A Data Structure-Based Approximate Belief Propagation Decoder for Polar Codes

Menghui Xu[1,2,3], Weikang Qian[4], Zaichen Zhang[2,3], Xiaohu You[2,3], Chuan Zhang[1,2,3,*]

[1]Lab of Efficient Architectures for Digital-communication and Signal-processing (LEADS)

[2]National Mobile Communications Research Laboratory, Southeast University, Nanjing, China

[3]Purple Mountain Laboratories, Nanjing, China

[4]University of Michigan-Shanghai Jiao Tong University Joint Institute, Shanghai Jiao Tong University, Shanghai, China

{mhxu, chzhang}@seu.edu.cn, qianwk@sjtu.edu.cn

***Abstract*—Polar code, as the first code that can probably achieve the capacity of B-DMCs, has received great attention. Belief propagation (BP) decoding algorithm, a paralleled de-coding approach for polar codes, suffers from high hardware complexity. In this paper, we devoted ourselves to proposing a data structure-based approximate BP decoder for polar code. Multiple simulations have been done. The simulation results show that by reforming the data structure of the received channel message and introducing the approximate computing schemes, significant hardware reduction has been made compared to its conventional counterpart. The hardware architecture and corresponding implementation results are also given in this paper.**

***Index Terms*--Polar code, belief propagation decoder, approximate computing, hardware consumption.**

I. Introduction

Polar code, which has been discovered by Arıkan in his breakthrough paper [1], has received significant attentions. It is known as the first error correction code which is able to reach the Shannon capacity for binary-input discrete memoryless channels. For now, polar code has been chosen by Enhanced Mobile Broadband (eMBB) control channel for 5G. The successive cancellation (SC) algorithm and belief propagation (BP) algorithm are the two conventional decoding algorithms for polar code [2]. The SC decoder decodes bit in a serial manner with low hardware complexity, but it requires long decoding latency. On the contrary, the belief propagation algorithm is a more popular implementation for polar code for its inherent high parallelism. It decodes faster and therefore has a higher throughput than the serial SC decoder. The complexities for SC decoding algorithm is $\mathcal{O}(LN \log N)$ [3]–[5] and $\mathcal{O}(IN \log N)$ [6] for BP decoding algorithm. Today, faces with the data transmission requirements raised by 5G communication system, these two polar decoders have to increase its list size or iteration number. Therefore, reducing the implementation complexity of the polar decoders has been a great issue for the designers.

Plenty of work has been done to prove that 100% accuracy is not always necessary in computation [7], [8]. For example, in the error-resilient systems like handwritten recognition and image processing, a few erroneous pixels do not affect human rec-ognizing the image. However, approximate computing circuit has the unique advantage of high hardware efficiency as they trade computing accuracy for hardware consumptions. As the decoders are always work in noisy channels, decoders for error correction code is also like an noisy system. In addition, BP decoders can be implemented under noisy circumstance [9]. Therefore, we introduce the approximate computing scheme to relieve contradictions between performance and consumption. Researches for approximate LDPC decoders has been done in prior works [10], [11].

In this paper, we proposed a data structure-based approximate BP decoder by reforming the data structure of channel massage and making use of the high hardware efficiency of approximate computing scheme. The remainder of this paper is organized as follows. Section II gives a brief review of polar codes and belief propagation decoding algorithms. Section III presents the architecture of the proposed data struture-based approximate BP decoder. In Section IV, designing methodology is given. Section V gives the implementation results and compares the hardware consumption of architecture with the conventional one. Section VI draws conclusion for this paper.

II. Review of Polar Codes and BP Decoding

A. Polar Codes

In general, polar code is defined by the parameter set of (N, K, A, u_{A^c}). $N = 2^n$ represents the length of polar code, K denotes the number of information bit, A denotes the set of information bit, and u_{A^c} denotes value for frozen bits. In the process of polar encoding, we constructed vector $x_1^N = (x_1, x_2, ..., x_{N-1}, x_N)$ based on source vector $u_1^N = (u_1, u_2, ...u_{N-1}, u_N)$ as follows:

$$x_1^N = u_1^N G_N = u_1^N B_N F^{\otimes n}, \tag{1}$$

where G_N denotes the generator matrix and B_N is the bit reversal permutation matrix. $F^{\otimes n}$ represents the *Kronecker* power for $F = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$. After massage vector x_1^N is sent over channels W^N, the polar decoder receives the channel output $y_1^N = (y_1, y_2, ..., y_{N-1}, y_N)$. Finally, polar decoder decides the vector u_1^N based on the channel output y_1^N, A and u_{A^c}. More details can be referred to [1].

