2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)
Workshops - W4: First Int. Workshop on Enabling Technologies for TeraHertz Comm.

Terabits-per-Second Throughput for Polar Codes

Altuğ Süral [1], E. Göksu Sezer [1], Yiğit Ertuğrul [1], Orhan Arıkan [1,2] and Erdal Arıkan [1,2]
[1] *POLARAN LTD.*
[2] *Bilkent University*
Ankara TR-06800, Turkey
{altug.sural, goksu.sezer, yigit.ertugrul, orhan.arikan, erdal.arikan}@polaran.com

***Abstract*—By using Majority Logic (MJL) aided Successive Cancellation (SC) decoding algorithm, an architecture and a specific implementation for high throughput polar coding are proposed. SC-MJL algorithm exploits the low complexity nature of SC decoding and the low latency property of MJL. In order to reduce the complexity of SC-MJL decoding, an adaptive quantization scheme is developed within 1-5 bits range of internal log-likelihood ratios (LLRs). The bit allocation is based on maximizing the mutual information between the input and output LLRs of the quantizer. This scheme causes a negligible (0.1 $<$ dB) performance loss when the code block length is N = 1024 and the number of information bits is K = 854. The decoder is implemented on 45nm ASIC technology using deeply-pipelined, unrolled hardware architecture with register balancing. The pipeline depth is kept at 40 clock cycles in ASIC by merging consecutive decoding stages implemented as combinational logic. The ASIC synthesis results show that SC-MJL decoder has 427 Gb/s throughput at 45nm technology. When we scale the implementation results to 7nm technology node, the throughput reaches 1 Tb/s with under 10 mm^2 chip area and 0.37 W power dissipation.**

***Index Terms*—Application specific integrated circuits, polar codes, terabits-per-second throughput, successive-cancellation decoding, majority-logic decoding, quantization**

I. Introduction

It is foreseen that within the next decade there will be demand for forward error correction (FEC) codes operating at Terabit-per-second (Tb/s) data rates for certain beyond-5G applications [1]. The demand for higher data rates can be seen by looking at the recent standardization activities. For wired connections, the IEEE 802.3ba Ethernet standard specifies 100 Gigabit-per-second (Gb/s) throughput over optical media [2]. In the wireless domain, the IEEE 802.15.3d standard ratified in 2017 defines a 100 Gb/s system using frequencies in the 252 - 322 Gigahertz (GHz) range [3]. The 2018 Ethernet Roadmap [4] foresees demand for Terabit-per-second (Tb/s) data rates for 2020 and beyond.

This paper studies the feasibility of achieving Tb/s data rates using polar codes. Part of the challenge of reaching Tb/s with polar codes is generic, common to all FEC schemes, and stems from limitations of the VLSI technology. A second set of difficulties are specific to polar codes, arising from the inherently sequential nature of the decoding of polar codes. We investigate both aspects of the challenge and propose solutions. We begin by giving an overview of the problem.

A. VLSI technology challenges for Tb/s FEC

For several decades, FEC data rates could be increased by advances in VLSI technology, in accordance with technology forecasts known as Moore's law and Dennard's scaling law [5]. Although transistor dimensions still continue to shrink in accordance with the Moore's law, transistor switching speeds (clock frequencies) cannot keep increasing due to power density constraints [6]. With the clock frequency reaching practical limits at around 1-5 GHz, implementing Tb/s FEC schemes in VLSI requires highly parallel and deeply pipelined implementation architectures. This in turn makes implementation issues, such as chip area and power density, to move to the forefront as major design parameters, along with traditional measures of FEC performance such as coding gain or gap-to-capacity. The design and implementation of Tb/s FEC codes involves a complex tradeoff between a large set of parameters.

In the Tb/s regime, I/O bottleneck and excessive memory usage emerge as two important generic problems. To see the scale of the I/O problem, consider as an example of a FEC system with a coding rate $R = K/N$ carrying K bits of information in code blocks of length N bits. Suppose the receiver front-end provides the decoder with soft information in the form of log-likelihood ratios (LLRs) at a rate of $\frac{\gamma}{R}$ LLRs-per-second with a precision of Q bits-per-LLR, where γ is the throughput in b/s. Let f_c be the clock frequency for the interface between the decoder and the receiver front-end and P is the number of spatially parallel decoders connected to the front-end. The interconnect bus width at this interface will then have to contain at least

$$W = \frac{\gamma}{f_c}\frac{Q}{R} = \frac{(NRPf_c)}{f_c}\frac{Q}{R} = NPQ \qquad (1)$$

978-1-5386-9358-2/19/$31.00 ©2019 IEEE

wires assuming that each wire in the bus carries binary signals. For example, with $\gamma = 1$ Tb/s, $f_c = 1$ GHz, $Q = 3$ bits, and $R = 1/2$, we have $W = 6000$. For the given W, a set of (N,P) values can be $(512, 4)$, $(1024, 2)$, $(2048, 1)$. This example clearly shows the difficulty of increasing γ while f_c is held fixed. In order to alleviate the I/O bottleneck, we consider in this paper a relatively high rate code with $R = 5/6$, and try to minimize Q by using an quantization scheme that is information-theoretically as efficient as possible, as suggested in [7].

