International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 4, No. 1, March 2015, pp. 6~12
ISSN: 2089-4864

An Efficient VLSI Architecture for Nonbinary LDPC Decoder with Adaptive Message Control

R. Varatharajan
Department of ECE, Sri Lakshmi Ammal Engineering College, Chennai, India

Article Info

Article history:

Received Sep 26, 2014
Revised Nov 13, 2014
Accepted Dec 2, 2014

Keyword:

Adaptive message control
Decoding
Extended min-sum
Non binary low-density parity-check (LDPC) codes
VLSI architecture

ABSTRACT

A new decoder architecture for nonbinary low-density parity check (LDPC) codes is presented in this paper to reduce the hardware operational complexity and power consumption. Adaptive message control (AMC) is to achieve the low decoding complexity that dynamically trims the message length of belief information to reduce the amount of memory accesses and arithmetic operations. A new horizontal nonbinary LDPC decoder architecture is developed to implement AMC. Key components in the architecture have been designed with the consideration of variable message lengths to leverage the benefit of the proposed AMC. Simulation results demonstrate that the proposed nonbinary LDPC decoder architecture can significantly reduce hardware operations and power consumption as compared with existing work with negligible performance degradation.

Corresponding Author:

R. Varatharajan
Professor & Head Department of ECE, Sri Lakshmi Ammal Engineering College, Chennai, Inda
Email: varathu21@yahoo.com

1. INTRODUCTION

Low-density parity-check (LDPC) codes [1], [2] are considered as one of the most powerful capacity-approaching codes. LDPC codes can be constructed in both binary domain and Galois fields (i.e., GF(2^m), where m >1). Binary LDPC codes have been studied extensively and adopted in many communication protocols, such as DVB-T2, WiMax, etc. In general, a very long code length is necessary for binary LDPC codes to approach the channel capacity. Low-Density Parity-Check (LDPC) codes have recently received a lot of attention because of their admirable performance and have been widely considered as a promising candidate error-correcting coding scheme for many real applications in telecommunications and magnetic storage. Nonbinary LDPC codes constructed in Galois fields offer improved performance at a moderate code length. In addition, nonbinary LDPC codes can be combined with high order modulations to increase the bandwidth efficiency. Due to these features, design and implementation of nonbinary LDPC codes have become critical for many emerging applications such as underwater acoustic communications.

A key challenge in the application of nonbinary LDPC codes is their high decoding complexity, as each symbol in the codeword is decoded using a long message. A lot of research effort aims at reducing the decoding complexity of nonbinary LDPC codes at the algorithm level. To deal with the problem that computational complexity increases exponentially with, the extended Min-Sum (EMS) was proposed in [3] where only the most significant n_m entries in a message were used in the decoding. A decoding technique developed in [4] conducted the EMS with a reduced complexity of $o(n_m \log_2 n_m)$ with minor performance degradation. It should be noted that these algorithm-level techniques do not explicitly consider the complexity in the implementation of nonbinary LDPC decoders.

Different from these existing works targeting hardware implementation cost, the focus of this paper is to reduce the hardware operational complexity in nonbinary LDPC decoder architectures. This enables

efficient decoding suitable for emerging applications such as underwater acoustic sensor networks [5] that are under the severe resource (e.g., energy) constraints. It was reported that memory accesses and arithmetic operations are the two major contributors to the operating cost in LDPC decoders. As the amount of memory accesses and arithmetic operations is largely determined by the message length, reducing message length is deemed as an effective way for efficient decoding. Different from the EMS which maintains a constant message length for every symbol; the proposed AMC adjusts the message length adaptively, which can reduce the message length at the required performance.

In this paper, a horizontal sequential VLSI architecture for the nonbinary LDPC decoder employing the AMC is developed. The design of the key components in this architecture, such as variable node and check node update units, is optimized by exploiting the variable length sorters, which can be dynamically configured in different functional units to accommodate variable message lengths. The AMC is implemented by a low-complexity approximation method to avoid hardware overheads and performance impact. A mapping table based approach is proposed to conduct searching operations with low complexity. We apply AMC to EMS to address the memory and throughput issues caused by the worst case message length. Note that AMC can also be employed in other decoding update rules such as the Min-Max algorithm. In addition, the proposed AMC can be generally applied to most existing decoder architectures (sequential, partial parallel and fully parallel).

