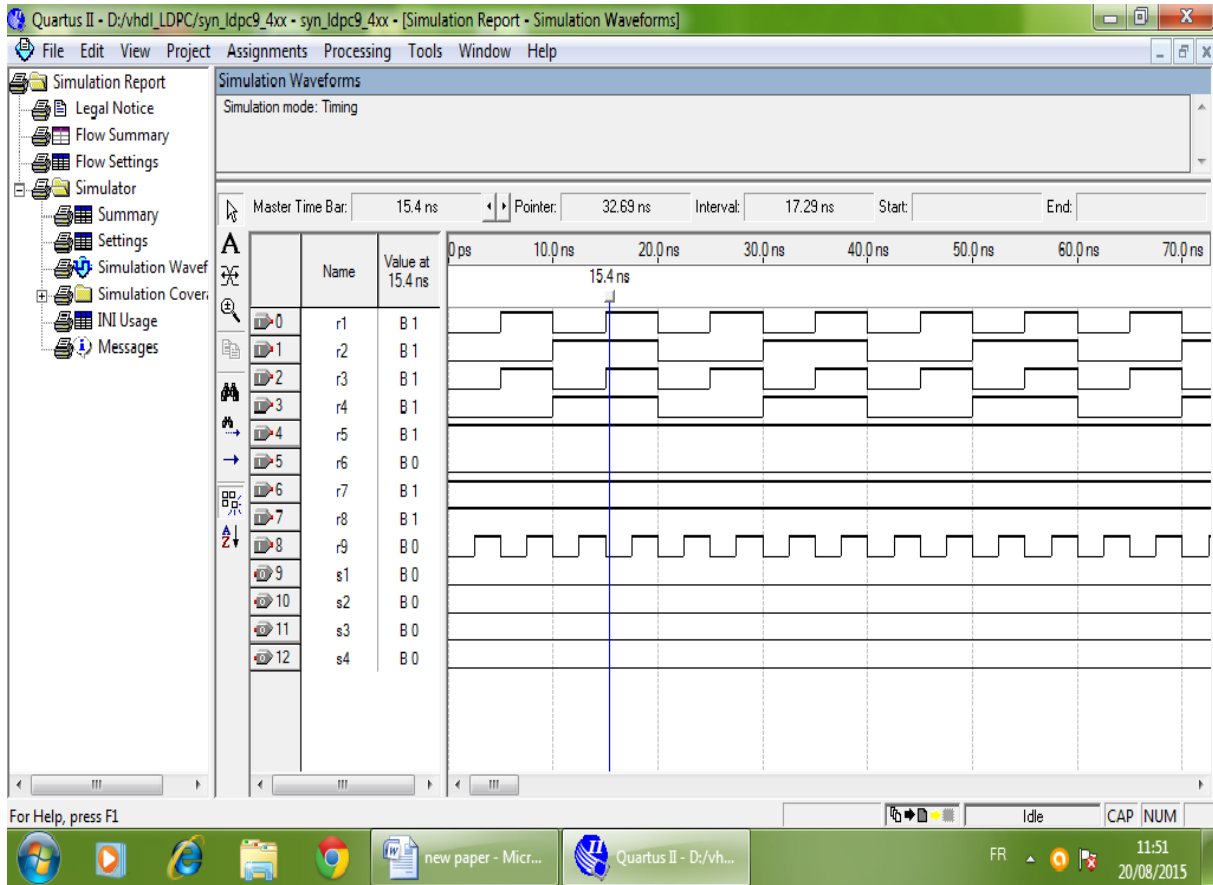


Figure 5. Different values of the syndrome become null after correction.

5. Comparison of Algorithms

The proposed algorithm of bit flipping decoding has made it possible to minimize the number of iterations compared to the basic algorithm. if we take for example the LDPC code (12,6), the number of possible syndromes which we can find is $2^6 - 1$ and for each syndrome different from zero, we can do almost four iterations to correct the errors, which gives $4 \times (2^6 - 1)$ iterations in totality. On the other hand, the proposed algorithm, it removes the iteration for twelve syndromes and reduces iterations for the rest of the cases from 4 to 2 iterations.

