An implementation of fast polar codes decoder with reducing internal memory and supporting flexible code rate

Ping Luo\(^1, 2, a\), Wu Guan\(^3\), Liping Liang\(^3\), and Xin Qiu\(^1\)

Abstract This letter proposes a fast simplified successive-cancellation (FSSC) polar decoder architecture, supporting any code rate. With the parameter \(M\), which is the maximum limit length of a special polar node, the authors present a novel scheme for online identification of special node in a polar code. In addition, under the parameter \(M\), the proposed decoder has a well-optimized architecture to reduce area, power and energy consumption, that due to require less internal memory using cross-layer calculation and less hardware resources for special node without pipeline technology. Synthesis and post-layout simulate results, based in TSMC 65nm CMOS technology, show that the consumption of hardware resources is reduced by 25%. The architecture and circuit techniques reduce the power to 54.9mW for an energy efficiency of 77.22 pJ/b.

Keywords: polar codes, fast simplified successive-cancellation (FSSC), online calculation, cross-layer calculation, energy efficient

Classification: Integrated circuits

1. Introduction

Polar codes have drew signification attention due to the fact that they have low complexity successive-cancellation (SC) decoding algorithm and achieve the capacity of binary input memory-less symmetric-channels (BMC) [1]. However, the serial nature of SC tends to high latency and low throughput when implemented on hardware [2, 3, 4, 5]. In order to address this issue, simplified SC (SSC), Fast-SSC, SSC with maximum-likelihood decoding (ML-SSC) and belief propagation (BP) algorithm were proposed in [6, 7, 8, 9, 10, 11, 12, 13, 14]. To improve the error-correction performance of polar codes with the short and moderate code length, SC-list (SCL), Cyclic redundancy check aided successive cancellation list (CA-SCL) and BP-list (BPL) were proposed to improve the error-correction performance shown in [15, 16, 17, 18, 19, 20, 21, 22, 23].

In order to reduce the number of nodes traversing the polar decoding tree, [7, 8] proposed four special nodes to improve the throughput, collectively referred to as Fast-SSC decoder. The hardware of Fast-SSC decoder rely on the identification of type and length of special nodes in a polar code. Therefore, the identification of special nodes is processed off-line by defining a specific instruction set and the instructions are stored in a specific memory [8, 24]. The instruction dependent on the rate of polar code and when the rate changes, the instructions has to be regenerated. Although [25, 26] realize online calculation of flexible code rate, but it requires complex combinatorial logic and flexible memory design for various node lengths. In addition, when the length of special node is large, the circuit has a large hardware implementation complexity using pipeline technology for single parity check (SPC) and repeat (Rep) nodes [8, 27]. In [28], Fast-SSC decoder reduced memory by storing the node variables related to the lower stages of the decoding tree into a single memory word. In [24], the decoder reduced memory of partial sum bits and input memory by removing the input buffer. Although [28] and [24] reduce the memory of decoder, the memory can still be optimized.

This letter proposes a new polar decoder called M-fast-SSC with a limit parameter \(M\) which is the maximum limit length of special node. Under the parameter \(M\), a new scheme for the on-line identification of special polar nodes is presented. In the SC-based polar decoding, the delay of process element (PE) operation is small, and there is only one subtractor and three multiple selectors. Based on this fact, cross-layer calculation strategy is presented to reduce the internal memory of the decoder. Overall M-Fast-SSC polar decoder has been implemented and synthesized in TSMC 65nm CMOS technology-node. The proposed decoder provides a decoding throughput of 1.53 Gbps at 400MHz with the worst-case 1.08 volts and 125°C. The results show that the proposed M-fast-SSC decoder architecture is 25% more energy-efficient and 10% more area efficiency, Compared with the Fast-SSC decoder as shown in [24].

The remainder of the letter is organized as follows. Section 2 briefly reviews the principle of the polar codes. Section 3 proposes on-line identification and cross-layer calculation architectures. Section 4 shows the design architecture and decoding merging strategy of the entire decoder. Section 5 introduces the implementation and comparison of decoders. Finally, Section 6 concludes the letter.