In order to illustrate the memory problem mentioned above, suppose that the decoder in the preceding example is implemented in a deeply-pipelined fashion, using D pipeline stages, where D is the decoder latency measured in number of clock cycles. Thus, we are assuming that there are PD codewords inside the decoder at any moment, the codewords spread over the successive stages of decoding in an assembly-line fashion. The memory requirement for this architecture may be estimated as

$$M_{\text{Req}} = \frac{\gamma}{f_c}\frac{D\overline{Q}}{R} = \frac{NRPf_c}{f_c}\frac{D\overline{Q}}{R} = NPD\overline{Q}, \qquad (2)$$

where $\overline{Q}$ is the average number of bits per LLR value inside the decoder. The product NPD emerges a significant parameter for controlling M_{Req}. The number of pipeline stages D is related to N in a manner that is specific to the code family and decoder type within that code family. For example, for the basic successive cancellation (SC) decoding method for polar codes, the smallest value of D is $2N - 2$ (achieved by using a fully parallel implementation), making the product NP quadratic in N. Such a quadratic growth in M_{Req} as a function of N severely limits the length of codes that can be used, leading to inferior coding gains. In this paper we seek a remedy to this problem by introducing a hybrid decoding algorithm that has a lower latency D than the SC decoding algorithm. The hybrid algorithm combines SC decoding with Majority Logic (MJL) decoding, as discussed below. As a further measure to reduce M_{Req}, we implement a variable-length quantization scheme inside the decoder so as to minimize $\overline{Q}$ for a given performance.

B. Relation to previous work

Polar codes were introduced in [8]. Polar codes are closely related to Reed-Muller (RM) codes [9], [10]. Many existing decoding algorithms for polar codes were originally devised for RM codes [11], [12]. This is true for the two decoding algorithms of interest in this paper, namely, SC decoding and MJL decoding. In fact, MJL decoding was the original decoding method for RM codes [10]. The distinctive feature of MJL decoding is its inherently parallel nature. The SC decoding method provides better coding gain at the expense of being serial in nature (increased latency). In this paper we combine the best features of SC decoding and MJL decoding. We use a soft-decision version of MJL decoding [13], [14]. The implementation presented below takes advantage of specific techniques for speeding up the SC decoder. These include methods to recognize specific constituent codes of the given polar code and decodes them quickly as described in [15], [16], [17], [18], [19], and [20].

A hybrid SC-MJL decoder implementation for polar codes was reported in [21]. That design relied on using combinational logic and aimed to provide a flexible architecture that could operate at various different coding rates. Unlike [21], here we focus on throughput only and use a fully unrolled and pipelined SC-MJL architecture to decode particular code segments faster. Similar to [16] and [17], the repetition (REP) and single parity-check (SPC) code segments are decoded by MAP [22] and Wagner [23] decoders respectively.

The outline of this paper is as follows. Section II gives a short review of polar coding and introduces the SC-MJL decoding with adaptive quantization. Section III presents the unrolled SC-MJL decoder architecture. Section IV presents the communication performance and ASIC implementation results of the SC-MJL decoder. Finally, Section V summarizes the main results with a brief conclusion.

II. Polar Codes and SC-MJL Decoding

This section starts with a short review of polar codes. Then, in Section II-B, the proposed SC-MJL decoding algorithm is introduced. Finally, in Section II-C, the adaptive quantization scheme used in this paper is presented.

A. Review of polar codes

Polar codes are a class of linear codes. Here, we consider only polar codes over the binary field $\mathbb{F}_2$. For every $n \geq 1$, there exists such a code with block length $N = 2^n$ and a transform matrix $G_N = G^{\otimes n}$ where $G^{\otimes n}$ is the n^{th} Kronecker power of a *kernel* matrix $G = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. In polar coding, the user data d_1^K is first embedded in a transform input vector u_1^N and the codeword is obtained as $x_1^N = u_1^N G_N$. A set $\mathcal{A}$ indicates which coordinates of u_1^N carries the data d_1^K. We write $u_{\mathcal{A}}$ to denote the data-carrying part of u_1^N. The remaining part of u_1^N is denoted $u_{\mathcal{A}^c}$ and is frozen to zero. We write $u_{\mathcal{A}} = d_1^K$ and $u_{\mathcal{A}^c} = 0$ to indicate the composition of the transform input u_1^N. For a description of the details of polar coding, we refer to [8].

