ISSN: 9772477798001

Journal of Measurements, Electronics, Communications, and Systems (2015) CS0115-01
www.jmecs.org/vol1a

Improvement of LDPC Forward Error Correction Capability and Computation Time Based on FPGA Implementation

A. Tarigan, R. Purnamasari, and E. M. Saputra*
School of Electrical Engineering,Telkom University, Bandung, 40287, Indonesia

Abstract
LDPC is one of channel coding technique which can achieve nearest limit of Shannon's capacity. The focus of this paper is to improve the limitation of LDPC bit flipping algorithm for error correcting process in DVB-S2. This work proposed message passing as the error correcting algorithm and used FPGA Cyclon II for the implementation. This paper worked with two different types of matrices. They were matrix (8, 16) and matrix (24, 48). Matrix (24,48) had $w_c = 4$ and $w_r = 8$, and matrix (8, 16) had $w_c = 2$ and $w_r = 4$. The results of this research would present the capability of message passing algorithm and the use FPGA Cyclon II's resources for this algorithm. Moreover, this research would prove message passing algorithm can provide more than one bit error correction.

Keywords: Error Correction; FPGA; LDPC; Matrix; Message passing;

1. Introduction

In recent years, "Low Density Parity Check" or LDPC becomes more popular, as a channel coding technique, due to high demands of good quality of service. LDPC was originally found by R.G.Gallager in 1962. This technique has capability which is known as the closest technique that can reach maximum limit of Shannon's capacity. Moreover, LDPC has low complexity for hardware implementation and suitable for recent information technology's market which need large bandwidth, high reliability, and good capability for facing bad channel noise [1]. Nowdays, there are many standards and researches which use LDPC as their channel coding technique.

Recently, LDPC technique has been used for several standards such as WiMax, WPAN, DVB-T2, and DVB-S2. As example at WPAN sector, the LDPC decoder has been used to support multiple code rates for IEEE 802.15.3c [2]. In the multi-Gbit/s research, it used LDPC as its decoder [3]. It is also used for MIMO decoder [4]. Even digital video broadcasting technology uses LDPC for its inner channel encoding [5,6,7]. The recent research implemented the LDPC technique as a DVB-S2 decoder on FPGA board [5]. The research used 6 x 12 LDPC matrix, code rate ½, and bit flipping algorithm in the decoder block [5]. As the result, it only needed minimum resources of FPGA board [5].

The bit flipping algorithm has two limitiations[5]. The first problem is about the big amount of time in the computation process and the possibility of unlimited loop. The second problem is that the bit flipping algortihm can only correct one bit error at the codeword. Based on that limitations, this research tried to reduce the computation time in the bit flipping algorithm and improve the possibility to correct more than one bit error at the codeword.

The message passing algorithm has the capability to work in the parallel error correction process. With big sparses in the LDPC matrix, it would make the algorithm to have more capabilities in error correction.

The bigger sparse of LDPC matrix will give more capability in the error correction process [1,8,9] On the other hand, FPGA has resources limitation. The first objectivity of this research tried to design and to balance the use of LDPC matrix with the FPGA resources. Short computation process would be the next challange. The balance of the good design and the short computation time were also needed in this. The third objectivity was to change the design into VHDL codes and to avoid unconnected schemes. The last objectivity was to test the capability of the design in the use of FPGA's resources and the error correction capabilities.

In this research, the matrices which used were matrix (8,16) and matrix (24,48). Both of matrices has the same code rate. The code rate is ½. But, the differences of both matrices were in weight of column and weigth of row.

The results of this research showed that the using of message passing algorithm could do more than one bit error correction and the computation time only took 13 clocks or 260 ns. The impelementation schemes would be done on FPGA Cyclon II board. For synthesizing, Quartus II 12.1 web edition would be used as the interface.

$$H = \begin{bmatrix} 1010000000000101 \\ 1100000000001010 \\ 0010100001010000 \\ 0001010100100000 \\ 0100001010010000 \\ 0001101000001000 \\ 0000010001000110 \\ 0000000110100001 \end{bmatrix}$$

Fig 1. Parity Check Matrix (8, 16) Regular LDPC Representation.

