Journal of Circuits, Systems, and Computers
Vol. 22, No. 6 (2013) 1350039 (12 pages)

DOI: 10.1142/S0218126613500394

World Scientific
www.worldscientific.com

DESIGN OF AN AREA-EFFICIENT HIGH-THROUGHPUT SHIFT-BASED LDPC DECODER*

YUN-CHING TANG[†,‡,§], HONG-REN WANG[†],
HONGCHIN LIN[†,¶] and JUN-ZHE HUANG[†]

[†]*Department of Electrical Engineering,*
National Chung Hsing University,
Taichung 402, Taiwan

[‡]*Department of Electronic Engineering,*
Hsiuping University of Science Technology,
Taichung 412, Taiwan
[§]*tangyc@hust.edu.tw*
[¶]*hclin@nchu.edu.tw*

Received 4 February 2012
Accepted 31 July 2012
Published

An area-efficient high-throughput shift-based LDPC decoder architecture is proposed. The specially designed (512, 1,024) parity-check matrix is effective for partial parallel decoding by the min-sum algorithm (MSA). To increase throughput during decoding, two data frames are fed into the decoder to minimize idle time of the check node unit (CNU) and the variable node unit (VNU). Thus, the throughput is increased to almost two-fold. Unlike the conventional architecture, the message storage unit contains shift registers instead of de-multiplexers and registers. Therefore, hardware costs are reduced. Routing congestion and critical path delay are also reduced, which increases energy efficiency. An implementation of the proposed decoder using TSMC 0.18 μm CMOS process achieves a decoding throughput of 1.725 Gbps, at a clock frequency of 56 MHz, a supply voltage of 1.8 V, and a core area of 5.18 mm^2. The normalized area is smaller and the throughput per normalized power consumption is higher than those reported using the conventional architectures.

Keywords: Low-density parity-check codes; VLSI decoder architectures; shift-based LDPC decoder.

1. Introduction

With the advances in information transmission, the error correction code has been used to correct the transmission errors and reduce required transmission energy. The low-density parity-check (LDPC) code first introduced by Gallager in 1962[1] is a

*This paper was recommended by Regional Editor Eby G. Friedman.
§Corresponding author.

binary linear block code, which is capable of approaching the Shannon limit. Since the last decade, advances in VLSI technology have generated interest in the use of LDPC[1–3] codes. Several specifications have adopted the LDPC codes as an error correction code to enhance transmission quality, such as DVB-S2, 10 GBASE-T, WiFi (802.11 n), WiMax (802.16e) and the fourth-generation mobile communications (4 G).

The belief propagation algorithm (BPA)[4] with slight simplification of the theoretical algorithm provides accurate decoding capability. Blanksby *et al.*[5,6] implemented a 1-Gb/s fully parallel decoder using BPA, but the circuit occupied a large chip area. Complex circuits can be simplified by using the min-sum algorithm (MSA),[7] which provides acceptable accuracy.

In addition to decoding algorithms, circuit complexity can be further reduced in the structured LDPC by using a regular structure of parity-check matrix (H matrix), which is more appropriate for hardware implementation compared to a randomly generated H matrix. Commonly used structured LDPC codes include quasi-cyclic LDPC (QC-LDPC) codes,[8] array LDPC codes[9] and Reed Solomon-based LDPC codes (RS-LDPC).[10] The H matrices are composed of shifted sub-matrices that must be carefully selected, since the numbers and sizes of sub-matrices as well as the minimum number of cycles (i.e., girth) affect the decoding performance and circuit complexity.

These LDPC decoders can be implemented using fully parallel,[11] partial parallel,[12] or serial[13] architectures. In the fully parallel architecture, each check or variable node requires a processor. Therefore, its throughput is very high. However, it occupies huge chip area due to numerous processors and complex interconnections caused by a quite large number of irregular edges. On the contrary, the interconnect complexity can be reduced by employing the serial architecture, which is impractical because of small throughput, and thus limits the applications of serial LDPC decoders. The partial parallel architecture significantly reduces the node processing units rather than the fully parallel architecture. Therefore, this trade-off choice has been widely used in many studies.[14] For example, the early version[12] employs the many multiplexers (mux) and de-multiplexers (demux) with long latency. The other flexible architecture[15] results in greater circuit complexity and smaller throughput than the standard architectures do.