2. REVIEW OF NONBINARY LDPC CODES

A nonbinary LDPC code is defined by its parity check matrix (PCM) $\mathbf{H} = [h_{ij}]$, which is an $M \times N$ sparse matrix with low density of nonzero entries. The nonzero entries of **H** take values from a Galois field $GF(2^m)$. A length-N vector **x** with entries having values from $GF(2^m)$ is a codeword if and only if $\mathbf{Hx} = \mathbf{0}$. Each entry in the codeword is called a symbol. An LDPC code with PCM **H** can be represented by a bipartite graph called Tanner graph , which consists of two categories of nodes; that is, N variable nodes v_i, $1 \le i \le N$ and M check nodes c_j , $1 \le j \le M$. A variable node v_i is connected with a check node c_j if and only if h_{ji} in the PCM **H** is nonzero.

The decoding process of nonbinary LDPC codes operates on the messages that represent the probability distribution of symbols. A message is a length-2^m vector recording the 2^m belief information of a symbol subject to channel noise, where each belief information indicates the probability of this noisy symbol to be one of the 2^m elements in $GF(2^m)$. The decoding process is essentially iterative message exchange and update between the check nodes and variable nodes in the Tanner graph representation. The messages on the Tanner graph exchange and update in two directions. Variable node messages (VNMs), denoted by **q**, pass from the variable nodes to the check nodes; and check node messages (CNMs), denoted by **r**, pass from the check nodes to the variable nodes. VNMs are initialized with channel messages, which are the input to the decoder. Then, VNMs are sent to the check nodes to update the CNMs. The new CNMs are then sent back to the variable nodes and used together with the channel messages to update the VNMs. This procedure is known as the belief propagation.

There are mainly two types of decoding algorithms – sum-product algorithm (SPA) and min-sum (MS), of which the latter one is a mathematical approximation of SPA where the sum of product is replaced by the maximum product term. Due to its relatively simple operation, MS is often chosen for practical applications. Further reduction in the complexity of MS is desirable; one such approach is the extended MS (EMS) algorithm.

3. MIN-SUM WITH ADAPTIVE MESSAGE CONTROL

In this section, an adaptive message control method that dynamically adjusts the message length of a symbol during the decoding iteration was developed. A truncation scheme is proposed for the proposed AMC-EMS algorithm.

A. Adaptive Message Control (AMC)

Intuitively, when the distribution of belief information is more concentrated, a message with a smaller number of entries might be sufficient to retain the same amount of the belief information. Exploiting this fact, we propose to adaptively control the message length during the decoding iteration. Our approach leads to two major advantages. First, due to the casual nature of channel noise, channel messages for different symbols may have different statistics. In comparison with the EMS which maintains a fixed number of entries for all channel messages, the proposed AMC can lower the average message length while retaining the same amount of belief information. Second, as the decoding proceeds, the belief information will gradually concentrate around the correct element in the case of convergence. Thus, fewer entries are needed

in the message of a symbol, i.e., the message can be truncated even more. These features are essential for reducing the computational complexity of decoding nonbinary LDPC codes.

The major decoding operation at a check node is to refine the estimated message of a symbol based on the messages of other symbols that are correlated by the PCM H. The elementary check node operation in the Min-Sum (MS) decoding algorithm [3], [4] can be expressed as

$$r=q1 \oplus q2 \quad (1)$$

Where q_1 and q_2 are the length 2^m variable node messages, and the basic operation is defined as $r^{\alpha} = \max(q1^{\beta}+q2^{\gamma})$, where r^{α}, $q1^{\beta}$,$q2^{\gamma}$ are the entries in messages r, q1,q2 corresponding to α,β,γ respectively.

On the other hand, each variable node improves the fidelity of belief information based on the received messages from multiple check nodes connected by the PCM. The elementary operation at a variable node can be expressed as [3], [4]

$$q = r1+ r2 \quad (2)$$

Which sums up the belief information associated with the same finite field element. The hardware complexity in implementing (1) and (2) is determined by the lengths of variable node messages (VNMs) and check node messages (CNMs). Intuitively, when the distribution of belief information is more concentrated, a shorter message might be sufficient to retain most of the belief information.

The basic idea of AMC is to keep as few entries in a message as possible without incurring much information loss. It has been demonstrated that message truncation can be implemented by finding the minimal n that satisfies

$$q(n+1) \le q(1) + \ln \quad (1-\zeta)/\zeta \quad (3)$$

Where ζ is the confidence factor that determines the tradeoff between performance and operational complexity, and q(k) indicates the kth entry in the log domain representation of the message q that is sorted in order.

In this case, the truncation criteria can be recast as

$$q(n+1) \le \ln (1-\zeta)/\zeta \quad (4)$$

where the threshold is used to truncate messages.

The operation of AMC in a nonbinary LDPC decoder can be summarized as follows.