B. The proposed SC-MJL decoding

The proposed SC-MJL decoding is given in Algorithm 1. Initially, the recursive block length parameter $M = N$ and ℓ_1^N is the channel log-likelihood ratio (LLR) vector with

$$\ell_i = \log\left(\frac{W(y_i|x_i=0)}{W(y_i|x_i=1)}\right),$$

where $W(y|x)$ is the channel transition probability density function. v_1^N is an indicator vector of the frozen coordinates defined as

$$v_i = \begin{cases} 1, & \text{if } i \in \mathcal{A}^c \\ 0, & \text{if } i \in \mathcal{A}. \end{cases}$$

The building blocks of the decoder are f, g and d functions. The function $\mathrm{f}(\ell, \ell')$ for any two LLR values ℓ and ℓ' is defined as

$$\mathrm{f}(\ell, \ell') = 2\tanh^{-1}(\tanh(\frac{\ell}{2})\tanh(\frac{\ell'}{2})),$$

which can be approximated [24] as

$$\mathrm{f}(\ell, \ell') \approx \mathrm{sgn}(\ell\ell')\min(|\ell|, |\ell'|). \tag{3}$$

The function $\mathrm{g}(\ell, \ell', \alpha)$ for any ℓ and ℓ' and any $\alpha \in \{0, 1\}$ is defined as

$$\mathrm{g}(\ell, \ell', \alpha) = (1-2\alpha)\ell + \ell'. \tag{4}$$

The function $\mathrm{d}(\ell, v)$ for any ℓ and frozen bit indicator v is defined as

$$\mathrm{d}(\ell, v) = \begin{cases} 0, & \text{if } v = 1 \\ 0, & \text{if } v = 0 \text{ and } \ell \geq 0 \\ 1, & \text{if } v = 0 \text{ and } \ell < 0. \end{cases} \tag{5}$$

Algorithm 1 combines SC decoder with certain shortcuts such as MJL decoding, Wagner decoding, etc. For details of SC decoding we refer to [8], and to [13] for MJL decoding. A precise statement of the MJL decoder as used here is given as Algorithm 2 with a generic block length N_{MJL}. The algorithm has $\log N_{\text{MJL}} + 1$ stages. For the i^{th} stage, the MJL algorithm decodes $\binom{\log M}{i}$ number of bits in parallel. For each bit, the algorithm calculates a final LLR value ℓ_j using the given f (3) and g (4) functions. After all $\hat{x}_1^M$ bits are decoded, the encoded $\hat{u}_1^M$ sequence is computed by using the bit-reversal permutation matrix B_M and the generator matrix G_M [8].

The flowchart representation of Algorithm 1 is shown in Fig. 1. The decoding complexities of Wagner and MAP decoders are upper bounded by N_{LIM} parameter, which denotes the maximum decodable block length in a single time step. When M is equal to N_{MJL}, the MJL decoding algorithm is used. In other case, the f (3) and g (4) functions divide the length-M polar code into two length-M/2 polar code branches until one of the special code segments appears. Both functions are applied element-wise to odd $\ell_{1,\text{odd}}^M$ and even $\ell_{1,\text{even}}^M$ elements of ℓ_1^M vector. Moreover, the partial update logic (PSUL) calculates the systematic decision output of the decoder when $M = N$. It is represented with a set of XOR ($\oplus$) operations. When $M < N$, PSUL calculates

Algorithm 1: SC-MJL

Inputs : ℓ_1^M, v_1^M, M **Output:** $\hat{u}_1^M$
if $v_1^M = 1$ **then** // R = 0
  $\hat{u}_1^M = \mathrm{d}(\ell_1^M, v_1^M = 1) = 0$
else if $v_1^M = 0$ **then** // R = 1
  $\hat{u}_1^M = \mathrm{d}(\ell_1^M, v_1^M = 0)$
else if $M \leq N_{\text{LIM}}$ **and** $v_1 = 1$ **and** $v_2^M = 0$ **then**
  $\hat{u}_1^M = \mathrm{d}(\ell_1^M, v_1^M = 0)$ // Wagner dec.
  $p = \mathrm{mod}(\sum_{i=1}^M \hat{u}_i, 2)$ // of R = (M-1)/M
  $r = \mathrm{argmin}(|\ \ell_1^M\ |)$
  $\hat{u}_r = \hat{u}_r \oplus p$
else if $M \leq N_{\text{LIM}}$ **and** $v_1^{M-1} = 1$ **and** $v_M = 0$ **then** // MAP decoding of R = 1/M
  $\hat{u}_1^M = \mathrm{d}(\sum_{i=1}^M \ell_i, v = 0)$ // Eq. (5)
else if $M = N_{\text{MJL}}$ **then** // MJL ∀ R
  $\hat{u}_1^M = \text{MJL}(\ell_1^M, v_1^M, N_{\text{MJL}})$
else // SC ∀ R
  $l_1^{M/2} = \mathrm{f}(\ell_{1,\text{odd}}^M, \ell_{1,\text{even}}^M)$ // Eq. (3)
  $\hat{z}_1^{M/2} = \text{SC-MJL}(l_1^{M/2}, v_{1,\text{odd}}^M, \frac{M}{2})$
  $r_1^{M/2} = \mathrm{g}(\ell_{1,\text{odd}}^M, \ell_{1,\text{even}}^M, \hat{z}_1^{M/2})$ // Eq. (4)
  $\hat{x}_1^{M/2} = \text{SC-MJL}(r_1^{M/2}, v_{1,\text{even}}^M, \frac{M}{2})$
  $\hat{u}_{1,\text{odd}}^M = \hat{z}_1^{M/2} \oplus \hat{x}_1^{M/2}$ // PSUL
  $\hat{u}_{1,\text{even}}^M = \hat{x}_1^{M/2}$
return $\hat{u}_1^M$

