Enhanced Belief Propagation Decoder for 5G Polar Codes With Bit-Flipping

Yifei Shen, Wenqing Song, Yuqing Ren, Houren Ji, Xiaohu You, *Fellow, IEEE*, and Chuan Zhang, *Member, IEEE*

***Abstract*—Due to its parallel propagation property, the belief propagation (BP) polar decoding can achieve high throughput and has drawn increasing attention. However, the BP decoding is not comparable with the successive cancellation list (SCL) decoding in terms of the error correction performance. In this brief, two BP flip (BPF) decoding algorithms are proposed. Compared with the existing BPF decoding, the generalized BPF (GBPF) decoding identifies error-prone bits more efficiently with a redefinition of bit-flipping. Furthermore, the GBPF decoding is optimized by decreasing the searching range by half, leading to the enhanced BP flip (EBPF) decoding with reduced complexity and improved performance for 5G polar codes. The hardware architecture is provided and implemented using SMIC 65nm CMOS technology. The results show that, compared with the state-of-the-art SC flip decoders, the proposed EBPF decoder exhibits 30% throughput improvement under comparable error correction performance.**

***Index Terms*—Polar codes, belief propagation, bit-flipping, 5G, hardware implementation.**

I. Introduction

As THE standard error correction codes for the fifth generation (5G) enhanced mobile broadband (eMBB) control channels [1], polar codes [2] have attracted considerable attention from both academia and industry. During the standardization of polar codes, the successive cancellation list (SCL) decoding [3] was selected as the baseline algorithm [4], which can achieve satisfactory frame error rate (FER) performance at the cost of long decoding latency. Pursuing a high throughput to meet the eMBB requirements, increasingly many works have focused on the belief propagation (BP) polar decoding [5]. It can outperform the successive cancellation (SC) decoding [6], and its parallel message propagation contributes to high throughput. Moreover, the BP decoding has the potential to support both polar and LDPC codes in uniform hardware [7].

However, the original BP decoding is inferior to the SCL decoding in terms of the FER performance. To tackle this issue, the authors of [8] summarize four solutions, followed by a series of related works for optimization. The four solutions are bit guessing for binary erasure channels [9], artificial noise adding [10], min-sum (MS) function modification [11]–[13], and multi-graph realization (i.e., BP list decoding) [14]–[16]. Similar to the second solution, adding perturbations to a-priori log-likelihood ratio (LLR) information can improve the performance, such as the post-processing method [17]. By contrast, the bit-flipping strategy [18] is a more radical approach, which changes the LLR of identified bits directly to $\pm\infty$. Such a BP flip (BPF) strategy can approach the performance of SCL-16 decoding through additional hundreds of decoding attempts. Despite this, the BPF decoding [18] does not reflect the concept of *flipping*. It tries to assign each error-prone bit to both '0' and '1'. Moreover, the flip set is generated based on the distribution of *rate*-1 nodes under the Gaussian approximation (GA) construction. In other words, the identified bit may yield a high error rate for the SC decoding but exhibit high reliability for the BP decoding.

To identify error-prone bits more efficiently, we propose a generalized BPF (GBPF) decoding algorithm with a redefinition of bit-flipping. Furthermore, the GBPF decoding is enhanced by reducing the identifying range, bringing about the enhanced BPF (EBPF) decoding with slight performance improvement. The EBPF decoding approaches the complexity of BP decoding at medium to high signal-to-noise ratios (SNRs). The hardware of the proposed BPF decoders is synthesized by Synopsys Design Compiler. The results show the advantage of the EBPF decoder in terms of throughput and its area reduction compared with the GBPF.

The remainder of this brief is organized as follows. Section II reviews 5G polar codes and BP decoding. The proposed BPF algorithms are introduced in Section III. Section IV evaluates their FER performance through simulation results. Section V discusses the hardware architecture and shows the implementation results. Section VI concludes this brief.

