Strategies for Reducing Decoding Cycles in Stochastic LDPC Decoders

Di Wu, Yun Chen, Qichen Zhang, Yeong-luh Ueng, *Senior Member, IEEE*, and Xiaoyang Zeng

***Abstract*—This brief presents three strategies, including initialization based on Look Up Table (LUT), postprocessing based on bit flipping and hard decision based on the posterior information, to reduce the number of decoding cycles (DCs) for stochastic low-density parity-check decoding. For the standard IEEE 802.3an code, simulation indicates a 73.6% reduction in the average number of DCs with a satisfactory bit error rate. Moreover, hardware implementation shows that the area required for the proposed decoder is significantly reduced.**

***Index Terms*—Bit-flipping algorithm, high-throughput decoder, low-density parity-check (LDPC) codes, stochastic decoding.**

I. Introduction

Low-density parity-check (LDPC) codes [1] are adopted in a wide range of communication standards, such as the IEEE 802.3an standard [2]. The increasing demand for higher throughput applications has promoted in-depth research into fully parallel decoder architectures. However, conventional fully parallel decoders that contain substantial parallelism encounter difficulties due to wire congestion [3], limiting the core utility and constituting the prevailing bottleneck for the next-generation communication systems.

Stochastic computation was first applied to LDPC decoding in [4]. Stochastic decoders transmit soft messages in the bit serial method [5], mitigating the wiring overhead. In subsequent research, various stochastic decoders have been proposed, such as the edge memory (EM) [6], majority-based tracking-forecast memory (MTFM) [7], and delayed stochastic (DS) [8] decoders, providing alternatives for the next-generation high-throughput communication systems.

However, the latency for the MTFM decoder [7] is 800 ns, while the latencies for the min-sum algorithm (MSA)-based decoders are, on average, much smaller [9]. In order to resolve the latency problem, the challenge is focused on reducing the average number of decoding cycles (DCs) required for stochastic decoders. In [10], the authors initialized the first bit transmitted to the hard decision unit using the sign bit for the channel log-likelihood ratio (LLR) in a (48, 24) LDPC code. However, no effective demonstration for long codes was given. In [11], a new hard decision method that used the latest output from a variable node (VN) as the code bit was proposed. This method was able to reduce the entire decoding period, but the decoding performance was not so good. In [12], a clockless decoding method that eliminates the need to synchronize the node signals after each DC was proposed.

In this brief, three different strategies are proposed to reduce the average number of DCs required for stochastic LDPC decoders. First, an initialization method based on LUTs and counters is proposed, which is able to achieve an acceptable bit error rate (BER). Second, a simple postprocessing strategy based on a modified bit flipping (BF) algorithm is combined with the stochastic decoding procedure in order to ensure that the stochastic decoders converge more rapidly when the number of parity check errors is below a controlled threshold. Third, a hard decision strategy is proposed that makes use of the posterior information rather than the soft messages from the VN to the check node (CN). For the (6, 32)-regular (2048, 1723) LDPC code adopted in the IEEE 802.3an standard, simulation indicates a 73.6% reduction in the average number of DCs at 4.4 dB. Hardware implementation shows that VN units occupy the biggest area of the proposed decoder and the area of the decoder is reduced compared to the decoders in [6], [7], and [9].

II. Stochastic Decoding

Stochastic LDPC decoding has evolved from the sum-product algorithm (SPA) [13], [14], which is an iterative decoding algorithm that exchanges the probabilities between the VNs and the CNs. Stochastic computation systems use Bernoulli sequences to represent decoding messages in the probability domain, where the ratio of 1 s in the sequence represents the message probability [5]. In the conventional SPA, the CNs and the VNs exchange complete decoding messages in each iteration. However, in stochastic decoding, the CNs and the VNs only exchange 1 b of the message represented using the Bernoulli sequence during each DC [5].

A. Algorithm Description

To illustrate the concept more conveniently, the stochastic decoding is introduced using a degree-3 CN and a degree-2 VN.

