A Reduced Routing Network Architecture for Partial Parallel LDPC Decoders

Houshmand Shirani-Mehr[1], Tinoosh Mohsenin[2] and Bevan Baas[1]

[1] ECE Department, University of California, Davis, [2] CSEE Department, University of Maryland Baltimore County

***Abstract*— A novel partial parallel decoding scheme based on the matrix structure of LDPC codes proposed in IEEE 802.15.3c and IEEE 802.11ad standards is presented that significantly simplifies the routing network of the decoder, and the class of parity-check matrices for which the method can be used is defined. The proposed method results in an almost complete elimination of logic gates on the routing network, which yields improvements in area, speed and power, with an identical error correction performance to conventional partial-parallel decoders. A decoder for the (672,588) LDPC code adopted in the IEEE 802.15.3c is implemented in a 65 nm CMOS technology including place & route with both proposed permutational decoder, and conventional partial-parallel architecture. The proposed permutational LDPC decoder operates at 235 MHz and delivers a throughput of 7.9 Gbps with 5 decoding iterations per block. The proposed permutational decoder has a throughput 30% higher and is approximately 24% smaller than the conventional partial-parallel decoder.**

I. Introduction

Low density parity check codes (LDPC) were first introduced by Gallager in 1962 [1]. Their superior error correction performance and highly parallelizable decoding algorithms have resulted in adoption in many recent communication standards, including 10 Gigabit Ethernet (10GBASE-T) [2], digital video broadcasting (DVB-S2) [3], 60 GHz Wi-Fi (IEEE 802.11ad) [4] and 60 GHz WPAN (IEEE 802.15.3c) [5]. However, LDPC decoders exhibit high interconnection complexity, which presents challenges for efficient hardware implementations that meet throughput requirements of emerging communication systems [6]. LDPC decoding architectures can be divided into two main categories: full-parallel and partial-parallel.

In full-parallel decoders, every processing node is implemented in hardware and these nodes are connected through global wires based on the parity-check matrix [7]. They have the highest theoretical throughput [8], but the global interconnection complexity results in low hardware utilization, high circuit area and lower than exptected throughput improvements [9]. Consequently, these decoders are not widely used unless very high throughputs are required.

In partial-parallel architectures, a subset of check nodes and variable nodes of the parity check matrix is implemented in hardware, and by employing memory units and changing routing network between nodes, the update stage for different partitions of the matrix are processed. Adjusting the interconnection network to different partitions is achievable by utilizing Permuter networks [10], or in general a network of muxes [11]. The objective of these networks is to change the path of every bit of messages transmitted between check nodes and variable nodes in different cycles. Therefore, a significant number of muxes is required, which results in a substantial hardware overhead, and a considerable power dissipation due to constant toggling over cycles. Additionally, since these muxes are in the critical path of the signals passing through decoder, a decline in the throughput is observed.

Alleviating the interconnection complexity by improving implementation techniques based on properties of structured LDPC codes [12], or improving the switching networks in general [13], has been a constant field of research in recent years. However, still a considerable number of logic gates are used to change the configuration of routing network between cycles. In [14] a decoder supporting four LDPC codes in IEEE 802.15.3c requires 63,168 multiplexer inputs for the routing network.

In this work, a new decoding scheme based on the matrix structure of LDPC codes proposed for IEEE 802.15.3c and IEEE 802.11ad standards is presented. This new method results in almost complete elimination of logic gates on the routing network of the decoder, and provides significant improvements in area, throughput and power, with no degradation in BER performance. The scheme is based on the layered belief propagation (LBP) algorithm [15], which improves the speed of convergence by a factor of two compared to regular belief propagation decoding [16]. The class of matrices for which the decoder can be used is described, and the proposed scheme is implemented in 65 nm CMOS technology including place & route for the (672,588) LDPC code adopted in IEEE 802.15.3c standard. Finally, the results are compared to a conventional partial-parallel decoder designed for the same code.

The paper is organized as follows: Section 2 describes the layered belief propagation algorithm with Min-Sum; Section 3 introduces the generalized set of parity-check matrices for which the method can be utilized; Section 4 presents a detailed description of decoding scheme and its properties; Section 5 proposes the hardware implementation of the decoding scheme on (672,588) code used in IEEE 802.15.3c standard and the comparison of the results; Finally, Section 6 concludes the paper.

