A High-Throughput FPGA Implementation of Quasi-Cyclic LDPC Decoder

Hossein Gharaee, Mahdie Kiaee and Naser Mohammadzadeh

[1]Iran Telecommunication Research Center
[2]Department of Computer Engineering, Shahed University, Iran

Abstract
Quasi-cyclic low-density parity-check (QC-LDPC) codes are an important subclass of LDPC codes that are known as one of the most effective error controlling methods. Many important communication standards such as DVB-S2 and 802.16e use these codes. In this paper, an FPGA implementation of a partial-parallel QC-LDPC decoder is proposed based on the sum-product algorithm. We use a modified version of TPMP[1] algorithm to improve the number of clock cycles, resource usage, and power consumption. The decoder is implemented for a code length of 672 with code rate of $\frac{3}{4}$. Our implementation is achieved to maximum throughput of 3.3 Gbps with frequency of 280 MHz and its power consumption is less than 150mW.

Key words:
QC-LDPC decoder; Time Scheduling; TPMP Algorithm; FPGA Implementation;

1. Introduction

Low Density Parity Check (LDPC) codes are a kind of linear block codes that were discovered by Gallagher in 1962 [1]. Some advantages of these codes are their high error-correction performance as close as to the Shannon limit and sparse parity check matrix [2]. Moreover, the intermediate complexity of decoding with a considerable level of parallelism in hardware implementation made these codes suitable for the new wireless communication systems [3]. LDPC codes are used in industrial standards like wireless LAN (IEEE 802.11n), Mobile WiMax (IEEE 802.16e) and 10 Gb/s Ethernet (10GBASE-T). In addition, LDPC codes are widely used in satellite television, space communications, magnetic storage in hard disk drives, and optical networking [2, 4-6].

The decoding of LDPC codes is based on the belief propagation algorithm which is known as Sum-Product algorithm (SPA) and needs complex number calculations. Min-sum algorithm (MSA) is a simple kind of SPA [7]. The Sum-Product algorithm has better performance in error correction than Min-Sum algorithm but simple check node unit in MSA needs smaller area and low memory [2]. So, designing a decoder based on SPA with simple check node process unit is an important factor in decoder architecture. LDPC decoders have complex number calculations that cause to low delay and high throughput and also need strong hardware architecture [8].

There are three methods for hardware implementation of LDPC decoders including: Fully-parallel ,serial, and semi-parallel [9]. The fully-parallel architecture implements all check nodes and variable nodes of parity check matrix as a process unit, the serial architecture implements only one check node unit and variable node unit, and the semi-parallel architecture is between fully-parallel and serial [9].

Regardless of parallelism, time scheduling has also a strong effect on the implementation of decoder. The timing of decoder is done in two ways: Two-Phase-Message-Passing (TPMP) decoding and Turbo-Decode-Message-Passing decoding that is known as layered decoding. Convergence of decoding in layer decoding is two times faster than TPMP with 50% reduction in iteration [10, 11].

The TPMP algorithm has optimal error-correction performance, but large exponential numbers, look-up table and multiplicative operations increase its hardware difficulty [12]. Some simplified algorithms are proposed based on the TPMP algorithm. Sum-Product Log-Domain algorithm uses log-likelihood ratio (LLR) which avoids exponential computations and numerical instability. Min-Sum algorithm has less complexity but suffers from heavy performance loss [13].

The layer decoding is the most common way in LDPC decoding. In recent years, most of decoders were based on TPMP, but today due to higher convergence speed of TDMP, using the layer decoding is more common [13].

This paper is organized as follows: an overview of the prior works on LDPC decoder is presented in Section 2. In Section 3, some basic concepts of LDPC decoding are explained. Section 4 discusses the details of our proposed architecture and in section 5 the proposed time scheduling

[1] Two-phase-Message-passing

Manuscript received March 5, 2017
Manuscript revised March 20, 2017

is discussed. Section 6 and 7 detail pipeline and parallelism of decoder and Section 8 shows the implementation results. Section 9 concludes the paper.

2. Related Work

In TPMP algorithm, updating of check nodes and variable nodes is performed in discrete units and calculation of variable to check node messages is done after completion of check to variable node message calculation, but in the layered architecture, parity check matrix is divided into m layers that m is the number of rows in parity check matrix and the processing must be done for all m layers and when the row is processed, the processing of column is started. Moreover, the processing has multiple iterations and each iteration of each layer itself has sub-iterations [11, 14-17].

In [18], an innovative dual-shift stochastic-detection (DSSD) technique is proposed to Deal with partial-parallel cascaded TPMP decoder weaknesses that mitigates computational resources [18].

There are different Parallelism methods in check node and variable node units. In the first type, M check node units (CNUs) and N variable node units (VNUs) operate in parallel. Each CNU updates one row and each VNU updates one column in one clock cycle. Therefore, Z rows of a block row are updated in Z clocks and Z columns of a block column are updated in Z clocks, thus, each iteration takes 2×Z clock cycles. In the second type, Z check node processing units and Z variable node processing units can operate simultaneously. Each CNU updates each row in W_c clock cycles and Z rows of a block row will be updated at the same time. Similarly, each VNU updates a column in W_v clock cycles and Z columns of a block column are updated at the same time. Different methods such as $\frac{Z}{4}$ of CNUs and $\frac{Z}{4}$ of VNUs, or Z CNUs and N×Z VNUs or 2×Z CNUs and Z VNUs were implemented [19]. The overall architecture of layer decoding is shown in Figure 1.