2. Review of polar codes

For a polar code \(PC(N, K)\), the channel is split into \(N\) channels, and the \(K\) channels with the highest reliability are selected to transmit information bits represented by set \(\mathcal{I}\), and \((N-K)\) channels are used to transmit frozen bits represented...
by set \( \mathcal{C} \), usually to zero. Let \( u_{0:N-1} = \{x_0, x_1, \ldots, x_{N-1}\} \) and \( u_{0:N-1} = \{u_0, u_1, \ldots, u_{N-1}\} \) denote the encoded and input vectors respectively, and polar codes are encoded using matrix multiplication \( x_{0:N-1} = u_{0:N-1} G_{n \times n} \), where the \( G_{n \times n} \) is generator matrix calculated by the \( n \)th Kronecker power of the matrix \( F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \).

Let \( y \) denote the received noisy-codeword vectors corresponding to transmitted polar-encoded bits \( x_{0:N-1} \) and SC decoding estimates bits sequentially, starting from \( \hat{u}_0 \). The estimated value of \( \hat{u}_i \) depends on the vector \( y \) and previously decoded bits \( \hat{u}_{0:i-1} \). If \( i \in \mathcal{C} \text{ or } P_{\text{pry-noisy}}(\hat{u}_{0:i-1}|u_{0:N-1}) < 1 \), \( \hat{u}_i = 0 \). Otherwise, \( \hat{u}_i = 1 \). An illustration of the SC decoder is shown in Fig. 1 for \( PC(16,8) \), modified from [24]. The number of layers for the decoding tree is represented by the variable \( S \) from bottom to top, numbered from 0 to \( n \). For a parent node \( v \) that contains LLR values \( a_{0:2^S-1} = \{a_0', a_1', \ldots, a_{2^S-1}'\} \), the LLR values, which pass to left child nodes \( a_{0:2^S-1} = \{a_0', a_1', \ldots, a_{2^S-1}'\} \), are called \( F \) function as Eq. (1). The LLR values, that passed to right child nodes \( a_{0:2^S-1} = \{a_0', a_1', \ldots, a_{2^S-1}'\} \), is called \( G \) function calculated by Eq. (2). The partial sums \( \beta' \) for a parent node \( v \) are calculated as Eq. (3) with the left child node \( \beta' = \{\beta_0', \beta_1', \ldots, \beta_{2^S-1}'\} \) and the right child node \( \beta'' = \{\beta_0'', \beta_1'', \ldots, \beta_{2^S-1}''\} \).

\[
\begin{align*}
\alpha_i' & = \text{sgn}(a_i')\text{sgn}(a_{i+2^S-1}')\min(|a_i'|, |a_{i+2^S-1}'|), \\
\alpha_i'' & = a_{i+2^S-1}' + (1 - 2\beta_i')\alpha_i', \\
\beta_i' & = \begin{cases} 
\beta_i'' & \text{if } i \leq 2^S-1, \\
\beta_{i-2^S} & \text{otherwise},
\end{cases}
\end{align*}
\]

In the decoding tree shown in Fig. 2, Fast-SSC can reduce the number of traversed nodes of the decoding tree by using four special nodes. SSC algorithm proposed \( Rate-0 \) and \( Rate-1 \) nodes whose leaf nodes are all frozen bits or all information bits shown in [7]. For a parent node of \( Rate-0 \) or \( Rate-1 \), the decoding tree do not have to traverse its child nodes, and the nodes can be calculated at the parent node where partial sum bits are all zeros for \( Rate-0 \) node or can be made using hard decisions for \( Rate-1 \) nodes. Besides, single-parity check (SPC) and repetition (Rep) were proposed in [8]. A Rep node contains a single information bit and all other bits are frozen. Then, the decision is made by adding the LLR values together and extracting the sign bit of the result. The partial sum bits for \( Rep \) node are the same as the sign bit. For a parent node of SPC, The partial sum bits are obtained by using hard decisions on \( a_{0:2^S-1} \). Consequently, the parity check for the partial sum bits must be zero. If the parity constraint is satisfied, the decoding of the SPC node is assumed successful. If the parity is not satisfied, it means that there is at least one error. To satisfy the parity check constraint, the bit with the least reliable \( a_i' \) is found and flipped.

