When Channel Coding Hits the Implementation Wall

Claus Kestel, Matthias Herrmann, Norbert Wehn
Microelectronic Systems Design Research Group, TU Kaiserslautern
67663 Kaiserslautern, Germany
{kestel, herrmann, wehn}@eit.uni-kl.de

***Abstract*—The continuous demands for higher throughput, higher spectral efficiency, lower latencies, lower power and large scalability in communication systems impose large challenges on the baseband signal processing. In the future, throughput requirements far beyond $100\,\mathrm{Gbit/s}$ are expected, which is much higher than the tens of $\mathrm{Gbit/s}$ targeted in the 5G standardization. At the same time, advances in silicon technology due to shrinking feature sizes and increased performance parameters alone will not provide the necessary gain, especially in energy efficiency for wireless transceivers, which have tightly constrained power and energy budgets. The focus of this paper lies on channel coding, which is a major source of complexity in digital baseband processing. We will highlight implementation challenges for the most advanced channel coding techniques, i.e. Turbo codes, Low Density Parity Check (LDPC) codes and Polar codes and present decoder architectures for all three code classes that are designed for highest throughput.**

I. Introduction

Mobile communication plays a central role in our information society and is a key enabler for the connected world. The first generation of mobile communication systems has shifted the communication from landline to handheld devices, followed by the third (3G) and fourth generations (4G) that marked the advent of the mobile internet. We have seen a tremendous increase in data rates over the different generations, e.g., the Global System for Mobile Communications (GSM) featured about $10\,\mathrm{kbps}$, the Universal Mobile Telecommunications System (UMTS) about $2\,\mathrm{Mbit/s}$, and Long Term Evolution Advanced (LTE-A) about $1\,\mathrm{Gbit/s}$. The newest standard, 5G, features data rates $>10\,\mathrm{Gbit/s}$. Beyond 5G, data rates towards $1\,\mathrm{Tbit/s}$ are expected [1]. However, increasing data rates are not the only driver. With the advent of 5G, wireless communication is finding its way into a variety of applications with largely diverse requirements. Massive Internet of Things (IoT) demands for extremely low power, massive content applications (e.g. video streaming, cloud office) call for very high throughput and the use of wireless communication in control applications (e.g. autonomous driving, remote control, tactile internet) requires response times in the order of $1\,\mathrm{ms}$. Thus, communication has to support a huge throughput-latency range (5 orders of magnitude in throughput, 3 orders of magnitude in latency) and a large diversity of applications with different Bit Error Rate (BER)/Frame Error Rate (FER) requirements [9].

The tremendous improvement in mobile communication has to be considered in the context of the progress in the microelectronic industry, which started with the invention of the transistor in the late 1940s [23], coincidentally at the same time when Shannon published his famous article "A mathematical theory of communication" [22]. In the following decades, the semiconductor industry achieved an exponential increase in the number of transistors on a single chip, known as Moore's law [17], which is a key driver of our information society. In today's $14\,\mathrm{nm}$ technology, 38 million transistors can be integrated on $1\,\mathrm{mm}^2$ of silicon.

For many decades, improvement in silicon process technology provided better performance, lower cost per gate, higher integration density and lower power consumption. However, we have reached a point where Moore's law is slowing down for the following reasons [3], [12], [2]:

Cost of technology: The cost per wafer from $28\,\mathrm{nm}$ technology to $7\,\mathrm{nm}$ has more than doubled. At the same time, the area density (yield not considered) increased by $6\times$. Thus, the cost per mm^2 is still decreasing but not with the same pace as in the past.

Design cost: The average IC design cost in $14\,\mathrm{nm}$ is about 80 million \$, compared to 30 million \$ for a $28\,\mathrm{nm}$ planar device, according to Gartner. It will cost 270 million \$ to design a chip in $7\,\mathrm{nm}$. This creates a situation where less and less designs each year have enough volume to be able to support the cost of the design, since the cost must be amortized over the number of parts produced. Thus, the use of the most advanced technology nodes is no longer economically viable for all products.

