Flexible LDPC Decoder Design for Multi-Gb/s Applications

Chuan Zhang, *Student Member, IEEE*, Zhongfeng Wang, Jin Sha, Li Li, and Jun Lin

***Abstract*—Low-density parity-check (LDPC) codes are one of the most promising error correcting codes (ECC) approaching Shannon capacity and have been adopted in many applications. However, efficient implementation of high-throughput LDPC decoders adaptable for various channel conditions still remains challenging. In this paper, a low-complexity reconfigurable VLSI architecture for high-speed LDPC decoders is presented. *Shift*-LDPC codes are incorporated within the design, and have shown not only comparable decoding performance to computer generated random codes but also high hardware efficiency in high-speed applications. Single-minimum Min-Sum decoding scheme and non-uniform quantization scheme are explored to reduce the complexity of computing core and the memory requirement. The well-known Benes network is employed to construct the configurable permutation network to support multiple *shift*-LDPC codes with various code parameters. The ASIC implementation results of an (8192, 7168) (4, 32)-regular *shift*-LDPC decoder demonstrate a maximum decoding throughout of 3.6 Gb/s at 16 iterations, which outperforms the state-of-the-art design for high speed flexible LDPC decoders by many times with even less hardware.**

***Index Terms*—Low-density parity-check (LDPC) codes, flexible structures, iterative decoding, very large scale integration (VLSI) architecture, error correction codes.**

I. Introduction

LOW-DENSITY parity-check (LDPC) codes, which were originally introduced by Gallagher in his PhD dissertation in the early 1960s [1], had been long time ignored for the requirement of high complexity computation. Since their rediscovery by MacKay and Neal [2], [3], LDPC codes have become one of the most attractive topics of interest in both academia and industry. Compared with turbo codes, LDPC codes are well-suited for wireless, optical, and magnetic recording systems due to their near Shannon limit error correcting capacity, low error floor, reasonable implementation complexity as well as high intrinsic degrees of parallelism. With these remarkable characteristics, LDPC codes have been recently adopted in several industrial standards such as Wireless Local Area Networks (WLAN) (IEEE 802.11n) [4], Wireless Metropolitan Area Networks (WMAN) (IEEE 802.16e) [5], China's Digital Television Terrestrial Broadcasting standards (DTTB) [6], and Digital Video Broadcasting-Satellite-Second Generation (DVB-S2) [7].

The locally optimal, yet the most complex, iterative decoding algorithm of LDPC codes is the "Belief Propagation" (BP), which is also known as "sum-product" [8]-[10]. Prior literatures have demonstrated that LDPC codes with BP decoding provide performance very close to the Shannon limit. Taking [8] as an example, for binary-input AWGN channels, simulation results show that the designed codes using a block length of 10^7 have a threshold that is within 0.0045 dB of the Shannon limit at the bit error rate (BER) of 10^{-6}. Although BP decoding provides excellent performance, it is too complex for hardware implementation. A better tradeoff between hardware complexity and error correction performance can be achieved with another well known decoding algorithm called "Min-Sum" (MS) [11]-[13]. With the aid of linear post processing of check node message approximation such as normalization and additive offset, the degradation caused by MS decoding can be greatly compensated [14], [15].

Although significant development on both construction and implementation of LDPC codes has occurred in the past several years, the efficient realization of high speed LDPC decoders still remains a challenge for ever-increasing high speed applications. Prior literatures have demonstrated that quasi-cyclic (QC) LDPC codes can be a preferred candidate when code lengths are short [16]. But for large code lengths, significant increase in hardware complexity turns to be quite a problem to tackle [17]. In order to deal with this problem, Sha *et al.* proposed a new class of implementation-oriented LDPC codes, namely *shift*-LDPC codes in [18]. It is shown that *shift*-LDPC codes can perform as well as randomly generated codes. And the *shift*-LDPC decoder with multi-Gb/s throughput can be implemented at very low hardware cost.

Manuscript received August 19, 2008. Revised on March 11, 2009. This work was supported by the National 863 Program of China under Grant No.2008AA01Z135, and the National Nature Science Foundation of China under Grant No. 60876017. This paper has been presented in part at the IEEE Asia Pacific Conference on Circuits and Systems, Macau, China, December 2008.

C. Zhang, J. Sha, L. Li, and J. Lin are with the Institute of VLSI Design, Jiangsu Provincial Key Laboratory of Advanced Photonic and Electronic Materials, National Laboratory of Solid State Microstructures and Department of Physics, Nanjing University, Nanjing 210093, People's Republic of China (e-mail: zhangchuan01@yahoo.com; shajin@nju.edu.cn; lili@nju.edu.cn; njuphylinjun@yahoo.com).

Z. Wang is with Broadcom Corporation, 5300 California Avenue, Irvine, CA 92617 USA (e-mail: zfwang@broadcom.com).

Recently, the implementation of decoders satisfying a class of LDPC codes with different code parameters (e.g., code lengths and code rates) has become a popular issue that attracts attentions of researchers, because in certain applications such as wireless communication, the code parameters should hold great flexibility to adapt to varying channel conditions. Furthermore, flexible LDPC decoders capable of meeting various service requirements and interference conditions can also achieve as high hardware efficiency as specific ones. Prior literatures have proposed several flexible LDPC decoder architectures, of which the most notable ones are by Mansour and Shanbhag [19], Masera *et al.* [20], Zhang *et al.* [21], and Lee and Ryu [22]. Mansour and Shanbhag in [19] designed a programmable decoder chip for 2048-bit architecture-aware (AA-) LDPC codes. Employing the turbo-decoding message-passing (TDMP) algorithm, the decoder could achieve a throughput of 640 Mb/s, which is not sufficient for modern high speed communication systems (e.g., 802.15). Masera *et al.* in [20] proposed an implementation of flexible LDPC decoder which could be tailored to decode both IEEE 802.11n and IEEE 802.16e LDPC codes. The decoder implemented the *low-traffic* BP (LTBP) algorithm which achieved a remarkable interconnection reduction between nodes. The Benes [23] networks were used to establish non-blocking connections among processing elements. Although the routing complexity was low, this architecture is proved to be not suitable for high speed applications either. Zhang *et al.* in [21] presented a decoder architecture for multi-rate QC-LDPC codes in broadband broadcasting systems (i.e., China's DTTB). The targeting throughput is 26 Mb/s. And the modified Min-Sum algorithm (MMSA) was employed. Lee and Ryu in [22] presented a flexible LDPC decoding architecture proposed to support multiple code rates and code lengths achieving high throughput. The decoder occupies an area of 16.3 mm^2 and can run at the clock frequency of 212 MHz resulting in 1 Gb/s decoding throughput. However, due to the inherent characteristics of the employed broadcasting interconnection, further improvement of hardware efficiency is prohibited.