II. Preliminaries

A. 5G Polar Codes

For 5G eMBB channels, the focus is on the finite code length, of which the mother code length is less than 1024

Manuscript received February 2, 2020; accepted March 7, 2020. Date of publication March 31, 2020; date of current version May 6, 2020. This work was supported in part by NSFC under Grant 61871115 and Grant 61501116, in part by the Jiangsu Provincial NSF for Excellent Young Scholars under Grant BK20180059, in part by the Six Talent Peak Program of Jiangsu Province under Grant 2018-DZXX-001, in part by the Distinguished Perfection Professorship of Southeast University, in part by the Fundamental Research Funds for the Central Universities, and in part by the SRTP of Southeast University. This brief was recommended by Associate Editor C. Studer. *(Corresponding author: Chuan Zhang.)*

Yifei Shen, Yuqing Ren, Houren Ji, Xiaohu You, and Chuan Zhang are with the Laboratory of Efficient Architectures for Digital-Communication and Signal-Processing, Southeast University, Nanjing 211189, China, aslo with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 211189, China, also with the Quantum Information Center, Southeast University, Nanjing 211189, China, and also with the Purple Mountain Laboratories, Nanjing 211189, China (e-mail: chzhang@seu.edu.cn).

Wenqing Song is with the School of Electronic Science and Engineering, Nanjing University, Nanjing 210023, China.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2020.2984536

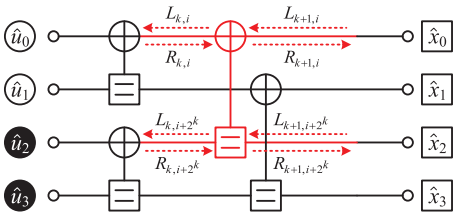

Fig. 1. The factor graph of BP decoding with $N = 4$.

and 512 for uplink (UL) and downlink (DL), respectively. In some cases with long messages or payloads, the information is divided into two segments separately inducted into two polar codes. For a polar code of transmission length E and information dimension K, it can be denoted as an (E, K) polar code. An m-bit cyclic redundancy check (CRC) code is concatenated as an outer code to help the decoding.[1] After channel polarization, the most reliable K' $(= K + m)$ bit channels are allocated for conveying the information and CRC bits, forming the unfrozen set $\mathcal{A}$. The resulting bit vector is denoted as $\boldsymbol{u}$. Afterward, polar encoding is performed by $\boldsymbol{x} = \boldsymbol{u}\mathbf{G}$ to generate the codeword, where $\mathbf{G} = \mathbf{F}^{\otimes n}$ is the n-th Kronecker product of $\mathbf{F} = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$, $n = \log_2 N$, and N is the mother code length. The above operations together with three interleavers and circular buffer form the encoding chain (Fig. 2 in [19]), and their reverse operations take place in the decoding chain. In this brief, we focus on the procedure from the N-length LLRs after sub-block de-interleaver to the estimated message.

B. BP Decoding

The BP decoding iteratively propagates intrinsic messages in parallel. Denote the right-to-left and left-to-right messages as $L_{k,i}$ and $R_{k,i}$, respectively, where k and i represent the stage and row indices. According to the bit channel allocation, the a-priori LLR values of unfrozen bits and frozen bits, i.e., $R_{0,i}$, are 0 and ∞, respectively. The L-message at the rightmost stage, i.e., $L_{n,i}$, is related to the channel output. As shown in Fig. 1, the decoding factor graph is established based on $\mathbf{G}$-matrix, within which the message is propagated following Eq. (1)

$$\begin{cases} L_{k,i} = g(L_{k+1,i}, R_{k,i+2^k} + L_{k+1,i+2^k}), \\ L_{k,i+2^k} = g(L_{k+1,i}, R_{k,i}) + L_{k+1,i+2^k}, \\ R_{k+1,i} = g(R_{k,i}, R_{k,i+2^k} + L_{k+1,i+2^k}), \\ R_{k+1,i+2^k} = g(R_{k,i}, L_{k+1,i}) + R_{k,i+2^k}. \end{cases} \tag{1}$$

The function $g(x, y)$ has an exact form of $\log \frac{1+e^{x+y}}{e^x+e^y}$ and can be approximated to the MS function. To mitigate the approximation error, the offset MS (OMS) algorithm is employed in this brief. For the four formulas in Eq. (1), we learn the four offset parameters by the deep neural network to make the OMS achieve the performance of the exact form. The results are quantized with two fractional bits, which are 0, 0, 0.25, and 0.25, in turn.