1) Initialization: Initialize the sequence from VN v_i using the probability P^i_{init} for the channel LLR L_i according to

$$P_{i\to j} = P^i_{\text{init}} = \frac{e^{L_i}}{(e^{L_i}+1)}. \quad (1)$$

Manuscript received August 4, 2015; revised November 21, 2015; accepted February 19, 2016. Date of publication February 25, 2016; date of current version August 25, 2016. This work was supported in part by the National Natural Science Foundation of China under Grant 61306043, by Shanghai Natural Science funding under Grant 13ZR1401800, by the National Natural Science Foundation of China under Grant 61234002, by the National Science and Technology Major Project of China under Grant 2011ZX 03003-003-03, and by the Ministry of Science and Technology of Taiwan under Grant MOST 103-2622-E-007 -029 -CC2. This brief was recommended by Associate Editor C. Shing-chow.

D. Wu, Y. Chen, Q. Zhang, and X. Zeng are with the State Key Laboratory of Application Specific Integrated Circuit & System, Fudan University, Shanghai 200433, China (e-mail: wud12@fudan.edu.cn; chenyun@fudan.edu.cn; 13212020038@fudan.edu.cn; xyzeng@fudan.edu.cn).

Y. Ueng is with the Department of Electrical Engineering and the Institute of Communications Engineering, National Tsing Hua University, Hsinchu 30013, Taiwan (e-mail: ylueng@ee.nthu.edu.tw).

Digital Object Identifier 10.1109/TCSII.2016.2535038

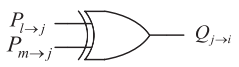

Fig. 1. Structure of a degree-3 CN unit.

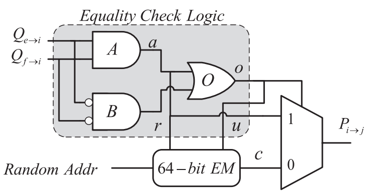

Fig. 2. Structure of a degree-2 VN unit.

2) CN Update: Each VN v_i sends the sequence having probability $P_{i\to j}$ to the corresponding CN c_j, and the c_j output sequence having the probability $Q_{j\to i}$ is updated as

$$Q_{j\to i} = P_{l\to j}(1 - P_{m\to j}) + P_{m\to j}(1 - P_{l\to j}). \tag{2}$$

3) VN Update: Each CN c_j sends the sequence having probability $Q_{j\to i}$ to the corresponding VN v_i, and the v_i output sequence having the probability $P_{i\to j}$ is updated as

$$P_{i\to j} = \frac{Q_{e\to i}Q_{f\to i}}{Q_{e\to i}Q_{f\to i} + (1 - Q_{e\to i})(1 - Q_{f\to i})}. \tag{3}$$

In a practical stochastic decoder, (4) rather than (3) is usually adopted [6], which assumes that $Q_{e\to i}$ and $Q_{f\to i}$ are saturated soft messages that are close to either 0 or 1

$$P_{i\to j} = Q_{e\to i}Q_{f\to i}. \tag{4}$$

4) Termination: Set the VN code word bit to 1 if $P_{i\to j} > 0.5$. Discontinue decoding if the termination condition is satisfied. Otherwise, return to step 2).

B. Stochastic Decoding Elements

The CN function is executed using an XOR gate, as shown in Fig. 1. The computation is performed with assumptions that uniform distribution and no relativity exist in the two input Bernoulli sequences in adequate clock cycles, guaranteeing that any errors that occur between the output probability in Fig. 1 and that produced using (2) are negligible. For the input sequence having the probability $P_{l\to j}$ ($P_{m\to j}$), the probability that bit $l = 1$ ($m = 1$) is $P_{l\to j}$ ($P_{m\to j}$). The probability that output bit $j = 1$ is calculated as

$$\begin{aligned} P(j = 1) &= P(l = 1) \cdot P(m = 0) + P(l = 0) \cdot P(m = 1) \\ &= P_{l\to j} \cdot (1 - P_{m\to j}) + (1 - P_{l\to j}) \cdot P_{m\to j}. \end{aligned} \tag{5}$$

With an infinite sequence length, the probability that $j = 1$ is $Q_{j\to i}$, i.e., the exact probability of the output Bernoulli sequence. There are a total of 32 inputs and 32 outputs for a degree-32 CN, and each output is generated by an XOR for the other 31 inputs, except for the corresponding input.

