Design and Implementation of a Polar Codes Blind Detection Scheme

Carlo Condo, Member, IEEE, Seyyed Ali Hashemi, Student Member, IEEE, Arash Ardakani, Student Member, IEEE, Furkan Ercan, Student Member, IEEE, and Warren J. Gross, Senior Member, IEEE

Abstract—In blind detection, a set of candidates has to be decoded within a strict time constraint, to identify which transmissions are directed at the user equipment. Blind detection is required by the 3GPP LTE/LTE-A and fifth generation (5G) standards. With the selection of polar codes in 5G, the issue of blind detection of polar codes needs to be addressed. A polar code blind detection scheme has been recently proposed where the user ID is transmitted instead of some of the frozen bits. We propose an architecture to implement an improved version of such scheme. A first, coarse decoding phase helps selecting a subset of candidates that is decoded by a more powerful algorithm: an early stopping criterion is also introduced for the second decoding phase. The architecture relies on a tunable decoder that can be used for both phases. The architecture is synthesized and implementation results are reported for various system parameters. The reported area occupation and latency, obtained in 65-nm CMOS technology, are able to meet 5G requirements.

Index Terms—Polar code, blind detection, 5G.

I. INTRODUCTION

Blind detection requires the receiver of a set of bits to identify if said bits compose a codeword of a particular channel code. In 3GPP LTE/LTE-Advanced standards blind detection is used by the user equipment (UE) to receive control information related to the downlink shared channel. The UE attempts the decoding of a set of candidates, to identify if one of the candidates holds its control information, in a process similar to self-synchronization [1]. Blind detection is required in the 5th generation wireless communication standard (5G) as well: ongoing discussions are considering a substantial reduction of the time frame allocated to blind detection, from 16μs to 4μs. Blind detection must be performed frequently, and involves a high number of decoding attempts in a limited time [2]; it can thus lead to large implementation costs and high energy consumption. Blind detection solutions for codes adopted in existing standards can be found in [3]–[5].

Polar codes are a class of capacity-achieving error correcting codes, introduced by Arikan in [6]. They are characterized by simple encoding and decoding, and have been selected for use in 5G [7]. In [6], the successive-cancellation (SC) decoding algorithm has been proposed as well. It is optimal for infinite code lengths, but its error-correction performance degrades quickly at moderate code lengths. In its original formulation, it also suffers from long latency. SC list (SCL) decoding has been proposed in [8] to improve the error-correction performance of SC, at the cost of increased latency.

Blind detection of polar codes has been recently addressed in [9], where a blind detection scheme fitting within 3GPP LTE-Advanced and 5G requirements has been proposed. It is based on a two-step scheme: a first SC decoding phase helps selecting a set of candidates, subsequently decoded with SCL. An early stopping criterion for SCL is also proposed to reduce average latency. Another recent work on polar code blind detection [10] proposes a metric on which the outcome of the blind detection can be based.

In this brief, we propose a flexible decoder architecture that implements an improved blind detection scheme and its early stopping criterion, where we consider SCL also in the first decoding phase for better detection accuracy results. The architecture relies on an SCL decoder with tunable list size, that can be used for both the first and second decoding stages. A high-performance, low complexity selection circuit is proposed as well, to perform the selection of candidates between the two decoding phases. The architecture of the whole system is synthesized and implementation results are reported for various system parameters, showing that in CMOS 65 nm technology, 5G blind detection timing constraints can be reached with 5 parallel decoders.

II. PRELIMINARIES

A. Polar Codes

A polar code \( P(N, K) \) is a linear block code of length \( N = 2^n \) and rate \( K/N \), and it can be expressed as the concatenation of two polar codes of length \( N/2 \). This is due to the fact that the encoding process is represented by a modulo-2 matrix multiplication as \( x = uG^{\otimes n} \), where \( u = [u_0, u_1, \ldots, u_{N-1}] \) is the input vector, \( x = [x_0, x_1, \ldots, x_{N-1}] \) is the codeword, and the generator matrix \( G^{\otimes n} \) is the \( n \)-th Kronecker power of the polarizing matrix \( G = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \). The polarization effect
brought by polar codes allows to divide the $N$-bit input vector $\mathbf{u}$ between reliable and unreliable bit-channels. The $K$ information bits are assigned to the most reliable bit-channels of $\mathbf{u}$, while the remaining $N-K$, called frozen bits, are set to a predefined value, usually 0. Codeword $\mathbf{x}$ in the hardware-friendly version of [11], is computed as

