Copyright Notice

Energy Consumption of Channel Decoders for OFDM-based UWB Systems

Timo Lehnigk-Emden, Christian Brehm, Torben Brack, Norbert Wehn
Microelectronic System Design Research Group, University of Kaiserslautern,
Erwin-Schroedinger Str., 67663 Kaiserslautern, Germany
{lehnigk,brehm,brack,wehn}@eit.uni-kl.de

Friedbert Berens, Cem Derdiyok
STMicroelectronics N.V.,
1228 Plan-les-Ouates, Geneva, Switzerland
{friedbert.berens,cem.derdiyok}@st.com

***Abstract*— Ultra Wide Band (UWB) systems are designed for short range transmission with very high throughput demands (480 Mbit/s). Current UWB systems specify convolutional codes (CC) for channel coding. However, it was already shown that more sophisticated Low-Density Parity-Check Codes (LDPC) can outperform CC. This gain of communications performance comes with the penalty of a higher implementation complexity.**

In this paper, the currently specified convolutional code and an LDPC code are compared concerning their decoders' implementation complexity with special emphasis on its energy consumption. In this context, we evaluate an advanced iteration control for LDPC decoders in the environment of the future WIMEDIA UWB industry standard. The number of iterations is directly related to the LDPC decoder's energy consumption. Both decoders are implemented on a rapid prototyping platform to evaluate the energy consumption of LDPC decoding compared to CC decoding. Measurements show that the energy consumption of both decoders are nearly identical while LDPC codes come along with a more than 3 dB better communications performance.

***Index Terms*— LDPC, energy, low power, iteration control, implementation, WIMEDIA, UWB, decoder, architecture, FPGA**

I. Introduction

The existing WIMEDIA UWB standard [1] for short range devices specifies a data rate of up to 480 Mbit/s within a range of around 2 m. The currently deployed channel coding scheme is based on traditional convolutional code (CC) with a constraint length $K = 7$ and code rates between $1/3$ and $3/4$. By using an LDPC code as channel code, a very large coding gain can be achieved [2][3][4][5]. It was shown that the implementation complexity in sense of silicon area of such an LDPC decoder is below 0.3 mm^2 using a 65 nm technology from STMicroelectronics [6]. Due to those facts we are going to propose the LDPC decoder to incorporate into the WIMEDIA UWB standard.

LDPC codes were invented by Gallager in 1963 [7]. They were almost forgotten for nearly 30 years because of their implementation complexity and rediscovered by MacKay in the mid-90s and enhanced to irregular LDPC codes by Richardson et. al. in 2001 [8]. Now they are to be used for forward error correction in a vast number of upcoming standards like DVB-S2 [9], WiMax (IEEE 802.16e) [10], and wireless LAN (IEEE 802.11n) [11]. Providing very high decoding throughput and outstanding communications performance, they will probably become the channel coding scheme of choice for years to come.

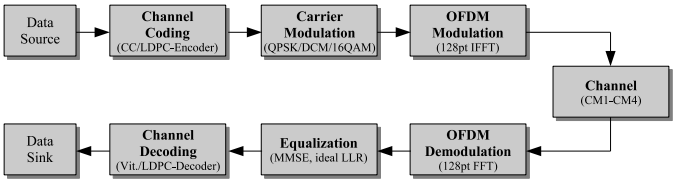

Fig. 1. WIMEDIA UWB simulation chain

Many UWB applications are in the consumer and portable domain. Hence energy consumption is of great relevance. It is often argued that the complexity of an LDPC decoder is much higher than the complexity of a Viterbi decoder. Thus, it is assumed that the energy consumption of LDPC decoding is also larger than the decoding process of a convolutional code.

In this paper, the energy consumption of an LDPC decoder is investigated. On system level we exploit an advanced iteration control scheme which is investigated in a WIMEDIA UWB environment. Simulations show the saving of iterations without losing any communications performance. Thus iteration control can efficiently be exploited to save energy in the decoding process.

