Research Article

Design of Multi-Mode Semi-Parallel LDPC Decoders for WiMAX Standards

Jutaphet Wetcharungsri* and Narong Buabthong
Department of Electrical Engineering, Faculty of Engineering, Thammasat University, Rangsit Campus, Khlong Nueng, Khlong Luang, Pathum Thani, 12120 Thailand

Pakon Thuphairo
Department of Computer Engineering, Faculty of Engineering, Rajamangala University of Technology Rattanakosin, Nakhonpathom, 73170 Thailand

Paramin Sangwongngam and Keattisak Sripimanwat
NECTEC, National Science and Technology Development Agency (NSTDA), Khlong Nueng, Khlong Luang, Pathumthani 12120, Thailand

Abstract

Demands for highly efficient Low-density parity check (LDPC) encoders and decoders have significantly risen in a wide range of applications, including deep-space communications, satellite modems, mobile communications, and storage. In this article, a pipelining technique was applied to an implementation of synchronous semi-parallel LDPC decoders over FPGA in order to reduce the intrinsic latency due to the fact that a shuffle network, check node, and some modules in VNP units are combinational circuits. Our technique proposed that a certain number of registers can be intentionally placed at specific stages in the conventional design, thus realizing pipeline structures. Three methods, named 'Scheme 1', 'Scheme 2', and 'Scheme 3', were proposed. The first method employs a single-stage exchange pipeline while the second one makes use a double-stage exchange pipeline. The third technique improves upon the second one by creating a variable node processor (VNP) update pipeline. Our proposed architecture focuses on not only an aspect of path delays in a large LDPC network but also the optimization of FPGA resources, e.g., the number of used slices. To test the performance of the proposed schemes, the simulations of the conventional design and each proposal were performed. The numerical results indicate that improvements in important parameters, *i.e.,* clock frequency, latency, and FPGA resource utilization, can be achieved by of the methods. All of the propose techniques are capable of raising throughput and reducing latency of the design dramatically. In Scheme 3, the throughput is increased by approximately 84 percent that of the conventional technique. Furthermore, it is noteworthy that the proposed methods with additional registers have exploited unused flip-flops in the LUTs that have been already occupied, hence achieving more effective utilization of precious FPGA resources, especially when commercial aspects are considered.

Keywords: Low-Density Parity Check (LDPC) codes; latency; semi-parallel LDPC decoders; retiming; Field Programmable Gate Array (FPGA); WiMAX; IEEE 802.16e.

***Correspondence** : jutaphet.wetcharungsri@nectec.or.th

1. Introduction

Channel coding is an essential part of reliable communication links. Low density parity check (LDPC) codes and their derivatives are considered as eligible channel codes for future generation high data rate communications for various practical applications because of their superior performance in offering near-capacity limits, flexible choices of coding parameters, and a good trade-off between performance and complexity as compared to other channel coding schemes. Various communication standards in a wide spectrum ranging from wired communications, namely 802.3an, G.hn/G.9960, DVB-C2; wireless communications, that is Mobile WiMAX, IEEE 802.11n; and broadcasting, such as DVB-T2/S2, DMB-T/H, have adopted LDPC codes as a channel coding scheme for error correction in their standards. Furthermore, LDPC is application not only in the area of communication technology but hard disk storage technology as well. Partial preliminary results of this work appeared in [15].

Even though LDPC codes were invented several decades before the successful emerging technology of turbo codes implemented in various standards, they were struggling with the intrinsic characteristics of the random and high degree of interconnection among computation nodes in their circuits, resulting in difficulties in the circuit design. After the revival of LDPC in 1991, implementation aspects have been unabatedly focused on, and significant results have been reported. In wireless communications, development of the LDPC design mainly aims at supporting standards, reducing implementation areas requiring low power consumption. Other design goals include mitigation of error floors [1].

A design of LDPC decoders was first proposed in an international conference in English language in 2001 by C. Howland and A. Blanksby [2]. They proposed fully-parallel decoding architectures that achieve high throughput but consume large areas and incur high interconnections. This could achieve 1 Gbps and 2 Mbps at 64 MHz and 128 KHz clock frequency, respectively. In contrast, in 2004, M. Cocco *et al*. [3] proposed serial architectures. In 2006, Z. Cui and Z. Wang [4] proposed semi-parallel architectures that provide trade-offs in combining the two aforementioned architectures.

K. K. Gunnam *et al*. [5] proposed that the value–reuse property in the offset min-sum (OMS) decoding algorithm was applied and that turbo decoding message passing (TDMP) was employed in irregular LDPC for the WiMAX IEEE 802.16e standard. The proposed method is classified as horizontal layered decoding, which is a sub-class of a layered decoding algorithm. Compared with belief propagation, this technique can reduce the number of memory used by 55 %, decrease the number of interconnections required by 50%, and increase the throughput by 100% (*i.e.* two times). However, the computational complexity and the power consumption of the proposed technique are still considerably high. In 2008 Xin-Yu Shih *et al* [6] proposed a reordering of the base matrix and overlapped operation of main computational units of CNP and VNP for WiMAX. While the overlapped operations technique reduces decoding latency by 68.75% and increases hardware utilization ratio, it causes a critical path delay in the synchronous circuit of the LDPC decoder.

