A Building Block VLSI Design of an Information Decoder Using VHDL

Denner Paganoti de Almeida
Dept.of Electrical Engineering
Federal University of Paraná
Curitiba, Brazil
dpaganoti7@gmail.com

Jefferson Rodrigo Schuertz
Dept.of Electrical Engineering
Federal University of Paraná
Curitiba, Brazil
jeffersonschertz.eng@gmail.com

Sibilla Batista da Luz França
Dept.of Electrical Engineering
Federal University of Paraná
Curitiba, Brazil
sibilla@eletrica.ufpr.br

***Abstract*—Error correction codes allow the detection and correction of errors in a message that was possibly corrupted during the transmission or storage process. These codes add redundancy bits to the message before the transmission, enabling the decoder to recover the original one since the number of errors does not exceed the error-correcting capacity. Using a hardware architecture purposed for one of these decoders, based on information sets, a VLSI circuit, correspondent to one block of this decoder, was designed. The circuit that works as a reducing matrix that is the core of this decoder. It gives two output matrices, Gr and Gr_0, that are used to generate candidate messages. It had been made the description of the circuit in VHDL and the synthesis, layout and subsequent verifications were performed through the Cadence's tools, which resulted in an ASIC ready to be manufactured. The project was designed in the 130 nm technology.**

Keywords— Error correction code, decoder, information set, VDHL, VLSI.

I. INTRODUCTION

The error correction codes are used to improve reliability during the transmission and storage of data. Due to interference and noise in the channel and imperfections in the storage media, the original message could be corrupted, undermining the information. To recover the message, a series of bits, called parity bits (or redundancy bits), are added to the original message. The error correction codes, in general, are represented by the pair (n,k), where n is the number of bits of the codeword and k is the number of information bits, hence $m=n-k$ is the minimal Hamming distance d_{min}, that represents the number of positions in which two codewords differ. The application of these codes is represented in figure 1 [1]. The message **x** is codified by an encoder where a series of parity bits are included. During the transmission process, the message is corrupted by the noisy channel, changing the original content. Due to the redundancy bits, the message can be restored by the decoder through a series of logical and arithmetical operations.

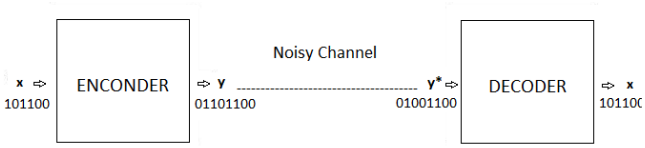

Fig. 1. Typical usage of error-correcting codes [1].

Several algorithms for decoding were studied and mentioned in the literature in the past few years. One of them proposes an optimized hardware implementation, based on information sets, to decode block codes [2]. The use of information sets in the decoding process was used by many researchers [2-8] who applied this concept in cryptographic problems. Prange [3] was the first to theorize the use of information sets for decoding cyclic codes. McEliece [5] developed a cryptographic public key system. The repercussion of these researches influenced others works who, in applying this theory, proposed the use of suboptimal decoding algorithms [11], which, when implemented in hardware, present better performance in terms of asymptotic complexity, but with a slightly lower error probability to the method by maximum likelihood when n tends to infinity. An information set is defined as any set of k linearly independent columns in the generator matrix **G**. The algorithm uses the most reliable symbols from the received message to reconstruct the original message. The hardware architecture proposed in [2] is composed of five blocks, as shown in figure 2. The decoding consists of the following steps: from a received message **x**, two vectors are generated. The first is **r**, corresponding to the abrupt decision, and the other one is **s**, corresponding to the reliability of symbols, sorted by decreasing order. Using this vector **s**, reduction operations are performed in the generator matrix **G**. These operations are based on the Gauss-Jordan elimination method and result in a matrix **Gr**. Another matrix $\mathbf{Gr_0}$ is also generated, consisting of '1's in the pivot's positions. From these matrices, some