Decreasing performance gain: Over the last 10 years, the semiconductor industry has succeeded in doubling the transistor density every 2 to 2.4 years. However, the performance gains have been much smaller, such that less than 40% performance improvement of today's processors come from semiconductor technology. Until 2033, performance scaling across 7 technology nodes (spanning from today to 2033) exhibits only $1.7\times$ frequency improvement from semiconductor technology.

Power density/dark silicon: Since the power per gate decreases slower than the transistor density increases, the power per mm^2 continuously increases. In consequence, if a chip has to be operated with the same power density over different technology nodes, i.e. with the same Thermal Design Power (TDP), not all transistors can switch at the same time (named "dark silicon"), or the frequency has to be reduced. As dis-

cussed before, frequency at nominal supply voltage is expected to improve by $1.7\times$ until 2033 (e.g. for high performance circuits from today's $2.5\,\mathrm{GHz}$ to $4.2\,\mathrm{GHz}$), whereas power density increases at the same time by $8\times$. As a result, frequency has to be reduced nearly by the same factor (e.g. will stall to $0.5\,\mathrm{GHz}$ for high performance circuits), if the same chip is operated at constant power density. Thus for the future, reducing the power consumption of a transistor becomes more important than improving its performance. After the "geometrical scaling" era, which was followed by the "equivalent transistor scaling" era, we are entering the third era of scaling, named "power scaling".

Delay due to interconnect: Global wiring quadratically increases with its length. In $14\,\mathrm{nm}$ technology, the interconnect delay for $1\,\mathrm{mm}$ wire is about $400\,\mathrm{ps}$, which is one to two orders of magnitude slower than the speed of light in such material. In the future, interconnect resistance will exponentially increase for tight metal pitches that are used in data paths. Performance scaling across 7 technology nodes, spanning from today to 2033, exhibits a 9% technology node-to-node performance improvement for data paths without wires. However, if wires are considered, this improvement turns into a 10% node-to-node penalty for data paths loaded with tight pitch metal routing, i.e. the delay will increase.

Variability and reliability: With the continuously decreasing feature sizes, variations in the device parameters largely increase, resulting in large circuit performance/power fluctuations. In addition, the reliability of the circuits decreases due to, e.g., aging effects, hot carrier injection and soft errors.

Channel coding, or Forward Error Correction (FEC), is a core technology component in any digital baseband and is a major source of power consumption, silicon area and largely contributes to the overall latency and throughput limitations. In [9] it is shown that channel decoding is the most computationally intensive part in a 5G terminal baseband chip. In this paper we consider the most advanced channel codes, i.e. Turbo codes, LDPC codes and Polar codes [7]. These codes, which are part of many communication standards, provide excellent communications performance but imply huge implementation challenges.

Let us consider the progress in channel coding in relation to the progress of microelectronics. As a case study, we consider two Turbo code decoders. Both decoders were designed with the same design methodology and have a very similar architecture that exploits spatial parallelism and processes several sub-blocks on corresponding Maximum a Posteriori (MAP) decoders in parallel. The first decoder is a fully UMTS compliant Turbo code decoder in $180\,\mathrm{nm}$ technology. Under worst case Process, Voltage and Temperature (PVT) conditions a maximum frequency of $166\,\mathrm{Mhz}$ is achieved, which results in a throughput of $71\,\mathrm{Mbit/s}$ at 6 decoding iterations. The total area is $30\,\mathrm{mm}^2$ [14]. The second decoder is a fully LTE compliant Turbo code decoder in $65\,\mathrm{nm}$ technology, achieving a maximum frequency of $450\,\mathrm{MHz}$ under worst case PVT conditions, resulting in a throughput of $2.15\,\mathrm{Gbit/s}$ at 6 decoding iterations and $7.7\,\mathrm{mm}^2$ area [13]. Three semiconductor technology nodes are between $180\,\mathrm{nm}$ and $65\,\mathrm{nm}$ technology. We observe a throughput increase by $30\times$ although the improvement of frequency, which is limited by the critical path inside the MAP decoder, is only $3\times$. The improvement in area efficiency (throughput/area) is $118\times$. Progress in microelectronics contributed to a largely increased area efficiency but much less to the frequency increase. The throughput increase mainly originates from code design, i.e. conflict-free Turbo code interleavers that enable efficient implementation and a high degree of parallelism, advanced algorithmic and architectural features, like next-iteration initialization, optimized radix-4 kernel, re-computation, advanced normalization to reduce internal bit widths, etc.

