Low-Complexity Modified Trellis-Based Min-Max Non-Binary LDPC Decoders

Xinmiao Zhang
SanDisk Corporation, Milpitas, CA 95035, U.S.A
Email: xinmiao.zhang@sandisk.com

***Abstract*—**Non-Binary Low-Density Parity-Check (NBLDPC) codes over $GF(q)$ ($q > 2$) have better errorcorrecting performance than their binary counterparts when the codeword length is moderate. In this paper, modified trellis-based Min-max decoders are proposed for NB-LDPC codes. By relaxing the constraints on which messages can be included, the trellis syndrome computation is simplified without sacrificing the errorcorrecting performance. In addition, the iterative comparisons needed in computing the check-to-variable messages are replaced by one-step message selection. The decoding complexity of NB-LDPC codes grows substantially with q, and small q is preferred to achieve low complexity and high speed for data storage systems. Further simplifications are enabled by making use of the properties of $GF(4)$. Instead of three trellis syndromes, a single global syndrome is computed and stored in the check node processing. Efficient implementation architectures are also developed in this paper. Compared to prior efforts, the proposed designs require smaller area, consumes much less power, achieves higher throughput, and also has slightly better error-correcting performance.

***Index Terms*—**Low-Density Parity-Check (LDPC) codes, min-max algorithm, Non-binary, VLSI Design

I. Introduction

Compared to binary Low-Density Parity-Check (LDPC) codes, Non-Binary (NB)-LDPC codes constructed over $GF(q)$ ($q>2$) have better error-correcting capability when the codeword length is moderate. On the other hand, the complexity of the decoder increases significantly with q. Data storage systems require very high throughput. For these applications, NB-LDPC codes over $GF(4)$ received much attention recently due to their performance-complexity tradeoff.

The decoding of NB-LDPC codes can be simplified using the Extended Min-Sum (EMS) [1] and Min-max [2] algorithms, both of which are log-domain approximations to the belief propagation. The Min-max algorithm replaces the 'sum' with 'max' in the check node processing. It has lower complexity than the EMS algorithm with negligible performance loss. In addition, all simplification schemes developed for the EMS algorithm can be directly extended for the Min-max algorithm.

The check node processing is the most complicated step in NB-LDPC decoding. The iterative computations in the forward-backward scheme [3] limit the achievable throughput and the storage of intermediate results leads to large memory requirement. By representing the variable-to-check (v2c) messages as nodes in a trellis, the computations of the check-to-variable (c2v) messages are mapped to constructing paths on the trellis [4]. Only a few sorted nodes need to be stored in this approach and all c2v vectors can be computed in parallel. Nevertheless, the paths for the messages in each vector are constructed serially. The Simplified Min-Max Algorithm (SMMA) [5] allows more than one node from each stage of the trellis to be included in a c2v message computation. Although the corresponding architecture is very regular, a large number of 'min' and 'max' units are required. In the basis construction Min-max decoder [6], all the messages in a c2v vector are computed from a basis consisting of a few v2c messages using a simple logic network. It requires substantially less area than the SMMA when q is not small. However, it has noticeable performance loss when $q = 4$. Different from all prior methods, the trellis-based EMS decoder [7] first computes syndromes of the trellis using nodes from every stage. Then the c2v messages to variable node n are derived by excluding the contributions of the nodes in stage n from the syndromes. This scheme potentially allows further computation reduction. Nevertheless, the complicated syndrome computation and c2v message derivation steps in [7] make the hardware complexity of this algorithm much higher than that of the SMMA.

In this paper, novel modifications to the trellis-based Min-max decoder are proposed. By relaxing the constraints on which nodes can be included in a trellis syndrome, the number of hardware units required for the syndrome computation is substantially reduced. A simplified c2v message calculation method is also developed. It does not need the iterative updates as in the trellis-based decoder of [7], and allows all messages in a vector to be computed in one clock cycle with simple hardware. These modification schemes have been presented in [8]. By making use of the properties of $GF(4)$, it was discovered in this paper that a single global syndrome instead of three trellis syndromes need to be computed and stored in the check node processing. As a result, the decoder complexity is further reduced. Efficient hardware implementation architectures are developed to implement the modified trellis-based Min-max decoders. For a (2016, 1764) code over $GF(4)$, the

Manuscript received March 5, 2015; revised November 20, 2015.
Corresponding author email: xinmiao.zhang@sandisk.com
doi:10.12720/jcm.10.11.836-842

proposed decoders require less area and consume around 12% less power than the SMMA decoder, which has the lowest hardware complexity among available designs for *GF*(4) codes. Moreover, the proposed decoders achieve higher clock frequency and slightly better error-correcting performance.

This paper is organized as follows. Section II introduces NB-LDPC codes and the trellis-based decoding algorithm. Details about the proposed modified trellis-based decoders are presented in Section III. After the VLSI implementation architectures are presented and compared to prior designs in Section IV, conclusions are drawn in Section V.