$$a_i^1 = \text{sgn}(a_i)\text{sgn}(a_{i+2^{l-1}-1}) \min(|a_i|, |a_{i+2^{l-1}}|),$$

$$a_i^2 = a_{i+2^{l-1}} + (1 - 2\beta_i^r)\alpha_i,$$  \hspace{1cm} (1)

while $\beta$ is computed as

$$\beta_i = \begin{cases} \beta_i^l \oplus \beta_i^r, & \text{if } i < 2^{t-1}, \\ \beta_i^r, & \text{otherwise}, \end{cases}$$

where $\oplus$ denotes the bitwise XOR operation. The SC operations are scheduled according to the following order: each node receives $\alpha$ first, then sends $a_i^1$, receives $\beta_i^l$, sends $a_i^2$, receives $\beta_i^r$, and finally sends $\beta$. When a leaf node is reached, $\beta_i$ is set as the estimated bit $\hat{u}_i$:

$$\hat{u}_i = \begin{cases} 0, & \text{if } i \in \mathcal{F} \text{ or } a_i \geq 0, \\ 1, & \text{otherwise}, \end{cases}$$

where $\mathcal{F}$ is the set of frozen bits.

SC decoding suffers from modest error-correction performance with moderate and short code lengths. To improve it, the SCL algorithm was proposed in [8]. It is based on the same process as SC, but each time that an information bit is estimated at a leaf node, both its possible values 0 and 1 are considered. A list of $L$ codeword candidates is stored, so that a bit estimation results in $2L$ new candidates, half of which must be discarded. To this purpose, a path metric (PM) is associated to each candidate and updated at every new estimate: the $L$ paths with the lowest PM survive. In [12], the hardware-friendly formulation of the PM is

$$PM_{l} = \begin{cases} PM_{l-1}, & \text{if } \hat{u}_i = \frac{1}{2}(1 - \text{sgn}(a_i)), \\ PM_{l-1} + |a_i|, & \text{otherwise}, \end{cases}$$

where $l$ is the path index and $\hat{u}_i$ is the bit estimate $i$ in $l$.

### B. Blind Detection

The physical downlink control channel (PDCCH) is used in 3GPP LTE/LTE-Advanced to transmit the downlink control information (DCI) related to the downlink shared channel. The DCI carries information regarding the channel resource allocation, transport format and hybrid automatic repeat request, and allows the UE to receive, demodulate and decode. In LTE standard R8 [2], a cyclic redundancy check (CRC) is attached to the DCI payload before transmission. The CRC is masked according to a 16-bit ID of the UE to which the transmission is directed, or according to one of the system-wide IDs. Finally, the DCI is encoded with a convolutional code. The UE is not aware of the format with which the DCI has been transmitted: it thus has to explore a combination of PDCCH locations and formats, and DCI formats, in the common search space (CSS) and UE-specific search space (UESSS) and attempt decoding to identify useful DCIs, for a total of 44 candidates. This process is called blind decoding, or blind detection. For each PDCCH candidate in the search space, the UE performs channel decoding, and demasks the CRC with its ID. If no error is found in the CRC, the DCI is considered as carrying the UE control information.

In [9], polar codes have been considered within the aforementioned framework, and a blind detection scheme has been proposed. Frozen bit positions are selected to transmit the UE ID, $C_1$ candidates are received at the same time: in this case, $C_1 = 44$. The $C_1$ candidates are decoded with the simple SC algorithm, and a PM is obtained for each candidate, equivalent to the LLR of the last decoded bit: thanks to the serial nature of SC decoding, the LLR of the last bit can be interpreted as a reliability measure on the decoding process. The PMs are then sorted, to help the selection of the best candidates to forward to the following decoding phase. $C_2$ candidates are in fact selected to be decoded with the more powerful SCL decoding algorithm, that guarantees a better error-correction performance, at a higher implementation complexity. In [9], an early stopping criterion has been proposed as well. The first phase requires the full decoding of each candidate, to identify the $C_2$ codewords sent to the second phase. In the second phase, all codewords whose ID does not match the UE ID are discarded. Thus, as soon as the ID is shown to be different, the decoding can be interrupted. Since SC-based decoding algorithms estimate codeword bits sequentially, the ID evaluation can be performed every time an ID bit is estimated. In case the estimated bit is different from the UE ID bit, the decoding is stopped.

