Design, Simulation and Hardware implementation of Low Density Parity Check Decoders using Min-Sum Algorithm

Abdessalam.Ait madi[1], Anas.Mansouri[2], Ali.Ahaitouf[1]

[1]Sidi Mohammed Ben Abdellah University, Faculty of Sciences and Technolgy, Signals Systems and Components Laboratory B.P, 2202, Fez, V.N 30000 Morocco

[2]Sidi Mohammed Ben Abdellah University, National School of Applied Sciences of Fez, Signals Systems and Components Laboratory, Fez, Morocco

Abstract

A Variable Node Processing Unit (VNPU) and a Check Node Processing Unit (CNPU) are designed in order to be used in Low Density Parity Check (LDPC) decoding by the Min-Sum Algorithm (MSA). The designed blocks are fully parallel and flexible to be used for different block length when a regular (3, 6) LDPC codes are required. The proposed VNPU and CNPU have been first designed and implemented in software using Simulink tool following a modular design approach. In a second step, these blocks were described and simulated using Very High Speed integrated circuits Hardware Description Language (VHDL). Comparison between these two implementations shows that the proposed high level methodology is efficient to test and validate digital circuits before being implemented on desired Field Programmable Gate Array (FPGA) device.

Keywords: *MSA, LDPC, VHDL, hardware, implementation, FPGA.*

1. Introduction

LDPC codes are widely used in more applications for next generation data assessment such Digital Video Broadcasting for the second generation digital satellite and terrestrial television broadcasting system (DVB- S2) [1], (DVB-T2) [2]. The LDPC codes [3] can be efficiently decoded with the original belief-propagation (BP) algorithm or the sum-product algorithm (SPA). The BP algorithm is recognized by its good error correcting performance [4] but needed many multiplications to update check and variable nodes which make it difficult to use for efficient hardware implementation. Several modified versions of the BP decoding algorithm have been introduced such log-likelihood ratio belief propagation (LLR-BP) in which the multiplications are transformed in additions. In spite of this considerable simplification, the LLR-BP remains still more difficult for digital circuits' implementation. In fact it need the Look-Up Table (LUT) to implement the hyperbolic tangent and the involution transform function $-\log(\tanh(x))$ in the "$\tanh-rule$" and the Gallager's approach LLR-BP based algorithm. Indeed, the use of the LUTs introduces quantization effects and decoding delay leading to some performances degradation. Thus, the non-LUT-based approaches, which employ only combinational logic only, were introduced in order to avoid the unbreakable delay of LUTs. For example the MSA algorithm [5, 6] which greatly reduces the hardware implementation complexity and computations induces no negligible performance degradation due to the overestimation in the outgoing message from the check to variable node. Despite this drawback the MSA remains the main approximated version that consumes less area.

Designing the entire LDPC decoder in VHDL becomes a very difficult task when the sizes of the parity check matrix increase. The LDPC decoders can be constructed using a modular approach and the basic LDPC decoding operations. So it is possible to use auxiliary tools in this development. In order to test and validate the design a simple Matlab application script receives the parity check matrix of the code, interprets it and accordingly, creates

and connects a full set of module units needed to implement the required LDPC decoder.

Before implementing the designs into FPGA devices we can use high level abstraction to verify its functionality. In this case the CNPU and the VNPU designs are first developed and tested in the Simulink tool. The VHDL codes are then written by hand. After checking errors and warnings in ModelSim, the correctness of the functionality is verified by using a test bench written also in VHDL. The modules developed in VHDL codes are integrated in a Simulink tool for co-simulation by using Electronic Design Automation (EDA) Simulator Link for direct hardware design verification. The Matlab script is used to call and connect the integrated blocks and sends stimuli to running designs to be validated and tested. As response, the designs outputs are sent back to the matlab script. The most advantage of the co-simulation is the real system testing, therefore suppressing all possible misinterpretations present in a pure simulator. In other cases, co-simulation may be the only way to simulate a complex design in a reasonable amount of time.

This paper is organized as follow. Section II presents an overview of LDPC codes. The processing units VNPU and CNPU for LDPC decoder are developed in section III. The section IV describes the details for the VNPU and the CNPU designing in Simulink tool. The implementation in VHDL codes of its processing units is developed in section V when section VI presents an overview of the HDL co-simulation between ModelSim and Simulink. Finally, the simulation results are shown in the section VII and section VIII concludes the paper.