B. BP Decoding Algorithm

The factor graph [12] for a 8-bit polar code is given in Fig. 1. There are two types of nodes in this factor graph. Coordinate (i, j) means the labeled node is in stage number i and column number j. The received channel massages are passed and updated iteratively through this factor graph.

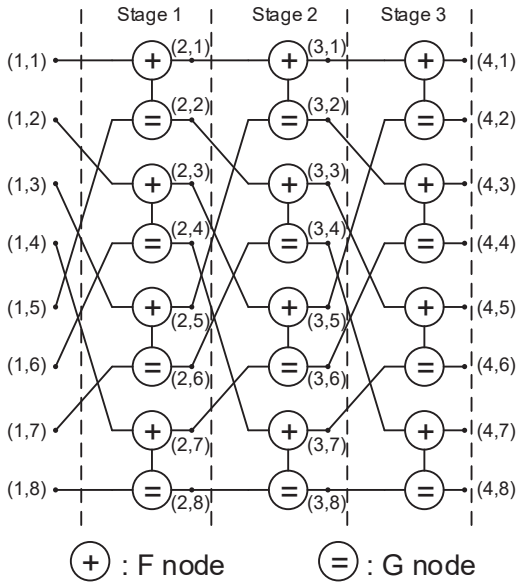

Figure 1. 8-bit belief propagation decoding.

The channel massage in BP decoding is represented by log likelihood ratio (LLR). Two types of LLR massages are involved in BP decoder: left-to-right message R and right-to-left message L. The channel massage are initiated as follows:

$$L_{n+1,j} = ln\frac{P(y_j|x_j = 0)}{P(y_j|x_j = 1)}, \tag{2}$$

$$R_{1,j} = \begin{cases} 0, & \text{if } j \in A, \\ \infty, & \text{if } j \in A^c. \end{cases} \tag{3}$$

In this paper the round-trip schedule is employed in the BP decoding. In BP decoding, channel messages are undated iteratively according to Eq. (4).

$$\begin{cases} L_{i,j} = f(L_{i+1,2j-1}, g(L_{i+1,2j}, R_{i,j+N/2})), \\ L_{i,j+N/2} = g(f(R_{i,j}, L_{i+1,2j-1}), L_{i+1,2j}), \\ R_{i+1,2j-1} = f(R_{i,j}, g(L_{i+1,2j}, R_{i,j+N/2})), \\ R_{i+1,2j} = g(f(R_{i,j}, L_{i+1,2j-1}), R_{i,j+N/2}). \end{cases} \tag{4}$$

where

$$f(x, y) = \frac{1 + xy}{x + y}, \tag{5}$$

$$g(x, y) = x \cdot y. \tag{6}$$

When BP decoder finish the decoding iterations, u_1^N can be decided at the first column as follows:

$$\hat{u}_j = \begin{cases} 0, & \text{if } R_{1,j} \times L_{1,j} \geq 0, \\ 1, & \text{else.} \end{cases} \tag{7}$$

III. Proposed Approximate BP Decoder

A. Quantization Schemes

Before doing data structure conversion, we need to fix the proper quantization scheme for the original channel massage. We use $(256, 128)$ polar code for simulations. According to the simulations, channel massages are mainly distributed in the interval of $[-8, 8]$. Therefore, for $(256, 128)$ polar code, we use 1 bit sign and 3 bits integers for the original channel massage.

Fig. 2 is the simulation for $(256, 128)$ polar code. Here, $(1 - 3 - f)$ denotes the channel massage is represented by 1 bit sign, 3 bits integer and f bits fractional. As the simulation results shown, the quantization scheme $(1 - 3 - 3)$ performs the best compared with other quantization schemes. So in this paper, we use $(1 - 3 - 3)$ as the quantization scheme for the original channel massage in order to achieve satisfying decoding performance.