3. Proposed on-line identification and cross-layer calculation architectures

3.1 Limiting the length of special nodes
A polar code \( PC(N,K) \) where \( N, K \) are the length of code and information bits respectively, contains one or more types of special nodes among \( Rate-0 \), \( Rate-1 \), (SPC), and (Rep) nodes showed in [7, 8]. In hardware implementation, the variety of special nodes length and type significantly increase complexity of circuit on control and multiple selection logic. Especially for large-length special nodes, like SPC and Rep nodes require complex circuit design using pipeline technology, presented in [8, 27].

This letter proposes a new polar decoder called M-fast-SSC decoder with a limit parameter \( M = 2^S \) and \( q = 3 \) for (a) \( Rate-0 \), (b) \( Rate-1 \), (c) SPC, (d) Rep nodes.
power of two. Therefore, we define limit parameter \( M = 2^q \), as \( q \geq 2 \). Under the parameter \( M \), a special node with length \( L = 2^p \), as \( p > q \), can be divided into \((2^{p-q})\) special nodes of length \( M \), where the special nodes all belong to the set of \{rate-0, rate-1, SPC, Rep\}. The division method of these four special nodes is shown in Fig. 3. Rate-0 and Rate-1 nodes with length of \( L = 2^p \), as \( p > q \), can be divided into \((2^{p-q})\) Rate-0 and Rate-1 nodes with length of \( M \), respectively. A SPC node with length of \( L = 2^p \), as \( p > q \), can be divided into a SPC node and \((2^{p-q} - 1)\) Rate-1 nodes with length of \( M \). A Rep node with length of \( L = 2^p \), as \( p > q \), can be divided into \((2^{p-q} - 1)\) Rate-0 nodes and a Rep node with length of \( M \).

### 3.2 Identification of special nodes under parameter \( M \)

For a polar code \( PC(N, K) \), we define a corresponding frozen-bits pattern with a bit vector of length \( N \), like \( \text{FP} = [000101...11111] \), where ‘0’ and ‘1’ indicate the position of frozen-bits and information-bits, respectively. The four special nodes mentioned above in a polar code can be uniquely represented by frozen-bits pattern. For instance, rate-0, rate-1, SPC and Rep nodes with length of four can be indicated by \([0000], [1111], [0111], [0001]\), respectively.

The frozen-bits pattern with length of \( M \) indicates the type and length of a special node, but it is not always a special node with a length of \( M \), and there may be special nodes with a length less than \( M \) in the frozen-bits pattern. Therefore, we try to search for the special node with largest length under the limit parameter \( M \). Let \( \{ R_{0:3,y} \} \) denote the set that contains four frozen-bits patterns of Rate-0, Rate-1, SPC, Rep nodes with length of \( y \). The proposed scheme first searches and judges whether the bit vector of \( \text{FP}_{0:M-1} \) is the frozen-bits pattern of current node with length \( M \), belongs to the set \( \{ R_{0:3,y} \} \). If \( \text{FP}_{0:M-1} \) belongs to \( \{ R_{0:3,M} \} \), the current node is successfully identified as a special node, if not, then recursively search and determine whether the \( \text{FP}_{0:M/2-1} \) and \( \text{FP}_{M/2:M-1} \), these are obtained by dividing the original vector \( \text{FP}_{0:M-1} \) into two left and right vectors with length \( M/2 \), belong to the set \( \{ R_{0:3,M/2} \} \). The above recursive search and judgment process is repeated until all the split nodes are judged to be special nodes or the node length is reduced to four. ML decoding is used to end the current node decoding.