During the design step of the LDPC decoder, there are a number of factors which must be taken into account such as area, speed, energy dissipation per bit, latency, path delay, error floor and error performance gap from Shannon limit [7]. Generally, in synchronous design, the global clock signal is supposed to travel along the global wires toward several synchronous components. Focusing on synchronous LDPC decoding architectures, path delay affects the overall throughput of the system. The path delay in synchronous LDPC decoding architectures will slow down the clock frequency, thus

affecting the throughput. Even though non-flooding updating schemes are employed, multi-standard LDPC IP cores, which have recently become necessary [7], incur problematic long path delays. There are a number of ways in which the path delay issue can be overcome. One powerful technique is retiming [8–10] in which registers are added and/or removed from the circuit at appropriate locations in such a way that the functional behaviour of the system is not disturbed but the overall performance is enhanced [9].

In [11–12], register insertion on long-length paths in the presence of outdated message updates was proposed with evaluation results. It is evident that the longest wire length is shortened from 4 mm to 2 mm. However, this work does not address how to solve the outdated incoming messages.

In LDPC codes, decoding is performed iteratively. The steps involved are the variable node output initialization, updating check node (horizontal update), variable node update (vertical update) and finally the decision making through iterative decoding. In the above - mentioned conventional algorithm, the node computations are performed by using all of the updated messages including those of the longer nets which impose limits on increasing clock frequency. A flooding-type update-schedule algorithm [11] overcomes this issue by inserting a register in a long length path. Advantages from this technique are the shortening of the wire length by half; however, this algorithm does not address the outdated incoming messages. Another method to increase the clock speed by reducing the path delay is proposed in [12]. In this technique, longest wires of the decoder are divided onto a number of short wires with the help of pipeline register. Messages sent along the pipelined paths are over the multiple clock cycles, thus reducing the critical path delay without compromising the bit error rate performance.

In this paper, improvements regarding two important issues in LDPC decoder design, *i.e.*, a clock frequency and latency, are discussed. A pipelining scheme is proposed and investigated with a modification of its controller in the design in order to maintain its performance in terms of bit error rate (BER). Retiming is taken into account by the use of a synthesis tool for the purpose of adding registers at appropriate locations. Moreover, the proposed algorithms have also been evaluated in terms of BER and hardware utilization.

The remainder of the paper is organized as follows. Section 2 introduces the IEEE 802.16e Standard and LDPC decoding algorithm, particularly λ-min algorithm, and the hardware architectures of the decoder together with functional designs are presented in Section 3. The proposed methods are presented in Section 4. Next, the experiment and the results are shown and discussed in Section 5. Finally, Section 6 concludes this article.

2. Preliminaries

This section provides necessary background information, namely, the IEEE 802.16e, standard and an LDPC decoding algorithm.

2.1 Review of the IEEE 802.16e Standard

This sub-section highlights the IEEE 802.16e standard and decoding algorithm for LDPC. Various parity check matrices H of LDPC are discussed in IEEE 802.16e standard specification [13] covering numerous code rates. LDPC codes, a class of linear block codes, are defined by a parity check matrix H, which is an $M \times N$ binary sparse matrix where M is the number of parity check equations and N is the code word length. For IEEE 802.16e standard, the parity-check matrices H are irregular and are expanded from a base matrix H_b with the size $m \times n$ where $m = M / Z$ and $n = N / Z$. The

factor Z is called an expansion factor and denotes a size of the parity-check sub-matrices ranging between 24 and 96. The code rate is $R = 1 - \frac{M}{N}$, assuming that the M rows of the H matrix are linearly independent. In an irregular LDPC matrix, not all columns and rows have the same number of one's as a constant value. 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 variable nodes and the check nodes, respectively. The LDPC parity-check matrix in the IEEE 802.16e standard is defined as follows.

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

where $P_{i,j}$ of 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 provided in Table 1 with the factor $Z = 96$ are particularly opted for this work.

