Disclosing the LDPC Code Decoder Design Space

Torben Brack, Frank Kienle, Norbert Wehn
Microelectronic System Design Research Group*
University of Kaiserslautern
Erwin-Schrödinger-Straße
67663 Kaiserslautern, Germany
{brack, kienle, wehn}@eit.uni-kl.de

Abstract

The design of future communication systems with high throughput demands will become a critical task, especially when sophisticated channel coding schemes have to be applied. LDPC codes are one of the most promising candidates because of their outstanding communications performance. One major problem for a decoder hardware realization is the huge design space composed of many interrelated parameters which enforces drastic design trade-offs. Another important issue is the need for flexibility of such systems.

In this paper we illuminate this design space with special emphasis on the strong interrelations of theses parameters. Three design studies are presented to highlight the effects on a generic architecture if some parameters are constraint by a given standard, given technology, and given area constraints.

1 Introduction

Low-Density Parity-Check (LDPC) codes are one of the best performing channel codes known today. Invented by Gallager in 1963 [2], they were almost forgotten for nearly 30 years. Rediscovered by MacKay in the mid-90s and enhanced to irregular LDPC codes by Richardson et. al. in 2001 [3], they are now to be used for forward error correction in a vast number of upcoming standards like DVB-S2 [4], WiMax (IEEE802.16e), and wireless LAN (IEEE 802.11n). Providing very high decoding throughput and outstanding communications performance, they will probably become the channel coding scheme of choice for years to come. Major competitors to LDPC codes are Turbo Codes which are already applied in the current UMTS [5] standard.

*The work presented in this paper was supported by the European IST project 4MORE 4G MC-CDMA multiple antenna system On chip for Radio Enhancements [1]

Recent LDPC decoder hardware implementations reached astonishing throughputs by massive parallelization of the decoding process. However, these implementations often restrain the codes supported to only a small subclass or even one specific code at all [6]. To provide service flexibility, partly parallel achitectures become mandatory. The variety of the LDPC decoders published [6][7][8][9][10] makes a meaningful comparison almost impossible, particularly due to the lack of a representative cost function. Thus, all relevant decoder parameters have to be considered for evaluation. The contributions of this paper are

- Exploration of the multi-dimensional design space for LDPC decoders, emphasizing the complex interrelations between the parameters involved in the design process.
- Presentation of a synthesizable generic LDPC decoder template which allows for efficient design reuse for different design parameters and target applications.
- Demonstration of three sophisticated design studies with respect to area, throughput, and communications performance, emphasizing the often underestimated service flexibility regarding block length and code rate.

The paper is structured as follows: Section 2 shortly introduces LDPC codes, followed by the design space exploration in Section 3. Section 4 introduces the decoder architecture template, the design studies are presented in Section 5.

2 LDPC Codes at a Glance

LDPC codes are linear block codes defined by a sparse binary matrix H, called the parity check matrix. The set of valid codewords C satisfies

$$Hx^T = 0, \qquad \forall x \in C. \tag{1}$$

A column in H is associated to a codeword bit, and each row corresponds to a parity check. A nonzero element in

3-9810801-0-6/DATE06 © 2006 EDAA

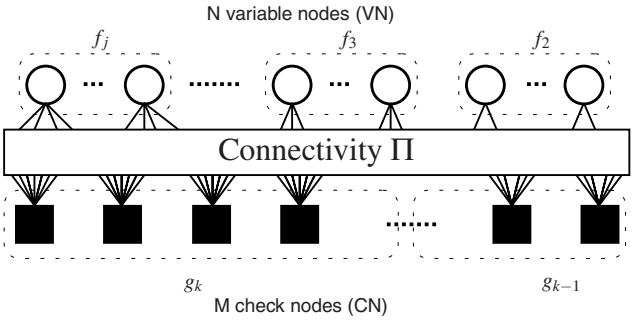

Figure 1. Tanner Graph for an Irregular LDPC Code

a row means that the corresponding bit contributes to this parity check. The complete code can best be described by a Tanner graph [3], a graphical representation of the associations between code bits and parity checks. Code bits are shown as so called variable nodes (VN) drawn as circles, parity checks as check nodes (CN) represented by squares, with edges connecting them accordingly to the parity check matrix. Figure 1 shows a Tanner graph for a generic irregular LDPC code with N variable and M check nodes with a resulting code rate of $R = (N-M)/M$.