The dynamic energy consumption of decoder architectures is mainly determined by the internal capacitances, register and memory accesses and toggling rates which are strongly data dependent. Hence it is important to use a sufficiently large and representative data set for energy estimation. Software simulations are inappropriate: energy models of simulations on high abstraction levels are too inaccurate, on the other hand, simulations on gate or transistor levels require an enormous large computation time. Hence we decided to use a rapid prototype environment and to perform real measurements. Both decoder systems, i.e. CC and LDPC decoder, are mapped onto this platform which is based on a Xilinx Virtex-4 FPGA. The platform is fed with data extracted from the UWB simulation chain [2]. It is well known that energy values from FPGAs can not be one-to-one transfered to an ASIC implementation. But the purpose of our investigation is the relative comparison between a CC and LDPC decoder and it is a feasible assumption that the relative comparison is also valid for an ASIC implementation.

The paper is structured as follows: Section II gives a short

Parameter	Value
Data rate	53Mbit/s to 960Mbit/s
Data carriers	100
FFT size	128 points
Symbol Duration	312.5ns (incl. Guard)
Channel Coding	CC with K = 7 LDPC Code
Carrier Modulation	QPSK, DCM, 16-QAM
Channel Models	CM1, CM2, CM3, CM4

TABLE I
WIMEDIA PHYSICAL LAYER PARAMETERS

overview over the WIMEDIA UWB system, followed by an introduction to LDPC codes and its decoding algorithms in Section III. Section IV concentrates on the custom-built LDPC code for the operation an an UWB system. Section V introduces into the various iteration control techniques whereas Section VI describes the architecture of our LDPC decoder. Results of software simulations, implementation complexity and power measurements are presented in the Section VII. Section VIII concludes the paper.

II. WIMEDIA UWB SYSTEM MODEL

The WIMEDIA UWB standard is based on a multiband OFDM air interface with and without frequency hopping. The overall US UWB band ranging from 3.1 GHz to 10.6 GHz is split into 14 subbands using 528 MHz of bandwidth with 128 OFDM subcarriers. The subbands are grouped to form 5 band groups. Four of them contain 3 subbands and one consists of 2 subbands. In this paper we will focus on the first band group ranging from 3.1 GHz to 4.8 GHz. In order to evaluate the communications performance of the WIMEDIA UWB standard a SystemC based simulation chain has been implemented for the full data path of the system. The basic structure of the simulation chain and thus the WIMEDIA UWB transmission and receiver chain is depicted in Figure 1.

The WIMEDIA uses a standard convolutional coding scheme with a constraint length of $K = 7$. For the purpose of this paper an LDPC encoder has been added to the standard chain. After the channel coding the coded data stream is interleaved and then mapped onto QPSK symbols for the lower data rates and DCM or 16-QAM symbols for the higher data rates ranging from 300 Mbit/s to 960 Mbit/s. These symbols are then used in the OFDM modulator to generate the OFDM symbols to be transmitted over the channel. The channel models used correspond to the IEEE 802.15.3a channels CM1 to CM4 [1][2]. The needed hopping sequences for the TFI mode are generated by the channel models.

In the receiver the signal is demodulated and equalized using an MMSE equalizer with an ideal LLR calculation. In the presented simulation results ideal channel estimation is assumed. The resulting demapped and deinterleaved soft-information is then processed by the channel decoder which is either a Viterbi decoder or an LDPC decoder.

The key parameters of the WIMEDIA UWB system are depicted in Table I.