ISSN: 9772477798001

This paper consisted of five sections. The first sections is about the background. Section 2 shows the overview of LDPC, message passing algorithm, and the purpose of using the matrices and the algorithm. Section 3 discusses about the system models. Section 4 presents the results of the implementation. Section 5 presents the conlusion.

2. LDPC and Message Passing Algorithm

This section will discuss about the LDPC parity check matrix and the message passing algorithm.

2.1 LDPC Parity Check Matrix

Low Density Parity Check technique is still in part of linear block coding. Although it's the part of linear block, LDPC matrix is different from linear block code matrix or hamming code matrix. At the linear block code, the matrix will depend by the weight or the total numbers of bits '1'. It means that linear block code matrix will contain more numbers of bits '1' than the amount of the sparses. On the other hand, LDPC matrix has more sparses [1,8]. Because of the dependence of the sparse, LDPC is also called low density matrix.

LDPC matrix has two types of weights. These are the weight of the coloum (w_c) and the weight of the row (w_r) [1,8]. The weight of coloum represents the numbers of '1' in each of the coloum [1,3, 4]. The weight of row represents the numbers of '1' in each of the row [1,8]

LDPC matrix can be categorized into two different types of matrix, regular and irregular [3,4]. The regular matrix is the matrix which has a constant weight in its rows and its coloumns [8,9]. The irregular matrix is the opposite of the regular matrix. The irregular matrix contains varied weight in its rows and its coloumns [8,9]. Due to minimum resources of FPGA Cylone II, this research used regular LDPC matrix which has lower complexity and needs less resources of FPGA board for implementation compared to the irregular; However, as a consequence, the use of the regular matrix could not achieve the maximum capability of the LDPC decoder.

LDPC matrices which used in this research were regular LDPC matrix (8,16) and regular LDPC matrix (24,48). Both of the matrices had code rate ½ . The bits '1' were placed randomly in the matrix.

The illustrations figure 1 and 2 show matrix (8,16) and matrix (24,48) in the matrix representations and tanner graph representations. Matrix (8, 16) has w_c = 2 and w_r = 4. Because of the weight, in tanner graph representation, every variable nodes of matrix (8,16) connects with two parity nodes. And, each of parity nodes connects with four variable nodes. Differently from matrix (8,16), matrix (24, 48) has w_C = 4 and w_r = 8. The connections between variable nodes and parity nodes are different too. Each variable nodes connects with four parity nodes. And, each of parity nodes connects with eight variable nodes.

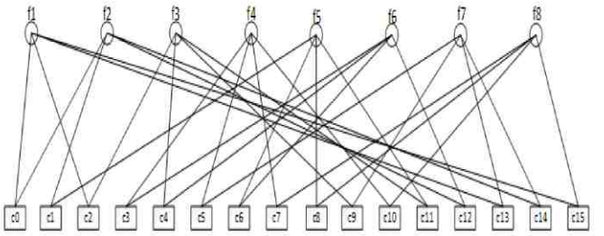

Fig 2.Tanner Graph Representation Matrix (8, 16) Regular LDPC Representation.

Matrix (8,16) has the minimum requirement to fullfill the requirement of LDPC matrix, but it is still not the best matrix to be used in implementation. This kind of matrix would be good to be implemented in the low resource hardware. The Gallager's suggestion for good LDPC matrix was the matrix which has $w_c \leq 3$ and $w_r < w_c$ [1]. Because of the limitation of FPGA's resources, bigger matrix would not able to be implemented. For this reason, Matrix (24,48) has achieved minimum requirement of Gallager sugesstion to be called as the good LDPC matrix.

The research would compare both capability and resources utilization of each matrix to one another. The comparison results would provide some suggestions for system developer for developing his own suitable LDPC decoder system.