The VN function described in (4) is shown in Fig. 2. The structure is based on a 64-b EM [6]. The shaded rectangle is called the Equality Check Logic (ECL) block. One input bit is sourced from the CN, and the other is sourced from the channel information. When the two inputs are the same, the EM is considered to be in a nonhold state and is updated using the regenerative bit r, and the output is equal to the input. When one of the inputs differs from the other, the EM is considered to be in the hold state and remains the same, and the output is equal to the conservative bit c chosen from the EM based on a 6-b random address.

For the input sequence $\{e\}$ ($\{f\}$) having the probability $Q_{e\to i}$ ($Q_{f\to i}$), the probability that $e=1$ ($f=1$) is $Q_{e\to i}$ ($Q_{f\to i}$). For the output sequence $\{a\}$ of AND gate A, the probability that $a=1$ is $P(a=1) = P(e=1) \cdot P(f=1) = Q_{e\to i} \cdot Q_{f\to i}$. For the output sequence $\{o\}$ of OR gate O, the probability that $o=1$ is $P(o=1)$. As only the regenerative bit r is stored in the EM, the ratio of 1 s in the EM is equal to $P(a=1)$ theoretically. Hence, the probability that the conservative bit c from the EM is 1, i.e., $P(c = 1)$, is equal to $P(a = 1)$. Thus, the probability that the VN output bit $i = 1$ can be calculated as

$$\begin{aligned} P(i = 1) &= P(o = 1) \cdot P(a = 1) + P(o = 0) \cdot P(c = 1) \\ &= P(o=1) \cdot P(a = 1) + (1 - P(o=1)) \cdot P(a=1) \\ &= P(a = 1) = Q_{e\to i} \cdot Q_{f\to i}. \end{aligned} \tag{6}$$

A degree-6 VN can be composed of two degree-3 subunits and one degree-2 subunit, as mentioned in [6]. The degree-2 subunit utilizes a 2-b internal memory instead of the EM in order to reduce the number of hold states, while the degree-3 subunit utilizes the EM. By implementing this method, the probability in nonhold states can be effectively reduced without needing to enlarge the area of the VNs.

III. Proposed Decoder Architecture

Fig. 3 shows the proposed decoder architecture. First, the channel LLRs in the Input Buffer are converted to the initialization values and the channel information in the probability domain, using the LUTs in the VN Initialization block and the LLR to Probability block, respectively. Second, in the Channel Bit Generation block, the random Bernoulli sequences for the VNs are generated using the channel probabilities and the random numbers from the Random Number Generation block. In the meantime, the initialization values are sent to the VN Update block. Third, the stochastic decoding functions in (2) and (4) are performed in the CN Update block and the VN Update block, respectively. Fourth, the Hard Decision block generates the code words and sends them to the Post Processing block to determine whether the postprocessing can begin. If the postprocessing is not started, the VNs and the CNs continue to exchange message bits. In the postprocessing, if all parity checks are satisfied or if the maximum number of DCs is reached, the code word is stored in the Output Buffer block and the decoding operation terminates.

There are a number of factors that cause the prolonged DCs required for stochastic decoders. First, the cycle time required for the internal transmission to achieve the required necessary data precision is extensive. Second, the stochastic LDPC decoder requires a lengthy initialization period [6] compared to the Normalized MSA (NMSA)-based LDPC decoder, which only consumes a single clock cycle to store the channel LLRs as the initial soft messages. Third, the degradation in the switching activity further reduces the convergence speed, subsequently leading to the latching problem referred to in [5].

The four shaded blocks depicted in Fig. 3 are the modified modules in the proposed architecture. The remaining unshaded modules are the same as those presented in [6]–[8]. The LUT-based initialization procedure is employed in both the

