Improvement of Flexible Semi-Parallel LDPC Decoders over FPGA for WiMAX Standards

Jutaphet Wetcharungsri[1,2], Narong Buabthong[1], Pakon Thuphairo[3]

Paramin Sangwongngam[2], Keattisak Sripimanwat[2]

[1]*Department of Electrical Engineering, Thammasat University, Pathumthani, Thailand*

[2]*National Electronics and Computer Technology Center (NECTEC), NSTDA, Pathumtani, Thailand*

[3] *Department of Computer Engineering, Rajamangala University of Technology Rattanakosin, Nakhonpathom, Thailand*

E-mail: *jutaphet.wetcharungsri*@nectec.or.th

***Abstract* — Hardware design of LDPC codes is vital to many fields including communications and storage technologies. This paper proposed an improved architecture employing a well-known technique, pipelining, for an implementation of semi-parallel Low-density parity-check codes (LDPC) decoders over field programmable gate array (FPGA). The proposed architecture concerns an issue of path delays in a LDPC network. Essentially, shuffle network and check node unit are combinational circuit that incurs delays. A number of registers are inserted into the design at some places to gain the pipeline behaviour. The semi-parallel architecture which aims at achieving a good trade-off between hardware resources on an FPGA, such as the number of used slice, and bit-error rate performance is studied. From numerical results, it is shown that clock frequency, latency, and FPGA resource utilization are improved.**

Keywords: low-density parity-check (LDPC) codes, latency, semi-parallel LDPC decoders, retiming, Field Programmable Gate Array (FPGA), WiMAX, IEEE 802.16e

I. INTRODUCTION

Low-density parity-check (LDPC) codes have been invented for a long period. It has gained popularity from industrial sectors and academic community since it offers near-capacity limits, flexible choices of coding parameters, and a good trade-off between performance and complexity. In industrial areas, it was chosen in several standards including mobile WiMAX.

In synchronous LDPC decoding architectures, a path delay contributes to a reduction of clock frequencies, thus a throughput. Even though non-flooding updating schemes are employed, multi-standard LDPC IP cores which have become necessary recently [1] incur problematic long path delays.

In synchronous design signals require multiple clock cycles to travel along global wires. Retiming [2–4] is a powerful technique to alleviate the path delay problem.

In [5–6] N. Onizawa *et al* proposed register insertion on long-length paths in presence of outdated message updates and provided evaluation results. Based on their results, the longest wire length is shortened from 4 mm to 2 mm. However, this work does not address a mechanism to solve the outdated coming messages.

In this paper, clock-frequency and latency improvement of LDPC decoders via pipelining is proposed and investigated with modification of its finite state machine (FSM) of the design in order to maintain its bit error rate performance. Retiming is also automatically done by using the synthesis tool with the effort of inserting registers at the appropriate places. Moreover, the implementation of LDPC decoders for practical usage has been done for evaluation of the proposed methods.

This paper is organized as follows; Section II introduces background on LDPC decoding algorithms and LDPC decoder architectures are presented in Section III. The proposed methods are presented in Section IV. Next, the experiment and results are shown in Section V. Finally, Section VI concludes this paper.

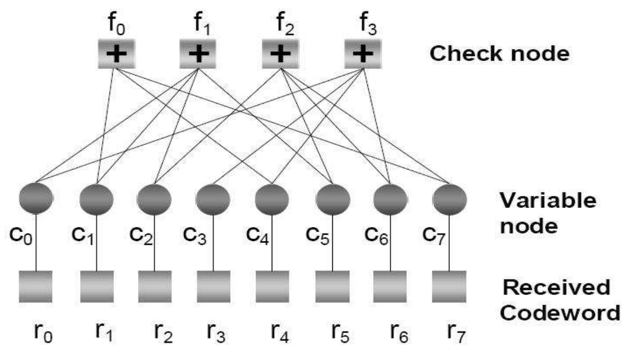
II. LDPC DECODING ALGORITHMS

This section revisits a WiMAX standard on LDPC codes and decoding schemes.

A. Standard of IEEE 802.16e

The IEEE 802.16e specification [7] defines various parity-check matrices H of LDPC codes for several code rates. The algorithm starts with deriving the Tanner graph for parity check matrix H as in Fig. 1. For the size of the parity-check matrix H can be defined as $M \times N$ where M is the number of parity check equations and N is the code word length. The factor Z which is called an expansion factor and denotes a size of the parity-check sub-matrices ranges between 24 and 96. The matrix H is expanded from a base matrix H_b with a size $m \times n$ where $m = M / Z$ and $n = N / Z$. The specified code rates include 1/2, 2/3A, 2/3B, 3/4A, 3/4B, and 5/6. It is noted that H can be partitioned into two portions H_1 and H_2 which are the information nodes and the parity-check nodes. The LDPC parity-check matrix in the IEEE 802.16e is defined as follows.

