International Journal For Technological Research In Engineering
Volume 4, Issue 5, January-2017 *ISSN (Online): 2347 - 4718*

AREA AND DELAY EFFICIENT DESIGN OF BINARY LDPC ARCHITECTURE

S.Hareesh[1], Mohd Shahbaz Khan[2], Dr M.Gurunadha Babu[3]
[1]PG Scholar, [2]Associate Professor, [3]Professor, HOD,
Dept of ECE, CMR Institute of Technology, Kandlakoya, Medchal, Ranga Reddy, Telangana.

Abstract: We address the problem of decoding non binary low-density parity-check (LDPC) codes over finite fields GF (q), with reasonable complexity and good performance. A one minimum only decoder for Trellis-EMS (OMO T-EMS) and for Trellis-Min-max (OMO T-MM) is projected in this paper. during this novel approach, we tend to avoid computing the second minimum in messages of the check node processor, and propose efficient estimators to infer the second minimum worth. By doing so, we greatly scale back the quality and at a similar time improve latency and turnout of the derived architectures compared to the prevailing implementations of EMS and Min-max decoders. This resolution has been applied to numerous NB-LDPC codes constructed over totally different Galois fields and with different degree distributions showing altogether cases negligible performance loss compared to the best EMS and Min-max algorithms.. By choosing appropriate correction factors or offsets, we show that the EMS decoder performance is quite good, and in some cases better than the regular BP decoder. The optimal values of the factor and offset correction are obtained asymptotically with simulated density evolution. Our simulations on ultra-sparse codes over very-high-order fields show that non binary LDPC codes are promising for applications which require low frame-error rates for small or moderate codeword lengths. The EMS decoder is a good candidate for practical hardware implementations of such codes.

I. INTRODUCTION

Low density parity check (LDPC) codes designed over GF (q) (also referred to as GF(q)-LDPC codes) have been shown to approach Shannon limit performance for q = 2 and very long code lengths [1, 2, 4, 5]. On the other hand, for moderate code lengths, the error performance can be improved by increasing q [6, 7]. However this improvement is achieved at the expense of increased decoding complexity. A straightforward implementation of the belief propagation (BP) algorithm to decode GF (q)-LDPC codes has computational complexity dominated by O (q 2) operations for each check sum processing. As a result, no field of order larger than q = 16 was initially considered. Extending the ideas presented, a more efficient approach using Fourier transforms over GF (2q) was presented. The description of this algorithm in the log-domain has been given. Note that the Fourier transform is easy to compute only when the Galois field is a binary extension field with order q = 2p. In that case, this approach allows to reduce the computational complexity of the BP algorithm to O (p2 p). Consequently, results for 2 p = 256 were reported with this method. The formulation of this algorithm was further elegantly and conveniently modified based on the introduction of a tensoral representation. With this representation, the generalization of BP decoding over GF (2) to any field of order q = 2p becomes very natural. We present in details the BP algorithm using tensoral notations in the first part of this paper. Simplified iterative decoding of GF (q)-LDPC codes have also been investigated. For q = 2, the min-sum (MS) algorithm with proper modification has been shown to result in negligible performance degradation (less than 0.1 dB for regular LDPC codes) while performing additions only, and becoming independent of the channel conditions. Extension of this approach to any value q seems highly attractive.

Unfortunately, such extensions are not straightforward as many simplifications cannot be realized in conjunction with Fourier transforms. Presents a log-domain BP decoder combined with a FFT at the check node input. However combining log-values and FFT requires a lot of exponential and logarithm computations, which may not be very practical. To overcome this issue, the authors propose the use of a look-up table (LUT) to perform the required operations. Although simple, this approach is of limited interest for codes over high order fields since the number of LUT accesses grows in q log2 (q) for a single message. As a result, for fields of high order, unless the LUT has a prohibitively large size, the performance loss induced by the LUT approximation is quite large. The MS algorithm is extended to any finite field of order q. Although only additions are performed and no channel information is necessary, its complexity remains O (q 2). As a result, only small values of q can be considered by this algorithm and for q = 8, a degradation of 0.5 dB over BP decoding is reported. Simplifications of BP decoding of GF (q)-LDPC codes have also been considered for non binary signaling. In this paper, we develop a generalization of the MS algorithm which not only performs additions without the need of channel estimation, but also with the two following objectives: (i) a complexity much lower than O(q 2) so that finite fields of large order can be considered; and (ii) a small performance degradation compared with BP decoding. The first objective is achieved by introducing configuration sets, which allow to keep only a small number of meaningful values at the check node processing (note that while the check node processing is O (q 2), that of the variable node processing is only O (q)). The second objective is achieved by applying at the variable node processing the correction techniques of to the proposed algorithm.