Algorithm 2: MJL

Inputs : ℓ_1^M, v_1^M, N_{MJL} **Output:** $\hat{u}_1^M$
Set $M = N_{\text{MJL}}$ and $c = 0$ // dec. counter
for i = 0, 1, ..., $\log M$ **do** // serial
  r = find rows ($\sum_{\text{columns}} G_M = 2^i$)
  for j = 1, 2, ..., $\binom{\log M}{i}$ **do** // parallel
    c = c + 1
    Calculate ℓ_j using f (3) and g (4) functions for only the r[th] rows
    $\hat{x}_{r(j)} = \mathrm{d}(\ell_j, v_c)$ // Eq. (5)
$\hat{u}_1^M = \hat{x}_1^M\ B_M\ G_M$
return $\hat{u}_1^M$

the feedback $\hat{z}_1^{M/2}$ for the input of g functions. At the end of PSUL, M can increase up to $M = 2^i M$ for $i \in \{0, 1, ..., \log N - \log M\}$. The estimated user data $\hat{d}_1^K$ is extracted from the estimated transform vector $\hat{u}_1^N$ at the end of decoding operation.

C. Adaptive quantization of the LLRs

The chip area of the SC decoder is dominated by the memory and the register chains in the deeply-pipelined architecture [16]. Implementation practice shows that using 5 or 6-bit precision for each LLR value causes tolerable performance loss [25]. We propose to reduce LLR precision even further (1-5 bits range of LLRs)

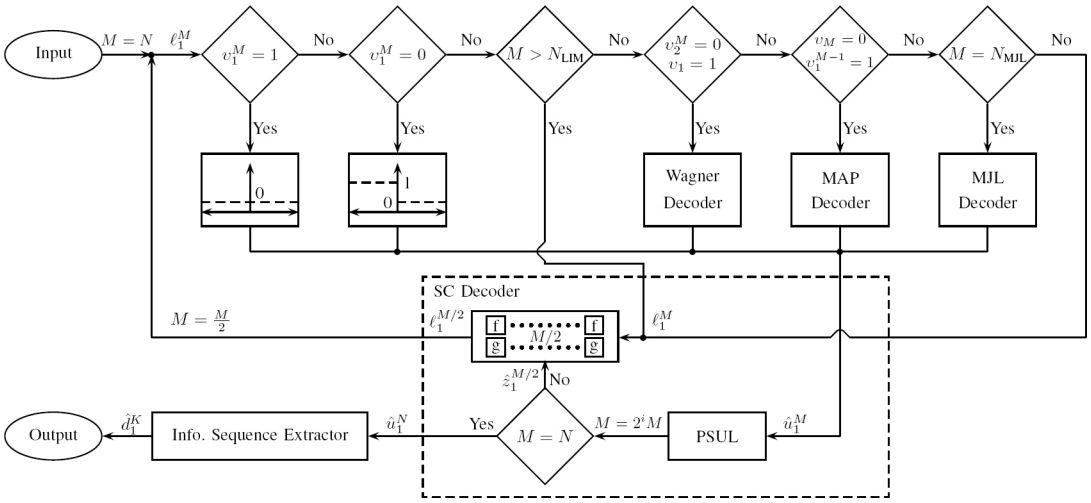

Fig. 1: The recursive structure of SC-MJL decoding algorithm where N is the code block length, K is the number of information bits, v_1^M is the indicator vector of the frozen coordinates with variable constituent block length M, N_{MJL} is the block length of MJL decoder and N_{LIM} is the maximum block length of Wagner and MAP decoders.

using an adaptive quantization technique. The bit allocation is based on maximizing the mutual information between input and output LLRs of the quantizer. Unlike using lookup tables [26], here we use the regular f and g functions with custom input data width. The data width or, in other words, the number of required quantization bits is optimized using input LLR distribution of each constituent polar code. For example, a rate-1 polar code segment with an arbitrary block length can be represented with one bit (the sign bit). Since polarization takes place, using large number of bits is not necessary for the polarized code segments. In this way, the LLRs located on those paths can have adaptive quantization levels.

Applying adaptive quantization to (1024,854) polar code, the internal LLR bit precision is shown in Fig. 2. The number of quantization bits are illustrated on each line. For example, the second half of the (1024,854) polar code uses one less quantization bits by dropping the redundant least significant bit. The adaptive quantization method has a significant impact on reducing the chip area as well as the power dissipation of the SC-MJL decoder as shown in Section IV-B.