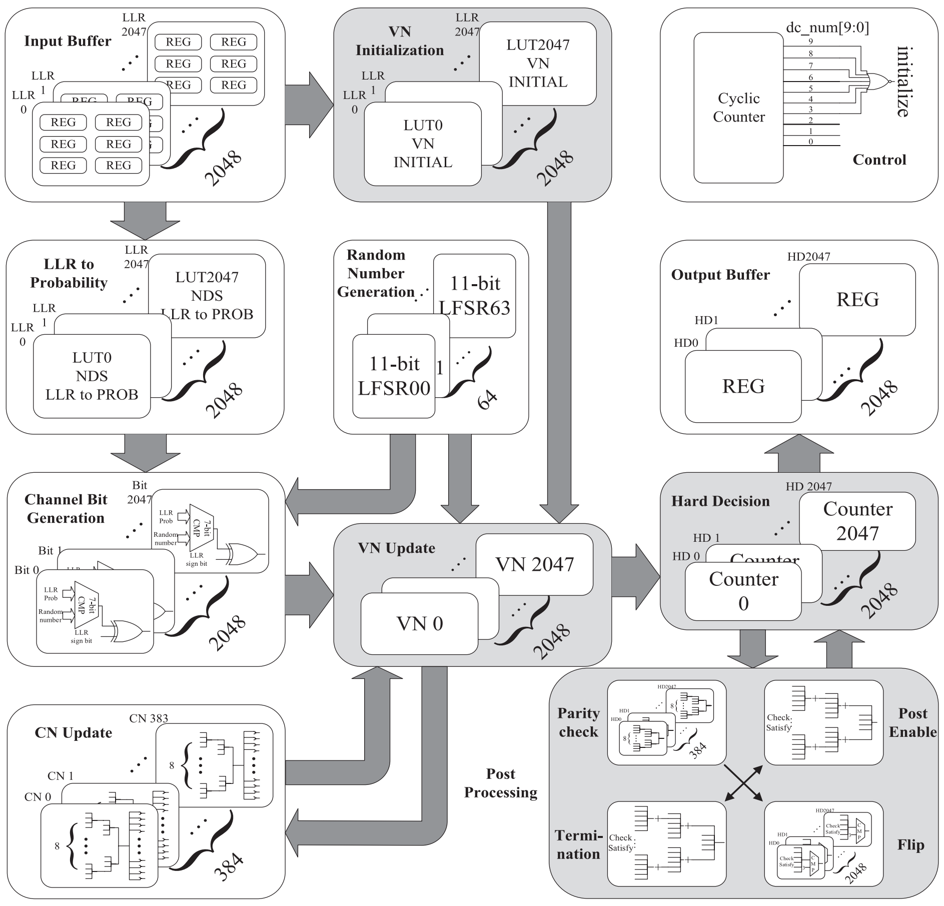

Fig. 3. Proposed decoder architecture.

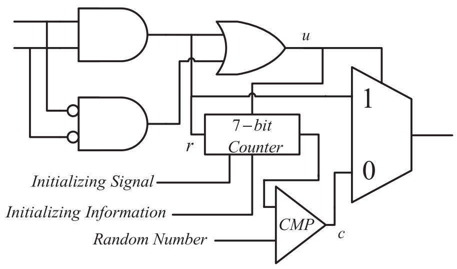

Fig. 4. Modified degree-2 subnode.

VN initialization unit and the VN update unit, the modified BF-based postprocessing procedure is employed in both the hard decision unit and the postprocessing unit, and the posterior-information-based hard decision method is employed in the VN update unit.

A. LUT-Based Initialization

The proposed LUT-based initialization is intended to shorten the initializing period that is required for the EM-based stochastic decoder, and provides precise channel values to accelerate the convergence. In this brief, a counter-based VN unit is used in both the LUT-based initialization strategy and the posterior-information-based hard decision strategy. The structure of the proposed degree-2 VN subunit is shown in Fig. 4. Compared to the EM-based VN unit, the proposed VN unit replaces the EM with a 7-b up/down saturation counter, which stops counting after reaching the up/down boundary. The counter can be initialized in one clock cycle based on the initializing value generated by the VN Initialization block, which is less than that required for the EM-based decoder.

Using the proposed one-step initializing structure based on LUTs, the counter is initialized based on the initializing information if the initializing signal portrayed in Fig. 4 is 1. The subsequent comparator compares the random number from the Random Number Generation block and the counter value in order to generate the conservative bit which is used as the output in the hold state. If the random number is smaller than the counter value, the output value is set to 1; otherwise, it is set to 0. In the nonhold state, the input logic value, which is the same as the regenerative bit, is directly passed to the adjacent CNs. The function of the counter-based VN is able to operate in the same manner as the EM-based VN, tracking the probability of the regenerative bit. When considering the hardware implementation, each VN unit is matched with a single LUT. For the proposed decoder, each channel LLR is quantized to a 6-b signed binary value with a 3-b fraction, and the corresponding probability is a 7-b binary value.