II. LITERATURE SURVEY

Min-Max decoding for non binary LDPC codes

Iterative decoding of non-binary LDPC codes is currently performed using either the Sum-Product or the Min- Sum algorithms or slightly different versions of them. Several low-complexity quasi-optimal iterative algorithms are proposed for decoding non-binary codes. The Min-Max algorithm is one of them and it has the benefit of two possible LLR domain implementations: a standard implementation, whose complexity scales as the square of the Galois field’s cardinality and a reduced complexity implementation called selective implementation, which makes the Min-Max decoding very attractive for practical purposes.

Simplified Trellis Min-Max Decoder Architecture for Non-Binary Low-Density Parity-Check Codes

Non-binary Low-Density Parity-Check (NB-LDPC) codes have become an efficient alternative to their binary counterparts in different scenarios such as: moderate codeword lengths, high order modulations and burst error correction. Unfortunately, the complexity of NB-LDPC decoders is still too high, especially for the check node processing, which limits the maximum throughput achievable. Although a great effort has been expended to overcome this disadvantage, the decoders presented in literature are still away from high speed implementations for high order fields. In this paper a simplified Trellis Min-Max (TMM) algorithm is proposed, where the check node messages are computed in a parallel way using only the most reliable information. The proposed check node algorithm is implemented using an horizontal layered schedule. The complete decoder architecture has been implemented in a 90 nm CMOS process for the (837,726) NB-LDPC code over GF (32), achieving a throughput of 660 Mbps at 9 iterations based on post layout results. This decoder increases hardware efficiency in 110% compared to the existing solutions for the same code.

III. EXISTING WORK

Notations related to the Galois field: • GF(q) = {0, 1, . . ., q−1}, the Galois field with q elements , where q is a power of a prime number. Its elements will be called symbols, in order to be distinguished from ordinary integers. • a, s, x will be used to denote GF (q)-symbols. • a , s , x will be used to denote vectors of GF (q)-symbols. For instance, a = (a 1, . . . , a I) ∈ GF (q) I , etc. Notations related to LDPC codes: • H ∈ MM,N (GF(q)), the q-ary check matrix of the code. • C , set of codewords of the LDPC code. • Cn(a), set of codewords with the n th coordinate equal to a; for given 1 ≤ n ≤ N and a ∈ GF (q) . • x = (x 1, x 2, . . . , x N) a q-ary codeword transmitted over the channel. The Tanner graph associated with an LDPC code consists of N variable nodes and M check nodes representing the N columns and the M lines of the matrix H. A variable node and a check node are connected by an edge if the corresponding element of matrix H is not zero. Each edge of the graph is labeled by the corresponding non zero element of H. Notations related to the Tanner graph: • H, the Tanner graph of the code. • n ∈ {1, 2, . . ., N} a variable node of H . • m ∈ { 1 , 2, . . ., M } a check node of H . • H (n), set of neighbor check nodes of the variable node n . • H (m), set of neighbor variable nodes of the check node m . • L (m), set of local configurations verifying the check node m ; i.e. the set of sequences of GF (q)-symbols a = (a n) n∈H (m) , verifying the linear constraint: Xn∈H(m) $h_{m,n}$ an = 0 • L (m | a n = a), set of local configurations a verifying m , such that a n = a; for given n ∈ H (m) and a ∈ GF (q) . 2 An iterative decoding algorithm consists of an initialization step followed by an iterative exchange of messages between variable and check nodes connected in the Tanner graph. Notations related to an iterative decoding algorithm: • γn(a), the a priori information of the variable node n concerning the symbol a. • γ˜n(a), the a posteriori information of the variable node n concerning the symbol a. • αm,n(a), the message from the check node m to the variable node n concerning the symbol a. • βm,n(a), the message from the variable node n to the check node m concerning the symbol a.