$$H = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,n} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ P_{m,1} & P_{m,2} & \cdots & P_{m,n} \end{bmatrix}$$



f_0
f_1
f_2
f_3
Check node
Variable node
c_0 c_1 c_2 c_3 c_4 c_5 c_6 c_7
Received Codeword
r_0 r_1 r_2 r_3 r_4 r_5 r_6 r_7

Fig. 1 : Bipartite or Tanner graph representation of an LDPC code mainly consisting of variable nodes, check nodes, and edges

TABLE I INFORMATION AND CODEWORD LENGTHS FOR EACH CODE RATE SPECIFIED BY THE IEEE 802.16E STANDARD [7]

Code length N (bits)	Sub-matrix size, factor Z	Information (bits)			
		rate 1/2	rate 2/3	rate 3/4	rate 5/6
2304	96	1152	1536	1728	1920

where $P_{i,j}$ with a size $Z \times Z$ is either one of a set of circularly right-shifted identity matrices or an all-zero matrix. The LDPC codes that are shown in Table 1 corresponding to the factor $Z = 96$ are adopted for this work.

B. λ-min Algorithm based on Log-likelihood Ratio

Although the log-domain algorithm remarkably reduces resources for hardware implementation, the decoding algorithms can be further modified. Therefore, λ-min algorithm [8] are introduced based on log-likelihood ratio algorithm. The difference is that λ-min algorithm will be a summation with only the 2nd-4th minimum inputs LLR messages to the variable node. After iteration, the computed codeword will be decided whether the decoded bits have been corrected or not. If the decoded bits are not corrected, the repetition of steps 2-4 will be repeated.

III. LDPC DECODER ARCHITECTURES

The parameters of each the decoding algorithms should be selected as the optimum point to retrieve the advantages of using each algorithm when they are combined. Some of parameters require experiments or simulations to reach the best trade-off. As the results, the decoder is selected to use λ=3. The LDPC decoder should be equipped with shift-index ROMs and the shuffle network for the semi-parallel interconnections so that the message exchange between node processors will be pseudo-random, which can increase the decoding performance in aspect of accuracy. A proper decision process can be as the end point of the decoding algorithms that decided if the corrected codeword has been obtained. This process can reduce the computation time due to the reduction of the number of iterations.

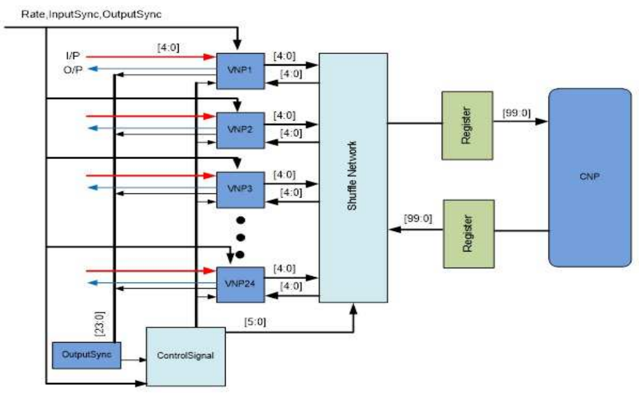


Fig. 2. The proposed method that a number of registers are placed into specific areas in the design.

A. Pipeline Technique

By using the pipeline technique, not only the variable processor can operate the messages at a higher clock frequency but also the entire decoder that is dominated over by a single global clock. In addition, by designing the VNP to work on some operations of the current variable node and the next variable node that shared the same VNP concurrently, the number of the clock cycles per iteration will be reduced.

B. Semi-parallel FPGA-based LDPC Decoder

The design of semi-parallel FPGA-based LDPC decoder is adopted here since it offers a good trade-off between implementation areas and latency.

IV. PROPOSED TECHNIQUES

Insertion of registers in LDPC decoders can be found in [4]. pipeline concept is not new in digital circuit design which employs it to shorten any critical path in a design so as to a higher clock frequency is applicable. Moreover, register retiming is another method to reduce the critical path by relocating registers in the path without affecting the functionality of circuit [4] propose a method for retiming large circuits.