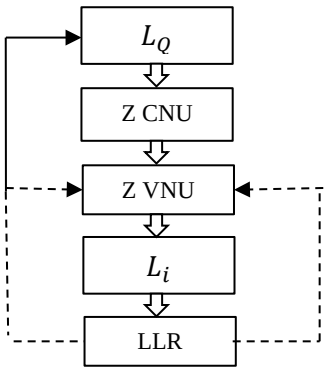

Fig. 1 Architecture of layer decoding

L_i is the sum of variable nodes messages to check nodes in a layer and L_Q is the received message from channel in first iteration and received message from previous iteration in next iterations that is used to initialize check node messages.

Since in layer decoding the processed information in each layer is the output of the previous layer processes, fetching calculated values from RAM memory should be done with no data conflict.

In [20], the idle times to calculate the correct information in layer decoding is provided for each layer. Maximum number of clock cycles is equale to $T_{max} = M \times (N + 1)$. After the row process of the first layer, row operations of the second layer and column operations of the first layer starts at the same time. In this schedule, the W clock cycles are considered to prepare data for row operation of the second layer. Since the parity check matrix is related to Quasi-Cyclic codes, the row weigh is 1 for each sub-matrice.Therefore, before completion of column operations of a block row, the row processing of the second layer is not possible. On the other hand, since the row and column processes are limited to the row weight and because of different weights of different rows, the timing must be appropriate to the particular parity check matrix defined for decoder, thus, the flexibility decreases. In the proposed architecture timing and the pipelining were considered with respect to row weight of each layer which causes inflexibility of decoder for different code rates [20].

Decoding scheduling depends on design limitations. In [21], a half-row layered decoder with reduced routing network was presented in which variable node parallelism is equal to half of code length. Routing network that is responsible for routing of check node data to variable node is eliminated by changing the shift value of each block data sent from variable node to check node. The plan proposed in [21] is a trade-off between complexity and throughput that provides good results in energy and area. The code length is considered 672 and the size of sub-matrix is equal to 42. Permutation network is used to link between variable and check nodes, and parallel degree for check node units is 42 and for variable node units is 672. In half-row design, the parallel degree for variable node is reduced to 336 from 672 but the degree of check node remains 42. Due to the complexity of check node unit and high density of hardware resources used in check node unit and permutation network, proposing a reduced parallell degree architecture is necessary. Moreover, to support several expansion and parallelism factors in decoding, a great flexibility for interconnection network of variable nodes and check nodes is required. Use of fixed connections based on $Z \times Z$ in parity check matrix, supports expansion and parallelism factors less than Z. However, less parallel degree needs less hardware resources, therefore, some hardware resources remain idle in low parallel degree [19].

The other important point is that the RAM memory that is used to store transferred messages between decoder nodes. In some designs, registers are preferred because of faster access time at the expense of high power and high area [22]. In fact, choosing register or memory in decoding is a trade-off between power and area. In the architecture proposed in [23], an input buffer is used for transferred data which consist of an array of RAMs. The number of RAMs is N , which N is the column size of H matrix and depth of each RAM is Z which is the expansion factor.

Some of LDPC decoders are implemented in graphics processor unit (GPU) that is a low-cost and flexible software-based multi-core architecture for complex computations [24]. The architecture in [24] proposed a real-world digital signal processing (DSP) LDPC decoder that efforts to map the algorithm onto the massively parallel architecture of GPU and fully utilize GPU's computational resources to reach a high throughput and performance that can take advantage of the multi-core computational power provided by GPU. Also this plane uses the early termination (ET) algorithm which is used to converges faster to true code word in decoding and avoid unnecessary computations[24].

.In most recent designs of LDPC decoders, permutation networks were used for transferring messages and most of the hardware resources are allocated to it [15, 16, 19, 20, 25-28]. Therefore, the removal of permutation network without increasing complexity in routing between nodes is important. If the row and column weights of parity check matrix are low, permutation network complexity will be low. However, if the number of lines between variable and check nodes is high, i.e. for a code with high rate and throughput, routing and hardware complexity of permutation network will be a critical issue [25]. In contrast, fixed connections between variable nodes and check nodes can be used instead of permutation networks. Fixed connections not only reduce hardware resources but also increase the flexibility of decoder for both regular and non-regular parity check matrices. Moreover, it provides the capability of reconfiguration to implement both structured and random codes without the limitations of parity check matrix [29].

In [28], a fully pipelined QC-LDPC decoder is presented that consists of M check nodes and N variable nodes that has high parallel degree which supports variable block sizes and multiple code rates. The proposed architecture is implemented based on Quasi-Cyclic features and layered decoding through efficient utilization of permutation network and small check node design that has reduced interconnect complexities, area, power consumption and less memory bits used. The proposed architecture reduces hardware resource utilization by using only one permutation network rather than preprocessing and post-processing permutation network in some other decoders.