### 3.3 Implementation of on-line identification

In the decoding under the special node limit length parameter \( M \), we propose a circuit implementation scheme based on a finite state machine (FSM). As described in section 3.2, the FSM circuit needs to search and identify the longest possible special node, and it simultaneously needs to be able to recursively search and identify special node until the node length reaches four. Therefore, the state for the special node with length of \( M \) is set the highest priority, the state for the special node with length of \( M/2 \) is set the next highest priority, and the state of ML is set the lowest priority. In the actual parallel identification of the circuit, the possible frozen-bit pattern can be identified as a special node of various lengths. At the time, FSM selects the state with highest priority to jump and identified. Fig. 4 depicts the logic jump diagram of the finite state machine for limit parameter \( M = 16 \). It should be noted that there is actually only one IDLE state, and the Fig. 4 is just to make the figure clearer. Table I shows the jump control logic of state diagram. S16 represents the identification state of a special node of length 16, and S8_0 and S8_1 represent the identification state of a special node of length 8 for left and right nodes, respectively. For the special node state definition with a node length of four, such as S4_0, S4_1, S4_2, S4_3, these are consistent with the above description.

In our design, we trade off hardware resources, decoding throughput and identification complexity, therefore, \( M = 32 \) is chosen as the limit parameter. The identification circuit based on FSM is implemented and synthesized on TSMC 65 nm CMOS technology using the general purpose (GP) standard cell library with the worst PVT values of 1.08 V and 125°C. The synthesis results show the area of the identification circuit is only 457 \( \mu \text{m}^2 \) and 565 \( \mu \text{m}^2 \) under the clock of 0.5 GHz and 1 GHz, respectively. In addition, the identification circuit is not a critical timing path in the design.

### 3.4 Cross-layer strategy based on memory rearrangement

In order to reduce hardware area, this paper proposed cross-layer calculation scheme to reduce the size of internal LLRs memory. Cross-layer calculation for LLRs requires multi-level serial (PEs) structure and memory rearrangement. The PE units use the low-latency architecture proposed in [5]. Fig. 5 shows the compression of four LLRs to one LLR by using three PE units across the one-layer calculation method. The critical path delay of PEs architecture is the sum of two subtractors and six 2-to-1 multiplexers. The calculation method decreases the size of internal LLRs memory, but
increase combinational logic delay with two PE units connected in series. Fig. 6 shows the internal memory structure that requires to be rearranged using across one-layer calculation scheme. This calculation scheme directly calculates from the 8th layer to the 6th layer, and does not need to store the value of the LLRs of the 7th layer.

In order to balance the delay between registers in the entire design under limit parameter $M$, the cross-layer calculation just adopts one-cross-layer scheme. In cross-layer calculation case, it is reasonable to choose $(32+16)$ PE units. Table II shows the comparison results between non-cross-layer calculation and cross-layer calculation scheme for size of internal memory with polar code length $N = 1024$. The results show that the area of internal memory for the cross-layer calculation scheme is 67% smaller than that of the non-cross-layer.

4. Design of overall decoder architecture

4.1 Architecture overview

In this letter, we implemented M-Fast-SSC polar-decoder using the proposed FSM identification architecture under the limit parameter $M = 32$, shown in Fig. 7, supporting $N = 1024$ bits with any code rate. The A-Router part is the $(32+16)$ PE units used to implement Eq. (1) and Eq. (2). The B-Router part is used to calculate the partial sum bits as Eq. (3). After the channel LLRs and the corresponding frozen bit pattern are loaded, the circuit starts to decode and estimated bits are stored in the output memory of code-words.

4.2 Architecture of processor

The process block manipulates the special nodes, such as $SPC$, $Rep$ and maximum likelihood decoding ($MLD$), that is similar to the literature [8, 27]. Although the implementation of special node computing processor is similar, the proposed process is implemented without pipeline technology and reduces the hardware complexity under the limited parameter $M = 32$. An adder tree composed of $M - 1$ adders with $\log_2 M$ levels of depth calculates the $Rep$ nodes with the maximum length of $M$. A compare-select (CS) tree composed of $M - 1$ comparators with $\log_2 M$ levels of depth finds the index of the least reliable bit for the $SPC$ nodes with the maximum length of $M$. Under the parameter $M = 32$, the calculation of the longest special node for $SPC$ and $Rep$ can be completed in one clock cycle.