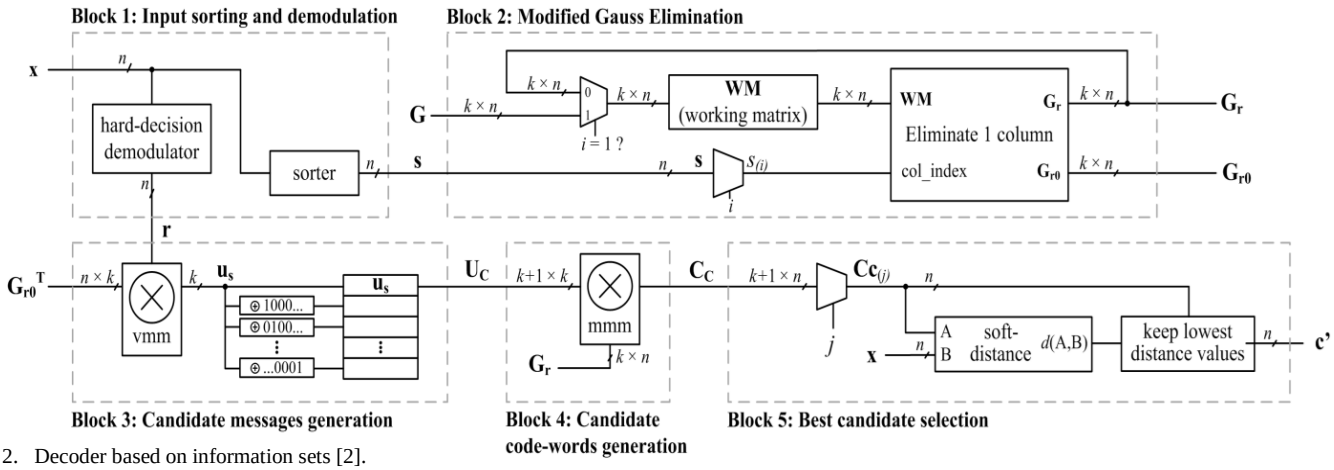

Fig. 2. Decoder based on information sets [2].

candidate messages are obtained and afterward, some candidates codewords. The message considered more reliable is chosen comparing all the candidate's codewords with the received message **x**, setting as the decoded the one who resembles it the most.

In this paper is presented the VLSI design of Block 2, as demonstrated by the figure 3, that performs reduction operations in the generator matrix **G**, from the order established by the elements of the reliability vector **s**. This matrix **G** has dimensions *(n, k)*, where *n* is the number of columns, *k* is the number of rows, and its format is **G=[I|P]**, where **I** is an identity matrix and **P** is a parity matrix $k \times (n-k)$. The reduction process applied in matrix **G** needs no more them $n-d+1$ clock cycles [2]. The Gr matrix is the matrix **G** reduced with *k* rows linearly independents and $\mathbf{Gr_0}$, with the same dimensions of **G**, and '1' in the pivot positions. Basically, the main contribution of this work is to reproduce in VHDL the main block of the decoder presented in [2] and then perform the synthesis, simulation and layout in a circuit design tool integrated to the point of generating a Device ready to be produced by the industry, the previous works had arrived only in the reproduction of the algorithm in a programmable logic device.

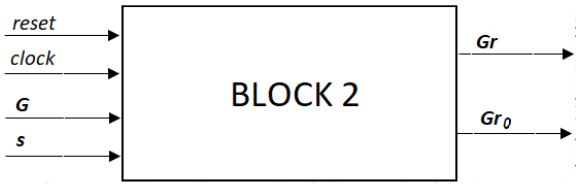

Fig. 3. Block diagram of the matrix scheduler.

After the study of the algorithm, the Block 2 was described in VHDL, targeting to design the VLSI layout circuit. Because of the high level of concentration and miniaturization of transistors in digital circuits, it is necessary to use a technique that describes all the steps in the design process of a chip [9-10]. These steps, known as VLSI design flow, encompass every conception processes of an ASIC, since its definition of the functionality until its manufacturing in a foundry. The design flow starts by describing the circuit into a hardware language (HDL) and goes through stages of synthesis, layout generation, simulations, post-layout verification and manufacturing. In the synthesis step, the code is transformed in an RTL code, which has the same functionalities of the original code, but in the form of a netlist. From this netlist, a layout is generated. In this stage, the standard cells and the power buses are positioned on this layout. The I/O pins are inserted, and cells are routed. In the next stage, layout verification is made to validate if it is correct. After that, it is simulated and, if it works properly, it is sent to be manufactured in a foundry. Following this design technique, an ASIC, correspond in Block 2, has been projected in 130 nm technology.