Parity check matrices of LDPC decoders include regular [12, 30] and irregular [31] structures. Irregular codes have a better performance than the regular codes and provide better protection for some bits of input code word and provide greater reliability for data because the codes with greater degree converge faster and assist the codes with less degree [22]. The architecture design for irregular codes is more challengeable and should be designed flexible to support matrices with different number of rows and columns. Moreover, a way should be proposed that provides optimal communication between variable and check nodes.

There is multi-dimensional design space for LDPC decoder that consists of decoding scheduling, optimization of check node simplification, parallelism expansion, pipeline stage optimization and etc. To design a high-throughput decoder, the design should be done based on multi-dimensional optimization. The optimization depends on the constraints such as performance, power, area, and hardware resources [31]. In implementation of error-correcting codes, high throughput, the area-efficient implementation of integrated circuits and enforcement of limitations on power and speed is important. The realization of high-throughput decoder for wireless networks (supports data rates up to several hundred megabits per second) needs to high parallelism or increasing the clock frequency, which effect on the overhead of area and power. However, the parallel combination of check nodes is not fully implemented yet which it could increase throughput significantly with an acceptable complexity [4]. The overall architecture of [26] is based on DSP algorithms and parallelism that shows Designing the optimal decoder requires a tradeoff between hardware complexity, throughput, and performance.

Many decoding methods were proposed according to the iterative decoding which includes high-parallel degree and low-parallel degree. In decoders with high parallelism, check node units, variable node units and interconnections are implemented in a single chip. All messages are calculated in parallel and each iteration of decoding is performed in one clock cycle. Decoders with high parallelism have short decoding delay and high-throughput, but have large silicon area. In contrast, decoders with a low degree of parallelism require less processing units and memories with higher density instead of separate registers, thus the area is smaller and lower throughput is provided [13].

In this article, a QC-LDPC decoder for LAN (IEEE 802.11ad) standard, with $\frac{3}{4}$ code rate and the code length of 672 is implemented. The architecture of QC-LDPC code is based on sum-product algorithm. To create an acceptable

trade-off between parallelism level of check node units, variable node units, and maximum throughput, a scheduling with semi-parallel structure is proposed. In this approach, two categories of pipeline related to synchronization of inner operations of check nodes plus synchronization of variable and check node operations are presented. By creating pipeline and simultaneity in variable and check node operations with a proper degree of parallelism, the decoding clock cycles decreases.

3. Background

In this section, some terminologies and concepts are given that would help in the description of the proposed architecture in the next section.

a. LDPC codes

An LDPC code is defined with a parity check matrix $H_{m\times n}$ which n is code length (number of bits in code word) and m is the number of parity check equations. Parity check matrix is shown by a bipartite tanner graph with two sets of nodes. This graph consists of variable nodes (VNs) for each column and check nodes (CNs) for each row. The tanner graph of parity check matrix H is shown in Figure 2.

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

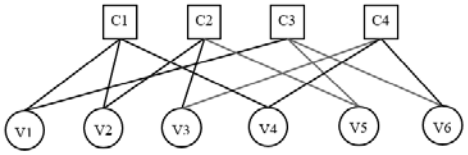

Fig. 2 Tanner graph of parity check matrix H

QC-LDPC codes are one of the main branches of LDPC codes [4]. Because of excellent decoding performance and Quasi-Cyclic structure of these codes, they need less hardware resources and have less hardware implementation complexity.

In QC-LDPC codes, the parity check matrix is divided into $M_b \times N_b$ square sub-matrixes in size $z \times z$ which is in form of $M = M_b.Z$ and $N = N_b.Z$ [7].

LDPC codes have a close performance to Shannon limit by using Belief-Propagation (BP) algorithm. BP algorithm or equivalently sum-product algorithm (SPA) has the best decoding performance but its complex computing increases hardware resources. To reduce the complexity of SPA, a modified simple algorithm is derived of SPA which is called Min-Sum algorithm (MSA). MSA decreases the complexity of SPA with a reduction in decoding performance [17]. SPA and MSA are widely used rather than other LDPC decoding algorithms [7]. In both decoding algorithms, the H matrix is first initialized with channel output and other decoding steps are based on transferring the messages from check nodes to variable nodes and vice versa. Introduced algorithms are according to soft decision which decision about code word is based on received possibilities from the nodes.

b. Sum-Product Log-Domain Algorithm

For simplifying the computation in sum-product algorithm, the log likelihood ratio of prior (messages received from channel) and posterior (messages transferred between check nodes and variable nodes) probability is used. P_i^{int} is the Prior probability of ith bit of received messages from channel, so According to LLR[2] equation, the log likelihood ratio of prior probability is shown in Equation (1).

$$L_i = log\left(\frac{1-P_i^{int}}{P_i^{int}}\right) = LLR\left(P_i^{int}\right) \quad (1)$$

