A Fully Parallel LDPC Decoder Architecture Using Probabilistic Min-Sum Algorithm for High-Throughput Applications

Chung-Chao Cheng, Jeng-Da Yang, Huang-Chang Lee, Chia-Hsiang Yang, and Yeong-Luh Ueng

***Abstract*—This paper presents a normalized probabilistic min-sum algorithm for low-density parity-check (LDPC) codes, where a probabilistic second minimum value, instead of the true second minimum value, is used to facilitate fully parallel decoder realization. The comparators in each check-node unit (CNU) are connected through an interconnect network based on a mix of tree and butterfly networks such that the routing and message passing between the variable-node units (VNUs) and CNUs can be efficiently realized. In order to further reduce the hardware complexity, the normalization operation is realized in the VNU rather than in the CNU. An early termination scheme is proposed in order to prevent unnecessary energy dissipation for both low and high signal-to-noise-ratio regions. The proposed techniques are demonstrated by implementing a (2048, 1723) LDPC decoder using a 90 nm CMOS process. Post-layout simulation results show that the decoder supports a throughput of 45.42 Gbps at 199.6 MHz, achieving the highest throughput and throughput-to-area ratio among comparable works based on a similar or better error performance.**

***Index Terms*—High-throughput decoder, low-density parity-check (LDPC) codes, min-sum algorithm.**

I. Introduction

Low-density parity-check (LDPC) codes [1], [2] have received increased attention due to their excellent error-correction capabilities. An LDPC code can be decoded iteratively using either the sum-product algorithm [2] or the hardware-friendly normalized min-sum algorithm (NMSA) presented in [3] based on its Tanner graph, which is a bipartite graph consisting of variable nodes (VNs) and check nodes (CNs). In [1] and [2], two-phase message passing, which divides the decoding operations in a single iteration into the VN-operation and the CN-operation phases, was used.

Nowadays, LDPC codes have been adopted in many standards, and a wide range of decoders have been implemented for both wireless and wired communications [4]–[7]. For wired communications, high-throughput decoders for high-rate LDPC codes are required. For example, the rate-0.84 regular (2048, 1723) LDPC code constructed based on a Reed-Solomon (RS) code [8], is adopted in the IEEE 802.3an standard (10GBASE-T) [9], where the throughput requirement is 6.67 Gbps. For next-generation Ethernet and optical communications, a throughput of more than 10 Gbps is desired.

In [10], the authors showed that the high check-node degree for high-rate LDPC codes leads to greater complexities in hardware, interconnects, and timing, which are difficult to manage using a fully parallel architecture without degradation in error performance. Consequently, partially parallel architectures, where either VNs or CNs are partitioned into several groups and each group is processed sequentially, were used in [10]–[13]. Although partially parallel decoders can maintain an adequate error-rate performance, these decoders can only achieve a maximum throughput of about 10 Gbps.

To meet throughput requirements in excess of 10 Gbps, non-programmable fully parallel architectures are required [14]. To reduce the number of interconnects and, hence, increase utilization, the Split-Row Threshold Algorithm (SRTA) presented in [15] and [16] can be used, where the parity-check matrix (variable nodes) is divided into several block columns (groups), and the local minimum in each group is compared with a predefined threshold value. The authors in [15] demonstrated that their decoder for the (2048, 1723) RS-LDPC code, where the variable-node and check-node degrees are 6 and 32, respectively, can achieve a utilization of 97% using 16 partitions. However, this decoder suffers from a loss in error performance of 0.2–0.3 dB compared to the conventional NMSA [15].

The complex interconnects in a fully parallel decoder can also be reduced using a bit-serial architecture [17]. Moreover, it was shown in [18] that serial data processing in a bit-serial decoder can be deeply pipelined using digit-online arithmetic. The stochastic decoding method presented in [19] and [20] can also be used to avoid complex interconnects, where messages are conveyed in Bernoulli sequences with a sense that the probability of observing a “1” in a stream is equal to the original probability. It was reported in [19] that the (2048, 1723) bit-serial decoder occupies a small area. However, in order to achieve a throughput of more than 10 Gbps, the stochastic decoder should be operated at high signal-to-noise ratios (SNRs) and a high clock frequency of 750 MHz. In addition, stochastic decoding will introduce significant latency compared to the bit-parallel approach.

In this paper, a fully parallel bit-parallel architecture is proposed in order to achieve the desired throughput for next-generation high-throughput applications. The resultant difficulties are overcome by utilizing the following techniques. A Normalized Probabilistic Min-Sum Algorithm (NPMSA), where a probabilistic second minimum value, instead of the true second minimum value, is proposed in order to simplify the CN operation.

Manuscript received September 12, 2013; revised December 19, 2013, January 30, 2014, and February 22, 2014; accepted March 12, 2014. Date of publication April 07, 2014; date of current version August 26, 2014. This paper was recommended by Associate Editor B.-H. Gwee.

C.-C. Cheng, J.-D. Yang and H.-C. Lee are with the Department of Electrical Engineering, National Tsing Hua University, Hsinchu, Taiwan (e-mail: s9961818@m99.nthu.tw; dad77911dad@yahoo.com.tw; akula.lee@gmail.com).

C.-H. Yang is with the Department of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan (e-mail: chy@nctu.edu.tw).

Y.-L. Ueng is with the Department of Electrical Engineering and the Institute of Communications Engineering, National Tsing Hua University, Hsinchu, Taiwan (e-mail: ylueng@ee.nthu.edu.tw).

Digital Object Identifier 10.1109/TCSI.2014.2312479

To minimize the core area and the wire length, the comparators in each CN unit (CNU) are connected through an interconnect network based on a mix of tree and butterfly networks. The resultant CNU is partitioned into several sub-units, where the local minimum value for each partition is first established via a comparison tree, and then the global minimum values are determined by comparing the local minimum value with the minimum values from its neighboring partitions. Accordingly, the VN units (VNUs) are partitioned such that the routing and message passing between the VNUs and the CNUs can be efficiently realized. In order to further reduce the hardware complexity, the normalization process is carefully executed in the VNU rather than in the CNU. An early termination scheme is proposed so as to avoid unnecessary energy dissipation for both the low and high SNR regions. The proposed techniques are demonstrated by implementing a fully parallel (2048, 1723) RS-LDPC decoder using a 90 nm CMOS process. Post-layout simulation results show that this decoder achieves the highest throughput and throughput-to-area ratio (TAR) among comparable works.

The remainder of this paper is organized as follows. Section II describes the proposed NPMSA. Section III presents the proposed decoder architecture, including the early-termination scheme. Chip implementation and comparison results are detailed in Section IV. Section V concludes the paper.

II. LDPC Codes and Decoding Algorithms

A. Review of LDPC Codes Constructed Based on RS Codes

The (2048, 1723) RS-LDPC code used in this paper is constructed based on the (64, 2) extended RS code [8]. The RS-LDPC code is regular and the dimensions of its parity-check matrix (PCM) are 384 × 2048, where the column weight and row weight are 6 and 32, respectively. The PCM can be divided into $6 \cdot 32$ submatrices, where the dimensions of each submatrix are 64 × 64. In addition, each submatrix is a permutation matrix. Since the rank of this PCM is $2048 - 1723 = 325$, the PCM is not full rank and there are 59 dependent rows in the matrix. There are 384 CNs and 2048 VNs in the corresponding Tanner graph.