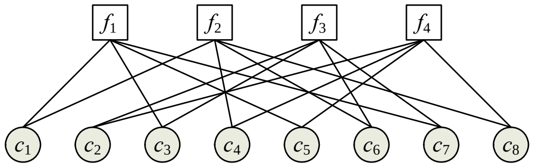
This work presents a flexible LDPC decoder architecture, which aims to have features of flexibility and high-throughput while using very low complexity hardware. The decoder implements the offset MS decoding algorithm for *shift*-LDPC codes which own the merits of memory efficiency, high parallelization, and low routing complexity. The Benes network is employed to realize the interconnection networks, which can be easily programmed to support multiple shift-LDPC codes with different code rates and code lengths. Moreover, non-uniform quantization schemes are explored to reduce the memory requirement. Based on the proposed design techniques, an (8192, 7168) (4, 32) *shift*-LDPC decoder with flexible code rate and code length is implemented to demonstrate the merits of the proposed techniques. It is shown that about 40% hardware reduction can be achieved compared with the state-of-the-art design for high speed flexible LDPC decoding. With SMIC 0.18 μm CMOS technology, 3.6 Gb/s decoding throughput can be obtained at 16 decoding iterations.

The remainder of this paper is organized as follows. A brief review of *shift*-LDPC codes is provided in Section II. In Section III, an efficient decoding approach and simulation results are presented. The design of the high-speed flexible *shift*-LDPC decoder is presented in Section IV. The implementation results and comparisons with other references are presented in Section V. Finally, Section VI concludes the paper.

II. Review of Shift-LDPC Codes

A. Construction of Shift-LDPC Codes

LDPC codes are a set of linear block codes corresponding to the $(n\text{-}k)\times n$ parity check matrix $\mathbf{H}$, which has very low density of 1's. With the aid of the bipartite graph called Tanner graph (TG), LDPC codes can be effectively represented [24]. There are two kinds of nodes in TG, *variable nodes* (*v-nodes*) and *check nodes* (*c-nodes*). And check node f_i is connected to variable node c_j only when the element h_{ij} of $\mathbf{H}$ is a 1. Fig.1 shows the parity check matrix $\mathbf{H}$ and its corresponding TG of an (8, 4)-regular LDPC code.



(a)

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

(b)

Figure 1: An LDPC code example. (a) Tanner graph. (b) Parity check matrix.

Shift-LDPC codes are a subclass of LDPC codes proposed by Sha *et al.* [18], which proves to be a satisfying approach for very high throughput applications. The parity check matrix defining an (n, k) (j, l)-regular *shift*-LDPC code of length $n = p\times l$ can be expressed as

$$\mathbf{H}=\begin{bmatrix}\mathbf{H}_{1,1}&\mathbf{PH}_{1,1}&\cdots&\mathbf{P}^{l-1}\mathbf{H}_{1,1}\\\mathbf{H}_{2,1}&\mathbf{PH}_{2,1}&\cdots&\mathbf{P}^{l-1}\mathbf{H}_{2,1}\\\cdots&\cdots&\ddots&\cdots\\\mathbf{H}_{j,1}&\mathbf{PH}_{j,1}&\cdots&\mathbf{P}^{l-1}\mathbf{H}_{j,1}\end{bmatrix}. \quad (1)$$

The leftmost j sub-matrices are $p\times p$ random permutation matrices, where there is exactly one entry 1 in each row and each column and 0's elsewhere. The matrix $\mathbf{P}$ demotes a $p\times p$ permutation matrix obtained by cyclically shifting the columns of the identity matrix $\mathbf{I}$ to the right by single step.

B. Performance for Shift-LDPC Codes

Experimental simulation results show that *shift*-LDPC codes can achieve as excellent performance as randomly generated LDPC codes of similar code lengths and code rates for the AWGN channels [18]. On the other hand, Cui and Wang have shown that any QC-LDPC codes can be converted to an extended *shift*-LDPC code [25], where each sub-matrix has at most two 1's per row (and per column) while shift structure remains unchanged.

C. Decoding Schedule for Shift-LDPC Codes

The message passing schedule for decoding an (n, k) (j, l)-regular *shift*-LDPC code is illustrated in Fig. 2.

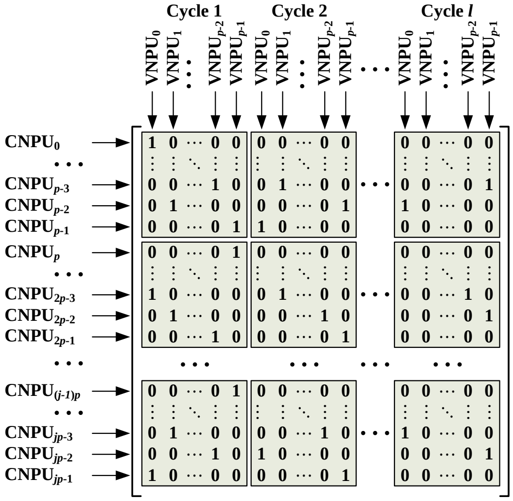


Figure 2: Message passing schedule for a regular *shift*-LDPC code.

One *check node processing unit* (CNPU) and one *variable node processing unit* (VNPU) complete the message updating associated to one row and one column of matrix **H** in one clock cycle, respectively. The messages corresponding to all p×j rows are processed in parallel. And totally, p VNPUs perform message updating in parallel. Hence, l clock cycles are required to complete both the row message updating and column message updating in one decoding iteration. The major advantage with *shift*-LDPC decoding is that the interconnection between two kinds of processing nodes is very simple [18], which directly leads to high clock speed and low hardware complexity.

D. Flexible Shift-LDPC Codes

In order to support the requirement for multiple code rates and code lengths and achieve the best tradeoff among decoding performance, hardware complexity, and throughput, fully flexible *shift*-LDPC decoders are in pressing demand.

The flexible (n, k) (j, l)-regular *shift*-LDPC codes have several major characteristics:

1) The block size of sub-matrices can change between 1 to p;
2) The row number of sub-matrices can change between 1 to j;
3) The column number of sub-matrices can be any integer number if overall memory requirement is satisfied.