Table 3. Show the difference in the number of iterations between the proposed algorithm and basic algorithm

	Proposed Algorithm	Basic Algorithm
LDPC (7,3)	we can correct 2^3-1 error without iteration	the number of iterations to correct indeterminate errors
LDPC (9,4)	Part 2: correction without iteration. Part 3 : it suffices to add two proper syndrome of the part (2).	2^4-1 possible syndromes. 4 possible iteration if a received codeword equals 101010100
LDPC (10,5)	Part 2: correction without iteration. Part 3 : it suffices to add two proper syndrome of the part (2)	2^5-1 possible syndrome. the number of iterations to correct indeterminate errors
LDPC (12,6)	Part 2: correction without iteration. Part 3 : it suffices to add two proper syndrome of the part (2)	2^6-1 possible syndromes. 4 possible iterations of a single case.
LDPC (n, k)	Part 2: correction of k code word received without iteration. Part 3 : it suffices to add two proper syndrome of the part (2)	the total number of iteration of indefinite code words

6. Conclusions and Perspectives

6.1. Conclusions

- The bit flipping decoding algorithm proposed in this article is based on the design of minimizing iteration according to the new concept of proper syndrome. The distribution of the equation of parity in a table facilitates the access to the nodes of variables which it is necessary to toggle to find a null syndrome.
- The bit flipping decoding algorithm of a proposed LDPC code also allows us to completely eliminate and eliminate basic decoding iteration. Particularly with respect to a number n of the syndrome, such as n the length of the matrix of parity H .
- The block P_2 in our proposed table directly gives the variable nodes r_i that we must return to find a null syndrome. The block P_3 , just add two proper syndromes to correct the errors.

6.2. Perspectives

The proposed algorithm can be improved in future work by implementation on the FPGA, Digital Video Broadcasting-Second Generation (DVB-S2).

Acknowledgements

I thank Professor Doctor Moulay Brahim SEDRA, who was appointed to the post of Dean of the Faculty of Science and Technology of Errachidia by the President of the Moroccan government, for his support throughout the duration of my research.

I also thank Doctor Habti Idrissi ANASS, the expert in the field of coding decoding and especially the RS, BCH and LDPC codes for his support.

I also thank Professor Damir Mohammed the vice-president of Ibn Tofail University of Kenitra Morocco for his support.

Thank you also to all my colleagues and friends in the engineering science and modeling laboratory. I express my deepest sympathy and wish them well.

REFERENCES

- [1] C. Shannon, A Mathematical Theory of Communication, Bell Syst. Tech. Journal, vol.27, pp. 623-656, 1948.
- [2] R.G. Gallager, Low-Density Parity-Check Codes, IRE Transaction on Information Theory, vol. IT-8, pp. 21-28, January 1962
- [3] D.J. MacKay and R.M. Neal, Good Codes Based on Very Sparse Matrices, in Cryptography and Coding 5th IMA Conference, vol. 1025, pp. 100-111, 1995.
- [4] A. H. El-Maleh, MA. Landolsi and EA. Alghoneim, "Window-constrained interconnect-efficient progressive edge growth LDPC codes", international journal of electronics and communications VOL. 67, 2013.
- [5] [5] B. Ahn, S. Yoon, and Jun Heo, "Low complexity syndrome-based decoding algorithm applied to block turbo codes", Access IEEE, Vol. 6, PP. 26693-26706, 2018
- [6] T. T. Nguyen-Ly, V. Savin, K. Le, D. Declercq, F. Ghaffari, and O. Boncalo, "Analysis and design of cost-effective, high-throughput LDPC decoders," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. PP, no. 99, pp. 1–14, Dec 2017.
- [7] L. Shiva Nagender Rao, K. Venkata Sathyajith, Y. Nagaraju Yadav. Encoding and Decoding of LDPC Codes using Bit Flipping Algorithm in FPGA, International Journal of Innovative Research in Computer and Communication Engineering, Vol. 5, Issue 8, August 2017.
- [8] B. Unal, A. Akoglu, F. Ghaffari, B. Vasić, "Hardware implementation and performance analysis of resource efficient probabilistic hard decision LDPC decoders", IEEE Trans. Circuits Syst. I Reg. Papers, vol. 65, no. 9, pp. 3074-3084, Sep. 2018.
- [9] Irina E. Bocharova; Boris D. Kudryashov; Vitaly Skachek, 2019 IEEE International Symposium on Information Theory (ISIT), Added to IEEE Xplore 26 September 2019.