B. Normalized Min-Sum Algorithm [3]

Let $Q_{ji}[k]$, which is produced during the kth iteration, denote the variable-to-check (V2C) message from VN j to CN i. Similarly, the check-to-variable (C2V) message from CN i to VN j, which is produced during the kth iteration, is denoted as $R_{ij}[k]$. Let $I_C[j]$ denote the set of CNs connected to VN j. Similarly, $I_R[i]$ denotes the set of VNs (bit nodes) connected to CN i. For regular LDPC codes, each CN has the same degree d_c, and each VN has the same degree d_v. In other words, for regular LDPC codes, $d_c = |I_R[i]|$ and $d_v = |I_C[j]|$ for each CN i and each VN j, respectively. At the kth iteration, the CN and VN operations are described as follows:

VN operations: For each VN j, compute $Q_{ji}[k]$ corresponding to each of its CN neighbors i, i.e., $i \in I_C[j]$, according to $Q_{ji}[k] = \lambda_j + \sum_{i' \in I_C[j] \backslash \{i\}} R_{i'j}[k-1]$, where λ_j is the channel (reliability) value of VN j.

CN operations: For each check node i, compute $R_{ij}[k]$ corresponding to each of its VN neighbors j, i.e., $j \in I_R[i]$, according to

$$R_{ij}[k] = S_{ij}[k] \times \alpha \times \min_{j' \in I_R[i] \backslash \{j\}} |Q_{j'i}[k]| \tag{1}$$

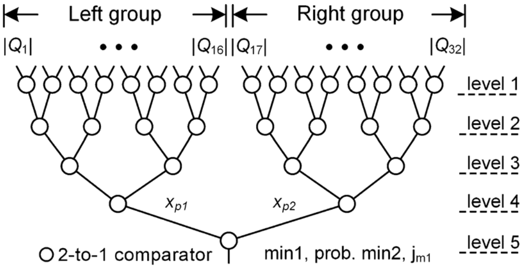


Fig. 1. Tree-structure-based MVF for a CNU with 32 inputs.

where $S_{ij}[k] = \prod_{j' \in I_R[i] \backslash \{j\}} sgn(Q_{j'i}[k])$ and α is a normalization factor. In the last iteration, i.e., $k = N_{it}$, a hard decision for each VN j is made based on the sign of the *a posterior* probability (APP) Λ_j, which is given by $\Lambda_j[N_{it}] = \lambda_j + \sum_{i \in I_C[j]} R_{ij}[N_{it}]$.

C. Normalized Probabilistic Min-Sum Algorithm

For high-rate LDPC codes, where the check-node degree is usually large, and is significantly larger than the variable-node degree, the computational complexity of the decoding is dominated by the CN operation. Take the regular (2048, 1723) code as an example, where $d_c = 32$ and $d_v = 6$, it can be seen that the CN operation is much more complicated than the VN operation. Consequently, efficient techniques were devised to overcome this challenge.

In order to produce the d_c C2V messages for check node i, it can be seen from (1) that the first minimum value, $\min 1_i$ and the second minimum value, $\min 2_i$, must be determined among d_c V2C messages. A module called the minimum-value finder (MVF) can be used to determine both of these minimum values together with the index value for $\min 1_i$. These minimum values and the index are then used by the d_c C2V-message calculators to calculate the d_c C2V messages.

Specifically, the first minimum value $\min 1_i$ is defined as $\min 1_i = \min_{j' \in I_R[i]} |Q_{j'i}[k]|$, and the corresponding index is denoted as $j_{m1,i}$. The second minimum value $\min 2_i$ is defined as $\min 2_i = \min_{j' \in I_R[i] \backslash \{j_{m1,i}\}} |Q_{j'i}[k]|$. Establishing the $\min 1_i$ value can be accomplished through $|I_R[i]| - 1$ comparisons. After the $\min 1_i$ value is determined, additional $|I_R[i]| - 2$ comparisons are required in order to ascertain the $\min 2_i$ value. In [21], an efficient tree-structure based algorithm (the TS approach) was proposed to realize the MVF. The TS approach uses connection units to simultaneously determine the first two minimum values and, hence, the delay of the resultant MVF is only half compared to that of the conventional sorting-based method. As a result, a high-speed MVF can be devised using the TS approach. This TS approach requires $(2^{s+1} - 3)$ comparisons in order to determine the $\min 1_i$ and $\min 2_i$ values from 2^s inputs. In this paper, a tree-structure-based MVF is proposed, where connection units are not used to determine the second minimum value. Instead, the last competitor of the first minimum is treated as the second minimum. As a result, the number of comparators can be further reduced by half by removing the connection units, while still retaining the reduced delay when compared to [21].

The proposed MVF is detailed as follows. As shown in Fig. 1, the $|I_R[i]| - 1$ comparisons are arranged based on an s-level tree structure, where $s = \lceil \log_2 |I_R[i]| \rceil$, in order to determine the $\min 1_i$ value, where $|I_R[i]| = d_c = 32$ and $s = 5$. In this tree, two groups of size-2^{s-1} leaves (inputs), called the left and right groups, are at the top of the tree. If the two inputs of the last-level

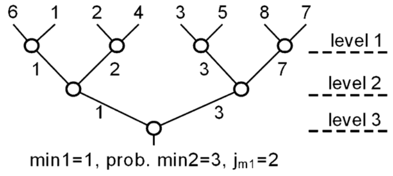


Fig. 2. Tree-structure-based MVF for a CNU with 8 inputs.

TABLE I
PROBABILITY DISTRIBUTION OF x_{p2} (THE PROBABILISTIC $\min 2$)

$min2$	$min3$	$min4$	$min5$	other
51%	26%	13%	6%	4%

comparison are denoted as x_{p1} and x_{p2}, then either x_{p1} or x_{p2} is the desired $\min 1_i$ value. Consider the case where x_{p1} is the desired $\min 1_i$ value, which is also the minimum value of the left group shown in Fig. 1. Although the other input x_{p2} may not be the correct $\min 2_i$ value, it is the minimum value among the other group of size 2^{s-1} (right group values). In other words, x_{p2} (the last competitor of the first minimum) is at least the $(2^{s-1}+1)$th minimum value among the 2^s inputs. If the $\min 2_i$ value is generated using this method, the resultant second minimum value, i.e., x_{p2}, is called the probabilistic $\min 2_i$ value. In other words, the last competitor of the first minimum in a tree-structure-based MVF is taken as the probabilistic second minimum. Fig. 2 shows an example that demonstrates how to obtain the probabilistic second minimum for 8 inputs.