III. Decoding Algorithms of Flexible Shift-LDPC Codes

In order to implement a flexible *shift*-LDPC decoder which is suitable for multiple codes with different code rates and code lengths efficiently, the hardware complexity becomes the major issue which should be taken into consideration. Especially when the supported code lengths are large, the memory can even dominate the whole area consumption of the decoder. For the sake of high memory efficiency of flexible *shift*-LDPC decoders, two versions of decoding schemes — single minimum decoding and non-uniform quantization scheme are well explored in this work. Based on extensive decoding performance simulation, the comparisons between different decoding schemes are shown and the optimal decoding algorithm is discussed.

A. Single Minimum Scheme

The Single Minimum Min-Sum Algorithm

```
Initialization: $L(q_{ij}) = y_i,\ i = 0,1,\ldots,N-1;$
for $k = 0$ step $1$ until $k_{\max}$ or $\hat{c}\mathbf{H}^T = 0$ do
  begin
    Step. 1 $\beta_{\min} = \min_{i\in V_j} \beta_{ij};$
        if $\tilde{N} = 1$
            $\beta'_{ij} = \begin{cases} \beta_{\min} + \tilde{\beta}(if\ \beta_{ij} = \beta_{\min}) \\ \tilde{\alpha}\times\beta_{\min} \end{cases};$
        else $\beta'_{ij} = \tilde{\alpha}\times\beta_{\min};$
        $L(r_{ji}) = \prod_{i'\in V_j\backslash i} \alpha_{i'j}\cdot\beta'_{ij};$
    Step. 2 $L(q_{ij}) = L(c_i) + \sum_{j'\in C_i\backslash j} L(r_{j'i});$
    Step. 3 $L(Q_i) = L(c_i) + \sum_{j\in C_i} L(r_{ji});$
    Step. 4 $\hat{c}_i = \begin{cases} 1, & L(Q_i) < 0 \\ 0, & L(Q_i) \geq 0 \end{cases};$
  end
Output: decoded bit $\hat{c}_i$
```

Although MS algorithm can achieve comparable decoding performance to SP algorithm, much less implementation complexity is required [19]. In scenarios of decoding large-length *shift*-LDPC codes, the hardware consumption caused by the computation and storage of both the minimum magnitude and the 2nd minimum magnitude still cannot be neglected. To further reduce hardware complexity, several approaches have been proposed [26]-[28], among which is the single minimum scheme [26]. Requiring only one minimum magnitude, single minimum scheme can help achieve significant hardware reduction with slight performance loss. And the decoding algorithm with this scheme can be summarized in the above four major steps, where $\tilde{N}$ defines the number of the minimum values, with the scaling factor $\tilde{\alpha} = 0.75$ and the offset factor $\tilde{\beta} = 0.125$.

B. Non-Uniform Quantization Scheme

When decoding *shift*-LDPC codes, soft messages passing in either direction of edges should be stored in memories. In general, outstanding decoding performance can be attained only when the code length is reasonably large (e.g., $N \geq 1000$ bits). Thus, memory requirement is significantly high for the implementation of *shift*-LDPC decoders. Since the routing complexity as well as the memory requirement is linearly proportional to the word-length of soft messages. Using less number of quantization bits for each soft message can be an effective approach to further reduce the implementation complexity [29]. However, simply reducing the word length of soft messages usually leads to significant performance degradation. In this section, an optimized non-uniform quantization scheme is proposed with 4-bits per soft message to achieve comparable decoding performance to that using 6-bit per message in the prior design [18].

TABLE I. PROPOSED 4-BIT NON-UNIFORM QUANTIZATION

Data in 5-bit uniform quantization	Corresponding data in 4-bit non-uniform quantization	Data representation in 4-bit non-uniform quantization
s0000	s0.0000	s000
s0001	s0.0001	s001
s0010	s0.0010	s010
s0011	s0.0011	s011
s0100	s0.0100	s100
s0101	s0.0101	s101
s0110	s0.0101	s101
s0111	s0.0101	s101
s1000	s0.1000	s110
s1001	s0.1000	s110
s1010	s0.1000	s110
s1011	s0.1000	s110
s1100	s0.1100	s111
s1101	s0.1100	s111
s1110	s0.1100	s111
s1111	s0.1100	s111

Note: s denotes the sign bit of a soft message.

Since CNPUs find only the minimum magnitudes, the 4-bit non-uniformly quantized messages can be directly used in CNPUs without conversion. However, in order to make the variable-to-check message $L(q_{ij})$ more precise, both the Exp (expansion) block and Comp (compression) block are employed in VNPUs to convert 4-bit quantized non-uniformly messages to 5-bit uniformly quantized messages before computation and backward after computation. The detailed conversion between two different formats is shown in Table I.

C. Simulation Results and Comparisons

An (8192, 7168) (4, 32)-regular *shift*-LDPC code example is considered in this work. Fig. 3 illustrates the bit error rate (BER) performance comparisons of the discussed decoding schemes for an AWGN channel with various E_b/N_0 and $I_{\max}=50$, where $I_{\max}$ is the maximum allowed decoding iterations. Note that the normal Min-Sum algorithm with 4-bit non-uniform quantization can achieve the similar decoding performance as the normal Min-Sum algorithm with (6:3) uniform quantization. The BER of the single minimum MS algorithm employing 4-bit non-uniform quantization suffers only 0.03 dB performance loss, compared with that of the normal Min-Sum algorithm with (6:3) uniform quantization.

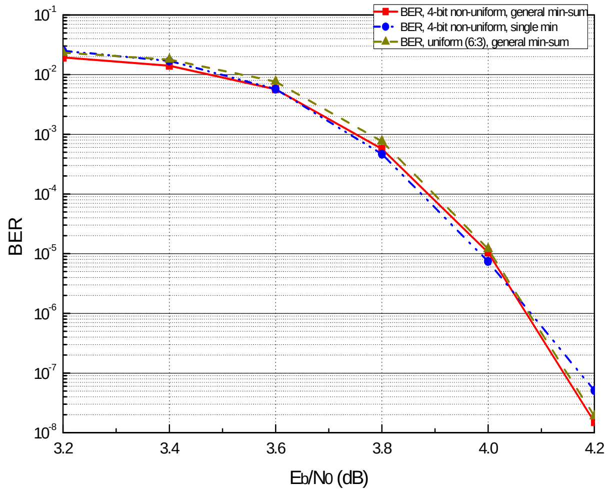


Figure 3: BER comparisons between different decoding algorithms for the (8192, 7168) (4, 32) shift-LDPC code.