2. Overview of LDPC codes and Decoding Algorithm

2.1 LDPC codes

The LDPC codes are binary linear block codes that have a low density parity check matrix H. The LDPC encoder encodes a K length inputs binary message (x_0, x_1... x_{K-1}) into a N bits systematic LDPC codeword X=(x_0, x_1, ..., x_{K-1}, x_K , …, x_{N-1}). The valid codeword X∈C have to satisfy

$$HX^T = 0 \qquad \forall X \in C \qquad (1)$$

Where H is the parity check matrix and C is the set of the valid codeword. Each column in H is associated to a bit of the codeword and each row corresponds to a parity check. In the Tanner graph [7], the codeword bits are shown as the variable nodes (VN) and the parity check as the check nodes (CN). The VN is connected by edge to a CN in the Tanner graph if and only if the corresponding codeword bit takes part in the corresponding parity check equation. The number of edges on each node is called the node degree (see figure 1).

An LDPC is called (d_v, d_c) regular LDPC code if in its bipartite graph, every VN is connected to d_v CN and every CN is connected to d_c VN; otherwise it is called an irregular LDPC code.

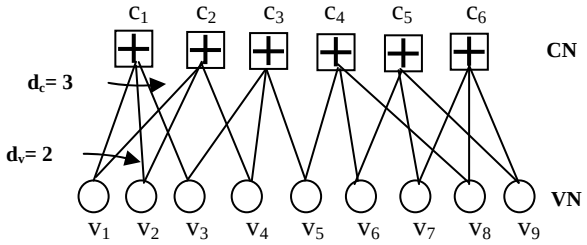

Fig. 1 Tanner graph of a binary regular (2, 3)-LDPC code of length 9. The graph has 9 VN and 6 CN.

2.2 Min-Sum Algorithm [5, 6]

The decoding procedure of the LDPC code uses the message-passing algorithm. It exchanges soft-information iteratively between VN and CN. The decoder halts if equation (1) is satisfied. If the decoding algorithm does not halt after a prefixed maximum number of iterations the decoding procedure fails. VN and CN are updated respectively by equation (2) and (3). In this approach log-likelihood ratios (LLRs) are used as messages exchanged between VN and CN connected by edges. On the figure 2 (b) each CN m computes the messages $L_{m\to n}(x_n)$ to send to each VN n as

$$L_{m\to n}(x_n) \approx \prod_{n'\in N(m)\backslash n} sign\left(Z_{n'\to m}(x_{n'})\right) \left(\min_{n'\in N(m)\backslash n}\left(\left|Z_{n'\to m}(x_{n'})\right|\right)\right) \qquad (2)$$

Where $L_{m\to n}(x_n)$ is the LLR of the n^{th} symbol which sent from CN m to VN n and $N(m)\backslash n$ is the set of the VNs which are connected to the CN m with the variable node n excluded when $Z_{n\to m}(x_n)$ is the LLR of the n^{th} symbol which is sent from the VN n to the CN m.

In the variable node update step (see figure 2 (a)), each VN n computes the message $Z_{n\to m}(x_n)$ to send to each CN m as

$$Z_{n\to m}(x_n) = L(x_n|y_n) + \sum_{m'\in M(n)\backslash m} L_{m'\to n}(x_n) \qquad (3)$$

Where $L(x_n|y_n)$ is the LLR of the n^{th} symbol received from the channel and $M(n)\backslash m$ is the set of the CNs

which are connected to the VN n with the CN m excluded .The LLR of the VN n is performed by equation (4) which used to make hard decision, such $\hat{x}_n = 0 \quad if \quad Z_n(x_n) \geq 0 \quad or \quad \hat{x}_n = 1 \quad elsewhere$

$$Z_n(x_n) = L(x_n|y_n) + \sum_{m' \in M(n)} L_{m' \to n}(x_n) \qquad (4)$$

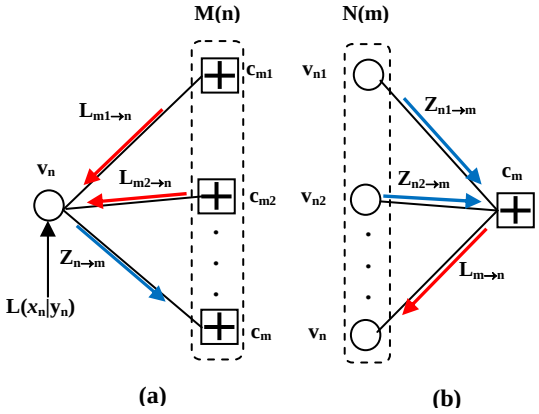

Fig. 2 Message updates in the MSA. (a): VN update and (b):CN update

3. Processing units for LDPC decoder

Iterative decoder can be constructed by considering each VN and CN of the tanner graph as processing units, and a connection between them as a bidirectional communication channels through which the processed information is sent (see figure 2 (a) and (b)). Since, the decoder works on the soft information, the sent messages between nodes are real values. In order to represent these values in fixed-point two's complement representation for reduced hardware implementation, we need to quantize them authorizing some performance loss as a result of the quantization.