We see that microelectronics cannot keep pace with the increased requirements coming from communication systems. Thus, the design of communication systems is no longer just a matter of spectral efficiency or BER/FER. When it comes to implementation, channel coding requires a cross-layer approach covering information theory, algorithms, parallel hardware architectures and semiconductor technology to achieve excellent communications performance, high throughput, low latency, low power consumption, and high energy and area efficiency [21].

$100\,\mathrm{mm}^2$ area is a feasible size for a baseband processor chip [3]. Let us assume that 10% of this area is allocated to the FEC Intellectual Property (IP). On $10\,\mathrm{mm}^2$ about 400 million transistors can be integrated. Due to the fact that the power envelope for future communication systems cannot be largely increased, designs are more and more power constrained. Thus, a $1\,\mathrm{W}$ power envelope is feasible for the FEC IP, resulting in a power density of about $100\,\mathrm{mW/mm}^2$. The maximum frequency is upper bounded to $1\,\mathrm{GHz}$ due to power and design issues. If the power is constrained, increasing the throughput requires decreasing the energy per decoded bit by the same order. For example, for $1\,\mathrm{Tbit/s}$ data throughput, 1000 information bits have to be decoded in each clock cycle of length $1\,\mathrm{ns}$ with an energy budget of only $1\,\mathrm{pJ}$ per decoded bit. For comparison, $1\,\mathrm{pJ}$ is the same order of magnitude as performing a $32\,\mathrm{bit}$ integer multiplication in $28\,\mathrm{nm}$ technology or one order of magnitude lower than accessing a single bit from a state-of-the-art external Dynamic Random Access Memory (DRAM). Due to the high design cost of a baseband chip in advanced semiconductor technology, a large volume is required. This means that the corresponding System-on-Chip (SoC) has to be used in various applications and the FEC IP must be very flexible to support various coding schemes, data rates, latency and BER/FER requirements. Adding flexibility to any architecture has a negative impact on throughput, energy efficiency and area, which is called flexibility/efficiency trade-off. For example, specialized hardware is 2 to 3 orders of magnitude more efficient than processor based solutions, that offer largest flexibility [4].

There are discrepancies between information theory objectives and efficient implementation objectives. Advanced channel codes like Turbo codes and LDPC codes combine irregularity and iterative/sequential decoding techniques to

achieve very good communications performance. On the implementation side, however, large locality, regularity and large parallelism are mandatory to obtain energy efficient, high-throughput architectures.

Large parallelism is a must for high throughput and low latency. Sub-functions of an algorithm that have no mutual data dependencies can easily be parallelized by spatial parallelism. This is, e.g., the case for the Belief Propagation (BP) algorithm for LDPC decoding in which all check nodes can be processed independently from each other. The same applies for the variable nodes. The situation is different for the MAP algorithm where the calculation of a specific trellis step depends on the result of the previous trellis step. This is a sequential behavior and the different trellis steps cannot be calculated in parallel. All iterative algorithms have this sequential behavior, since data dependencies exist between the different iterations. The part of an algorithm that cannot be parallelized due to data dependencies strongly limits the overall performance, which is known as Amdahl's Law [6]. Functional parallelism/pipelining is an efficient technique to speed up algorithms with data dependencies. We can "unroll" the iterations and insert buffers between the different pipeline stages. In this way, all iterations can be calculated in parallel but on different data sets. Spatial and functional parallelization are implementation techniques only, i.e. they are not changing the algorithmic behavior. We can also modify the decoding algorithm to enable parallelism. Let us again consider the MAP algorithm. The data dependencies in the forward/backward recursion can be broken up by exploiting the fact that the trellis has a finite memory due to the underlying constraint length of the code. This property enables the splitting of the trellis into sub-trellises that can be processed independently of each other. However, some acquisition steps are mandatory at the border of the sub-trellises to get the correct probabilities. The length of the sub-trellises and the corresponding acquisition length impact the communications performance.