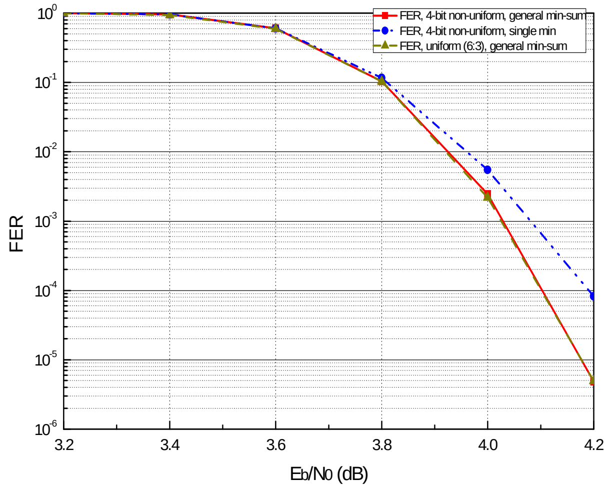


Figure 4: FER comparisons between different decoding algorithms for the (8192, 7168) (4, 32) shift-LDPC code.

Fig. 4 displays the frame error rate (FER) performance according to the algorithms presented in Fig. 3. From Fig. 4, we can observe that the performance gap between the normal Min-Sum algorithm with 4-bit non-uniform quantization and the normal Min-Sum algorithm with (6:3) uniform quantization is quite small. Also it can be found that at the FER of 10^{-4}, the single minimum MS algorithm employing 4-bit non-uniform quantization, with a significant complexity reduction, is only less than 0.1 dB away from the normal Min-Sum algorithm with (6:3) uniform quantization These observations confirm that the proposed decoding algorithms offer effective trade-off between the performance of iterative decoding and the corresponding complexity.

IV. FLEXIBLE SHIFT-LDPC DECODER ARCHITECTURE

Based on the proposed modified MS algorithm, this section discusses components of a flexible *shift*-LDPC decoder architecture, which is suitable for decoding multiple *shift*-LDPC codes with different code rates as well as different code lengths. In comparison to prior architectures presented in

[19]-[22], our design has the following unique characteristics:

1) High throughput. Owing to the intrinsic parallelism characteristics of *shift*-LDPC codes, the proposed decoder architecture can easily achieve multi-Gb/s throughput that satisfies very high speed communication systems;
2) Low hardware complexity. Adopting the modified MS algorithm with both single minimum decoding and non-uniform quantization scheme, the proposed architecture proves to be hardware efficient.

A. Overall Decoder Architecture

Fig. 5 shows the proposed flexible *shift*-LDPC decoder architecture, which is mainly composed of an array of *p* VNPUs, an array of *p*×*j* CNPUs, a CNPU local shuffle network, and a configurable global permutation network.

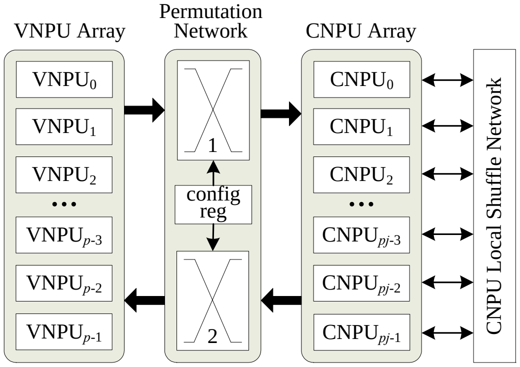


Figure 5: Overview of the proposed flexible *shift*-LDPC decoder architecture.

Each block of the decoder specifies a particular function being executed. The array of *p* VNPUs and the array of *p*×*j* CNPUs compute the variable-to-check messages and check-to-variable messages, respectively. The configurable global permutation network between the VNPU array and CNPU array, which is dynamically configurable, can accommodate different code rates, various code lengths, and distinct sub-matrix sizes. The CNPU local shuffle network helps reduce the interconnection complexity significantly.

B. CNPU local shuffle network

Thanks to the intrinsic structured features of *shift*-LDPC codes, considerable reduction of interconnection complexity can be easily attained by employing the CNPU local shuffle network. According to the decoding schedule mentioned above, *p* VNPUs execute concurrently in one clock cycle and totally *l* clock cycles are needed to complete one decoding iteration. As shown in the following, when decoding an (*n*, *k*) (*j*, *l*)-regular *shift*-LDPC code, the row process intermediate messages are transmitted via CNPU local shuffle network.

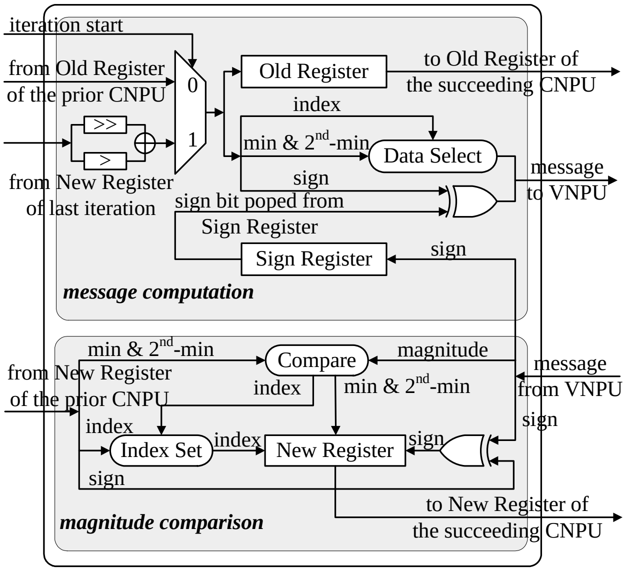
```
Scheduling with CNPU Local Shuffle Network
Iteration Start
if in clock cycle $m,\ m = 2, 3, ..., l$
  pass intermediate result from CNPU$_i$ to CNPU$_{(i+1)\bmod p}$
else if in clock cycle $m,\ m = 1$
  pass intermediate result from CNPU$_{(i+l-1)\bmod p}$ to CNPU$_i$
End Iteration
```

Consequently the established global permutation network between *p* VNPUs and *p*×*j* CNPUs can be reused in each clock cycle and therefore reduces the interconnection complexity.