### III. PROPOSED BLIND DETECTION SCHEME

We substantially improve the effectiveness of the blind detection scheme in [9] by considering SCL also for the first decoding phase. We consider a list sizes $L_1 \geq 1$ for the first decoding phase, and $L_{\text{max}} > L_1$ for the second one. Table I shows the signal-to-noise ratio (SNR) requirements to meet given block error rate (BLER), missed detection rate (MDR), and false alarm rate (FAR) targets for the proposed blind detection scheme in comparison with [9]. We set $C_2 = 5$, $L_{\text{max}} = 8$ and $L_1 = 2$: it is a representative case for which $L_{\text{max}}$ guarantees good error-correction performance, and at which SCL decoders can be implemented with reasonable complexity. It can be seen that the proposed scheme improves the SNR requirements of [9] in terms of BLER, MDR, and FAR, for 0.4 dB, 0.4 dB, and 0.7 dB, respectively.

We now show the impact of the early-stopping criterion of [9] on the proposed blind detection scheme. Fig. 1 shows the average number of estimated bits when the early-stopping criterion is used. The ID values assigned to the $C_1$ candidates are randomly selected. The results in Fig. 1 consider each
of the $C_2$ candidates separately, since the number of candidates of length $N_1$ and $N_2$ in the second phase depends on the PMs received from the first phase, and thus on channel SNR. In the solid curves, the UE ID was sent through the considered code, while in the dashed curves it was not sent. For $N = 256$, in the case that the UE ID was sent, as the channel conditions improve, the number of estimated bits increases until stabilizing at a maximum average value. When the SNR is low, it is in fact more likely that the codeword carrying the UE ID is not selected to be among the $C_2$ candidates. Thus, the second phase easily encounters ID bits different from the UE ID early in the decoding process. At higher SNR, the codeword with the UE ID falls among the $C_2$ candidates with rising probability. The decoder tasked with its decoding does not interrupt the process, reaching 100% estimated bits, while the remaining $C_2 - 1$ decoders stop the decoding early, thus averaging the estimated bit percentage at a stable value. For $N = 512$ the trend is different: at low SNR values, the percentage of estimated bits is very close to 100%. As the SNR value increases, the average starts to decrease, until it settles on a stable value. At low SNR, it is in fact unlikely that a codeword with $N = 512$ is among the $C_2$ second phase candidates if the UE ID is not matching: the longer $N$ and lower rate contribute to a higher decoding reliability during the first phase. What if a frozen bit is found, the paths are not split, and the bit is decoded by storing the appropriate memory offsets for every channel condition: since among the $C_2$ candidates there is never one carrying the UE ID, all second phase decoders tend to stop the decoding early, at a percentage mostly influenced by the position of bits assigned to the ID. The use of SCL in the first decoding phase increases the probability of having the right candidate among the $C_2$ sent to the second phase, with respect to the scheme in [9]. Since the early stopping criterion helps interrupting unwanted decoding, the increased effectiveness of the overall blind detection scheme results in higher percentages of average estimated bits when the UE ID was sent.

### IV. HARDWARE ARCHITECTURE

To evaluate the implementation cost of the proposed blind detection scheme, we designed a decoder architecture that supports it, portrayed in Fig. 2. An array of flexible list size SCL decoders handles both the first and second decoding phase. A dedicated module selects the $C_2$ candidates for the second phase according to the criteria described in Section II-B.

#### A. SCL Decoder Architecture

We based our SCL decoder architecture on that of [13] and [14]: the decoding process follows the one described in Section II-A for a list size $L_{\text{max}}$. Most of the datapath and memories are instantiated $L_{\text{max}}$ times: multiple candidates are stored at the same time, with the best candidate being selected at the end of the decoding. While in [13] and [14] the final candidate is selected according to a CRC check, in the proposed architecture no CRC is considered, and the validity of the final candidate is based on the matching ID and PM value.

The SCL decoder progresses by computing (1) and (2), and stores the intermediate LLR values in internal memories. These calculations are performed by $L_{\text{max}}$ parallel sets of $P$ processing elements (PEs), with $P$ being a power of 2. When a leaf node is reached, the decoder controller module identifies the leaf node as either an information bit or a frozen bit. If a frozen bit is found, the paths are not split, and the bit is estimated only as 0, and the $L$ memories are updated with the same bit or LLR values. Instead, in case of an information bit, both 0 and 1 are considered, so that paths are split, and the PMs updated for the $2L$ candidates according to (5). The PMs are then sorted, identifying the $L$ surviving paths.