II. Development, Simulation and Tools Configurations

In the VLSI design flow, the first step is to implement the code of the algorithm of the desired circuit using a hardware description language. Using VHDL, the algorithm of Block 2 has been implemented, therefore it is possible to set it to work with matrices of any dimensions. This algorithm works as follows: from the vector **s**, a pivot is settled in the position defined by $L \times \mathbf{s}(L)$, where *L* is the row index of the matrix **G**. At this position, the element is verified to be equals to '1'. If it is not '1', the positions below the pivot in the column $\mathbf{s}(L)$ are verified to find an element like that and so, change the rows. If none of the positions bellow the pivot contains a '1', it moves forward to the next column indicated by $\mathbf{s}(L)$, and the new pivot gets the position $L \times \mathbf{s}(L+1)$. If there is '1' in the pivot position, an XOR operation is done between the pivots row and all those elements of $\mathbf{s}(k)$, ensuring at the end of it that $\mathbf{s}(L)$ is linearly independent. With every element at the row being '0', except the pivot, the index *L* is incremented by one and $\mathbf{Gr_0}$ receives a '1' in the same position as one of the pivots. This process should occur *k* times within $n-d+1$ clock cycles. By the end of this process, the result must be a matrix $\mathbf{Gr_0}$, that is a matrix made entirely of zeroes and receives '1' in the pivot positions, and **Gr**, which is the matrix **G** with *k* linearly independent columns. In figure 4, this process is made in a matrix **G**(7, 4, 3) with the vector **s** = (6, 5, 7, 3, 2, 1, 4). The circuit was designed on these dimensions. It is possible to see that the reduction process occurred in columns 6, 5 e 7 of the matrix **G**, making them linearly independent. It is also possible to see that the reduction process failed in the column $\mathbf{s}(L) = 3$, because a pivot equals to '1' cannot be found, and so it moves forward the column $\mathbf{s}(L) = 2$. By the end of 5 cycles of the clock, the process generated two matrices, **Gr** and $\mathbf{Gr_0}$, both with k linearly independent columns, finishing the process, as expected.

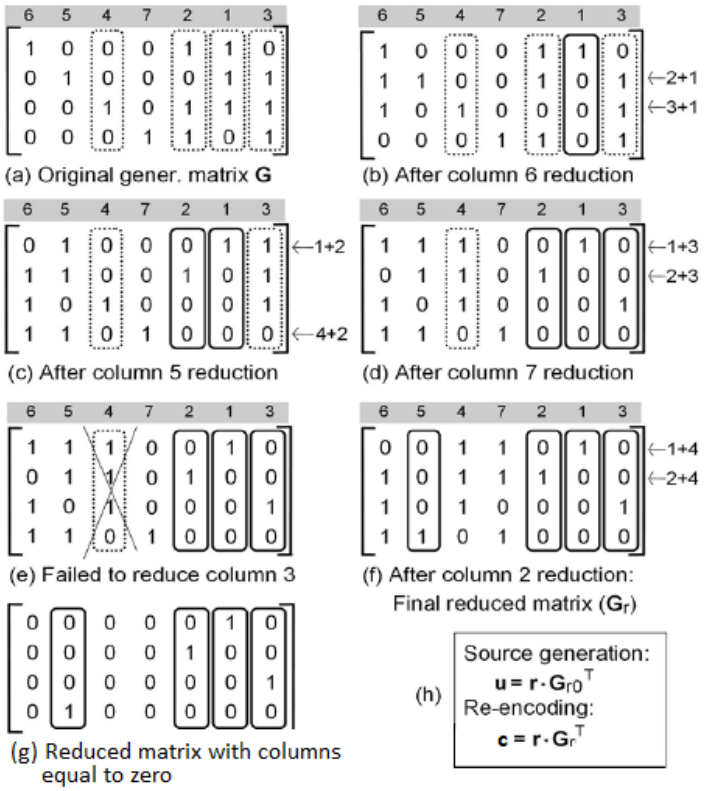

Fig. 4. Block 2 algorithm [12].