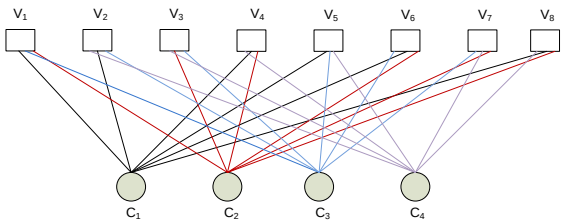


Fig.1. Bipartite or Tanner graph representation of an LDPC code mainly consisting of variable nodes, check nodes, and edges**.**

Table1. Information and codeword lengths for each code rate as specified by the IEEE 802.16e standard [13]**.**

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

2.2 λ-min Algorithm based on Log-Likelihood Ratio

The λ-min algorithm of LDPC decoding [14] is a modified version of belief propagation. It begins with a derivation of the Tanner graph illustrated in Fig. 1 for parity check matrix (H-matrix) based on IEEE 802.16e where the nodes of the graph are described as two sets of WiMAX H-matrix, variable node (V_n) and check node (C_m). An edge connects a check node to a variable node if and only if H_{mn} is non-zero. The decoding iteration finishes when either a decoded codeword satisfies all parity-check equations or the maximum iteration number is reached. The log-likelihood ratio algorithm can be first introduced here to describe a decoding scheme, then the λ-min algorithm will be further demonstrated. The log-likelihood ratio (LLR) of a binary case with two possible outcomes, namely, $u_i = 0$ and $u_i = 1$, can be computed as follows:

$$L(u_i) = \log \frac{P(u_i = 0 \mid y_i)}{P(u_i = 1 \mid y_i)} = \log \frac{P(y_i \mid u_i = 0)P(u_i = 0)}{P(y_i \mid u_i = 1)P(u_i = 1)} \quad (1)$$

where $P(u_i = \mathrm{a})$ denotes the probability that u_i takes on the value $a \in \{0,1\}$.

Similarly, $\lambda_{n \to m}(u_i) = \log \frac{P(q_{n \to m}(0))}{P(q_{n \to m}(1))}$

Where $q_{n\to m}(0)$ and $q_{n\to m}(1)$ denote the messages from a bit node n to a check node m in case that the probability of the bit u_i is 0 and 1, respectively, calculated with all check nodes connected except only a check node m.

Also, $\Lambda_{m\to n}(u_i) = \log \frac{P(\mathrm{r}_{m\to n}(0))}{P(r_{m\to n}(1))}$

where $r_{m\to n}(0)$ and $r_{m\to n}(1)$ denote the messages from a check node m to a bit node n in case where the probability of the bit u_i is 0 and 1, respectively, based on all check nodes connected except only a check node m. The decoding algorithm will proceed in four steps: initializing, updating check nodes, updating variable nodes, and checking termination criteria as follows.

2.2.1 Initialization

In the initialization step, the LLR messages are initialized as follows:

$$\lambda_{n\to m}(u_i) = L_{ch}(u_i), \text{ for } n \in N(\mathrm{m}),\ m \in M(\mathrm{n}) \quad (2)$$

and $$\Lambda_{m\to n}(u_i) = 0, \text{ for } n \in N(\mathrm{m}),\ m \in M(\mathrm{n}) \quad (3)$$

where $N(m)$ and $M(n)$ denote the set of variable nodes that is connected to a check node m and denote the set of check nodes that is connected to a variable node n, respectively.

2.2.2 Check node update

The incoming LLR messages $\Lambda_{m\to n}(u_i)$ from a variable node n in the initialization step update a check node m, which can be expressed as

$$\Lambda_{m\to n}(u_i) = \mathrm{sign}\left(\prod_{n'\in N(\mathrm{m})\backslash \mathrm{n}} \lambda_{n'\to m}\left(\frac{u_i}{2}\right)\right) \cdot \Phi^{-1}\left(\sum_{n'\in N(\mathrm{m})\backslash n} \Phi\left(\lambda_{n'\to m}\left(\frac{u_{n'}}{2}\right)\right)\right) \quad (4)$$

where $\Phi(\mathrm{x}) = \log\left|\tanh\left(\frac{x}{2}\right)\right| = \log\left(\frac{e^x - 1}{e^x + 1}\right)$. (5)

Although the log-domain algorithm remarkably reduces the resource requirement of hardware implementation, the decoding algorithms can be further modified. Therefore, the λ-min algorithm is introduced based on the log-likelihood ratio algorithm. The difference is that the λ-min algorithm calculates only the specified λ lowest magnitude of the extrinsic information $\lambda_{n\to m}$ in Eq. (4). The parameter λ ranges from two to the total number of variable nodes attached to the check node m.

2.2.3 Variable node update

The LLR for each variable node can be updated via the following expression:

$$\lambda_{n\to m}(u_i) = L_{ch}(u_i) + \prod_{m'\in M(n)\backslash m} \Lambda_{m'\to n}(u_i). \quad (6)$$

2.2.4 Stop criteria

Overall, the posteriori LLR probability of the bit node n can be obtained by the summation of all inputs LLR messages to the variable node n as follows:

$$L_{post}(u_i) = L_{ch}(u_i) + \prod_{m'\in M(n)} \Lambda_{m'\to n}(u_i). \quad (7)$$

At the end of each iteration, the variable node *n* will be decoded, and vice versa. After the iteration, the codeword will be decided as "1" or "0" based on the posteriori LLR probability of the bit node n. Then, the stop criteria are checked to determine whether the decoding meets the criteria that are set. The decoding completes if either the pre-set iteration number is reached or if the codeword

satisfies the check equations. Otherwise, Step 2–4 are repeated.