REALIZATIONS OF THE MIN-SUM DECODING FOR NON BINARY LDPC CODES

A. Min-Sum decoding

The Min-Sum decoding is generally implemented in the log probability domain and it performs the following operations: Initialization • A priori information: γn(a) = − ln (Pr(xn = a | channel)) • Variable node messages: αm,n(a) = γn(a) Iterations • Check node processing

$$\beta_{m,n}(a) = \min_{\substack{(a_{n'})_{n'\in\mathcal{H}(m)}\\ \in\mathscr{L}(m|a_n=a)}} \left(\sum_{n'\in\mathcal{H}(m)\setminus\{n\}} \alpha_{m,n'}(a_{n'}) \right)$$

Variable node processing

$$\alpha_{m,n}(a) = \gamma_n(a) + \sum_{m'\in\mathcal{H}(n)\setminus\{m\}} \beta_{m',n}(a)$$

A posteriori information

$$\tilde{\gamma}_n(a) = \gamma_n(a) + \sum_{m\in\mathcal{H}(n)} \beta_{m,n}(a)$$

For practical purposes, messages _m,n (a) and _m,n (a). Should be normalized in order to avoid computational instability (otherwise they could “escape” to infinity). The check node processing, which dominates the decoding complexity, can be implemented using a forward – backward computation method.

B. Equivalent iterative decoders

The term of *equivalent (iterative) decoders* will be employed several times through this paper. We begin this section by providing its rigorous definition. Consider the a posteriori information available at a variable node n after the lth decoding iteration: it defines an order between the symbols of the Galois field, starting with the most likely symbol and ending with the least likely one. Note that the most likely symbol may correspond to the minimum or to the maximum of the a posteriori information, depending on the decoding algorithm.

IV. PROPOSED WORK

Let us define the parity check matrix H with M rows and N columns. Each non-zero element hm, n of H belongs to the Galois field GF (q = 2 p). In this paper, we only consider regular NBLDPC codes with constant row weight dc and column weight dc. Let N (m) (M (n)) be the set of variable nodes (check nodes) connected to a check node (variable node) m (n). Let Qm,n (a) and Rm,n (a) be the messages from variable node to check node and from check node to variable node for each symbol a ∈ GF(q) respectively. Ln(a) denotes the channel information and Qn (a) the a posteriori information. Let c = c1, c2,⋯ , cN and y = y1, y2,⋯ , yN be the transmitted codeword and received symbol sequence respectively, with y = c+e and e is the error vector introduced by the communication channel. The log-likelihood ratio (LLR) for each received symbol is obtained as Ln(a) = log[P(cn = zn|yn)/P(cn = a|yn)] where zn is the symbol associated to the highest reliability. The previous definition ensures that all messages Ln(a) are non-negative and that the smaller the value, the more reliable the message. Algorithm 1 includes the T-EMS check node algorithm where the first step consists in the delta domain transformation of input messages. This transformation ensures that the most reliable messages are always in the first row of ΔQm,n(ηj) and the rest of the symbols are reordered and considered as deviations of the most reliable one, according to step 1. Step 2 involves the calculus of check node's syndrome β using the most reliable symbol zn for each check node incoming message.

A. Estimators for the Second Minimum Value

A first natural solution for the estimation of min2 is to make use of a scaled version of the first minimum, min1 described in

$$min''_2(a) = min_1(a) \times \gamma_p$$

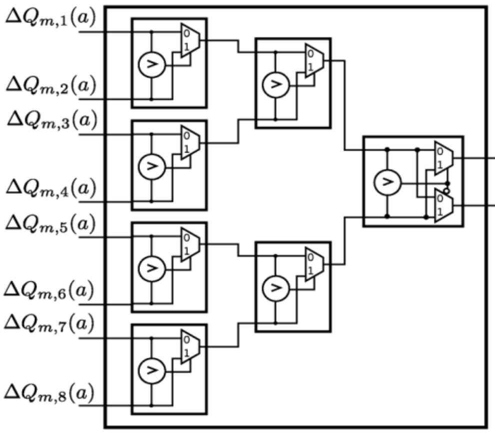
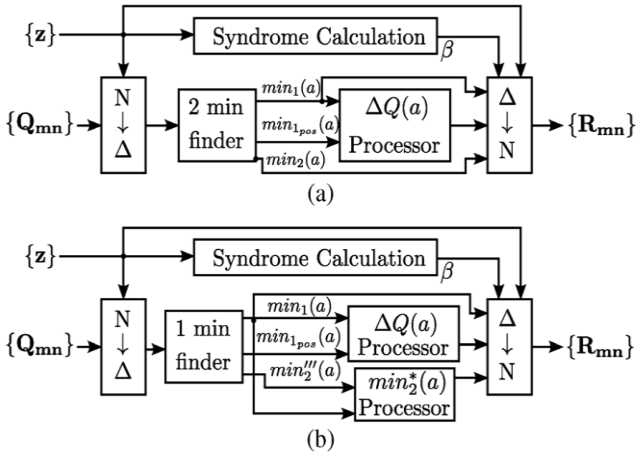


Fig. 1. Second minimum estimation based on a radix-2 one-minimum finder. Example for an eight inputs tree.