3.1 VN Processing unit: VNPU

The architecture of the VNPU is the same as the other LDPC decoder designs [8]. In fact, the parallel design can be developed with only combinatorial logic. Therefore, the d_v successive $L_{m\to n}(x_n)$ messages coming from the CNPU are added together with $L(x_n|y_n)$ to form $Z_n(x_n)$ in equation (4). The most significant bit (MSB) of $Z_n(x_n)$ is the sign bit which is used for hard decision to estimate the n^{th} bit in the received word from the channel, one for the negative value and zero for the positive value .To update each output $Z_{n\to m}(x_n)$ in each iteration, the corresponding message value $L_{m\to n}(x_n)$ sent from the CN m to the VN n should be subtracted from $Z_n(x_n)$ (see figure 3). This type of implementation requires an adder $\sum$ capable of adding $d_v + 1$ inputs of s bits fixed-point two's complement representation as well as d_v outputs s bits substractors to perform d_v subtractions, where s is the length of binary fixed-point number. This means that a high number of gates is required to implement just a single processing unit, but has the great advantage of a minimum delay system (high throughput).

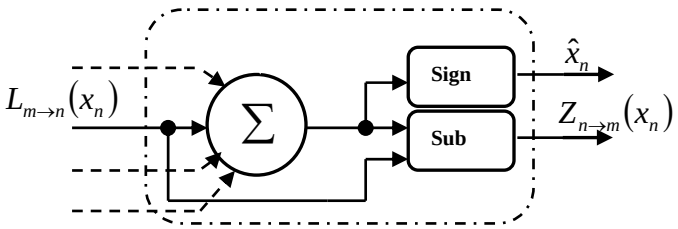

Fig. 3 VNPU block diagram considering parallel configuration: update of the message sent from VN n to CN m

3.2 CN Processing unit: CNPU

This module receives d_c inputs. In the same way as the VNPU, each input is a s bit fixed-point two's complement representation number. To compute equation (2) in hardware we separate the operation into sign and magnitude calculations of the two incoming messages according to the following equation.

$$out = sign(a).sign(b).\min(|a|,|b|) \qquad (5)$$

Figure 4 represents a simplified boxplus unit shown in [8] for two inputs *a* and *b* in which the overall sign *Final_sign* is performed by using a logical two input, sing(a) and sign(b), XOR operator. The *Final_sign* signal is set to zero for the positive sign and to one elsewhere. By using a *mux* block, the output *min* or *–min* of the comparator will be driven to the signal *out* taking into account the sign of the *Final_sign* signal value.

To construct a modular CNPU, the boxplus unit will be instantiated in parallel manner to carry out the d_c messages $L_{m\to n}(x_n)$. The complexity of the boxplus operation on the parallel implementation requires a boxplus chain of d_c inputs. As results the number of gates required increases when compared to serial or semi parallel configuration. In spite of this disadvantage, parallel configurations are still used for high throughput applications.

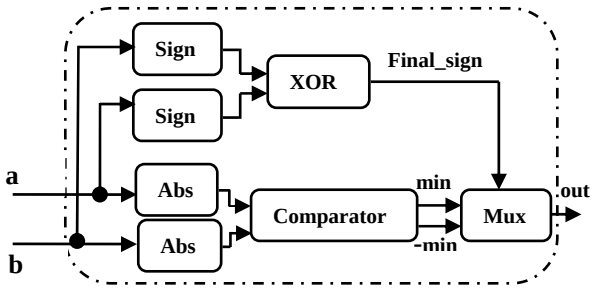

Fig. 4 Boxplus diagram considering parallel configuration.

4. Implementation of the VNPU and the CNPU designs in Simulink tool

The VNPU *ms_vnp36_top* shown in figure 5 has been designed in Simulink using basic adder block from the Simulink library. The variable node has fourth inputs the first three *in_vn1*, *in_vn2* and *in_vn3* are coming from various check nodes to which they are connected and the fourth one is the *LLR* message coming from the channel communication system. It performs the operations given in (3) and (4) and passes the outputs *out_vn1*, *out_vn2* and *out_vn3* to the check nodes and then the hard decision is making, to estimate the n^{th} bit in the received word in the same manner as described in (4).