The number of edges on each node is called the node degree. If the node degree is identical for all nodes, the corresponding LDPC code is called regular, otherwise it is called irregular. Note that the communications performance of an irregular LDPC code is known to be generally superior to which of regular LDPC codes. The degree distribution of the VNs $f_{[j,\ldots,3,2]}$ gives the fraction of VNs with a certain degree, with j the maximum variable node degree. The degree distribution of the CNs can always be expressed as $g_{[k,k-1]}$ with k the maximum CN degree, meaning that only CNs with degree k and $k-1$ will occur.

2.1 Optimal Decoding

LDPC codes can be decoded using the message passing algorithm [2]. It exchanges soft-information iteratively between variable and check nodes. Updating the nodes can be done with a canonical, two-phased scheduling: In the first phase all variable nodes are updated, in the second phase all check nodes respectively. The processing of individual nodes within one phase is independent and can thus be parallelized.

The exchanged messages are assumed to be log-likelihood ratios (LLR). Each variable node of degree i calculates an update of message k according to:

$$\lambda_k = \lambda_{ch} + \sum_{l=0,l\neq k}^{i-1} \lambda_l, \tag{2}$$

with λ_{ch} the corresponding channel LLR of the VN and λ_i the LLRs of the incident edges. The check node LLR update can be done in an either optimal or suboptimal way, trading of implementation complexity against communications performance. Optimal check node decoding can be done calculating:

$$tanh(\lambda_k/2) = \prod_{l=0,l\neq k}^{i-1} tanh(\lambda_l/2), \tag{3}$$

Equation 2 and Equation 3 combined yield the Sum-Product or Belief-Propagation algorithm. Hardware realizations of this function can become very complex, especially if different CN degrees have to be supported.

2.2 Suboptimal Decoding

The simplest suboptimal check node algorithm is the well-known Min-Sum algorithm [11], where the incident message with the smallest magnitude mainly determines the output of all other messages:

$$\lambda_k = \prod_{\forall i,i\neq k} \text{sign}(\lambda_i) \cdot \min_{\forall i,i\neq k}(|\lambda_i|) \tag{4}$$

The resulting communications performance can be improved by scaling the updated messages with a so called message scaling factor (MSF). In this paper, we will always refer to the Min-Sum algorithm while assuming an MSF of 0.75. The resulting performance comes close to the optimal Sum-Product algorithm only for high rate LDPC codes ($R \geq 3/4$) with relatively large CN degree. For lower code rates the communications performance strongly degrades.

Thus a more sophisticated suboptimal algorithm has to be used for low rates. The λ-3-Min algorithm [11] uses only the three smallest absolute input values and applies a correction term to make up for the introduced approximation. While increasing implementation complexity, this decoding scheme almost approaches the optimal algorithm for any given code rate. Thus, the λ-3-Min algorithm is used if a wide range of code rates has to be supported. For area-optimized decoder implementations for high rate codes the Min-Sum algorithm is applied. In Section 5 we present designs using both algorithms for their respective domains.

3 The LDPC Decoder Design Space

The hardware realization of an LDPC decoder is determined by many strongly interrelated parameters. This section illuminates the large, multi-dimensional design space which is relevant for partly parallel LDPC decoder implementations. The complexity of this design space also makes evaluation of published LDPC decoders a very difficult task due to the lack of a representative cost function.

The three major metrics for the evaluation of an LDPC decoder are VLSI performance, communications performance, and the supported service flexibility. The VLSI performance is defined by the chip *area*, *throughput* and energy

Design Parameters	VLSI Parameters		Communications Performance	Service Parameters	
	Area	Throughput		Block Length	Code Rate
LDPC Code	More edges increase RAM area. Higher code rate flexibilitys increase logic area.	More edges decrease throughput. Lower edge/(R·VN)-ratio increases throughput.	Irregular LDPC codes perform better than regular LDPC codes.	Smaller block length reduces number of edges and irregularity.	Higher code rates reduce irregularity.
Algorithm	Larger area allows for more optimal decoding algorithms.		Optimal algorithms perform better than suboptimal ones.	Smaller block lengths are more suitable for subopt. algorithms.	High code rates are more suitable for suboptimal algorithms.
Iterations		Throughput is inverse proportional to the number of iterations.	More iterations increase communications performance.	Larger block lengths require more iterations.	Higher code rates require less iterations.
Quantization	Larger area allows for higher quantization.		Increased performance with higher quantization.		Higher code rates allow for smaller quantization.
Architecture Parallelism	Increased logic area by higher parallelism.	Throughput proportional to parallelism.	Parallelism can limit communications performance.	Larger block sizes allow for higher parallelism.	