We see that there are various possibilities to increase the throughput by parallel architectures. However, high throughput architectures and especially heavily pipelined architectures imply another challenge. A huge amount of data has to be stored inside the architecture since many blocks are processed in parallel. But storing and accessing data is costly in terms of energy. For example, reading/writing 8 bit in a medium-sized Static Random Access Memory (SRAM) in $28\,\mathrm{nm}$ technology costs in the order of $1\,\mathrm{pJ}$. At least one order of magnitude difference in energy exists between storing/accessing (SRAM or DRAM) and computing. Thus, re-computing can often be more energy efficient than storing and accessing. In heavily pipelined architectures, data is typically stored in registers. Storing 8 bits in registers costs one order of magnitude less energy and is about $3\times$ faster than storing in an SRAM. However, registers have to be driven by a clock tree. If a design contains many registers, the clock tree can become the dominant power consumer. As an example, we consider a heavily pipelined high-throughput Polar code decoder with successive cancelation decoding algorithm (see Section IV), which achieves a throughput of $636\,\mathrm{Gbit/s}$ at a frequency of $621\,\mathrm{MHz}$ under worst case PVT in a low V_t $28\,\mathrm{nm}$ Fully Depleted Silicon on Insulator (FD-SOI) technology for a 1024/512 code. The total area is about $3.12\,\mathrm{mm}^2$, half of which accounts for registers only. 47% of the total power consumption amounts for the clock, 24% amounts for the registers and only 29% for the combinatorial logic. Thus, optimizing storage is a major challenge to reduce power. Efficient quantization is a very powerful approach at algorithmic level since the bit width linearly contributes to the memory requirements.

As stated earlier, routing can also largely contribute to the latency and energy consumption. A signal has to travel at least $7\,\mathrm{mm}$ if it has to be transmitted from one corner to the other in our FEC IP block of size $10\,\mathrm{mm}^2$. With a frequency of $1\,\mathrm{Ghz}$, it will take 3 clock cycles, i.e. 2 pipeline stages have to be inserted in the wire in $14\,\mathrm{nm}$ technology. This largely increases the latency. Thus, a high locality in the architecture is important to minimize wiring.

Table I summarizes the implementation properties of the different code classes. In the following, we consider channel decoder architectures optimized for highest throughput. All designs were carried out in our research group, utilizing the same design methodology and same semiconductor technology, enabling fair comparison.

II. Turbo Code Decoding

Among the three code classes, Turbo codes are most challenging with respect to high-throughput decoding. The decoding structure consists of two component decoders connected through an interleaver/de-interleaver. These component decoders work cooperatively by exchanging extrinsic information in an iterative loop. State-of-the-art component decoders apply the MAP algorithm that calculates the log-likelihood probabilities for each bit in a block by a forward and a backward recursion on the trellis, respectively. Thus, decoding is inherently serial at component decoder and at Turbo code decoder level.

We can use some of the aforementioned techniques (e.g. unrolling and pipelining) at the algorithmic and the architectural level to parallelize the decoding at MAP and Turbo code decoder level. For a detailed overview and discussion of such a highly parallel decoder we refer to [24]. Figure 1 shows the corresponding layout of the decoder that achieves $102\,\mathrm{Gbit/s}$ information bit throughput for a block length of $128\,\mathrm{bit}$ on a low V_t $28\,\mathrm{nm}$ FD-SOI technology under worst case PVT conditions, running with $800\,\mathrm{MHz}$ and performing 4 decoding iterations. Different colors represent the eight different MAP decoders originating from the 4 unrolled iterations (each iteration requires two MAP decoders). Although this architecture achieves an information bit throughput of more than $100\,\mathrm{Gbit/s}$ for small block sizes, it suffers from limited flexibility in terms of block sizes and varying number of iterations, a large area and high power consumption.

