VHDL Design and FPGA Implementation of LDPC Decoder for High Data Rate

A. Boudaoud, M. El Haroussi, E. Abdelmounim
Applied physics department of FST
Hassan I University
Settat, Morocco

***Abstract*—In this work, we present a FPGA design and implementation of a parallel architecture of a low complexity LDPC decoder for high data rate applications. The selected code is a regular LDPC code (3, 4). VHDL design and synthesis of such architecture uses the decoding by the algorithm of BP (Believe propagation) simplified "Min-Sum". The complexity of the proposed architecture was studied; it is 6335 LEs at a data rate of 2.12 Gbps for quantization of 8 bits at the second iteration. We also realized a platform based on a co-simulation on Simulink to validate performance in BER (Bit Error Rate) of our architecture**.

Keywords—error correcting codes; LDPC codes; BP "Min-Sum"; VHDL language; FPGA

I. INTRODUCTION

LDPC codes were discovered by Gallager [1][2] in the early 1960. This remarkable discovery has been largely ignored by researchers for nearly 20 years, until the work of Tanner in 1981, in which he provided a new interpretation of the LDPC codes from a graphical perspective. Tanner's work has also been ignored by theorists for about 14 years until the late 1990s, when some coding researchers began to investigate the graphic codes and iterative decoding. Their research led to the rediscovery of Gallager's codes. They showed that a long LDPC codes with iterative decoding based on the Believe Propagation enable a performance error representing only a fraction of a decibel away from the Shannon limit [3][6][7][8]. This discovery makes the LDPC codes powerful competitors relative to turbo codes for error control when high reliability is required. LDPC codes have the advantage of turbo codes, it does not require a long interleaving to achieve a good error performance. Thus in 2004, an LDPC code was first standardized in a satellite broadcast DVB-S2 [9].

In this work, we are interested in building a regular LDPC code and study its performances in terms of complexity, data rate, latency and BER versus SNR for various iterations and quantifications.

We began by recalling the principle of LDPC codes in the first part; the second part is devoted to the implementation of said decoder and the last one to validate our design.

II. THE LDPC CODES

A. Principle Of LDPC Codes

An LDPC code can be represented by its parity check matrix (noted H) or by a bipartite graph (Tanner graph). In the example of Fig. 1, the rows of the matrix are represented by squares and are called check nodes, the columns of the matrix are represented by circles and are called data nodes and the"1" represent the edges in the graph.

$$H = \begin{vmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}$$

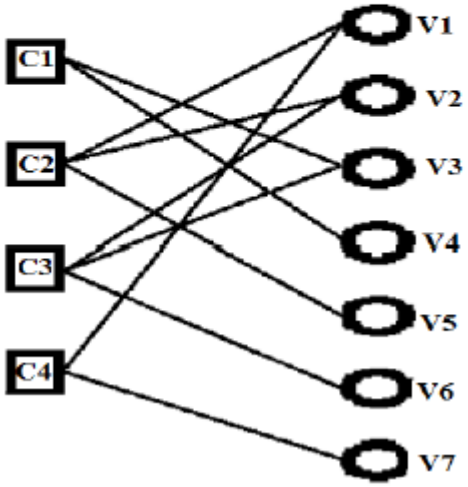

Fig. 1. Example of parity check matrix and its correspondent Tanner Graph

B. Encoding of LDPC codes

The encoding operation consists first in finding a generator matrix G such that $G.H^T = 0$. The work of T. J. Richardson and Urbanke R.L [4] showed that the check matrix must undergo a pre-processing before the encoding operation. The aim of this pre-processing is to put this matrix in a lower pseudo-triangular form, as shown in Fig. 2, using only permutations of rows or columns. This matrix is composed of 6 sparse sub-matrix, referenced A, B, C, D, E and a lower triangular T sub-matrix. The size of T sub-matrix is (m-g)×(m-g) where g is smaller as possible. Once the H pre-processing is completed, the coding principle is based on the resolution of the system represented by the equation (1) [4]. Where C is a code word:

$$C.H^T=0^T \qquad (1)$$

The pre-processing algorithm is described by Jean-Baptist Doré [10].

