Low-Latency Modulation- and Correlation-Adaptive ORBGRAND-AI Decoder

Zeynep Ece Kizilates[1], Arslan Riaz[1], Akshaya Bali[1], Moritz Grundei[2],
Muriel Medard[3], Ken R. Duffy[4], Rabia Tugce Yazicigil[1]
[1]Department of Electrical and Computer Engineering, Boston University, Boston, MA, USA,
[2]Department of Electrical and Computer Engineering Technical University of Munich, Munich, Germany,
[3]Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA,
[4]Department of Electrical and Computer Engineering & Mathematics Northeastern University, Boston, MA, USA

***Abstract*—We present the first integrated symbol-level decoder using the ORBGRAND-AI algorithm, supporting any moderate-redundancy binary linear code. By leveraging channel noise correlation, the design improves both decoding performance and energy efficiency. Unlike conventional decoders that rely on off-chip bit-level reliability calculations and interleavers—adding latency and complexity—our architecture adopts a fully integrated, modulation-adaptive approach. It processes raw I/Q data directly, computes noise correlation-aware soft information via multivariate Gaussian models, and eliminates the need for interleaving, hence improves latency. A novel pseudo-BPSK simplification enables the reconfiguration of a single demapper unit to support higher-order modulations, such as 4/16/64-QAM, using shared compute units while also improving hardware efficiency. Fabricated in 40nm CMOS, the decoder achieves 5.1Gbps throughput at 1.61pJ/bit and delivers the highest reported area efficiency of 11.3Gbps/mm^2 for 64-QAM at FER=10^{-5}. These results demonstrate a practical, reconfigurable, low-latency, and energy-efficient decoder chip.**

***Index Terms*—GRAND, error correction, soft-input, correlation, interleavers, ORBGRAND-AI**

I. Introduction

As wireless technologies evolve towards 6G, emerging applications demand low-latency, high-reliability communication under tight delay constraints [1]. To meet these requirements, recent research has focused on developing efficient decoding algorithms and hardware architectures for error correction. Traditional decoders, including polar decoders [2], [3] and recent proposals [4], [5], assume independent noise per bit and rely on bit-level log-likelihood ratios (LLRs) that do not incorporate noise correlation. To align with this assumption, they employ interleavers to disrupt correlation (Fig. 1a). While effective at mitigating correlation, interleaving introduces latency, increases buffer overhead, and reduces effective channel capacity [6]. These trade-offs conflict with low-latency goals.

Guessing Random Additive Noise Decoding (GRAND) [7] and its variants offer a codebook-agnostic decoding alternative by focusing on noise patterns rather than codewords. This noise-centric approach allows the GRAND family to incorporate correlated noise models to address these challenges. Recent extensions of GRAND, such as Symbol-Level Ordered Reliability Bits GRAND (ORBGRAND) [8] and ORBGRAND Approximate Independence (ORBGRAND-AI) [6], address these challenges by optimizing decoding at the symbol level. ORBGRAND-AI, in particular, introduces blockwise soft information modeling to capture correlation structures across neighboring symbols. This not only improves decoding performance by up to 2dB in the presence of temporally correlated noise but also reduces buffering delays, achieving an effective end-to-end latency gain, as illustrated in Fig. 1b-c.

These algorithms require symbol- or block-level soft information, which standard off-chip bit-level demappers cannot provide. To support this in practice, the decoder must internally generate correlation-aware soft information on-chip. In this work, we present the first integrated, modulation-adaptive symbol-level decoder in 40nm CMOS, featuring an on-chip demapper that computes noise correlation-aware symbol-level soft information. To support higher-order modulations without increasing hardware complexity, we demonstrate a modulation-adaptive pseudo-BPSK demapper that simplifies high-order QAM constellations by reconfiguring the shared compute units at no additional hardware cost. The fabricated decoder operates at 100MHz and at target Frame Error Rate (FER) 10^{-5}, achieves 5.1Gbps throughput at 1.61pJ/bit for 64-QAM with $\rho = 0.75$, showing improved performance with both increasing modulation order and noise correlation.