II. NB-LDPC Codes and Trellis-Based Decoder

An LDPC code is defined by a very sparse parity check matrix *H* or the associated Tanner graph. Each row of *H* corresponds to a check equation, and each column is associated with a received symbol. A vector, *c*, is a codeword iff $cH^T = 0$. A row (column) of *H* is represented by a check (variable) node in the Tanner graph. If the entry of *H* in the *ith* row and *jth* column, $h_{i,j}$, is nonzero, then the corresponding check and variable nodes are connected by an edge in the Tanner graph. In the decoding process, messages regarding the probabilities that the received symbol equals each of the possible values are iteratively passed through the edges in the Tanner graph to find a codeword.

Algorithm A: The Min-max Algorithm

Initialization: $u_{m,n}(\alpha) = \gamma_n(\alpha)$

Iterations:

- *Check node processing*

$$v_{m,n}(\alpha) = \min_{[a_j] \in \mathcal{L}(m|a_n=\alpha)} \left(\max_{j \in S_v(m)\backslash n} u_{m,j}(a_j) \right) \quad (1)$$

- *Variable node processing*

$$u'_{m,n}(\alpha) = \gamma_n(\alpha) + \sum_{i \in S_c(n)\backslash m} v_{i,n}(\alpha)$$

$$u_{m,n}(\alpha) = u'_{m,n}(\alpha) - u'_{m,n}(\hat{\alpha}) \quad (2)$$

- *A posteriori information computation*

$$\tilde{\gamma}_n(\alpha) = \gamma_n(\alpha) + \sum_{i \in S_c(n)} v_{i,n}(\alpha)$$

For a NB-LDPC code over *GF*(*q*), each message vector consists of *q* Log-Likelihood Ratios (LLRs) in the Min-max algorithm. The LLRs for a message vector are defined as $l(\alpha) = \log(P(\hat{\alpha})/P(\alpha))$, where α is an element of *GF*(*q*) and $\hat{\alpha}$ is the most likely element. Each LLR is non-negative, and the smaller the LLR, the more reliable the corresponding message. Let the LLR vector from check (variable) node *m*(*n*) to variable (check) node *n*(*m*) be $v_{m,n}$ ($u_{m,n}$). $S_c(n)$ ($S_v(m)$) is the set of check (variable) nodes connected to variable (check) node *n*(*m*). Let $\mathcal{L}(m|a_n = \alpha)$ be the set of sequences of finite field elements $[a_j]$ $(j \in S_v(m)\backslash n)$ such that $\sum_{j \in S_v(m)\backslash n} h_{m,j} a_j = h_{m,n}\alpha$. This set is also referred to as the configuration set [1]. Assume that the multiplications of the entries of *H* are taken care of by separate units. Represent the LLR vector from the channel for variable node *n* by γ_n. The Min-max algorithm is described in Algorithm A.

The hard decision of the *n*th received symbol is made as $\arg \min_\alpha(\tilde{\gamma}_n(\alpha))$ at the end of each decoding iteration. The decoding stops when a codeword is found or the maximum iteration is reached. The check node processing is the most complicated step. The EMS algorithm is only different from the Min-max algorithm in that the 'max' in (1) is replaced by 'sum'. Accordingly, simplification schemes for the EMS algorithm can be also extended to the Min-max algorithm.

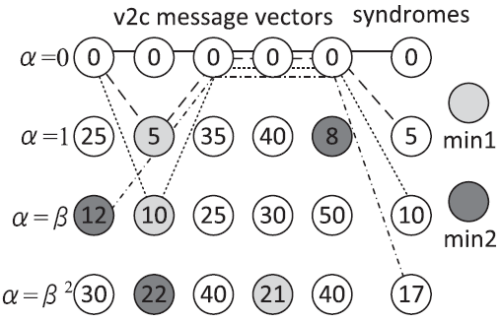

Fig. 1. Example trellis for codes over *GF*(4)

The trellis-based EMS decoder [7] makes use of a trellis representation of the v2c messages. The trellis consists of d_c columns of nodes if the row weight of the code is d_c. The nodes in each column represent a v2c vector. The trellis can be transformed by defining $\hat{u}_{m,n}(\alpha) = u_{m,n}(\alpha + \hat{\alpha})$ [5]. In the $\hat{u}_{m,n}$ trellis, the LLR corresponding to the zero field element is always zero. Fig. 1 shows an example of such a trellis for a code over *GF*(4). Let β be a primitive element of *GF*(4). Then the elements of *GF*(4) are $0, 1, \beta, \beta^2$. In [7], an extra column is added to the trellis to represent the syndromes. Let $\mathcal{T}(m|\alpha)$ be the configuration set of the sequences of d_c symbols $[a_j]$ $(j \in S_v(m))$ such that $\sum_{j \in S_v(m)} a_j = \alpha$. The syndromes for the EMS algorithm are defined as

$$w(\alpha) = \min_{[a_j] \in \mathcal{T}(m|\alpha)} \left(\sum_{j \in S_v(m)} u_{m,j}(a_j) \right)$$