2.2 LDPC Message Passing Algorithm

Both of message passing algorithm and bit flipping algorithm has same basic process. Both of them work in the exchange of information between check nodes and variable nodes. But, the bit flipping algorithm has some vulnerability. It takes big amount of computation time. In it process, the bit flipping would check all of the bits in the codeword one by one (from the first bit until the last bit). It would take a lot of time. Moreover, the error correction process might produce infinite loops due to undecide true codeword. In addition, the last research [5] showed that the algorithm could only correct one bit error. Message passing algorithm has solutions for all of this limitations.

The simultaneous work of variable nodes and check nodes [8,9,10], the variable nodes and the check nodes decide the true codeword at the same time, makes the computation time shorter than the bit flipping. These are few steps of error correction process of message passing algorithm.

At the first, all of variable nodes receive information or the codeword from channel [9]. The first state is also known as the initialization process. The information which come from the channel is the corrupted version of the original codeword. At the second, variable nodes send all of the information, which it has received, to its adjacent check nodes [9].

ISSN: 9772477798001

For example, at matrix (8,16), variable nodes c0 will send the information to parity nodes f1 and f2. This state is also known as variable to parity check process. At the third, check nodes will do calculations to decide whether the information is correct one or not. At this step, each check node will do modulo-2 operation to all of its adjacent nodes value [9]. If the result of modulo-2 operation is '1', parity nodes will inverse its values [9]. At the fourth, check nodes send the calculation results back to variable nodes [9]. Example, the parity check f1 sends all of the results to c0, c2, c13, and c15. At the fifth state, variable nodes will decide the correct codeword. At decision process, the variable nodes will decide the true value by comparing the values between channel and the related check nodes[9]. The major value will be decided as the true value.

At the last, the message passing algorithm does the rechecking process. This process is same as the second step. If the requirement of true codeword has been reached, the process would be stopped and the true codeword would be presented. But, if the result has not yet been fullfilled, the message passing will repeat all over the process again. The full explanations about these process had been explained by B.M. Leiner [9].

All of the error detectors have decision regions which help the detector to decide correct version of the codeword from the output of channel [8]. At linear block code technique, big size matrix would make the error correction process better. It means that each of linear block code matrix has its limitation in correcting errors. It is caused by the capability of its decision regions [8]. It would be same in the LDPC. If the LDPC has bigger size and more amounts of sparse bits, then the better of error correction process would become. It means that the bigger size matrix has better decision regions.

Both of variable nodes and check nodes have function to decide the true codeword from the error one, but the most important part is happen in the check nodes calculations process. The reason is because only in this process all of the bits in the codeword would have interferences to one another. The big sparses matrix would minimize the interferences between the bits [1].

Since the bits '1' placed randomly, the decision region was hardly to be determined. A series of tests would be given to the LDPC decoder's system, matrix (8,16) system and matrix (24,48) system, for testing its decision regions and the capabilities in error correction. Each of the systems would be tested from one bit error until N bits error, or it means that each of the systems would be tested until the systems could not do the error correction anymore.

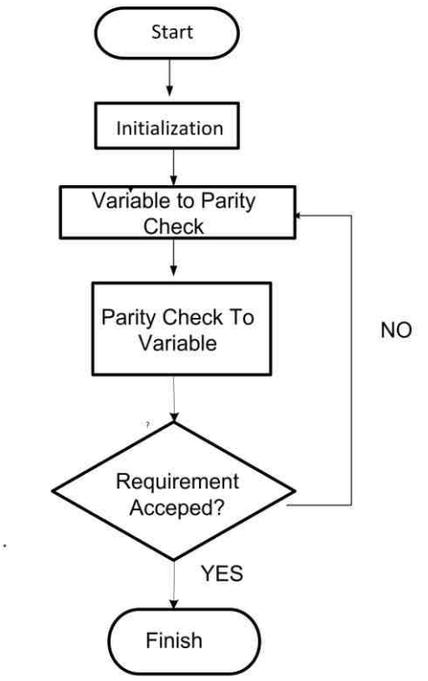

Fig 3. Message Passing Algorithm Flowchart

3. System Model and Implementation

In general, the system had three main blocks. There were encoding block, channel block, and decoding block. This picture below presents general system model which is used in this work.