Regarding the early termination (ET) strategy, if the CRC-based ET is applied, the CRC length is not long enough in UL, which will degrade the performance slightly. Also, in DL, an additional interleaving is required after each iteration for CRC detection. Hence, we employ the original $\mathbf{G}$-matrix based ET.

[1]The value of m is specified as 11 for most cases in UL (message length is larger than 19), and is constantly equals to 24 in DL.

Algorithm 1: GBPF Decoding

```
û ← BP(∅)// original BP decoding
if CRC_detect(û_A) succeeds then
    return û_A // interleaver is required for DL
else
    for t = 1 to T do
        û^(t) ← BP({S^(t)});
        if CRC_detect(û_A^(t)) succeeds then
            return û_A^(t);
    return û_A^(T);
```

III. Proposed BPF Decoding Algorithms

A. Generalized BPF Decoding

Bit-flipping was first applied to the SC decoding, proposed in [20] and improved by [21]. If the CRC is not satisfied after the original SC decoding, a flip set $\mathcal{S}$ consisting of T error-prone bits is generated, followed by additional T rounds of decoding. Inspired by [22], bit-flipping was introduced into the BP decoding [18], in which the flip set is composed of the first bit of each *rate*-1 node. In [18], each identified error-prone bit is flipped to both '0' and '1', inducing additional $2T$ attempts. However, this method does not reflect the concept of flipping, i.e., flipping the bit in one direction. In this section, a generalized BPF decoding algorithm is proposed with a different scheme to identify error-prone bits.

First, the bit-flipping strategy is redefined. For an identified bit i, it is flipped to $1 - \hat{u}_i$ based on the estimation of the first-round decoding. In specific, the flipping operation can be written by function $\mathtt{BP}(\mathcal{I})$ ($\mathcal{I}$ is the index set of flipped bits):

$$\forall i \in \mathcal{A},\ R_{0,i} = \begin{cases} +\infty, & i \in \mathcal{I} \text{ and } \hat{u}_i = 1, \\ -\infty, & i \in \mathcal{I} \text{ and } \hat{u}_i = 0, \\ 0, & i \in \mathcal{A}/\mathcal{I}. \end{cases} \tag{2}$$

The intuitive approach to establish the flip set $\mathcal{S}$ is to choose the indices at which the absolute LLR values are small, which is the same as [20]. In the SC decoding, the erroneous bits are caused by the channel noise or the forward propagation, and most of them exhibit a low absolute LLR value. Flipping an error bit caused by propagation does not work. However, in the BP decoding, it is probable to correct the error by flipping them due to the flooding-like bidirectional propagation. In this manner, the rule to generate the flip set is denoted by a function, $\mathtt{gen\_FS\_GBPF}(\boldsymbol{L}_0, T)$, defined by

$$\mathtt{gen\_FS\_GBPF}(\boldsymbol{L}, T) : \mathcal{S} \leftarrow i \in \mathcal{A} \text{ of } T \text{ smallest } |\boldsymbol{L}_i|.$$

Since the values of $R_{0,i}$ for information bits are all zero, $L_{0,i}$ denotes the likelihood of bit estimates. The procedure of the GBPF decoding is listed in Algorithm 1.

B. Enhanced BPF Decoding

The GBPF decoding suffers from a high sorting complexity, selecting the least reliable T bits from K' unfrozen bits. On the other hand, even though flipping the propagation-caused error bits can eliminate the mistake, identifying the noise-generated error bits can more effectively mitigate the effects of propagation and loop reinforcement. Thus, the GBPF decoding evolves into EBPF decoding by reducing the searching range.