II. Layered Belief Propagation Decoding

An LDPC code is uniquely defined by a binary $M \times N$ parity check matrix H. The number of rows in the matrix, M, is equal to the number of check nodes in the decoder, the number of columns, N, is the number of variable nodes, and the 1's in the matrix determine how check nodes and variable nodes are connected. In horizontal layered decoding, the parity check matrix rows are divided into a number of layers, Y, and the variable node messages are updated after processing each layer during one iteration. In this work, layered scheduling with normalized Min-Sum [17] as the update procedure in the check nodes is utilized.

The following definitions are used throughout the paper:

λ_j Log-likelihood ratio of channel information (*a priori value*) for j-th variable node.

R_{ij} Message from check node i to variable node j.

Q_{ij} Message from variable node j to check node i.

Q_j Sum of check node messages and channel information in variable node j (*a posteriori probability ratio*).

The set of variable nodes connected to check node i is denoted by $V(i)$, and this set excluding variable node j is shown by $V(i)\backslash j$. The decoding algorithm is summarized in the following steps:

1) *Initialization:* Q_j values are initialized by the log-likelihood ratio of channel information (λ_j), and all the messages between check nodes and variable nodes are set to zero.

978-1-4673-0323-1/11/$26.00 ©2011 IEEE

2) *Processing of layers:* Assume $L_0, L_1, ..., L_{Y-1}$ to be the layers of the matrix, then for every decoding iteration:

```
for $k = 0 : (Y-1)$ do
    for $i \in$ check nodes of $L_k$ do
        $Q_{ij} = Q_j - R_{ij(old)} \quad (1)$
        $R_{ij} = Sfactor_{MS} \times \prod_{j' \in V(i) \backslash j} \mathrm{sign}(Q_{ij'}) \times \min_{j' \in V(i) \backslash j} (|Q_{ij'}|) \quad (2)$
        $Q_j = Q_{ij} + R_{ij} \quad (3)$
    end for
end for
```

Where $R_{ij(old)}$ represents the stored value of R_{ij} from previous iteration and $Sfactor_{MS}$ is the correction factor for the normalized Min-Sum. By storing $R_{ij(old)}$ values in check nodes and passing Q_j values through them in each cycle, all the processing for this step can be done in check nodes.

3) *Syndrome check and Termination of Decoding:* Based on Q_j values in every step, the estimated bits for the output are generated by variable nodes based on the following:

$$\hat{x_i} = \begin{cases} 1, & \text{if } Q_i \leq 0 \\ 0, & \text{if } Q_i > 0 \end{cases} \tag{4}$$

If the estimated bits satisfy all the parity check equations, or the number of iterations exceeds a predefined maximum value (I_{max}), then the decoding is terminated.

III. Permutational Parity Check Matrices

As mentioned earlier, in layered scheduling the H matrix is divided into a number of layers, Y. Each of these layers contains a certain number of check nodes, M_l (i.e. $M_l = M/Y$). Furthermore, assume any particular partitioning over the columns of the matrix, dividing them into U groups of N_c columns. Each of these column partitions are called a *column group* here. Additionally, assume $Submatrix(l, c)$ to be the submatrix in layer l and column group c. Figure 1 (a) shows an example parity check matrix with 4 layers and 4 column groups. For this matrix, $Submatrix(1, 1)$ is shown by A.

A. Valid Mapping

A mapping from the column groups of layer L_1 to column groups of layer L_2 is called *valid* if it has the following two properties:

1) It is one-to-one,
2) It maps every non-zero submatrix in layer L_1 to an equal or an all-zero submatrix in layer L_2,

Figure 1 (b) shows a mapping that is *valid* from layer 1 to layer 2 of the matrix in part (a).

B. Permutational Matrix

For a parity check matrix, if a permutation of layers shown by P = $(L_0, L_1, ..., L_{Y-1})$ and a mapping MP exists such that:

1) $\forall k \in \{0, ..., Y-2\}$: MP is *valid* from layer L_k to L_{k+1},
2) MP is *valid* from layer L_{Y-1} to layer L_0.

then the matrix is called *Permutational* with valid mapping MP. Figure 1 (c) shows that the matrix in part (a) is permutational with valid mapping of part (b).