TABLE I
OVERVIEW ON IMPLEMENTATION PROPERTIES

Code	Decoding algorithms	Parallel vs. serial	Locality	Compute kernels	Transfers vs. compute
Turbo code	MAP	serial/iterative	low (interleaver)	Add-Compare-select	compute dominated
LDPC code	Belief propagation	parallel/iterative	low (Tanner graph)	Min-Sum/add	transfer dominated
Polar code	Successive cancelation/List	serial	high	Min-Sum/add/sorting	balanced

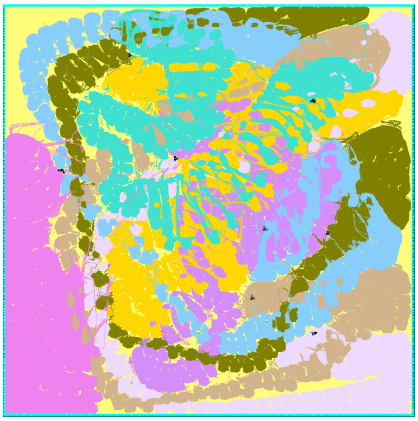

Fig. 1. 102 Gbit/s Turbo code decoder, area $23.61\,\mathrm{mm}^2$

III. LDPC CODE DECODING

LDPC code decoding is based on an iterative message exchange between variable and check nodes on the Tanner graph that is represented by the parity check matrix H. This BP has unlike the Turbo code decoding some inherent parallelism, since all check nodes (variable nodes) can be processed independently from each other. The decoder throughput is mainly limited by the iterative data exchange between the check and variable nodes. Hence, in contrast to Turbo code decoding, the BP algorithm is data transfer and not compute dominated. The result of each check node or variable node calculation has to be spread via the edges of the Tanner graph to all other connected nodes. The Tanner graph has very limited locality to provide good communications performance, which challenges an efficient implementation of the data transfers.

Let us consider an LDPC block code of length N and a parity check matrix H that has $\#edges(H)$ 1-entries (number of 1's in H equals the number of edges in the Tanner graph). Furthermore, let $\#proc\_edges(A)$ denote the number of edges that can be processed in one clock cycle by an architecture A and I the number of iterations to complete the decoding of a block. A runs with frequency f. Then, the corresponding coded bit throughput $T(H, A)$ can be estimated as follows:

$$T(H,A) = \frac{\#proc\_edges(A)}{\#edges(H)} \cdot N \cdot \frac{1}{I} \cdot f \text{ [bit/s]} \qquad (1)$$

We can classify different decoder architectures depending on the degree of parallelism $P = \frac{\#proc\_edges(A)}{\#edges(H)}$:

Partially parallel architectures: Only a subset of edges and nodes are processed in parallel, i.e. $(P < 1)$. These architectures are very common for large block sizes that use quasi-cyclic (QC) block codes. Here, several QC matrices are processed in parallel. This can be performed either in a row-based or in a column-based manner. The implementation challenge for this class of architectures lies in an efficient memory scheme and a corresponding flexible routing network, since the values calculated by the check and variable nodes have to be buffered. Hence, these architectures are memory dominated. Special constraints are imposed on H to avoid memory access conflicts. Depending on the structure of H, layered decoding becomes possible, which improves the communications performance.

Fully parallel architectures at iteration level: All edges are processed in parallel, i.e. $(P = 1)$. This architecture corresponds to a one-to-one mapping of the Tanner graph into hardware. Check nodes and variable nodes are instantiated and the corresponding edges are hardwired. Because of the low locality in the Tanner graph, routing congestion is a big challenge in these architectures. Thus, these architectures are routing dominated. Moreover, layered decoding is unfeasible.

Unrolled fully parallel architectures: These architectures are similar to the fully parallel architectures but the iterations are unrolled and pipelined, i.e. $(P = I)$. In every clock cycle, a new block is processed. Advantages of this architecture are 1) highest throughput and, 2) less routing congestion since the pipelined architecture implies mainly local wires.

To achieve a throughput larger than $100\,\mathrm{Gbit/s}$, only the last architectural approach is feasible [20], [10]. Let us consider the IEEE 802.11ad code with $N = 672$ and $\#edges(H) = 1890$ and a frequency of $400\,\mathrm{MHz}$ that is feasible in a low V_t $28\,\mathrm{nm}$ FD-SOI technology. We assume 9 decoding iterations. Based on Equation 1, a throughput of $10\,\mathrm{Gbit/s}$ results for the partially parallel (row-wise), $29\,\mathrm{Gbit/s}$ for the fully parallel and $268\,\mathrm{Gbit/s}$ for the unrolled fully parallel architecture.