For practical applications of LDPC decoders, IC chips tend to have low cost and low power. In this paper, a specially designed H matrix of LDPC decoder was implemented using the partial parallel LDPC architecture with the special shift-register technique to achieve the best performance, including a figure of merit defined as throughput divided by normalized power.[16,17]

The organization of this paper is as follows. After the background is discussed, the design of the proposed LDPC code is described. Next, the architecture of the partial parallel LDPC decoder is proposed. Then, the experimental results and comparison are presented. The final section is the conclusion.

2. Background

The QC-LDPC codes are structured LDPC codes with good regularity, which is appropriate for hardware implementation. Figure 1 illustrates its H matrix ($H_{m\times n}$), which is composed of $p \times p$ shifted unitary matrices $S_{i,j}$, where i and j mean the row and column indices are integers from 1 to m/p and n/p, respectively. This QC-LDPC code with block length n has column and row weights of m/p and n/p, which represent the number of 1's in a column and a row, respectively. Each matrix $S_{i,j}$ in an H matrix is a $p \times p$ unitary matrix shifting to the right $s_{i,j}$ times, where $0 \leqq s_{i,j} < p - 1$.

Figure 2 shows that the conventional LDPC decoding architecture[12] can be divided into four major modules: variable node units (VNUs), check node units (CNUs), and two message storage units (Δ and Λ registers). The message storage units store the updated CNU and VNU data. The CNUs start to work only after the entire VNU computation is complete, and vice versa. In the QC-LDPC H matrix in Fig. 1, the shifted unitary matrices can be calculated column by column in the VNU operation and then compared row by row in the CNU operation. This partial parallel architecture significantly reduces the node processors to p rather than to n or m in the fully parallel architecture. That also results in reduced routing complexity. However, the additional mux and demux are required.[12] Registers with mux and demux can be replaced by memories like register files, which are marked by the dashed lines in Fig. 2. Besides, data "re-order" blocks for aligning updated data in the correct positions are needed before data are restored in the registers. In specific

$$H_{m\times n} = \begin{bmatrix} S_{1,1} & S_{1,2} & \cdots & S_{1,(n/p)-1} & S_{1,(n/p)} \\ S_{2,1} & & & & S_{2,(n/p)} \\ \vdots & & \ddots & & \vdots \\ S_{(m/p)-1,1} & & & & S_{(m/p)-1,(n/p)} \\ S_{(m/p),1} & S_{(m/p),2} & \cdots & S_{(m/p),(n/p)-1} & S_{(m/p),(n/p)} \end{bmatrix}$$

Fig. 1. H matrix of QC-LDPC code.

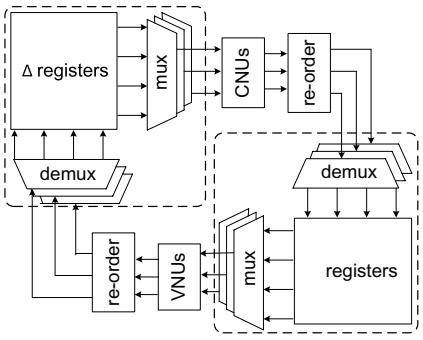

Fig. 2. Architecture of conventional partial parallel LDPC decoder.

permutations of the H matrix,[12] some overlapped scheduling processes are also used to reduce latency, but the idling time of CNUs or VNUs still approximates to 50%.