According to the channel polarization, the quality of each bit channel is different. In this brief, we use the reliability

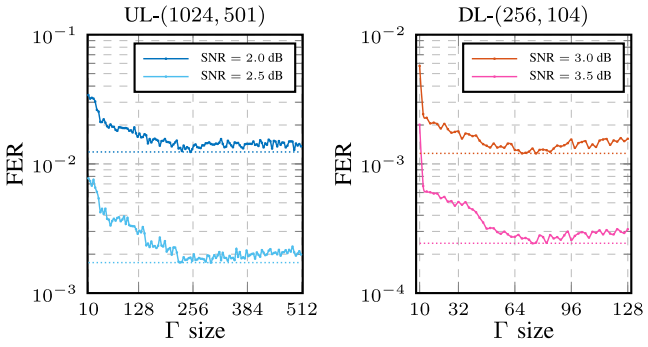

Fig. 2. FERs of the EBPF decoding with different Γ sizes.

sequence provided by 3GPP [23] to measure the bit channel quality. If a bit channel is reliable enough, even though the absolute LLR value over it is small, the estimate is considered correct. Such bit indices can be excluded from the search range, and the remaining indices constitute a new searching set Γ. The criterion of flip set generation follows:

$$\texttt{gen\_FS\_EBPF}(\boldsymbol{L}, T) : \mathcal{S} \leftarrow i \in \Gamma \text{ of } T \text{ smallest } |\boldsymbol{L}_i|.$$

IV. Error Correction Performance

In this section, the FER performance of the proposed algorithms is provided and compared with the BPF and SCL decoding. The maximum iteration number is set as $I_{\max} = 50$. The codeword is modulated by binary-phase shift keying (BPSK) and transmitted over additive white Gaussian noise (AWGN) channels. The definition of code rate follows 3GPP standard: $R = K'/E$.

A. Set Size for EBPF Decoding

Fig. 2 shows the effect of Γ size (γ) on FER performance. When γ equals to K', the EBPF is equivalent to the GBPF decoding. With a drop of the size, the FER gradually decreases and reaches an optimum at about $\gamma = K'/2$. However, the continued reduction of γ results in significant performance loss due to the elimination of less reliable bit channels.

Thus, the value of γ is set as $K'/2$, with which the set Γ corresponds to the least reliable $K'/2$ bit channels.

B. Performance With Bit-Flipping

Fig. 3 compares the FER performance between the BPF [18] and GBPF. Also, the curves of the BP and SCL decoding, and the bound of the GBPF are provided as references, where the bound is obtained by $T = K'$. The T values in the BPF are fixed, respectively equaling to 100, 119, and 123 for $R = \frac{1}{3}, \frac{1}{2}$, and $\frac{2}{3}$. Hence, the flip set sizes for the GBPF are assigned as 200, 238, and 246 accordingly to keep the additional attempt numbers the same. Shown in Fig. 3, the GBPF obtains a gain of about 0.1 dB compared with the BPF, close to the performance bound SCL-4. The performance gap results from the different flip sets. Even though the BPF attempts two flip directions for each bit so that it definitely hit the correct value, the set $\mathcal{S}$ is generated based on GA construction that is derived from the SC decoding, which is not as accurate for BP.

The comparison between the GBPF and the EBPF is presented in Fig. 4, where both UL-(1024, 501) and DL-(256, 104) codes are provided. It can be seen that the EBPF improves the performance by up to 20% while reducing the sorting range by half.

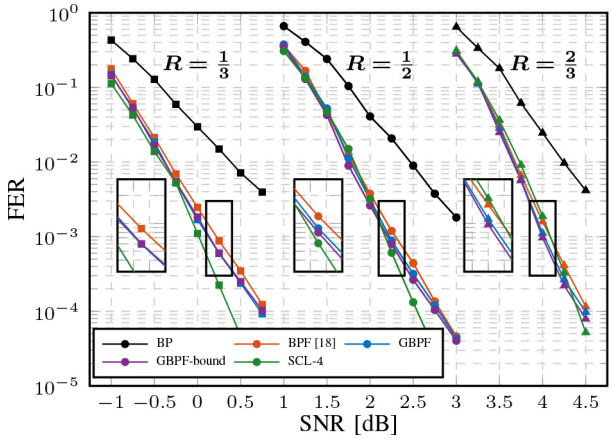

Fig. 3. FER performance comparison for UL polar codes of $E = 1024$.