3. Hardware architectures of LDPC decoder

In this section, a hardware-based architecture of the semi-parallel FPGA-based LDPC decoder is illustrated in Fig. 2 The semi-parallel FPGA-based LDPC decoder comprises several VNPs, one CNP, a shuffle network, and one controller. The parameters of each of the decoding algorithms should be optimally selected as to obtain the advantages due to each algorithm being combined. Some of the parameters require experiments or simulations to determine the best trade-off. According to our simulation, the decoder is set 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, a property that can increase the decoding performance in terms 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 decrease in the number of iterations. This design of semi-parallel FPGA-based LDPC decoder is adopted here since it offers a good trade-off between implementation areas and latency. To realize the hardware circuit of the main components, the CNP and VNP, a set of equations in the decoding algorithm explained in Section 2 is used for its implementation in digital logic circuits. From Equations (4)-(6), the CNP and VNP can be implemented as illustrated in Fig. 3 and Fig. 4.

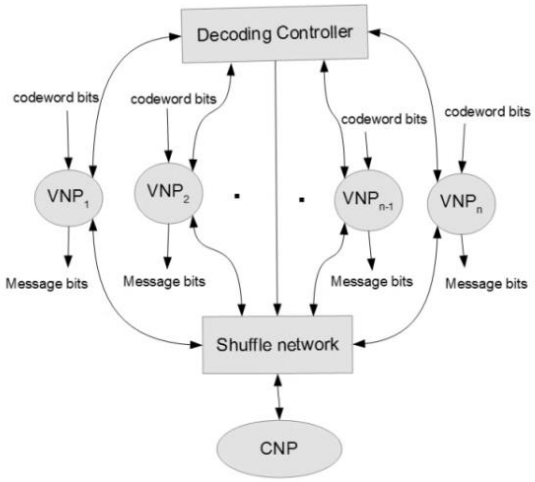


Fig.2. The architecture of flexible LDPC decoder**.**

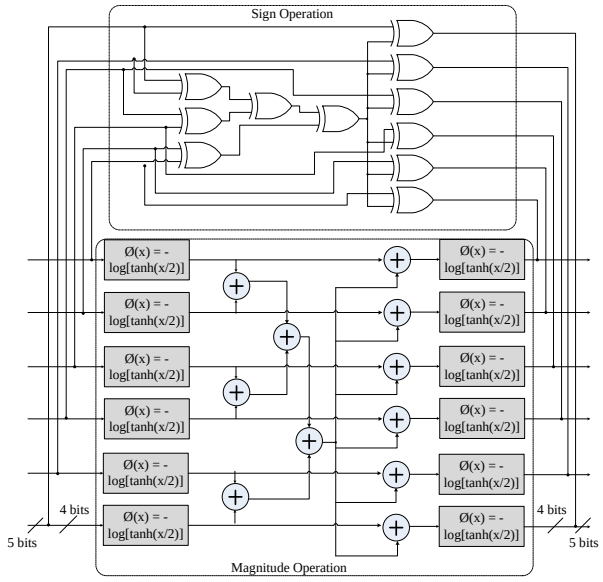


Fig.3. The architecture of CNP**.**

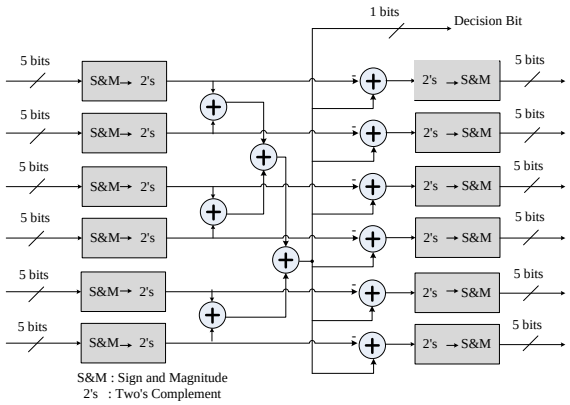


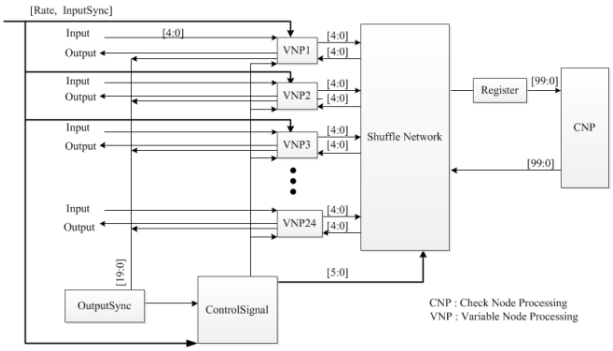
Fig.4. The architecture of VNP**.**

4. Proposed Techniques

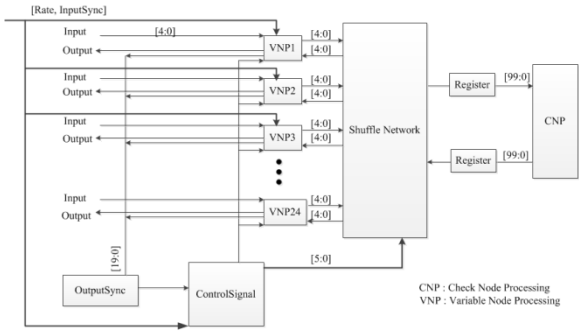
In general, the technique employing additional register insertion in LDPC decoders is described in [4]. The pipeline concept is a promising technique that researchers have been using for many years back for digital logic design. Through pipelining techniques, the length of the critical path can be reduced thus ensuring a higher clock frequency. Furthermore, another approach to reduce the critical path is retiming. This technique is based on relocation of the registers that are responsible in a critical path whereas functionality remains unchanged [4].

The exact locations for inserting registers are determined by using information obtained from the synthesis report. In this work, for each proposed technique, a stage of register is inserted into the critical path. Next, the entire circuit is re-synthesised. Therefore, the process of register insertion is done in a manually iterative style. Nevertheless, it should be noted that the delay reports are estimated at the synthesis level. The anticipated results may not be achieved after place-and-route process, which will be seen more clearly in Section 5.

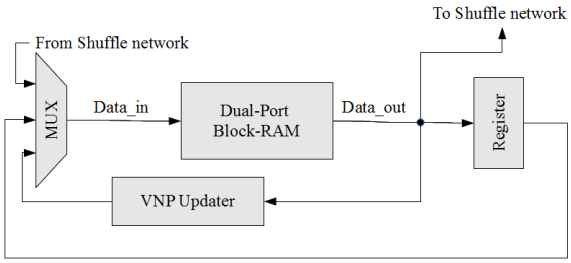