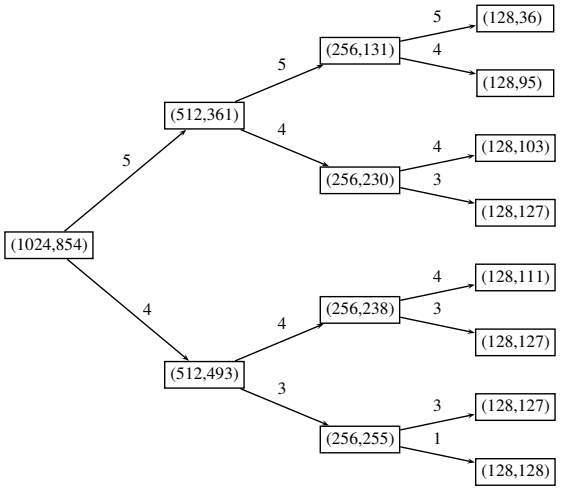

Fig. 2: Adaptive quantization of the constituent codes of SC-MJL(1024,854) for $128 \leq M \leq 1024$. The number of quantization bits are written on the lines.

III. Unrolled SC-MJL decoder architecture

We propose unrolled and deeply pipelined SC-MJL decoder architecture with fully-parallel processing units. We take advantage of bit-reversal decoding to operate on neighboring LLRs. The SC decoder, denoted as $\text{SC}(N, K)$, consists of two sub-decoders which have the same block length $\frac{N}{2}$ with a different payload $K_i = \frac{N}{2} R_i$. In general, $\text{SC}(N, K)$ is decoded in four steps: f, $\text{SC}(\frac{N}{2}, K_1)$, g and $\text{SC}(\frac{N}{2}, K_2)$. As a small example, the architecture of $\text{SC}(16, 9)$ is shown in Fig. 3. The ℓ_1^{16} LLRs at the input with $16 \times Q$ bits are stored during the processing duration of f function plus $\text{SC}(8, 2)$ decoder (denoted as $\mathcal{L}(\text{SC}_1)$) until $\hat{z}_1^8$ becomes ready at the input of g. Likewise, $\hat{z}_1^8$ is stored until $\hat{x}_1^8$ is ready.

The proposed SC-MJL decoder architecture for $N = 16$ and $K = 9$ is shown in Fig. 4. First, the adaptive quantization block (abbreviated as Adp. Q.) reduces the input LLR quantization from Q to Q' bits. Then, f function, $\text{MJL}(8, 2)$ decoder, g function, and $\text{Wagner}(8, 7)$ decoder are activated consecutively. When $\hat{z}_1^8$ and $\hat{x}_1^8$ are ready, PSUL calculates the systematic output $\hat{u}_1^{16}$. Each decoding operation takes one time step except PSUL, which performs combinational XOR operations. Therefore, the total latency of $\text{SC-MJL}(16, 9)$ is 4 time steps, which is considerably smaller than 30 time steps as in the $\text{SC}(16, 9)$ decoder. Furthermore, the $\text{MJL}(8, 2)$

decoder architecture is shown in Fig. 5. It utilizes nine adders, four f functions, two d functions, one g function and one XOR gate such that each f function contains a comparator and an XOR gate and each g function has two adders and one multiplexer.

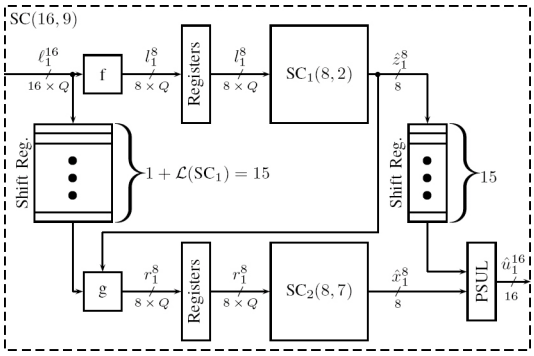

Fig. 3: SC(16,9) architecture.

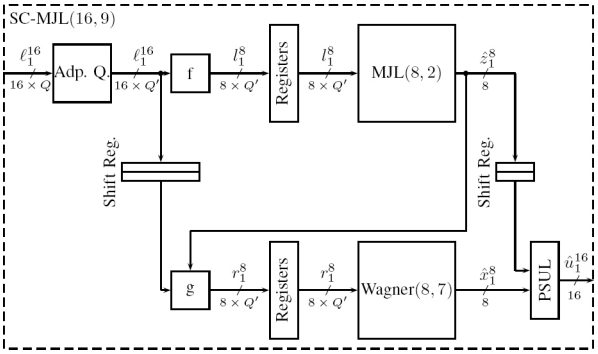

Fig. 4: Proposed SC-MJL(16,9) architecture.

A. Register Balancing

The proposed SC-MJL decoder simultaneously processes different codewords in a sequence of decoding stages. The complex operations in the sequential stages and strict setup/hold time requirements may cause a throughput bottleneck in the decoder. The critical path, where the worst negative slack (WNS) is minimum, may limit the frequency and reduce the throughput. In order to avoid this, register balancing is performed in HDL level to merge the consecutive short paths by removing the registers in between those paths. The locations of remaining registers are chosen according to combinational delay of the merged stages. Applying register balancing enables SC-MJL decoder to perform multiple calculations within a clock cycle. It reduces both the latency and the memory usage of the decoder. For example, the latency of SC-MJL(16,9) decoder reduces by two clock cycles when the given registers in Fig. 4 are removed without violating the WNS.