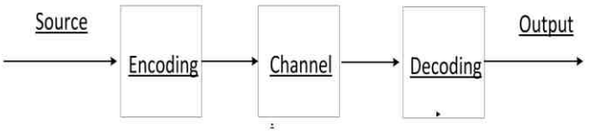

Fig 4. General System Model

3.1 Clock and Reset

This research used 50 MHz of internal FPGA's clock frequency [11]. The clock distributed equally to all of the blocks. To controll the system to work properly, enable buttons were needed.

When the enable was activated, the block which related to it would started to work. The enables would be activated by clock's counter. The table below presents the timing of enables activation.

The reset button was generated by FPGA's switch button 1. Same as clock, the reset would distributed to all of the blocks. If reset button was on, then all of the counters would be backed to 0.

The table 1 showed that clock are built with counter and there is the comparison of speed of the clock between using 8,16 and using 24,48 for parity check.

ISSN: 9772477798001

Tabel 1. Enable activation

No	Block	Counter or timing to activate enable	
		Matrix (8,16)	Matrix (24,48)
1	Encoding	1	1
2	Channel	18	39
3	Serial to parallel	19	40
4	Error Correction	24	41
5	Parallel to Serial	33	56

3.2 LDPC Encoding

Generally, encoding block has function to add some redundant bits into source bit or information. Source bits with redundant bits is called codeword. The redundant bits would increase the capability of the information to face error probabilities which is caused by channel. The true codeword has to full fill Eq. (1). C is the codeword, and H is the parity check matrix.

$$c.H^T = 0 \quad (1)\ [3]$$

Since both of matrices using coderate ½, the codeword would be twice bigger than the source. The matrix (8,16) needed 8 bits as input, and the codeword would be 16 bits long. The matrix (24,48) needed 24 bits source, and the encoder output would be 48 bits. The process to get the codeword is bacially same as with decoding process [9].

The input of encoder block matrix (8,16) "10001001" would be the information input for encoding block matrix (8,16). The codeword would be "1000100101101100" as the output. In matrix (24,48), the 24 bits long would be needed as the input, "100010011010111100010011". The encoding results was "100010011010111100010011000001110000100110011110".

3.3 Channel

This research used binnary symmetric channel type. It means that probability of receiving bit '0' by sending bit '1' is same as probability of receiving bit '1' by sending bit '0' [3], or probability of receiving '0' by sending '0' is same as receiving '1' by sending '1' [8].

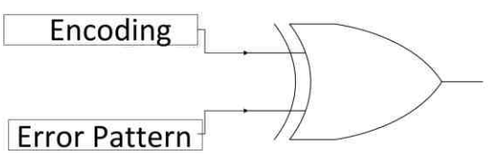

Fig. 5 Illustration of Channel

The channel block worked by using modulo-2 operation. The operation would be done between the codeword and channel error pattern. Error pattern was a group of bits would give exact positon of error in the codeword. Example, the error pattern for making eror at first bit in matrix (8,16) was "100000000000000". Wrong codeword would be the output of channel block.

3.4 LDPC Decoding

There were two steps of decoding process. There were the error correction process and codeword into source bits transformation. There were two tasks in the error correction. At first, the system had to decide whether the channel output was correct or not. The second steps, if it was false, than the decoder had to correct the codeword until it was true. This step would use message passing algorithm.

The codeword into source transformation was the process to transform the codeword into its original information source form. These were serial paralel block, decoder, paralel to serial block. At this stage, the error correction process were designed to finish in 13 clocks.

A. Serial Paralel Block

This block had a function to change serial form of the channel output into parallel form. Parallel computation prefer to be used in FPGA implementation because the process will be faster than the serial form.

Since the output of telecommunication channel is modeled in serial form, the convertion into parallel form is needed.