To emulate quasi similar hardware functionality the blocks set used in this design works in fixed-point two's complement format with a fixed word length to 8 bits, the MSB bit corresponding to the sign, the next two MSBs bits represent the integer part and the five remain bits represent the fractional part. Thus, all the data will be delimited in the [minimum, maximum] interval in the 8 bit fixed-point two's complement format. Consequently, if a value computed by the block is outside this interval, it will be immediately replaced by the maximum (respectively minimum) value achievable in the used representation, depending on the sign.

The Simulink design in figure 6 represents the CNPU *ms_cnp36_top*. It uses various blocks from Simulink library (absolute, min, xor, switch…). The block have six outputs, a positive value *|out_cni| (i=1 to 6)* is the minimum of the fifth value *|in_cnj| (j=1 to 6 and j#i)*, as example *|out_cn1|* is the minimum of the set *(|in_cn2|, |in_cn3|, |in_cn4|, |in_cn5|, |in_cn6|)*.The product $\prod_{j\neq 1} sign(in\_cnj)$ is the sign of *out_cn1*. All the blocks used in this design works also in 8 bits fixed-point two's complement representation, in the same way as the VNPU design.

The functionality of these designs can be verified using Simulink tool. Once the two Simulink designs VNPU and CNPU are verified and tested, we used the Matlab script to connect them in order to construct the LDPC decoder.

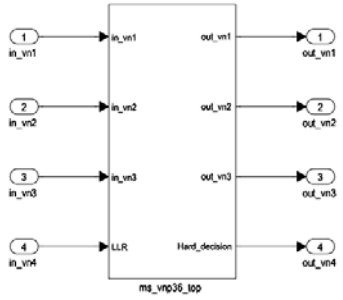

Fig. 5 VNPU design in the Simulink tool for $d_v = 3$ considering parallel configuration.

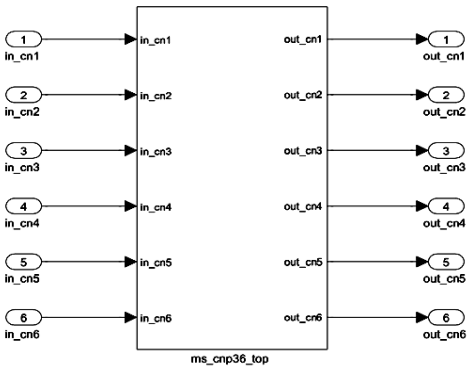

Fig. 6 CNPU design in the Simulink tool for $d_c = 6$ considering parallel configuration.

5. Implementation of the VNPU and the CNPU designs using VHDL codes

The work consists of transforming the two developed designs into computerized representation by using the Very High speed Hardware Description languages (VHDL). The blocks used in these designs are described and tested individually following a modular approach. After checking the eventual errors and warnings in each module, all the elements were wired to produce the required top designs. The VNPU top design *ms_vnp36_top* includes six adders, three unary minus operator and one comparator to zero which is used to make the hard decision. The CNPU *ms_cnp36_top* include twelve instantiated boxplus unit connected by internal signals. Each boxplus is implemented according to the figure 4. All the blocks used in this designs works in 8 bits

fixed-point two's complement representation.

6. HDL co-simulation with ModelSim and Simulink

The EDA Simulator Link software consists of MATLAB functions that establish communication links between the HDL simulator (ModelSim) and MATLAB and a library of Simulink blocks that one may use to include HDL simulator designs in Simulink models for co-simulation. When linked with Simulink, the HDL simulator works as the server and the Simulink tool as the client (see figure 7). By connecting these tools, the link simplifies verification by simulating the implementation and original specification directly. The Simulink tool is used to create test signals and software test benches for HDL code, it also provide a behavioral model for an HDL simulation. In this case, the HDL simulator responds to the simulation requests received from the co-simulation blocks in the Simulink model. In the linked session, the simulation progress and results can be followed both in the Simulink and in the HDL simulator. In addition, the EDA Simulator Link software allows us to validate and simulate the design in several environments. In our test strategy the test scenario allows the VNPU and the CNPU designs, to be tested either in Matlab, Simulink and ModelSim.
Using the Block Parameters dialog box for an HDL Co-simulation block, one can configure the input and output ports that correspond to signals (including internal signals) of a given HDL block or module. One can also specify sample times and fixed-point data types for individual block output ports if desired. The type of communication and communication settings used for exchanging data between the simulation tools can be also specified. One can specify the period and rising-edge or falling-edge for each clock to apply to a specified module. To manage the HDL simulator from the Simulink tool, the Block Parameters dialog box can be as well used to write a Tool Command Language (Tcl) commands to run before or after simulation.