II. Algorithm and Mathematical Modeling

This section describes the ORBGRAND-AI algorithm [6] assuming a Gauss-Markov noise model. The decoder directly processes the received symbol sequence $Y^{n_s} = (Y_1, Y_2, \ldots, Y_{n_s})$ of length n_s where each symbol corresponds to m_s bits depending on the modulation order. The sequence is divided into non-overlapping blocks of size b, as $Y^{n_s} = (Y_1^b, Y_2^b, \ldots, Y_{n_s/b}^b)$. To capture correlation in the channel, the noise across each block is modeled as a Gauss-Markov process, resulting in a multivariate Gaussian distribution. For each block Y_j^b, the decoder evaluates the probability that it corresponds to a transmitted symbol pattern $\mu_i \in \chi^b$ as:

$$\mathcal{N}(Y_j^b; \mu_i; \Sigma) = \frac{\exp\left[-\frac{1}{2}(Y_j^b - \mu_i)^T \Sigma^{-1} (Y_j^b - \mu_i)\right]}{\sqrt{(2\pi)^b |\Sigma|}} \quad (1)$$

This work was supported by the Defense Advanced Research Projects Agency (DARPA), grant no. HR00112120008 and National Science Foundation (NSF) ECCS Award numbers 2128517 and 2128555. The opinions, findings, and conclusions expressed in this material are those of the author(s) and do not necessarily reflect the views of these organizations.

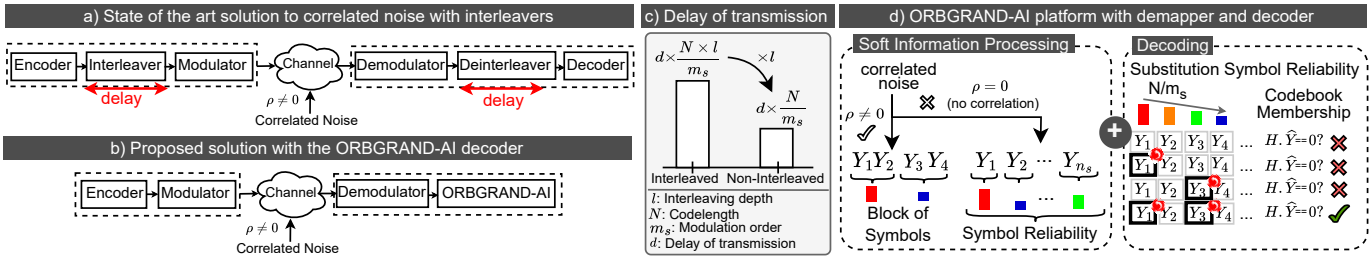

Fig. 1. a) Conventional communication systems employing interleavers, b) Proposed symbol-level decoder using ORBGRAND-AI-based system, c) Latency from block interleaving, where data is written and read in different orders, d) ORBGRAND-AI chip with soft information processing via demapper/decoder.

Here, $\Sigma \in \mathbb{R}^{b\times b}$ is the noise covariance matrix, with elements $\Sigma_{ij} = \sigma^2\rho^{|i-j|}$, where ρ is the noise correlation coefficient and σ^2 is the noise variance. We define δ for each candidate pattern with $\delta_j^i = (Y_j^b - \mu_i)^T\Sigma^{-1}(Y_j^b - \mu_i)$. The candidate with the smallest δ_j^i (i.e., $\delta_j^{\min}$) corresponds to the most likely transmitted pattern and is tested for codebook membership with the parity-check matrix $H^{(N-K)\times N}$ where N is the codeword length and K is the number of information bits. If the syndrome check $H \cdot y^N = 0$ is satisfied, decoding is complete. Notice that for $b = 1$, this reduces to a standard Euclidean distance, aligning with the Symbol-Level ORBGRAND algorithm [8].