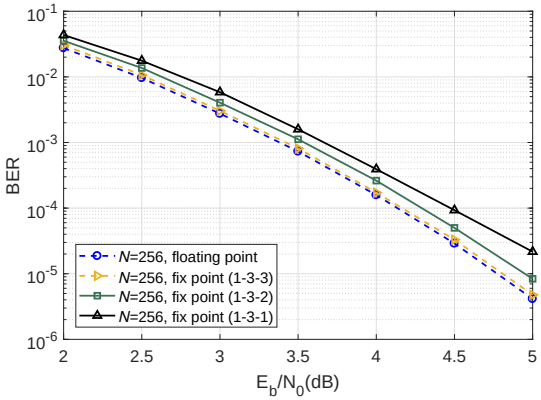

Figure 2. Simulation results for BP decoders.

B. Data Structure Conversion

As shown in Eq.s (5) and (6), the magnitude comparison is the main operation for G nod in BP decoding. Therefore, in the conventional BP decoder, channel massages are mostly stored in the sign-magnitude form (SMF) for computation. However, 100% accurate data are not necessary in approximate computing, so it is reasonable for us to extract the key information from the original data and omit the others.

Consider that the major computations of the G node processing are magnitude comparisons, the key information for G node is the locations of the first nonzero bit of the two input data. On the other hand, additions are carried out in the F node. The result of the processing in F node is largely determined by the most significant bits (MSB). Therefore, the key information for F node is the MSB of the two input data. Since the first bit of MSB is fixed to 1, we only need to store the second and the third significant bits.

In this paper, a sign-location-magnitude form (SLMF) is first proposed in Fig. 3. According to Section III.A, we presume that the original data stored in the SMF is 1 bit sign and 6 bits magnitude. As shown in Fig. 3, the sign of the input data is still stored in the first bit. Then, the next 3 bits represent the location of the first nonzero bit of the original input data. After that, the k-bit MSB of the magnitude will be reserved and the rest are omitted. Here k is an parameter and in this paper, we let $k = 2$.

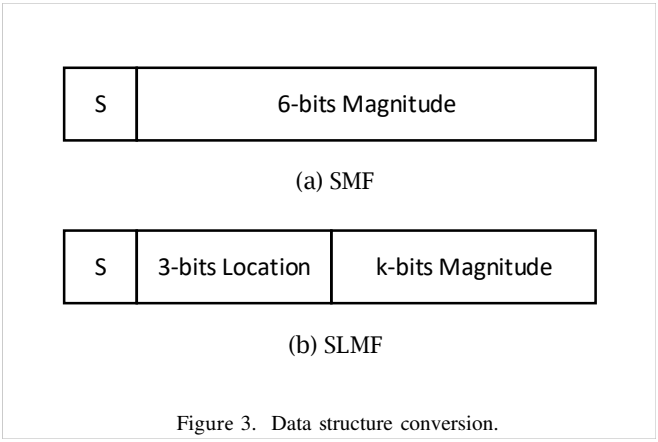

Figure 3. Data structure conversion.

For example, shown in Fig. 4, the received channel massage is 0001011 which is stored in the SMF. In the data structure conversion, first we keep the sign bit unchanged. Then, since the first nonzero bit is the fourth bit of the input data, the location bits in the SLMF are set to 110. After that, the second and the third significant bits is stored in the magnitude bits in SLMF, which are 0 and 1 in this case. The rest bits of the original input are just ignored. Therefore, after the data structure conversion, the received channel massage 0001011 is converted to its SLMF, 011001. Compared with its SMF, the key information of the input data is retained. The length of the data has been reduced to 6 bits, which will reduce the hardware consumption of the polar decoders.

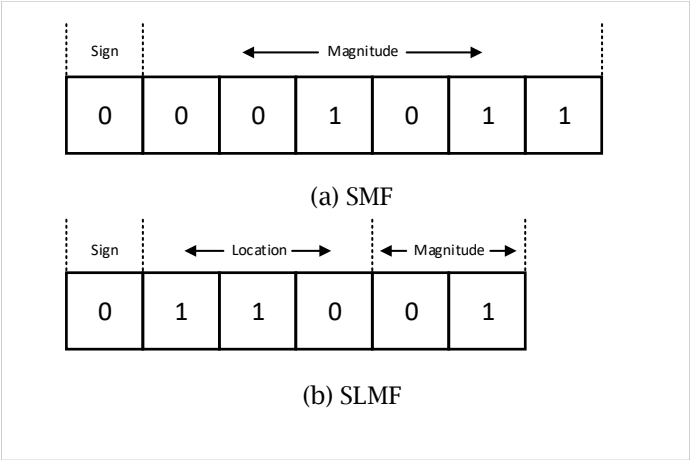

Figure 4. Data Structure Conversion for 0001011.