The syndrome computation is similar to the c2v message computation, except that no v2c vector is excluded. Then $v_{m,n}$ is derived by excluding the contributions of $u_{m,n}$ from the syndromes. Since the syndromes only need to be computed once for all c2v message vectors from the same check node, much redundancy has been eliminated. Let $\hat{v}_{m,n}(\alpha) = v_{m,n}(\alpha + \hat{\alpha})$ be the transformed c2v messages. Apparently, $\hat{v}_{m,n}(0) = 0$. Since 'min' is the last computation done in the check node processing, only the most reliable nodes, and hence a small number of nonzero-LLR nodes in the trellis, which are called

deviation nodes, contribute to the c2v outputs. It was proposed in [7] to consider only n_c nodes with the smallest LLR in each nonzero row of the trellis and limit the number of deviation nodes in each configuration of $\mathcal{T}(m|\alpha)$ to n_r.

The syndrome computation in [7] does not allow more than one node from a stage of the trellis to be included in any configuration. As a result, the LLRs of quite a few configurations need to be compared to derive the syndromes, even if the involved finite field is very small. The example shown in Fig. 1 is for $n_c = 2$ and $n_r = 2$.

The three nonzero elements of *GF*(4) satisfy $1+\beta = \beta^2$. Hence, with $n_c = 2$, $w(\beta^2)$ takes the minimum of the following four values: $min1(\beta^2)$, $min1(1) + min1(\beta)$, $min1(1) + min2(\beta)$, and $min2(1) + min1(\beta)$. Here $min1(\cdot)$ and $min2(\cdot)$ denote the minimum and second minimum LLR, respectively, in a row of the trellis. For any α, $min1(\alpha) \leq min2(\alpha)$. Hence $min2(1)+min2(\beta)$ does not need to be considered. Nevertheless, it is possible that the min1 nodes for two different rows, such as the $min1(1)$ and $min1(\beta)$ in Fig. 1, belong to the same stage. Such two min1 nodes do not form a legal configuration according to [7]. In this case, the cross-over sums of the min1 and min2 values for those two rows need to be considered. Hence three adders are required to compute a syndrome. Other syndromes are computed in a similar way. Each path in Fig. 1 shows the node(s) included in the configuration corresponding to the syndrome.

Let the configuration corresponding to $w(\alpha)$ be $\eta^{(\alpha)} = [\eta_0^{(\alpha)}, \eta_1^{(\alpha)}, \ldots, \eta_{d_c-1}^{(\alpha)}]$. $\hat{v}_{m,n}(0)$ are initialized to zero, and all other c2v messages are initialized to the maximum possible LLR. In [7], the c2v messages are computed as

$$\hat{v}_{m,n}(\alpha-\eta_n^{(\alpha)}) = \min(\hat{v}_{m,n}(\alpha-\eta_n^{(\alpha)}), w(\alpha)-\hat{u}_{m,n}(\eta_n^{(\alpha)})) \quad (3)$$

To compute the c2v messages to variable node *n*, the finite field elements and LLRs of the v2c messages from this node are subtracted. If $\eta_n^{(\alpha)} \neq 0$, then $\alpha - \eta_n^{(\alpha)} \neq \alpha$. Therefore, $\hat{v}_{m,n}(\alpha)$ for $\alpha \neq 0$ will not be covered by (3) if $\eta^{(\alpha)}$ has a deviation node in stage *n*. In [7], it is set to $min1(\alpha)$ if this min1 node is not in stage *n*. Otherwise, it is set to $min2(\alpha)$. Different α may lead to the same $\alpha - \eta_n^{(\alpha)}$. Hence, the computations in (3) are repeated for each syndrome, and the 'min' operation is taken. Such iterative computations require a large number of comparator-register loops and prohibit all the messages in a c2v vector from being generated simultaneously.

III. Modified Trellis-Based Min-Max Decoders

By relaxing the constraints on the configurations and analyzing the possible updates in the c2v message computations, modified trellis-based Min-max decoders are proposed next. The proposed design requires only a fraction of the computations to derive the syndromes, and the iterative operations are eliminated from the c2v message calculations. Moreover, by utilizing the property that there are only three nonzero elements in *GF*(4), a single global syndrome instead of three syndromes needs to be computed and stored in the check node processing. It will be shown in Section IV that the proposed modifications and simplifications lead to fewer gates, lower power and higher clock frequency in the Min-max decoder.

It was found in [5] that allowing multiple nodes in the same stage of the trellis to be included in a configuration only introduces negligible performance loss. The reason is that a node in the same stage can be considered as an approximation of a node with the same finite field element from another stage [6]. Such an approximation leads to over or underestimation of the LLRs, and either compensates or deviates the approximations that have already been made in the EMS or Min-max algorithms. We propose to incorporate this relaxation into the configurations for the trellis-based Min-max decoder from [7]. Fig. 2 shows the bit error rates (BERs) of NB-LDPC decoding algorithms for a (3780, 3212) code over *GF*(4) under the AWGN channel. This regular code has $d_c = 27$ and column weight $d_v = 4$. The maximum iteration number was set to 15. It can be observed that the proposed modified trellis-based Min-max decoder even has slightly better performance than the SMMA.