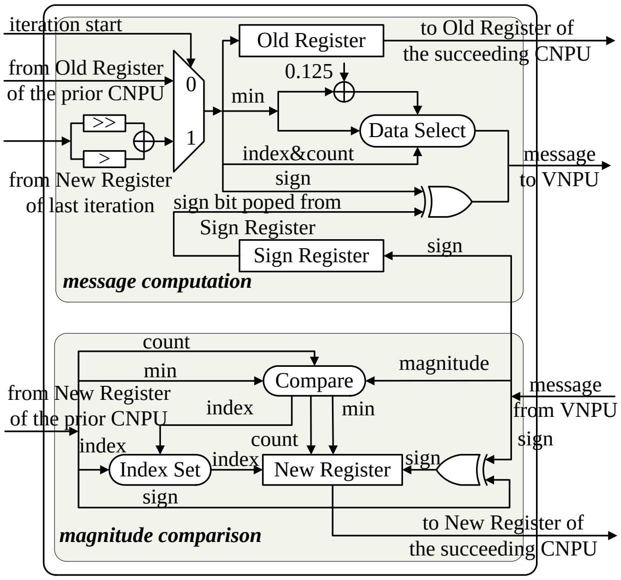
C. Architecture of Check Node Processing Unit

The *check node processing unit* (CNPU) executes the check-to-variable message computation and the magnitude comparison between the variable-to-check and the intermediate result of row process. To implement the CNPU efficiently, both the single minimum scheme and the 4-bit non-uniform quantization scheme are used in our design. Fig. 6(a) shows the architecture of CNPU using 4-bit non-uniform quantization scheme. And the architecture of CNPU employing both single minimum and 4-bit non-uniform quantization schemes is shown in Fig. 6(b).

The old register, new register, and sign register store the row process results of last iteration, the intermediate results of row process, and the sign bits of the check-to-variable messages respectively. Due to the adoption of single minimum scheme and non-uniform quantization scheme, the routing complexity and memory usage of CNPU will be significantly decreased.



(a)



(b)

Figure 6: Architecture of check node processing unit. (a) CNPU using 4-bit non-uniform quantization scheme. (b) CNPU using both single minimum and 4-bit non-uniform quantization schemes.

The sequence operations in each decoding iteration can be scheduled as shown in Fig. 7 and explained as follows. At the beginning of every iteration (Clock Cycle 1), $p\times j$ CNPUs are fed with $p\times j$ variable-to-check messages from p VNPUs, each CNPU with one message, respectively. After the check node processing, $p\times j$ check-to-variable messages are passed to p VNPUs. And the variable node updating computation is finished finally. All the above operations are done in the first clock cycle. During the check node process, the row process result of last iteration is transferred from the New Register of $\mathrm{CNPU}_{(i+\mathrm{l}-1)\mathrm{mod}p}$ to the Old Register of CNPU_i, while the New Register of CNPU_i begins to store the row process intermediate result of the present iteration. However, in Clock Cycle m (m=2,3,…,l), the the Old Register of CNPU_i stores contains the same results as the Old Register of $\mathrm{CNPU}_{(i+\mathrm{l}-1)\mathrm{mod}p}$ and the New Register of CNPU_i is updated by the row process intermediate result of the present iteration continuously.

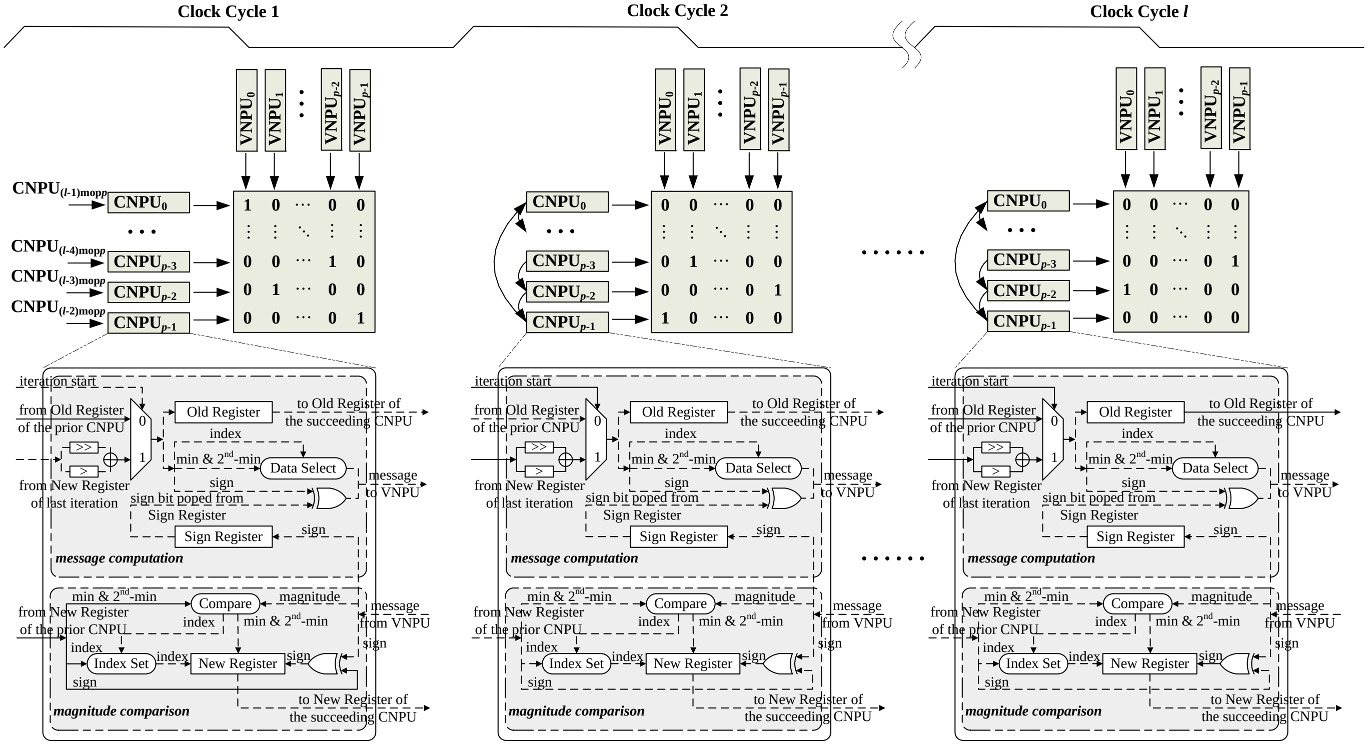


Figure 7: Decoding schedule within one iteration.

D. Architecture of Variable Node Processing Unit

Fig. 8 illustrates the architecture of VNPU which adopts the same structure as that in [30]. In order to employ the non-uniform quantization scheme, the Exp block and the Comp block are introduced in VNPU.