Figure 2 shows the layout of the LDPC decoder in the same technology and PVT assumptions as the Turbo code decoder. Different colors represent the individual iterations (in total 9). The power consumption is about $1.5\,\mathrm{W}$, resulting in $5.6\,\mathrm{pJ/bit}$. If 4 iterations are sufficient, the area reduces to $1.3\,\mathrm{mm}^2$ and the power consumption to $700\,\mathrm{mW}$, yielding an energy efficiency of $2.5\,\mathrm{pJ/bit}$.

IV. POLAR CODE DECODING

Successive Cancelation (SC), Successive Cancelation List (SCL) and BP are the most prominent decoding algorithms for Polar codes. Decoding corresponds to a traversal of the corresponding Polar Factor Tree (PFT) in which the received log-likelihood ratios from the channel are processed by the tree nodes. SC and SCL decoding is a depth-first traversal of the PFT, BP a breadth-first traversal iterating from the leaves to the

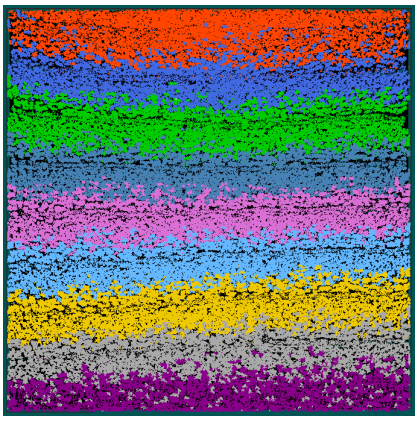

Fig. 2. 268 Gbit/s LDPC code decoder, area 2.8 mm^2

root and backwards. Different operations have to be performed at the tree nodes visited during the traversal, depending on the decoding algorithm. Corresponding node results are forwarded to the child/parent nodes, respectively. BP needs a large set of iterations to achieve the error correction performance of SC and is not well suited for very high throughput and low latency [5]. SC and SCL, on the other hand, have sequential behavior due to the mandatory depth-first tree traversal. To achieve a very high throughput, the tree traversal can be unrolled and pipelined [11], very similar to the iteration unrolling in Turbo code and LDPC code decoding. Whenever a node is visited during the tree traversal, a corresponding pipeline stage can be instantiated. In this way, for a block length of N (= number of leaves in the PFT), the maximum number of pipeline stages is $2 * (2N - 2) + 1$. The complexity of the decoding architecture is directly proportional to the size of the PFT, which can be reduced by various transformations. For example, if a subtree represents a repetition code or a parity check code, the corresponding subtree can be replaced by a single node. Alike, we can merge rate-0 and rate-1 nodes into its parent nodes [19] or use majority logic decoding in subtrees.

As already said in the introduction, heavily pipelined high-throughput architectures require a huge amount of registers, which cost area and power. This memory requirement can be reduced at algorithmic level, e.g. by efficient quantization. Techniques like retiming enable the reduction of registers at (micro-) architectural level. Beyond that, registers can be replaced by latches. Latches are much more energy and area efficient than registers and largely reduce the load for the clock tree.

Here, we consider a 1024/512 Polar code that is decoded with an SC algorithm. The same technology and PVT setup as for the Turbo code and LDPC decoder is applied. The unoptimized PFT has 2047 nodes, corresponding to 4093 pipeline stages. The tree optimization yields a reduction to 385 stages. Implementing this tree in a fully pipelined architecture would require about 310 KB memory. Retiming with a frequency constraint of 800 MHz results in a partially pipelined architecture with only 111 pipeline stages, which require about 90 KB memory. The power consumption of this design is 5.7 W, of which more than 70% is consumed in registers and the clock tree. In a further step, we replace registers by latches. This process is fully automated and compatible with a standard VHDL synthesis flow and standard libraries. In the new design, the area is reduced from 3.12 mm^2 to 2.95 mm^2, the power consumption from 5.7 W to 3.3 W, and the throughput is increased from 636 Gbit/s to 764 Gbit/s. Only 33% of the overall power accounts for registers and clock tree. The design is running with a frequency of 746 MHz, which results in a coded bit throughput of 764 Gbit/s. Figure 3 shows the layout of the Polar code decoder. Each color represents a pipeline stage (111), the black color is memory.