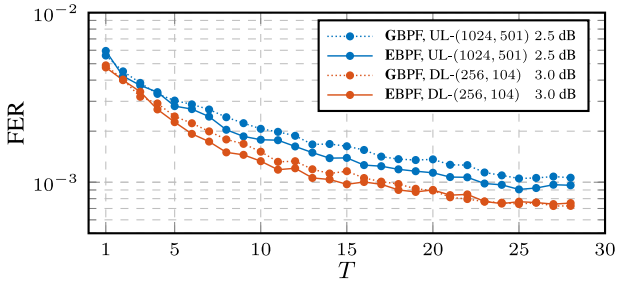

Fig. 4. FERs of the two proposed algorithms with different T values.

C. Comparison With SCL Decoding

Fig. 5 shows the performance comparison of the EBPF, SCL, and SC flip (SCF) decoding. Considering the requirements of eMBB scenarios, we focus on the target FER of 10^{-3}. For UL and DL cases, the EBPF with respectively $T = 10$ and 20 matches the SCL-2 at the target FER and outperforms the SCL-2 at lower SNRs. Since the 5G construction is not friendly to BP decoding (refer to Fig. 9 in [24]), the SCL-2 surpasses the EBPF below the target FER. Compared with the SCF at FER$= 10^{-3}$, the EBPF exhibits a 0.06 dB gain for UL-(1024, 501) codes but is inferior to the SCF for DL-(256, 104) codes.

The corresponding iteration numbers are recorded in the right sub-figure, representing complexity. At medium to high SNRs, the EBPF decoding achieves similar complexity to BP, requiring $4 \sim 8$ iterations at the target FER.

V. Hardware Implementation

Fig. 6 shows the overall architecture: memories unit, BP decoder, termination unit, flipping unit, and control unit.

A. Overall Architecture

We employ the single-column architecture [25] to implement the BP decoding, with which the L- and R- message propagations use one set of $N/2$ processing elements (PEs) and routing networks. The LLRs are quantized to $Q = 7$ bits, within which the right-most two bits represent the fractional part. Fig. 7 shows the EBPF performance with different quantization schemes, it can be seen that 6-bit strategy degrades the performance significantly, and 7-bit strategy is sufficient to keep exactly the same performance as the floating-point.

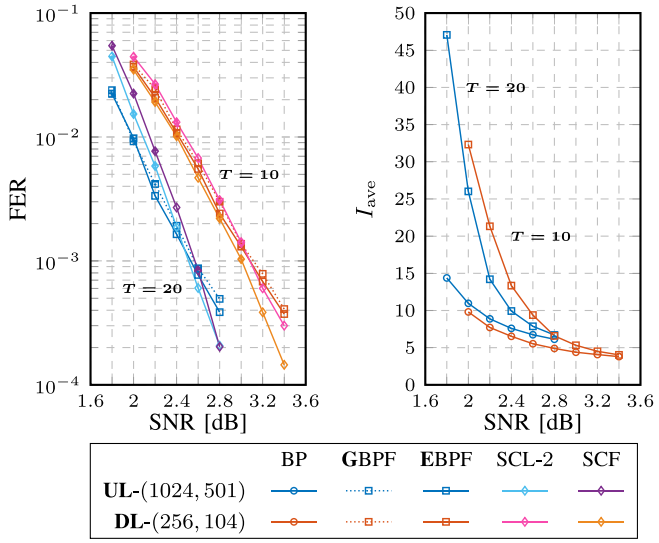

Fig. 5. **Left**: FER performance comparison among the EBPF, SCL, and SCF; **Right**: average iteration number comparison between the EBPF and BP.

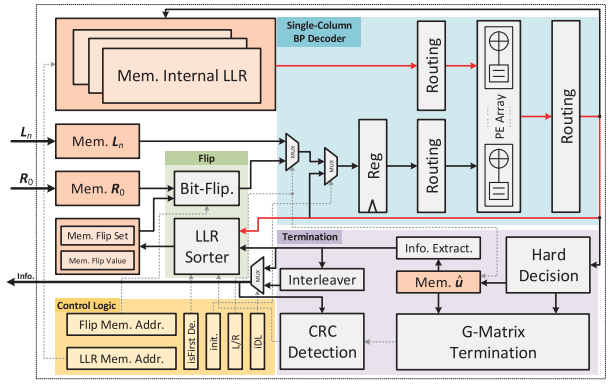

Fig. 6. Hardware architecture of GBPF decoder with solid line: data signal, dashed line: control signal, and red line: critical path (3.13 ns @Table I).