If the membership check fails, the decoder evaluates the likelihood of each candidate pattern across all blocks. The likelihood L_j^i of a pattern μ_i for symbol block Y_j^b becomes:

$$L_j^i = \frac{\mathcal{N}(Y_j^b;\mu_i;\Sigma)}{\sum_{k=1}^{2^{bm_s}}\mathcal{N}(Y_j^b;\mu_k;\Sigma)} = \frac{\exp\left[-\frac{1}{2}\delta_j^i\right]}{\sum_{k=1}^{2^{bm_s}}\exp\left[-\frac{1}{2}\delta_j^k\right]} \quad (2)$$

The denominator requires normalization over 2^{bm_s} candidates, which is computationally expensive for hardware implementation. In this work, we address this issue by employing the max-log approximation, which preserves the relative ordering of likelihoods while eliminating costly operations:

$$L_j^i \approx \frac{\mathcal{N}(Y_j^b;\mu_i;\Sigma)}{\max_{\mu_k}\mathcal{N}(Y_j^b;\mu_k;\Sigma)} \Rightarrow \ln L_j^i \approx \delta_j^{\min} - \delta_j^i \quad (3)$$

The soft information is then sorted to determine the most likely symbol substitutions across all blocks. Based on this sorted list, the decoder generates candidate patterns that will replace/swap the blocks of symbol using logistic weight (LW) ordering, where LW is defined as the sum of the indices of the substituted blocks in the sorted list. The generated patterns are then mapped to the received sequence to form the next most likely codeword candidates, which are iteratively subject to codebook membership check. An example is shown in Fig. 1d, where blocks are iteratively substituted with their next most likely patterns until the membership check is satisfied.

III. Chip Architecture

This section describes the chip architecture, including the demapper and the decoder designs (Fig. 2a), as well as the dynamic activation of each pipelined stage (Fig. 2b).

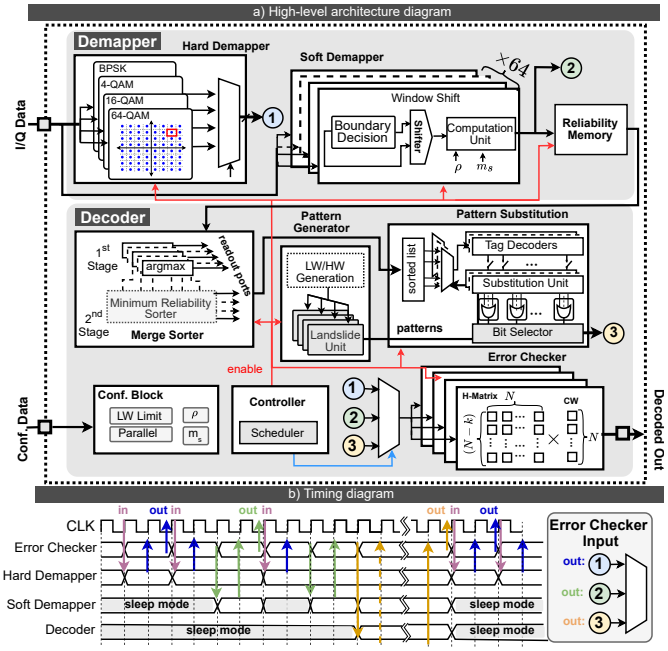

Fig. 2. a) Top-level architecture of the ORBGRAND-AI decoder, b) The timing diagram showing the three-staged pipelined sequence of each module.

A. Demapper Design and Pipelined Architecture

1) Hard Demapping: The first stage performs hard demapping by translating the received symbols Y^{n_s} into a bit sequence $y^N = (y_1, \ldots, y_N)$ using modulation-specific lookup tables for BPSK, 4/16/64-QAM. The resulting bits are then passed to the error checker, where the codebook membership check is performed by computing the matrix-vector product $H \cdot y^N$. If the sequence is a valid codeword, the decoding terminates early by clock-gating the remaining pipeline stages.