If we divide the index set $I_R[i]$ into two subsets $I_{R,1}[i]$ and $I_{R,2}[i]$, where $|I_{R,1}[i]| = \lceil |I_R[i]|/2 \rceil$ and $|I_{R,2}[i]| = |I_R[i]| - |I_{R,1}[i]|$, then the probabilistic $\min 2_i$ can be mathematically written as

$$\mathrm{Prob.}\min 2_i = \max\left\{\min_{j' \in I_{R,1}[i]} |Q_{j'i}[k]|, \min_{j' \in I_{R,2}[i]} |Q_{j'i}[k]|\right\} \tag{2}$$

If we assume the occurrence of the second minimum is uniformly distributed over the inputs of the MVF, the probability that x_{p2} (the probabilistic $\min 2_i$) is the correct second minimum is $2^{s-1}/(2^s-1)$. For our application, simulation results show that the probability is 51% for $s=5$, i.e., $d_c = 32$, which confirms the derivation. The probability of x_{p2} being one of the second to fifth minimum values for $d_c = 32$ is 96%, as shown in Table I.

Although the value of the probabilistic $\min 2_i$ is different from the value of the $\min 2_i$, with a probability of about 50%, it will be used to update the C2V message that will be passed to only a single neighboring VN. In addition, an appropriate normalization factor α can be used to compensate for the effect of the optimistic probabilistic $\min 2_i$ value in order to enhance the error-rate performance. Accordingly, (1) can be modified as follows:

$$R_{ij}[k] = S_{ij}[k] \times \alpha \times \begin{cases} \min 1_i, & \text{if } j \neq j_{m1,i} \\ \mathrm{Prob.}\min 2_i, & \text{if } j = j_{m1,i} \end{cases}. \tag{3}$$

The resultant algorithm is called the NPMSA. The optimal normalization factor α for it is related to the CN degree d_c of the LDPC code considered and can be determined using either the BER simulation or the density evolution described in [3].

Fig. 3 shows the floating-point bit-error-rate (BER) simulation results for the (2048, 1723) code using different algorithms. It can be seen that the use of the NPMSA introduces a performance degradation of only 0.05 dB compared to the NMSA based on a BER of 10^{-6} and $N_{it} = 11$. However, when $N_{it} = 100$, the NPMSA achieves almost the same error-rate performance compared to the NMSA. In contrast, the use of the SRTA [15] induces a performance loss of about 0.2 dB compared to the NMSA. It can also be seen that the error performance for the NPMSA when $N_{it} = 7$ is similar to that of the SRTA using $N_{it} = 11$.

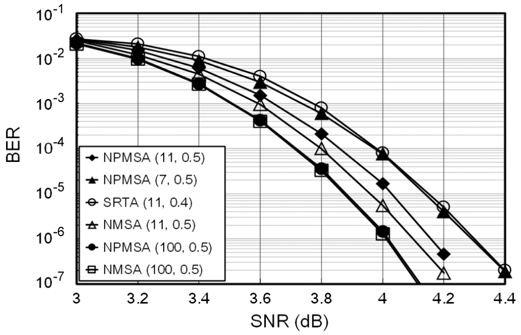


Fig. 3. BER versus SNR for the (2048, 1723) code. The numbers in the parentheses denote (N_{it}, α). 16 partitions are used in the SRTA [15].

TABLE II
COMPARISON OF CN OPERATIONS

Decoding Algorithm	NMSA [3]	SRTA [15]	NPMSA
Minimum-value Finder	[21]	[15]	Proposed
2-input Comparisons	23424	18432	11904 (13824‡)
Error-rate Loss♯	0 dB	0.2 dB	0.05 dB

‡ This value is calculated based on an ASIC implementation, where the comparators in each CNU are connected through an interconnect network based on a mix of tree and butterfly networks with 4 partitions.

♯ Based on $N_{it} = 11$ and BER $= 10^{-6}$.

Table II shows a comparison of the complexity of CN operations executed using the NMSA, the SRTA, and the proposed NPMSA for the (2048, 1723) code, where the complexity is measured based on the number of 2-input comparisons. It is worth noting that there are 2048 VNs and 384 CNs in the decoding Tanner graph. For the NMSA, the efficient MVF presented in [21] is used, since the first two minimum values can be found correctly using this approach. Compared to the NMSA combined with the MVF proposed in [21], the proposed NPMSA can achieve a 49.1% reduction in the number of comparisons with a 0.05 dB loss in error-rate performance from the algorithmic perspective. Compared to the SRTA, the proposed NPMSA can achieve a 35.4% reduction in the number of comparisons and a 0.15 dB enhancement in error-rate performance from the algorithmic perspective. Although the proposed ASIC implementation will increase the complexity of the CN operations, the reduction in complexity achieved by using the proposed NPMSA is still significant compared to both the NMSA and the SRTA.

The reason that the NPMSA achieves a better error performance compared to that of the SRTA, and a similar performance compared to that of the NMSA for the (2048, 1723) code, is described in the following. In the NPMSA, one of the d_c C2V messages is generated using the probabilistic $\min 2$ value, and the other $d_c - 1$ C2V messages are generated using the correct $\min 1$ value. Since $d_c = 32$ for the (2048, 1723) code, the majority of the C2V messages (about 97%) are generated using the correct $\min 1$ value. Hence, the NPMSA can achieve a similar error-rate performance compared to that of the NMSA, although the probability of selecting the correct second minimum

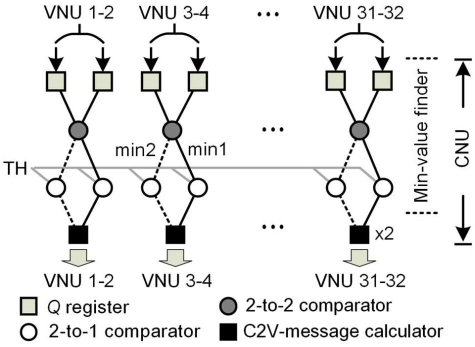


Fig. 4. Architecture of the CNU for the SRTA-based decoder with 16 partitions presented in [15].

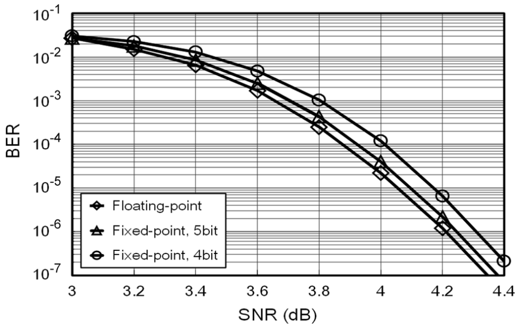


Fig. 5. BER versus SNR for the (2048, 1723) code using NPMSA and different quantization schemes, where $N_{it} = 9$.

is around 50% assuming uniform distribution. In the SRTA, either the threshold value, the correct $\min 1$ value, or the correct $\min 2$ value, are used to generate C2V messages. However, if the number of partitions is large, e.g., 16 partitions, there is no guarantee that the C2V messages are generated using the correct $\min 1$ and $\min 2$ values with a high probability. Hence, a performance degradation is observed for the SRTA, although it is also modified from the NMSA.

III. Hardware Architecture

Although the proposed NPMSA can reduce the computational complexity of the CNUs with a minimal error-rate loss, the hardware architecture must still be designed carefully in order to minimize the area, delay, and energy. Several techniques are proposed for achieving these goals.