The previously proposed flexible LDPC decoder architecture is a minimal design where only necessary components are presented. One drawback of the previously proposed architecture is that the shuffle network and the check node unit are made up only of combinational circuits, resulting in prominently long combinational paths. Incorporating this approach with FPGA implementation will result in a poor utilization of resources since the succeeding connected flip flop is unoccupied. Fig. 5 shows the location of the registers. In the first proposed scheme, a register is inserted between the output of the shuffle network and the input of CNP. In the second proposed scheme, which is more efficient than the first, registers are inserted in the 2nd stage. After this improvement, it is observed according to the synthesis report that the critical paths exist only inside some VNPs. Improving the situation inside VNPs can lead to good results on the synthesis level. Therefore, for the third proposed scheme, in VNP22, VNP23, and VNP24, the VNP updater is not the same as those in the rest of VNPs. The VNP Updater input signal is routed from the 2nd pipeline register which will be automatically retimed with Xilinx XST, a synthesis software. This will result in a higher clock frequency reported at the synthesis level. Fig. 7 shows the finite state machine that are internally modified for all proposed schemes.



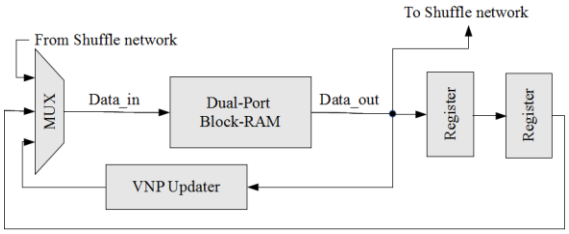
(a) Overall architecture of Scheme 1: a stage of register is added between shuffle network and CNP



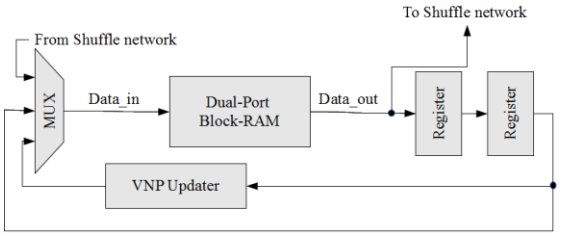
(b) Overall architecture of Scheme 2 and Scheme 3: a couple stage of registers is added between shuffle network and CNP



(c) Register insertion in all VNPs of Scheme 1



(d) Register insertion in all VNPs of Scheme 2



(e) Register insertion in VNP22 - VNP24 of Scheme 3. The rest of VNPs are the same as those in Scheme 2.

Fig.5. The proposed methods that a number of registers are placed into specific areas in the design.

In synchronous digital circuit design, pipeline insertion may not impact only the modification of data path but also how the receiving modules capture the output of a pipelined circuit with the right timing. The pipeline behaviour may not be easily implemented in some specific architectures, for instance, a well-known sequential problem, namely a pipeline stall, can arise in a general purpose processor. In this work, Dual-Port Block-RAMs are used in VNPs so read and write operations can be performed simultaneously. Moreover, the read-write addresses change sequentially in the processes of exchange and VNP Update. Hence, the controller can be easily modified to create a pipeline behaviour with a trivial penalty of clock cycles for which the RAMs have to wait when compared to the non-pipelined version. The operations are illustrated in Fig. 6.