A. WIMEDIA Convolutional Code and Viterbi Algorithm

The WIMEDIA UWB standard system currently uses a non systematical convolutional code with rate $1/3$ which can be adapted by puncturing (erasing some parity bits) to the rates $11/32$,$1/2$,$5/8$ and $3/4$. The encoder is a simple 6-step shift register with the following taps for the parities p_1 to p_3 in polynomial form:

$$p_1 = \{1, 0, 1, 1, 0, 1, 1\}$$
$$p_2 = \{1, 1, 1, 0, 1, 0, 1\}$$
$$p_3 = \{1, 1, 1, 1, 0, 0, 1\}$$

The decoding of the CC code at the receiver is done by the soft input Viterbi algorithm, which means that the decoder uses soft-information for decoding but the results are hard decoded bits. The Viterbi algorithm (VA) was introduced in 1967 as a method to decode binary convolutional codes [12]. The VA finds the most likely sequence of state transitions (a path) through a trellis with a finite number of N states $S^{(m)}$, given by the sequence of received symbols y. In other words the VA gives us the most likely codeword which was sent. This property is known as Maximum Likelihood (ML) decoding.

The basic idea of the VA is quite simple. A branch metric $\gamma_{k,k+1}^{m,m'}$ is assigned to each possible state transition $S_k^{(m)} \rightarrow S_{k+1}^{(m')}$. In each decoding cycle and for all states the path with the best sum of the transition metrics, called the *local survivor*, is selected. The decision bits indicating the local survivor for each state and each trellis cycle are stored in a survivor memory. In a proceeding traceback step the local survivors are read from this memory back in time to extract the most likely sequence.

III. LDPC CODES

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 a row means that the corresponding bit contributes to this parity check. The complete code can best be described by a Tanner graph [8], 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 2 shows a Tanner graph for a generic LDPC code with N variable and M check nodes with a resulting code rate of $R = (N - M)/N$.

The number of edges supplying each node is called the node degree. If the node degree is constant for all CNs and VNs, 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

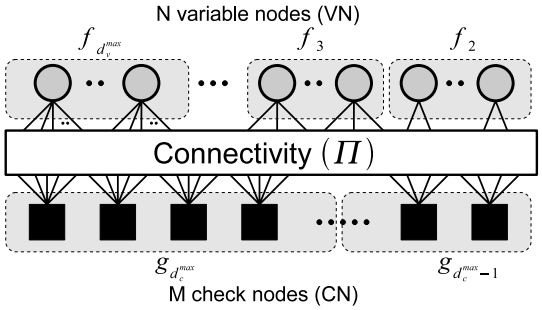

Fig. 2. Tanner graph for an irregular LDPC code

of the VNs $f_{[d_v^{max},\ldots,3,2]}$ gives the fraction of VNs with a certain degree, with d_v^{max} the maximum variable node degree. The degree distribution of the CNs can be expressed as $g_{[d_c^{max},d_c^{max}-1]}$ with d_c^{max} the maximum CN degree, meaning that only CNs with two different degrees occur [13].

To obtain a good communications performance of an LDPC code the degree distribution should be optimized with respect to the codeword size N. The degree distribution can be optimized by density evolution as shown in [13]. Furthermore, the resulting Tanner graph should have cycles as long as possible to ensure that the iterative decoding algorithm works properly. A cycle in the Tanner graph is defined as the shortest path from a VN back to its origin without traveling an edge twice. Especially cycles of length four have to be avoided [8].

IV. UWB LDPC Codes

A. UWB LDPC Code Design

Fulfilling the latency constraints of the UWB system (less than 2 μs) in conjunction with a small resulting decoder silicon area was one of the major prerequisites. To guarantee these we had to design an ultra sparse LDPC code [2][6]. The chosen degree distribution is $f_{[3,2]} = \{3/4, 1/4\}$ for the variable nodes and $g_{[11]} = \{1\}$ for the check nodes. It is important to note that an LDPC code with a denser degree distribution, i.e. $f_{[6,3,2]} = \{1/4, 1/2, 1/4\}$ and $g_{[14]} = \{1\}$ can gain up to 0.5 dB in terms of communications performance compared to our chosen one. However, the resulting decoder would result in a twice as large silicon area compared to our approach. Furthermore, the layered constraint could not be preserved which would involve a high throughput penalty.