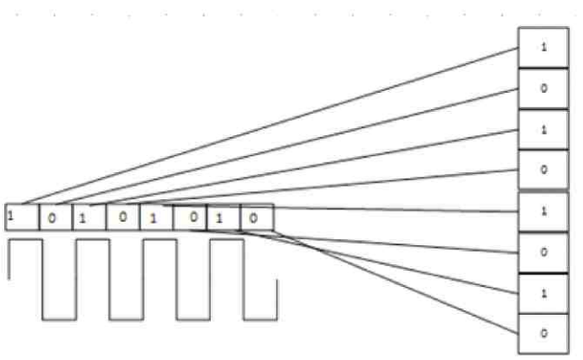

Fig 6. Serial to Paralel Block

The figure 6 shows that the information in serial form which contain "10101010" bits is converted into paralel form.

B. Decoder Block or Error Correction Block

Decoder block had function to do error correction. This block consisted of several section or sub-systems.

The first sub-system was input receiver. The input sub-system received the parallel form of informations, and it would be continued into the variable nodes sub-system.

ISSN: 9772477798001

The second sub-system was variable nodes. In this sub-systems, each of connections between variable nodes and check nodes had to be given a value. As example at matrix (8,16), the connection between c0 and f1 would be named as d0, and the connection between c0 and f2 would be named as e0. At the initialization state, the input sub-systems would send the channel output to the variable nodes. Moreover, in the initizialization, both of d0 and e0 had same value as c0.

The third sub-systems was check nodes. The check nodes sub-systems function was to decide whether all of informations which have come were correct or not. This system worked same as check nodes process at message passing algorithm which has been described before.

The fourth sub-system was variable nodes calculation. This sub-system would decide the correct codeword. This sub-system used voting for the decision. The voting worked by choosing the majority information, from channel and parity nodes, as the correct version of the information.

The last sub-system was codeword validation. This sub-system would re-check the new codeword and decide whether it fulfilled the requirement or not.

C. Paralel Serial Block

The outputs of error correction block were in parallel form. On the other hand, the original information was in serial form. The parallel serial block converted it into serial form.

Figure 7 illustrates the work of paralel serial block. Each block in the illustration contains one type information which is '1' or '0'. The '1' will be converted into the 'high' signal, and the '0' will be converted into the 'low' signal.

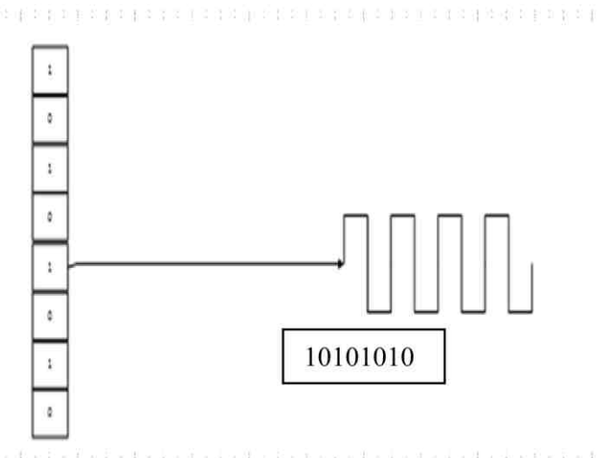

Fig 7. Paralel to Serial Block

3.5 LDPC Implementation

This research used FPGA Cyclone II EP2C20 . The Quartus II 12.1 web edition was used as synthesize tool.

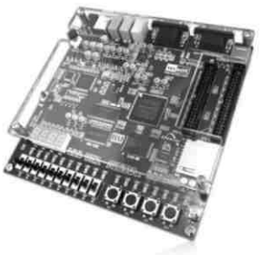

Fig 8. FPGA Cyclone II [11]

The table below represents the features of FPGA Cyclone II EP2C20 specifications. The LEs referes to total numbers of logic elements in FPGA Cyclone II EP2C20 [11]. The M4K block refers to memory block of the FPGA [11]. The total RAM Bits refers to total momories of the FPGA[11]. The Embedded Multiplier refers to the maximum number of 9 bits multiplier [5]. The PLLs refers to phase loop lock numbers [11]. The Max I/O pins refers to maximum pins which the FPGA has [11].