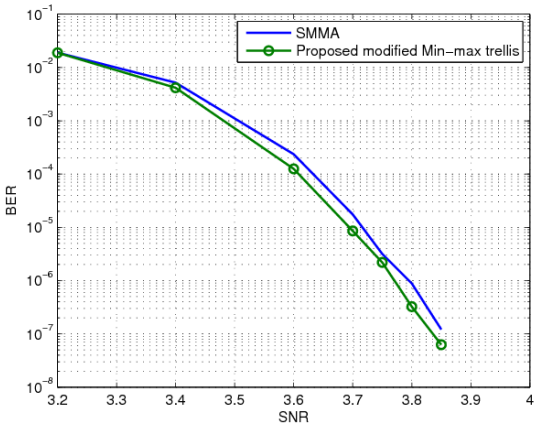

Fig. 2. Simulation results of NB-LDPC decoding algorithms for (3780, 3212) code over *GF*(4) under AWGN channel

For the Min-max algorithm, the 'sum' in the check node processing for the trellis-based EMS algorithm should be replaced by 'max'. Previously, when $n_r = 2$, crossover maximums of the min1 and min2 values need to be computed in case that the min1 nodes belong to the same stage of the trellis. By removing the constraint that the nodes for a configuration have to come from different stages, all the cross-over computations are eliminated, and the syndromes are derived solely from the min1 values. As a result, the number of 'max' units for computing a syndrome is reduced from three to one. In the case that n_r is larger, much more cross-over computations need to be done if the nodes for a configuration have to come from different stages of the

trellis. For example, when n_r = 3, 7 cross-over computations are needed for the possible combos of three finite field elements. On the other hand, only one computation over the three min1 nodes is sufficient if they are allowed to be from the same stage. Accordingly, when n_r is larger, more substantial complexity reduction would be achieved on the syndrome computation by removing the constrains on the nodes.

In [7], the c2v messages are derived using (3), and the computations are repeated for each syndrome corresponding to a nonzero finite field element. The 'min' operation in (3) is necessary because different α may lead to the same $\alpha - \eta_n^{(\alpha)}$. As a result, $q-1$ clock cycles are required to derive a c2v message vector from the syndromes for codes over *GF*(*q*), despite that all d_c c2v vectors from the same check node can be computed in parallel. Since the 'min' operation needs to be done for each c2v message with a nonzero finite field element, it adds a large number of feedback loops consisting of comparators and registers. Through analyzing the possible updates that can be made by (3) when the syndromes are computed solely from the min1 nodes, a simplified method is proposed in the following to derive the c2v messages without any iterative computation for the case of n_r = 2. Since there are three nonzero elements in *GF*(4), 2 is the largest possible value of n_r for codes over *GF*(4).

First consider the case that the configuration corresponding to $w(\alpha)$ $(\alpha \neq 0)$ has only one deviation node in stage *i*. Accordingly, $w(\alpha) = min1(\alpha)$. Also $\eta_i^{(\alpha)} = \alpha$, and $\eta_n^{(\alpha)} = 0$ for $0 \leqslant n < d_c$ and $n \neq i$. The initial values of $\hat{v}_{m,n}(\alpha)$ for $0 \leqslant n < d_c$ and $\alpha \neq 0$ are set to the largest possible LLR. As a result of the computations in (3), $\hat{v}_{m,n}(\alpha)$ for $n \neq i$ becomes $w(\alpha) = min1(\alpha)$. For $n = i$, the equation in (3) would update $\hat{v}_{m,i}(\alpha - \eta_i^{(\alpha)}) = \hat{v}_{m,i}(\alpha - \alpha) = \hat{v}_{m,i}(0)$. However, this value has been initialized to zero, and will not get updated. No value has been derived for $\hat{v}_{m,i}(\alpha)$ through (3), and $\hat{v}_{m,i}(\alpha)$ is the min1 value for row α in the trellis. From [7], $\hat{v}_{m,i}(\alpha)$ is set to $min2(\alpha)$ to exclude the contribution of the v2c messages from variable node *i*. In this process, the c2v messages derived are $\hat{v}_{m,n}(\alpha)$ $(0 \leqslant n < d_c)$, which have the same finite field element as the syndrome $w(\alpha)$. If there are other syndromes whose configurations have one deviation node, then the corresponding computations will derive distinct c2v messages. Hence none of the c2v messages will be updated for a second time through (3) using the syndromes with one deviation node.