The L_i values are put into vector $L = [L_1\ L_2\ \ldots\ L_n]$. In the first step, variable nodes are initialized with L vector. The second step is related to check nodes process. In this step, log likelihood ratio of posterior probability of ith bit that is received from jth equation is shown in Equation (2). Thus, the sent message from check nodes to variable nodes is equal to:

$$E_{ij} = 2\tanh^{-1}\left(\prod_{i'\in B_j, i'\neq i}\tanh\frac{B_{ij}}{2}\right) \quad (2)$$

The process of variable nodes is the third step of decoding that is shown in Equation (3). In this step, L_j is the first probability received from channel and $\sum_{i'\in M_j} E_{i'j}$ is sum of messages sent from check nodes to jth variable node.

$$A_j = L_j + \sum_{i'\in M_j} E_{i'j} \quad (3)$$

In final step, according to computed value in equation (3), the Hard-Decision is made to derive the vector z. In LLR function, if $A_j \leq 0$, then $q_j(0) > q_j(1)$ and the jth bit of vector Z equals to 1 else it equals to 0. Then, in the final step of each iteration, the code word is estimated. If the estimated code word in vector Z won't satisfy the Equation (4), the code word is invalid and decoder starts next

[2] logarithm likelihood ratio

iteration until a valid code word is received or the maximum iteration is reached [20].

$$H.Z^T = 0 \qquad (4)$$

If the derived code word is incorrect, Equation 5 is computed that B_{ij} is input value for next iteration and in first iteration B_{ij} is extracted from H matrix which is initialized with L vector [20].

$$B_{ij} = L_j + \sum_{i' \in M_j, i' \neq i} E_{i'j} \qquad (5)$$

4. The Proposed Architecture

As mentioned in the previous sections, in addition to parallelism, the timing of decoding affects the design of decoder. Previously proposed designs of LDPC decoders implemented layer decoding. Data dependency between each layer is the problems of layer decoding. Propagation of messages in each layer begins when the posterior information corresponding to previous layer have been updated that prevents the pipeline between two layers. Compared with layer decoding, two phase message passing decoding has less data dependency with the expense of doubling the number of iterations in the same BER[3]. These two types of decoding have limited advantages, therefore, the timing of decoding depends on design constraints [21]. The implementation of E_{ij} is the most complex part of decoder. In log likelihood ratio decoding, the E_{ij} is converted to Equation (6) [4]:

$$E_{ij} = 2\tanh^{-1}\left(\prod_{i' \in B_j, i' \neq i} \tanh\frac{B_{ij}}{2}\right) \qquad (6)$$

Therefore, division and logarithm calculations are eliminated and hardware complexity is reduced due to two kinds of LUT[4] for $\tanh\frac{B_{ij}}{2}$ and $\tanh^{-1}\left(\prod_{i' \in B_j, i' \neq i} \tanh\frac{B_{ij}}{2}\right)$.

To simplify the operation of check node, E_{ij} is divided into several parts and the process is done in several modules, including $tanh$ LUT, multiplier Uni,t and $tanh^{-1}$ LUT. Matrix row multiplication is conducted by using cyclic shift register which reuse of hardware causes to reduce hardware resources.

Block diagram of check node unit is shown in Figure 3. After n clock cycles required for variable node processes and checking all the possibilities of A_j, the binary vector of length n bits (the number of columns of H matrix) will be start. Block diagram of variable node is shown in Figure 4.