A. Reduction in Routing Complexity

It was reported in [15] that a 5-bit fully parallel NMSA-based (2048, 1723) decoder can only achieve a utilization of 38%. The reason for the low utilization is that the two minimum values and the corresponding index for each check node should be generated and propagated throughout the entire decoder, resulting in complex interconnects. To reduce the routing complexity of an LDPC decoder, the SRTA, which can be viewed as an approximation of the NMSA, was used in [15] and [16]. In the SRTA, the parity-check matrix is divided into several sub-matrices in a column-wise form such that a CNU is partitioned into several sub-units, where only local minimum values for each sub-matrix, rather than the global minimum values for the entire matrix are found and compared with a predetermined threshold TH, as shown in Fig. 4. Using this method, the global-wire interconnects can be significantly reduced and a utilization of 97% can be achieved for the 5-bit decoder presented in [15]. However, the required first minimum value, $\min 1$, may not be the correct one since the decoding messages are not fully exchanged. As a result, this decoder suffers from an error performance loss of 0.2 dB compared to the conventional NMSA, as shown in Fig. 3.

Compared to the NMSA, the NPMSA has a lower computational complexity and, hence, has a potential to be used for devising decoders with lower routing complexity. However, in order to achieve a compatible utilization compared to the SRTA-based decoder presented in [15], using the NPMSA is not sufficient. Consequently, two techniques are used in the proposed fully parallel NPMSA-based decoder:

Reduction in quantization bits: Fewer quantization bits are explored in order to reduce hardware complexity in the NPMSA-based decoder. It was reported in [15] that the quantization loss for the 5-bit SRTA-based decoder is 0.1 dB. As shown in Fig. 5, the wordlength of the channel value in the proposed NPMSA-based decoder can be further reduced to 4 bits with the same error performance loss of 0.1 dB. Although a smaller number of quantization bits is used, the proposed 4-bit decoder still provides a better error-rate performance compared to the 5-bit SRTA-based decoder. The utilization of this directly-mapped 4-bit NPMSA-based decoder is 75%.

Mixed interconnect network: A mixed interconnect network is proposed in order to reduce routing complexity and improve utilization. To implement the MVF of a single d_c-input CNU, the tree structure shown in Fig. 1 is commonly used. Although this structure can minimize the usage of comparators, it is no longer optimal when considering the routing complexity in a fully parallel LDPC decoder. The tree comparison structure results in a tree interconnect topology for the MVF of a CNU. This means that a MVF receives d_c V2C messages from d_c neighboring VNUs and passes the intermediate results through comparators that are connected based on a tree interconnect topology. The resultant $\min 1$ and probabilistic $\min 2$ values produced in the last-level comparator are then used by the d_c C2V-message calculators to calculate the d_c C2V messages, which are then fed back to the original d_c VNUs for use in the operations in the next iteration. The drawback of the pure tree-interconnect structure is that the d_c C2V-message calculators must be located in the same vicinity to receive the minimum values produced from the last-level comparator. This will result in long global wires from the d_c C2V-message calculators to the d_c neighboring VNUs, as shown in Fig. 6(a). For high-rate LDPC codes, which usually have a larger d_c value, the associated global interconnects is complicated, resulting in routing congestion.

In order to reduce the number of global wires, a comparator can be added at the last level of the MVF such that a butterfly interconnect can be formed between the last two levels of the MVF, as shown in Fig. 6(b). Using this architecture, it is not necessary to locate the d_c C2V-message calculators in the same vicinity since there are two sources that can produce the desired minimum values. Consequently, the d_c C2V-message calculators can be divided into two groups, and the C2V-message calculators within each group are closely located, as shown in Fig. 6(b). Accordingly, the d_c VNUs can also be divided into two groups. Using such an arrangement, the VNUs belonging to the same group can receive the C2V messages from its corresponding group of C2V-message calculators. This provides a better initial condition for placement and routing in the physical

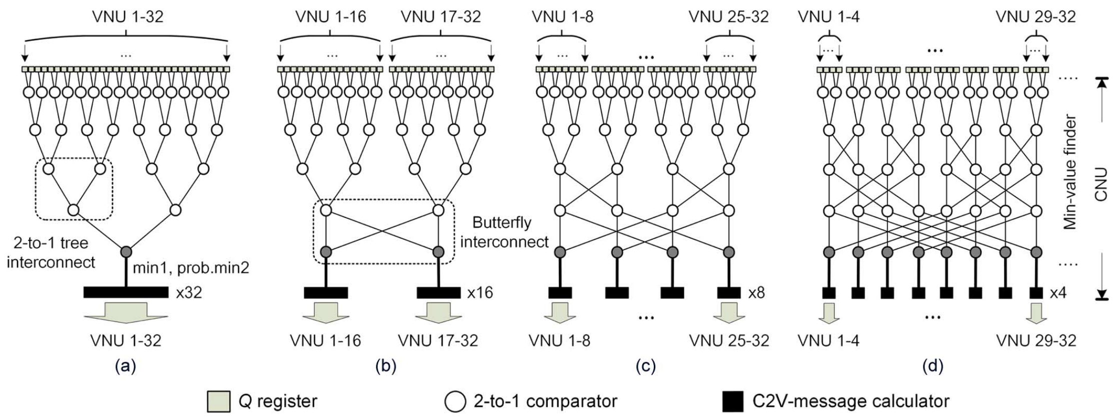


Fig. 6. Architecture for the proposed CNU at different partition levels (P).

TABLE III
COMPLEXITY COMPARISON FOR A SINGLE 32-INPUT MVF

Partitions	# of 2-to-1 Tree Interconnects	# of Butterfly Interconnects	# of Comparators
1	15	0	31
2	14	1	32
4	12	4	36
8	8	12	48

design stage, but at the cost of increased number of comparators. An interconnect network with a mix of tree and butterfly networks is therefore used to determine an optimized structure that considers both area and wire length.

Fig. 6 shows the interconnect topologies for a single CNU at different partition levels (P). The number of 2-to-1 tree interconnects, butterfly interconnects, and comparators for different partition schemes are listed in Table III. It can be seen that the number of comparators increases as the partition level increases. For the NPMSA, the number of 2-input comparisons used in the 384 CNUs is 11904, 12288, 13824, and 18432, when using 1, 2, 4, and 8 partitions, respectively. The number of 2-input comparisons required when using $\mathrm{P} = 4$ is also included in Table II. It is worth noting that the number of 2-input comparisons required when using $\mathrm{P} = 1$ is identical to that calculated from the algorithmic perspective shown in Table II. It can be seen that the hardware implementation of the NPMSA still requires fewer comparators compared to the NMSA, and the same number of comparators compared to the SRTA, even when $\mathrm{P} = 8$ is used for the proposed mixed network. By comparing Fig. 6 with Fig. 4, it can be seen that the proposed MVF can obtain more accurate minimum values. An advantage of this mixed interconnect network is that information exchanged between partitions is not lost so that there is no error performance degradation compared to the SRTA-based decoder.