2) Soft Demapping under Correlated Noise: If the hard-demapped bit sequence fails the membership check, the second pipeline stage is activated to compute correlation-aware soft information and identify the most likely symbol mappings under correlated noise. As described in Section II, this requires evaluating the simplified soft information, δ_j^i metric for each candidate symbol pattern μ_i. The number of candidate patterns per block grows exponentially with both the modulation order m_s and block size b, reaching 2^{bm_s}, (e.g., 4096 for 64-QAM, $b = 2$), making full evaluation impractical in hardware.

To reduce complexity and enable uniform computation across all modulation schemes, we adopt a pseudo-BPSK simplification that reduces the search space to four symbol candidates per block, following the approximation introduced

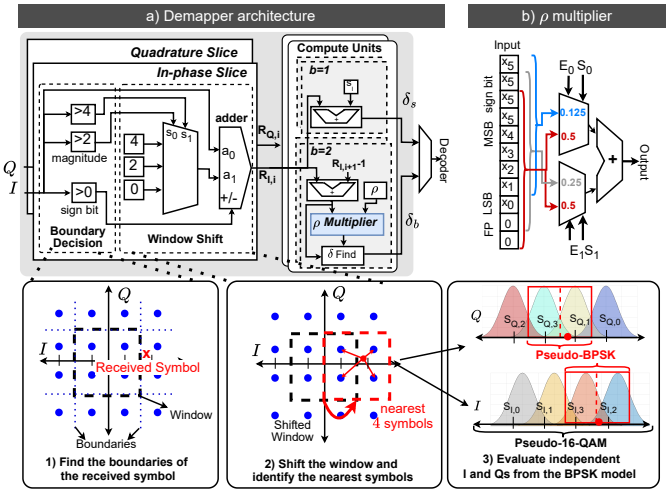

Fig. 3. a) Soft demapper and pseudo-BPSK simplification for higher-order modulations, b) Rho (ρ) multiplier details for noise-correlation computation.

in [8] with negligible (<0.3dB) loss in decoding performance. Fig. 3 illustrates the boundary decision circuit, which identifies the region of the received symbol and selects the two nearest constellation points for each I and Q axis via a window shift. This effectively binarizes the I and Q components, which are then treated as independent binary symbols throughout the decoding. As a result, all modulations are treated as pseudo-BPSK, enabling shared fixed-point compute units and simplifying decoder design.

For each of the resulting pseudo-BPSK candidates, δ_j^i is computed independently for I and Q. If $b = 2$ (correlated case), this involves a matrix-vector multiplication with Σ^{-1}, which is implemented using a shift-add-based ρ multiplier (Fig. 3b), reducing area by $5.3\times$ over conventional multipliers. For uncorrelated noise ($b = 1$), the demapper simplifies to basic Euclidean distance computation using adder/subtractors.

B. Decoder Architecture

1) Merge Sorter: Once the soft demapper generates block-level soft information, a two-stage merge sorter ranks all substitution candidates in descending order of likelihood. The first stage performs local sorting by comparing the δ_j^i values within each block using argmax modules, which initially identify the most likely symbol substitutions (corresponding to $\delta_j^{\min}$) and continue sorting if a valid codeword has not yet been found. Each argmax module includes a subtractor that computes $\ln L_j^i = \delta_j^{\min} - \delta_j^i$, enabling the second-stage sorter to operate directly on the simplified likelihoods.

The second stage recursively merges sorted lists using a minimum reliability sorter proposed in [4]. This two-stage design decouples local and global sorting: unlike the flat sorting architecture in [4], which requires pipelining to meet timing, our merge-based approach reduces comparator depth and eases critical path constraints. As shown in Fig. 4a, a MUX controller sequentially reads elements from each sublist and merges them into a global list. The first stage continuously updates these sublists until the list is full. If an element is selected from a sublist, the MUX controller fetches the next