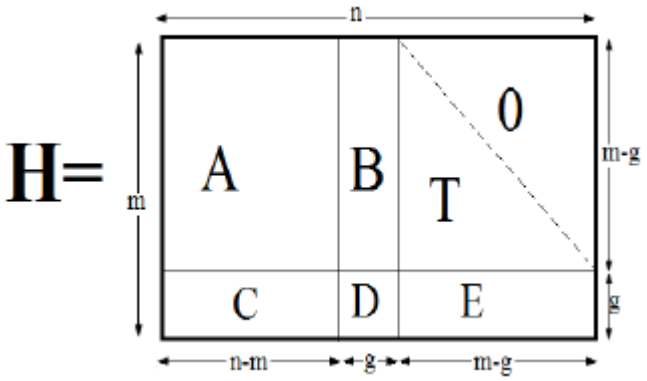

Fig. 2. Parity check matrix represented in lower pseudo-triangular form

C. Decoding of LDPC Codes

Decoding the LDPC codes is done from iterative algorithm; the most used is the BP (Belief Propagation). In our work, we have used the BP "Min-Sum" adapted to the hardware implementation. The algorithm consist to update, first the data nodes after, check nodes at each iteration and at the end make a decoding "Hard" decision that is the most likely codeword [10].

III. FPGA Implementation of LDPC Decoder

LDPC code discussed in this document is characterized by the H parity matrix given in (3), after we made the necessary transformations on H [9] to determine the generator G matrix. This G matrix is the basis of the LDPC encoder which calculates the word C code from the u information as follows:

$$C=u.G \qquad (2)$$

The LDPC decoder is designed in VHDL and implemented on the EPC4CE115F29C7 type of FPGA Altera using the simplified BP "Min-Sum". The decoder circuit is given in Fig. 5. Table I summarizes the complexity (in Logic Elements LE), data rate , and decoder latency for 2nd, 10th and 20th iterations quantized on 5, 6, 7, 8 and 11 bits.

Operators used to update variable nodes and check nodes are illustrated in Fig. 3 and Fig. 4 respectively.

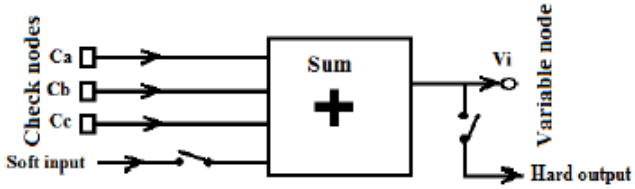

Fig. 3. Data nodes updating operator

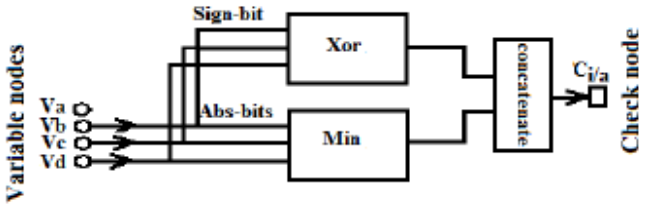

Fig. 4. Check nodes updating operators

The functional simulation on Quartus II tool (see Fig. 6) shows the parallel computing implemented, allowing the updating of outputs after the first active edge of the clock. Where, a maximum latency equals to one clock cycle.

$$H = \begin{bmatrix}
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1\\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&1&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&1&0\\
0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&1\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&1&0&0&0\\
0&1&0&0&0&0&1&0&0&1&0&0&0&0&1&0&0&0&0&0\\
0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&1&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1
\end{bmatrix} \qquad (3)$$