The proposed mixed interconnect network also relaxes routing congestion and facilitates physical implementation. The hierarchical layout methodology, which is used in [15] to reduce the routing complexity introduced by a fully parallel architecture, is therefore not required in this work. In the hierarchical layout process, each block is first implemented independently as a macro, and then several blocks are integrated through global routing. Due to the tangled tree-structure interconnection, only a limited number of interconnections are allowed for message passing in order to reduce routing complexity. Hence, the authors in [15] proposed the SRTA to minimize the number of interconnects between blocks at the cost of a 0.2–0.3 dB performance loss. In this work, butterfly interconnections are used to distribute routing wires, thereby avoiding routing congestion without introducing any performance loss. By leveraging the mixed interconnect network, the global routing complexity is significantly reduced. The interconnections are described only at the RTL level. The mixed interconnect network also reduces the effort required to implement a fully parallel LDPC decoder using automatic CAD tools, and the layout design process is now the same as the standard digital design flow. That is, the layout of each block does not need to be first created independently. Instead, the complete design is placed and routed through joint optimization across block boundaries, resulting in a more compact layout.

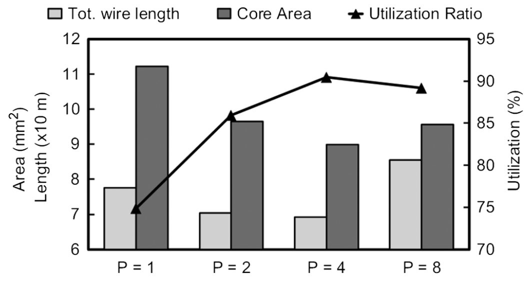


Fig. 7. Hardware cost evaluation for different partition levels (P).

To determine the optimal partitioning scheme, the total wire length and core area for partitions $\mathrm{P} = 1, 2, 4, 8$ are evaluated. As shown in Fig. 7, an optimal partition scheme exists for $\mathrm{P} = 4$ due to the trade-off among the global routing complexity, the local routing complexity, and the number of comparators. A similar trend is seen for the core area since it is directly affected by the routing complexity given the same timing constraint. Compared to the directly-mapped architecture $(\mathrm{P} = 1)$, the core area and wire length of the optimal architecture $(\mathrm{P} = 4)$ are reduced by 19.8% and 10.9%, respectively. Fig. 7 shows the corresponding chip utilization. $\mathrm{P} = 4$ also has the highest utilization (91%) because it has the shortest wire length and the smallest core area, resulting in the highest area efficiency.

B. Reduction in Hardware Complexity

Fig. 8 shows the proposed fully parallel LDPC decoder architecture, where $\mathrm{P} = 4$ is adopted. Since the dimensions of the PCM for the RS-LDPC code are 384 × 2048, this decoder consists of 384 CNUs and 2048 VNUs, where the V2C messages are stored in Q registers in the CNUs. The channel values are stored in λ registers in the VNUs, as shown in Fig. 9. It can be

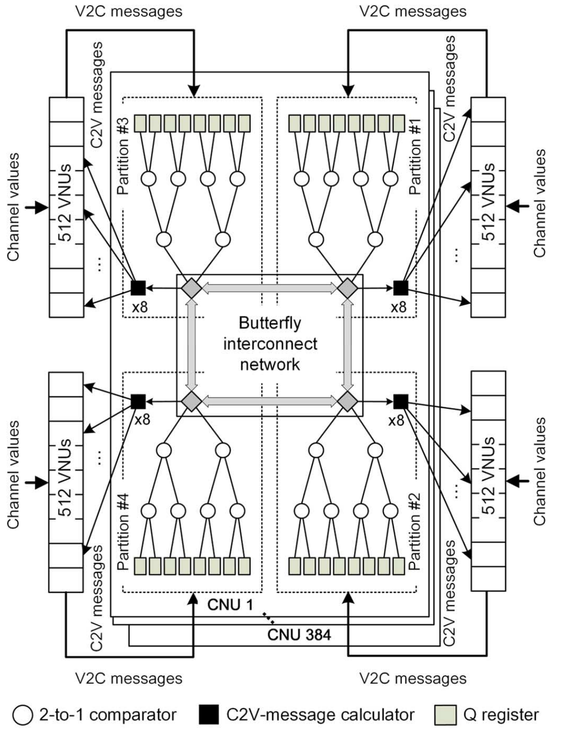


Fig. 8. Block diagram for the proposed LDPC decoder.

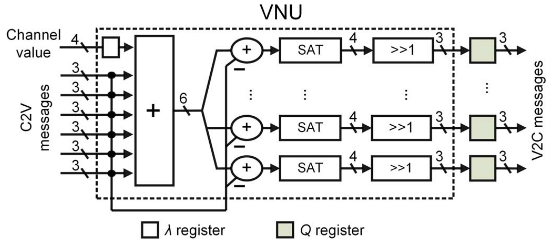


Fig. 9. Block diagram for the VNU.

seen from Fig. 8 that each CNU is partitioned into 4 sub-units, and butterfly interconnects are used to exchange messages between these partitions. In each partition, the local minimum is generated through the comparison tree. The local minimum is then compared with the minimum values from its neighboring partitions through the butterfly interconnects. The global $\min 1$, probabilistic $\min 2$, and the associated index are, hence, obtained. The C2V messages are then updated using these global minimum values and index. Using the updated C2V messages, the V2C messages can be updated accordingly. In the fully parallel implementation, all the C2V and V2C messages are updated in parallel during one cycle. As a result, N_{it} cycles are required to accomplish N_{it} decoding iterations.

Through simulation, it can be found that using 0.5 as the normalization factor α in the NPMSA achieves the best error-rate performance among the considered normalization factors 0.3, 0.4, 0.5, 0.6, and 0.7. As a result, the normalization factor used in hardware implementation was selected as 0.5. Based on this normalization factor, the decoder hardware can be further optimized through the techniques described below.

The first technique is that normalization (multiplication by 0.5) can be implemented using hard-wired shifting. Normally, the implementation complexity for the normalization would increase when moving the normalization logics from CNUs to VNUs, since the number of VNUs is greater than the number of CNUs. However, this is not the case in this work, since the scaling factor for the normalization was carefully chosen to be 0.5, so that the normalization can be implemented using hardwired shifting, introducing almost zero hardware overhead. This movement even provides two benefits for reducing hardware complexity. First, the wordlength of the messages exchanged between the VNUs and the CNUs can be reduced by 1 bit for all datapaths. As a result, the total number of wires and, hence, the routing complexity, can be reduced. Second, the wordlength of all the comparators in the CNUs can be reduced by 1 bit, yielding a more area-efficient implementation for the CNUs, where a 2-bit wordlength is used for comparators.