C. Conventional Architecture for G Node

As mentioned above, channel massages are mostly stored in the sign-magnitude form (SMF) for computation in the conventional BP decoder. As shown in Fig. 5, let a_s and b_s denote the sign bits of two input data a and b. a_M and b_M denote the magnitude bits of the input data. The output data S is consist of sign bit s_s and magnitude bit s_M.

In the conventional architecture, the G node compares two SMF input from the high-order to the low-order bit by bit until it finds two different bits with the same location. However, when the magnitudes of two input data are very close, it becomes a waste of time and power to compare in this way. The conventional architecture may compare almost every bit to figure out the smaller one of the two input data. But this is meaningless. The two numbers are actually the same magnitude. Therefore, the choice of number for the out put data may not have significant affection on the final decoding performance.

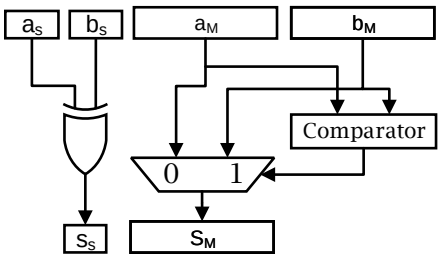

Figure 5. Conventional G node.

D. Proposed G Node

Here, a data structure-based approximate architecture for G node is proposed. The architecture is based on data in sign-location-magnitude form (SLMF). For the proposed architecture of approximate G node, only the location bits of the input data are compared. The rest bits are directly chosen according to the comparison result of the location bit. Fig. 6 gives the proposed architecture. To be mentioned, the magnitude bits can also be compared if better Error Rate is required.

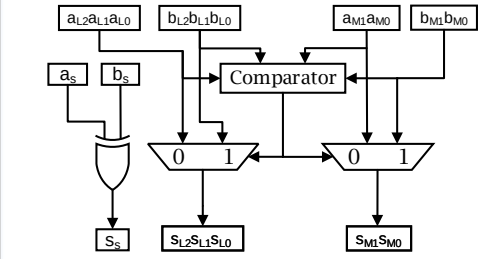

Figure 6. Architecture for data structure-based approximate G node.

If $a_{[L2:L0]} \geq b_{[L2:L0]}$, then we get the output of the approximate G node $s_L = a_L$ and $s_M = a_M$. Otherwise we have $s_L = b_L$ and $s_M = b_M$. However, in case that

$a_{[L2:L0]} = b_{[L2:L0]}$ and the magnitude bits of a is smaller, the approximate G node will predict wrong computing result.

The Error Rate (ER) is analyzed as follows. If we have random number a and b with uniform distribution, the probability of $a_{[L2:L0]} = b_{[L2:L0]}$ and the probability of the $a_{Mk-1:M0} < b_{Mk-1:M0}$ can be computed:

$$\begin{cases} P(a_{[L2:L0]} = b_{[L2:L0]}) = (\frac{1}{2})^3, \\ P(a_{[Mk-1:M0]} < b_{[Mk-1:M0]}) = \frac{2^k-1}{2^{k+1}}. \end{cases}$$

Therefore, the ER of G node is derived by Eq. (8):

$$ER = (\frac{1}{2})^3 \cdot \frac{2^k-1}{2^{k+1}} = \frac{2^k-1}{2^{k+4}}. \tag{8}$$

If better ER is required, the magnitude bits of the input can also be compared when the two location bits are the same. Here, simulation results of $(256, 128)$ polar codes are shown in Fig. 7. When 1 bit of the magnitude bits are compared in G node, the proposed decoder decodes with much less hardware cost with negligible performance loss compared with the conventional decoder.