To design a good H matrix, girth which is the minimum cycle, or cycle length, significantly determines the decoding performance. The conventional QC-LDPC and RS-LDPC codes were further enhanced to the partition-and-shift LDPC (PS-LDPC) code[18] which theoretically optimizes the girth. The algorithm counts $2t$ translations between the shifted unitary matrices. If and only if there exists a closed loop of $2t$ cycles exits in the tanner graph, then Eq. (1) holds,[18] where "mod" means modulus operation and $s_{\alpha 1,\beta 1}$, $s_{\alpha 2,\beta 2}, \ldots, s_{\alpha 2t,\beta 2t}$, represent the shifted numbers in the shifted unitary matrix. The minimum cycle number, $2t$, is the girth.

$$\mathrm{mod}[(-1)^1 s_{\alpha_1,\beta_1} + (-1)^2 s_{\alpha_2,\beta_2} + \cdots + (-1)^{2t-1} s_{\alpha_{2t-1},\beta_{2t-1}} + (-1)^{2t} s_{\alpha_{2t},\beta_{2t}}, p] = 0. \quad (1)$$

Here, this PS-LSPC code algorithm is adopted to maximize the girth.

3. Design of LDPC Code

Figure 3 shows the proposed H matrix using PS-LDPC codes which can be partitioned into 4×4 sub-blocks. Each sub-block contains k shifted unitary matrices. The column and row weights of each sub-block are 1 and k, respectively.

In our design, all diagonal sub-blocks (H_{14}, H_{23}, H_{32} and H_{41}) are zero matrices. The advantage is to avoid the data access in memories at the same triggered edge. For example, if all sub-blocks are non-zero matrices, after the CNUs work for the last row (H_{41}, H_{42}, H_{43} and H_{44}), the updated data for H_{41} must be stored in the memory as shown in Fig. 2 and read out at the same time for the next VNU operation of the first column (H_{11}, H_{21}, H_{31} and H_{41}). The register-file memories cannot be applied in this situation. The solution is to use flip-flop based registers. However, the mux and demux before and after the registers increase the critical paths and routing complexity. With zero diagonal sub-blocks, the critical paths can be shortened and routing complexity can be reduced.

To further reduce the number of mux and power consumption, the sub-blocks H_{11}, H_{22}, H_{33} and H_{44} are composed of k unitary matrices without shift. Figure 4 illustrates the proposed parity-check matrix, in which "I" indicates a $p \times p$ unitary matrix, and each sub-block has k shifted or non-shifted unitary matrices. The parameter k gives the coding rate of $(k-1)/k$. The coding length is $4 \times k \times p$, where p can be used to adjust the coding length. Therefore, it is a very flexible approach to provide

$\mathbf{H_{11}}$	$\mathbf{H_{12}}$	$\mathbf{H_{13}}$	$\mathbf{H_{14}}$
$\mathbf{H_{21}}$	$\mathbf{H_{22}}$	$\mathbf{H_{23}}$	$\mathbf{H_{24}}$
$\mathbf{H_{31}}$	$\mathbf{H_{32}}$	$\mathbf{H_{33}}$	$\mathbf{H_{34}}$
$\mathbf{H_{41}}$	$\mathbf{H_{42}}$	$\mathbf{H_{43}}$	$\mathbf{H_{44}}$

Fig. 3. Partition of the H matrix.

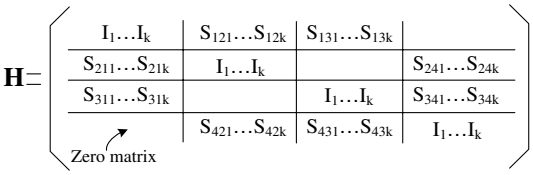

Fig. 4. The proposed general form of H matrix.

0	0	55	104	46	52		
121	72	0	0			111	110
107	118			0	0	34	22
		27	97	85	106	0	0

Fig. 5. The (1, 024, 3, 6) parity check matrix used for implementation.

various coding rates and coding length. In other words, if we want to have higher coding rates, the k is increased, which corresponds to increased unitary matrices in a sub-block. This paper evaluated the implementation of an H matrix (1,024, 3, 6) with girth of 8 as shown in Fig. 5, where 1,024 is the code length, and 3 and 6 are the column and row weights, respectively. The unitary matrices are 128×128. The integers in Fig. 5 indicate the numbers of right shifts in the unitary matrices.