Next consider the case that the configuration corresponding to $w(\alpha)$ $(\alpha \neq 0)$ has two deviation nodes, and they are in stage *i* and *j*. Note that $\eta_i^{(\alpha)} + \eta_j^{(\alpha)} = \alpha$ and $\eta_i^{(\alpha)} \neq \eta_j^{(\alpha)}$. Since $w(\alpha) = min1(\eta_i^{(\alpha)}) + min1(\eta_j^{(\alpha)})$ $(w(\alpha) = \max(min1(\eta_i^{(\alpha)}), min1(\eta_j^{(\alpha)}))$ in the Min-max algorithm) and $\alpha - \eta_i^{(\alpha)} = \eta_j^{(\alpha)}$, (3) is reduced to $\hat{v}_{m,i}(\eta_j^{(\alpha)}) = \min(\hat{v}_{m,i}(\eta_j^{(\alpha)})$. Similarly, $\hat{v}_{m,j}(\eta_i^{(\alpha)})$ should be updated as $\min(\hat{v}_{m,j}(\eta_i^{(\alpha)}), min1(\eta_i^{(\alpha)}))$. $\hat{v}_{m,i}(\eta_j^{(\alpha)})$ and $\hat{v}_{m,j}(\eta_i^{(\alpha)})$ may have been computed previously from other syndromes. Even so, because $i \neq j$, they were set to $min1(\eta_j^{(\alpha)})$ and $min1(\eta_i^{(\alpha)})$, respectively, as discussed in the previous paragraph. Therefore, the updating of $\hat{v}_{m,i}(\eta_j^{(\alpha)})$ and $\hat{v}_{m,j}(\eta_i^{(\alpha)})$ can be skipped since their values will not change. For $n \neq i, j$, $\hat{v}_{m,n}(\alpha)$ is updated by making use of $w(\alpha)$. In addition, $\hat{v}_{m,i}(\alpha)$ and $\hat{v}_{m,j}(\alpha)$ are set to $min1(\alpha)$ or $min2(\alpha)$ depending on whether the min1 node for row α is in stage *i* or *j*. These c2v messages have the same finite field element as the syndrome. Therefore, they will not be updated again when the computations in (3) are repeated for other syndromes.

In summary, the c2v message computation for the case of n_r = 2 can be carried out according to the simplified process in Algorithm B. For a given $\alpha \neq 0$, $\alpha\beta$ and $\alpha\beta^2$ are the other two nonzero elements of *GF*(4). In Algorithm B, $idx(\alpha)$ denotes the stage index of the min1 node in the row of α in the trellis. This algorithm can be applied to both the EMS and Min-max algorithms. It generates exactly the same results as (3), and does not bring any performance loss. To compute a c2v message, only one of the three values need to be selected by multiplexors controlled by simple logic. Hence, the area requirement is substantially lower than that for implementing (3). In addition, Algorithm B generates all the messages in a c2v vector in one clock cycle, and can achieve much higher throughput.

Algorithm B: Simplified c2v Message Computation

for each $\alpha \neq 0$

if $\eta^{(\alpha)}$ has one deviation node (in stage $idx(\alpha)$)

$$\hat{v}_{m,n}(\alpha) = \begin{cases} min1(\alpha) \text{ if } n \neq idx(\alpha) \\ min2(\alpha) \text{ if } n = idx(\alpha) \end{cases}$$

if $\eta^{(\alpha)}$ has two deviation nodes (in stages $idx(\alpha\beta)$,$idx(\alpha\beta^2)$)

$$\hat{v}_{m,n}(\alpha) = \begin{cases} w(\alpha) \text{ if } n \neq idx(\alpha\beta), idx(\alpha\beta^2) \\ min1(\alpha) \text{ if } n = idx(\alpha\beta) \text{ or } idx(\alpha\beta^2) \\ \quad \text{and } n \neq idx(\alpha) \\ min2(\alpha) \text{ if } n = idx(\alpha\beta) \text{ or } idx(\alpha\beta^2) \\ \quad \text{and } n = idx(\alpha) \end{cases}$$

From Algorithm B, $w(\alpha)$ only affects the output c2v messages when there are two deviation nodes in the path corresponding to $w(\alpha)$. This happens only if $min1(\alpha) \geq \max(min1(\alpha\beta), min1(\alpha\beta^2))$. In this case, $w(\alpha)$ equals the second smallest among $min1(\alpha)$,

$min1(\alpha\beta)$, and $min1(\alpha\beta^2)$. Since there are only three nonzero field elements in *GF*(4) and hence three min1 values, the syndrome values that can possibly become the output c2v message is always the second smallest among $min1(1)$, $min1(\beta)$, and $min1(\beta^2)$. As a result, only one single global syndrome, *w*, needs to be derived. From this analysis, the modified trellis-based Min-max check node processing for *GF*(4) codes can be simplified as in Algorithm C. In this algorithm, $f(\alpha)$ is a flag indicating whether there is one or two deviation nodes in $\eta^{(\alpha)}$. The calculations above the dashed line derive the global syndrome *w* and the flag $f(\alpha)$ for each nonzero $\alpha \in$ *GF*(4). Those under the line recover the c2v messages by making use of the global syndrome and flags. The c2v messages derived through Algorithm C are exactly the same as those in Algorithm B even though only one instead of three syndromes are used.