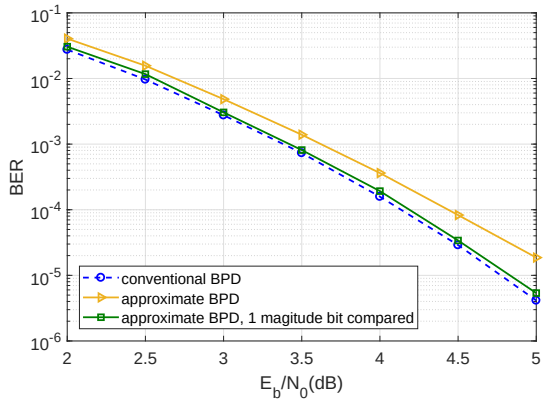

Figure 7. Simulation results of approximate BP decoders.

E. Conventional Architecture for F Node

As afore mentioned, channel massages are stored in the sign-magnitude form (SMF) for computation in the conventional BP decoder. In the conventional F node, addition or subtraction are carried out. The channel messages in SMF could not be used directly in F node. The conventional F node use their complement forms as the input and then doing addtion or subtraction. After the computing in F node, the channel massages are trasmitted to their SMF back.

As Fig. 8 shows, the original input and output channel message are stored in the SMF. Similarly, a_s and b_s denote the sign bits of two input data a and b. a_M and b_M denote the magnitude bits of the input data. The output data S is consist of sign bit s_s and magnitude bit s_M. AOU adds 1 to its input. As shown in the figure, there are totally 3 data format conversion before and after the operation which results in long critical path delay.

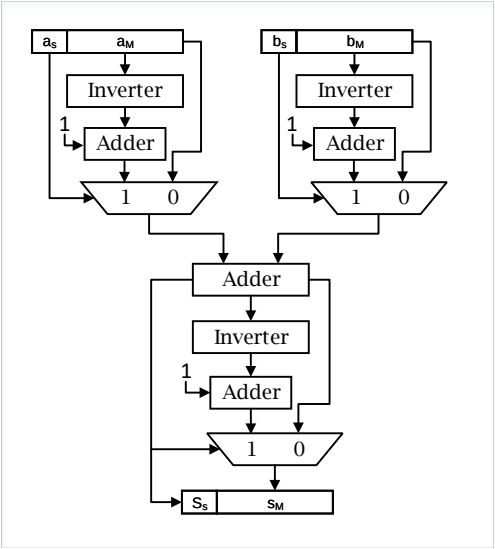

Figure 8. Conventional F node.

F. Proposed F Node

Since mutiple data format conversion result in significant critical path delay, it is better to remove these time-consuming operations. Therefore, an data structure-based approximate architecture for F node is proposed. Details are shown in Fig. 9.

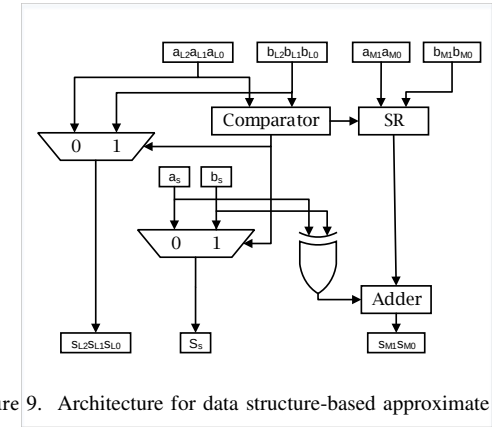

Figure 9. Architecture for data structure-based approximate F node.

As shown in Fig. 9, like we mentioned above, a, b are the SLMF input channel messages and s denotes the output channel message. let a_s and b_s denote the sign bits of two input data a and b. a_M and b_M denote the magnitude bits of the input data. The output data S is consist of sign bit s_s and magnitude bit s_M.

In our proposed F node, the location bits of the two input data is compared first. Then, one of the magnitude bits will be reduced or expanded by certain times through the shift registers in Fig. 9. After that, addition or subtraction is carried out according to the two input sign bits. In this design, we only need to perform a 3-bit addition or subtraction. The highest bit is fixed to 1 and it represents the first nonzero bit of the input. The rest two bits are the magnitude bits of the input.