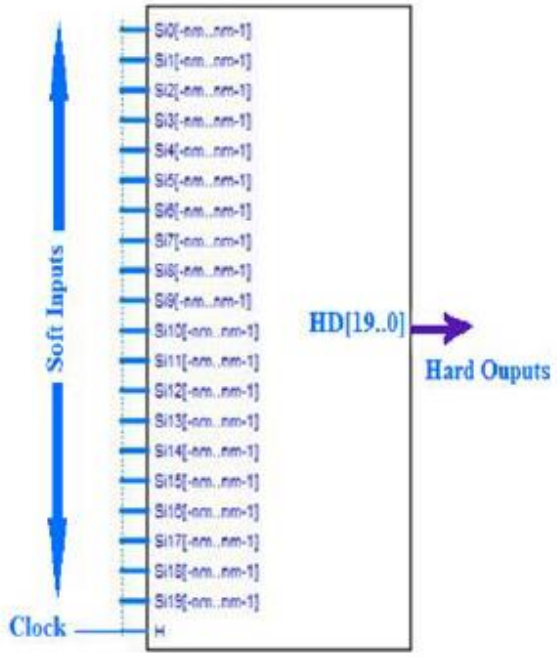

Fig. 5. External scheme of our LDPC decoder circuit

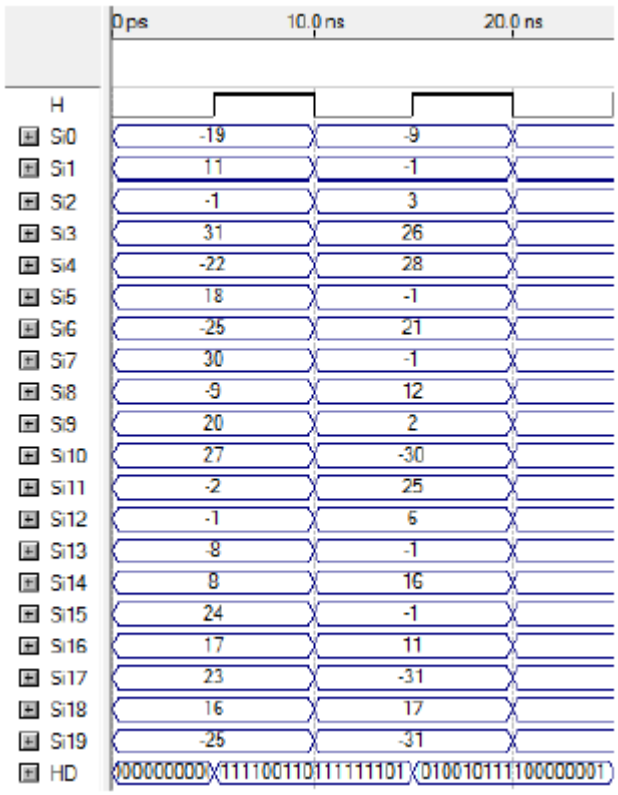

Fig. 6. Example of the decoder functional simulation

The evolution of complexity versus the iterations and the number of quantization bits is shown in Fig. 7. This shows that the complexities of the 10th and the 20th iteration are multiplied respectively by 5 and 10, relative to the 2nd iteration, whatever the number of quantization bits.

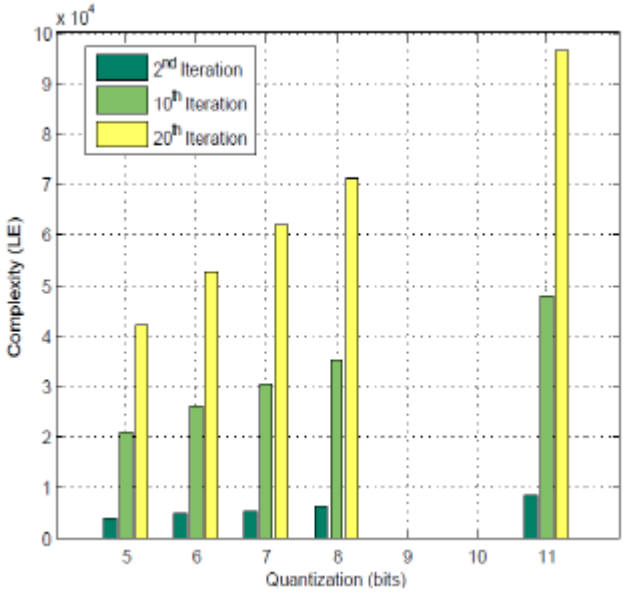

Fig. 7. Complexity evolution depending on the number of iterations and the number of quantization bits