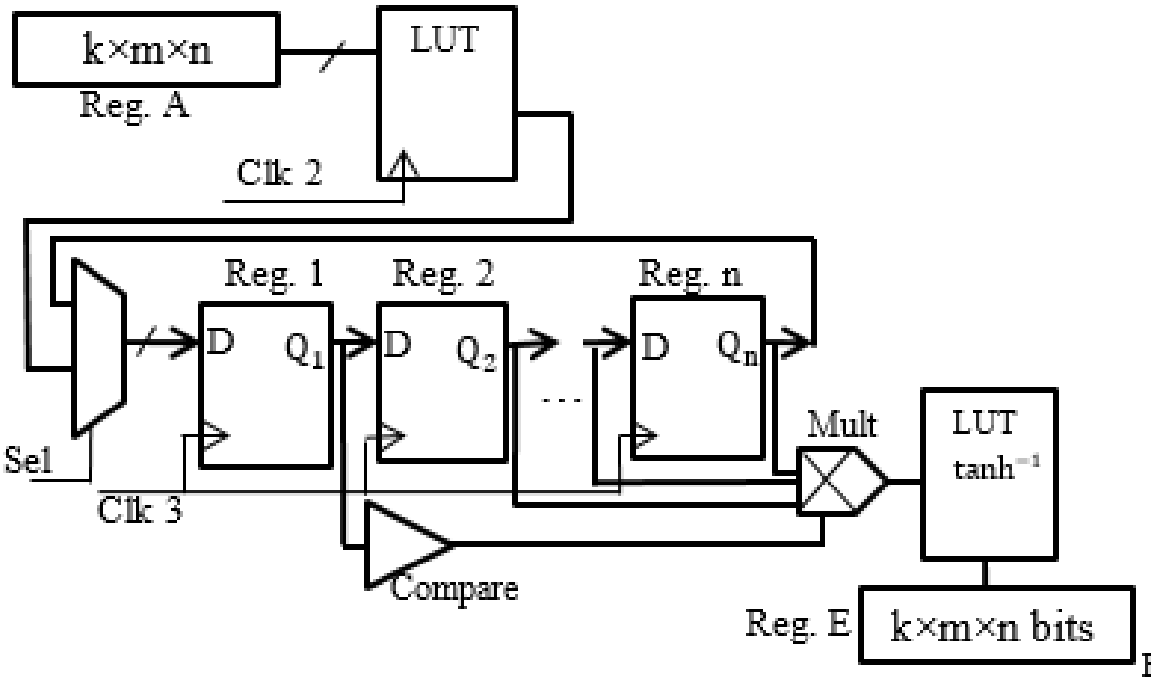

Fig. 3 Block diagram of check node unit

[3] Bit Error Rate

[4] Lookup table

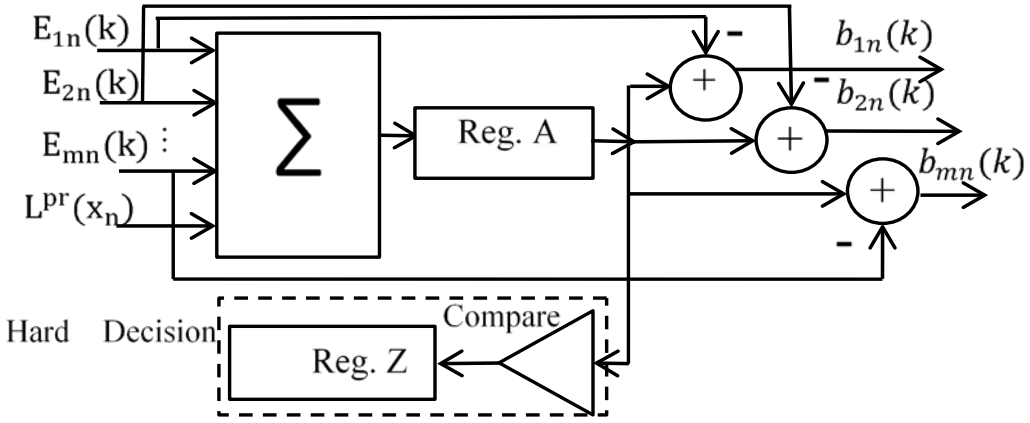

Fig. 4 Block diagram of variable node

5. Time Scheduling

Because of better error correction performance of two-phase-message-passing decoding than layer decoding, a new time scheduling is proposed that row and column operations are done simultaneously without data interfering. Without pipeline between inner check node operations and simultaneity of check node and variable node processes, the number of clock cycles will significantly increase. In the proposed architecture, the number of clock cycles is equal to m+5 where m is the number of rows in parity check matrix. In the prior layered architectures, if the number of parallel processing units is equal to m for check node units and n for variable node units, the number of clock cycles required for row and column processing was $2 \times Z$ and if the parallelism degree of CNUs and VNUs increase to Z, the number of clock cycles for row and column processing is equal to weigh of rows and columns in matrix that the total number is:

$$number\ of\ cycles = (W_r + W_c) \times N_{layer}$$

Therefore, the number of clock cycles for a decoder according to parity check matrix with $m \times n$ cyclic sub-matrix of size $Z \times Z$ is different based on row and column weight. Reducing the clock cycles without increasing the parallelism between CNUs and VNUs is not possible. Therefore, implementation of TPMP decoding with the appropriate time scheduling that reduces the number of clock cycles and parallelism degree, has better error correction performance and hardware complexity than layered decoding.

6. Pipeline

In TPMP, after completing processing in check nodes, processing in variable nodes begins. In CNU, the value of $tanh$ of each row is derived in one clock, so m clocks are required and also $tanh^{-1}$ computation is the same. Totally if all three operations on check node block is conducted sequentially, $3 \times m$ clocks are required. Operation of variable node unit is done in n clocks which equal to columns of parity check matrix. Hard decision is done in one clock and finally $3 \times m + n + 1$ clock cycles are required.

As mentioned above, decoding speed and throughput are the major issues in LDPC decoder implementation. By overlapping of check node and variable node operations, in addition to reducing clock cycles and increasing throughput, the hardware resources will be decreased.

In this article, two sets of pipelines related to check nodes and variable nodes processing are presented. The first proposed pipeline is related to medial operations of CNU, including $tanh$, row multiplication and $tanh^{-1}$. The Second one is related to simultaneity of check node operations and variable node operations. If the size of parity check matrix H is 4×6, decoding timing without simultaneity is shown in Figure 5. Three stages of check node operations are done simultaneously and variable node operations consist of column addition begin after check node process and take n clock cycles.

The second kind of pipeline is shown in Figure 6 which consists of simultaneous operation of CNU and VNU. After check node operation of 1st and 2nd row in 5th clock, addition of variable nodes begin with 3rd row operation and the operations of other columns are done in the same way. Thus by the proposed two-stage pipeline, the number of clock cycles is reduced.