4.3 Strategy for reducing the steps of decoding

For a polar code sequence, the frozen bits are always more concentrated at the beginning of decoding sequence. Therefore, the jumping head strategy is used to reduce the number of decoding cycles. For example, the decoding process can start from the first information bit with the index of 127 for a polar code $PC(1024, 512)$. In addition, merging schemes for $Rate-0$ and $Rate-1$ nodes can improve the throughput further. The calculation of LLRs, partial sum bits and processor for $Rate-0$ nodes can be skipped to reduce the number of decoding clock cycles as shown in Fig. 8. The partial sum bits can be directly calculated by bypassing the sign bit of the LLRs value for $Rate-1$ nodes as shown in Fig. 9.
Table III  Comparison of the polar decoder for $PC(1024, 512)$. 

| Algorithm | Encoding | Technology (nm) | Supply (V) | PE | Quantization ($Q_1, Q_c, Q_f$) | Power (mW) | Area (mm$^2$) | Frequency (MHz) | Power Density (mW/mm$^2$) | Latency (µs) | Coded T/P (Mbps) | Information T/P (Mbps) | Area Efficiency (Gps/mm$^2$) | Energy (pJ/impo.bit) |
|-----------|----------|----------------|------------|----|-------------------------------|-----------|--------------|---------------|------------------------|-------------|-----------------|-------------------|--------------------------|-------------------|
| Our work  | Fast-SSC | Non-systematic  | 65          | 32+16 | (6,5,1) | 54.90     | 0.31          | 400            | 177.10             | 0.72        | 1526            | 711               | 4,59                | 77.22              |
| [24]-2020 | Fast-SSC | Non-systematic  | 65          | 64  | (6,5,1) | 79.90     | 0.38          | --             | 210.26             | 0.66        | 1557            | 778               | 4.10                | 102.65             |
| [29]-2019 | Fast-SSC | systematic      | 65          | 64  | (6,5,1) | 189.09    | 0.64          | 430            | 295.45             | 0.46        | 2213            | 1107              | 3.46                | 169.89             |
| [30]-2018 | Fast-SSC | systematic      | 65          | 64  | (6,5,1) | 114.00    | 0.44          | 450            | 295.09             | 0.56        | 1829            | 915               | 4.16                | 124.69             |
| [28]-2017 | Fast-SSC | systematic      | 65          | 64  | (6,5,1) | 173.75    | 0.57          | 420            | 259.09             | 0.51        | 2000            | 1000              | 3.51                | 173.07             |
| [8]-2014  | Fast-SSC | systematic      | 65          | 64  | (6,5,1) | 160.78    | 0.60          | 450            | 304.82             | 0.60        | 1719            | 860               | 2.87                | 188.41             |

Fig. 9  The forward calculation strategy for Rate-1 nodes reduces the number of clock cycles.

5. Implementation results and comparison

The proposed polar decoders under the limit parameter $M = 32$ have been implemented in verilog code, and synthesized in Synopsys Design-Compiler tool using TSMC 65 nm CMOS technology node with the worst PVT values of 1.08V and 125℃. The netlist with 0.224 mm$^2$ area has been imported in IC Compiler tool for the physical design using ten metal-layer. The final chip layout of our decoders occupies 0.31 mm$^2$ of core area and the power of decoders is analyzed using Prime Time (PT) tool.