Table 1. Important Parameters of the LDPC Decoder Design Space

consumption. *Communications performance* can be determined by the packet error rate (PER) for a given signal-to-noise ratio (SNR) which has to be compared to other currently applied channel coding schemes like UMTS Turbo Codes [5].

Flexibility is mainly associated with the permitted variability for the service parameters *block length* and *code rate*. Further design parameters exist like the applied *LDPC code* which can be regular or irregular, implemented *decoding algorithm*, maximum number of *iterations*, *quantization* level, and the *parallelism* deployed in data processing. Table 1 shows the important parameters of the design space and their interrelation. Any of this parameters can be constrained by either externally supplied standardizations or requirements of new applications.

Due to space limitations of this paper, we will discuss only one parameter in detail. Since code rate flexibility is so often ignored in recent publications, we explore their influence on other parameters within the design space.

Increasing the code rate reduces the desired irregularity of the Tanner graph of the *LDPC code* because there are fewer check nodes in comparison to the variable nodes. This makes it far more difficult to keep a sparse graph which is necessary for successful decoding. This in turn degrades the *communications performance* in addition to the redundancy loss for higher code rates, since irregular codes perform better than regular ones. Note that the performance gain between regular and irregular code becomes most significant for relative low code rates.

Using higher code rate, the achievable *throughput* is increased because the required sparseness of the graph results in fewer messages to be exchanged for each information bit. For lower code rates more *iterations* are needed to exploit the potential *communications performance*, which in turn decreases the *throughput*.

As pointed out in Section 2.2, LDPC codes with high code rates are suitable for simpler suboptimal decoding *algorithms*, which leads to decreased *area* consumption of the decoder. However, if flexibility demands low code rates, the λ-3-Min algorithm becomes mandatory. Further more, the highest code rate which implies the highest CN degree influences the *area* needed for the decoding units.

Because of the simpler suboptimal *algorithms* applicable for high rate codes, the quantization parameter has to be considered. These algorithms allow for more inaccurately *quantization* without appreciable impact on *communications performance* compared to the *algorithms* available for lower code rates. Smaller *quantization* however permits strong reduction of the RAM and logic *area* utilized by the decoder. As before, flexibility may prohibit this kind of optimization.

Finally, the decoder *area* is not only determined by the lowest and highest code rate, but also by the number of different code rates supported, which often increases the implementation complexity of the decoder control logic.

4 Decoder Architecture Template

A generic architecture template becomes mandatory to allow for an efficient design reuse for different target applications. This template has to be fully adaptable regarding all design parameters presented in Section 3.

For an efficient architecture, some restrictions had to be set up:

- A partly parallel approach which can process a certain number of edges per clock cycle concurrently is necessary to support reasonable throughput and flexibility at the same time.
- The message exchange network which represents the Tanner graph has to be flexible and of manageable

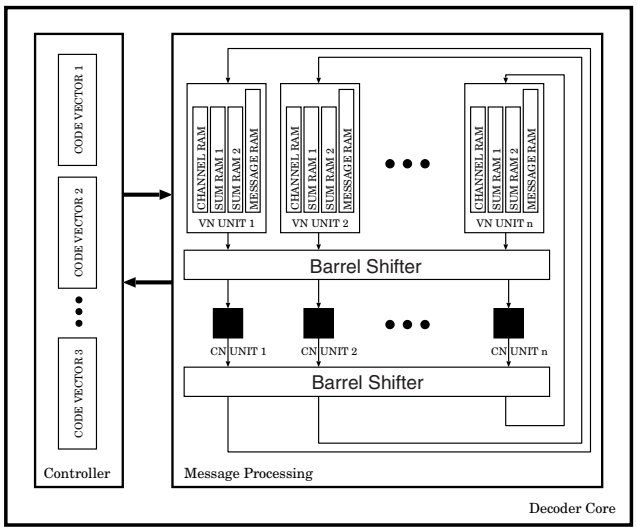

Figure 2. Decoder Architecture Template

complexity without limiting the usable code space to much. Especially routing congestion has to be avoided.

- Check node processing has to be done in a serial manner. While limiting processing speed to one message per clock cycle for each node, this also reduces potential memory conflicts resulting in an enhanced flexibility and support of a much larger code range without the excessive area overhead of parallel check node implementations.