Algorithm C: Single-syndrome modified trellis-based Min-max check node processing for $GF(4)$ codes

for each $\alpha \neq 0$
  if $min1(\alpha) < \max(min1(\alpha\beta), min1(\alpha\beta^2))$
    $f(\alpha) = 0$
  else
    $f(\alpha) = 1$
$w =$ the second smallest of $min1(1), min1(\beta), min1(\beta^2)$

- -

$\hat{v}_{m,n}(0) = 0$
for each $\alpha \neq 0$
  if $f(\alpha) = 0$

$$\hat{v}_{m,n}(\alpha) = \begin{cases} min1(\alpha) \text{ if } n \neq idx(\alpha) \\ min2(\alpha) \text{ if } n = idx(\alpha) \end{cases}$$

  else

$$\hat{v}_{m,n}(\alpha) = \begin{cases} w \text{ if } n \neq idx(\alpha\beta), idx(\alpha\beta^2) \\ min1(\alpha) \text{ if } n = idx(\alpha\beta) \text{ or } idx(\alpha\beta^2) \\ \qquad \text{and } n \neq idx(\alpha) \\ min2(\alpha) \text{ if } n = idx(\alpha\beta) \text{ or } idx(\alpha\beta^2) \\ \qquad \text{and } n = idx(\alpha) \end{cases}$$

IV. VLSI Architectures for Modified Trellis-Based Decoders

By making use of the properties of *GF*(4), efficient VLSI architectures are developed next to implement the proposed modified single-syndrome Min-max algorithm with $n_r = 2$ for codes over *GF*(4).

As discussed previously, the proposed schemes allow multiple nodes in the same stage of the trellis to be included in a configuration. Hence only one pair of configurations need to be considered for computing $w(\alpha)$ ($\alpha \neq 0$) over *GF*(4): one with deviation node $min1(\alpha)$, and one with the other two min1 nodes as the deviation nodes. Although one single syndrome can be used instead as shown in Algorithm C, the LLRs of the two configurations in every pair still need to be compared to derive the flags as follows

$$\begin{cases} f(1) = min1(1) < \max(min1(\beta), min1(\beta^2))\,?\,0:1 \\ f(\beta) = min1(\beta) < \max(min1(1), min1(\beta^2))\,?\,0:1 \\ f(\beta^2) = min1(\beta^2) < \max(min1(1), min1(\beta))\,?\,0:1 \end{cases}$$

Instead of carrying out 6 min or max comparisons as in the above equations, 3 pair-wise comparisons are done among the three min1 values. The results are shared to generate not only the flags, but also the global syndrome, *w*, using simple control logics.

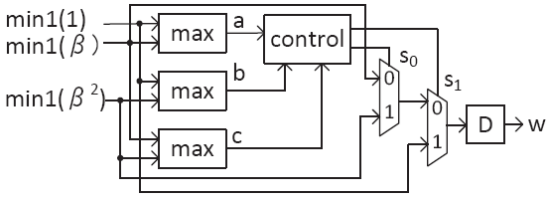

Fig. 3. Global syndrome computation architecture for codes over *GF*(4)

Fig. 3 shows the architecture for computing the global syndrome and flags. Each pair of the min1 values are compared first. Assume that the 'max' unit outputs '0' if its upper input is larger than its lower input. Then $s_0 = b \oplus c$ and $s_1 = a \oplus b$, where $\oplus$ means XOR. In addition, the flags are generated using the comparator outputs as

$$\begin{cases} f(1) = a'b' \\ f(\beta) = ac' \\ f(\beta^2) = bc \end{cases}$$

where '+' denotes logic OR and $'$ is logic NOT. These logics are included in the control block and are not explicitly shown in Fig. 3. If three syndromes need to be computed, the same three comparators in Fig. 3 can be used. However, three copies of the multiplexors with different control signals and registers would be required to derive and store the three syndromes. Therefore, computing and storing a single global syndrome requires much smaller area.

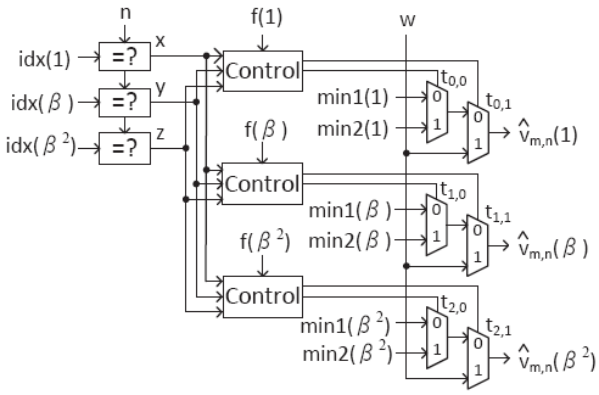

Fig. 4. c2v message computation architecture for codes over *GF*(4)