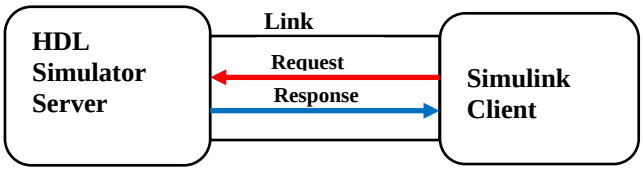

Fig. 7 Structue of the testing system in HDL co-simulation between ModelSim and Simulink

7. Results and discussions

7.1 Validation of the Simulink design

The Simulink environment is used in this design approach. On figure 8, the test vector (*0.5,-1, 1,-1*) is applied to the VNPU unit designed in the Simulink tool, the outputs are (*-1, 0.5, 1.5, 1*). From this result, one can see that the estimated bit value is one. We can also see the equivalent in 8 bits fixed-point of the inputs in (*in1_sf, in2_sf, in3_sf, llr_sf*) displays. As seen in figure 9, the CNPU units also designed in the Simulink tool yield the output vector (*0.4375, -0.4375, 0.4375, -0.4375, -0.4375, -0.437*) as a result for the corresponding 8 bits fixed-point inputs values (*0.4375, -1.969, 1.438, -4, -0.4375, -0.9688*) displayed in (*in1_sf, in2_sf, in3_sf, in4_sf, in5_sf, in6_sf*).From this example of simulation, the output magnitude |*out_cn1*| is equal to *0.4375*, which correspond to the minimum value of the set (*1.969, 1.438, 4, 0.4375, 0.9688*). Because, the number of a negative sign in the set (*-1.969, 1.438, -4, -0.4375, -0.9688*) is even, the sign of the output *out_cn1* will be positive.

7.2 Validation of the VHDL design

In the same way as in the Simulink designs, similar test vectors are applied to the VHDL model via a testbenchs written furthermore in VHDL. The VNPU and the CNPU behavioral simulations are shown in figure 10 and 11, which represents both the inputs and the outputs, signals in 8 bits fixed-point two's complement representation, and their equivalent in real format. The signs of the inputs and the outputs values are indicated by the MSBs value, a zero in MSB yield a positive result otherwise the results is negative. As example, the VNPU output value *out_vn1* is *11100000* in binary 8 bits fixed-point two's complement representation and -1 in real representation , for the CNPU the magnitude value | *out_cn1*| is *00001110* in binary 8 bits fixed-point two's complement format when *0.4375* is his value in the real format. It can be easily seen that we obtain the same results as shown above in Simulink design simulation.

7.3 Simulink and ModelSim Co-simulation

In this simulation, the co-simulate hardware VNPU and CNPU HDL inputs signals are coming from Simulink while the HDL outputs signals are driven back to the Simulink tool. In order to validate and test behavioral simulation of these VHDL models, the same stimuli used above in the Simulink designs simulations are applied to the HDL designs (see figure 12 and 13). One can clearly,

see that the similar results are obtained when comparing these HDL modules behavioral simulations with their corresponding in the Simulink tool.

7.4 MSA algorithm performances

A regular LDPC code (10, 5) check matrix H is used in this prototype design in the same way as in ref [9]. It is very smaller than matrix used in reality. The fixed degree node of each VN and CN are three and six, respectively. Encoded bits are binary-phase-shift-keying (BPSK) modulated and transmitted over the simulated Additive White Gaussian Noise (AWGN) channel. The number of maximum iterations is set to 16 at each Signal to Noise ratio (SNR) Eb/N0 value. For each SNR value, the codeword of length 10 have been transmitted until 2000 errors occurred or 500 codeword transmitted. The simulation program halts when the decoded codeword is valid or the maximum of the number of iterations is reached. In order to validate and simulate hardware implementation, the processing unit VNPU and CNPU designed in Simulink tool are called from the Matlab script application. The test scenario of the HDL co-simulation is as follow:

- The VNPU designed by the Simulink tool is used.
- The CNPU unit designed in VHDL is used as the co-simulate HDL component module instance in the Simulink tool.
- The matlab script application calls and connects the VNPU and CNPU designs to emulate the hardware implementation LDPC decoder using MSA algorithm.

Figure 14 shows the BER (Bit-Error-Rate) versus SNR, obtained for the LDPC decoder from Matlab in floating and fixed point. One can clearly see the good agreement between the performances in term of the BER of the MSA algorithm in floating-point and 8 bits fixed-point representation.

When comparing the MSA implementation both in Matlab, with VNPU and CNPU in the Matlab script managed Simulink and the designed VNPU in the Simulink associated to the VHDL CNPU modelization, we obtain quasi similar performances of the BER versus SNR evolution as shown on figure 15. The messages exchanged between the modules are all represented in 8 bits fixed-point two's complement format.