TABLE I. DECODER PEFORMACES FOR DIFFERENT ITERATIONS AND NUMBER OF QUANTIZATION BITS

Iteration	Characteristics	Quantization				
		5bits	6bits	7bits	8bits	11bits
2nd ,10th and 20th	Pins number	121	141	161	181	241
2nd	Complexity (LE)	3817	4715	5515	6335	8595
	Freq. Max (Mhz)	109.06	108.11	106.61	106.04	99.82
	Latency (ns)	9.17	9.25	9.38	9.43	10.02
	Throughput (Gb/s)	2.18	2.16	2.13	2.12	2.00
10th	Complexity (LE)	20881	26035	30595	35151	47755
	Freq. Max (Mhz)	106.94	105.60	104.92	103.84	97.96
	Latency (ns)	9.35	9.47	9.53	9.63	10.21
	Throughput (Gb/s)	2.14	2.11	2.10	2.08	1.96
20th	Complexity (LE)	42126	52685	61945	71171	96705
	Freq. Max (Mhz)	105.85	104.83	103.59	102.08	95.79
	Latency (ns)	9.45	9.54	9.65	9.80	10.44
	Throughput (Gb/s)	2.12	2.10	2.07	2.04	1.92

IV. VALIDATION OF THE DECODER

After functional simulation on Quartus II, we validated our decoder in the digital transmission chain designed on the Simulink tool (see Fig. 8)[5]. This chain of Co-simulation also allowed us to measure the BER performance based on the SNR for various iterations and different quantization bits.

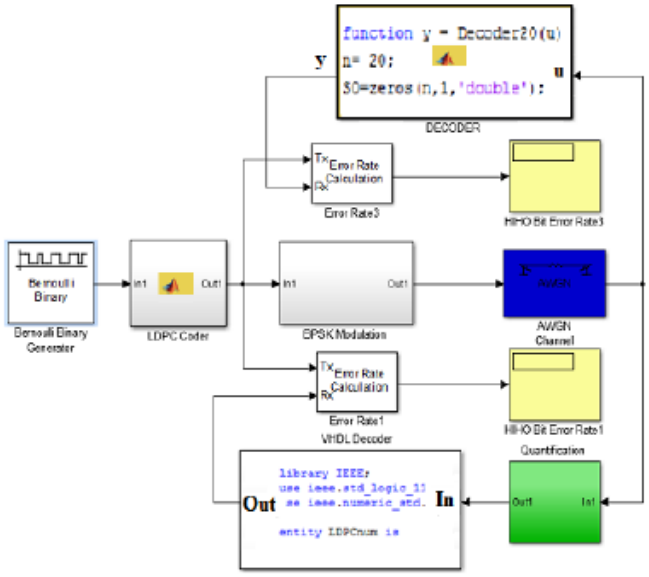

Fig. 8. Validation platform of our decoder circuit on Matlab/Simulink

Fig. 9 shows the BER performance of the decoder for the real data (2nd, 10th and 20th iteration), where one can see that the value of the SNR won in the 10th and 20th iteration,

compared with the second iteration for a given BER is negligible in comparison to the complexity, which is multiplied respectively by 5 and10.