Figure 6 shows the data for decoding capability, which was tested by comparing bit-error rates (BER) of the proposed PS-LDPC code and the randomly generated

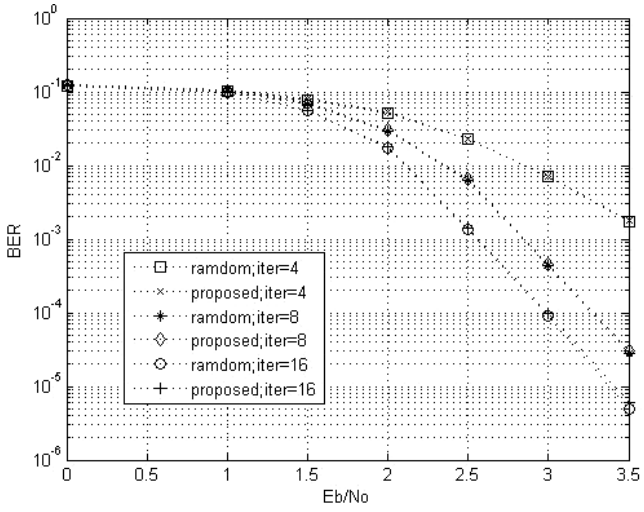

Fig. 6. Performance comparison of the randomly generated and proposed LDPC codes with different iterations for BPSK modulation.

LDPC code with the same (1,024, 3, 6) matrix properties by using BPSK modulation with 4, 8 and 16 iterations. The performance of the proposed LDPC code with girth = 8 is virtually identical to that of the randomly generated LDPC code with girth = 6. Since the curve of eight iterations is close to that of 16 iterations, eight iterations were adopted in the following implementation.

4. Shift-Based LDPC Decoding Architecture

Figure 7 shows the partial parallel decoding architecture using the min-sum algorithm with the specially designed LDPC code given in Fig. 5. The variable node calculation contains 256 columns in the H matrix, so 256 VNUs work in parallel. Similarly, the check node calculation runs 128 rows, thus 128 CNUs work in parallel. Both the variable node and the check node operations require four clock cycles, respectively, in a decoding iteration.

For the optimal use of VNUs and CNUs with the minimal idle time, two frames of the received log likelihood ratios (LLR) are stored in the input buffers. The first frame is transferred to the VNUs with the updated data from the check-to-variable storage unit (CTVSU), if available, during the first four clocks. Then, the second frame of data is fed to the VNUs in the next four clocks. The process is named as "collection." The VNU outputs are stored in the variable-to-check storage unit (VTCSU), which is composed of shift registers. In the following "check" procedure, VTCSU data are sent to the CNUs for updating. The iteration is complete when the data are stored in the CTVSU. Note that the two data frames are computed using VNUs and CNUs alternatively to maximize throughput.

The architectures of the VTCSU and CTVSU message storage units differ from those of the conventional approaches given in Fig. 2. Either register-file memories or registers with mux and demux occupy large chip areas and consume more power compared to the proposed shift registers-based technique. Figure 8 shows the delay clock cycles for the VTCSU. That shows how the outputs of VNUs are scheduled to the inputs of CNUs. In Fig. 8(a), the sub-blocks marked by the bold lines give the

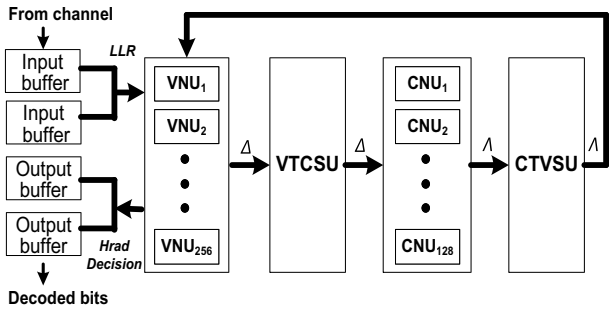

Fig. 7. Architecture of LDPC decoder.