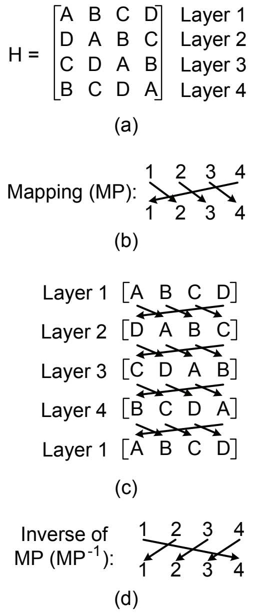


Fig. 1. A simple matrix with permutational structure: (a) The matrix defined by its layers and column groups; (b) a valid mapping (MP is valid from layer 1 to layer 2); (c) a permutation of layers mapped by MP implying matrix in (a) is permutational; (d) The inverse of MP which is used in designing the proposed decoder for matrix in part (a).

The definition of permutational parity check matrices covers a wide range of codes, including (672,336), (672,504) and (672,588) LDPC codes adopted in IEEE 802.15.3c (WPAN) standard and (672,336), (672,504) and (672,588) codes proposed in IEEE 802.11ad (WiFi). In the next section, it is shown that a new partial-parallel decoding scheme (called Permutational LDPC decoding here) can be developed for these matrices that significantly simplifies the routing network of the decoder, by eliminating almost all the gates on the network.

IV. Permutational LDPC Decoding

In the proposed method, the V-to-C and C-to-V routing networks are hard-wired based on one of the layers in the parity-check matrix. Then the V-to-C routing is coupled with another fixed wiring network, performing a constant shift over messages from variable nodes. Although the physical routing network has no change over different sub-iterations, the shift in the variable node messages produces different decoding process, matching to different layers. The method is explained in details in this section.

For a permutational parity-check matrix with $Y \times M_l$ rows and $U \times N_c$ columns, the architecture of the general permutational LDPC decoder is shown in Figure 2. The number of implemented check nodes are the number of rows in a layer (i.e., M_l), and the number of implemented variable nodes are the overall number of columns in the matrix (i.e., $U \times N_c$). The original routing network between check nodes and variable nodes is hard-wired based on the layer with highest row degree, L_{max}, and the coupled shifting network is implemented based on the inverse of valid mapping defined on matrix ($Mapping^{-1}$). In the first iteration, the variable node output signals are initialized by the log likelihood ratio of channel information.

In order to explain how the decoding process works, the architecture for the permutational parity-check matrix in Fig. 1 is shown in Fig. 3. For this matrix, since the row weights are the same for all