Table III shows the synthesis results for the proposed Fast-SSC decoder under the limit parameter $M$ against state-of-the-art Fast-SSC decoders, for $PC(1024, 512)$. All decoders have the same quantization scheme, voltage and technology process. As for the parallel calculation of PE, our decoder use (32+16) PE units for cross-layer calculation, but others all use 64 units. The proposed decoder has 31% less power and 18% less area compared to the decoders with next best values shown in [24]. This is mainly due to the reduction of internal memory using cross-layer calculation technology and less hardware resources for special node calculation without pipeline technology. In addition, smaller on-line recognition logic and fewer PE units also reduce the area of decoder. The proposed decoder has the highest area and energy efficiency, higher than [8, 24, 28, 29, 30]. Although the proposed scheme to limit the length of special nodes increases the number of clock cycles for decoding one frame of data, the skipping strategy and the merge operation reduce the clock cycles as described in Section 4.3. Therefore, the throughput has not significantly decreased compared with [24].

6. Conclusion

In this letter, a new energy-efficient non-systematic polar Fast-SSC decoder under the limit parameter $M$ is designed in 65nm CMOS technology. A novel scheme for on-line identification of special nodes is presented and implemented in hardware. With the proposed cross-layer calculation scheme, optimized internal memory of polar decoder is developed. Synthesis results show that the proposed architecture has significant advantages with respect to hardware reduction. Since the proposed cross-layer scheme is a general solution that optimizes internal memory, it can also be applied to any other type of SC-based polar decoders.

Acknowledgments

The authors would like to thank the editors and anonymous reviewers for their helpful and constructive comments. Meanwhile, this work was supported in part by the China National Key Research Program under Grant 2018YFB2201502.

References

[1] E. Arikan: “Channel polarization: a method for constructing capacity achieving codes for symmetric binary-input memoryless channels,” IEEE Trans. Inf. Theory 55 (2009) 3051 (DOI: 10.1109/TIT.2009.2021379).
[2] C. Leroux, et al.: “A semi-parallel successive-cancellation decoder for polar codes,” IEEE Trans. Signal Process. 61 (2013) 289 (DOI: 10.1109/TSP.2012.2223693).
[3] A.J. Raymond and W.J. Gross: “A scalable successive-cancellation decoder for polar codes,” IEEE Trans. Signal Process. 62 (2014) 5339 (DOI: 10.1109/TSP.2014.2347262).
[4] B. Yuan and K.K. Parhi: “Low-latency successive-cancellation polar decoder architectures using 2-bit decoding,” IEEE Trans. Circuits Syst. I, Reg. Papers 61 (2014) 1241 (DOI: 10.1109/TCSII.2013.2283779).

[5] R. Shrestha and A. Sahoo: “High-speed and hardware-efficient successive cancellation polar decoder,” IEEE Trans. Circuits Syst. II, Exp. Briefs 66 (2019) 1144 (DOI: 10.1109/TCSII.2018.2877140).

[6] C. Zhag and K.K. Parhi: “Latency analysis and architecture design of simplified SC polar decoders,” IEEE Trans. Circuits Syst. II, Exp. Briefs 61 (2014) 115 (DOI: 10.1109/TCSII.2013.2291065).

[7] A. Alamdar-Yazdi and F.R. Kschischang: “A simplified successive-cancellation decoder for polar codes,” IEEE Commun. Lett. 15 (2011) 1378 (DOI: 10.1109/LCOMM.2011.101811.111480).

[8] G. Sarkis, et al.: “Fast polar decoders: algorithm and implementation,” IEEE J. Sel. Areas Commun. 32 (2014) 946 (DOI: 10.1109/JSAC.2014.140514).

[9] Y. Ren, et al.: “Efficient belief propagation list decoding of polar codes,” 2019 IEEE 13th International Conference on ASIC (ASI-CON) (2019) 1 (DOI: 10.1109/ASICON47005.2019.8983597).

[10] S.M. Abbas, et al.: “High-throughput and energy-efficient belief propagation polar code decoder,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 25 (2017) 1098 (DOI: 10.1109/TVLSI.2016.2620998).

[11] Y. Yan, et al.: “An implementation of belief propagation decoder with combinational logic reduced for polar codes,” IEICE Electron. Express 16 (2019) 15 (DOI: 10.1587/elex.16.20190382).

[12] H. Ji, et al.: “Hardware implementation for belief propagation flip decoding of polar codes,” IEEE Trans. Circuits Syst. I, Reg. Papers 68 (2021) 1330 (DOI: 10.1109/TCSI.2020.3042597).