This approximation has been already proposed. However, by just applying the value of the minimum is usually Under estimated if we apply a value that mimics as much as possible the behavior of EMS or Min-max in the waterfall region. min1, where we draw the distributions of the true and their proposed estimators, the value of is on average smaller than the real, which leads to an important performance degradation in the error floor region. A second possible estimator makes advantage of a re-use of the hardware architecture. Using a radix-2 one-minimum finder is possible to determine an early estimation for the second minimum.

A one-minimum tree finder is presented. We include an extra multiplexor in the last stage, that allows extracting the looser term, denoted. By doing so and just using an extra multiplexor, this term can be used as an early estimator of the second minimum, which represents an upper-bound on the true minimum value. If the true value is located in the other half part of the tree that (branches of the minimum tree finder not connected to), then we obtain. In the other cases. Hence, the resultant value corresponds to provable upper bound on the true. A systematic overestimation of the second minimum value could lead also to performance degradation of the complete decoder, and we propose to combine and in order to get an estimator with a better statistical behavior.

$$min^*_2(a) = \frac{min''_2(a) + min'''_2(a)}{2} = \frac{min_1(a) \times \gamma_p + min'''_2(a)}{2}$$



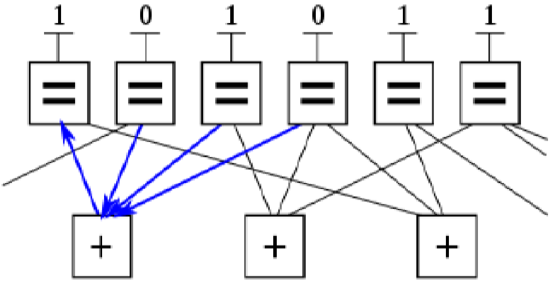
(a)

(b)

Fig. 2. Check node top architecture for T-EMS algorithm (a). Proposed OMO T-EMS/OMO T-MM check node architecture (b).

Decoding:

As with other codes, optimally decoding an LDPC code on the binary symmetric channel is the NP-complete problem, although techniques based on iterative belief propagation used in practice lead to good approximations. In contrast, belief propagation on the binary erasure channel is usually simple where it consists of iterative constraint satisfaction. For example, consider that the valid code word, 101011, from the example, is transmitted across a binary erasure channel and received with first and fourth bit erased to yield. Since the transmitted message must have full fill the code constraints, the message can be organized by written the message on the top of the factor graph. In this example, the first bit cannot yet be recovered, because all of the constraints connected to it have more than one unknown bit. In order to proceed with decoding the message, this procedure is then iterated. The new value for the fourth bit can now be used in conjunction with the first constraint to recover the first bit as shown below. This means that the first bit must be a 1 to satisfy the leftmost constraint.



Thus, the message can be decoded iteratively. For next channel models, the messages passed inside the variable nodes and check nodes are real numbers, which express probabilities and likelihoods of belief. This result can be validated by multiplying the corrected code word by the parity-check matrix H:

Because the outcome z (the syndrome) of this operation is the 3 × 1 zero vector, the resulting code word is successfully validated.

$$z = Hr = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Encoder:

Encoder uses generator matrix to encode the information bits in to the code word. Both generator and parity check matrix are interrelated, parity check matrix is given by

$$H = [\,PT \mid I\,] \text{ (or) } H = [\,In\text{-}P \mid I\,]$$

and the generator matrix is given by

$$G = [\,I \mid PT\,]$$

Initially parity check matrix is generated; using that matrix generator matrix is created by Gaussian elimination method. There are two types of parity matrices in LDPC coding one is Regular matrix and another one is irregular matrix. Regular matrix is one in which column Wc is same for all columns and row weight is given by

Wr = Wc(n/m)

we are using regular matrix of 3X7 (or) (n,k)=(7,3) i.e., where n represents total bits and k represents message bits, n-k=7-3=4 which represents check bits or parity bits.