As Fig. 10 shows, subtraction is performed directly without data format conversion. Therefore, only one of the three data format conversion is reserved for proposed architecture for F

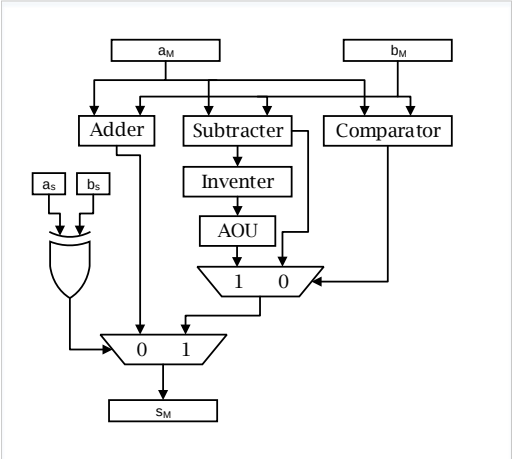

Figure 10. Proposed approximate architecture for the full adder.

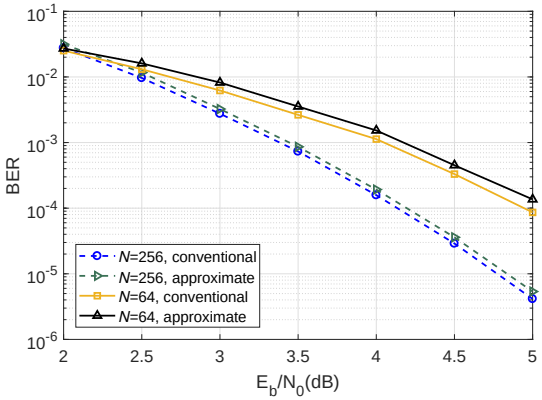

Figure 11. Simulation results of BP decoders with different bits of AOU.

node. $a_M + b_M$ and $a_M - b_M$ are computed simultaneously. After that, one of the computing result will be saved as the final magnitude for the output data s.

TABLE I. Operations of the Full Adder in F Node

a_s	b_s	Relative Size	s_s	s_M
0	0	-	0	$a_M + b_M$
1	1	-	1	$a_M + b_M$
0	1	$a_M \geq b_M$	0	$a_M - b_M$
0	1	$a_M < b_M$	1	$-(a_M - b_M)$
1	0	$a_M \geq b_M$	1	$a_M - b_M$
1	0	$a_M < b_M$	0	$-(a_M - b_M)$

The specific arithmetical operations of the full adder in F node is illustrated in table I. When $a_s \oplus b_s = 0$, the F node implement $a_M + b_M$. Otherwise, it implements $a_M - b_M$. When $a_M \geq b_M$, $s_s = a_s$. Otherwise, $s_s = b_s$.

Compared with the original F node, only 3-bit addition or subtraction is carried out, which significantly reduced the hardware consumption. In addition, subtraction is performed directly without data format conversion. Therefore, only one data format conversion is reserved in F node. Therefore, the critical path delay is significantly reduced.

Simulations are performed using $(256, 128)$ and $(64, 32)$ polar codes in Fig. 11. The channel massage is stored in SLMF, 1 bit sign, 3 bits location and 2 bits magnitude. The result shows that the proposed data structure-based approximate BP decoder decodes with little performance gap compared with the accurate decoder.

IV. Designing Methodology for the Proposed Approximate Polar BP Decoders

Although the proposed data structure-based approximate BP decoder successfully reduces the hardware consumption compared with the conventional counterpart, implemented without a careful designing method could result in significant performance loss for the approximate decoder. Here, a designing methodology for the proposed data structure-based approximate polar BP decoder is given as follows:

Step 1: Before designing a data structure-based approximate polar BP decoder, the received channel massage needs to be converted from its sign-magnitude form to the sign-location-magnitude form. In this paper, we use 1 sign bit, 3 location bits and 2 magnitude bits for $(256, 128)$ polar code.

Step 2: Approximate computing scheme is used for the computing nodes in conventional BP decoder in order to improve the hardware efficiency. Simulation result shows that the hardware consumption is reduced for the proposed approximate architecture without fatal performance loss.

Step 3: The degree of approximation should be adjusted according to the overall error rate and requirement on accuracy.

V. Hardware Implementation and Comparison

The architecture of the proposed data structure-based approximate BP decoder is implemented at the Register Transfer Level (RTL) using Verilog HDL and Synthesized to Nangate 65nm open library with Synopsys Design Compiler. Table II gives the implementation results of the proposed approximate and also conventional architecture.