As shown in Fig. 9, the two blocks can be easily derived based on the non-uniform quantization scheme. And the corresponding equations used to perform the expansion are expressed in (2).

$$\begin{cases} OUT[4]=IN[3] \\ OUT[3]=IN[2]IN[1] \\ OUT[2]=IN[2]\overline{IN[1]}+IN[2]IN[0]\,. \\ OUT[1]=\overline{IN[2]}IN[1] \\ OUT[0]=\overline{IN[2]}IN[0]+\overline{IN[1]}IN[0] \end{cases} \qquad (2)$$

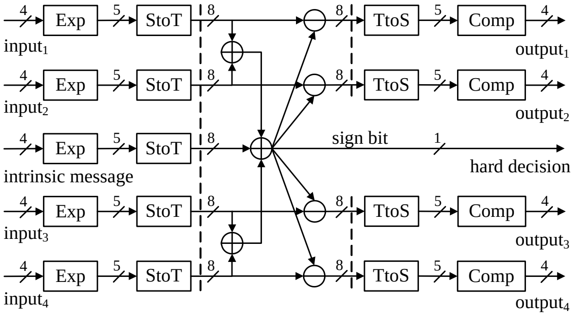


Figure 8: Architecture of variable node processing unit.

Similarly, the conversion formula for compression can be obtained as follows:

$$\begin{cases} OUT[3] = IN[4] \\ OUT[2] = IN[3] + IN[2] \\ OUT[1] = \overline{IN[2]}IN[1] + IN[3] \\ OUT[0] = \overline{IN[3]}IN[0] + IN[3]IN[2] + IN[2]IN[1] \end{cases} . \quad (3)$$

The overhead of this part of logic is small compared with the significant hardware reduction brought by the transformed Min-Sum algorithm. In this design, two stages of pipelining are added to VNPU to shorten the critical path.

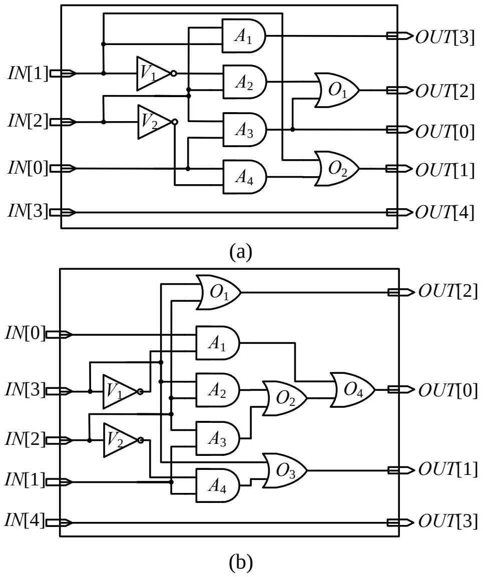


Figure 9: (a) Architecture of the Exp block. (b) Architecture of the Comp block.

E. Configurable Global Permutation Network

The configurable global permutation network implements the flexible interconnections between the CNPUs and VNPUs. Among the candidate structures for flexible permutation network are the Benes network and Banyan network. An N×N Benes network has $2\log_2 N$-1 stages, each of which contains $N/2$ 2×2 crossbar switches, and totally $N\log_2 N$- $N/2$ crossbar switches are in need. Compared with the N×N Banyan network, the N×N Benes network can support any number of inputs that is smaller than N, no matter whether it is a power of 2 or not. Thus, the Benes network is preferable for flexible application even though it consumes more crossbar switches than the Banyan network with the same number of inputs.

As shown in Fig. 10, the configurable global permutation network comprises two sub-networks and one configurable register. Sub-network 1 and Sub-network 2 shuffle data of variable-to-check messages and check-to-variable messages, respectively. And each sub-network comprises four 256×256 Benes networks. And the configurable register stores the control information using 7680 control bits.

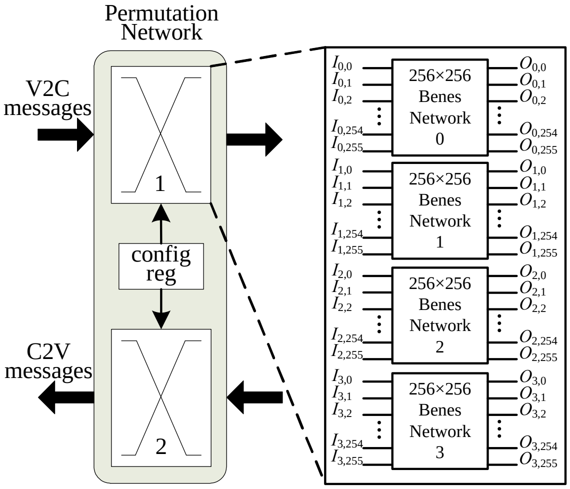

Permutation Network
V2C messages
C2V messages
1
2
config reg
256×256 Benes Network 0
256×256 Benes Network 1
256×256 Benes Network 2
256×256 Benes Network 3
$I_{0,0}$ $I_{0,1}$ $I_{0,2}$ $I_{0,254}$ $I_{0,255}$
$O_{0,0}$ $O_{0,1}$ $O_{0,2}$ $O_{0,254}$ $O_{0,255}$
$I_{1,0}$ $I_{1,1}$ $I_{1,2}$ $I_{1,254}$ $I_{1,255}$
$O_{1,0}$ $O_{1,1}$ $O_{1,2}$ $O_{1,254}$ $O_{1,255}$
$I_{2,0}$ $I_{2,1}$ $I_{2,2}$ $I_{2,254}$ $I_{2,255}$
$O_{2,0}$ $O_{2,1}$ $O_{2,2}$ $O_{2,254}$ $O_{2,255}$
$I_{3,0}$ $I_{3,1}$ $I_{3,2}$ $I_{3,254}$ $I_{3,255}$
$O_{3,0}$ $O_{3,1}$ $O_{3,2}$ $O_{3,254}$ $O_{3,255}$

Figure 10: Configurable global permutation network.

With the aid of the configurable global permutation network, the proposed *shift*-LDPC decoder provides enough flexibility to support any kind of *shift*-LDPC code, whose sub-matrices are smaller than the input size of Benes networks.

In general, the decoding throughput of a *shift*-LDPC decoder can be well estimated as follows:

$$\text{Throughput} = \frac{k \times f}{T \times I_{avg}}, \quad (4)$$

where f is the clock frequency determined by the critical path, T is the number of clock cycles required in one decoding iteration, and I_{avg} is the average number of iterations to process one code word. Since the critical path of the configurable global permutation network is quite long, it is necessary to make it well pipelined to attain higher decoding throughput. The 256×256 Benes networks employed have 15 stages of crossbars, which can be easily split into 3 parts. Each part equally has 5 stages of crossbars as shown in Fig. 11.