The actual architecture template is shown in Figure 2. The *CN Units*, represented as black boxes, implement either the Min-Sum or the λ-3-Min algorithm [11] introduced in Section 2.2. They read the message LLR belonging to the incident edges of the covered check nodes from the *Message RAM* which have to be subtracted from the sum of the related channel LLR in the *Channel RAM* and the accumulated extrinsic message LLR stored in one of the *Sum RAMs* during the previous iteration. This subtraction is carried out by the *VN Units* reflecting the variable node calculation given by Equation 2. After the check node processing, the resulting extrinsic information is stored back in the appropriate *Message RAM* and accumulated up into the other *Sum RAM*. At the beginning of the next iteration, the two *Sum RAMs* are interleaved. This technique was already presented in [10]. All messages are exchanged via two *Barrel Shifters* which are controlled accordingly to the *Code Vectors* stored inside the *Controller*. It also controls the CN and VN Units and accessing the RAMs. The *Code Vectors* represent the LDPC codes to be applied and can be either hard-coded or reloaded during runtime. By using barrel shifters for the message exchange network, the decoder can process all LDPC codes based on permutation matrices [4]. The construction of LDPC codes which can be adapted to barrel shifter based architectures was already demonstrated in [10][12].

5 Design Studies

In the following sections we present some LDPC code decoder designs which are representative for a large number of possible applications. They all rely on the architecture template introduced in Section 4 and incorporate all parameters elaborated in Section 3:

- Implementation of the DVB-S2 standard LDPC code [4] as an ASIC to demonstrate *high code rate flexibility* in a high throughput base-station application.
- Implementation of the proprietary 4MORE [1] LDPC code to show a very efficient FPGA implementation with *block length diversity*.
- Development of two highly flexible multi-purpose LDPC decoders which fit on *only* $1mm^2$ *in* $0.13\mu m$ *ASIC technology*.

Constraint design space parameters are marked bold in the corresponding tables.

5.1 DVB-S2 LDPC Code

The DVB-S2 satellite video broadcasting standard [4] was designed for an exceptional error performance at very low SNR ranges (up to PER $\geq 10^{-7}$ at -2.35dB E_S/N_0). Thus the specified codes use a large block length of **64800bit** with 12 different code rates ranging from $\mathbf{1/4}$ to $\mathbf{9/10}$. This results in large storage requirements for up to 285000 messages and demands high code rate flexibility at the same time to support all specified degree distributions. The transmission of multiple high definition television (HDTV) streams at 20Mbps each requires a very high throughput, especially for base-station applications. To be compatible with the enhanced DVB-ASI specifications, our design goal was **540Mbps** for the parallel transmission of 20 HDTV and 20 regular SDTV channels at the highest code rate.

Table 2 shows synthesis results of our decoder template for DVB-S2 LDPC code processing using the STM $0.13\mu m$ technology. The selected clock frequency of 270Mhz was mainly determined by memories. Thereby we traded off memory area and access time to yield an efficient implementation. Internal quantization for the messages was set to 6bit to eliminate any quantization loss. In contrast to our earlier published DVB-S2 decoder [9], we chose a check-node implementation of the λ-3-algorithm to reduce the logic area. Furthermore, the two independent permutation networks of the current template allow for much more decoding iterations per block. To compensate for the loss in communications performance introduced by using a suboptimal algorithm, we used 50 iterations instead of 30 which still fulfills the troughput constraints. Hence, even for the smallest code rate of $1/4$ a throughput of over 150Mbps was reached.

LDPC Code	see [4]
Block Length	64800bit
Code Rate	1/4 - 9/10
Parallelism	360
Quantization	6bit
Algorithm	λ-3-Min
Iterations	50
Comm. Perform.	see Section 5.1,[4]
Area[mm^2] 0.13μm@270Mhz	
Bit-Nodes	0.782
Check-Nodes	3.614
Controller	0.012
Network	0.974
Channel RAM	1.997
Message RAM	9.304
Sum RAM	4.640
Code Vectors	0.075
Overall Area	21.398
Throughput	157-542Mbps

Table 2. Synthesis Results for the DVB-S2 LDPC Code Decoder

5.2 4MORE LDPC Code

The 4MORE project [1] was initiated primarily to test and demonstrate advanced techniques for wireless data transmission. The channel decoding part of the transmission chain is based on an LDPC code.