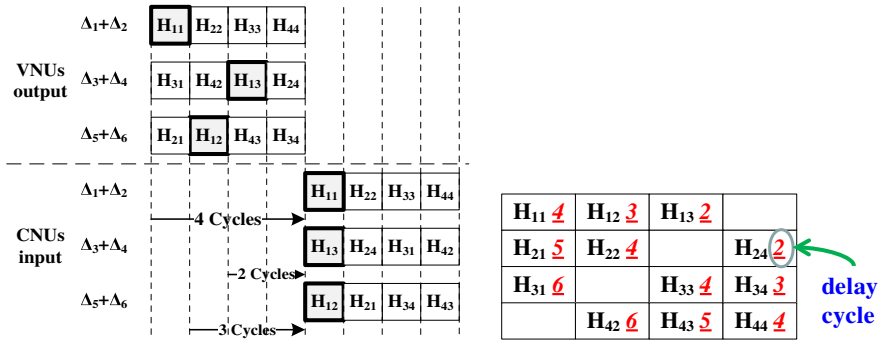

Fig. 8. (Color online) (a) Examples of delay clock cycles in the VTCSU. (b) Delay clock cycles for each sub-block.

examples of the delay cycles of three sub-blocks in the VTCSU. Sub-blocks H_{11}, H_{22}, H_{33} and H_{44} sequentially enter the CNUs after four clock cycles of the outputs of VNUs. However, the others require different delay clock cycles to enter VNUs as indicated by the underlined numbers in Fig. 8(b), which shows the sub-block H_{ij}, where i or j is from 1 to 4. The H_{14}, H_{23}, H_{32} and H_{41} are zero matrices.

Figure 9 shows an implementation of the VTCSU using 12 blocks of shift registers categorized as A1 to A4, B1 to B6 and C1 to C2. Figure 10 plots the corresponding timing diagram. The VNU outputs $(\Delta_1 + \Delta_2, \Delta_3 + \Delta_4, \Delta_5 + \Delta_6)$ enter the register blocks A4, B6, and C2; the outputs of A1 $(\Delta_1 + \Delta_2)$, B1 $(\Delta_3 + \Delta_4)$, and C1 $(\Delta_5 + \Delta_6)$ enter the CNUs. The shift registers A4 to A1 sequentially shift H_{11}, H_{22}, H_{33} and H_{44} sequentially without mux due to zero-shift unitary matrices. The shift

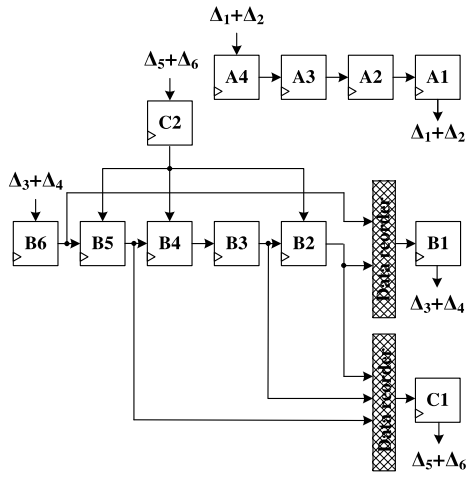

Fig. 9. Shift register-based architecture of VTCSU.