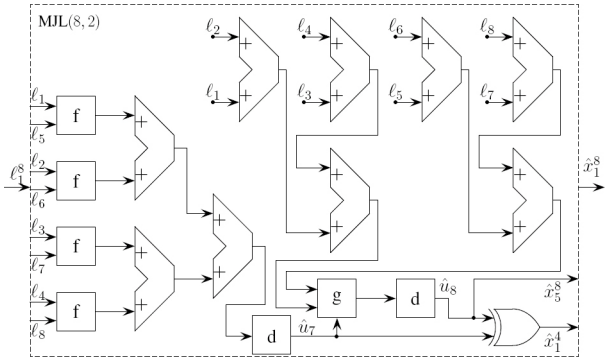

Fig. 5: MJL(8,2) decoder architecture.

IV. Implementation Study

The SC-MJL(1024,854) decoder is implemented on 45nm ASIC using the general purpose (GP) standard cell library (tcbn45gsbwp12tbc). The nominal PVT values are 45nm, 1.2V and 25°C. The implementation parameters are $N_{\mathrm{LIM}} = 32$ and $N_{\mathrm{MJL}} = 8$. In this configuration, the number of shortcuts are: 13 MJL, 13 SPC, 5 REP, 16 Rate-1 and 3 Rate-0. In addition to that the clock gating method is employed for the available registers to reduce the power dissipation.

A. Performance Results

Extensive simulations have been performed to obtain the communication performance results of the SC-MJL decoding algorithm and the adaptive quantization method. The simulations have been carried out with an AWGN channel and BPSK modulation for the (1024,854) code. The performance of the SC-MJL decoding algorithm with a variable N_{MJL} is shown in Fig. 6. As N_{MJL} increases, the performance deteriorates progressively. It is observed that $N_{\mathrm{MJL}} = 8$ causes a tolerable loss.

The communication performance of the SC and the SC-MJL decoders are shown in Fig. 7. There is almost 0.1 dB performance difference between SC and SC-MJL decoder. An additional performance loss occurs when the adaptive quantization is used. Applying register balancing does not introduce an additional performance degradation. However, using fixed $Q = 4$ bits quantization for both channel and internal LLRs causes more than 0.3 dB performance loss.

B. ASIC Implementation Results

The ASIC post-synthesis results of SC(1024,854) and SC-MJL(1024,854) decoders are shown in Table I. The SC-MJL decoder dissipates 1.5 times less power than the benchmark SC decoder, while having a smaller area. The proposed adaptive quantization and register balancing architecture further reduces the power dissipation by 1.4 and 2.3 times, respectively. Due to register balancing architecture, both latency and pipeline depth of the decoder reduce to 40 clock cycles. Since the throughput results of given implementations are the same, the most

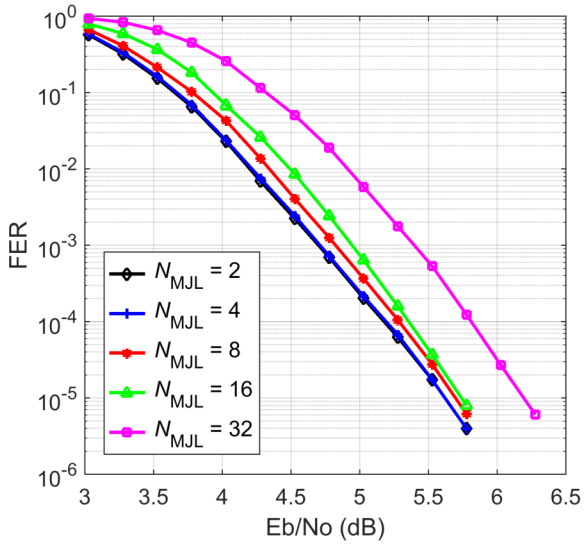

Fig. 6: Performance of (1024,854) polar code under SC-MJL decoding algorithm with $N_{\text{LIM}} = 32$ and a variable N_{MJL}.

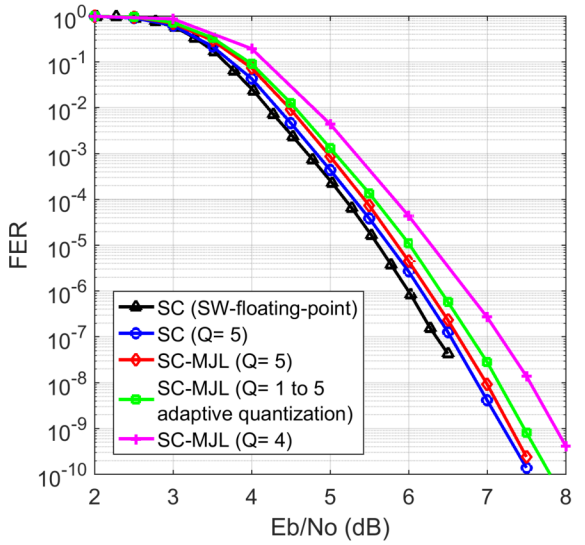

Fig. 7: The effect of LLR quantization on software and FPGA performance of (1024,854) polar code under SC and SC-MJL decoding with $N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$.