$$\begin{pmatrix} 1011001 \\ 1110100 \\ 1100010 \\ 0110001 \end{pmatrix}$$

Regular Parity Matrix

To transfer the above parity check matrix to standard form i.e H=[PT | I] Gaussian elimination method is applied to the above matrix. The matrix H is put into this form by applying elementary row operations which are interchanging two rows or adding one row to another modulo 2. The resulting parity matrix in its standard form H is as shown in the figure

$$\begin{pmatrix} 1011000 \\ 1110100 \\ 1100010 \\ 0110001 \end{pmatrix}$$

Standard Parity Matrix

If G is the generated matrix for (n, k) code then H is the generator matrix for (n, n-k) code. Therefore obtained parity matrix is translated to standard form generator matrix i.e., G =[I | P] as shown in fig

$$\begin{pmatrix} 0001110 \\ 0100111 \\ 0011101 \end{pmatrix}$$

Generator Matrix

Now the information message bits are encoded by multiplying it with above generator matrix i.e., C = [M][P] to obtain the codeword. Each structure labeled G{0,1,,,m-1},i are XOR structures performs modulo-2 operations on the incoming message bits and the resultant code words will be of N-bits. Let us consider an n-bit information message U = [101], and C is given by

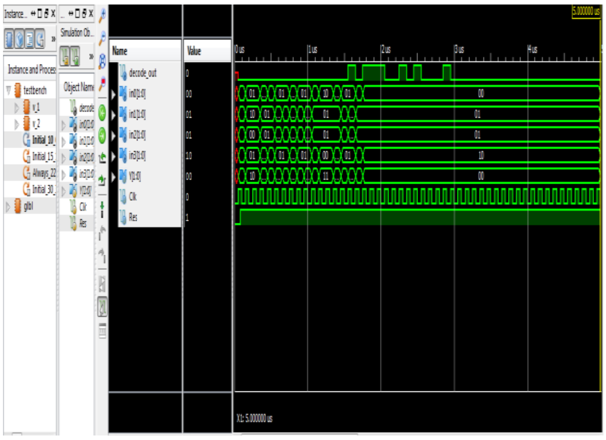
$$[101] \quad \begin{pmatrix} 1110 \\ 0111 \\ 1101 \end{pmatrix}$$

By multiplying message vector with generator matrix we obtain the codeword with parity (or) check bits C = [1 0 1 0 0 1 1]. Coding for this encoder part is done on VHDL and encoding is tested for various information bits satisfactorily.

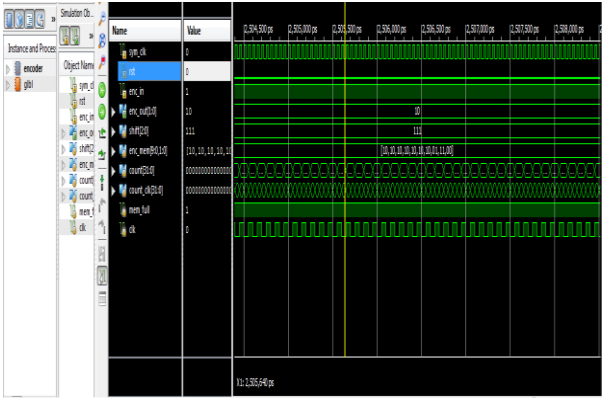
V. RESULTS

Simulation results

Decoder:



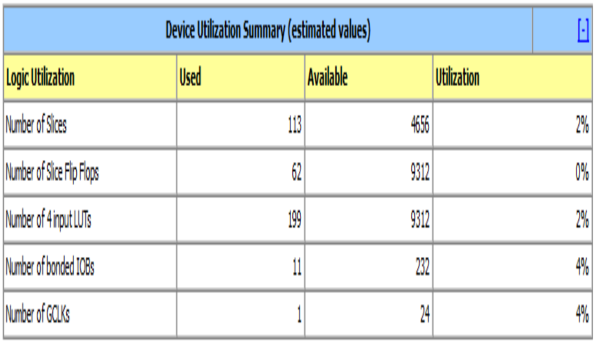
Encoder;



Synthesis results:

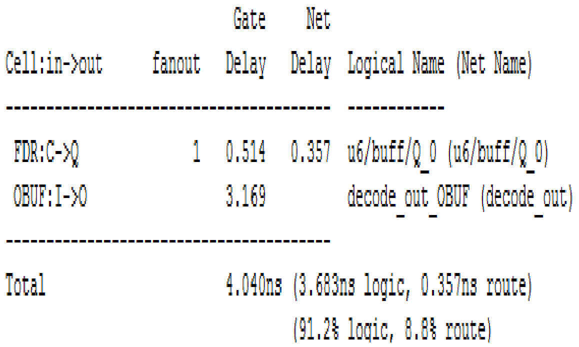
Decoder:

Design summary



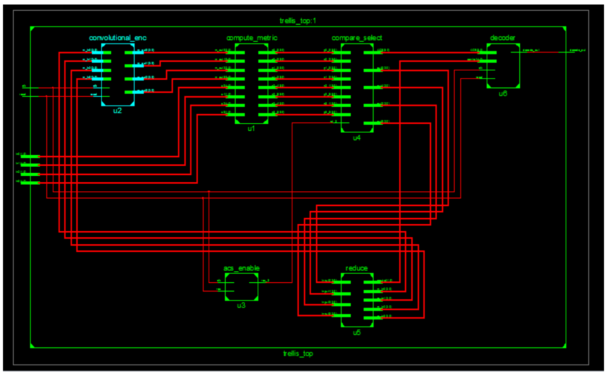
Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	113	4656	2%
Number of Slice Flip Flops	62	9312	0%
Number of 4 input LUTs	199	9312	2%
Number of bonded IOBs	11	232	4%
Number of GCLKs	1	24	4%

Timing Report:



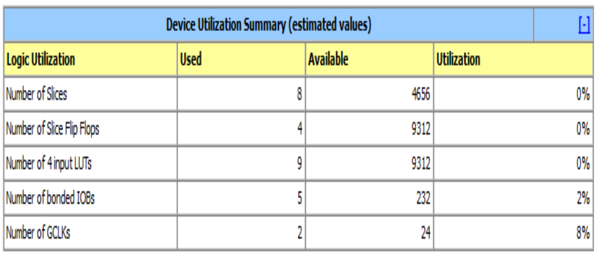
```
                              Gate    Net
Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
----------------------------------------  ------------
 FDR:C->Q              1   0.514   0.357  u6/buff/Q_0 (u6/buff/Q_0)
 OBUF:I->O                 3.169          decode_out_OBUF (decode_out)
----------------------------------------
Total                      4.040ns (3.683ns logic, 0.357ns route)
                                   (91.2% logic, 8.8% route)
```

RTL schematic



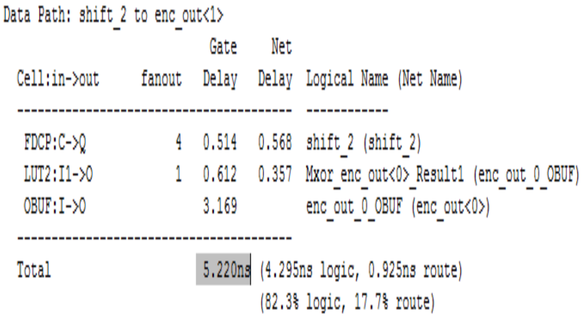
Encoder:

Design summary:



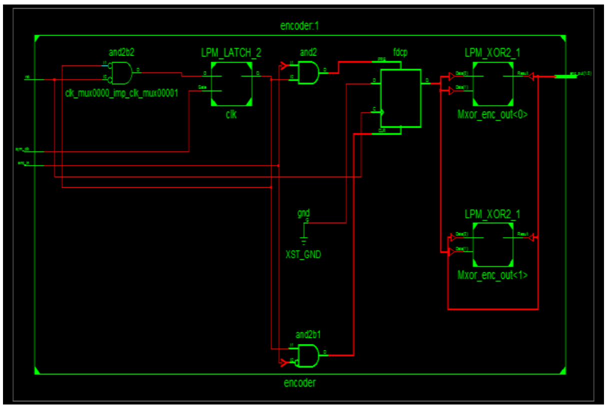
Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	8	4656	0%
Number of Slice Flip Flops	4	9312	0%
Number of 4 input LUTs	9	9312	0%
Number of bonded IOBs	5	232	2%
Number of GCLKs	2	24	8%

Timing Report:



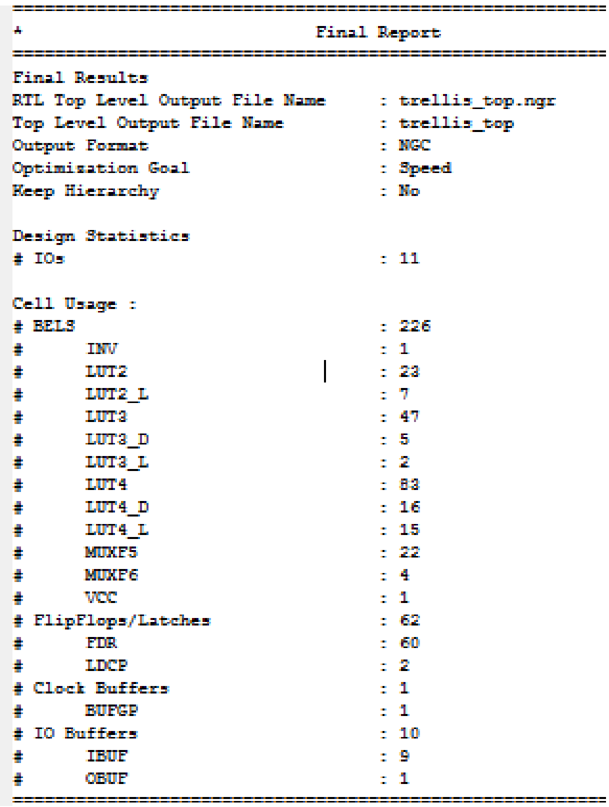
```
Data Path: shift_2 to enc_out<1>
                               Gate    Net
  Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   FDCP:C->Q             4   0.514   0.568  shift_2 (shift_2)
   LUT2:I1->O            1   0.612   0.357  Mxor_enc_out<0>_Result1 (enc_out_0_OBUF)
   OBUF:I->O                 3.169          enc_out_0_OBUF (enc_out<0>)
  ----------------------------------------
  Total                      5.220ns (4.295ns logic, 0.925ns route)
                                     (82.3% logic, 17.7% route)
```

RTL Schematic:



DECODER-CELL USAGE:

The DECODER architecture presented in this paper achieves a high performance in terms of both area and delay. This not only reduces the latency but also is increases the throughput rate. Our simulations on ultra-sparse codes over very-high-order fields show that non binary LDPC codes are promising for applications which require low frame-error rates for small or moderate codeword lengths. The EMS decoder is a good candidate for practical hardware implementations of such codes.



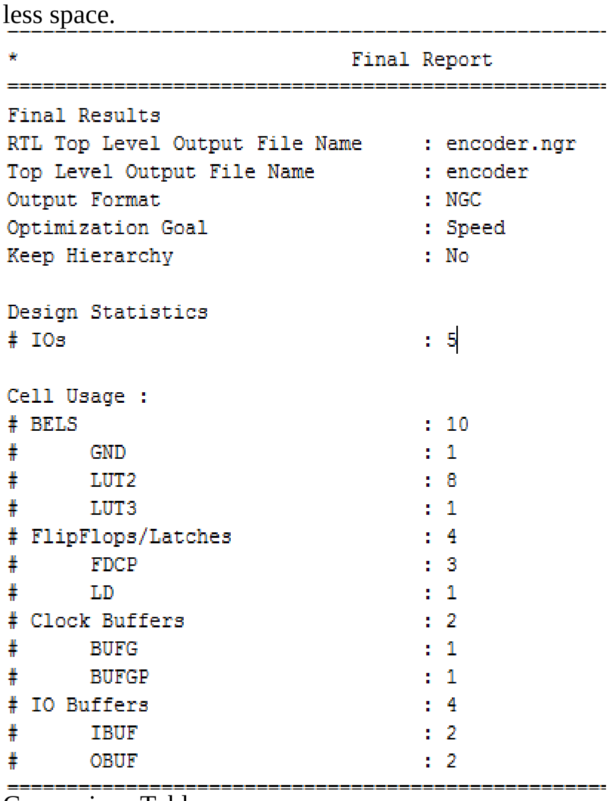
```
=========================================================================
*                           Final Report
=========================================================================
Final Results
RTL Top Level Output File Name     : trellis_top.ngr
Top Level Output File Name         : trellis_top
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : No

Design Statistics
# IOs                              : 11

Cell Usage :
# BELS                             : 226
#      INV                         : 1
#      LUT2                        : 23
#      LUT2_L                      : 7
#      LUT3                        : 47
#      LUT3_D                      : 5
#      LUT3_L                      : 2
#      LUT4                        : 83
#      LUT4_D                      : 16
#      LUT4_L                      : 15
#      MUXF5                       : 22
#      MUXF6                       : 4
#      VCC                         : 1
# FlipFlops/Latches                : 62
#      FDR                         : 60
#      LDCP                        : 2
# Clock Buffers                    : 1
#      BUFGP                       : 1
# IO Buffers                       : 10
#      IBUF                        : 9
#      OBUF                        : 1
=========================================================================
```

ENCODER-CELL USAGE

The proposed LDPC encoder has better output performance in terms of area when compared to that of the decoder architecture. The device utilization summary evaluated from the synthesis results clearly prove that the encoder occupies less space.