B. Decoding Algorithm

LDPC codes can be decoded using the message passing algorithm [7]. 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 d_v calculates an update of message k according to:

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

with λ_{ch} the corresponding channel LLR of the VN and λ_l the LLRs of the incident edges. The check node LLR update can be done in an either optimal or suboptimal way, trading off implementation complexity against communications performance. For our studies only the suboptimal way is relevant.

C. Suboptimal Decoding

The simplest suboptimal check node algorithm is the well-known Min-Sum algorithm [14], where the incident message with the smallest magnitude determines the output of all other messages:

$$\lambda_k = \prod_{l=0, l\neq k}^{d_c-1} \operatorname{sign}(\lambda_l) \cdot \min_{l=0, l\neq k} (|\lambda_l|). \tag{3}$$

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 d_c. It can be further optimized by multiplying each outgoing message with a message scaling factor (MSF) of 0.75. For lower code rates the communications performance strongly degrades.

D. Layered Decoding

Layered decoding applies a different message schedule than the classical two-phase decoding. It was originally proposed by Mansour [15] and denoted as turbo decoding message passing (TDMP), then it was referred to as layered decoding by Hocevar [16]. The basic idea is to process a subset of CNs and to pass the newly calculated messages immediately to the corresponding VNs. The VNs update their outgoing messages in the same iteration. The next CN subset will thus receive newly updated messages which improves the convergence speed and therefore increases communications performance for a given number of iterations [17]. In Section VI we present a partly-parallel architecture for layered decoding, processing each of these subsets in parallel.

V. Iteration Control

Iterative decoding algorithms show an inherent dynamic behavior, i.e. the number of iterations strongly depends on the signal-to-noise ratio which changes over time. Instead of using a fixed number of iterations, the number of iterations can be controlled by an intelligent iteration control mechanism. In [18] it was shown that iteration control is the most efficient technique for energy saving in a turbo decoder system without sacrificing communications performance. The arguments are also valid for LDPC decoding.

An efficient iteration control has to distinguish between decodable and undecodable blocks. LDPC codes already provide an implicit stopping criterion for decodable blocks by simply checking for an all-zero syndrome if a codeword has been successfully decoded, see Equation 1. Normally only this stopping criterion is used in state-of-the-art implementations. More advanced stopping criteria tackle especially the undecodable codewords. As soon as a codeword is most certainly too corrupted to be successfully decoded, the decoder stops

its decoding process. This approach puts the LDPC decoder in idle mode (i.e. clock gating is activated on gate level) in low SNR regions instead of wasting the maximum number of iterations all the time and thus wasting energy. We apply the stopping criterion presented in [19] to evaluate the achievable energy savings for the WIMEDIA UWB environment. This stopping criterion is based on a monitoring of the variable nodes reliability (VNR).

$$\Lambda^m = \lambda_{ch}^m + \sum_{l=0}^{d_v^m-1} \lambda_l^m, \quad VNR = \sum_{m=0}^{N-1} |\Lambda^m| \tag{4}$$

The VNR is the sum of the absolutes of all intermediate VN results. The calculation of this value has a very low implementation complexity. The decoding process is stopped if the VNR does not change or decreases within two successive decoding iterations. The stopping criterion has to be switched off when the VNR passes a threshold (VNR_{off}) which is SNR dependent. This threshold has to be determined only once for each code rate [20]. With an appropriate VNR_{off} the loss in communications performance can be decreased to zero. For more detailed information on this criterion the reader is referred to [20].

VI. LDPC Decoder Architectures

This chapter gives a summary of the decoder architecture, more detail can be seen in [2]. A partly parallel architecture template is sufficient to ensure the throughput for the UWB LDPC decoder. A parallelization of P is established, which means that P edges are processed per clock cycle. To ensure code rate and codeword size flexibility the functional nodes are realized in a serial manner. Thus each functional unit can accept one message per clock cycle as already mentioned. The mapping of the VNs and CNs to the functional units is determined by the given code structure. Efficient mapping of the nodes to the functional units can be always guaranteed by designing LDPC codes using permuted identity matrices. Always I VNs/CNs with the connection described by an I-by-I identity matrix are allocated to I distinct VFU and I distinct CFU respectively. This was shown by many LDPC decoder realizations ([21][22][23][24]) as well as in the case of DVB-S2 LDPC decoding [25]. The parallelization P is determined by the identity matrix size I [2].