The proposed flexible LDPC decoder architecture can be viewed as a minimal design where only necessary components exist. The main problem is that a shuffle network and check node unit are solely a combinational circuit causing long combinational paths. Especially for implementation over FPGA, designers can utilize available resource effectively since logic cells already contain flip-flops connecting with LUTs in a logic cell. The diagram in Fig. 2 illustrates

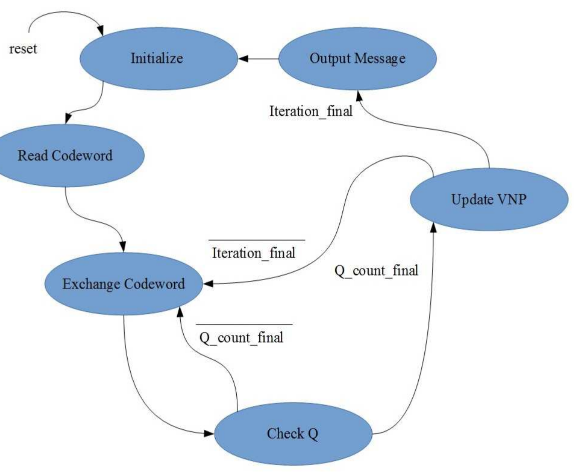


Fig. 3. Finite State Machine of LDPC Decoder

locations where the registers are placed. In Proposed 1, a single register is inserted between output of the shuffle network and input of CNP in Fig. 2. Then, the Proposed 2 further improves the latency by adding the 2nd stage registers. However, it is observed that the longest critical paths exists only inside some VNPs in Fig. 2. Those VNPs are a target that can be addressed. Therefore, in Proposed 3, we also add couple of registers into those VNPs, which contain different VNP Updater than the rest VNPs, resulting in a higher clock frequency. The graph in Fig. 3 illustrates the finite state machine that are internally modified for the propose technique.

V. Evaluation Results

The LDPC design was implemented by VHDL over FPGA chip Xilinx XC5VLX50. From Fig. 4 the BER performance of the implementation was given for various code rates.

The synthesis results for design are provided for conventional and proposed methods in Table II. The 'Proposed 1', 'Proposed 2', and 'Proposed 3' denote single-stage exchange pipeline, double-stage exchange pipeline, and double-stage exchange pipeline with variable node processor (VNP) update pipeline. It can be seen that the proposed methods using more registers to reduce the path delay exploit flip-flop in already used LUTs, resulting in effective utilization of FPGA resources.

Throughput and latency are shown in Table III. All proposed techniques are capable to increase throughput. The throughput of Proposed 1 is improved considerably. The Proposed 2 and 3 can level up throughput remarkably at least 200 %. The latency comparisons among conventional and our proposed methods are illustrated in Fig. 5. While the Proposed 1 can reduce the latency gradually, the Proposed 2 and 3 can decrease it significantly to 50 % of the conventional scheme without modification.

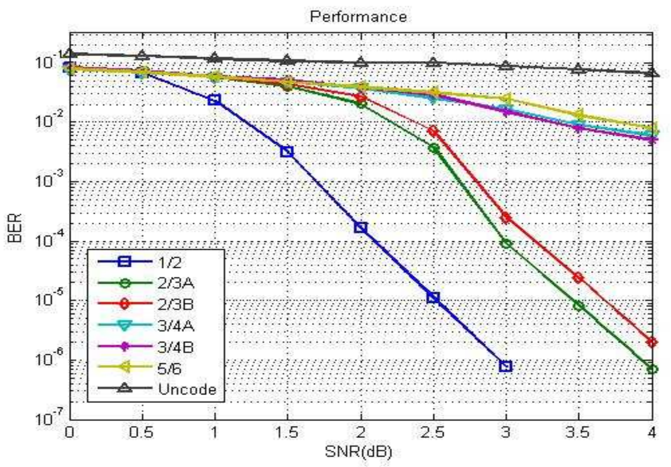


Fig. 4. BER performance of the decoder

VI. Conclusions

In this paper, the technique based on pipeline and retiming has been proposed, described and investigated. We realized the proposed decoder architecture on an FPGA and provided evaluation results for several aspects. From the results, our improved method which meets IEEE 802.16e can improve throughput of the LDPC decoders significantly. In the next step, improvement and analysis of the asynchronous LDPC decoder in the three aspects of digital circuit design, namely area, throughput and power will be studied.

TABLE II. Synthesis results of LDPC decoder for IEEE 802.16e on XC5VLX50 among Conventional and proposed Methods

Slice Logic Utilization	**No pipeline**	**Proposed 1**	**Proposed 2**	**Proposed 3**
Number of Slice Registers	1,233	2,504	2,908	3,185
Number of Slice LUTs	10,674	10,534	10,938	11,522
Number of fully used LUT Flip Flop pairs	1,017	1,588	2,685	2,874
Number of bonded IOBs	148	148	148	148
Number of Block RAM/FIFO	13	13	13	13

Acknowledgment

The authors would like to express gratitude towards the Electronics and Devices System (EDS) program, NECTEC, Thailand for use the facility and Faculty of Engineering, Thammasart University for their financial supports. Moreover, the authors would like to thank Mr. Sakdinan Jantarachote for his valuable discussion and comments.