CNU				
Cycle 1	r_1			
Cycle 2	Tr_1	r_2		
Cycle 3	MTr_1	Tr_2	r_3	
Cycle 4	$T^{-1}r_1$	MTr_2	Tr_3	r_4
Cycle 5		$T^{-1}r_2$	MTr_2	Tr_2
Cycle 6			$T^{-1}r_2$	MTr_2
Cycle 7				$T^{-1}r_3$

Fig. 5 Pipeline of CNU and VNU separately

CNU&VNU				
Cycle 1	r_1			
Cycle 2	Tr_1	r_2		
Cycle 3	MTr_1	Tr_2	r_3	
Cycle 4	$T^{-1}r_1$	MTr_2	Tr_3	r_4
Cycle 5		$T^{-1}r_2$	MTr_3	Tr_4
Cycle 6		$cr_1 + cr_2$	$T^{-1}r_3$	MTr_4
Cycle 7			$cr_{12} + cr_3$	$T^{-1}r_4$
Cycle 8				$cr_{123} + cr_4$

Fig. 6 Pipeline of CNU and VNU simultaneously

7. PARALLELISM

Simultaneous operations of variable nodes and check nodes reduce the number of clock cycles. Moreover, pipeline between processing units and reuse of the hardware, reduce the hardware complexity.

In addition, cyclic shift registers in check node processing unit reduce multipliers. Moreover, due to simultaneous operations in VNU, only two rows of the matrix elements are added together in a column order per clock cycle. Therefore, the number of required adders is equal to the number of matrix columns. Tree adder is proposed for VNU, which only one addition for per column is done in each clock cycle. The parallelism in variable nodes is shown in Figure 7. Another parallel operation in decoding corresponds to check nodes and variable node processors. Given that the number of clocks needed to complete per iteration is equal to m + 5, the greater size of m will also increase the number of cloc k cycles. Thus, by parallelizing of several decoding units including CNU and VNU, the number of clock cycles reduces. Reducing the number of clock cycles depends on the amount of increasing of parallelism degree. By two parallel decoding units operating together, the number of clock cycles reduces to $\frac{1}{2}$.

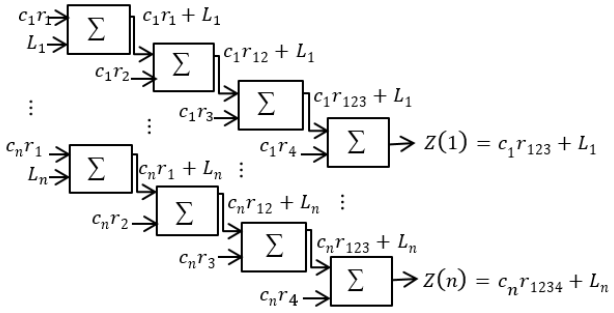

Fig. 7 The parallelism of variable node unit adders

The number of CNUs and VNUs working in parallel is equal to the number of rows in base parity check matrix. In fact, each row of base parity check matrix of QC-LDPC code is considered as a layer and row and column processing is done for each layer separately. The proposed parallelism in this paper reduces the decoder processor units. Table 1 compares the complexity of clock cycle in proposed method with earlier scholars. The clock cycles of decoding in three methods including TPMP, layered and our proposed timing. The number of clock cycles in each kind of decoding is computed in two types of low and high parallelism sequentially in type 1 and type 2. The fields of CN and VN show the parallel degree of variable nodes and check nodes.

According to Table 1, there is a trade-off between parallelism degree and number of clock cycles. Using high parallel units in TPMP and Layered decoding decreases the number of clock cycles and reducing the number of parallel units increases the number of clock cycles, but the comparison of proposed timing schedule with TPMP and Layered decoding shows that an acceptable level of parallelism consist of n variable nodes and m check nodes causes to the less number of clock cycles in the same degree of parallel units.

Table 1 Number of clock cycles

		TPMP		Layered		Proposed timing	
		Type1	Type2	Type1	Type2	Type1	Type2
Parallel units	VN	1	n × Z	N	Z	1	n
	CN	1	$m \times Z$	m	Z	1	m
Clock cycle		(m + n) × Z	2	2 × Z	Wc + Wr	$Z \times m + 5$	$Z + 5$

8. Experimental Results

A LDPC decoder which supports LAN (IEEE 802.11ad) standard has been synthesized on a XILINX VIRTEX6. The base parity check matrix size is 4×16 with the sub-matrix of size 42×42. Table 2 compares this decoder with the state of-the-art LDPC decoder of [5], [20], and [27].

Table 2 consists of rate and length of code words in decoder, used algorithm for decoding, maximum frequency of decoder, maximum number of decoding iteration, throughput, area of decoder and power that shows the fixed-point simulation results of TPMP decoding modes for the code length of 672-bit. The VLSI implementation results show that the proposed decoder occupies an area of $3.4mm^2$ and achieves maximum decoding throughput of 3360 Mbps with maximum 10 iterations. The estimated power consumption is 150 mW in frequency of 280 MHz. The results show that this decoder presents high Throughput with less power consumption and area by implementing sum-product algorithm in proposed time scheduling.