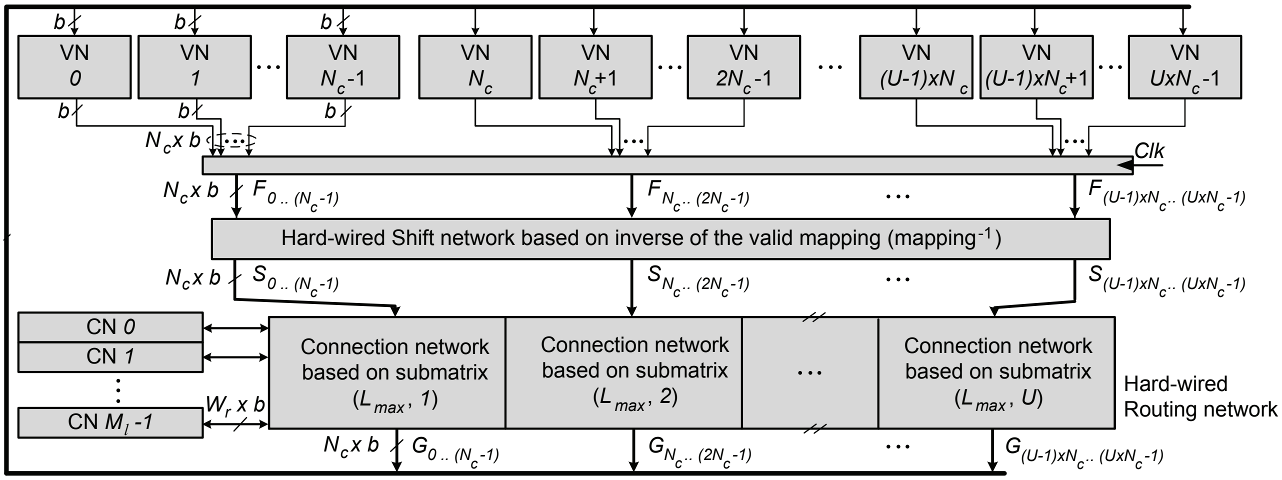

b
VN 0
VN 1
VN N_c-1
VN N_c
VN N_c+1
VN $2N_c$-1
VN $(U-1)xN_c$
VN $(U-1)xN_c$+1
VN UxN_c-1
$N_c x b$
Clk
$F_{0..(N_c-1)}$
$F_{N_c..(2N_c-1)}$
$F_{(U-1)xN_c..(UxN_c-1)}$
Hard-wired Shift network based on inverse of the valid mapping (mapping^{-1})
$S_{0..(N_c-1)}$
$S_{N_c..(2N_c-1)}$
$S_{(U-1)xN_c..(UxN_c-1)}$
CN 0
CN 1
CN M_l -1
$W_r x b$
Connection network based on submatrix (L_{max}, 1)
Connection network based on submatrix (L_{max}, 2)
Connection network based on submatrix (L_{max}, U)
Hard-wired Routing network
$G_{0..(N_c-1)}$
$G_{N_c..(2N_c-1)}$
$G_{(U-1)xN_c..(UxN_c-1)}$

Fig. 2. The schematics for general permutational LDPC Decoder

layers, there is no preferences in choosing the implemented layer in hardware, and layer 4 is chosen arbitrary. The shifting network is based on the inverse of mapping in Fig. 1 (b), which is shown in Fig. 1 (d) as well.

In Fig. 3 F_j is the registered output from variable node j, and $F_{0..(N_c-1)}$ represent the vector of F_j values from first N_c variable nodes. Signals after the shifting network are denoted by S_j, and G_j signals contain results from check nodes after check node processing is finished (Q_j in equation 3). Table I shows how the Q_j values corresponding to different column groups in the matrix in Fig. 1 part (a) change over time and in location in four sub-iterations of the first decoding iteration, and how they are updated by check nodes. For instance, following Q_j values for first N_c columns (first column group) can be summarized in these steps:

1) *Sub-iteration 0:* In initialization step, $Q_{0..(N_c-1)}$ vector is initialized by log-likelihood ratio from channel information, so $F_{0..(N_c-1)} = \lambda_{0..(N_c-1)}$,
2) *Sub-iteration 1:*
    a) shifting network results in movement of data corresponding to column groups, consequently $S_{3N_c..(4N_c-1)} = F_{0..(N_c-1)} = \lambda_{0..(N_c-1)}$,
    b) The S_j signals are transmitted to check nodes through connection matrix based on layer 4. So $S_{3N_c..(4N_c-1)}$ ($= \lambda_{0..(N_c-1)}$) are fed to check nodes through submatrix A, which is the proper connection matrix for $\lambda_{0..(N_c-1)}$ in processing of layer 1. Other λ_j values are connected to check nodes based on layer 1 as well. As the result, layer 1 is processed properly in first sub-iteration, and the updated Q_j's for first N_c columns are produced at $G_{3N_c..(4N_c-1)}$. These signals are transmitted to last N_c variable node units in the architecture. Let ${Q_j}^{(1)}$ represent Q_j after layer 1 is processed,
3) *Sub-iteration 2:*
    a) After direct registering of variable node outputs, ${Q_j}^{(1)}$ values corresponding to first N_c columns are registered in $F_{3N_c..(4N_c-1)}$. The fixed shift based on inverse mapping is done again, so $S_{2N_c..(3N_c-1)} = F_{3N_c..(4N_c-1)} = {Q_{0..(N_c-1)}}^{(1)}$.
    b) $S_{2N_c..(3N_c-1)}$ are transmitted to check nodes based on submatrix B, but in this sub-iteration these signals contain values corresponding to first column group. Passing ${Q_{0..(N_c-1)}}^{(1)}$ to check nodes through B is the proper configuration in layer 2. Examining other column groups shows that layer 2 is processed properly in current sub-iteration. ${Q_{0..(N_c-1)}}^{(2)}$ are generated in $G_{2N_c..(3N_c-1)}$, and are sent to variable nodes $2N_c$ to $3N_c - 1$,
4) *Sub-iterations 3 & 4:* Layers 3 and 4 get processed properly in following two sub-iterations. At the end of fourth sub-iteration, all four layers are processed and one decoding iteration is complete. Furthermore, at the end of fourth sub-iteration, the first N_c variable nodes in the architecture represent the first N_c columns of the matrix. Therefore the generated bits from variable nodes are in the correct order. By registering the outputs at the end of this sub-iteration, no shifting is needed for output bits.