energy efficient implementation is the last one with 2.4 pJ/bit. The post-synthesis results are scaled from 45nm to 7nm technology using the conservative scaling formulas in [1]. In addition to the scaling, each implementation utilizes two parallel decoders, which operate at 585.5 MHz frequency as the expected 2.2 GHz frequency is scaled down by a factor of 3.7. Another parameter is the area scaling, which is a multiplier to the chip area to obtain a reasonable power density for a feasible cooling off the chip. The results show that the proposed

TABLE I: ASIC post-synthesis results of (1024,854) polar code SC-MJL decoder with $N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$ at 45nm technology node.

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing	x	x	x	✓
Throughput (Gb/s)			427	
Frequency (MHz)			500	
Area (mm^2)	9.8	8.3	6.6	2.4
Power (W)	4.6	3.1	2.3	1.0
Area Eff. (Gb/s/mm^2)	43.5	51.4	65.0	175.2
Pow. Den. (W/mm^2)	0.47	0.38	0.36	0.42
Energy Eff. (pJ/bit)	10.9	7.3	5.5	2.4
Latency (μs)	0.31	0.25	0.25	0.08
Latency (Clock cyc.)	157	127	127	40

TABLE II: The expected ASIC post-synthesis results of (1024,854) polar code SC-MJL decoder with $N_{\text{MJL}} = 8$ and $N_{\text{LIM}} = 32$ at 7nm technology node. Each implementation consists of two identical spatially parallel polar decoders.

Decoding Algorithm	SC	SC-MJL	SC-MJL	SC-MJL
Quantization (bits)	6	6	5-to-1	5-to-1
Reg. Balancing	x	x	x	✓
Area Scaling	14.3	16.9	21.4	57.7
Throughput (Gb/s)			1000	
Frequency (MHz)			585.5	
Area (mm^2)			10	
Area Eff. (Gb/s/mm^2)			100	
Power (W)	1.69	1.14	0.85	0.37
Pow. Den. (W/mm^2)	0.17	0.11	0.09	0.04
Energy Eff. (pJ/bit)	1.69	1.14	0.85	0.37

implementation is expected to have 0.37 pJ/bit energy efficiency at 7nm while having 1 Tb/s throughput.

C. ASIC implementation comparison of SC-MJL with other high throughput polar decoders

The ASIC post-synthesis results of high throughput polar decoders are compared in Table III. Using the same scaling rule in [27] and [21], the normalized results show that the SC-MJL decoder is the most energy efficient decoder. Although it can operate at 1.5 lower frequency than the SC-Fast decoder, it has 3.2 times better area efficiency due to efficient merging of pipelined stages in the register balancing architecture.

V. Conclusion

In order to reach high throughput within the physical limits of the current VLSI technology, we proposed SC-MJL decoding algorithm with an adaptive quantization and register balancing architecture. Firstly, the SC-MJL decoder architecture reduces the pipelined depth of the SC algorithm by 1.2 times. In addition to that the proposed adaptive quantization scheme further reduces both computational and memory complexity of the SC-MJL decoder. The proposed decoding algorithm utilizes a deeply-pipelined and unrolled hardware architecture using combinational logic. In this architecture, the consecutive decoding stages are merged to further reduce

TABLE III: Comparison with the high throughput polar decoders.

Implementation	This work	[28]	[21]
Architecture	SC-MJL	SC-Fast	SC-Comb.
ASIC Technology	45nm	28nm	90nm
Supply Voltage (V)	1.2	1.0	1.3
Coded Throughput (Gb/s)	512	1275	2.6
Frequency (MHz)	500	1245	2.5
Latency (μs)	0.08	0.3	0.4*
Area (mm^2)	2.4	4.6	3.2
Power (W)	1.01	8.79	0.19
Converted to 28nm, 1.0 V using the scaling in [27], [21]			
Coded Throughput (Gb/s)	823	1275	8.2
Frequency (MHz)	804	1245	8.0*
Area (mm^2)	0.94[a]	4.63	0.31
Area Eff. (Gb/s/mm^2)	872	276	26
Power (W)	0.44[†]	8.79	0.04
Power Density (W/mm^2)	0.46	1.89*	0.12*
Energy Eff. (pJ/bit)	0.5[‡]	6.9	4.6

*Not presented in the paper, calculated from the presented results
[a]Normalized factor for area is $0.39 = (28/45)^2$
[†]Norm. factor for power is $0.43 = (28/45)(1.0/1.2)^2$
[‡]Norm. factor for energy eff. is $0.27 = (28/45)^2(1.0/1.2)^2$

the pipeline depth of the decoder to 40 clock cycles. The ASIC synthesis results show that the SC-MJL decoder has 427 Gb/s throughput at 45nm technology. When the results are scaled to 7nm, the throughput reaches Tb/s under 10 mm^2 chip area with 0.37 W power dissipation. Finally, the comparison with other high throughput implementations shows that the proposed SC-MJL decoder has a remarkable area and energy efficiency.