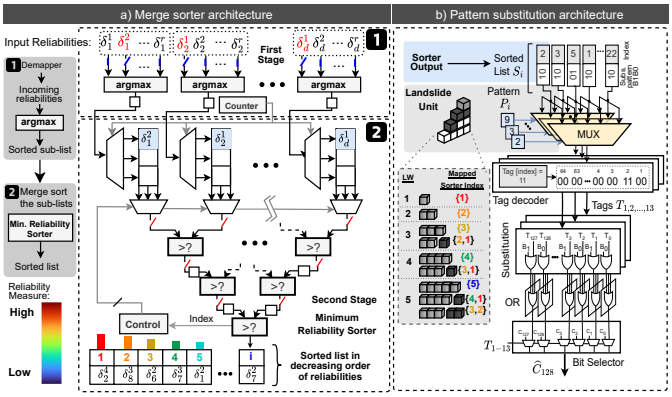

Fig. 4. a) Merge sorter architecture, b) Pattern substitution architecture.

element. Once a sublist is fully passed to the second stage, the tag controller disables that path, dynamically deactivating comparators to reduce switching activity.

2) Pattern Generation and Substitution: To generate codeword candidates from the sorted list of symbol substitutions in decreasing order of likelihood, the decoder uses a pattern generation and substitution unit as shown in Fig. 4b. The pattern generation unit provides sets of most likely symbol substitutions in increasing order of LW by using the landslide algorithm [9]. Each pattern specifies the indices of the symbol blocks to be substituted. These patterns are generated on the fly using a hardware-efficient implementation of landslide [4].

These generated patterns are passed to the pattern substitution unit, which first remaps the substitutions to their original indices and then updates the codeword bits at these locations using the corresponding substitution pattern. The remapping is done using MUXes that provide locations of substitutions as non-zero tags. These tags are used to place the substitution at its proper position in the 128-bit vector. Since the maximum HW is 13, up to 13 MUXes are instantiated—one per substitution. All substitutions are combined using bitwise OR logic to form candidate codewords, which are then tested iteratively by the error checker until one satisfies the membership check.

IV. Measurement Results

The die micrograph of the fabricated decoder in 40nm CMOS is shown in Fig. 5a. The chip consumes 8.18mW from a 0.95V supply at 100MHz, and its frequency-voltage scaling performance is illustrated in Fig. 5b. Fig. 5c presents the measured FER for a Cyclic Redundancy Check (CRC) code at $R = 0.875$ with $N = 128$ bits across various modulation schemes and noise correlation levels. The results show that decoding performance improves with higher correlation coefficients by up to 1 dB with $\rho = 0.5$. Notably, this improvement is achieved without the use of interleavers, preserving higher channel capacity and avoiding the additional latency overhead.

Fig. 6 presents the measured energy and latency, covering both demapping and decoding processes. At a target FER of 10^{-5} with 64-QAM, the chip consumes 1.61pJ/bit for $\rho = 0.75$ and 1.88pJ/bit for $\rho = 0$. With increased noise correlation, both energy and latency improve by up to $1.8\times$ and $2.3\times$, respectively, at the same E_b/N_0.

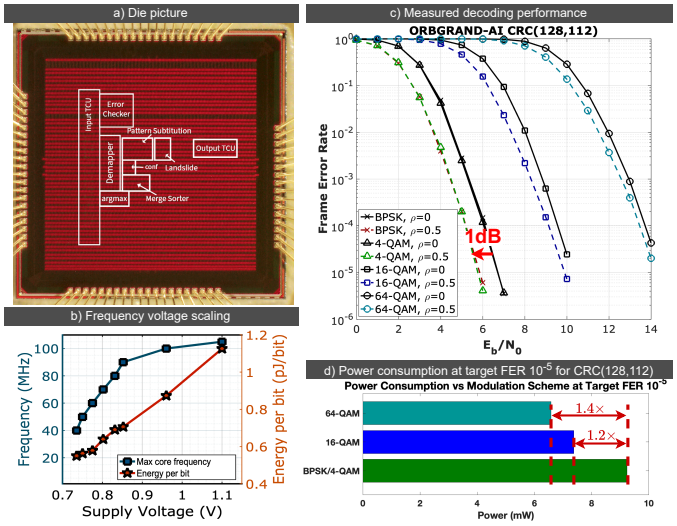

Fig. 5. a) Die micrograph, b) Frequency-voltage scaling, c) Measured decoding performance, d) Power consumption with different configurations.