The circuit of Block 2 described in VHDL, was simulated in the ModelSim tool. The results of the simulation are shown in Table I. It is possible to see that matrices **Gr** and $\mathbf{Gr_0}$ have the same results in figure 4. It is also possible to see which columns suffered the reduction process (columns 6, 5, 7, 2) and the pivots that were put in the correct positions in $\mathbf{Gr_0}$. In general, the Table I shows the gradual evolution of the k rows of matrices **Gr** and $\mathbf{Gr_0}$ along with the clock signal. Thus, it is

possible to conclude that the circuit works in a proper way. After this conclusion, this code had been implemented in an FPGA, Xilinx Spartan 3E. The results with logic utilization are shown in Table II.

TABLE I. VHDL CODE SIMULATION RESULTS

	1st clock	*2nd clock*	*3rd clock*	*4th clock*	*5th clock*
Gr	1000110	100011	1110010	1110010	0011010
	1100101	1100101	0110100	0110100	1011100
	1010001	1010001	1010001	1010001	1010001
	0001101	1101000	1101000	1101000	1101000
Gr_0	0000010	0000010	0000010	0000010	0000010
	0000000	0000100	0000100	0000100	0000100
	0000000	0000000	0000001	0000001	0000001
	0000000	0000000	0000000	000000	0100000

TABLE II. DEVICE UTILIZATION (SPARTAN 3E)

	Number of used elements	*FPGA utilization*
Flip Flops	71	1%
LUTs	1103	6%
Slices	583	6%

III. OBTAINED RESULTS

After circuit verification, it has been possible to start the design process of the ASIC on the Cadence CAD tools. The first step was the synthesis of the circuit, made in the Cadence RTL Compiler. Here, the VHDL code was synthesized into an RTL circuit, represented in a form of a netlist. This netlist contains the standard cells which will be used in the circuit. Through these standard cells is possible to create the layout of the ASIC in the next steps of the VLSI design flow. The post-synthesis summary is shown in Table III, it presents quantitative data regarding the consumption of logic units, the surface area occupied by the arrangement of IC cells and power consumption. The simulations results were the same as shown in Table I.

TABLE III. POST-SYNTHESIS SUMMARY

Number of standard cells used	502
The area occupied by the standard	5938 µm²
Total power	545 µW

With the circuit working in a proper way, was possible to move forward to the layout generation. This step was made in the Cadence Encounter. Here, the power and I/O buses were positioned in the circuit, as well as the standard cells, defined in the netlist generated in the last step. These elements were routed between them and the characteristics of the resulting layout are shown in Table IV.

TABLE IV. POST LAYOUT SUMMARY

I/O pins	79
Total number of layers	16
Number of tracks for routing	591
The total area of circuit layout	46873 µm²

With the layout ready, it was possible to do the verification and post-layout simulations on the Cadence Virtuoso. First, a transient simulation of the circuit was made, comparing the behavior of the layout with the schematic. The result of that was identical, proving that the layout works properly, enabling to advance to the verification. The DRC (which verifies if the layout is designed in accordance with the technology rules) and the LVS (which verifies if every component the layout is present in the schematic, and vice-verse) were made. With this step concluded, it was possible to advance to parasites extraction and post-layout simulation. In this simulation was possible to confirm that the layout was working properly, with the results to **Gr** and **Gr_0** being the expected. Finally, a pad frame, a necessary structure to make the connections of the design of the IC to the I/O, was designed to accommodate the circuit. The encapsulation chosen was the DIP (dual in-line package) model 40, which has 38 pins which can be used to I/O. Because the number of outputs of **Gr** and **Gr_0** is greater than 38 pins, it was necessary to design a serial-parallel converter, to provide the outputs of both matrices, one row per clock cycle. This circuit was included to Block 2 in some pad frame.

The simulation results are shown in Table V. Analyzing the results given by Table V, it is possible to verify that each row of **Gr** e **Gr_0** is presented sequentially at each clock cycle. With every step of the VLSI design flow simulated and concluded, the final layout of the ASIC was achieved, as shown in figure 5.