A permutation network has to realize the permutation of the identity matrix what can be done by a barrel shifter. To reach reasonable communications performance and minimize the area consumption, the check nodes are implemented with an optimized version of the Min-Sum algorithm (Section IV-C).

It applies layered decoding as described in Section IV-D at which the VN processing can be done within the check node block (CNB). This is achieved by storing the input information of the CFU in a FIFO and adding the new extrinsic information to the corresponding input message in the FIFO. Thus the correct a posteriori information is passed back to the channel memory which holds always the updated VN information. This basic concept was first presented in [15].

The input channel values as well as the extrinsic information in the message RAM are represented by 6 bit values, an a posteriori message in the channel RAM is represented by 9 bit. This quantization parameters could be changed to further reduce the resulting VLSI area at the possible cost of communications performance.

VII. Results

In this section simulation results of iteration control in the WIMEDIA UWB simulation chain are given. Furthermore implementation results on the rapid prototyping platform and energy measurement results are presented.

A. Simulation Results

We selected the 16-QAM case with an air data rate of 960 Mbit/s and the 128 point FFT for our simulations. The input quantization is 6 Bit for the LDPC and 4 Bit for the Viterbi decoder, which leads to a communications performance loss less than 0.2 dB in comparison to a floating point implementation. This is a good compromise between implementation complexity and communications performance degradation.

Figure 3 shows the average number of iterations for various SNRs with the inherent LDPC stopping criterion only and the combined criteria for decodable and undecodable blocks presented in the previous section. We can make the following observations:

- In the error floor SNR region (above 23 dB) the inherent stopping criterion reduces the number of iterations for decodable blocks to 3.
- In the waterfall SNR region (15 to 23 dB) the combined criteria yields a maximum average number of iterations of only 5.5. That means that the iteration control mechanism can save up to 32% of iterations which yields a substantial energy saving.
- The detection mechanism for undecodable blocks gives the highest gain in the non-convergence SNR region (below 15 dB). It saves up to 4.5 iterations, i.e. 45%, assuming a maximum number of 8 iterations.

The threshold VNR_{off} for the iteration control for undecodable blocks is selected in such a way that the communications performance loss is lower than 0.01 dB, which is depicted in Figure 4. This figure shows that the coding gain of the LDPC code compared to the convolution code is more than 4 dB at a PER of 10^{-3} for this configuration setup.

B. Implementation Results

As already addressed in the introduction, we used a rapid prototyping environment for energy estimation. This rapid prototype environment is based on a Virtex-4 FPGA. To allow a fair comparison with our LDPC decoder, we used the Xilinx Viterbi IP core [26]. The Xilinx Viterbi decoder is a highly optimized IP block. Both decoders were configured such that they show the same throughput.

The Viterbi and LDPC decoder were both implemented using the Xilinx ISE 9.1 tool chain. High effort and area optimization was selected in conjunction with a 100 MHz