Table III. Throughput and latency

Code Rate	Initial Flexible		Proposed 1 Single-Stage Exchange Pipelining		Proposed 2 Double-Stage Exchange Pipelining		Proposed 3 Double-Stage Exchange& VNP Update Pipelining	
	Throughput (Mbps)	Latency (ms)	Throughput (Mbps)	Latency (ms)	Throughput (Mbps)	Latency (ms)	Throughput (Mbps)	Latency (ms)
1/2	9.60	0.240	12.77	0.180	17.91	0.129	21.17	0.109
2/3A	14.03	0.164	18.99	0.121	26.65	0.086	31.50	0.073
2/3B	14.03	0.164	18.99	0.121	26.65	0.086	31.50	0.073
3/4A	18.89	0.122	25.12	0.092	35.25	0.065	41.67	0.055
3/4B	18.89	0.122	25.12	0.092	35.25	0.065	41.67	0.055
5/6	27.91	0.083	37.09	0.062	52.04	0.044	61.53	0.037

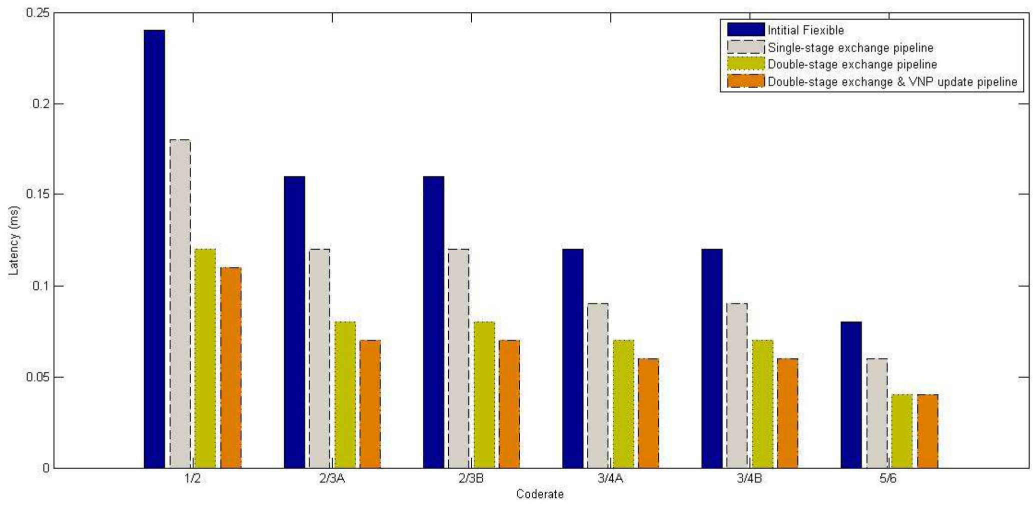


Fig. 5. Comparisons on latency among conventional and proposed methods

References

[1] T. Mohsenin and B. Baas, “Trends and Challenges in LDPC Hardware Decoders,” Forty-Third Asilomar Conf. on Signals, Systems and Computers, pp.1273 - 1277, 2009.

[2] C. Leiserson and B. Saxe, “Optimizing synchronous systems,” J. VLSI Comput. Syst., vol. 1, no. 1, pp. 41–67, 1983.

[3] C. Leiserson and B. Saxe, “Retiming synchronous circuitry,” Algorithmica, vol. 6, no. 1, pp. 5–35, 1991.

[4] N. Shenoy and R. Rudell, “Efficient implementation of retiming,” in Proc. IEEE/ACM Int. Conf. Computer-Aided Design, 1994, pp. 226–233.

[5] N. Onizawa, T. Ikeda, T. Hanyu, and V. Gaudet, “3.2-Gb/s 1024-b rate-1/2 LDPC decoder chip using a flooding-type update-schedule algorithm,” Circuits and Systems, 2007. MWSCAS 2007. 50th Midwest Symposium on , vol., no., pp.217,220, 5-8 Aug. 2007.

[6] N. Onizawa, T. Hanyu, and V. C. Gaudet, ``Design of High-Throughput Fully-Parallel LDPC Decoders Based on Wire Partitioning," IEEE Trans. on Very Large Scale Integration (VLSI) Systems, vol. 18, no. 3, pp. 482-489, Mar. 2010.

[7] Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment for Physical and Medium Access Control Layersfor Combined Fixed and Mobile Operation in Licensed Bands, IEEE P802.16e-2005, 2005.

[8] Guilloud, F., Boutillon, E., and Danger, J. L., “λ-Min Decoding Algorithm of Regular and Irregular LDPC Codes,” *Proceedings of the 3nd Interl. Symposium on Turbo Codes & Related Topics*, pp. 451–454, Brest, France, Sep. 2003.