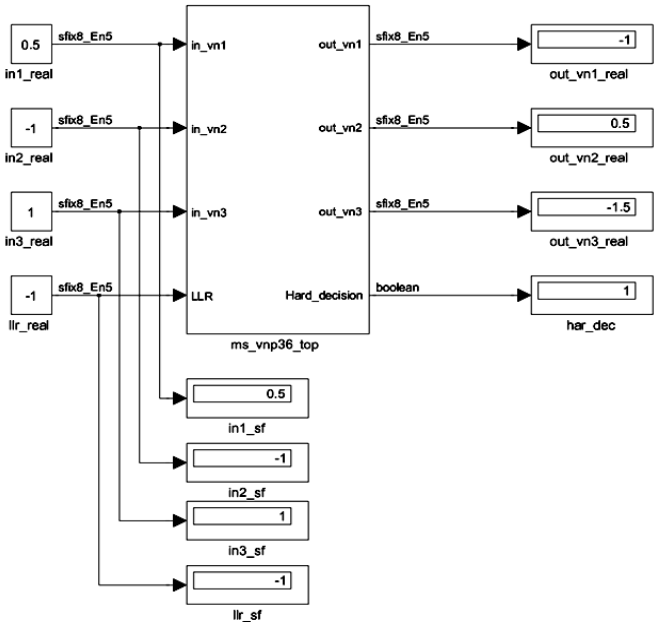

Fig. 8 VNPU functional validation in Simulink tool

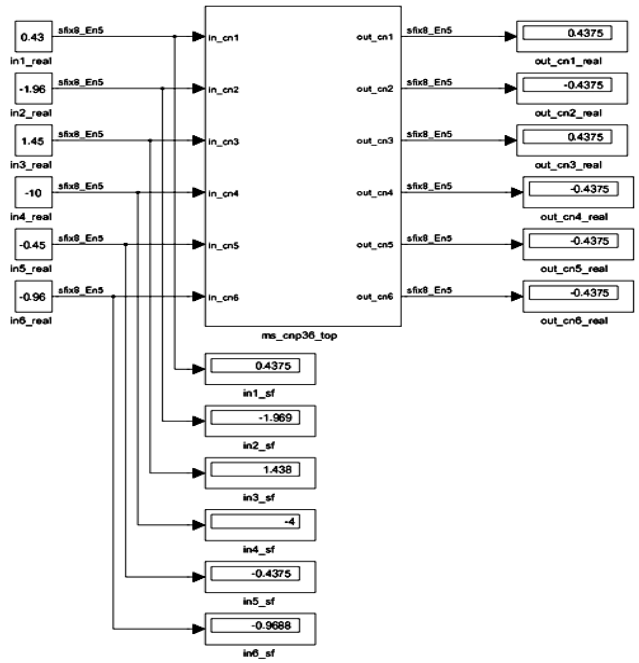

Fig. 9 CNPU functional validation in Simulink tool

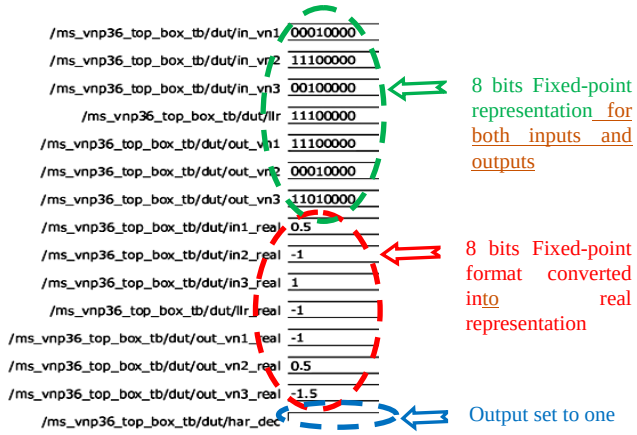

Fig. 10 VNPU functional validation in ModelSim tool

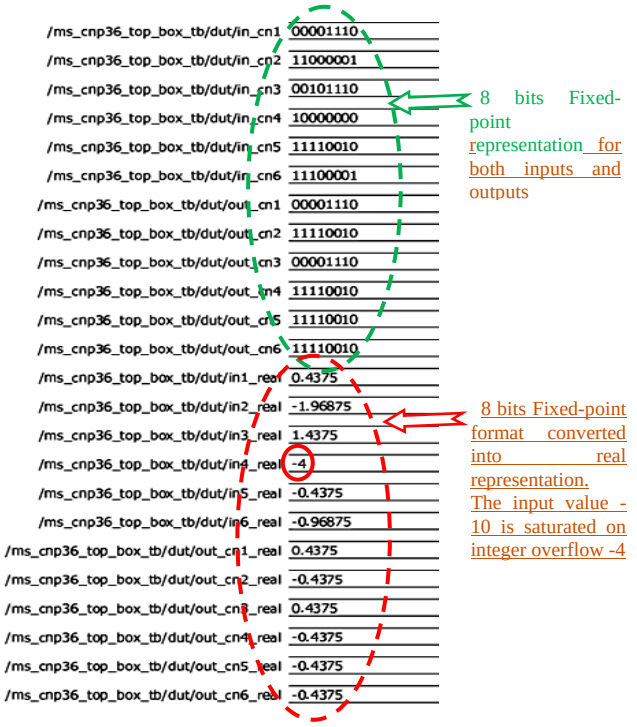