Each PE, with four inputs and two outputs, performs the OMS decoding, and the offset is switched between 0 and 0.25 according to L/R signal. The iteration begins with the L calculation and then the R-message propagation. Accordingly, one iteration consumes $2n$ clock cycles, within which the *intra*-BP **G**-matrix ET is operated at the last cycle.

CRC detection performs *inter*-BP termination. It terminates the additional decoding rounds and outputs the information sequence if the CRC is satisfied. The *Input Bits Interleaver* should be activated for DL channels. The sorter requires $\boldsymbol{L}_0$ directly from the BP decoder output and generates the flip set to be stored in the flipping memory with corresponding flipped values. Before each extra round, the flipped bit is read from the memory to modify the $\boldsymbol{R}_0$ data.

B. LLR Sorter

The LLR sorter selects the smallest T numbers from the K'-length absolute LLR array, using the bitonic merge sorting algorithm with a computational complexity of $\mathcal{O}(K' \log T)$. Define v as $\lceil \log_2 K' \rceil$, then a full sorting network has v super-layers. Each super-layer l includes l layers. Therefore, there are $v(v+1)/2$ layers and $2^{v-2} \cdot v \cdot (v+1)$ compare-and-select (CAS) units in a full sorting network. To avoid introducing a longer critical path, registers are inserted every four layers.

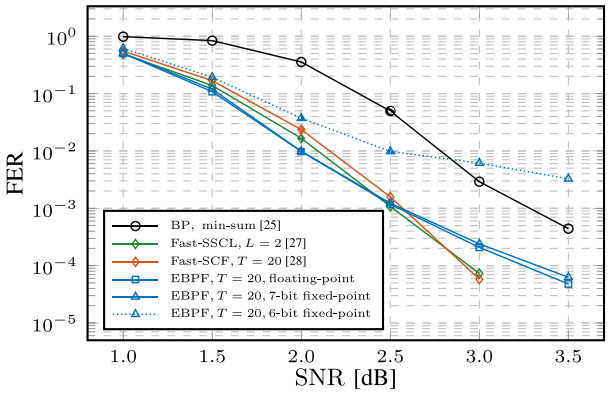

Fig. 7. FER performance comparison of different decoders for UL-(1024, 501) polar codes.

Since only T numbers need to be selected, most of CAS units at the last super-layer can be removed. For example, for the case of UL-(1024, 501) polar codes in Fig. 5, it needs 9740 CAS units for the GBPF decoder. However, the EBPF decoder requires only 3852 CAS units due to the half-reduced searching range. When $T = 20$, the two rightmost layers can be removed from the network. Therefore, GBPF and EBPF require 34 and 43 sorting layers, consuming 8 and 10 clock cycles, respectively.

C. Memory and Latency Analysis

For the single-column BP decoder, the required size of internal LLR memory is $(n-1) \times NQ$ bits. In addition to memory for input and output, the memory with $TQ + T$ bits is required for the bit-flipping. By contrast, for the SCL decoder, the number of memory bits for the internal LLRs is $L \times (N-1)Q$. Also, it requires additional memory for path metrics and partial sum.

The average number of clock cycles to perform one round of decoding is $2nI_{\text{ave}} + 1$, within which the last clock is for CRC detection. The high-parallel sorter is activated for $\mathcal{L}_{\text{sort}}$ clock cycles only after the first round with unsatisfied CRC. The average cycle number of decoding latency is related to the FER of the original BP decoding and equals to:

$$\mathcal{L}_{\text{ave}} = (2nI^0_{\text{ave}} + 1) + \text{FER}_{\text{BP}} \times (2nI^1_{\text{ave}} + 1 + \mathcal{L}_{\text{sort}}), \quad (3)$$

where I^0_{ave} and I^1_{ave} represent the average iteration number for the first round and additional rounds (maximum TI), respectively. Given the case in Fig. 5, at SNR $=$ 2.5 dB, the EBPF decoder requires almost 178.9 cycles to decode UL-(1024, 501) polar codes ($I^0_{\text{ave}} = 7.13$ and $I^1_{\text{ave}} = 193.3$). For the SCL decoder, the conventional architecture incurs a latency of $(3N-2)$ cycles [26].