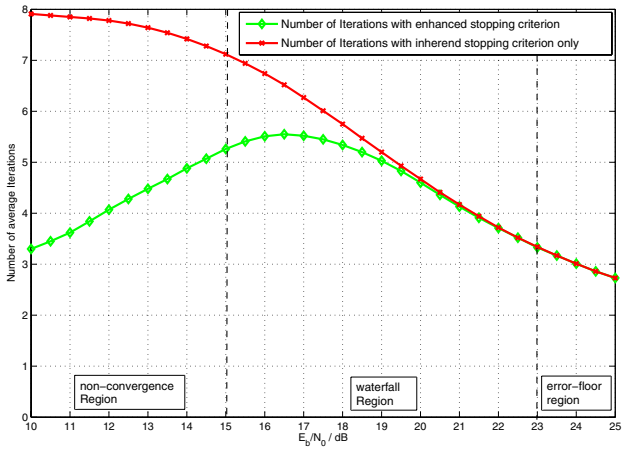

Fig. 3. Average number of iteration of the LDPC decoder vs. SNR

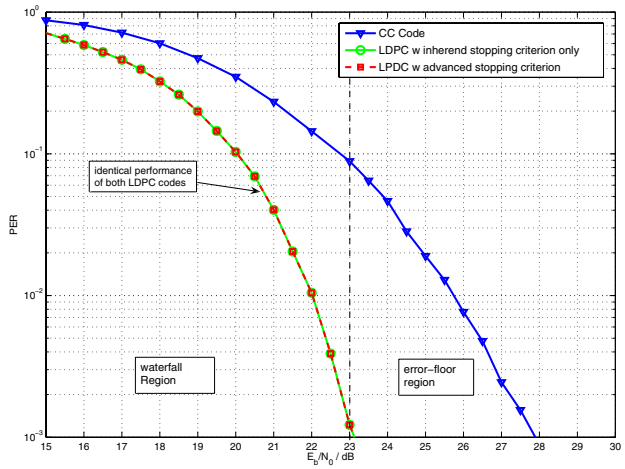

Fig. 4. Communications performance of LDPC code with and without advanced stopping criterion vs. CC code

clock frequency constraint. Table II shows the results for the LDPC decoder and the design space parameters. Table III shows the corresponding parameters for the CC decoder. Input and output RAMs which have the same size for both decoders are included in these numbers.

Both decoders achieved an equal net throughput of 100 Mbit/s, or 1 bit/clock cycle. Since the LDPC decoder is highly scalable, its throughput can be easily increased [2].

C. Energy Measurements Results

We measured the core power of the decoders including the input and output RAMs for each decoder respectively. Before starting the measurements the input RAM was loaded with representative data extracted from the UWB simulation chain. After loading we started the decoding process and measured the power and the decoding time to calculate the energy consumption, i.e. reading the input RAMs and writing the results into the output RAM are included in the corresponding energy budget.

Figure 5 shows the measured energy consumption of the two decoders. Since the LDPC decoding is an iterative process, we measured the energy for varying numbers of decoding iterations, ranging from 2 to 8. It can be seen that the LDPC

LDPC Code	$f_{[3,2]} = \{3/4, 1/4\}$ $g_{[11]} = \{1\}$
Codeword Size	**1200 bit**
Code Rate	**3/4**
Parallelism	30
Quantization	6 bit
Algorithm	MinSum+MSF, Layered Decoding
Iterations	8
Comm. Perform.	**see Figure 4, [2]**
Xilinx FPGA Virtex-4 XC4VLX60-10 @100 MHz	
LUTs	9,497
FlipFlops	4,310
Slices	5,996
BRAMs	13
Throughput	100 Mbit/s

TABLE II
IMPLEMENTATION RESULTS FOR UWB-LDPC DECODER

CC Code	non-systematic rate 1/3
Codeword Size	**1200 bit**
Code Rate	**1/3** to **3/4**
Implementation style	parallel
Input Quantization	4 bit
Algorithm	Soft input Viterbi
Comm. Perform.	**see Figure 4, [2]**
Xilinx FPGA Virtex-4 XC4VLX60-10 @100 MHz	
LUTs	3,547
FlipFlops	1,537
Slices	2,613
BRAMs	3
Throughput	100 Mbit/s

TABLE III
IMPLEMENTATION RESULTS FOR XILINX CC VITERBI DECODER

decoder consumes the same energy as the CC running with 7.5 iterations. In Figure 6 the mean energy consumption for decoding per block vs. the singal to noise ratio is figured out. In case of using only the inherent stopping criterion the LDPC code only in the non-convergence region needs more energy for decoding in mean. Combining the energy/iteration with the iteration numbers of Figure 3, the LDPC decoder with iteration control consumes in all cases less energy in mean than the Viterbi decoder for decoding one codeword. Moreover the convolutional decoder requires a channel interleaver to reach a reasonable communications performance. This interleaver consumes additional energy which has to be added to the Viterbi energy consumption of the CC itself.