Once the global syndrome and flags are derived, the c2v messages can be computed by the architecture in Fig. 4. First *n* is compared to the index of each min1 node. The output of the comparator is asserted if *n* equals the index. Depending on the flags and if the min1 nodes are in stage *n*, one of $min1(\alpha)$, $min2(\alpha)$ and *w* becomes $\hat{v}_{m,n}(\alpha)$ according to Algorithm C. It can be derived that the control signals of the multiplexors are $t_{0,0} = x$, $t_{0,1} = f(1)(y+z)'$, $t_{1,0} = y$, $t_{1,1} = f(\beta)(x+z)'$, $t_{2,0} = z$, $t_{2,1} = f(\beta^2)(x+y)'$. In the case that

three syndromes are available, $w(\alpha)$ would be connected to the multiplexor that outputs $\hat{v}_{m,n}(\alpha)$. Hence, the complexity of the c2v message computation architecture is not reduced by adopting the single-syndrome scheme.

The *a posteriori* information for a variable node is the sum of the channel information and the c2v vectors from all connected check nodes. Subtracting the sum by the c2v vector from check node *m*, the v2c vector to check node *m* is derived. The normalization for the v2c messages according to (2) is done by using a tree consisting of 2-input 'min' operators to find the smallest LLR in a vector. Then this smallest LLR is subtracted from each v2c message in the vector. When all the q messages in a vector are kept, the trellis transformation is a permutation on the messages in the vector, and hence is implemented by $\lceil \log_2 q \rceil$ stages of q 2-input multiplexors. For the purpose of conciseness, the multiplications by $h_{i,j}$ have been taken out of algorithms A through C. Multiplying a vector consisting of all field elements by $h_{i,j}$ is also a permutation on the vector, and hence can be also implemented by a network of switching logic. These trellis transformation and multiplication permutation network architectures are available in previous publications, such as [5], and are not repeated in this paper.

TABLE I: MIN-MAX DECODER COMPLEXITY COMPARISONS FOR A (2016, 1764) QCNB-LDPC CODE OVER *GF*(4) WITH D_C = 32 AND D_V = 4

	SMMA [5]	Basis-construction [6]	Proposed (3 syndromes [8])	Proposed (single syndrome)
logic gate (XOR)	277k	273k	266k (28k active for 1 clk/iter.)	257k (19k active for 1 clk/iter.)
registers	23k	16k	20k	20k
memory	60k	60k	60k	60k
# of clks	260	260	260	260

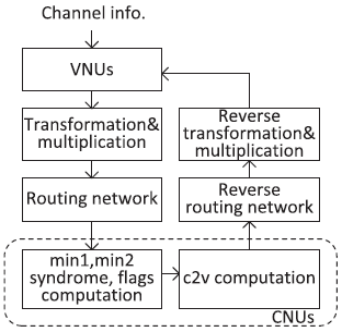

Fig. 5. Block diagram for sliced message-passing modified trellis-based Min-max decoder

Our decoder is designed for Quasi-Cyclic (QC) LDPC codes. The *H* matrix of these codes consists of submatrixes that are either zero or α-multiplied cyclical permutation matrixes. Such a permutation matrix is a cyclically shifted identity matrix whose nonzero entry in a row equals that in the previous row multiplied by α. The sliced message-passing scheme [9] is adopted for the decoder, and the block diagram is shown in Fig. 5. Let the size of each sub-matrix be $e \times e$. e Variable Node Units (VNUs) are adopted to process one block column of submatrixes simultaneously. The trellis transformation and multiplication by the *H* matrix entries are permutations inside each message vector. Since the offsets of the submatrxies of *H* are different, the v2c message vectors also need to pass through a routing network to be sent to the right Check Node Units (CNUs).

The number of CNUs equals the number of rows in *H*, and all rows are processed in parallel. For a row of *H*, there is only one nonzero entry in each block of *e* columns. Each CNU compares one v2c message vector with intermediate min1 and min2 vectors stored in registers and updates those values in one clock cycle. Assuming *H* is regular, it takes d_c clock cycles to find the min1 and min2 nodes. The syndrome computation architecture in Fig. 3 and c2v message computation architecture in Fig. 4 are also parts of the CNUs. From the min1 nodes, the syndromes are computed in one clock cycle. Then one c2v message vector is computed by each CNU at a time. Note that although the c2v message computation architecture in Fig. 4 is busy in each clock cycle, the syndrome computation architecture in Fig. 3 is only activated for one clock cycle in each decoding iteration. This helps to reduce the power consumption of the decoder. Since there is one CNU for each row of *H*, all c2v message vectors for the variable nodes in one block column of *H* are generated at a time. They are used to carry out the corresponding variable node processing right away, and the updated v2c message vectors are sent to the comparator parts of the CNUs to compute the min1 and min2 values to be used in the next decoding iteration. In this process, the v2c messages do not need to be stored, and the c2v messages are generated from the min1, min2, syndrome, and flags when needed.