All memories in the decoder are registers, enabling the internal LLR and $\beta$ values to be read, updated by the PEs, and written back in a single clock cycle. At the same time, the paths are either updated or split and updated, and the new PMs computed. In the following clock cycle, in case the paths were split, the PMs are sorted and the surviving paths selected. It should be noted that codes with different code lengths can be decoded by storing the appropriate memory offsets for every considered code in a dedicated memory, thanks to the recursive structure of SC-based decoders. Dedicated multiplexing initializes the memory of the subcode of desired length with the channel LLRs.

#### B. Flexible List Size SCL Decoder

A straightforward solution to implement the proposed blind detection scheme would entail the instantiation of a set of decoders with list size $L_1$ for the first decoding phase, and a set with list size $L_{\text{max}}$ for the second phase. Given the complexity of list decoders, this approach would lead to very large

![Fig. 2. Polar codes blind detection system architecture.](image-url)
area occupation and power consumption to meet the 5G timing requirements. In order to maximize resource sharing, we propose a flexible list size SCL decoder. The decoder has been sized for $L_{\text{max}} > L_1$, and the effective list size can be selected through a dedicated input.

The $L_{\text{max}} - L_1$ paths that are not used in the first decoding phase are used to decode up to $\lceil (L_{\text{max}} - L_1)/L_1 \rceil$ additional candidates at the same time; in order to exploit the unused paths, additional functional modules are necessary. The baseline decoder uses a single memory to store the channel LLR values, sharing it among the different paths. If different codewords have to be decoded at the same time, the channel memory needs to be instantiated not once, but $\lceil L_{\text{max}}/L_1 \rceil$ times. Moreover, the decoder relies on sorting and selection logic that identifies the surviving $L_{\text{max}}$ ones after paths are split. To support the parallel decoding of $\lceil L_{\text{max}}/L_1 \rceil$ candidates, as many sorting and selection modules targeting the selection of $L_1$ paths out of $2L_1$ are instantiated. If $L_1 = 1$ is selected, the path splitting and PM sorting steps are bypassed, reverting decoders to the standard SC case. Since a single set of SCL decoders can handle both decoding phases, the total number of decoders is $N_{\text{SCL}}$ (see Fig. 2). However, the effective number of decoders for the first decoding phase is $N_{\text{SCLmax}} = N_{\text{SCL}} \times \lceil L_{\text{max}}/L_1 \rceil$.

The early stopping technique described in Section II-B has been also implemented. The decoder receives as input the position of the ID bits and the value of the UE ID: every time a bit in an ID position is estimated, the bit value is compared to the expected UE ID bit. All paths whose estimated bit does not match the UE ID bit are deactivated. This operation is performed after the $L$ surviving paths have been selected, in order not to force the survival of unlikely paths and increase the FAR. In case all paths have been deactivated, the decoding is stopped. The early stopping logic can be activated and deactivated by means of a dedicated control signal. Since the same hardware is used for both decoding phases, early stopping is enabled only during the second one.

### C. PM Sorting and Candidate Selection

Fig. 3 depicts the architecture of the PM sorting and candidate selection block. It processes the output of the first decoding phase to select the $C_2$ candidates for the second phase, and selects the overall system output based on the results from the second phase. For each of the $N_{\text{SCL}}$ first phase decoders, a PM and a flag signaling a UE ID match are received. They are stored every time the respective $\text{Valid}$ signal is risen by the decoder. The $\text{Valid}$ signal is also used as an enable for the PM and UE ID match register address counter, and for the counter keeping track of how many codewords had a matching UE ID after the first phase. When all the $C_1$ candidates have gone through the first decoding phase, a $\text{Valid}$ signal is issued to the sorter module, that receives as input all the stored PMs. The sorter module returns the $C_2$ minimum PMs in as many clock cycles; each PM is compared to all the others, and a single clock cycle is necessary to identify the minimum one, that is excluded from the subsequent comparison. When the $C_2$ minima have been found, the sorter module considers how many candidates had a matching UE ID after the first phase, and selects the $C_2$ candidates for the second phase among them and those with the minimum PM values. The $C_2$ candidates are sent to the $N_{\text{SCLmax}}$ decoders by means of a dedicated counter. Returning PMs and UE ID match flags are received and compared by another selector: when all $C_2$ candidates have been decoded, the selected codeword, if any, is output.