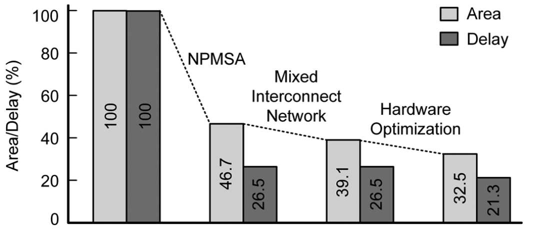


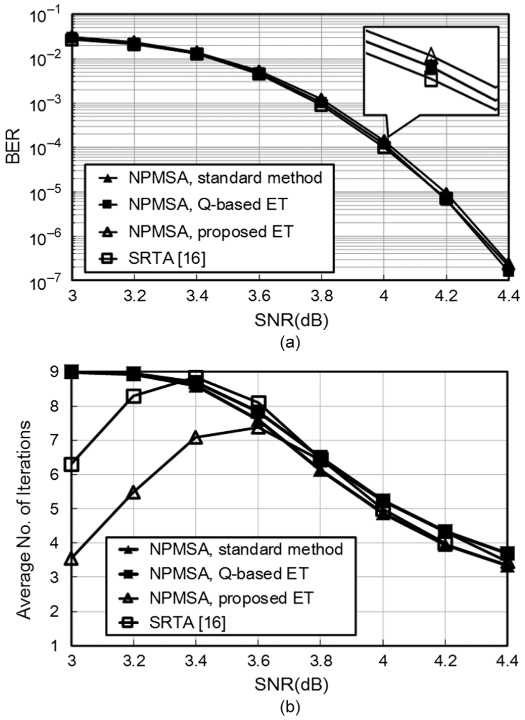
Fig. 10. Reduction in area and delay by applying NPMSA, a mixed interconnect network, and hardware optimization.

The second technique is that the 2-to-1 comparison can be efficiently realized via a look-up table (LUT) if a 3-bit wordlength is used in the CNU. A 3-bit datapath with sign-magnitude representation is used for the inputs of the CNU, where 1 bit is used for Sign and 2 bits are used for Magnitude. The Sign bit is processed by XOR gates, and the 2-bit Magnitude is processed by the comparators. Thus the wordlength used in the 2-to-1 comparator of the CNU is 2. The 2-bit Magnitude of the comparator output is determined via a look-up table. The “don’t-care” terms allow further logic simplification so as to reduce the hardware complexity.

Fig. 10 shows the cumulative reduction in both area and delay achieved after applying the proposed techniques, where the 5-bit fully parallel NMSA-based (2048, 1723) decoder reported in [15] is used as the baseline. In other words, the directly-mapped architecture without employing the NPMSA, the mixed interconnect network, and hardware optimization is used as the baseline design. The area and delay for this design are set to unity (100%) so as to illustrate the improvements in area and delay. Applying the NPMSA reduces the computational complexity of the decoding algorithm. An interconnect network based on a mix of tree and butterfly types used in the CNU simplifies the routing. Hardware optimization reduces the wordlength of the CNU and the wordlength of the messages between the VNUs and the CNUs. Logic synthesis estimates show that an overall reduction in area of 67.5% and an overall reduction in delay of 78.7% are achieved compared to the baseline design.

C. Reduction in Energy Consumption

At high SNR values, the majority of the received words (blocks) will become correct before the number of iterations reaches the maximum value N_{it}. Early termination can help avoid unnecessary decoding computations and, hence, reduce energy consumption. A decoded codeword is considered to be a valid codeword if all the checksum values for all the M check equations of the parity-check matrix (PCM) are 0, where M also denotes the number of CNs in the Tanner graph. As a result, the decoding process can be terminated if this condition is satisfied. Since the decoded codeword must be determined



(a)

(b)

Fig. 11. Performance evaluation for the (2048, 1723) code, where $N_{it} = 9$: (a) BER versus SNR and (b) average number of iterations versus SNR.

from the APP values, additional routing networks and XOR gates are required in order to obtain the M checksum values. Although this standard method can produce valid codewords if the received words are correctable, this method will increase the interconnect and gate complexity and introduce additional delay, especially for a decoder based on a fully parallel architecture.

In order to avoid these disadvantages, the authors in [17] proposed using the sign of the V2C messages rather than the APP values to determine the timing of the termination. Using this method, the value of $T_i[k] = \prod_{j' \in I_R[i]} sgn(Q_{j'i}[k])$ for each CN (check equation) i is calculated first. The decoding process is then terminated if the T_i values for all the M check equations of the PCM are zero. Since the T_i values are also calculated when the S_{ij} values are calculated, additional circuits are not required in order to obtain the T_i values. This Q-based method, which was implemented in the decoders presented in [15] and [17], only requires OR operations to be performed on the M values of T_i.

At low SNR values, an iterative decoder often cannot produce a valid codeword due to a high BER, even when the number of iterations reaches the maximum value N_{it}. The standard early-termination method and the methods presented in [15] and [17] are not efficient under this scenario. However, as is well-known, the instantaneous SNR of a channel varies with time. Hence, the channel SNR is occasionally low such that the BER is high. Under this scenario, early detection for an uncorrectable received word in the early decoding stages is desired in order to avoid unnecessary iterations and, hence, reduce energy dissipation [16], [22], [23]. The two methods presented in [22] and [23] track the fluctuation of the parity checksum at a certain number of iterations and compare its magnitude with threshold values which are SNR dependent. The method presented in [16] (an enhanced version of [15]) compares the parity checksum with predefined threshold values for three consecutive iterations after the number of iterations has reached a predetermined value.

TABLE IV
CHIP SUMMARY

Technology	90 nm CMOS		
Quantization (bits)	4		
N_{it}	9		
Chip/Core Area (mm^2)	12.67/9.6		
Gate Count[†]	3,428K		
Supply Voltage (V)	0.9		
f_{clk} (MHz)	199.6		
Latency (ns)	45.09		
Throughput (Gbps)	45.42		
Power (mW)	1,110		
SNR (dB)	3.0	3.6	4.4
Avg. Iter	3.55	7.38	3.45
Energy per Bit (pJ/bit)	9.6	20.04	9.4

[†] The number of equivalent NAND gates.

In this paper, an early-termination scheme is devised in order to reduce the number of decoding iterations for both the low and the high SNR regions. At the completion of each iteration, the checksum for each check node (parity-check equation) is first calculated by summing [over GF(2)] all the signs (hard decisions) of the APP values for the neighboring VNs. Then, $\rho[k]$, which denotes the summation of the checksum values for all the CNs at the kth iteration, is calculated. If $\rho[k] = 0$, it indicates that the decoder has produced a valid codeword and the decoding process can be terminated at the end of the kth iteration. Otherwise, the summation of the checksum values is compared with two predetermined thresholds TH_0 and TH_1 at two distinct iterations k and k', respectively, where $k' > k$. If $\rho[k] \geq TH_0$ and $\rho[k'] \geq TH_1$, then the decoding process can be terminated at the end of the k'th iteration. Details of the proposed scheme are summarized in Algorithm 1.