In general, the decoding process in the proposed scheme starts for each iteration with layer L_1, for which M is valid from L_{max} to L_1. Afterward, layer L_2 gets processed for which M is valid from L_1 to L_2. The last layer that gets processed in a decoding iteration is L_{max}, for which the output bits are registered. The only disadvantage of the proposed architecture comparing to the original layered-decoding partial-parallel architecture is that the syndrome check can not be done after processing each layer. Without adding extra logic to shift back the output bits to their proper place, syndrome check can only be done at the end of one complete iteration.

Although the shifting of Q_j values is done in hardware, for R_ij values shifting is not necessary, since registering is done inside every check node separately. Overall, by using a constant routing network, the movement between layers is managed without using any gates. The only gates that should be used in the routing network are for the column groups which have irregularity in the matrix structure. Also It should be mentioned that the complexity of the routing network is not changed dramatically. The shifting network and the connection network based on matrix are in series, so they can be assumed as one overall shifting network, comparable to any other v-to-c routing network in conventional partial-parallel decoders, but with no gates.

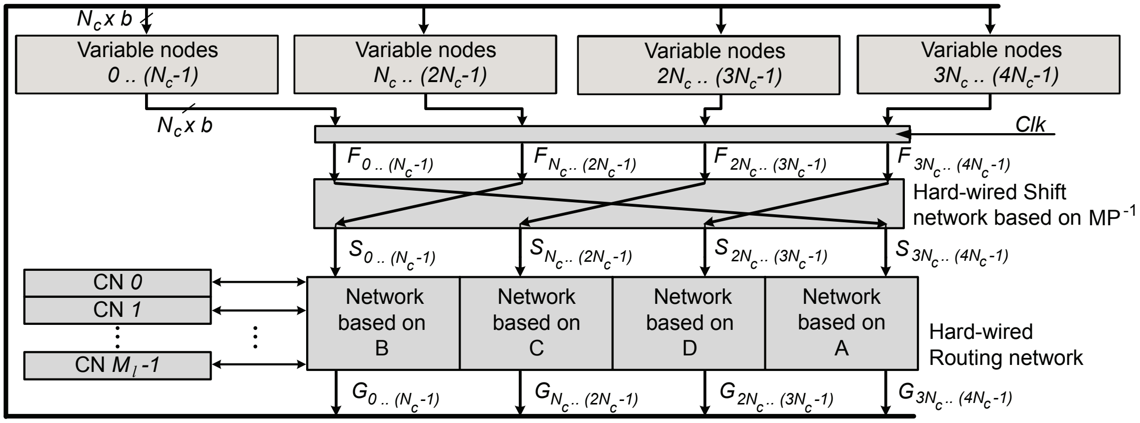
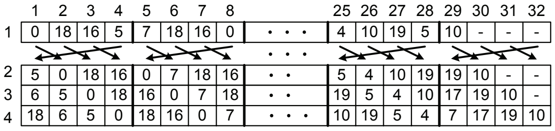


Fig. 3. The schematics of a permutational LDPC Decoder for the matrix defined in Fig 1 (a)

Sub-iteration number	Phase of Process	Q_j of first N_c columns	Q_j of second N_c columns	Q_j of third N_c columns	Q_j of fourth N_c columns	Sub-iteration Result
0	Initial Value	$\lambda_{0..(N_c-1)}$	$\lambda_{N_c..(2N_c-1)}$	$\lambda_{2N_c..(3N_c-1)}$	$\lambda_{3N_c..(4N_c-1)}$	-
1	Location in F	$0..(N_c-1)$	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	Layer 1 is processed
	Location in S	$3N_c..(4N_c-1)$	$0..(N_c-1)$	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	
	Eff. Conn. Mat.	A	B	C	D	
	Location in G	$3N_c..(4N_c-1)$	$0..(N_c-1)$	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	
2	Location in F	$3N_c..(4N_c-1)$	$0..(N_c-1)$	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	Layer 2 is Processed
	Location in S	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	$0..(N_c-1)$	$N_c..(2N_c-1)$	
	Eff. Conn. Mat.	D	A	B	C	
	Location in G	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	$0..(N_c-1)$	$N_c..(2N_c-1)$	
3	Location in F	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	$0..(N_c-1)$	$N_c..(2N_c-1)$	Layer 3 is Processed
	Location in S	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	$0..(N_c-1)$	
	Eff. Conn. Mat.	C	D	A	B	
	Location in G	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	$0..(N_c-1)$	
4	Location in F	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	$0..(N_c-1)$	Layer 4 is processed
	Location in S	$0..(N_c-1)$	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	
	Eff. Conn. Mat.	B	C	D	A	
	Location in G	$0..(N_c-1)$	$N_c..(2N_c-1)$	$2N_c..(3N_c-1)$	$3N_c..(4N_c-1)$	