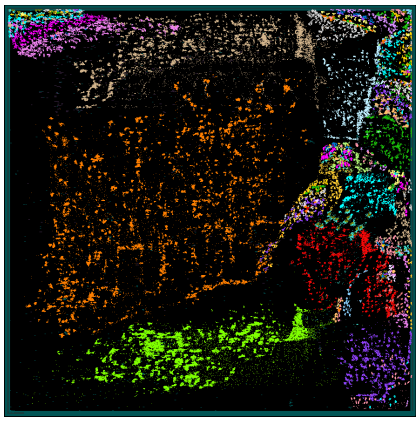

Fig. 3. 764 Gbit/s Polar code decoder, area 2.95 mm^2

V. Conclusion

Table II compares the different decoders in terms of implementation efficiency. It is important to mention that communications performance is not considered. It is important to mention that there is a strong interrelation between communications performance and implementation efficiency [15]. Code block lengths and code rates differ. Thus, the communications performance will be different for the various decoders [8].

However, some general conclusions can be drawn. LDPC decoder architectures run with lower frequency. This is due to the routing issue that originates from the BP algorithm. The LDPC decoder is superior to the Turbo code decoder in terms of throughput, area and energy efficiency. A reason for that is the fact that BP is inherently parallel. Thus, some researchers investigated the use of BP decoding also for Turbo codes [18], [16]. However, its communications performance cannot compete with the MAP decoding performance. The Polar code decoder has the highest throughput of all decoders. Area and energy efficiency for Polar code and LDPC code decoders are comparable for a low number of LDPC decoding iterations.

Throughputs beyond 100 Gbit/s are feasible for all three code classes by appropriate "unrolling" and utilizing heavy pipelining and spatial parallelism. However, this architectural approach is limited to small block sizes (area, power, routing

TABLE II
COMPARISON OF CHANNEL CODE DECODERS

Code	Blocksize [bit]	Code rate	Frequency [MHz]	Throughput [Gbit/s]	Area [mm^2]	Power [mW]	Area efficiency [$Gbit/s/mm^2$]	Energy efficiency [pJ/bit]
Turbo code (4 iter)	128	1/3	800	102	23.6	-	4.34	-
LDPC code (9 iter)	672	13/16	400	268	2.8	1500	95.7	5.6
LDPC code (4 iter)	672	13/16	400	268	1.3	700	215	2.5
Polar code	1024	1/2	746	764	2.95	3300	259	4.4

congestion) and small number of iterations (Turbo code, LDPC code), which negatively impacts the communications performance. Moreover, although pipelining largely increases the throughput and locality, it also increases the latency. The reduction of latency at the implementation level is very restricted. A large latency improvement is only possible by degrading the communications performance, e.g. by reducing the number of iterations or using simplified decoding algorithms, etc. All architectures suffer from limited flexibility in terms of block sizes (all three codes), varying number of iterations (Turbo code, LDPC code) and code rate flexibility (LDPC code and Polar code).

Summarized, the biggest challenges for very high-throughput decoder architectures are:

- Improving the communications performance under the aforementioned implementation constraints.
- Providing block size, code rate and algorithmic flexibility.
- Power density is in the order of $1\,\mathrm{W/mm^2}$, which is far too high for air cooled packages.

As discussed in the introduction, microelectronic progress will largely contribute to an improved area efficiency but not much to an increased performance and a reduced power density. Thus, further research is mandatory to keep pace with the increasing requirements on communication systems in terms of throughput, latency, power/energy efficiency, flexibility, cost and communications performance.

ACKNOWLEDGMENT

We gratefully acknowledge financial support by the DFG (project-ID: 2442/8-1) and the EU (project-ID: 760150-EPIC).

REFERENCES

[1] EPIC - Enabling Practical Wireless Tb/s Communications with Next Generation Channel Coding - https://epic-h2020.eu/results.

[2] Foundry Challenges in 2018 - https://semiengineering.com/foundry-challenges-in-2018/.

[3] International Roadmap for Devices and Systems (IRDS) 2017 Edition: More Moore - https://irds.ieee.org/images/files/pdf/2017/2017IRDS_MM.pdf.