TABLE V. DEVICE UTILIZATION (SPARTAN 3E)

	1st clock	*2nd clock*	*3rd clock*	*4th clock*
Gr	0011010	1011100	1010001	0001101
Gr_0	0000010	0000100	0000001	0100000

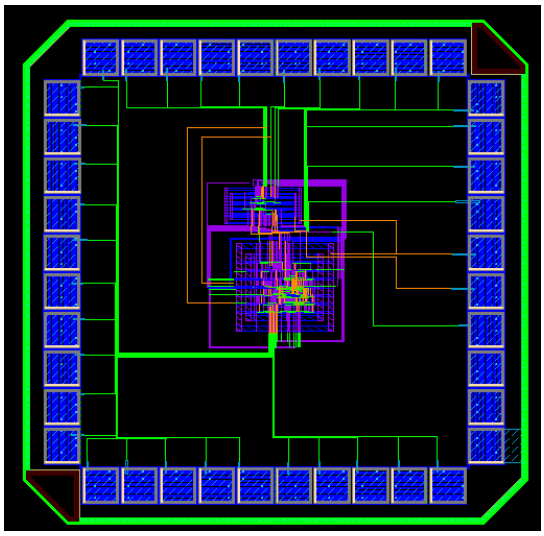

Fig. 5. Final circuit layout .

CONCLUSION

Following the steps described by the VLSI design flow, it was possible, from a VHDL code, to do the whole design process evolved in the design of an ASIC, ready to be manufactured. This digital integrated circuit was simulated in every step of its design process (coding, synthesis and layout), to confirm its functionality. Through these simulations it could prove that it is working properly, is possible to use it in the system which it was originally conceived and in any other system that uses matrix reduction.

REFERENCES

[1] V. A. Pedroni. Digital Electronics and Design with VHDL. Morgan Kaufmann, 2010.

[2] A. Gortan, R. P. Jasinski, W. Godoy and V. A. Pedroni, "Achieving near-MLD performance with soft information-set decoders implemented in FPGAs," 2010 IEEE Asia Pacific Conference on Circuits and Systems, Kuala Lumpur, 2010, pp. 312-315.

[3] E. Prange, "The use of information sets in decoding cyclic codes," IRE. Transactions on lriformation Theory, Vol. IT-8, pp. 5-9, Sep. 1962.

[4] B. G. Dorsch, "A decoding algorithm for binary block codes andj-ary output channels," IEEE Transactions on Information Theory, Vol. IT-20, pp. 391-394, May 1974.'

[5] R. J. McEliece, "A public-key cryptosystem based on algebrix coding theory," Jep Prop. Lab., California Inst, Technol., Pasadena, CA, Tech. Rep., Jan 1978.

[6] G. Clark, J. Cain, Error-Correction Coding For Digital Communication, Plenum Press, 1981.

[7] J. Coffey, R. Goodman, "The complexity of information set decoding," IEEE Transactions on Information Theory, Vol. IT-36, No. 5, pp. 1031-1037, Set. 1990.

[8] M. Fossorier, S. Lin, "Soft-decision decoding of linear block codes based on order statistics," IEEE Transactions on Information Theory, Vol. IT- 4I, No. 5, pp. 13791396, Sep. 1995.

[9] M. Fossorier, S. Lin, J. Snyders, "Reliability-based syndrome decoding of linear block codes," IEEE Transactions on Information Theory, Vol. IT-44, No. I, pp. 388-398, Jan. 1998.

[10] E. Brunvand. Digital VLSI Chip Design with Cadence and Synopsys CAD Tools. 1st ed. Addison Wesley Longman, 2009.

[11] G. G. Brante, D. N. Muniz, W. Godoy Jr"Information Set Based Soft-Decoding Algotithm for Block Codes" IEEE Latin America Transactions, vol. 9, no. 4. P. 463-469 July 2011.

[12] A. Sengupta, "Design Flow of a Digital IC: The role of digital IC\/SOC design in CE products," in IEEE Consumer Electronics Magazine, vol. 5, no. 2, pp. 58-62, April 2016.

[13] R. P.Jasinski, W. Godoy, A. Gortaan, S. B. L. França, V. A. Pedoni, "Efficient Hardware Implementation of Advanced Soft Information-Set Decoders in FPGAs," WSEAS Transactions on Communications, v. 12, p. 334-351, 2013.