### V. IMPLEMENTATION RESULTS

The architecture proposed in Section IV has been described in VHDL and synthesized in TSMC 65 nm CMOS technology. Table II reports the synthesis results for the architecture sized for a maximum code length $N_{\text{max}} = 512$, a maximum list size $L_{\text{max}} = 8$, $C_2 = 5$, and a target frequency $f = 1$ GHz. Various $N_{\text{SCLmax}}$ values have been considered, leading to different latencies and area occupations. Since during the first decoding phase $L_1 = 2$, the effective number of decoders $N_{\text{SCL}}$ is equal to $4N_{\text{SCLmax}}$, even if only $N_{\text{SCLmax}}$ are physically instantiated. Regarding the area, the $N_{\text{SCLmax}}$ SCL decoders contribute to the majority of the complexity, ranging from 97.8% when $N_{\text{SCLmax}} = 1$ to 99.7% when $N_{\text{SCLmax}} = 5$. The logic complexity of the PM sorting and candidate selection module remains almost unchanged at the variation of $N_{\text{SCLmax}}$, being mainly affected by $C_1$ and $C_2$. Memories have been synthesized with registers only, without the use of RAM, and account for 36% of the total area occupation.

The system worst case latency can be found as

$$T_{\text{bd}} = \left( \frac{C_1}{N_{\text{SCL}}} \left( \frac{T_{\text{SCL}}^1 + T_{\text{SCL}}^2}{2} \right) \right)^{N_{\text{SCLmax}}} + T_{\text{sort}} + \left( \frac{C_2}{N_{\text{SCLmax}}} \right) \max \left( \frac{T_{\text{SCL}}^1, T_{\text{SCL}}^2} \right),$$

where $T_{\text{SCL}}^1$ and $T_{\text{SCL}}^2$ are the SCL decoding latencies for codes of length $N_1$ and $N_2$, respectively, while $T_{\text{sort}}$ is the number of time steps required to sort the PM of the first decoding phase and obtain the $C_2$ candidates out of the $C_1$ candidate locations. Also, it is worth remembering that for the proposed architecture, $N_{\text{SCL}} = \lceil L_{\text{max}}/L_1 \rceil \times N_{\text{SCLmax}}$. The SCL decoding latency can be found as [12]

$$T_{\text{SCL}}^x = 2N_x + K_x + 16 - 2,$$

for $x \in \{1, 2\}$, where 16 is the latency introduced by the estimation of the UE ID bits. From the results presented in Table II, it is possible to see that even when considering the relatively old 65 nm technology node, the 16$\mu$s worst case latency target can be reached with a single SCL decoder running at a frequency of 1 GHz, while $N_{\text{SCLmax}} = 5$ guarantees a worst case latency of 3.6$\mu$s, meeting the 4$\mu$s target as well.

However, considering only the worst case latency is indeed an unrealistic scenario. To begin with, while there is no guarantee on how the $C_2$ candidates are distributed among $N_1$ and $N_2$, simulation results have shown that we can expect the $C_2$ candidates either to favor the shorter code length, or to be equally divided between $N_1$ and $N_2$ candidates. Thus, the factor $\left\lceil \frac{C_2}{N_{\text{SCLmax}}} \right\rceil \max \left( T_{\text{SCL}}^1, T_{\text{SCL}}^2 \right)$ in (6), i.e., the contribution of the second decoding phase, could be better expressed as

$$\left\lceil \frac{C_2}{N_{\text{SCLmax}}} \right\rceil T_{\text{SCL}}^1 + \left( \frac{C_2 - \lfloor C_2/2 \rfloor}{N_{\text{SCLmax}}} \right) T_{\text{SCL}}^2.$$