Algorithm 1 Proposed Early Termination Scheme

```
Set $N_{it}$, $Enable = 0$, $TH_0$ and $TH_1$
for $k = 1, \ldots, N_{it}$ do
    Execute CN and VN operations
    if $k <= N_{it}$ then
        if $\rho[k] = 0$ then
            Terminate decoding
        end if
        if $Enable = 0$ and $\rho[k] \geq TH_0$ then
            $Enable = 1$
        else if $Enable = 1$ and $\rho[k] \geq TH_1$ then
            Terminate decoding
        end if
    else
        Terminate decoding
    end if
    $k = k + 1$
end for
```

Obviously, the BER performance and the average number of iterations depend on the values of TH_0 and TH_1. The method for determining the threshold values basically follows the method used in [16]. We observe that for the majority of correctable received words, the $\rho[k]$ value decreases as the iteration number k increases and the probability that $\rho[k] \geq TH_0$ and $\rho[k'] \geq TH_1$, where $k' > k$, is low if TH_0 and TH_1 are large enough. Using several possible candidate values for TH_0 and TH_1, we performed simulation to obtain the corresponding BER and the average number of iterations. Then, the best

TABLE V
COMPARISON OF (2048, 1723) RS-LDPC DECODERS

	JSSC'08 [17]	ISCAS'11 [16]	TSP'10 [20]	TSP'11 [19]	JSSC'10 [13]	TCAS-I'12 [18]	This work
Decoding Algorithm	Bit-serial MSA	SRTA 16 Partitions	MTFM Stochastic	Delayed Stochastic	Bit-parallel Offset MSA	Digit-serial Offset MSA	Bit-parallel NPMSA
Technology	90 nm	65 nm	90 nm	90 nm	65 nm	65 nm	90 nm
Input Quantization (bit)	4	5	6	5	4	10	4
f_{clk} (MHz)	250	186	500	750	700	980	199.6
SNR @ BER = 10^{-7} (dB)	5	4.4	4.55	4.75	4.4	4.4	4.4
Iterations	8	15	400	600	8	8	9
Cycles per Iteration	4	1	1	1	12	12	1
Latency (ns)	128	80.65	800	800	137	97.96	45.09
Core Area (mm^2)	9.8	5.25	6.38	3.93	5.35	10.89	9.6
Throughput (Gbps)	16	25.40	2.56	2.56	14.9	20.9	45.42
Norm. Throughput in 90 nm	16	18.34	2.56	2.56	10.79	15.1	45.42
Norm. Area in 90 nm	9.8	10.07	6.38	3.93	10.26	20.88	9.6
Norm. Latency in 90 nm	128	111.66	800	800	189.9	135.64	45.09
Norm. TAR in 90 nm	1.63	1.82	0.4	0.65	1.05	0.72	4.73

values for TH_0 and TH_1 are 134 and 134, respectively. If we wish to efficiently determine the threshold values, we can let $TH_0 = TH_1$ such that the search space can be reduced. For the (2048, 1723) code considered in this paper, the best values for TH_0 and TH_1 can also be determined using the same threshold value.

Figs. 11(a) and 11(b) show the BER and the average number of iterations versus the SNR, respectively, for the (2048, 1723) code using the NPMSA and various early-termination (ET) schemes, where 4-bit fixed-point simulation is performed. As shown in these two figures, the proposed ET method introduces negligible difference in BER and achieves a reduction in the average number of iterations of 58% at an SNR of 3.0 dB compared to both the standard method and the Q-based method. The area overhead introduced by using the proposed method is only about 6.8% compared to that of the Q-based method. At an SNR of 3.0 dB, the proposed method achieves a reduction in energy of 58% compared to the Q-based method.

The major difference between the proposed method and the methods presented in [22] and [23] is that the thresholds used in the proposed method are independent of SNR, while the thresholds used in [22] and [23] are SNR dependent. Hence, the proposed methods are suitable for applications where SNR values are difficult to obtain. In contrast to the three thresholds that are used in the method presented in [16], two thresholds are used in the proposed method. In addition, comparison at consecutive iterations after the number of iterations has reached a predetermined value is not necessary in the proposed method. Hence, the average number of iterations when using the proposed method is expected to be lower compared to using the method presented in [16]. It can be seen from Fig. 11 that, based on almost the same BER performance, the proposed NPMSA-based decoder using the proposed ET method can achieve a reduction in the average number of iterations of 43.7% at an SNR of 3.0 dB compared to the SRTA-based decoder presented in [16].

IV. PERFORMANCE EVALUATION

A fully parallel (2048, 1723) LDPC decoder using a combination of NPMSA and the proposed early termination technique was designed using 90 nm CMOS technology. The chip features are summarized in Table IV. The core area is 9.6 mm^2 with a 91% density of standard-cell placement. The total chip area with I/O pads is 12.67 mm^2. The chip consumes 1110 mW at 199.6 MHz with a supply voltage of 0.9 V. Using the proposed early termination method, a reduction in energy dissipation of 60.6% (61.7%) can be achieved, and the resultant energy efficiency is 9.6 pJ/bit (9.4 pJ/bit) at SNR $= 3.0$ dB (4.4 dB).

A comparison of the metrics for the proposed LDPC decoder with published results for other LDPC decoders is summarized in Table V. It can be seen that the proposed LDPC decoder achieves the highest throughput and lowest latency based on a similar error-rate performance. The reason is that a bit-parallel architecture is used such that the number of cycles per iteration is reduced to only one. In addition, fewer iterations are needed to achieve a similar error performance. The proposed decoder occupies less normalized area than other designs, except for the stochastic decoders presented in [19] and [20]. The high area efficiency of the stochastic decoders, is traded off with a decreased throughput and a significantly increased latency. The throughput of the proposed decoder is much higher than that of the stochastic decoders. The proposed LDPC decoder achieves the highest normalized throughput-to-area ratio.

It is worth noting that the previous works considered in Table V, except for [16], did not address energy reduction for uncorrectable received words. As a result, the SRTA-based decoder presented in [16] is compared with the proposed NPMSA-based decoder for energy consumption since early termination is implemented in each for both low and high SNR regions. Based on almost the same BER performance (Fig. 11), the proposed NPMSA-based decoder combined with the proposed ET method can achieve a reduction in energy dissipation of 54.6% at SNR $=$ 3.0 dB compared to the SRTA-based decoder presented in [16].

V. CONCLUSION

For fully parallel LDPC decoders, several techniques have been presented for improving the complexity, throughput, and energy consumption. First, the NPMSA yields a lower computational complexity in the CNU. Compared with the conventional NMSA, the NPMSA requires 51% fewer comparators at the cost of a negligible error-performance loss of 0.05 dB. Second, a mixed interconnect network is adopted in the CNU to balance the interconnect complexity and the logic overhead, thereby reducing routing complexity and achieving a compact decoder. Consequently, a reduction in area of 19.8% is achieved. Third, hardware optimization is considered in order to further reduce the wordlength of the CNU and the messages between the VNUs and the CNUs. To terminate uncorrectable blocks and unnecessary decoding iterations, an early termination scheme is proposed with negligible error performance degradation, achieving a reduction in energy dissipation of 60.6%. The proposed fully parallel decoder architecture is a promising candidate for next-generation high-throughput applications.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.

[2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[3] J. Chen and M. Fossorier, "Density evolution for two improved BP-based decoding algorithms of LDPC codes," *IEEE Commun. Lett.*, vol. 6, no. 5, pp. 208–210, May 2002.