Fig. 11 CNPU functional validation in ModelSim tool

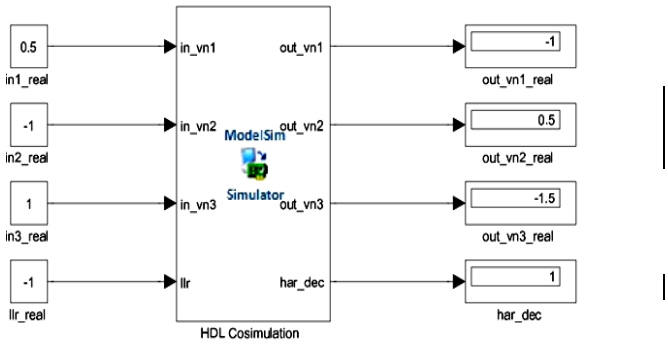

Fig. 12 VNPU functional validation in HDL co-simulation using Simulink tool and ModelSim

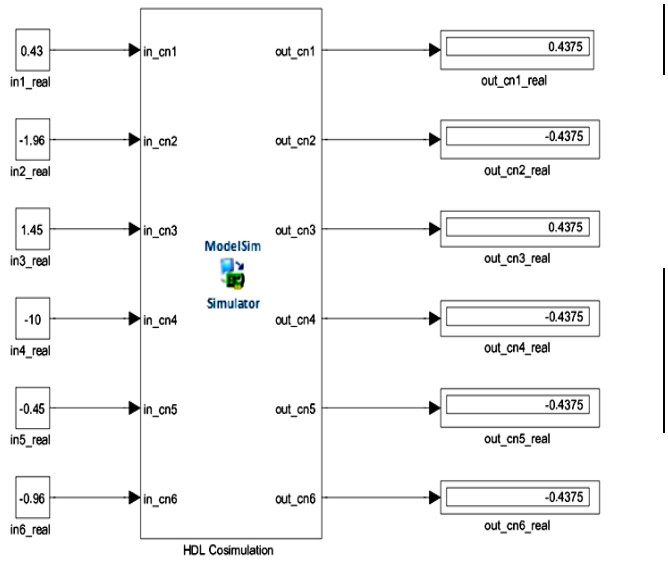

Fig. 13 CNPU functional validation in HDL co-simulation using Simulink tool and ModelSim