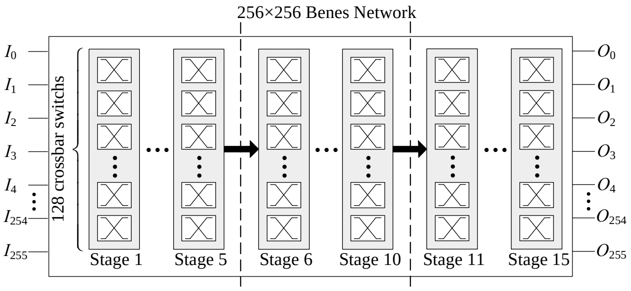

256×256 Benes Network
I_0 I_1 I_2 I_3 I_4 I_{254} I_{255}
O_0 O_1 O_2 O_3 O_4 O_{254} O_{255}
128 crossbar switchs
Stage 1
Stage 5
Stage 6
Stage 10
Stage 11
Stage 15

Figure 11: Pipelining scheme for the configurable global permutation network.

TABLE II. ASIC IMPLEMENTATION RESULTS AND COMPARISONS

Reference	Proposed I*	Proposed II**	[17]	[19]	[20]	[22]
Code Rate	$\frac{7}{8}$	$\frac{7}{8}$	$\frac{8}{9}$	$\frac{8}{16}:\frac{1}{16}:\frac{14}{16}$	$\frac{259}{512}$	$\frac{5}{6}$
Flexibility	Yes	Yes	No	Yes	Yes	Yes
Technology	180 nm, 1.8 V	180 nm, 1.8V	65 nm, 0.9 V	180 nm, 1.8 V	130 nm, —	180 nm, —
Quantization	4-bit	4-bit	4-bit	—	8-bit	—
Algorithm	Normalized Min-Sum	Single Min Min-Sum	Transformed Min-Sum	TDMP	LTBP	BP
Frequency	290 MHz	290 MHz	300 MHz	125 MHz	300 MHz	212 MHz
Throughput	3.6 Gb/s	3.6 Gb/s	2.1 Gb/s	640 Mb/s	31.2 Mb/s	745 Mb/s
Iteration	16	16	16	10	8	15
Code Length	8192	8192	9216	2048	1024	64800
Area	13.9 mm^2	13.1 mm^2	2.32 mm^2	14.3 mm^2	2.94 mm^2	16.3 mm^2
Area Scaled to 65 nm	1.81 mm^2	1.70 mm^2	2.32 mm^2	1.86 mm^2	0.74 mm^2	2.12 mm^2
TAR	1989 Mb·s^{-1}·mm^{-2}	2118 Mb·s^{-1}·mm^{-2}	905 Mb·s^{-1}·mm^{-2}	344 Mb·s^{-1}·mm^{-2}	42 Mb·s^{-1}·mm^{-2}	351 Mb·s^{-1}·mm^{-2}

* Proposed I algorithm employs 4-bit non-uniform quantization scheme.

** Proposed II algorithm employs single minimum and 4-bit non-uniform quantization schemes.

V. ASIC IMPLEMENTATION RESULTS

The proposed flexible decoders for the (8192, 7168) (4, 32)-regular *shift*-LDPC code are implemented in SMIC 0.18 μm Logic 1P6M Salicide 1.8/3.3 V process. The hierarchical design flow is followed with standard EDA tools: Cadence Verilog XL is used for simulation and verification, Synopsys Design Compiler is used for synthesis, Synopsys Prime Time is used for timing analysis, and Cadence SoC Encounter is used for floor planning, place, and route. The final layout plot of the design with both the single minimum and 4-bit non-uniform quantization schemes is shown in Fig. 12. And its operation is configurable according to three parameters: the block size of sub-matrices, the row number of sub-matrices, and the column number of sub-matrices.

Table II lists the ASIC implementation results of the decoders and comparisons with other prior references. The maximum number of iterations is set to 16 to offer a good tradeoff between decoding performance and throughput. And in order to perform a fair comparison of the hardware performance, the area consumption of the proposed designs is roughly scaled to 65 nm process.

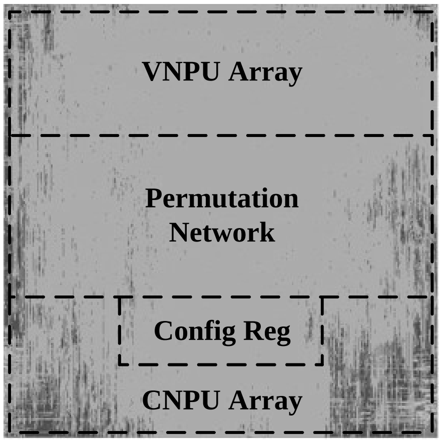


Figure 12: Final layout of the flexible shift-LDPC decoder with both the single minimum and 4-bit non-uniform quantization schemes.

It can be observed that the proposed design with 4-bit non-uniform quantization occupies 13.9 mm^2 and saves 2.81% silicon area compared with [19]. And the corresponding decoding throughput is more than one order of magnitude higher than [19]. Furthermore, the proposed design which employs both the single minimum and 4-bit non-uniform quantization schemes saves about 8.44% area compared to [19] while having small penalty in performance loss. The significant hardware reduction of the second approach is attributed to the savings of the 2nd -minimum. As summarized in Table II, although the hardware complexity of our designs is two times larger than [20], more than 185 times higher throughput can be achieved. Furthermore, compared with the latest published flexible LDPC decoder in [22], our efficient flexible decoder architecture for *shift*-LDPC codes results in 19.6% hardware complexity reduction due to adoption of the single minimum and 4-bit non-uniform quantization schemes, and it has about 2 orders of magnitude higher throughput by incorporating shifting structure. Even compared with [17], which is also suitable for very high decoding throughput but fails to provide flexibility of decoding, the proposed designs possess advantages in both throughput and area. In the Proposed I decoder, thanks to our non-uniform quantization approach, only 78.0% hardware resources are used. And the corresponding throughput is 71.4% higher than that of [17]. Moreover, Proposed II decoder achieves 26.7% area reduction and 71.4% throughput increase compared with [17]. It can also be expected that our designs can achieve higher clock speed if we really use 65 nm CMOS process and thus even higher throughput can be expected. Therefore we conclude that the proposed designs are well suited for very high speed communication systems.

To make the comparisons simple, one parameter introduced by [20] is employed in this work. It is named throughput-to-area ratio (TAR) and is computed as follows:

$$\mathrm{TAR} = \frac{\mathrm{Throughput}}{\mathrm{Area}}\,. \tag{5}$$

It is clear from Table II that the proposed designs are also much more efficient than any published flexible LDPC decoders according to this performance metric. Thus they are the most competitive design up to now and well suited for very high speed applications.