There were two important things that needed to care about in implementation. Firstly, each block of the system had to be assured that it worked properly as its functions. Secondly, each blocks had to be assured that there was no overlapping in the system.

4. Result and Analysis

From the previous section, we would know that the total amount of clocks which needed for matrix (8,16) were 22 clocks and matrix (24,48) were 38 clocks. There were 1 clock for serial paralel block, 13 clocks for error correction , 8 clocks for paralel serial block for matrix (8,16), and , 16 clocks for paralel serial block for matrix (24,48). If the FPGA's clock was 50 MHz, than the periode of one clock is equal to If 22 clocks were needed for implementation, then it would take around 440 ns to finish one cycle of matrix (8,16) decoding process. It would be 560 ns for matrix (24,48).

The error correction process it self would need 260 ns; However, as the consequences to work only in 13 clocks, the bigger matrix would need more of resources. Tabel 3 presents the results of FPGA resources utilization.

$$\frac{1}{50x10^{6}}\sec onds = 20ns$$

After around 300 error variations gave to the each systems, the results showed that matrix (8,16) had capability to correct maximum two error bits, and matrix (24,48) had capability to correct maximum three error bits. Yet, there were still restrictions in the error correction process. In matrix (8,16), if we gave errors at 6'th bit and 13'th
bit from the original codeword "1000100101101100", then the error codeword would be "1000101101101000". The error codeword would

ISSN: 9772477798001

become the input of the decoder. The error correction results showed that there was changes. The 6'th bit changed its values from '0' to '1', and the 13'th bit changed its values from '1' to '0'. It means that error correction worked properly.

On the other hand, if we put 6'th bit and 15'th bit as the error bits, then the result would be "0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0". It means that the error correction didn't
work properly this time. The error correcting decision region of matrix (8,16) had main role for these problems.

At section two has been described that each decoder has limitation due to the decision region. The most importance decision process in message passing was happen in the check nodes. It means that if the decision process or the decision region in the check nodes could be
discovered, than it would become easier to find the limitations of error correction process.

The figure 13 presents the check nodes decision region for correcting 6'th bit and 13'th bit. The 6'th bit decision is determined by f5 and f6 check nodes, and the 13'th bit is determined by f1 and f7. In general, decision regions for 6'th bit and 13'th have no a intersection.
In the f5 decision region, the C6 (representation of 6'th bit) would have a possibility to intersect with C1, C8, and C11, and in the f6 decision, the error correction process might have intersections possibility with C4, C3, and C12. Both the f5 and the f6 only have a intersection at C6.

In the f1 decision region, the intersections for 13'th bit correction might be happen with C0, C2, and C15, and in the f7, the intersections might be happen with C5, C9, and C14. Both the f1 and the f7 only have a intersection at C13.

Picture 14 presents the decision for correcting the 6'th bit and 15'th bit. The error correction process involved four parity nodes. The f1 and the f8 used for 15'th bit., and the f6 and f5 use for 6'th bit. In the decision regions, there is an intersection between decision region of 6'th bit and decision region of 15'th bit. Both the f5 and the f8 have an intersection at C8.