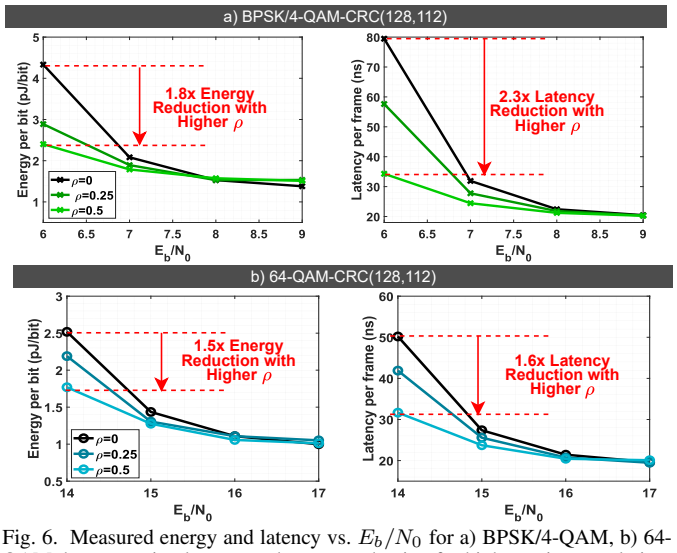

Fig. 6. Measured energy and latency vs. E_b/N_0 for a) BPSK/4-QAM, b) 64-QAM demonstrating latency and energy reduction for higher noise correlation.

In addition to the correlation-aware improvements, symbol-level decoding further enhances energy efficiency with higher-order modulations, as fewer symbols represent a fixed-length frame. This reduces the number of processed blocks, enabling dynamic deactivation of compute units and minimizing switching activity. In particular, several lateral comparison units in the sorter can be disabled, significantly reducing sorting effort and power consumption. This gain is shown in Fig. 5d and in Fig. 7. For the same target FER and code rate, we show measurement results for BPSK and 64-QAM and demonstrate a $1.4\times$ reduction in power consumption. These gains are enabled by the pseudo-BPSK simplification, which reduces the candidate list to a fixed, binary-sized set regardless of modulation order m_s. This reduces the complexity and resource load of the sorter, one of the most power- and area-critical components in GRAND-like decoders.

Furthermore, as proposed in [6], Fig. 7 shows that in the presence of correlated noise (see blue-highlighted columns), the decoder can operate with higher-rate codes for the same

Reference	This Work				[4] JSSC'24	[5] ISSCC'24	[2] VLSI'22	[3] JSSC'21
Technology (nm)	40				40	28	28	40
Code	CRC(Universal)				CA-Polar/CRC (Universal)	BOSS code	Polar	Polar
Decoding Algorithm	ORBGRAND-AI				ORBGRAND	BOSS dec.	SCL	SCL
Interleaver Req.	No				Yes	Yes	Yes	Yes
Code Length	Up to 128				Up to 256	128	1024	Up to 1024
Supply (V)	0.95				1.0	0.95	1.05	0.9
Quantization (Bits)	6-8*				6	N.R.	6	6
Frequency (MHz)	100				90	590	413	430
Core Area (mm^2)	0.45				0.4	0.37	0.59	0.64
Target FER	10^{-5}				10^{-5}	10^{-5}	10^{-5}	10^{-5}
Code Rate	0.875		0.914		0.94	0.1	0.5	0.5
Correlation (ρ)	0 (interleaved)		0.75		0	0	0	0
Modulation	BPSK	64QAM	BPSK	64QAM	N.A.	N.A.	N.A.	N.A.
Power (mW)	9.24	6.58	11.4	8.18	4.8	33.3	101.4	42.8
Energy per info. bit (pJ/bit)	2.97	1.88	2.53	1.61	1.2	48.6	219	26.4
Latency per frame (ns)	36	32	26	23	61.3	21.9	1100	N.R.
Info. Throughput (Gbps)	3.1	3.5	4.5	5.1	3.93	0.64	0.47	1.63
Normalized (40 nm, 0.95 V)								
Core Area (mm^2)	0.45				0.40	0.76	1.2	0.64
Power (mW)	9.24	6.58	11.4	8.18	4.33	47.6	118.6	47.7
Energy per info. bit (pJ/bit)	2.97	1.88	2.55	1.61	1.08	99.2	365.9	29.4
Info. Throughput (Gbps)	3.1	3.5	4.5	5.1	4.0	0.4	0.3	1.6
Area Efficiency (Gbps/mm^2)	6.88	7.77	10.0	11.33	10.0	0.6	0.3	2.5