Table 2. Throughput and area comparison of the proposed and prior LDPC decoder implementations

	[20]	[5]	[27]	The proposed approach
Code rate	$\frac{5}{6}$	$\frac{1}{2}$	$\frac{1}{2},\frac{2}{3},\frac{3}{4},\frac{5}{6}$	$\frac{1}{2},\frac{3}{4}$
Code length	1296	2304	1944	672
Algorithm	MSA TDMP	MSA TDMP	MSA TDMP	SPA TPMP
Freq. (MHZ)	230	100	250	280
Max Iteration	5	10	1-7	10
Throughput (Mbps)	767	183	672	3360
Area (cm^2)	3.12	6.25	3.67	3.4
Power (mw)	-	242	171.07	150

In Table 3, the logic resource utilisation of the FPGA is shown. The proposed architecture is compared with [28], [18] and [26]. The proposed decoder in [28] and [26] permutation network is used for transferring messages between variable nodes and check nodes, implements a layered decoder with Min-Sum algorithm. The recent architectures are compared with proposed architecture with new time scheduling in this article. The rows of this table consist of logic resources.

Table 3 shows that in our proposed approach registers are used to store decoder messages but in other refrences In addition to the registers, block RAMs are used that effect on decoding speed. Comparison of resources in proposed decoder with previous ones shows that number of slice Registers, the number of slice LUTs and LUT-FF pairs in our plane for the code rate of 4489 has better result rather than same code rate In [18]. And the number of fully used LUT- FF pairs and Number of occupied slice have optimized in code rate of 672 and 2304 rather than same ones in [26, 28].

Table 3 Synthesis results of LDPC decoder for IEEE 802.11ad on XILINX VIRTEX6

	[28]	[26]	[18]	The proposed Approach		
Code length	682-MSA	2304-MSA	4489-MSA	672	2304	4489
Number of slice Registers	N FIFO	3086 Block RAM=15	32435 N+M Block RAM	13712	44970	48320
Number of slice LUTs	35668	13555	63453	56832	58632	60562
Number of occupied slice	13229	4446	-	2968	3423	4744
Number of fully used LUT- FF pairs	-	3086	29760	6090	19973	27853

9. Conclusion

An efficient partially parallel decoder architecture based on sum-product algorithm for QC-LDPC was proposed in this paper. The proposed architecture has eliminated the permutation network for transferring data between check nodes and variable nodes, so complexity overhead of the switch network has removed. With synchronization between process units, based on proposed pipeline between VNUs and CNUs, the number of clock cycles, hardware resources, and power has been reduced.

The decoder block processing unit is proposed for rates $\frac{1}{2}$ and $\frac{3}{4}$ of length 672. The decoder has been implemented on FPGA the results show that decoders can achieve throughput of 3360Mbps that is better than the prior works.

References

[1] R. Gallager and L.-D. P.-C. Codes, "MIT press, 1963," *Low-Density Parity-Check Codes*.

[2] X. Pan, X.-f. Lu, M.-q. Li, and R.-f. Song, "A high throughput LDPC decoder in CMMB based on virtual radio," in *Wireless Communications and Networking Conference Workshops (WCNCW), 2013 IEEE*, 2013, pp. 95-99.

[3] A. Shevchenko, R. Maslennikov, and A. Maltsev, "Comparative analysis of different hardware decoder architectures for IEEE 802.11 ad LDPC code," in *MELECON 2014-2014 17th IEEE Mediterranean Electrotechnical Conference*, 2014, pp. 415-420.

[4] J. Kim and W. Sung, "Rate-0.96 LDPC decoding VLSI for soft-decision error correction of NAND flash memory," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, pp. 1004-1015, 2014.

[5] T. Heidari and A. Jannesari, "Design of high-Throughput QC-LDPC Decoder for WiMAX standard," in *2013 21st Iranian Conference on Electrical Engineering (ICEE)*, 2013, pp. 1-4.

[6] F. Angarita, J. Valls, V. Almenar, and V. Torres, "Reduced-complexity min-sum algorithm for decoding LDPC codes with low error-floor," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, pp. 2150-2158, 2014.

[7] B. Belean, S. Nedevschi, and M. Borda, "Application specific hardware architecture for high-throughput short-length LDPC decoders," in *Intelligent Computer Communication and Processing (ICCP), 2013 IEEE International Conference on*, 2013, pp. 307-310.

[8] J. Andrade, G. Falcao, and V. Silva, "Flexible design of wide-pipeline-based WiMAX QC-LDPC decoder architectures on FPGAs using high-level synthesis," *Electronics Letters*, vol. 50, pp. 839-840, 2014.

[9] C. Beuschel, "Fully programmable LDPC decoder hardware architectures," Universität Ulm, 2010.

[10] S.-I. Hwang and H. Lee, "Block-circulant RS-LDPC Code: code construction and efficient decoder design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, pp. 1337-1341, 2013.

[11] S. Huang, D. Bao, B. Xiang, Y. Chen, and X. Zeng, "A flexible LDPC decoder architecture supporting two decoding algorithms," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 2010, pp. 3929-3932.