[4] Y.-L. Wang, Y.-L. Ueng, C.-L. Peng, and C.-J. Yang, "Processing-task arrangement for a low-density full-mode WiMAX LDPC codec," *IEEE Trans. Circuits Syst. I*, vol. 58, no. 2, pp. 415–428, Feb. 2011.

[5] Y.-L. Ueng, B.-J. Yang, C.-J. Yang, H.-C. Lee, and J.-D. Yang, "An efficient multi-standard LDPC decoder design using hardware-friendly shuffled decoding," *IEEE Trans. Circuits Syst. I*, vol. 60, no. 3, pp. 743–756, Mar. 2013.

[6] S.-W. Yen, S.-Y. Hung, C.-L. Chen, H.-C. Chang, S.-J. Jou, and C.-Y. Lee, "A 5.79-Gb/s energy-efficient multirate LDPC codec chip for IEEE 802.15.3c applications," *IEEE J. Solid-State Circuits*, vol. 47, no. 9, pp. 2246–2257, Sep. 2012.

[7] Y. Sun, G. Wang, and J. R. Cavallaro, "Multi-layer parallel decoding algorithm and VLSI architecture for quasi-cyclic LDPC codes," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS)*, May 2011, pp. 1776–1779.

[8] I. Djurdjevic, J. Xu, K. Abdel-Ghaffar, and S. Lin, "A class of low-density parity check codes constructed based on Reed-Solomon codes with two information symbols," *IEEE Commun. Lett.*, vol. 7, no. 7, pp. 317–319, Jul. 2003.

[9] IEEE P802.3an, 10GBASE-T Task Force [Online]. Available: http://www.ieee802.org/3/an

[10] L. Liu and C.-J. R. Shi, "Sliced message passing: High throughput overlapped decoding of high-rate low density parity-check codes," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 11, pp. 3697–3710, Dec. 2008.

[11] J. Sha, J. Lin, Z. Wang, L. Li, and M. Gao, "Decoder design for RS-based LDPC codes," *IEEE Trans. Circuits Syst. II*, vol. 56, no. 9, pp. 724–728, Sep. 2009.

[12] Y.-L. Ueng, C.-J. Yang, K.-C. Wang, and C.-J. Chen, "A multimode shuffled iterative decoder architecture for high-rate RS-LDPC code," *IEEE Trans. Circuits Syst. I*, vol. 57, no. 10, pp. 2790–2803, Oct. 2010.

[13] Z. Zhang, V. Anantharam, M. J. Wainwright, and B. Nikolić, "An efficient 10GBASE-T ethernet LDPC decoder design with low error floors," *IEEE J. Solid-State Circuits*, vol. 45, no. 4, pp. 843–855, Apr. 2010.

[14] C. Roth, A. Cevrero, C. Studer, Y. Leblebici, and A. Burg, "Area, throughput, and energy-efficiency trade-offs in the VLSI implementation of LDPC decoders," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS)*, May 2011, pp. 1772–1775.

[15] T. Mohsenin, D. Truong, and B. Baas, "A low-complexity message-passing algorithm for reduced routing congestion in LDPC decoders," *IEEE Trans. Circuits Syst. I*, vol. 57, no. 5, pp. 1048–1061, May 2010.

[16] T. Mohsenin, H. Shirani-mehr, and B. Baas, "Low power LDPC decoder with efficient stopping scheme for undecodable blocks," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS)*, May 2011, pp. 1780–1783.

[17] A. Darabiha, A. C. Carusone, and F. R. Kschischang, "Power reduction techniques for LDPC decoders," *IEEE J. Solid-State Circuits*, vol. 43, no. 8, pp. 1835–1845, Aug. 2008.

[18] P. A. Marshall, V. C. Gaudet, and D. G. Elliott, "Deeply pipelined digit-serial LDPC decoding," *IEEE Trans. Circuits Syst. I*, vol. 59, no. 12, pp. 2934–2944, Dec. 2012.

[19] A. Naderi, S. Mannor, M. Sawan, and W. J. Gross, "Delayed stochastic decoding of LDPC codes," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5617–5626, Nov. 2011.

[20] S. S. Tehrani, A. Naderi, G.-A. Kamendje, S. Hemati, S. Mannor, and W. J. Gross, "Majority-based tracking forecast memories for stochastic LDPC decoding," *IEEE Trans. Signal Process.*, vol. 58, no. 9, pp. 4883–4896, Sep. 2010.

[21] C.-L. Wey, M.-D. Shieh, and S.-Y. Lin, "Algorithms of finding the first two minimum values and their hardware implementation," *IEEE Trans. Circuits Syst. I*, vol. 55, no. 11, pp. 3430–3437, Dec. 2008.

[22] D. Shin, K. Heo, S. Oh, and J. Ha, "A stopping criterion for low-density parity-check codes," in *Proc. IEEE 65th Vehicular Technol. Conf. (VTC2007-Spring)*, 2007, pp. 1529–1533.

[23] Z. Cui and Z. Wang, "An efficient early stopping scheme for LDPC decoding," presented at the 13th NASA Symp. VLSI Design, Jun. 2007.

Chung-Chao Cheng received the M.S. degree from National Tsing Hua University, Hsinchu, Taiwan, in 2007. He is currently working toward the Ph.D. degree in electrical engineering at the National Tsing Hua University, Hsinchu, Taiwan.

His research interests include error control coding, related VLSI designs, and their applications to communications.

Jeng-Da Yang received the B.S. degree from the National Chin-Yi University of Technology, Taichung, Taiwan, in 2010. He is currently working toward the M.S. degree in electrical engineering at the National Tsing Hua University, Hsinchu, Taiwan.

His research interests include error control coding and related VLSI designs.

Huang-Chang Lee received the B.S. and M.S. degrees in electrical engineering from National Chi Nan University, Taiwan, in 2002 and 2004. He is currently working toward the Ph.D. degree in electrical engineering at the National Tsing Hua University, Hsinchu, Taiwan.

His research interests include channel coding, digital signal processing, and their applications to communications.

Chia-Hsiang Yang (S'07–M'10) received the B.S. and M.S. degrees in electrical engineering from the National Taiwan University, Taiwan, in 2002 and 2004, and the Ph.D. degree from the University of California, Los Angeles, CA, USA, in 2010.

He then joined the faculty of the Electronics Engineering Department, National Chiao Tung University, Taiwan, as an Assistant Professor. His current research interests include energy-efficient integrated circuits and architectures for biomedical and communication signal processing.

Yeong-Luh Ueng received the Ph.D. degree in communication engineering from the National Taiwan University, Taipei, Taiwan, R.O.C., in 2001.

From 2001 to 2005, he was with a private communication technology company, where he focused on the design and development of various wireless chips. Since December 2005, he has been a member of the faculty of the National Tsing Hua University, Hsinchu, Taiwan, where he is currently an Associate Professor with the Department of Electrical Engineering and the Institute of Communications Engineering. His research interests include coding theory, wireless communications, and communication ICs.