Initialization

• Channel message truncation: $\dot{f}_j$ =AMC(f_j) where f_j is the received channel message.

• Variable node message: $q_{ij}= \dot{f}_j$, where q_{ij} is the variable node message from the variable node v_j to the check node c_i.

Iterations

• Permutation $q_{ij}^{\alpha} \rightarrow q_{ij}^{\alpha * hij}$, where h_{ij} is the nonzero PCM element, and the multiplication is conducted in GF(2^m).

• Check node update

$$r_{ij}= \bigoplus_{k \in M(i) \backslash j} q_{ij} \quad (5)$$

where M(i)\j is the set of neighbouring variable nodes of the check node c_i excluding the variable node v_j.

• Inverse permutation $r_{ij}^{\alpha} \rightarrow r_{ij}^{\alpha / hij}$, where h_{ij} is the nonzero PCM element, and the division is conducted in GF(2^m).

• Variable node update

$$q_{ij}=\text{AMC} (\dot{f}_j+ \sum_{k \in N(j) \backslash i}^{amc} r_{kj}) \quad (6)$$

where N(j)\i is the set of neighboring check nodes of the variable node v_j excluding the check node c_i, and $\sum^{AMC}$ means that the AMC is applied to all the intermediate results.
• Tentative decoding

$$c_j=\text{Max}\ (\hat{f}_j+\sum_{k\in N(j)} r_{kj}) \quad (7)$$

4. VLSI ARCHITECTURE FOR SEQUENTIAL AMC-BASED DECODER

In this section, a sequential nonbinary LDPC decoder architecture for the proposed AMC is presented. We first discuss the top level architecture and then detail the design of several key components such as the variable length sorter, variable node update unit (VNU), and check node update unite (CNU). The proposed AMC can be applied to different non binary LDPC decoding schedules, such as sequential, partially parallel and fully parallel architectures.

A. Horizontal Nonbinary LDPC Decoder Architecture

The design of horizontal nonbinary LDPC decoder employing with AMC is shown in figure 1. In this above architecture consists of variable node update unit, check node update unit, check sum unit, tentative decoding unit, permutation, inverse permutation and RAM. Random access memory is used to store the data. In this architecture RAM memory is divided into four divisions. RAMb is used to store the channel messages; RAMa is to store some bits, which are given to the RAMd to generate the intermediate check node messages .permutation is used to rearrange the order of inputs, which are coming from the variable node update unit. Inverse permutation is used to arrange the order of bits, which are from the check node update unit. Tentative decoding is used to do the iteration process. Check sum unit is used to check the channel messages with the values in that unit.

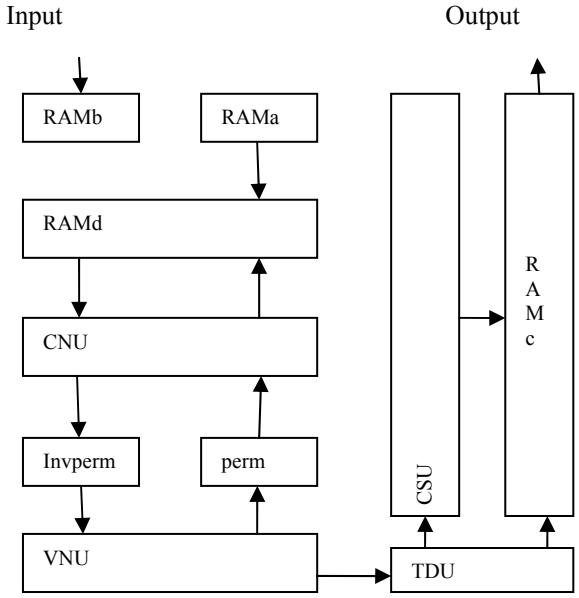

Figure 1. Block diagram of horizontal nonbinary LDPC decoder employing with AMC

At the beginning of the decoding, the truncated channel messages are loaded into the memory RAMb. Tentative decoding is conducted with the channel messages and the results are stored in the memory RAMc. If tentative decoding suceeds, decoding terminates and RAMc outputs the final result. Otherwise, the decoder initializes the intermediate check node messages (ICNMs) in the memory RAMd with the channel messages. After the initialization, the check node update unit (CNU) reads ICNMs from RAMd to perform the check node update. The CNMs from the CNU are inversely permuted, and then passed to the variable node update unit along with the associated channel message to perform variable node update. The variable

node update unit (VNU) also generates the messages for the tentative decoding unit (TDU) to conduct tentative decoding. If tentative decoding does not succeed, the VNMs from VNU are then permuted and sent back to the CNU to update the corresponding ICNMs, which are then stored in the RAMd. After this, the decoder proceeds to another variable node. This process continues until the check sum unit (CSU) decides to terminate because of either successful decoding or reaching the limit of iterations.