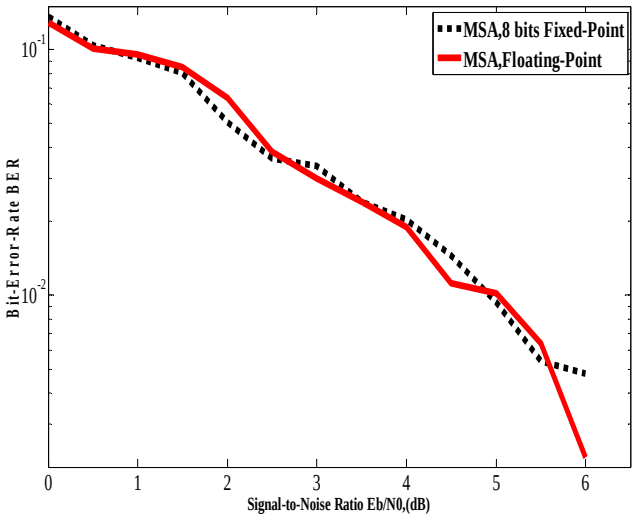

Fig. 14 BER obtained for LDPC decoder from Matlab in Floating and Fixed point

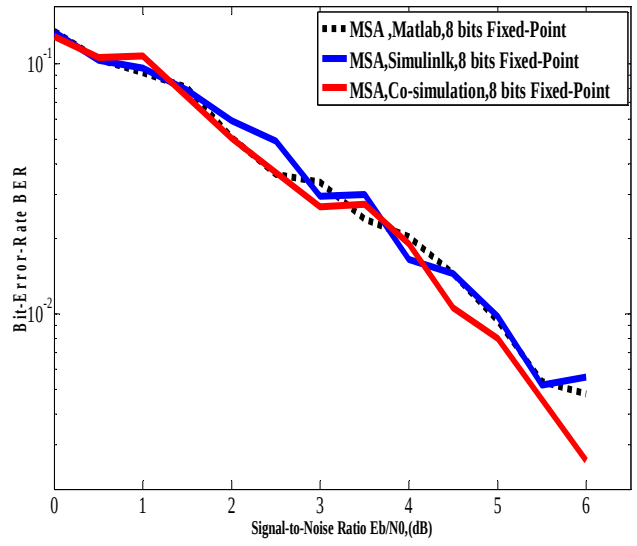

Fig. 15 BER obtained for LDPC decoder from Matlab,Simulink, ModelSim and Simulink co-simualtion

8. Conclusion

In this paper we have proposed an efficient high level modular approach to design the VNPU and the CNPU blocks for the MSA. It uses Matlab, Simulink and ModelSim. The EDA Simulator Link for use with Mentor Graphic ModelSim is used to allow communication between hardware component and Simulink model. The proposed strategy yield great advantages in terms of design complexity and development time.

References

[1] Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications. Draft ETSI EN 302 307 V1.1.1 (2004-06). European Standard (Telecommunications series).

[2] Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for second generation digital terrestriel television broadcasting system (DVB-T2). Draft ETSI EN 302 755 V1.1.1 (2009-09). European Standard (Telecommunications series).

[3] R.G.Gallager, “Low Density Parity Check Codes,” Cambridge, MA: MIT Press, 1963.

[4] D.J.C.Mackay,”Good Error-Correcting Codes Based on very Sparse Matrices,” IEEE Trans on information Theory, Vol .45, No. 2, March 1999, pp.399-431..

[5] M.P.C.Fossorier,M.Mihaljevic,H.Imai, “Reduced Complexity Iterative Decoding of Low Density Parity Check codes based on Belief Propagation,” IEEE Trans.Commun, Vol. 47,No.5, May 1999, pp. 673–680,.

[6] J.Chen,A.Dholakia,E.Eleftheriou,M.Fossier,X.-Y.Hu, ”Reduced-Complexity Decoding of LDPC codes,” IEEE Trans On Communications ,Vol 53,No. 8, August 2005, pp. 1288-1299.

[7] R.M.Tanner,”A recursive Approach to Low Complexity Codes,”IEEE Trans on information Theory, September 1981, Vol.IT-27, No.5, pp.533-547.

[8] G.Falcao, M.Gomes, J.Goncalves, P.Faia, V.Silva,“HDL Library of Processing Units for an Automatic LDPC Decoder Design,” IEEE PhD.Research in Microelectronics and Electronics (PRIME),pp.349-352, September 2006.

[9] S.M.Aziz, M.D.Pham,”Implemetation of Low Density Parity Check Decoders using a New High Level Design Methodology,” Journal of Computers, Vol.5, No.1, January 2010, pp.81-90.

Abdessalam Ait madi was born in Morocco. In 1993 he received the teaching degree in electronic engineering from the ENSET of Mohammedia (Morocco). In 2007 he received his master from the faculty of science and technology (FST) of Fez and joined the team of the Laboratory of Signals, Systems and Components (LSSC). His research interests include Channel coding, Optimization Techniques and Digital VLSI architecture for LDPC decoders.

Anas Mansouri received M.S. and Ph.D degrees in Microelectronics and Telecommunication from Faculty of sciences & technology, Fes, MOROCCO, in 2005 and 2009, respectively. He is a Assistant Professor in National School of Applied Sciences, Fes. His major research interests include VLSI and embedded architectures design, video and image Processing.

Dr. Ali Ahaitouf is actually **professor** in the faculty of science and technology of Fez, at the Sidi Mohammed Ben Abdellah University. He obtained his PhD in electronic in 1992 at the University of Metz in France, then he earned his Doctor title in physics in1998 in Fez. His research focuses on the semiconductor based components, designing analog and digital integrated circuits. He is currently head of the team components and microelectronics in the signal systems and component laboratory. He has published more than twenty articles in specialized review and he is co-author in the Analog Circuits : Application, Design and performances, Nova Sciences publishers, Inc.,2011, ISBN : 978-1-61324-355-8, Chapter 9 : Application of the ACO technique to optimization of analog circuits performances, pp 235–255.
.