This term caused the error. These are the explanations. At this process, the channel didn't bring corruption for the 8'th bit, but the errors were given in the 6'th bit and the 15'th bit. Since, there were intersection at the f5 and the f8, the parity nodes decided that C8 had an error. So, the parity nodes or check nodes assumed that they had three error bits, not two error bits, right now. After the parity nodes finished the calculation, they would sent the results to the variable nodes calculation sub-system. From that values, the variable nodes calculation would decide "the correct" codeword. After "the correct" codeword

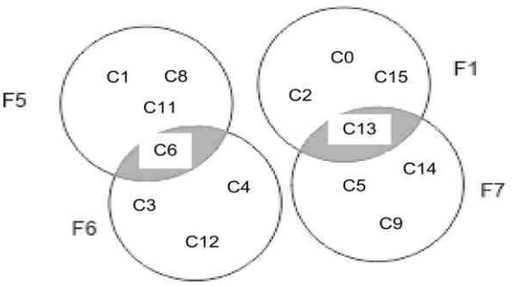

Fig 9 · Illustration of decision regions for correctiong error at the 6'th bit and the 13'th bit

had been decided, the validation would say that there was still an error in the codeword. The system would repeat the processes. Until the loop limit was reached, the system still could not give the true codeword. As the result, the system gave "0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0" as the output.

The picture below presents the decision regions for correcting error at 1'st bit. Since matrix (24, 48) had $w_c = 4$ and $w_r = 8$, every variable nodes would connect with four parity check nodes. Each of parity check nodes connected with eight variable nodes. It means that it would take four check nodes for correcting one error bit.

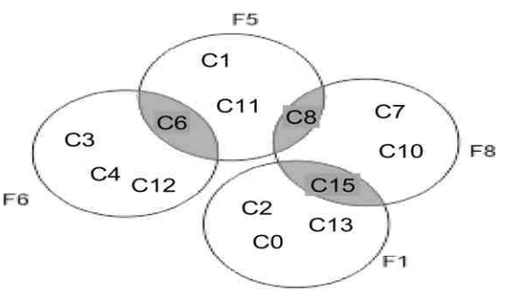

Fig 10. Illustration of decision regions for correcting error at 6'th bit and 15'th bit

If the error was at 1'st bit, the decision process would depend on the parity nodes f1, f2, f9 and f11. The f2 parity nodes have intersections in C1 with all parity nodes, C4 with the f1, and C2 with the f9. The f11 and the f9 parity nodes have intersection at C8; However, with this big number of interferences, it didn't affected the error correction for 1'st bit. On the other hand, in the matrix (8,16), for one error bit correction, if there were intersections more than one bit in the check nodes, then the error correction could not be done. The bigger sparse matrix would have

ISSN: 9772477798001

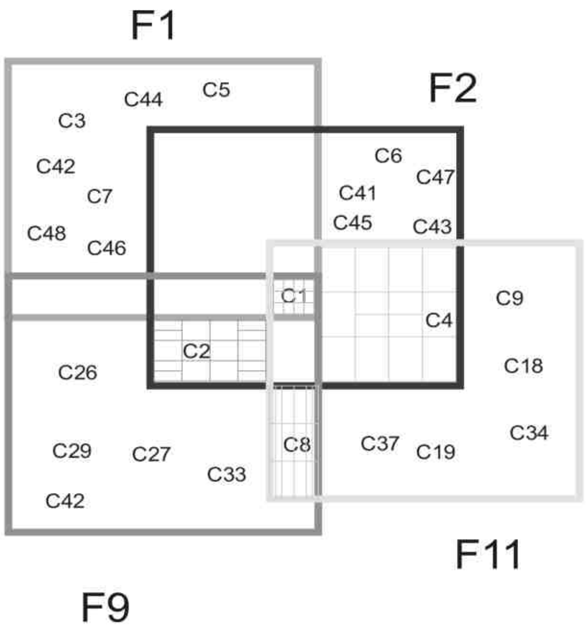

Fig 11. Illustration of decision regions for correcting error at 1'st bit

more capabilities of error correction. Moreover, the matrix (24,48) had the capability to correct until three error bits.

Example, The information '0 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 10' had three error bits on 1'st bit, 26'th bit, and 30'th bit. The error correction sub-system fixed the errors, and the result was '1 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 0 1 1 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 0'.

The full capability had not been reached. If the errors were at 1'st bit, 26'th bit, and 27'th bit, than the error correction result would be '0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0'.

The better decision regions would be needed to give more capabilities at error correction.