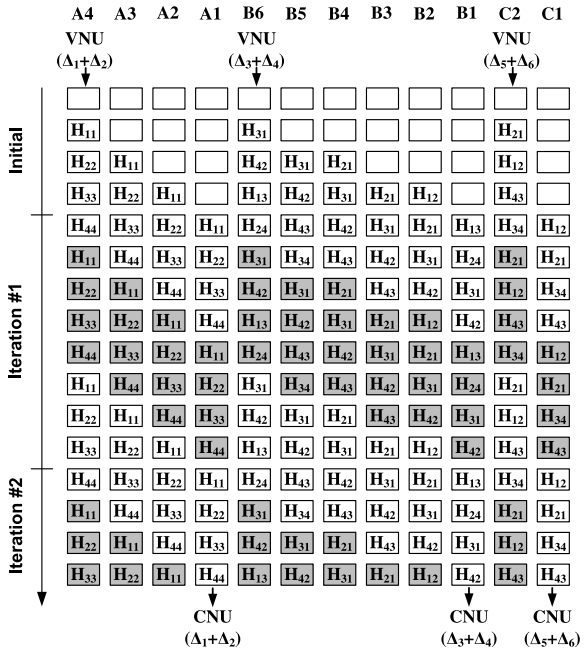

Fig. 10. Timing diagram of VTCSU.

registers B6 to B1 store H_{13}, H_{24}, H_{31} and H_{42}. The number of delay paths of $\Delta_3 + \Delta_4$ could be two or six, i.e., B6 → B1, or B6 → B5 → B4 → B3 → B2 → B1, respectively. The registers B5 to B2 are also shared with H_{12}, H_{21}, H_{34} and H_{43}, which are stored in the dedicated registers C2 and C1 as well. The delay paths of $\Delta_5 + \Delta_6$ delay either three clock cycles (i.e., C2 → B5 → C1 or C2 → B2 → C1) or five clock cycles (i.e., C2 → B4 → B3 → B2 → C1 or C2 → B5 → B4 → B3 → C1).

The proposed VTCSU has several advantages. It not only eliminates the demux, but also reduces the number of mux. The mux to re-order the data are inserted before the registers B1 and C1 due to the zero matrices of H_{14} and H_{41}. Therefore, the minimum number of delay cycles is two and the critical paths are reduced. The routing can be distributed between different registers to reduce routing complexity.

Table 1 compares the numbers of mux and demux used in the conventional register based architecture and the proposed shifter based design for the same LDPC code. The difference is the blocks marked by the dash lines in Fig. 2 and the VTCSU and CTVSU in Fig. 7. The area of a 2:1 mux can be estimated to be one-third of a

Table 1. Comparison of the mux and demux.

	4:1 mux	2:1 mux	1:4 de-mux
Conventional	6×1024	0	6×1024
Proposed	4×1024	6×1024	0

4:1 mux. The significant difference is no demux required in our design, so the gate count and the routing complexity are reduced.

5. Experimental Results and Comparison

The proposed shift-based (1,024, 3, 6) LDPC decoder using eight iterations with eight clocks per iteration was designed and implemented using the 0.18 μm CMOS process. Table 2 compares the simulated performance between the conventional and the proposed architectures. The gate count is reduced by 15%, so the reduced critical paths and routing complexity result in the maximum operating frequency increased by 25%. Note that the contribution of area reduction is mainly attributed to the VTCSU and CTVSU in Fig. 7. Those in the conventional LDPC decoder occupy approximately 47% of the chip area. If the area of VTCSU and CTVSU is reduced by 32% the net area improvement is about $47\% \times 32\% = 15\%$.

Figure 11 shows that microphotograph of the chip occupies an area of 10.8 mm^2 whereas the core occupies an area of 5.18 mm^2. The chip works at a clock frequency

Table 2. Comparison of simulated results.

	Conventional	Proposed
Cell gate count	436 K	373 K
Supply voltage (V)	1.8 V	1.8 V
Frequency (MHz)	80	100
Throughput (Mbps)	2275	2844
Throughput/power (Mbps/mW)	4.25	4.98

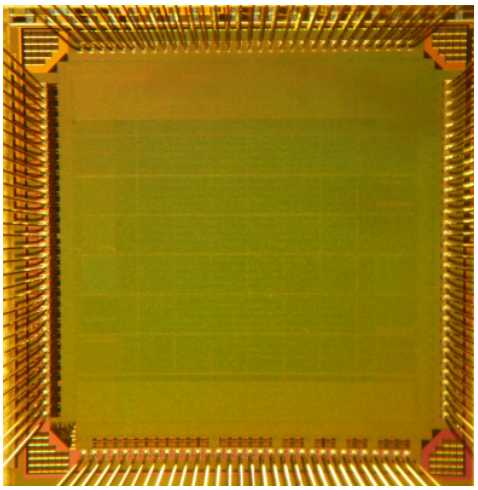

Fig. 11. (Color online) Die microphotograph of the proposed decoder.