Summarizing the energy consumption of the LDPC decoder is smaller than the energy consumption of the Viterbi decoder for the same throughput assuming realistic data sets. Since we measured the energy on an FPGA rapid prototyping board, the absolute values can not be directly transferred to an ASIC implementation. However it is feasible to conclude that the relative energy behavior of both decoders in an ASIC implementation is the same, i.e. the LDPC decoder consumes less energy than a Viterbi decoder in an ASIC implementation, too.

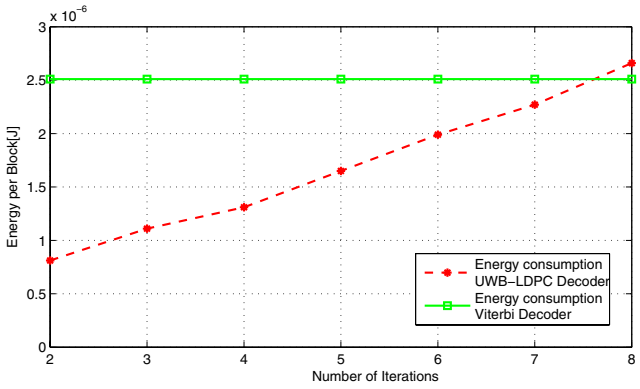

Fig. 5. Energy for UWB-LDPC and Viterbi Decoder, Energy per Block

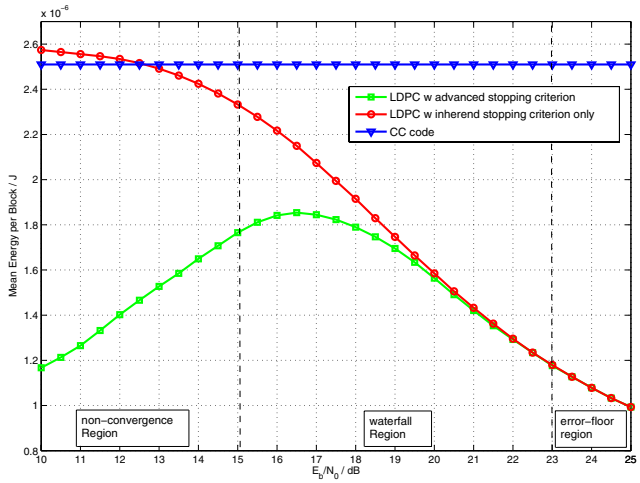

Fig. 6. Energy consumption vs. SNR for UWB-LDPC and Viterbi Decoder

VIII. Conclusions

We presented an implementation complexity comparison with special emphasis on energy consumption between an LDPC and a Viterbi decoder for the WIMEDIA UWB system. A sophisticated iteration control for the LDPC decoding process can drastically reduce the number of decoding iterations and consequently the energy. The LDPC decoder with iteration control requires in average less energy per codeword than the Viterbi decoder and gains 4 dB in sense of communications performance at a PER of 10^{-3}.

It is a feasible assumption that the energy ratio measured on a prototyping platform can be transferred to an ASIC implementation. Thus the ultra sparse LDPC codes [2][6] are good candidates for channel codes of the future WIMEDIA UWB standard, not only from an communications point of view but also from an energy point of view.

IX. Acknowledgement

This work has been partly carried out in the framework of the IST project PULSERS Phase II (FP6-027142), which is partly funded by the European Union.

References

[1] Standard ECMA-368, "High Rate Ultra Wideband PHY and MAC Standard." [Online]. Available: http://www.ecma-international.org

[2] T. Brack, F. Kienle, T. Lehnigk-Emden, M. Alles, N. Wehn, and F. Berens, "Enhanced Channel Coding for OFDM-based UWB Systems," in *Proc. International Conference on Ultra-Wideband (ICUWB 2006)*, Waltham, Massachusetts, Sept. 2006, pp. 255–260.