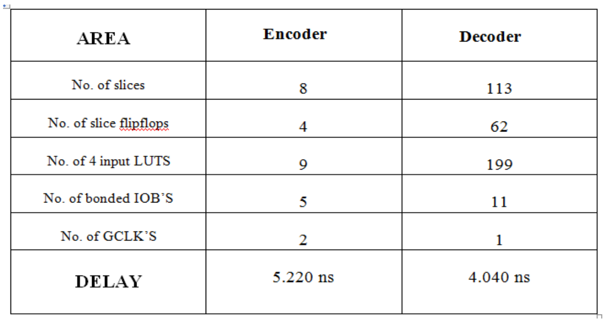


```
---------------------------------------------------------------------
*                           Final Report
=====================================================================
Final Results
RTL Top Level Output File Name     : encoder.ngr
Top Level Output File Name         : encoder
Output Format                      : NGC
Optimization Goal                  : Speed
Keep Hierarchy                     : No

Design Statistics
# IOs                              : 5

Cell Usage :
# BELS                             : 10
#      GND                         : 1
#      LUT2                        : 8
#      LUT3                        : 1
# FlipFlops/Latches                : 4
#      FDCP                        : 3
#      LD                          : 1
# Clock Buffers                    : 2
#      BUFG                        : 1
#      BUFGP                       : 1
# IO Buffers                       : 4
#      IBUF                        : 2
#      OBUF                        : 2
=====================================================================
```

Comparison Table:



AREA	Encoder	Decoder
No. of slices	8	113
No. of slice flipflops	4	62
No. of 4 input LUTS	9	199
No. of bonded IOB'S	5	11
No. of GCLK'S	2	1
DELAY	5.220 ns	4.040 ns

VI. CONCLUSION

In this paper a new method to estimate the second minimum value in message of the check node processor of NB-LDPC de-coders is proposed. This solution avoids the use of two-minimum finders, greatly reducing the check node complexity. The outgoing check node messages are calculated in a parallel way using only the most reliable symbols, reducing the overhead of the CN by a factor of four compared to the TEMS decoder. Using the layered schedule with the proposed check node algorithm reduces the required maximum number of iterations to achieve a desired performance. On the other hand, since the proposed approach does not make approximations on the reliability values used for derive the check node messages, the performance of the algorithm does not show any performance degradation. The simplifications applied to the T-EMS and T-MM algorithms reduce latency and area with respect to the original proposal, without introducing any significant performance loss.

REFERENCES

[1]. M. Davey and D. MacKay, “Low-density parity check codes over GF (q),” IEEE Commun. Lett., vol. 2, no. 6, pp. 165–167, 1998.

[2]. D. Declercq and M. Fossorier, “Decoding algorithms for nonbinary LDPC codes over GF (q),” IEEE Trans. Commun., vol. 55, no. 4, pp. 633–643, 2007.

[3]. V. Savin, “Min-max decoding for non binary LDPC codes,” in Proc. IEEE Int. Symp. Inf. Theory, 2008, pp. 960–964.

[4]. E. Li, K. Gunnam, and D. Declercq, “Trellis based extended min-sum for decoding nonbinary LDPC codes,” in Proc. Proc. 8th Int. Symp. Wireless Commun. Syst. (ISWCS), 2011, pp. 46–50.

[5]. E. Li, D. Declercq, and K. Gunnam, “Trellis-based extended min-sum algorithm for non-binary LDPC codes and its hardware structure,” IEEE Trans. Commun., vol. 61, no. 7, pp. 2600–2611, 2013.

[6]. Mansour.M .MandShanbhag.N.R, “High-throughput LDPCdecoders,”IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11,no. 6, pp. 976–996, Dec. 2003.

[7]. MC Davey and DJC MacKay, “Low density parity check codes over GF (q),” Information Theory Workshop, 1998, pp. 70–71, 1998.

[8]. X.Y. Hu and E. Eleftheriou, “Cycle Tanner-graph codes over GF(2b),” Information Theory, 2003. Proceedings. IEEE International Symposium on.