TABLE I
THE TABLE FOLLOWS Q_j VALUES CORRESPONDING TO COLUMNS OF THE MATRIX IN FIG. 1 AS THEY CHANGE LOCATION IN DECODER OF FIG. 3 IN DIFFERENT SUB-ITERATIONS, SO THAT IN EVERY SUB-ITERATION THESE VALUES ARE FED TO CHECK NODES THROUGH PROPER CONNECTION MATRIX, AND ALL FOUR LAYERS OF THE MATRIX ARE PROCESSED IN ONE DECODING ITERATION



	1	2	3	4	5	6	7	8		25	26	27	28	29	30	31	32
1	0	18	16	5	7	18	16	0	• • •	4	10	19	5	10	-	-	-
2	5	0	18	16	0	7	18	16	• •	5	4	10	19	19	10	-	-
3	6	5	0	18	16	0	7	18	• •	19	5	4	10	17	19	10	-
4	18	6	5	0	18	16	0	7	• • •	10	19	5	4	7	17	19	10

Fig. 4. A valid mapping defined for the parity-check matrix of the IEEE 802.15.3c (672,588) LDPC code; The mapping and the permutation (1,2,3,4) demonstrate that the matrix is *Permutational*.

V. CMOS IMPLEMENTATION FOR (672,588) LDPC CODE

In this section, the permutational decoder for the (672,588) LDPC code proposed in IEEE 802.15.3c standard is implemented in 65 nm CMOS technology, and the results are compared to a conventional partial-parallel decoder for the same code. The parity-check matrix of the code has M= 84 rows, N= 672 columns, and can be divided into Y= 4 layers and U= 32 column groups. Figure 4 shows the matrix and a mapping that is valid over the sequence of layers 1,2,3,4. From the definition, the parity-check matrix is permutational.

Based on the general architecture of the permutational decoder in Figure 2, 21 check nodes and 672 variable nodes are implemented in hardware for this code. The V-to-C and C-to-V routing networks are based on the connection matrix of layer 4, which has the largest row degree. Before the V-to-C routing network, a constant shifting network based on the inverse of valid mapping in Figure 4 is utilized. The variable node messages are initialized by the channel information, and after every 4 cycles, one iteration of decoding is done.

The layout for the proposed decoder is shown in Fig. 5. Table II compares the implementation results with a conventional partial-parallel decoder with the same number of implemented check nodes and variable nodes. It should be noted that the only difference between the proposed decoder and conventional partial-parallel decoders is in the routing network. Therefore, there is no degradation in the BER performance of the decoder.

The savings in the proposed method are the result of reducing

	ASSCC'10 [14]	ISCAS'11 [18]	CICC'07 [8]	Regular partial-Parallel Architecture	Proposed Architecture
CMOS fabrication process	65 nm	65 nm	0.13 μm	65 nm	65 nm
Code Length	672	672	660	672	672
Supported Code rates	1/2, 5/8, 3/4, 7/8	1/2, 5/8, 3/4, 13/16	0.73	7/8	7/8
Input Quantization (bits)	6	5	4	6	6
Gate count(k)	647	-	690	138	125
Core area (mm^2)	1.562	1.3	7.3	**0.891**	**0.718**
Maximum clock frequency (MHz)	197	150	300	180.2	**235**
Maximum Iteration Count (I_{max})	5	15	15	5	5
Throughput @ I_{max} (Gbps)	5.79	3.08	2.44	**6.05**	**7.9**