Note that this is still a conservative assumption, since it entails the $C_2$ candidates equally divided among the two code lengths. We can refine this assumption by taking into account the effect of early stopping. We can approximate the latency reduction with
a multiplicative factor $E_s$ associated to $T_{SCL}$. Consequently, the average latency of the blind detection system, for $N_{SCL} < C_2$, can be computed as

$$T_{bd} = \left[ \frac{C_1}{N_{SCL}} \left( \frac{T_{SCL}}{2} + \frac{T_{SCL}}{2} \right) + T_{sort} \right] + \left[ \frac{C_2}{N_{SCL}} \right] T_{SCL} E_s^1 + \left[ \frac{C_2}{N_{SCL}} \right] \frac{T_{SCL}}{2} E_s^2 , \tag{7}$$

while for $N_{SCL} \geq C_2$ it becomes

$$T_{bd} = \left[ \frac{C_1}{N_{SCL}} \right] \left( \frac{T_{SCL}}{2} + \frac{T_{SCL}}{2} \right) + T_{sort} + \max \left( \frac{T_{SCL}}{2} E_s^1, \frac{T_{SCL}}{2} E_s^2 \right). \tag{8}$$

Considering the number of UEs connected to the shared channel, blind detection is dominated by instances in which a particular UE ID is not sent. Thus, we can set $E_s$ as the fraction of bits expressed by the dashed curves in Fig. 1. The average latency results in Table II show substantial reduction with respect to the worst case latency case, within a more realistic framework. Even within the 65 nm technology node, with $N_{SCL} \geq 4$, the average latency is below 4 $\mu$s. With the latest technology nodes, a substantially higher frequency will be easy to achieve, along with proportionally smaller area occupation. It is consequently safe to assume that the 4 $\mu$s worst case latency target can be easily met for $N_{SCL} \geq 3$, and the average latency with $N_{SCL} \geq 2$.

### VI. Conclusion

In this brief, we proposed an architecture to implement an improved two-phase blind detection scheme, guaranteeing superior performance metrics, and reduced latency through early stopping. It is based on an SCL decoder with tunable list size used for both decoding stages. The architecture is synthesized and implementation results are reported for various system parameters. Area occupation and latency, obtained in 65 nm CMOS technology, are able to meet 5G requirements.

### REFERENCES

[1] S. T. Klein and Y. Wiseman, “Parallel Huffman decoding with applications to JPEG files,” Comput. J., vol. 46, no. 5, pp. 487–497, Jan. 2003.

[2] Physical Layer Procedures V.8.2.0, document TS 36.213, 3GPP, Sophia Antipolis, France, 2008.

[3] R. Moosavi and E. G. Larsson, “A fast scheme for blind identification of channel codes,” in Proc. IEEE Glob. Telecommun. Conf., Dec. 2011, pp. 1–5.

[4] T. Xia and H.-C. Wu, “Novel blind identification of LDPC codes using average LLR of syndrome a posteriori probability,” IEEE Trans. Signal Process., vol. 62, no. 3, pp. 632–640, Feb. 2014.

[5] J. Zhou, Z. Huang, C. Liu, S. Su, and Y. Zhang, “Information-dispersion-entropy-based blind recognition of binary BCH codes in soft decision situations,” Entropy, vol. 15, no. 5, pp. 1705–1725, 2013.

[6] E. Arikan, “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.

[7] BLER Performance of List Decoding for Enhanced Turbo Codes, V1.0.0, document TSG RAN WG1 #87, 3rd Gener. Partnership Project, Reno, NV, USA, Nov. 2016. [Online]. Available: http://www.3gpp.org/ftp/tsg_ran/WG1_RL1/TSGR1_87/Report/Final_Minutes_report..RAN1%2387_v100.zip

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

[9] C. Condo, S. A. Hashemi, and W. J. Gross, “Blind detection with polar codes,” IEEE Commun. Lett., vol. 21, no. 12, pp. 2550–2553, Dec. 2017.

[10] P. Giard, A. Balatsoukas-Stimming, and A. Burg, “Blind detection of polar codes,” in Proc. IEEE Int. Workshop Signal Process. Syst., Oct. 2017, pp. 1–6.

[11] C. Leroux, A. J. Raymond, G. Sarkis, and W. J. Gross, “A semi-parallel successive-cancellation decoder for polar codes,” IEEE Trans. Signal Process., vol. 61, no. 2, pp. 289–299, Jan. 2013.

[12] A. Balatsoukas-Stimming, M. B. Panfo, 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.

[13] S. A. Hashemi, C. Condo, and W. J. Gross, “A fast polar code list decoder architecture based on sphere decoding,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 63, no. 12, pp. 2368–2380, Dec. 2016.

[14] S. A. Hashemi, C. Condo, and W. J. Gross, “Fast simplified successive-cancellation list decoding of polar codes,” in Proc. IEEE Wireless Commun. Netw. Conf., Mar. 2017, pp. 1–6.