B. Variable Node Update Unit

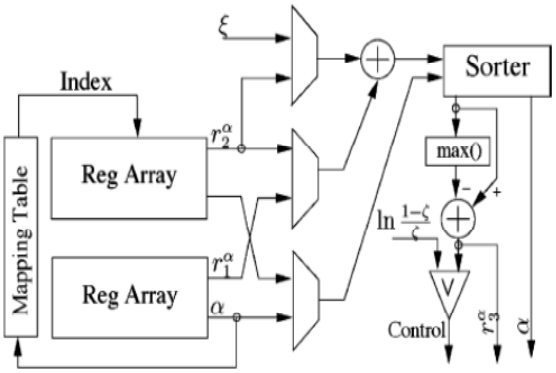

Figure 2. Implementation of the VNU unit

The design of VNU is shown in figure 2. The function of VNU is to compute two messages, where the belief information associated with the same Galois field element in the two messages is summed. Variable node update unit is used to truncate the message length. In this unit register array, mapping table, multiplexer, sorter, comparator are presented. The mapping table is utilized to sum up the entries associated with the same finite filed element. The final results are sent to the variable length sorter with the associated Galois field elements and AMC is then applied to the outputs of the sorter, starting from the largest entry in the sorter, when the entry is smaller than the threshold, the following entries are discarded. This reduces the hardware operations in the VNU.

C. Check Node Update Unit

The CNU produces the largest elements among all the combinations from two input messages. The proposed design of CNU is shown in figure 3. Figure shows that the complexity of CNU can be reduced to additions and insertion operations if the two input messages are sorted in the descending order. Here adopt this method in the CNU design. The complexity of the CNU depends on two factors.

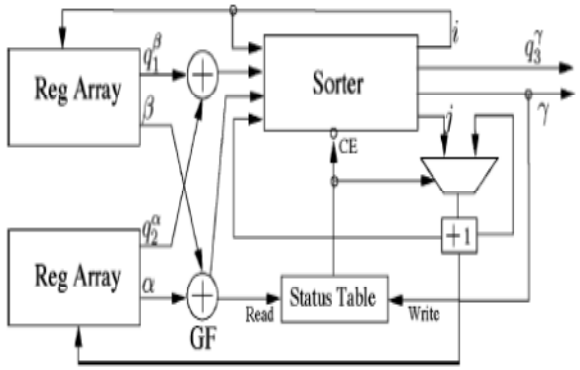

Figure 3. Implementation of the CNU unit

The first factor is the number of outputs, which determines how many insertion operations are needed. The second factor is the length of the sorter, which determines how many comparisons and shifting

operations are involved in each insertion operation. Employing the proposed AMC, the message length decreases thus becomes smaller during the iteration. This reduces the number of arithmetic operations and insertions. To address the second factor, the sorter is configured to be the same length of the shorter message thereby reducing the complexity of each insertion operation. The CNU also performs additions in the Galois field. However, additions in the Galois field are essentially bit-wise XOR operations, thus the complexity is much lower than the real number additions of belief information.

D. Variable Length Sorter

Sorting is an important operation in the CNU and VNU. In the CNU, the sorter chooses number of elements with the largest belief information as required for check node update. In the VNU, truncation is carried out and it discards the smaller elements and the remaining elements are sorted in order as the input of CNU.

The basic sorting operation is to insert a new element into a vector according to its magnitude. The proposed variable length sorter is illustrated in figure 4, where only the data path of belief information is shown as it determines the shifting operation. The sorter consists of number of stages to accommodate the longest message length. Each stage contains a comparator and a register. Each time new belief information comes in, it is compared with all the belief information in the active stages in parallel. The content of the first stage having larger belief information will be replaced by the new input, whereas its original belief information will be shifted to the right stage and so on. Sorter is an important operation in check node update unit and variable node update unit. Sorter consists of decoder, register, comparator, multiplexer, xor gate. The operation is parallel process. Employing AMC, the message length reduces gradually. Thus only the last stages are enabled. This is controlled by the longer one of the two messages in the check node update unit (CNU) or variable node update unit (VNU).