*BPSK/4-QAM: 6-bit Quantization
16-QAM: 7-bit Quantization
64-QAM: 8-bit Quantization
N.R.: Not reported
N.A.: Not applicable, these decoders rely on per-bit reliability

Fig. 7. Performance highlights and comparison with the state-of-the-art.

target FER. The improved performance under correlation enables the use of weaker codebooks without degrading decoding reliability, resulting in higher information throughput, up to $1.45\times$ compared to interleaved lower-rate designs.

The chip achieves the highest reported area efficiency of 11.3Gbps/mm^2 within a compact 0.45mm^2 core. We show that this design with 64-QAM modulation provides $10.9\times$, $3.1\times$, $1.3\times$, $7.9\times$ higher throughput compared to the decoders in [2]–[5]. The chip demonstrates high energy efficiency, consuming $136\times$, $16.5\times$, $30.2\times$ less energy per information bit compared to the designs in [2], [3], [5], but $1.3\times$ more than [4], which relies on off-chip bit-reliability calculation and interleavers and thus introduces higher end-to-end latency. By integrating demapping and decoding at the symbol level with support for correlated noise conditions, this decoder presents a promising solution realizing some of the low-latency objectives of 6G.

References

[1] G. Durisi, T. Koch, and P. Popovski, "Toward Massive, Ultrareliable, and Low-Latency Wireless Communication With Short Packets," *Proc. IEEE*, vol. 104, no. 9, pp. 1711–1726, 2016.

[2] D. Kam, B. Y. Kong, and Y. Lee, "Low-Latency SCL Polar Decoder Architecture Using Overlapped Pruning Operations," *IEEE TCAS-I: Regular Papers*, vol. 70, no. 3, pp. 1417–1427, 2023.

[3] Y. Tao, S.-G. Cho, and Z. Zhang, "A Configurable Successive-Cancellation List Polar Decoder Using Split-Tree Architecture," *IEEE JSSC*, vol. 56, no. 2, pp. 612–623, 2021.

[4] A. Riaz, A. Yasar, F. Ercan, W. An, J. Ngo, K. Galligan, M. Médard, K. R. Duffy, and R. T. Yazicigil, "A Sub-0.8-pJ/bit Universal Soft-Detection Decoder Using ORBGRAND," *IEEE JSSC*, pp. 1–15, 2024.

[5] D. Kam, S. Yun, J. Choe, Z. Zhang, N. Lee, and Y. Lee, "2.8 A 21.9ns 15.7 Gbps/mm² (128,15) BOSS FEC Decoder for 5G/6G URLLC Applications," in *2024 IEEE ISSCC*, vol. 67, 2024, pp. 50–52.

[6] K. R. Duffy, M. Grundei, and M. Médard, "Using Channel Correlation to Improve Decoding - ORBGRAND-AI," in *IEEE GLOBECOM*, 2023, pp. 3585–3590.

[7] K. R. Duffy, J. Li, and M. Médard, "Capacity-Achieving Guessing Random Additive Noise Decoding," *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4023–4040, 2019.

[8] W. An, M. Médard, and K. R. Duffy, "Soft Decoding without Soft Demapping with ORBGRAND," in *IEEE ISIT*, 2023, pp. 1080–1084.

[9] K. R. Duffy, W. An, and M. Médard, "Ordered Reliability Bits Guessing Random Additive Noise Decoding," *IEEE Trans. Signal Process.*, vol. 70, pp. 4528–4542, 2022.