D. Implementation Results

Table I presents the synthesis results using SMIC 65 nm technology. Owing to the LLR sorter, the GBPF decoder requires 2.18 times area consumption compared with our BP decoder. By reducing the sorting range, the area consumption drops by 26.8% for the EBPF. The average throughput of the EBPF decoder can achieve 1.8 Gbps and 3.7 Gbps at 2.5 dB and 4.0 dB, respectively.

Also, the state-of-the-art polar decoders are compared. The corresponding performances are shown in Fig. 7. Even though the BP decoders [25], [29] achieve the highest efficiency, they

TABLE I
65 NM CMOS SYNTHESIS RESULTS FOR PROPOSED GBPF AND EBPF DECODERS AGAINST THE STATE-OF-THE-ART BP, SCL, AND SCF DECODERS WITH $N = 1024$ AND $R = 1/2$

Decoders	**BP**	**GBPF**	**EBPF**	**BP** [25] [SVLSI'14]	**BP** [29] [TVLSI'17]	**Fast-SSCL** [27] [TSP'17]	**Fast-SCF** [28] [TCAS-I'20]
Quantization	7-bit	7-bit	7-bit	5-bit	5-bit	6-bit	6-bit
Parameter	—	$T = 20$	$T = 20$	—	—	$L = 2$	$T = 20$
Design	single-column OMS			double-column MS	CSFG-based BP [1] MS	semi-parallel $P_e = 64$	semi-parallel $P_e = 64$
Area [mm^2]	2.07	4.25	3.11	1.48	1.60	1.05	0.56
Max. Freq. [MHz]	373	319	319	300	334	885	—
Coded T/P @2.5 dB [Gbps]	2.66	1.81	**1.83**	—	—	1.86	1.40
Coded T/P @4.0 dB [Gbps]	4.36	3.72	**3.72**	4.67	10.70	1.86	1.51
Efficiency @4.0 dB [Gbps/mm^2]	2.11	0.88	1.20	3.16	6.69	1.78	2.71

[1] CSFG: Connected Subfactor Graph. In [29], the performance is analyzed with the normalized MS, and the 5-bit quantized decoder employs the MS.

suffer from a 0.75 dB performance loss at FER$= 10^{-3}$ compared with the EBPF decoder. The decoders in [27] and [28] process special constituent nodes directly, which are named as Fast-SSCL and Fast-SCF, respectively. They and EBPF have a similar performance and throughput at 2.5 dB, and the EBPF exhibits twice their throughput at 4.0 dB. However, the large area leads to the unsatisfactory efficiency of the EBPF. In [27] and [28], the number of PEs, P_e, is only 64. By contrast, our decoder requires 512 PEs. In addition, the depth of the internal LLR memory is 9, which is far larger than the Fast-SSCL requires.

VI. CONCLUSION

In this brief, two GBPF and EBPF algorithms are proposed. Future work will concentrate on multi-bit flipping and high-efficiency implementations.

REFERENCES

[1] *Chairmans Notes of Agenda Item 7.1.5 Channel Coding and Modulation, Ad-Hoc Chair*, document TSG RAN WG1 meeting #87, R1–1613710, 3GPP, Sophia Antipolis, France, Nov. 2016.

[2] 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, Jul. 2009.

[3] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[4] *Final Report of 3GPP TSG RAN WG1 #AH1_NR V1.0.0*, document TSG RAN WG1 meeting #88, R1–1701553, 3GPP, Sophia Antipolis, France, Feb. 2017.

[5] E. Arıkan, "Polar codes: A pipelined implementation," in *Proc. Int. Symp. Broad. Commun. (ISBC)*, 2010, pp. 11–14.

[6] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2009, pp. 1488–1492.

[7] N. Yang *et al.*, "Reconfigurable decoder for LDPC and polar codes," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2018, pp. 1–5.