The initializing value is related to the theoretical number of 1 s in the EM based on a linear mapping, which results in a more accurate initialization, a more rapid convergence, and a better BER performance. A 7-b counter-based VN unit produces a similar performance as a 64-b EM decoder. Provided that the number of 1 s in the 64-b EM is $x \in [0, 64]$, the number of 0 s is $64-x$, and the number of the counter is $64+x-(64-x)=2x$.

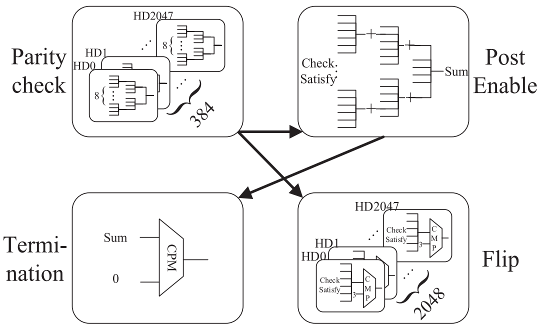

Fig. 5. Post processing unit.

B. BF-Based Postprocessing

As cycles exist in the LDPC factor graphs, parity check errors are difficult to correct due to the saturation of the soft messages. In particular, in a stochastic decoder, the saturation of the inner soft messages results in a latching issue [5], which consequently decelerates the convergence. Simulation results show that, when the number of bit errors that occur in a stochastic LDPC decoder is less than a certain threshold, the decoding process decelerates, which is similar to the statement presented in [8]. As BF decoders are able to perform well despite the saturated messages which usually appear at high SNRs, a modified BF-based postprocessing strategy is adopted here. Furthermore, a better BER performance can be achieved owing to the combination of two different algorithms.

The proposed architecture consists of four parts, as shown in Fig. 5. The Parity Check unit is similar to the CN Update block depicted in Fig. 3, executing 384 parity checks and passing the results to the other units. When all 384 results are 0, the current code word is deemed to be correct. Otherwise, the sum of all the parity check results would define the operation of the decoder in the Post Enable unit. The Post Enable unit then calculates the total of all the parity check results. If the total is less than the threshold of 30, which is chosen based on the decoding performance, the postprocessing operation is enabled. The decoding operation is terminated if the sum of all the parity check results is 0. The sum obtained by the Post Enable unit can be multiplexed here. The Flip unit computes the results for the six parity checks corresponding to each degree-6 VN unit. If the sum is greater than 3, the VN hard decision is considered to be wrong, and the corresponding code bit is flipped, which is different from the BF algorithm that flips the bit having the highest number of parity check errors. According to the simulation results, the proposed decoding procedure will, in most situations, terminate within three DCs after the postprocessing stage has begun. If the decoding process has not terminated, the decoding task is considered to have failed and the decoder stops decoding.

C. Posterior-Information-Based Hard Decision

In [6]–[8], the VN to CN (V2C) messages are used to perform the hard decision rather than the posterior information, resulting in a decreased convergence speed. In the proposed hard decision unit, the posterior information, rather than the V2C messages, is used to improve the performance in a stochastic decoder. The soft message bit is calculated by the VN subunit using an ECL block according to (4). Similarly, the posterior bit can be calculated using an additional ECL block, as shown in Fig. 6. The output from a VN unit to its corresponding hard decision unit is the posterior bit. If the bits to and from a specific CN (e.g., the in_0 and out_0 bits shown in Fig. 6) are the same, the update signal for the proposed VN unit is set to 1, and the hard decision unit is updated based on the posterior bit. Otherwise, the corresponding counter value does not get updated in the subsequent DC.

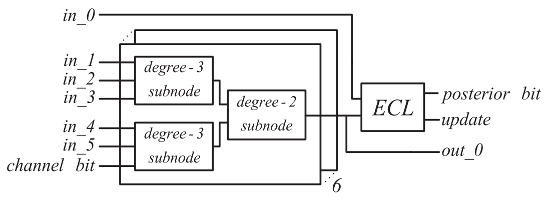

Fig. 6. Posterior bit for hard decision.