Benefit from the efficient architecture of the full adder in F node, the proposed data structure-based approximate BP decoder now has a 550 MHz clock frequency. Also, the proposed approximate decoder has significant advantage in hardware efficiency. The cost of arithmetic logic unit (ALUT) approximate decoder is reduced by 19.5% than its accurate counterpart. In addition, since data in SLMF only need 6 bit for storage, the number of registers reduces 22.1%. The total area is reduced by 16.1%. The most important thing is that the hardware reduction mentioned above introduces little performance degredation for the BP decoder.

VI. Conclusion

In this paper, a data structure-based approximate BP decoder for polar code is proposed. Channel massages are converted from its SMF to SLMF. Corresponding approximate computing schemes are used to balance decoding efficiency and hardware efficiency. The designing methodology for the proposed architecture is also presented. Simulation results

TABLE II. IMPLEMENTATION RESULTS FOR $(256, 128)$ POLAR CODE

Hardware overheads	This work	Conventional [13]	Reduction
Max. Frequency (MHz)	550	500	-
ALUT	263,780	327,757	19.5%
Registers	59,845	76,900	22.1%
Area (mm^2)	0.73	0.871	16.1%

proves that, compared with the original one, the proposed data structure-based approximate one has little performance loss. Our future work will be focus on the relationship between decoding performance of the proposed decoder and the error rate in the approximate achitecture.

ACKNOWLEDGEMENT

This work is supported in part by NSFC under grants 61871115 and 61501116, Jiangsu Provincial NSF for Excellent Young Scholars under grant BK20180059, the Six Talent Peak Program of Jiangsu Province under grant 2018-DZXX-001, the Distinguished Perfection Professorship of Southeast University, the Fundamental Research Funds for the Central Universities, the SRTP of Southeast University, and the Project Sponsored by the SRF for the Returned Overseas Chinese Scholars of MoE.

REFERENCES

[1] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[2] E. Arikan, "A performance comparison of polar codes and reed-muller codes," *IEEE Commun. Lett.*, vol. 12, no. 6, pp. 447–449, 2008.

[3] X. Liang, C. Zhang, M. Xu, S. Zhang, and X. You, "Efficient stochastic list successive cancellation decoder for polar codes," in *Proc. IEEE International System-on-Chip Conference (SOCC)*, 2015, pp. 421–426.

[4] F. Leduc-Primeau, S. Hemati, W. J. Gross, and S. Mannor, "A relaxed half-stochastic iterative decoder for LDPC codes," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, 2009, pp. 1–6.

[5] C. Zhang and K. Parhi, "Low-latency sequential and overlapped architectures for successive cancellation polar decoder," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2429–2441, 2013.

[6] E. Arıkan, H. Kim, G. Markarian, U. Ozgur, and E. Poyraz, "Performance of short polar codes under ML decoding," in *Proc. IEEE ICT-Mobile Summit*, Santander, Spain, Jun. 2009.

[7] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *IEEE European Test Symposium (ETS)*. IEEE, 2013, pp. 1–6.

[8] M. Xu, X. Liang, C. Zhang, Z. Wu, and X. You, "Stochastic bp polar decoding and architecture with efficient re-randomization and directive register," in *Proc. Signal Processing Systems (SiPS), IEEE International Workshop on*. IEEE, 2016, pp. 315–320.

[9] C. H. Huang, Y. Li, and L. Dolecek, "Belief propagation algorithms on noisy hardware," *IEEE Trans. Commun.*, vol. 63, no. 1, pp. 11–24, 2015.

[10] Y. Zhou, J. Lin, and Z. Wang, "Efficient approximate layered ldpc decoder," in *Proc. IEEE International Symposium on Circuits and Systems*, 2017, pp. 1–4.

[11] L. R. Varshney, "Performance of ldpc codes under faulty iterative decoding," *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4427–4444, 2011.

[12] J. Sha, X. Liu, Z. Wang, and X. Zeng, "A memory efficient belief propagation decoder for polar codes," *Communications, China*, vol. 12, no. 5, pp. 34–41, 2015.

[13] M. Xu, S. Jing, J. Lin, W. Qian, Z. Zhang, X. You, and C. Zhang, "Approximate belief propagation decoder for polar codes," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 1169–1173.