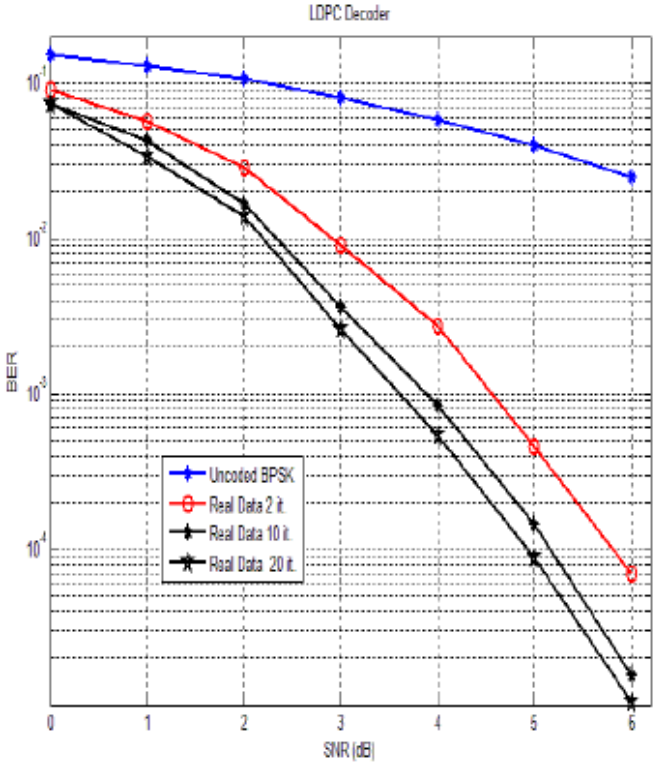

Fig. 9. BER Performances versus the SNR of the decoder for the 2nd 10th et 20th iteration (Real data)

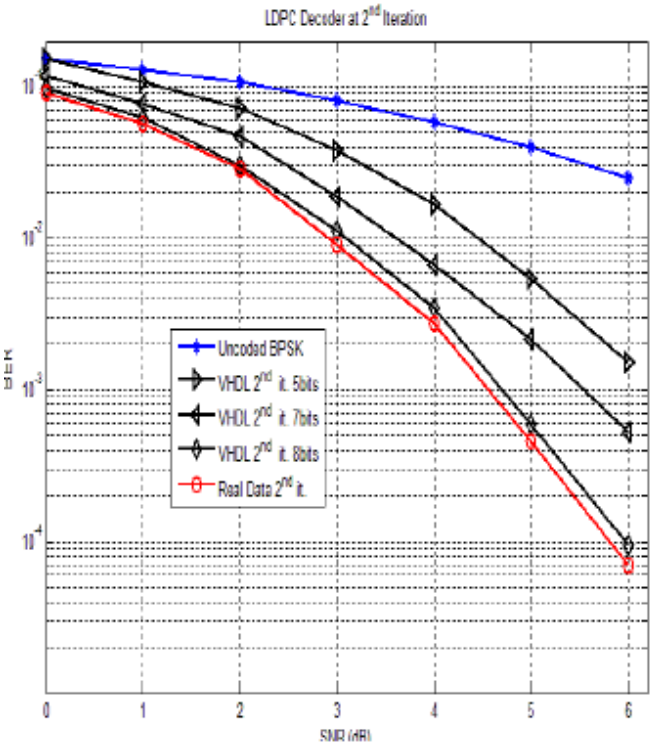

Fig. 10. BER performances for real data and VHDL implementation for the 2nd iteration

Fig. 10 shows this BER performance for the VHDL implementation for the second iteration (with quantifications of 5 bits, 6 bits, 7 bits and 8 bits). The results show that quantification of 8 bits gives BER performance very close to those of real data.

The comparison with other designs (see Table II), shows that our design has a very low complexity, higher data rate and acceptable BER performance.

TABLE II. COMPARISON WITH OTHERS DESIGN

References	[13] 2013	[12] 2009	[11] 2012	[14] 2008	This Work
LLR bitwidth	1 U	7 U	5 U	9 U	8 U
Clock (Mhz)	125.71	128	96	-	106.04
FPGA	Altera Cyclone4	Altera Stratix 2	Xilinx Virtex-5	Xilinx Virtex-2p	Altera Cyclone4
Algorithm	Hard decision	Min-sum	Min-sum with correction	Min-sum with correction	Min-sum
Iterations	12	8	10	10	2
Throughput (bps)	8.54G	465M	950M	1.54M	**2.12G**
ELs (k)	39.995	103.67	71.4	1.2	**6.34**
SNR(dB) at BER=10^{-4}	-	1.94	3.02	4.5	5.8
BER at SNR=1.5dB	-	$4\ 10^{-4}$	$3\ 10^{-2}$	-	$4\ 10^{-2}$

U: Uniform quantization

We note that:

For the data rate s in the table II, they are evaluated without removing the parity bits.

For the complexity, some authors have used Stratix and Virtex FPGA circuits, where the complexity is evaluated by different units of LE (Logic Element), which therefore requires an analysis that is performed as follows:

For Stratix FPGA from Altera, where the complexity is expressed in ALUTs: LE=1.25*ALUT.[15].