TABLE I
PROPOSED MODULES

Module	Parameter	
	Scaled Area	*Percentage*
VN Update	1.33 mm^2	48.4%
VN Initialization	0.32 mm^2	11.6%
Post Processing	0.17 mm^2	6.2%
Hard Decision	0.20 mm^2	7.3%

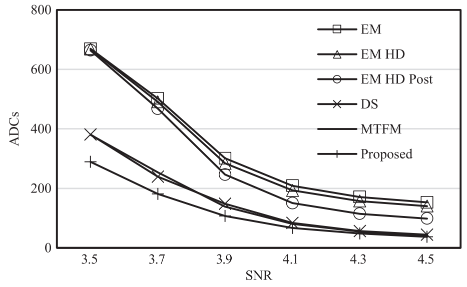

Fig. 7. Average reduction in DCs.

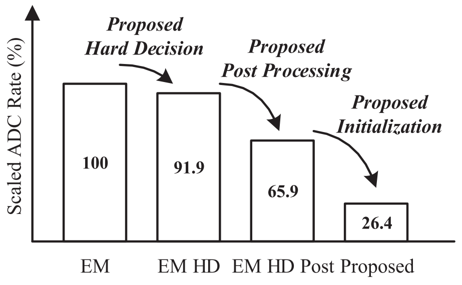

Fig. 8. ADC reduction rate at an SNR of 4.4 dB.

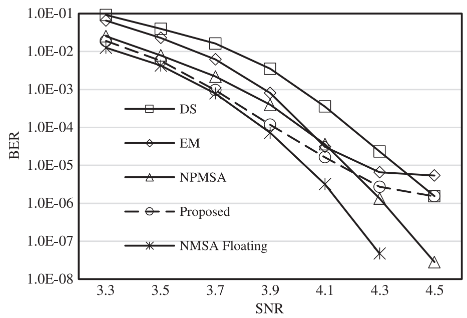

Fig. 9. BER in Additive White Gaussian Noise channel with BPSK.

TABLE II
COMPARISON TABLE

Performance	Implementation	Technology	Area (mm^2)	Scaled Area In 65nm (mm^2)	Frequency (*MHz*)	Code	DC/Iteration	Throughput (*Gb/s@dB*)	Throughput/Area (*Gb/s/mm²@dB*)
This Work	Fully parallel stochastic	65nm	2.75	2.75	500	(2048, 1723)	300 DCs	23.92@4.4 82.58@5.5	8.7@4.4 30.02@5.5
EM [6]	Fully parallel stochastic	FPGA	46097 Slices	46097 Slices	222	(1056, 528)	700 DCs	1.66@4.25	-
MTFM [7]	Fully parallel stochastic	90nm	6.38	3.19	500	(2048, 1723)	400 DCs	21.3@4.4 61.3@5.5	6.8@4.4 19.2@5.5
DS [8]	Fully parallel stochastic	90nm	3.93	1.97	750	(2048, 1723)	400 DCs	32.7@4.4 172.4@5.5	16.6@4.4 87.3@5.5
NPMSA [9]	Fully parallel	90nm	9.6	4.8	199.6	(2048, 1723)	9 Iterations	45.42@4.4	9.46@4.4

The hard decision also makes use of the up/down saturation counters, in which the sign bit represents the result of the hard decision. To ensure that the postprocessing and the one-step initialization operations perform better, these counters have been modified. During the initialization stage, these counters are initialized using either 1 $(\text{LLR} \geq 0)$ or -1 $(\text{LLR} < 0)$, depending on the channel LLRs. In the postprocessing stage, the sign bit register for the counter together with the Post Processing block constitute a modified BF decoder. The posterior information bits terminate the update process for the Hard Decision block, and except for the sign bit, the bits that are present in the counter remain the same. When the postprocessing operation is enabled, the counters in the Hard Decision block stop counting and the Flip block becomes operational. The sum of all the parity check results may change before the Flip block takes effect, causing the wrong code bits to be flipped. As a result, a counter is introduced here in order to synchronize all the signals and ensure that the decoder functions correctly.

IV. PERFORMANCE EVALUATION

A. Implementation

The proposed decoder is synthesized using 65-nm technology. Table I provides an overview of the area percentage of modified blocks in Fig. 3. The VN Update block accounts for the biggest area. The remaining three newly added blocks account for only 25.1% of the total area.

Fig. 7 shows the average DC (ADC) values when using the three proposed strategies. Also, included in Fig. 7 are the ADC values for the decoders based on the MTFM [7] and the DS [8] techniques. It can be observed that, at an SNR of 4.5 dB, the ADC values for the MTFM and DS decoders and the proposed decoder are nearly the same. Fig. 8 shows the ADC reduction ratio for the three proposed strategies. The proposed architecture achieves a reduction of 73.6% in the ADC compared to the EM decoder, resulting in an increase in throughput of four times with an area increase of only 25.1%.