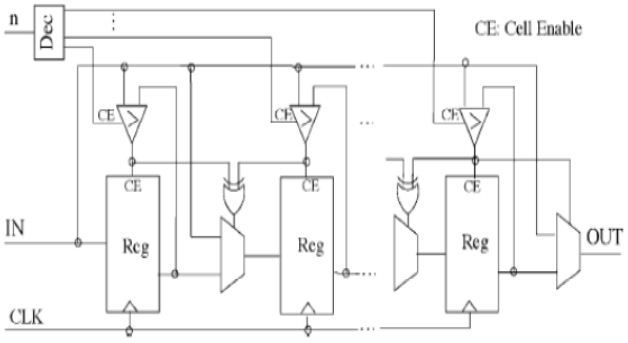

Figure 4. Variable length sorter

The major operations in the sorter are comparisons and shiftings. The number of these operations is mainly determined by the length of the message to be processed. The proposed AMC dynamically adjusts the message length during the iteration, thereby reducing the complexity of sorting.

E. Searching

Searching is another major operation in the check node update unit (CNU) and variable node update unit (VNU). In the VNU, the belief information in one message needs to search for its counterpart in another message to sums up the belief information. The output of the sorter is considered valid if and only if the corresponding Galois field element has not been generated previously. A searching operation has to be conducted on the current output to compare it with the previous outputs of the sorter.The mapping information between the Galois field elements and their indexes in a message is maintained by a mapping table of words with word length of bits. In the CNU, number of different elements need to be generated according to variable node messages. The output of the sorter is considered valid if and only if the corresponding Galois field element has not been generated previously. A searching operation has to be conducted on the current output to compare it with the previous outputs of the sorter. Mapping table is to records the status of Galois field elements; it is referred to as status table in the CNU. The existence of the element needs to be determined; a mapping table with only number of bits can be constructed. The proposed

mapping-based searching scheme is naturally low complexity in comparison with direct searching of the message, especially when the message is very long.

5. RESULT AND DISCUSSION

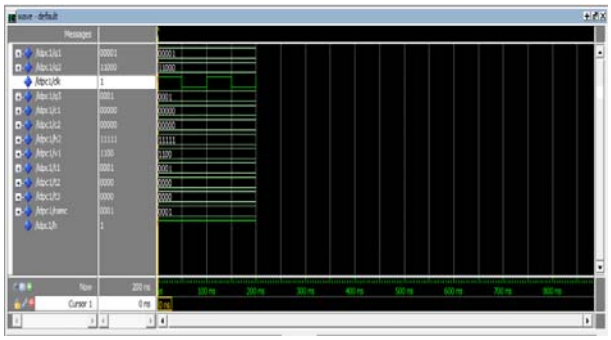

Figure 5. Horizontal nonbinary LDPC decoder employing with AMC

Table 1. Comparision Result For Various Methods

Parameter	**MS**	**EMS**	**AMC-EMS**
Memory Size	323232	214368	160476
Power	1	0.7	56mW

6. CONCLUSION

A new nonbinary low density parity check (LDPC) decoding architecture is proposed to reduce hardware operation and power consumption. A horizontal sequential VLSI architecture for the nonbinary LDPC decoder employs the AMC. The design of the key components in this architecture, such as variable node and check node update units, is optimized by exploiting the variable length sorters, which can be dynamically configured in different functional units to accommodate variable message lengths. The AMC is implemented by a low-complexity approximation method to avoid hardware overheads and performance impact. Here Adaptive Message Control method is used to reduce the message length. Thus the message length is reduced by applying adaptive message control to the variable node update unit, then the truncation is achieved. Simulation results and synthesis reports are analyzed.

REFERENCES

[1] R.G. Gallager. *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press. 1963.
[2] D.J.C. MacKay and R. Neal. "*Good codes based on very sparse matrices*". In Proc. Cryptography Coding, 5th IMA Conf. 1995, pp. 100–111.
[3] D. Declercq and M. Fossorier. "Decoding algorithms for nonbinary LDPC codes over GF(q)". *IEEE Trans. Commun.* vol. 55, no. 4, pp. 633–643, Apr. 2007.
[4] A.Voicila, D. Decercq, F. Verdier, M. Fossorier, and P. Urard. "*Low compleixty, low memory EMS algorithm for non-binary LDPC codes*". In Proc. ICC. 2007, pp. 671–676.
[5] J. Huang, S. Zhou, and P. Willet. "Nonbinary LDPC coding for multicarrier underwater acoustic communication". *IEEE J. Sel. Areas Commun.* vol. 26, no. 9, pp. 1684–1696, Sep. 2008.
[6] W. Tang, J. Huang, L. Wang, and S. Zhou. "*Nonbinary LDPC decoding by Min-Sum with adaptive message control*". In Proc. Int. Conf. Acoust. Speech, Signal Process. (ICASSP). 2011, pp. 3164–31.