The 4MORE LDPC code had to support three different block lengths of **1000, 2000, and 3000bit** with a coding rate of 4/5. The irregular LDPC codes we developed to fulfill this requirements have a degree distribution of $f_{[7,3,2]} = \{1/5, 3/5, 1/5\}$ and support a parallelism of 100. A **Xilinx Virtex4-LX100 FPGA** (49152 Slices, 240 BRAM) at **100Mhz** clock frequency was specified as implementation platform. The input quantization was limited to 4bit by the demodulator interface, but internal quantization can be set to 6bit to further improve communications performance by preventing error floors due to early saturation. Because of the relatively high code rate it was by far sufficient to use the suboptimal Min-Sum algorithm with a MSF of 0.75. The gain by using optimal decoding for this codes was simulated to be below 0.1dB. Within this project, all defined codes had to be integrated in the controller of the decoder due to the lack of an external storage. To minimize memory fragmentation, we utilized the complimentary dual-ported BRAM of modern Xilinx FPGA.

Table 3 shows synthesis results for 4bit and 6bit quantization obtained by using the Xilinx ISE 6.3 suite. We applied 10 iterations per decoded block, yielding a PER $\geq 10^{-3}$ above 5.8dB E_S/N_0 for a block length of 3000bit using the 6bit version. Hence, a maximum throughput of 180Mbps was achieved.

LDPC Code	see Section 5.2	
Block Length	1000,2000,3000bit	
Code Rate	4/5	
Parallelism	100	
Quantization	4bit	6bit
Algorithm	Min-Sum+MSF	
Iterations	10	
Comm. Perform.	see Section 5.2,[1]	
XC4VLX100 FPGA@100Mhz		
Bit-Nodes	3418 Slices	4822 Slices
Check-Nodes	4600 Slices	5300 Slices
Controller	166 Slices	
Network	2944 Slices	4830 Slices
RAM	69 BRAM	108 BRAM
Code Vectors	7 BRAM	
Overall Logic	11729 Slices	15534 Slices
Overall Memory	76 BRAM	115 BRAM
Throughput	131-180Mbps	

Table 3. Synthesis Results for the 4MORE LDPC Code Decoder

5.3 Towards Future Applications

To demonstrate the versatility of our architecture, we built two different very flexible LDPC decoders fitting on only **1mm²** employing a **0.13μm** technology. In consumer applications area is a very critical factor. One implementation focuses on high throughput, the other one on best as possible communications performance achievable for this area constraint. For maximum flexibility, both decoders support block lengths from **200-1000bit** and Tanner graphs with about 4000 edges. There are no further restrictions regarding the possible VN degree distributions, the maximum CN degree is constrained to 18.

The communications performance oriented decoder version supports all code rates from 1/4-4/5 and incorporates the λ-3-Min algorithm to guarantee sufficient performance for all code rates (see Section 2.2). To minimize any performance loss, 6bit quantization and 40 iterations per decoding process were used. Code rate flexibility for the high-throughput decoder was constrained to 3/4-4/5 to allow for the smaller Min-Sum implementation with only 4bit quantization (see Section 3). The saving in implementation complexity was used to double the parallelism and therefore nearly double the throughput. Furthermore, the number of iterations was decreased to 10, because this is sufficient for the supported high rate codes. We used high-speed memories with an access time below 2.8ns and a clock frequency of 333Mhz. This only slightly increases the decoder area, but allows for higher throughput.

Table 4 shows synthesis results for both decoders. The first column shows the throughput oriented implementation which achieves a very high throughput up to 324Mbps. The

	Throughput Oriented	Comm. Perf. Oriented
Block Length	**200-1000bit**	
Code Rate	$^3/_4$ - $^4/_5$	$^1/_4$ - $^4/_5$
Parallelism	60	30
Quantization	4bit	6bit
Algorithm	Min-Sum	λ-3-Min
Iterations	10	40
Comm. Perform.	see Section 5.3	
Area[mm^2] 0.13μm@333Mhz		
Bit-Nodes	0.256	0.178
Check-Nodes	0.279	0.335
Controller	0.008	0.012
Network	0.065	0.039
Channel RAM	0.062	0.066
Message RAM	0.080	0.118
Sum RAM	0.167	0.170
Code Vectors	0.041	0.078
Overall Area	**0.997**	**0.996**
Throughput	163-324 Mbps	15-47Mbps

Table 4. Synthesis Results for a 1mm^2 LDPC Decoder

second column shows the decoder which can compete with the communications performance of UMTS Turbo Codes. Figure 3 compares the UMTS Turbo Code and our LDPC decoder for 240 and 960bit block length with code rate $^1/_2$. Turbo Codes are known to be superior to LDPC codes for very small block lengths. Our LDPC decoder can reach the communications performance of UMTS Turbo Codes for block lengths of 960bit. Note that the assumed Turbo decoder uses 10 iterations applying the optimal Log-MAP algorithm, while the LDPC decoder still uses the suboptimal λ-3-Min algorithm. Although four times the iteration number of Turbo Codes are used, the achieved throughput still outperforms most Turbo Code decoder implementations.