[4] ISSCC Keynote 2014, Mark Horowitz - http://eecs.oregonstate.edu/research/vlsi/teaching/ECE471_WIN15/mark_horowitz_ISSCC_2014.pdf.

[5] S. M. Abbas, Y. Fan, J. Chen, and C. Y. Tsui. High-Throughput and Energy-Efficient Belief Propagation Polar Code Decoder. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 25(3):1098–1111, March 2017.

[6] Gene M. Amdahl. Computer architecture and Amdahl's Law. *Computer*, 46(12):38–46, 2013.

[7] Erdal Arikan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7):3051–3073, 2009.

[8] A. Balatsoukas-Stimming, P. Giard, and A. Burg. Comparison of Polar Decoders with Existing Low-Density Parity-Check and Turbo Decoders. In *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pages 1–6, March 2017.

[9] G. P. Fettweis and E. Matus. Scalable 5G MPSoC architecture. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pages 613–618, October 2017.

[10] R. Ghanaatian, A. Balatsoukas-Stimming, T. C. Müller, M. Meidlinger, G. Matz, A. Teman, and A. Burg. A 588-Gb/s LDPC Decoder Based on Finite-Alphabet Message Passing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 26(2):329–340, February 2018.

[11] P. Giard, G. Sarkis, C. Thibeault, and W. J. Gross. 237 Gbit/s unrolled hardware polar decoder. *Electronics Letters*, 51(10):762–763, 2015.

[12] Jörg Henkel, Lars Bauer, Nikil Dutt, Puneet Gupta, Sani Nassif, Muhammad Shafique, Mehdi Tahoori, and Norbert Wehn. Reliable on-chip systems in the nano-era: Lessons learnt and future trends. In *Proceedings of the 50th Annual Design Automation Conference on - DAC '13*, page 1, Austin, Texas, 2013. ACM Press.

[13] T. Ilnseher, F. Kienle, C. Weis, and N. Wehn. A 2.15GBit/s turbo code decoder for LTE advanced base station applications. In *2012 7th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, pages 21–25, August 2012.

[14] Michael J. Thul, Frank Gilbert, Timo Vogt, Gerd Kreiselmaier, and Norbert Wehn. A Scalable System Architecture for High-Throughput Turbo-Decoders. In *The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology*, volume 39, pages 152–158, January 2005.

[15] F. Kienle, N. Wehn, and H. Meyr. On Complexity, Energy- and Implementation-Efficiency of Channel Decoders. *IEEE Transactions on Communications*, 59(12):3301–3310, December 2011.

[16] Lianxiang Zhu, Jifeng Wang, and Shizhong Yang. Factor graphs based iterative decoding of turbo codes. In *IEEE 2002 International Conference on Communications, Circuits and Systems and West Sino Expositions*, volume 1, pages 46–50, Chengdu, China, 2002. IEEE.

[17] G. E. Moore. Cramming More Components Onto Integrated Circuits. *Electronics*, pages 114–117, April 1965.

[18] Ahmed Refaey, Sebastien Roy, and Paul Fortier. On the application of BP decoding to convolutional and turbo codes. In *2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers*, pages 996–1001, Pacific Grove, CA, USA, 2009. IEEE.

[19] 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*, 32(5):946–957, May 2014.

[20] P. Schläfer, N. Wehn, M. Alles, and T. Lehnigk-Emden. A new dimension of parallelism in ultra high throughput LDPC decoding. In *SiPS 2013 Proceedings*, pages 153–158, October 2013.

[21] Stefan Scholl, Stefan Weithoffer, and Norbert Wehn. Advanced iterative channel coding schemes: When Shannon meets Moore. In *2016 9th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, pages 406–411, Brest, France, September 2016. IEEE.

[22] Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(4):623–656, 1948.

[23] William Shockley. The Theory of p-n Junctions in Semiconductors and p-n Junction Transistors. *The Bell System Technical Journal*, 28:435–489, 1949.

[24] S. Weithoffer, C.A. Nour, N. Wehn, C. Douillard, and C. Berrou. 25 Years of Turbo Codes: From Mb/s to beyond 100 Gb/s. *International Symposium on Turbo Codes & Iterative Information Processing (ISTC), December, 2018, Hong Kong, China*, 2018.