For Virtex FPGA from Xilinx, the complexity is expressed in Slice and LUT, the approximate formula used is LE = Slice* 4*LUT * 0.83 [16].

V. CONCLUSION

In this paper, we designed in VHDL and implemented on the FPGA circuit an LDPC decoder, starting from its parity check matrix, and the determination of all the necessary means for its implementation, namely the generator matrix and decoding equations using the simplified method BP "Min-Sum". Then we tested and validated it on a platform developed in the Simulink software for the co-simulation with Dsp Builder software.

The results show that our design has a high data rate, low latency and very low complexity. The BER versus SNR can be further improved by the increase in the code size and keeping the same principle of parallelism.

REFERENCES

[1] R. G. Gallager, "Low-density parity-check codes", IRE Trans. Inform. Theory, vol. IT-8, pp. 21–28, (Jan. 1962).

[2] R. G. Gallager, "Low-Density Parity-Check Codes", Cambridge, MA: M.I.T. Press, (1963).

[3] C. E. Shannon, "A mathematical theory of communication", Bell Syst. Tech. J., vol. 27, pp. 379-423, (1948)

[4] T.J. Richardson and R.L Urbanke, "Efficient Encoding of Low-Density Parity-Check Codes", IEEE Trans. Inform. Theory, vol. 47, pp. 638-656, February 2001.

[5] A.Boudaoud, E.Abdelmounim, A. Barazzouk, J. Zbitou and M.Belkasmi, "FPGA Implementation of HIHO and SIHO Decoders for DSC Codes", IEEE explorer, pp. 1461 –1464, (April 2014).

[6] D. MacKay and R. M. Neal, "Near shanon limit performance of low density parity-check codes", Electronic Letter, (August 1996).

[7] J. Pearl, "Probabilistic reasoning in intelligent systems: networks of plausible inference", S. Mateo, Ed. Morgan Kaufmann Publishers, (1988).

[8] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Analysis of low density codes and improved designs using irregular graphs", Proceeding of 30th ACM Symp. on Theory of Computing, (1998).

[9] Digital Video Broadcasting (DVB) ; Second Generation framing structure, channel coding and modulation systems for broadcasting, Interactive services, news gathering and other broadband satellite applications, ETSI EN 302 307 Std. v1.1.1, (2004).

[10] Jean-Baptiste Doré, "Optimisation conjointe de codes LDPC et de leurs architectures de décodage et mise en oeuvre sur FPGA" , Thèse pour obtenir le grade de Docteur à l'INSA de Rennes, Spécialité : Electronique, (Soutenue le 26 Octobre 2007).

[11] H. Li, Y. S. Park, and Z. Zhang, "Reconfigurable architecture and automated design flow for rapid FPGA-based LDPC code emulation", in Proc. ACM/SIGDA Int. Symp. F. Program. Gate Arrays. Monterey, CA, USA: ACM, (Feb. 2012), pp. 167–170.

[12] H. Ding, S. Yang, W. Luo, and M. Dong, "Design and implementation for high speed LDPC decoder with layered decoding", in WRI Int. Conf. Commun. Mob. Comput. Yunnan: IEEE, (Jan. 2009), pp. 156–160.

[13] IPrium Ltd., I.6 LDPC encoder/decoder IP core short description, (2013). [Online] Available: https://www.iprium.com/bins/pdf/iprium_ug_ i6_ldpc_codec.pdf.

[14] S. M. E. Hosseini, K. S. Chan, and W. L. Goh, "A reconfigurable FPGA implementation of an LDPC decoder for unstructured codes", in Int. Conf. Signals Circuits Syst. Nabeul, Tunisia: IEEE, Nov. 2008, pp. 1–6.

[15] Altera Corporation, Stratix II Device Handbook, Volume 1. Page 14. [Online] Available: https://www.altera.com/literature/hb/stx2/stratix2_handbook.pdf

[16] Peter Hailes, Lei Xu, Robert G. Maunder, Bashir M. Al-Hashimi and Lajos Hanzo, "A survey of FPGA-based LDPC decoders", in IEEE Communications Surveys & Tutorials. Volume: 18, Issue: 2, (Second quarter 2016), pp. 1098-1122.