[3] S.-M. Kim, J. Tang, and K. K. Parhi, "Quasi-Cyclic Low-Density Parity-Check Coded Multiband-OFDM UWB Systems," in *IEEE International Symposium on Circuits and Systems*, May 2005, pp. 65–68.

[4] K.-B. Png, X. Peng, and F. Chin, "Performance Studies of a Multiband OFDM System Using a Simplified LDPC Code," in *IEEE Ultra-Wideband Systematic and Technologies 2004*, May 2004.

[5] J. Tang, T. Bhatt, and V. Stolpman, "Modified Min-Sum Algorithm for LDPC Decoders in UWB Communications," in *Proc. IEEE International Conference on Ultra-Wideband 2006*, Boston, Massachusetts, USA, Sept. 2006.

[6] M. Alles, T. Brack, T. Lehnigk-Emden, F. Kienle, N. Wehn, F. Berens, and A. Ruegg, "A Survey on LDPC Codes and Decoders for OFDM-based UWB Systems," in *Proc. 56th Vehicular Technology Conference (VTC Spring '07)*, Dublin, Ireland, Apr. 2007, to appear.

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

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

[9] 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.

[10] IEEE 802.16e, "Air Interface for Fixed and Mobile Broadband Wireless Access Systems," IEEE P802.16e/D12 Draft, oct 2005.

[11] IEEE 802.11n, "Wireless LAN Medium Access Control and Physical Layer specifications: Enhancements for Higher Throughput," IEEE P802.16n/D1.0, Mar 2006.

[12] J. K. Omura, "On the Viterbi decoding algorithm," *IEEE Transactions on Information Theory*, vol. 15, pp. 177–179, Jan. 1969.

[13] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of Capacity-Approaching irregular Low-Density Parity-Check Codes," *IEEE Transaction on Information Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[14] 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*, Brest, France, Sept. 2003, pp. 451–454.

[15] M. Mansour and N. Shanbhag, "High-Throughput LDPC Decoders," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 11, no. 6, pp. 976–996, Dec. 2003.

[16] D. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," in *Proc. IEEE Workshop on Signal Processing Systems (SiPS '04)*, Austin,USA, Oct. 2004, pp. 107–112.

[17] J. Zhang and M. Fossorier, "Shuffled Iterative Decoding," *IEEE Transactions on Communications*, vol. 53, pp. 209–213, Feb. 2005.

[18] M. J. Thul, T. Vogt, F. Gilbert, and N. Wehn, "Evaluation of Algorithm Optimizations for Low-Power Turbo-Decoder Implementations," in *Proc. 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02)*, Orlando, Florida, USA, May 2002, pp. 3101–3104.

[19] N. Wehn, F. Kienle, and T. Brack, "Method and device for controlling the decoding of a LDPC encoded codeword, in particular for DVB-S2 LDPC encoded codewords," European Patent Application No.05 009 477.0, Apr. 2005.

[20] F. Kienle and N. Wehn, "Low Complexity Stopping Criterion for LDPC Code Decoders," in *Proc. 2005-Spring Vehicular Technology Conference (VTC '05 Spring)*, Stockholm, Schweden, May 2005, pp. 606–609.

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

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

[23] M. Mansour and N. Shanbhag, "Architecture-Aware Low-Density Parity-Check Codes," in *Proc. 2003 IEEE International Symposium on Circuits and Systems (ISCAS '03)*, Bangkok, Thailand, May 2003.

[24] F. Kienle and N. Wehn, "Design Methodology for IRA Codes," in *Proc. 2004 Asia South Pacific Design Automation Conference (ASP-DAC '04)*, Yokohama, Japan, January 2004, pp. 459–462.

[25] 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, pp. 1530–1535.

[26] Xilinx Inc., "Viterbi Decoder V6.0, Data Sheet."