Table 3. Comparison of LDPC decoders with measured data.

	JSSC'02[11]	TCASI'06[12]	JSSC'08[15]	TVLSI'10[16]	TVLSI'10[18]	This work
CMOS technology	0.16 μm	0.18 μm	0.13 μm	90 nm	0.18 μm	0.18 μm
Parallelism	Fully	Partial	Fully	Fully	Partial	Partial
Spec.	(1,024, 512)	(1,024, 512)	(660, 480)	(1,024, 512)	—	(1,024, 512)
Code rate	0.5	0.5	0.73	0.5	2/5,3/5,4/5	0.5
Iterations	64	8	15	16	15	8
Supply voltage (V)	1.5	1.62	1.2	1.2	1.8	1.8
Frequency (MHz)	64	200	300	400	125	56
Throughput (Mbps)	1,000	985	3,300	13,210	104.5	1,725
Chip area (mm^2)	52.5	10.08	7.3	4.97	9.76	5.18
Normalized area ($10^{-3}\,mm^2$)	3.91	0.59	1.36	1.17	—	0.305
Power (mW)	690	—	1383	577	486	51.1
Normalized power (mW)	306.67	—	960.42	400	150	15.8
Normalized energy (pJ/bit/iter)	4.79	—	19.40	1.89	95.69	1.14
Throughput/normalized power (Mbps/mW)	3.26	—	3.44	33	0.7	109

of 56 MHz at a supply voltage of 1.8 V. The throughput is 1.725 Gbps because two frames are decoded simultaneously.

Table 3 summarizes the results of performance comparisons with measurement results reported in the literature. Due to various processes and supplied voltages in the literature, some papers[16,17] define the following three normalized parameters to evaluate circuit performance.

$$\text{Nomalized Area} = \frac{\text{Chip Area}}{(\text{Codeword})^2(1-\text{Code Rate}) \times (\text{Technology})^2}, \tag{2}$$

$$\text{Nomalized Power} = \frac{\text{Total Power}}{(\text{Core Power Supply})^2}, \tag{3}$$

$$\text{Nomalized Energy} = \frac{\text{Normalized Power}}{(\text{Throughput} \times \text{Iterations})}. \tag{4}$$

Owing to technology scaling, the chip area is inversely proportional to the square of channel length. Thus, the normalized area is usually defined as chip area divided by $(\text{Technology})^2$. Besides, if the H matrix is large, the chip area is also large. Therefore, the normalized area is further divided by the sizes of H matrix, which is proportional to $(\text{codeword}) \times [(\text{codeword}) \times (1 - \text{code rate})]$, as shown in Eq. (2).

Power consumption is proportional to the square of supply voltage, so the normalized power is given in Eq. (3). To evaluate the power efficiency, it is reasonable to compare the energy required for each bit per iteration. Therefore, the normalized energy is defined as the normalized power divided by the product of throughput and iteration numbers as shown in Eq. (4).

In this work, all three normalized factors are very small. Specifically, for high throughput efficiency, we propose one figure of merit: throughput divided by normalized power. This parameter is the highest in the proposed decoder than in the other decoders. Based on the above analyzes, the proposed architecture is area-efficient and power-efficient. It can decode more data in terms of unit area and unit energy.