VI. Conclusion

A flexible *shift*-LDPC decoder architecture for very high speed communication systems has been proposed. Both the single minimum scheme and the non-uniform quantization scheme are exploited to reduce the hardware consumption while maintaining similar decoding performance. The Benes network is employed to implement the configurable interconnections between the VNPU array and CNPU array which brings sufficient flexibility for multiple code rates, code lengths as well as sub-matrices sizes. Two (8192, 7168) (4, 32)-regular *shift*-LDPC decoder design examples are implemented in SMIC 0.18 μm Logic 1P6M Salicide 1.8/3.3 V process, which can achieve a maximum throughput of 3.6 Gb/s at 16 iterations with die size of only 13.9 mm^2 and 13.1 mm^2, respectively. By adopting more advanced technology and better optimized pipelining schemes, significantly higher clock speed and thus even higher throughput can be expected.

VII. Acknowledgment

The authors gratefully acknowledge support from the National 863 Program of China under Grant No.2008AA01Z135, and the National Nature Science Foundation of China under Grant No. 60876017.

References

[1] R. G. Gallager, *Low Density Parity Check Codes.* Cambridge, MA: MIT Press, 1963.

[2] D. J. C. MacKay and R. M. Neal, “Near Shannon limit performance of low density parity-check codes,” *Electron. Lett*, vol. 32, no. 18, pp. 1645-1646, Aug. 1996.

[3] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Trans.Inf. Theory*, vol. 45, no. 2, pp. 399-432, Mar. 1999.

[4] *IEEE Draft Standard for Information Technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 4: Enhancements for Higher Throughput,* IEEE Standard P802.11n, 2008.

[5] *IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1,* IEEE Standard 802.16e, 2008.

[6] *Framing structure, Channel coding and modulation for digital television terrestrial broadcasting system,* R. P. China Standard GB20600-2006, 2006.

[7] *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz satellite services,* European Standard EN 300 421, 2005.

[8] S.-Y. Chung, G. D. Jr. Forney, T. J. Richardson, and R. E. Urbanke, “On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit,” *IEEE Commun. Lett*, vol. 5, no. 2, pp. 58-60, Feb 2001.

[9] N. Wiberg, “Codes and decoding on general graphs,” Ph.D. dissertation, Dept. Elect. Eng., Linköping Univ., Linköping, Sweden, 1996.

[10] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Francisco, CA: Morgan Kaufmann, 1988.

[11] J. Chen and M. Fossorier, “Density evolution for two improved BP-based decoding algorithms of LDPC codes,” *IEEE Commun. Lett*, vol. 6, pp. 208-210, May 2002.

[12] E. Eleftheriou, T. Mittelholzer and A.Dholakia, “Reduced complexity decoding algorithm for low-density parity-check codes,” *IEE Electron. Lett*, vol. 37, pp. 102-104, Jan. 2001.

[13] J. Chen and M.P.C. Fossorier, “Decoding low-density parity-check codes with normalized APP-based algorithm,” in *Proc. IEEE Globecom*, TX, Nov. 2001, pp. 1026-1030.

[14] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X.-Y. Hu, “Reduced-complexity decoding of LDPC codes,” *IEEE Trans. Commun.*, vol. 53, pp. 1288-1299, Aug. 2005.

[15] N. Pandya and B. Honary, “Low-complexity decoding of LDPC codes,” *IEE Electron Lett*, vol. 43, pp. 990-991, Aug. 2007.

[16] H. Zhong and T. Zhang, “Block-LDPC: A practical LDPC coding system design approach,” *IEEE Trans. Circuits. Syst. I*, vol. 52, no. 4, pp. 766-775, Apr. 2005.

[17] H. Zhong, W. Xu, N. Xie, and T. Zhang, “Area-efficient min-sum decoder design for high-rate quasi-cyclic low-density parity-check codes in magnetic recording,” *IEEE Trans. Magn.*, vol. 43, no. 12, pp. 4117-4122, Dec. 2007.

[18] J. Sha, Z, Wang, M. Gao, and L. Li, “Multi-Gb/s LDPC code design and implementation,” *IEEE Trans.Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 2, pp. 262-268, Feb. 2009.

[19] M. M. Mansour and N. R. Shanbhag, “A 640-Mb/s 2048-bit programmable LDPC decoder ship,” *IEEE J. Solid-State Circuits*, vol. 41, no. 3, pp. 684-698, Mar. 2006.

[20] G. Masera, F. Quaglio, and F. Vacca, “Implementation of a flexible LDPC decoder,” *IEEE Trans. Circuits. Syst. II*, vol. 54, no. 6, pp. 542-546, June 2007.

[21] L. Zhang, L. Gui, Y. Xu, and W. Zhang, “Configurable multi-rate decoder architecture for QC-LDPC codes based broadband broadcasting system,” *IEEE Trans. Braodcasting*, vol. 54, no. 2, pp. 226-235, June 2008.

[22] J.-Y. Lee and H.-J. Ryu, “A 1-Gb/s flexible LDPC decoder supporting multiple code rates and block lengths,” *IEEE Trans. Consumer Electronics*, vol. 54, no. 2, pp. 417-424, May 2008.

[23] V. E. Benes, “Optimal rearrangeable multistage connecting networks,” *Bell Syst. Tech. J.*, no. 43, pp. 1641-1656, 1964.

[24] R. M. Tanner, “A recursive approach to low complexity codes,” *IEEE Trans.Inf. Theory*, vol. IT-27, no. 5, pp. 533-547, Sep. 1981.

[25] Z. Cui and Z. Wang, “Efficient decoder design for high-throughput LDPC decoding,” in *Proc. Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp. 1640-1643, Dec. 2008.

[26] A. Darabiha, A. C. Carusone, and F. R. Kschischang, “A bit-serial approximate min-sum LDPC decoder and FPGA implementation,” in *Proc. IEEE Symp. Circuits and Systems (ISCAS)*, pp. 149-152, May 2006.

[27] Z. Cui and Z. Wang, “Efficient message passing architecture for high throughput LDPC decoder,” in *Proc. IEEE Symp. Circuits and Systems (ISCAS)*, pp. 917-920, May 2007.

[28] Y. Chen and K. K. Parhi, “Overlapped message passing for quasi-cyclic low-density parity check codes,” *IEEE Trans. Circuits. Syst. I*, vol. 51, no. 6, pp. 1106-1113, Jun. 2004.

[29] Daesun Oh and K. K. Parhi, “Nonuniformly quantized min-sum decoder architecture for low-density parity-check codes,” in *Proc. the 18th ACM Great Lakes symposium on VLSI (GLSVLSI)*, pp. 451-456, May 2008.

[30] Z. Wang and Z. Cui, “Low-complexity high-speed decoder design for quasi-cyclic LDPC codes,” *IEEE Trans.Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 1, pp. 104-114, Jan. 2007.