[13] J. Liu and J. Sha: “Frozen bits selection for polar codes based on simulation and BP decoding,” IEICE Electron. Express 14 (2017) 20170026 (DOI: 10.1587/elec.14.20170026).

[14] Y. Shen, et al.: “Enhanced belief decoder framework for 5G polar codes with bit-flipping,” IEEE Trans. Circuits Syst. II, Exp. Briefs 67 (2020) 901 (DOI: 10.1109/TCSII.2020.2984536).

[15] L. Xiang, et al.: “Soft list decoding of polar codes,” IEEE Trans. Veh. Technol. 69 (2020) 13921 (DOI: 10.1109/TVT.2020.3021258).

[16] K. Wang, et al.: “Optimized sorting network for successive cancellation list decoding of polar codes,” IEICE Electron. Express 14 (2017) 18 (DOI: 10.1587/elec.14.20170735).

[17] I. Tal and A. Vardy: “List decoding of polar codes,” IEEE Trans. Inf. Theory 61 (2015) 2213 (DOI: 10.1109/TIT.2015.2410251).

[18] K. Niu and K. Chen: “CRC-aided decoding of polar codes,” IEEE Commun. Lett. 16 (2012) 1668 (DOI: 10.1109/LCOMM.2012.090312.121501).

[19] K. Niu and K. Chen: “Stack decoding of polar codes,” Electron. Lett. 48 (2012) 695 (DOI: 10.1049/el.2012.1459).

[20] Y. Yu, et al.: “Belief propagation bit-flip decoder for polar codes,” IEEE Access 7 (2019) 10937 (DOI: 10.1109/ACCESS.2019.2891951).

[21] L. Dai, et al.: “Architecture optimizations for BP polar decoders,” IEEE 3rd International Conference on Electronic Information and Communication Technology (ICEICT) (2020) 445 (DOI: 10.1109/ICEICT51264.2020.9334270).

[22] Y. Yang, et al.: “Noise-aided belief propagation list bit-flip decoder for polar codes,” International Conference on Wireless Communications and Signal Processing (WCSP) (2020) 807 (DOI: 10.1109/WCSP49889.2020.9299849).

[23] A. Elkelesh, et al.: “Belief propagation list decoding of polar codes,” IEEE Commun. Lett. 22 (2018) 1536 (DOI: 10.1109/LCOMM.2018.2850772).

[24] F. Ercan, et al.: “Energy-efficient hardware architectures for fast polar decoders,” IEEE Trans. Circuits Syst. I, Reg. Papers 67 (2020) 322 (DOI: 10.1109/TCSI.2019.2942833).

[25] S.A. Hashemi, et al.: “Rate-flexible fast polar decoders,” IEEE Trans. Signal Process. 67 (2019) 5689 (DOI: 10.1109/TSP.2019.2944738).

[26] S.A. Hashemi, et al.: “Rate-flexible fast polar decoders,” 2019 IEEE Information Theory Workshop (ITW) (2019) 1 (DOI: 10.1109/ITW44776.2019.8899275).

[27] Y. Zhou, et al.: “High-speed and hardware-efficient successive cancellation polar decoder,” IEEE Trans. Circuits Syst. II, Exp. Briefs 66 (2019) 227 (DOI: 10.1109/TCSII.2018.2877140).

[28] F. Ercan, et al.: “Reduced-memory high-throughput fast-ssc polar code decoder architecture,” Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS) (2017) 1 (DOI: 10.1109/SiPS.2017.8110014).

[29] F. Ercan, et al.: “Operation merging for hardware implementations of fast polar decoders,” J. Signal Process. Syst. 91 (2019) 995 (DOI: 10.1007/s11265-018-1413-4).

[30] G. Pascal, et al.: “Fast low-complexity decoders for low-rate polar codes,” J. Signal Process. Syst. 90 (2018) 675 (DOI: 10.1007/s11265-016-1173-y).