6. Conclusion

A special PSLDPC parity-check matrix equivalent to the randomly generated matrix is proposed. The proposed matrix is suitable for chip implementation with small area and power consumption for high throughput applications. Unlike the conventional architecture, no demux are required, and mux as well as critical paths are also reduced. In an implementation with TSMC 0.18 μm CMOS process, the proposed architecture occupies 10.8 mm^2 with core area of 5.18 mm^2. The measured throughputs are 1.46 Gbps at 1.62 V and higher than 1.7 Gbps at 1.8 V. The small normalized area and the high throughput per normalized power indicate that the proposed LDPC decoder is area-efficient and power-efficient and should be considered for use in the future LDPC decoders.

References

1. S. Haykin, *Communication Systems* 4th edn. (John Wiley & Sons, Inc).
2. R. Gallager, Low-density parity-check codes, *IRE Tans. Inf. Theory* **7** (1962) 21−28.
3. T. Ohtsuki, LDPC codes in communications and broadcasting, *IEICE Trans. Commun.* **E90-B3** (2007).
4. D. J. C. MacKay, Good error-correcting codes based on very sparse matrices, *IEEE Trans. Inf. Theory* **45** (1999) 399−431.
5. X. Y. Hu, E. Eleftheriou, D. M. Arnold and A. Dholakia, Efficient implementations of the sum-product algorithm for decoding LDPC codes, *IEEE Global Telecommun. Conf.* **2** (2001) 1036.
6. K. Kasai, T. Shibuya and K. Sakaniwa, Detailedly represented irregular low-density parity-check codes, *IEICE Trans. Fundamentals* **E86-A** (2003) 2435−2444.
7. J. Chen and M. C. Fossorier, Decoding low-density parity-check codes with normalized APP-based algorithm, *Proc. IEEE GLOBECOM*, Vol. 2 (Nov. 2001), pp. 1026−1030.
8. M. P. C. Fossorier, Quasi-cyclic low-density parity-check codes from circulant permutation matrices, *IEEE Trans. Inf. Theory* **50** (2004) 1788−1793.
9. J. L. Fan, Array codes as low-density parity-check codes, *Proc. 2nd Int. Symp. Turbo Codes and Related Topics* (Sep. 2000), pp. 543−546.
10. I. Djurdjevic, J. Xu, K. Abdel-Ghaffar and S. Lin, A class of low-density parity-check codes constructed based on Reed-Solomon codes with two information symbols, *IEEE Comm. Lett.* **7** (2003) 317−319.
11. A. Blanksby and C. Howland, A 690-mw 1-Gb/s 1024-b, rate-1/2 low-density parity-check code decoder, *IEEE J. Solid-State Circuits* **37** (2002) 404−412.
12. S. H. Kang and I. C. Park, Loosely coupled memory-based decoding architecture for low density parity check codes, *IEEE Trans. Circuits Syst. I* **53** (2006) 1045−1056.

13. E. Yeo, P. Pakzad, B. Nikolić and V. Anantharam, VLSI architectures for iterative decoders in magnetic recording channels, *IEEE Trans. Magn.* **37** (2001) 748−755.
14. T. Zhang and K. K. Parhi, VLSI implementation-oriented $(3, k)$-regular low-density parity-check codes, *IEEE Workshop on Signal Processing Systems* (*SiPS*) (Sept. 2001), pp. 25−36.
15. B. Xiang, R. Shen, A. Pan, D. Bao and X. Zeng, An area-efficient and low-power multirate decoder for quasi-cyclic low-density parity-check codes, *IEEE Trans. VLSI Syst.* **18** (2010) 1447−1460.
16. A. Darabiha, C. Carusone and F. R. Kschischang, Power reduction techniques for LDPC decoders, *IEEE J. Solid-State Circuits* **43** (2008) 1835−1845.
17. N. Onizawa, T. Hanyu and V. C. Gaudet, Design of high-throughput fully parallel LDPC decoders based on wire partitioning, *IEEE Trans. VLSI Syst.* **18** (2010) 482−489.
18. J. L. Membe and J. M. F. Moura, Partition-and-shift LDPC codes, *IEEE Trans. Magnet.* **41** (2005).