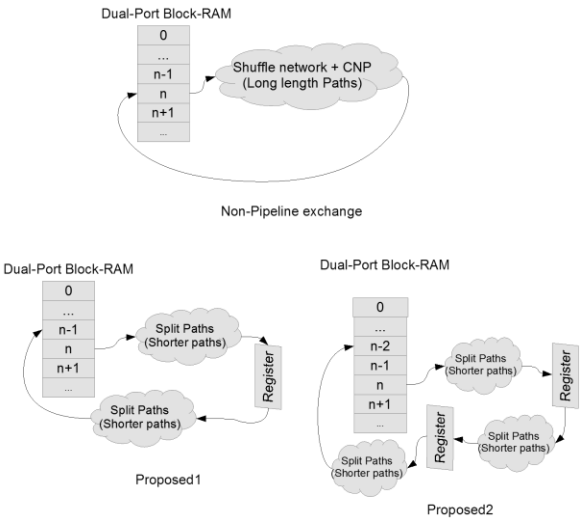


Fig.6. Conceptual illustration of pipelining and retiming**.**

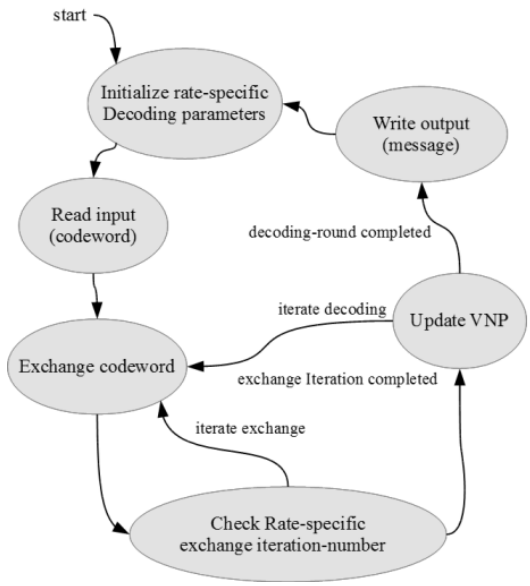


Fig.7. Finite State Machine of LDPC Decoder.

In VNP22, VNP23, and VNP 24 of Scheme 3, the VNP Update input is the output of the second register stage that is already used for timing the RAM output fed back to the RAM itself in Scheme 2. The tool will take charge of retiming the registers to obtain a better clock frequency. However, the clock frequency operated at P&R may differ in the opposite direction due to the fact that the synthesized circuit may contain a high routing complexity, thus leading to an unexpected clock speed that is lower than one reported at the synthesis process. Fig. 7 illustrates the modified finite state machine for the proposed techniques in order to change the timing of some control signals that handle message exchanges between VNPs and CNP through the shuffle network.

5. Results and discussion

The performance of the synthesized circuit varies dramatically depending on the options chosen through the tool and even the particular versions of the tool. The best possible way to insert a pipeline is beyond the scope of this work, it has to be more investigated to achieve higher performance by a variety of related issues.

5.1 The synthesis and place-and-route results

The synthesis and place-and-route results for our designs are provided in Table 3. 'Scheme 1', 'Scheme 2', and 'Scheme 3' denote a design with a single-stage exchange pipeline, a double-stage exchange pipeline, and a double-stage exchange pipeline with variable node processors (VNPs) update pipeline. It can be seen that the proposed methods use more registers to reduce the path delay and exploit flip-flops in already used LUTs, resulting in effective utilization of FPGA resources. Explicit options, constraints and other parameters set in the software tool, Xilinx ISE Design Suite 14.7, are provided in Table 2 for reference purposes.

5.2 Throughput and latency

Throughput and latency are shown in Table 4. All of the proposed techniques are capable to increase throughput from that of the non-pipeline design. Scheme 3 achieves the highest throughput.

The performance of relevant decoders that were implemented on FPGAs is shown in Table 5. Since each decoder has been proposed with different architectures, purposes and implementations, their overall performance cannot be concluded merely based on a single parameter. However, as seen in the Table 5, our architecture occupies reasonable amounts of FPGA resources, especially, the smaller requirement on block RAMs and the clock frequency that stays in the range among those of the previously proposed decoders.

Table2. Altered options, constraints and other sensitive parameters.