[12] Y.-H. Chen, C.-L. Chu, and J.-S. He, "FPGA implementation and verification of LDPC minimum sum algorithm decoder with weight (3, 6) regular parity check matrix," in *Electronic Measurement & Instruments (ICEMI), 2013 IEEE 11th International Conference on*, 2013, pp. 682-686.

[13] B. Xiang, D. Bao, S. Huang, and X. Zeng, "An 847–955 Mb/s 342–397 mW dual-path fully-overlapped QC-LDPC decoder for WiMAX system in 0.13 m CMOS," *IEEE Journal of Solid-State Circuits*, vol. 46, pp. 1416-1432, 2011.

[14] A. Ramez and A. jannesari, "Hardware implementation of LDPC decoder for WiMax standard," *Journal of Electronics Industry*, vol. 4, pp. 29-38, 2013.

[15] Y. S. Park, D. Blaauw, D. Sylvester, and Z. Zhang, "Low-power high-throughput LDPC decoder using non-refresh embedded DRAM," *IEEE Journal of Solid-State Circuits*, vol. 49, pp. 783-794, 2014.

[16] X. Zhao, Z. Chen, X. Peng, D. Zhou, and S. Goto, "High-parallel performance-aware LDPC decoder IP core design for WiMAX," in *2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2013, pp. 1136-1139.

[17] Y. Sun and J. R. Cavallaro, "VLSI architecture for layered decoding of QC-LDPC codes with high circulant weight," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, pp. 1960-1964, 2013.

[18] M. H. L. Lim and W. L. Goh, "High-throughput dual-shift stochastic-detection quasi-cyclic LDPC decoder," in

Information, Communications and Signal Processing (ICICS) 2013 9th International Conference on, 2013, pp. 1-5.
[19] M. Awais, A. Singh, E. Boutillon, and G. Masera, "A novel architecture for scalable, high throughput, multi-standard LDPC decoder," in *Digital System Design (DSD), 2011 14th Euromicro Conference on*, 2011, pp. 340-347.
[20] Z. Luan, Y. Pei, and N. Ge, "A fast convergence and area-efficient decoder for quasi-cyclic low-density parity-check codes," in 2013 19th Asia-Pacific Conference on Communications (APCC), 2013, pp. 458-462.
[21] M. Li, F. Naessens, P. Debacker, P. Raghavan, C. Desset, M. Li, *et al.*, "An area and energy efficient half-row-paralleled layer LDPC decoder for the 802.11 AD standard," in *SiPS 2013 Proceedings*, 2013, pp. 112-117.
[22] Y. S. Park, D. Blaauw, D. Sylvester, and Z. Zhang, "A 1.6-mm 2 38-mW 1.5-Gb/s LDPC decoder enabled by refresh-free embedded DRAM," in *2012 Symposium on VLSI Circuits (VLSIC)*, 2012, pp. 114-115.
[23] Y. Chen, X. Chen, Y. Zhao, C. Zhou, and J. Wang, "Design and implementation of multi-mode QC-LDPC decoder," in *Communication Technology (ICCT), 2010 12th IEEE International Conference on*, 2010, pp. 1145-1148.
[24] G. Wang, M. Wu, Y. Sun, and J. R. Cavallaro, "A massively parallel implementation of QC-LDPC decoder on GPU," in *Application Specific Processors (SASP), 2011 IEEE 9th Symposium on*, 2011, pp. 82-85.
[25] C.-W. Sham, X. Chen, F. C. Lau, Y. Zhao, and W. M. Tam, "A 2.0 Gb/s throughput decoder for QC-LDPC convolutional codes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, pp. 1857-1869, 2013.
[26] J. Wetcharungsri, N. Buabthong, S. Jantarachote, P. Sangwongngam, and K. Sripimanwat, "Field-programmable gate array implementation of low-density parity-check codes decoder and hardware testbed," in *TENCON Spring Conference, 2013 IEEE*, 2013, pp. 104-107.
[27] C. Yu, H.-S. Chuang, B.-S. Lin, P.-H. Cheng, and S.-J. Chen, "Improvement on a block-serial fully-overlapped QC-LDPC decoder for IEEE 802.11 n," in *2014 IEEE International Conference on Consumer Electronics (ICCE)*, 2014, pp. 446-447.
[28] S. A. Zied, A. T. Sayed, and R. Guindi, "Configurable low complexity decoder architecture for Quasi-Cyclic LDPC codes," in *Software, Telecommunications and Computer Networks (SoftCOM), 2013 21st International Conference on*, 2013, pp. 1-5.
[29] R. Li, J. Zhou, Y. Dou, S. Guo, D. Zou, and S. Wang, "A multi-standard efficient column-layered LDPC decoder for software defined radio on GPUs," in *2013 IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2013, pp. 724-728.
[30] J. C. Porcello, "Designing and implementing low density parity check (ldpc) decoders using fpgas," in *2014 IEEE Aerospace Conference*, 2014, pp. 1-7.
[31] M. Karkooti, P. Radosavljevic, and J. R. Cavallaro, "Configurable, high throughput, irregular LDPC decoder architecture: Tradeoff analysis and implementation," in *IEEE 17th International Conference on Application-specific Systems, Architectures and Processors (ASAP'06)*, 2006, pp. 360-367.