TABLE II
COMPARISON OF RESULTS OF PERMUTATIONAL LDPC DECODER AND OTHER DECODERS

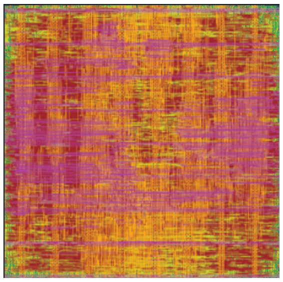


Fig. 5. layout implementation of proposed decoder for (672,588) code

the number of gates used for adjusting interconnection network to different layers. In the partial-parallel architecture, one 4:1 mux is used for every pin of every check node, resulting in overall $2 \times 21 \times 32 (= 1344)$ 4:1 muxes in the routing network. On the other hand, in the proposed permutational architecture, only a few 2:1 muxes are used for the columns corresponding to irregularities in the matrix structure (last 3 columns in this code). The overall number of gates on the routing network is $2 \times 3 \times 21 (= 126)$ 2:1 muxes. This means over 96% decrease in gate count on the routing network, which results in 1.26 times improvement for area. Additionally, since the 4:1 muxes in partial-parallel architecture are on the critical path of signals passing through check nodes, reducing them to at most 2:1 muxes yields further improvements in throughput. The proposed decoder is 1.3 times faster.

VI. CONCLUSION

A new routing network architecture based on characteristics of parity-check matrix of LDPC codes proposed in the IEEE 802.15.3c and 802.11ad is presented which results in almost complete elimination of gates on the routing network of the decoder, and provides improvements in area, throughput and power, with no decline in BER performance. The class of matrices for which the architecture could be used is defined, and the general architecture is presented. Implementing the decoder in 65 nm CMOS technology including place & route for (672,588) LDPC code in the IEEE 802.15.3c shows over 96% reduction in number of gates on the routing network, which results in 1.26× improvement in area and 1.3× improvement in throughput. The definition of permutational parity-check matrices in this work can also be used as a guideline in generating LDPC codes for emerging standards resulting in more efficient hardware implementations.

REFERENCES

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

[2] "IEEE P802.3an, 10GBASE-T task force," http://www.ieee802.org/3/an.

[3] "T.T.S.I. digital video broadcasting (DVB) second generation framing structure for broadband satellite applications.," http://www.dvb.org.

[4] "802.11ad full proposal submission," http://mentor.ieee.org/802.11/dcn.

[5] "IEEE standard for information technology - specific requirements. part 15.3," *IEEE Std 802.15.3c-2009 (Amendment to IEEE Std 802.15.3-2003)*, pp. c1 –187, 12 2009.

[6] T. Mohsenin and B. Baas, "Trends and challenges in LDPC hardware decoders," in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, Nov. 2009, pp. 1273–1277.

[7] A. Blanksby and C. J. Howland, "A 690-mW 1-Gb/s 1024-b, rate 1/2 low-density parity-check code decoder," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 3, pp. 404–412, Mar. 2002.

[8] A. Darabiha, A.C. Carusone, and F.R. Kschischang, "A 3.3-Gbps bit-serial block-interlaced Min-Sum LDPC decoder in 0.13-um CMOS," in *IEEE Custom Integrated Circuits Conference*, Sep. 2007, pp. 459–462.

[9] T. Mohsenin, D.N. Truong, and B.M. Baas, "A low-complexity message-passing algorithm for reduced routing congestion in LDPC decoders," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 5, pp. 1048–1061, May. 2010.

[10] M. Karkooti, P. Radosavljevic, and J.R. Cavallaro, "Configurable, high throughput, irregular LDPC decoder architecture: Tradeoff analysis and implementation," in *ASAP*, sept. 2006, pp. 360 –367.

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

[12] T. Brack, F. Kienle, and N. Wehn, "Disclosing the LDPC code decoder design space," in *DATE*, march 2006, vol. 1, p. 6 pp.

[13] G. Malema and M. Liebelt, "Interconnection network for structured low-density parity-check decoders," in *APCC*, oct. 2005, pp. 537 –540.

[14] Shiang-Yu Hung et al., "A 5.7gbps row-based layered scheduling LDPC decoder for IEEE 802.15.3c applications," in *A-SSCC*, nov. 2010, pp. 1 –4.

[15] M. Mansour and N.R. Shanbhag, "High-throughput LDPC decoders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 6, pp. 976–996, Dec. 2003.

[16] P. Radosavljevic, A. de Baynast, and J.R. Cavallaro, "Optimized message passing schedules for LDPC decoding," in *ACSSC*, 28 2005-nov. 1 2005, pp. 591 –595.

[17] J. Chen, A. Dholakia, E. Eleftheriou, and M. Fossorier, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, no. 7, pp. 1288–1299, Aug. 2005.

[18] M. Weiner, B. Nikolic, and Z. Zhang, "LDPC decoder architecture for high-data rate personal-area networks," in *ISCAS*, may 2011, pp. 1784 –1787.