6 Conclusion

In this paper, we presented for the first time comprehensively the design space of LDPC decoders. Thereby, the strong and often underestimated interrelations between relevant parameters were shown, especially focusing on flexibility. We presented a high throughput LDPC decoder implementation of the DVB-S2 standard, a highly efficient decoder realization on a Xilinx FPGA, and to the best of our knowledge the smallest flexible LDPC decoders ever published, including one which can keep up with UMTS turbo code communications performance.

References

[1] M. G. M.-C. multiple antenna system On chip for Radio Enhancements. http://ist-4more.org.

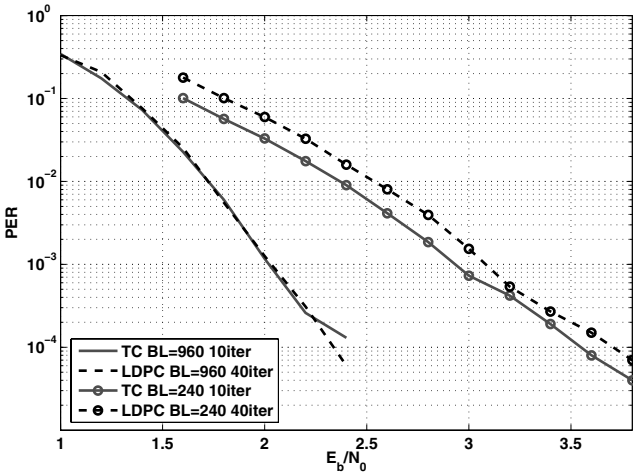

Figure 3. Communications Performance for Rate 1/2 LDPC and UMTS Turbo Codes

[2] R. G. Gallager. *Low-Density Parity-Check Codes*. M.I.T. Press, Cambridge,Massachusetts, 1963.

[3] T. Richardson and R. Urbanke. The Renaissance of Gallager's Low-Density Pariy-Check Codes. *IEEE Communications Magazine*, 41:126–131, Aug. 2003.

[4] European Telecommunications Standards Institude (ETSI). Digital Video Broadcasting (DVB) Second generation framing structure for broadband satellite applications; EN 302 307 V1.1.1. www.dvb.org.

[5] Third Generation Partnership Project. 3GPP home page. www.3gpp.org.

[6] A. Blanksby and C. J. Howland. A 690-mW 1-Gb/s, Rate-1/2 Low-Density Parity-Check Code Decoder. *IEEE Journal of Solid-State Circuits*, 37(3):404–412, Mar. 2002.

[7] D. Hocevar. LDPC Code Construction with Flexible Hardware Implementation. In *Proc. 2003 International Conference on Communications (ICC '03)*, pages 2708–2712, May 2003.

[8] M. Mansour and N. Shanbhag. High-Throughput LDPC Decoders. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 11(6):976–996, Dec. 2003.

[9] F. Kienle, T. Brack, and N. Wehn. A Synthesizable IP Core for DVB-S2 LDPC Code Decoding. In *Proc. 2005 Design, Automation and Test in Europe (DATE '05)*, Munich, Germany, Mar. 2005.

[10] E. Boutillon, J. Castura, and F. Kschischang. Decoder-first code design. In *Proc. 2nd International Symposium on Turbo Codes & Related Topics*, pages 459–462, Brest, France, Sept. 2000.

[11] F. Guilloud, E. Boutillon, and J. Danger. λ-Min Decoding Algorithm of Regular and Irregular LDPC Codes. In *Proc. 3nd International Symposium on Turbo Codes & Related Topics*, pages 451–454, Brest, France, Sept. 2003.

[12] F. Kienle and N. Wehn. Design Methodology for IRA Codes. In *Proc. 2004 Asia South Pacific Design Automation Conference (ASP-DAC '04)*, pages 459–462, Yokohama, Japan, Jan. 2004.