[8] A. Elkelesh, S. Cammerer, M. Ebada, and S. T. Brink, "Mitigating clipping effects on error floors under belief propagation decoding of polar codes," in *Proc. IEEE Int. Symp. Wireless Commun. Syst. (ISWCS)*, 2017, pp. 384–389.

[9] A. Eslami and H. Pishro-Nik, "On bit error rate performance of polar codes in finite regime," in *Proc. IEEE Annu. Allerton Conf. Commun. Control Comput. (Allerton)*, 2010, pp. 188–194.

[10] A. Ç. Arlı and O. Gazi, "Noise-aided belief propagation list decoding of polar codes," *IEEE Commun. Lett.*, vol. 23, no. 8, pp. 1285–1288, Aug. 2019.

[11] B. Yuan and K. K. Parhi, "Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders," *IEEE Trans. Signal Process.*, vol. 62, no. 24, pp. 6496–6506, Dec. 2014.

[12] W. Xu, Z. Wu, Y.-L. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, 2017, pp. 1–6.

[13] B. Dai, R. Liu, and Z. Yan, "New min-sum decoders based on deep learning for polar codes," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, 2018, pp. 252–257.

[14] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief propagation decoding of polar codes on permuted factor graphs," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2018, pp. 1–6.

[15] N. Doan, S. A. Hashemi, M. Mondelli, and W. J. Gross, "On the decoding of polar codes on permuted factor graphs," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2018, pp. 1–6.

[16] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Belief propagation list decoding of polar codes," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1536–1539, Aug. 2018.

[17] S. Sun, S.-G. Cho, and Z. Zhang, "Post-processing methods for improving coding gain in belief propagation decoding of polar codes," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2017, pp. 1–6.

[18] Y. Yu, Z. Pan, N. Liu, and X. You, "Belief propagation bit-flip decoder for polar codes," *IEEE Access*, vol. 7, pp. 10937–10946, 2019.

[19] V. Bioglio, C. Condo, and I. Land, "Design of polar codes in 5G new radio," 2018. [Online]. Available: arXiv:1804.04389.

[20] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, "A low-complexity improved successive cancellation decoder for polar codes," in *Proc. IEEE Asilomar Conf. Signal Syst. Comput. (Asilomar)*, 2014, pp. 2116–2120.

[21] C. Condo, F. Ercan, and W. J. Gross, "Improved successive cancellation flip decoding of polar codes based on error distribution," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, 2018, pp. 19–24.

[22] Z. Zhang, K. Qin, L. Zhang, H. Zhang, and G. T. Chen, "Progressive bit-flipping decoding of polar codes over layered critical sets," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2017, pp. 1–6.

[23] *5G NR: Multiplexing and Channel Coding, Version 15.2.0*, 3GPP, Sophia Antipolis, France, Rep. 38.212, Jul. 2018.

[24] A. Elkelesh, M. Ebada, S. Cammerer, and S. T. Brink, "Decoder-tailored polar code design using the genetic algorithm," *IEEE Trans. Commun.*, vol. 67, no. 7, pp. 4521–4534, Jul. 2019.

[25] Y. S. Park, Y. Tao, S. Sun, and Z. Zhang, "A 4.68 Gb/s belief propagation polar decoder with bit-splitting register file," in *IEEE Symp. VLSI Circuit Dig. Tech. (VLSI)*, 2014, pp. 1–2.

[26] A. Balatsoukas-Stimming, M. Bastani Parizi, and A. Burg, "LLR-based successive cancellation list decoding of polar codes," *IEEE Trans. Signal Process.*, vol. 63, no. 19, pp. 5165–5179, Oct. 2015.

[27] S. A. Hashemi, C. Condo, and W. J. Gross, "Fast and flexible successive-cancellation list decoders for polar codes," *IEEE Trans. Signal Process.*, vol. 65, no. 21, pp. 5756–5769, Nov. 2017.

[28] F. Ercan, T. Tonnellier, and W. J. Gross, "Energy-efficient hardware architectures for fast polar decoders," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 1, pp. 322–335, Jan. 2020.

[29] S. M. Abbas, Y. Fan, J. Chen, and C. Tsui, "High-throughput and energy-efficient belief propagation polar code decoder," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 3, pp. 1098–1111, Mar. 2017.