Options, constraints		**Altered as**
Synthesis	Optimization Goal	Speed
	Optimization Effort	High
	Register Balancing	Yes
Implementation	Placer Effort Level	High
	Placer Extra Effort	Normal
	Global Optimization	Speed
	Optimization Strategy	Speed
	Place & Route Extra Effort	Normal
	Enable Multi-Threading	2
Other parameters		**Usages**
UCF File		Not used, no clock constraints
Clock frequency obtained at P&R level		Best case Achievable on P&R report

Table3. Synthesis results of LDPC decoder for IEEE 802.16e on XC5VLX50 among conventional and proposed methods**.**

Slice Logic Utilization	**non-pipeline**	**Scheme 1**	**Scheme 2**	**Scheme 3**
Number of Slice Registers	1,357	1,851	2,379	2,742
Number of Slice LUTs	9,531	8,557	8,770	9,381
Number of fully used LUT Flip Flop pairs	1,227	1,736	2,161	2,442
Number of LUT Flip Flop pairs used	9,661	8,672	9,006	9,681
Fully LUT FFs/LUT FFs ratio	0.127	0.200	0.240	0.252
Number of bonded IOBs	148	148	148	148
Number of Block RAM/FIFO	14	15	14	17

Table 4. Throughput and latency**.**

Code Rate	**Non-pipeline**		**Scheme 1 Single-Stage Exchange Pipelining**	
	Throughput (Mbps)	**Latency (ms)**	**Throughput (Mbps)**	**Latency (ms)**
1/2	4.59	0.50	7.27	0.32
2/3A	6.83	0.34	10.81	0.21
2/3B	6.83	0.34	10.81	0.21
3/4A	9.03	0.26	14.30	0.16
3/4B	9.03	0.26	14.30	0.16
5/6	13.33	0.17	21.11	0.11
Code Rate	**Scheme 2 Double-Stage Exchange Pipelining**		**Scheme 3 Double-Stage Exchange& VNP Update Pipelining**	
	Throughput (Mbps)	**Latency (ms)**	**Throughput (Mbps)**	**Latency (ms)**
1/2	8.23	0.28	8.46	0.27
2/3A	12.24	0.19	12.59	0.18
2/3B	12.24	0.19	12.59	0.18
3/4A	16.19	0.14	16.65	0.14
3/4B	16.19	0.14	16.65	0.14
5/6	23.90	0.10	24.59	0.09

Table5. Comparison of FPGA-based WiMAX LDPC Decoders.

LDPC Decoders	**Implementation Technology**	**Clock frequency (MHz)**	**LUTs (10^3)**	**Flip flops (10^3)**	**Block RAMs**
Scheme 3	Virtex-5 (xc5vlx50-1ff676)	111.66	9.38	2.44	17
[16] (Processing module 4)	Virtex-5 (110LXT)	192	19	10	92
[17]	Virtex-2 (XC2V8000)	61.3	4.377	1.734	70
[5]	Virtex-2 (XC2V8000ff152-5)	110	2.982, 5.664, 11.028	1.582, 3.165, 6.33	38, 73, 100
[18]	Stratix II (EP2SI80FI020C3)	73	-	15.594	-

5.3 Power consumption

The entire circuit of each decoder was tested for power consumption using Xilinx Power Estimator (XPE). The report indicated that the various proposed designs consume comparable amounts of power as shown in Table 6.

Table6. Power estimation of the proposed decoders**.**

Proposed design	Estimated power consumption (watt)
Non-pipelined	0.430
Scheme 1	0.431
Scheme 2	0.434
Scheme 3	0.434

The increase in the power consumption of each propose is proportional to the required amount of additional FPGA resources, for instance, the number of flip-flops occupied for pipelining. Nevertheless, more accurate power estimation can be obtained when a level of switching activity of the decoder circuit is incorporated.

5.4 BER performance

The LDPC design was implemented using VHDL over Xilinx XC5VLX50-

1FF676 FPGA chip, as shown in Fig. 8, the BER performance of the implementation was illustrated for various code rates.

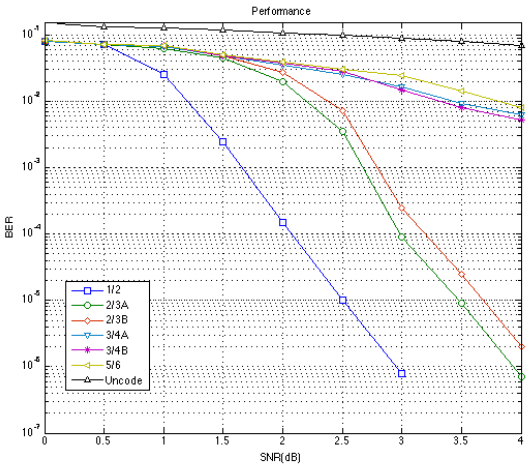


Fig.8. BER performance of the decoder.