ACKNOWLEDGMENT

The work has been supported by EPIC project funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 760150.

REFERENCES

[1] "EPIC - Enabling practical wireless Tb/s communications with next generation channel coding." [Online]. Available: https://epic-h2020.eu/results.

[2] D. Law, D. Dove, J. D'Ambrosia, M. Hajduczenia, M. Laubach, and S. Carlson, "Evolution of ethernet standards in the IEEE 802.3 working group," vol. 51, pp. 88–96, August 2013.

[3] "IEEE 802.15.3d-2017 - IEEE standard for high data rate wireless multi-media networks amendment 2: 100 Gb/s wireless switched point-to-point physical layer." [Online]. Available: https://standards.ieee.org/findstds/standard/802.15.3d-2017.html.

[4] "Ethernet roadmap 2018." [Online]. Available: https://ethernetalliance.org/wp-content/uploads/2016/03/EthernetRoadmap-2018-Side1-1600x1200.jpg.

[5] B. Nikolic, "Design in the power-limited scaling regime," vol. 55, pp. 71–83, January 2008.

[6] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *2011 38th Annual International Symposium on Computer Architecture (ISCA)*, pp. 365–376, June 2011.

[7] A. Winkelbauer and G. Matz, "On quantization of log-likelihood ratios for maximum mutual information," in *2015 IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 316–320, June 2015.

[8] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, pp. 3051–3073, July 2009.

[9] D. E. Muller, "Application of boolean algebra to switching circuit design and to error detection," vol. EC-3, pp. 6–12, September 1954.

[10] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *Transactions of the IRE Professional Group on Information Theory*, vol. 4, pp. 38–49, September 1954.

[11] E. Arikan, "A survey of reed-muller codes from polar coding perspective," in *2010 IEEE Information Theory Workshop (ITW)*, pp. 1–5, IEEE, June 2010. 00008.

[12] I. Dumer, "On decoding algorithms for polar codes," March 2017. [Online]. Available: http://arxiv.org/abs/1703.05307.

[13] V. D. Kolesnik, "Probabilistic decoding of majority codes," *Probl. Peredachi Inform.*, vol. 7, pp. 3–12, 1971.

[14] I. Dumer and R. Krichevskiy, "Soft-decision majority decoding of reed-muller codes," *IEEE Transactions on Information Theory*, vol. 46, pp. 258–264, Jan 2000.

[15] A. Alamdar-Yazdi and F. R. Kschischang, "A simplified successive-cancellation decoder for polar codes," *IEEE Communications Letters*, vol. 15, pp. 1378–1380, December 2011.

[16] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, "Fast polar decoders: Algorithm and implementation," *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 946–957, May 2014.

[17] M. Hanif and M. Ardakani, "Fast successive-cancellation decoding of polar codes: Identification and decoding of new nodes," *IEEE Communications Letters*, vol. 21, pp. 2360–2363, Nov 2017.

[18] P. Giard, G. Sarkis, C. Thibeault, and W. J. Gross, "Multi-mode unrolled architectures for polar decoders," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, pp. 1443–1453, Sept 2016.

[19] B. Yuan and K. K. Parhi, "Low-latency successive-cancellation polar decoder architectures using 2-bit decoding," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, pp. 1241–1254, April 2014.

[20] B. Yuan and K. K. Parhi, "Low-latency successive-cancellation list decoders for polar codes with multibit decision," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, pp. 2268–2280, Oct 2015.

[21] O. Dizdar, *High Throughput Decoding Methods and Architectures for Polar Codes with High Energy-Efficiency and Low Latency*. PhD thesis, Bilkent University, 2017.

[22] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. on Comm.*, vol. 47, pp. 673–680, May. 1999.

[23] R. Silverman and M. Balser, "Coding for constant-data-rate systems," *Transactions of the IRE Professional Group on Information Theory*, vol. 4, pp. 50–63, September 1954.

[24] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. on Comm.*, vol. 47, pp. 673–680, May. 1999.

[25] C. Leroux, A. J. Raymond, G. Sarkis, and W. J. Gross, "A semi-parallel successive-cancellation decoder for polar codes," *IEEE Transactions on Signal Processing*, vol. 61, pp. 289–299, Jan 2013.

[26] S. A. A. Shah, M. Stark, and G. Bauch, "Design of quantized decoders for polar codes using the information bottleneck method," in *SCC 2019; 12th International ITG Conference on Systems, Communications and Coding*, pp. 1–6, Feb 2019.

[27] C. Wong and H. Chang, "Reconfigurable turbo decoder with parallel architecture for 3gpp lte system," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, pp. 566–570, July 2010.

[28] P. Giard, C. Thibeault, and W. J. Gross, *High-Speed Decoders for Polar Codes*. Springer, 2017. pp. 66–67.