B. Comparison

Fig. 9 shows the BER performance of the (2048, 1723) LDPC code respectively using the EM [6], the DS [8], the NMSA, the normalized probabilistic MSA (NPMSA) [9], and the proposed algorithms. Except for the NMSA curve, all curves are fixed point. Consequently, it is obvious that the error floor for the EM-based decoder can be effectively lowered by implementing the proposed strategies, meaning that the proposed decoder can achieve a satisfactory performance. A comparison between the different decoders is presented in Table II. As can be seen, the throughput/area efficiency of the DS decoder is better, but its BER performance is worse. Although the number of DCs required by this work is reduced compared to other stochastic decoders, it is not comparable with that of the NPMSA's, as the concept of DCs in stochastic decoding is inequivalent to the concept of iterations used in MSA [5]. The proposed decoder and the NPMSA decoder have a similar throughput/area efficiency at an SNR of 4.4 dB.

V. CONCLUSION

Three decoder strategies have been presented, including LUT-based initialization, BF-based postprocessing, and the posterior-information-based hard decision technique, designed to reduce the number of DCs required for stochastic LDPC decoders. Simulation has shown that the average number of DCs required can be reduced by 73.6% in comparison with the EM decoder. Hardware implementation using SMIC 65-nm technology shows that the proposed decoder achieves a throughput of 82.58 Gb/s at an SNR of 5.5 dB, which is an increase of 34.7% when compared to the MTFM decoder, and achieves a better BER performance compared to other stochastic decoders presented in [6]–[8].

REFERENCES

[1] R. G. Gallager, *Low Density Parity Check Codes*. Cambridge, MA, USA: MIT Press, 1963.

[2] The IEEE P802.3an 10GBASE-T Task Force. [Online]. Available: www.ieee802.org/3/an

[3] D. Wu *et al.*, "A high-throughput LDPC decoder for optical communication," in *Proc. IEEE 10th Int. Conf. ASIC*, 2013, pp. 1–4.

[4] V. Gaudet and A. Rapley, "Iterative decoding using stochastic computation," *Electron. Lett.*, vol. 39, no. 3, pp. 299–301, Feb. 2003.

[5] S. S. Tehrani, S. Mannor, and W. J. Gross, "Survey of stochastic computation on factor graphs," in *Proc. 37th Int. Symp. Multiple-Valued Logic*, May 2007, pp. 1–6.

[6] S. Sharifi Tehrani, S. Mannor, and W. J. Gross, "Fully parallel stochastic LDPC decoders," *IEEE Trans. Signal Process.*, vol. 56, no. 11, pp. 5692–5703, Nov. 2008.

[7] S. S. Tehrani *et al.*, "Majority-based tracking forecast memories for stochastic LDPC decoding," *IEEE Trans. Signal Process.*, vol. 58, no. 9, pp. 4883–4896, Sep. 2010.

[8] A. Naderi, S. Mannor, M. Sawan, and W. J. Gross, "Delayed stochastic decoding of LDPC codes," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5617–5626, Nov. 2011.

[9] C.-C. Cheng, J.-D. Yang, H.-C. Lee, C.-H. Yang, and Y.-L. Ueng, "A fully parallel LDPC decoder architecture using probabilistic min-sum algorithm for high-throughput applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 9, pp. 2738–2746, Sep. 2014.

[10] M. Maamoun, R. Bradai, A. Naderi, R. Beguenane, and M. Sawan, "Controlled start-up stochastic decoding of LDPC codes," in *Proc. IEEE 11th Int. New Circuits Syst. Conf.*, 2013, pp. 1–4.

[11] K.-L. Huang, V. Gaudet, and M. Salehi, "Output decisions for stochastic LDPC decoders," in *Proc. 48th Annu. Conf. Inf. Sci. Syst.*, 2014, pp. 1–5.

[12] C. Ceroici and V. C. Gaudet, "FPGA implementation of a clockless stochastic LDPC decoder," in *Proc. IEEE Workshop Signal Process. Syst.*, 2014, pp. 1–5.

[13] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[14] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.