6. Conclusion

In this work, our propose techniques, ‘Scheme 1’, ‘Scheme 2’, and ‘Scheme 3’, which are based on an idea of adding a pipeline followed by retiming, have been proposed in order to lower the latency caused by the shuffle network and the check node units. This work focuses an implementation of synchronous semi-parallel LDPC decoders, according to the IEEE 802.16e standard over FPGA. The λ-min algorithm based on log-likelihood Ratio is adopted for the study. The proposed decoder architectures on an FPGA and the evaluation results were provided from several aspects, namely, synthesized and place-and-route results, throughput, latency, power consumption, and BER. From the results, our methods can, significantly, improve the throughput of the LDPC decoders and the utilization of the FPGA logic blocks. Scheme 1, Scheme 2 and Scheme 3 exhibited their potential to increase throughput and decrease latency of the design considerably, namely by eighty percent of those of the conventional technique. In addition, the proposed methods utilize

unused flip-flops in the LUTs to create the pipeline processing, and also use the hard-core Dual-Port Block-RAM to facilitate the pipelined read-write operations. Therefore, these methods improve the utilization of valuable FPGA resources, especially when commercial perspectives are considered.

7. Acknowledgements

The authors would like to express gratitude to NECTEC, National Science and Technology Development Agency (NSTDA), for the financial support. The authors would like to thank Sakdinand Jantarachote and Tharathorn Promsa-ard for invaluable discussions.

8. References

[1] Xiaoheng Chen; Jingyu Kang; Shu Lin; Akella, V. "Hardware Implementation of a Backtracking-Based Reconfigurable Decoder for Lowering the Error Floor of Quasi- Cyclic LDPC Codes", *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 2931–2943. Vol. 58, Issue 12, Dec. 2011.

[2] Howland, C.; Blanksby, A., "Parallel decoding architectures for low density parity check codes," in *Proc. of the 2001 IEEE International Symposium on Circuits and Systems (ISCAS 2001)*, vol.4, Sydney, Australia, pp.742–745 vol. 4, 6–9 May 2001.

[3] M. Cocco, J. Dielissen, M. Heijligers, A. Hekstra, and J. Huisken, "A scalable architecture for LDPC decoding," in *Proc. of the IEEE Conf. Design Automation and Test in Europe (DATE)*, Feb. 2004, vol. 3, pp. 88–93.

[4] Z. Cui and Z. Wang, "Area-efficient parallel decoder architecture for high rate QC-LDPC codes," *in Proc. of the IEEE ISCAS, May 2006*, pp. 5107–5110.

[5] K. Gunnam, G. Choi, M. Yeary, and M. Atiquzzaman, "VLSI architectures for layered decoding for irregular LDPC codes of WiMAX," in *Proc. of the IEEE International. Conference on Communications (ICC'07)*, pp. 4542–4547, Jun. 2007.

[6] Xin-Yu Shih; Cheng-Zhou Zhan; Lin, Cheng-Hung; An-Yeu Wu, "An 8.29 mm2 52 mW Multi-Mode LDPC Decoder Design for Mobile WiMAX System in 0.13 μm CMOS Process," *IEEE Journal of Solid-State Circuits*, vol.43, no.3, pp.672–683, Mar. 2008.

[7] T. Mohsenin and B. Baas, "Trends and Challenges in LDPC Hardware Decoders," in *Proc. of the Forty-Third Asilomar Conf. on Signals, Systems and Computers*, pp.1273–1277, 2009.

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

[9] C. Leiserson and B. Saxe, "Retiming synchronous circuitry," *Algorithmica*, vol. 6, no. 1, pp. 5–35, 1991.

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

[11] 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," in *Proc. of the 50th Midwest Symposium on Circuits and Systems, MWSCAS 2007*, vol., no., pp.217–220, 5–8 Aug. 2007.

[12] 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.

[13] Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment for Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation

in Licensed Bands, IEEE P802.16e-2005, 2005.

[14] F. Guilloud, E. Boutillon and J. L. Danger, "λ-Min Decoding Algorithm of Regular and Irregular LDPC Codes," in *Proc. of the 3rd International Symposium on Turbo Codes & Related Topics*, Brest, 2003, pp. 451–454.

[15] Jutaphet Wetcharungsri, Narong Buabthong, Pakon Thuphairo, Paramin Sangwongngam, Keattisak Sripimanwat, "Improvement of Flexible Semi-Parallel LDPC Decoders over FPGA for WiMAX Standards" in *Proc. of the ICICTES2014*, Ayutthaya, Thailand.

[16] François Charot, Christophe Wolinski, Nicolas Fau, François Hamon, "A Parallel and Modular Architecture for 802.16e LDPC Codes," dsd, 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools, pp. 418-421, 2008

[17] K. Zhang and X. Huang "A low-complexity rate-compatible LDPC decoder", *Proc. Asilomar Conf. Signals, Syst., Comput.*, pp.749 -753 2009

[18] Shuangqu Huang; Bo Xiang; Bei Huang; Chen, Yun; Xiaoyang Zeng, "A flexible architecture for multi-standard LDPC decoders," *ASIC, 2009. ASICON '09. IEEE 8th International Conference on* , vol., no., pp.493,496, 20-23 Oct. 2009