The hardware complexities of the proposed decoders are analyzed for an example (2016, 1764) QCNB-LDPC code over *GF*(4) and summarized in Table I. The *H* matrix of this code has 4 × 32 nonzero sub-matrixes of dimension 63 × 63. Hence, d_c = 32 and d_v = 4. The proposed decoder employs 63 VNUs and 63 × 4 CNUs. From simulations, the decoding takes 6.5 iterations on average. Considering the 8 stages of pipelining in the decoder, around 6.5 × (d_c + 8) = 260 clock cycles are needed to decode a word. On 28nm CMOS technology, the proposed decoder can easily achieve 500Mhz clock frequency, and accordingly 6.78Gbps throughput. The SMMA [5] and basis-construction Min-max decoders [6] have lower complexity than other existing designs, and their complexities are also listed in Table I. Note that the basis-construction decoder has performance loss for *GF*(4) codes. It can be seen from this table that the proposed modified trellis-based decoders have slightly lower gate count. However, the syndrome computation architecture is only activated for one clock cycle in each decoding

iteration. Considering this, the power consumption of the modified trellis-based decoders is 12% lower. By computing one single global syndrome instead of three syndromes, the area of the CNU in the proposed modified trellis-based decoder can be further reduced by 12%, although this does not lead to significant overall decoder complexity reduction. Besides the better errorcorrecting performance as shown in Fig. 2, the proposed decoders also have fewer levels of logic in the CNUs, and hence can achieve higher clock frequency.

V. Conclusions

Modified trellis-based Min-max decoders for NBLDPC codes are proposed in this paper. By relaxing the constraints on which nodes can be included in a configuration, the number of hardware units required for computing the trellis syndromes has been reduced by three times. In addition, the proposed simplified c2v message computation allows all messages in a vector to be computed in one clock cycle using simple hardware. Further simplification is achieved by computing only one single syndrome through making use of the properties of *GF*(4). Compared to the best previous design, the proposed decoders have lower hardware complexity and can achieve slightly better error-correcting performance. Future research will address further taking advantage of the syndrome to reduce the redundancy in c2v message computation.

References

[1] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(*q*)," *IEEE Trans. on Commun.*, vol. 55, no. 4, pp. 633-643, Apr. 2007.

[2] V. Savin, "Min-Max decoding for non binary LDPC codes," in *Proc. IEEE Intl. Symp. on Info. Theory*, pp. 960-964, Toronto, Canada, Jul. 2008.

[3] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-Domain decoding of LDPC codes over *GF*(*q*)," in *Proc. IEEE Intl. Conf. on Commun.*, Paris, France, Jun. 2004, pp. 772-776.

[4] X. Zhang and F. Cai, "Reduced-complexity decoder architecture for non-binary LDPC codes," *IEEE Trans. on VLSI Syst.*, vol. 17, no. 7, pp. 1229-1238, Jul. 2011.

[5] X. Chen and C. Wang, "High-throughput efficient non-binary LDPC decoder based on the simplified min-sum algorithm," *IEEE Trans. on Circuits and Syst.-I*, vol. 59, no. 11, pp. 2784-2794, Nov. 2012.

[6] F. Cai and X. Zhang, "Relaxed min-max decoder architectures for nonbinary low-density parity-check codes," *IEEE Trans. on VLSI Syst.*, vol. 21, no. 11, pp. 1229-1238, Nov. 2013.

[7] E. Li, D. Declercq, and K. Gunnam, "Trellis-based extended minsum algorithm for non-binary LDPC codes and its hardware structure," *IEEE Trans. on Commun.*, vol. 61, no. 7, pp. 2600-2611, Jul. 2013.

[8] X. Zhang, "Modified trellis-based min-max decoder for non-binary LDPC codes," in *Proc. Intl. Conf. Computing, Networking and Commun.*, Anaheim, CA, Feb. 2015.

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

Xinmiao Zhang received her Ph.D. degree from the University of Minnesota in 2005. She has been a Timothy E. and Allison L. Schroeder Assistant Professor 2005-2010, and then a tenured Associate Professor at Case Western Reserve University 2010-2013. Currently, she is a Principal Research Engineer at SanDisk. Her research interests include VLSI architecture design for error-correcting coding, signal processing, and cryptography. Dr. Zhang received a National Science Foundation CAREER award in January 2009.She is also the recipient of the Best Paper Award at the ACM Great Lakes Symposium on VLSI 2004. She authored the book "VLSI Architectures for Modern Error-Correcting Codes" (Taylor and Francis, 2015), and has published more than 70 papers on BCH, LDPC, Reed-Solomon decoders and the Advanced Encryption Standard (AES) algorithm. She is a member of the IEEE CASCOM, VSA, and DISPS technical committees, and served on the committees of many conferences, including ISCAS, SiPS, ICASSP, GlobalSIP, ICC, NVMW, and GLSVLSI. She is currently an associate editor for the IEEE Transactions on Circuits and Systems-I.