Tabel 2. FPGA Cyclone II EP2C20 Specifications [5]

Features	LEs	M4K RAM Block	Total Ram Bits	Embedded Multiplier	PLLs	Max I/O pins
	18752	52	239616	26	4	315

Tabel 3. Results of FPGA Resources Utilization

Matrix	*total logic elements*	total combinational functions	dedicated logic register	total pins	total memory bits	*total logic elements*
(8,16)	28%	22%	17%	11%	7%	28%
(24,48)	72%	57%	31%	11%	16%	72%

5. Conclusions

The Message passing algorithm gave improvement for error correction capabilities than the bit flipping algorithm. The computation was shorter, and the error correction could fix more than one errors.

The message passing could correct maximum two error bits for matrix (8,16) and maximum three error bits for matrix (24,48); However, better decision region will be needed for achieving more capabilities of error correction.

The systems could finish the error correction process in 13 clocks or around 260 ns; However, the bigger matrix would need more resources of FPGA as the consequence.

Acknowledgment

This work was supported by Telkom University Digital Technique Laboratory.

**aditia.tarigan.id@ieee.org,*
**ritapurnamasari@telkomuniversity.ac.id,*
**maydhona@telkomuniversity.ac.id*

References

[1]. GALLAGER, R. G. (1962*). Low-Density Parity-Check Codes. IRE TRANSACTIONS ON INFORMATION THEORY, 21-28.*

[2] Xin-Ru, Lee., Chih-Lung Chen, Hsie-Chia Chang, and Chen-Yi Lee. (2015). *A 7.92 Gb/s 437.2 mW*

ISSN: 9772477798001

Stochastic LDPC Decoder Chip for IEEE 802.15.3c Applications. IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS.

[3] Meng Li, Youngjoo Lee, Yanxiang Huang and Liesbet Van der Perre. (2015). *Area and energy efficient 802.11ad LDPC decoding processor.*

[4] Chia-Hsiang Chen, Wei Tang, Zhengya Zhang. (2015). *A 2.4mm2 130mW MMSE-Nonbinary-LDPC Iterative Detector-Decoder for 4×4 256-QAM MIMO in 65nm CMOS. IEEE International Solid-State Circuits Conference*. Michigan: IEEE

[5] Purnamasari, R., Wijanto, H., & Hidayat, I. (2014). *Design and implementation of LDPC (Low Density Parity Check) coding technique on FPGA (Field Programmable Gate Array) for DVB-S2 (Digital Video Broadcasting-Satellite). Aerospace Electronics and Remote Sensing Technology (ICARES), 2014 IEEE International Conference on (hal. 83-88)*. Yogyakarta: IEEE.

[6] ETSI EN 302 755 V1.3.1: "*Digital Video Broadcasting; Frame structure channel coding and modulation for a second generation digital terrestrial television broadcasting system (DVB-T2)* ".

[7] ETSI EN 302 307-1 V1.4.1: "*Digital Video Broadcasting; Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 1: DVB-S2*".

[8] Haykin, S. (2000). *Communication Systems 4th Edition.* New York: John Wiley & Sons, Inc

[9] Leiner, B. M. (2005*). LDPC Codes – a brief Tutorial*.

[10] Siegel, Paul H. *An Introduction to Low-Density Parity-Check Codes*. University of California, San Diego

[11] *Cyclone II Device Handbook, Volume 1* . (2008). San Jose: *www.Altera.com.*

Author information

Aditia Tarigan is an undergraduate student at Telkom University, Indonesia. Since 2012, he has been a research assistant at Telkom University RF Electronic Communication Lab.

Rita Purnamasari is a lecture at Telkom University, Indonesia. She received Master in Telecommunication Engineering from the Telkom Institute of Technology, Indonesia, in the 2012 Currently, She is the head of Digital Electronic Laboratory at Telkom University.

Efa Maydhona Saputra is a lecture at Telkom University, Indonesia. He received Master in Telecommunication Engineering from the Telkom Institute of Technology, Indonesia, in the 2013.