Low Complexity Stopping Criteria for UMTS Turbo-Decoders

Frank Gilbert, Frank Kienle, Norbert Wehn
Microelectronic System Design Research Group, University of Kaiserslautern
Erwin-Schroedinger-Strasse, 67663 Kaiserslautern, Germany
Email: {gilbert, kienle, wehn}@eit.uni-kl.de

***Abstract*— Turbo-Codes are part of the third generation wireless communications system (UMTS). A Turbo-Decoder consists of two soft-in soft-out component decoders, which exchange information (soft-values) in an iterative process. The number of iterations for decoding strongly depends on the channel characteristic which can change from block to block due to fading. In this paper, we present two new stopping criteria which can be implemented on dedicated hardware or DSP with negligible overhead. The new criteria operate on the sum of the absolute soft output values, calculated after each component decoder and is referred to as sum reliability. We compare the communications performance and average number of iterations of our proposed criteria to other criteria in literature using a fixed-point 8-state Turbo-Decoder implementation in an UMTS FDD-Downlink chain. An analysis of the arithmetic complexity and memory demand yields minimal overhead with excellent performance compared to other stopping criteria.**

I. Introduction

The outstanding forward error correction provided by Turbo-Codes, which were introduced in 1993 [1], made them part of today's communication standards, e.g. UMTS [2]. They consist of concatenated component codes that work on the same block of information bits, separated by interleavers. Key to the performance of Turbo-Codes is the iterative exchange of interleaved information between the component decoders. The iterative nature of the decoding process yields a high computational complexity. The design metrics latency and energy consumption increase linearly with the number of decoding iterations. On the other hand the decoder has to perform a certain number of iterations before reaching a satisfactory degree of confidence. The number of iterations for decoding strongly depends on the channel characteristic, which can change from block to block due to fading. In some cases, successful decoding is impossible even with an infinite number of iterations. Therefore, selecting an appropriate value for the number of iterations requires a *trade-off between decoding performance and implementation complexity*. Standard Turbo-Decoders are implemented with a fixed number of iterations (typically between 5 to 10 iterations).

We propose to use an *iteration control criterion* with a variable number of iterations. For each datablock, the number of iterations performed is determined by the number of passes before a certain condition or rule for stopping is satisfied. The stopping condition determines, whether a datablock is already successfully decoded or if the datablock is to disturbed to be decoded error free at all. It is computed based on information available during decoding. At the end of each iteration, the decoder performs a check on the condition for stopping. If the condition is true, the iterative process is terminated, and the decoded data sequence is send to the decoder's output. Otherwise the iterative process continues.

Research on iteration control criteria (see Section III) has focused on detecting convergence, *i.e.* successful decoding. The resulting criteria are able to reduce the average number of iterations for improved channel characteristics (*i.e.* a high SNR region). We argue, that iteration control criteria should also consider undecodable datablocks. For low SNR with a fair amount of undecodable datablocks high effort is *wasted* by the decoder, as the maximum number of iterations is carried out. Detecting undecodable datablocks early in the decoding process would allow to reduce the average number of iterations over all channel characteristics.

This work has been partially supported by the *Deutsche Forschungsgemeinschaft (DFG)* under Grant We 2442/1-2

Efficient iteration control for both scenarios - successful and unsuccessful decoding - is important for efficient Turbo-Decoder implementations. It will reduce decoding delay and can be exploited in several ways:

- low power/low energy operation by shutting down idle blocks or by voltage scheduling techniques [3], [4],
- throughput increase by starting decoding of the next block earlier [5],
- decoding performance improvement by spending more iterations on blocks that are expected to be decodable with more iterations than a previously used fixed number of iterations [6].

Of course the benefits have to offset the time and energy spend on iteration control. The stopping rule of a iteration control criteria should be computable easily from data available during normal decoding operation.

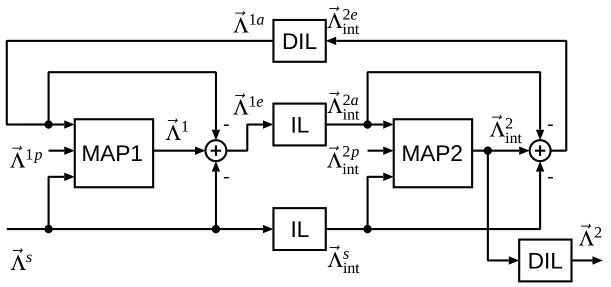
In this paper we present new stopping criteria and compare these to other criteria in literature using a fixed-point 8-state Turbo-Decoder implementation regarding three aspects: the communications performance has to fulfill the requirements of the UMTS standard, the number of normalized iterations has to be as small as possible and an implementation complexity with negligible overhead.

II. System Model

Channel coding in general enables error correction in the receiver side by introducing redundancy (*e.g.* parity information) in the encoder. In Turbo-Codes, the original information, denoted as *systematic information* ($\vec{x}^s$), is transmitted together with the *parity information* ($\vec{x}^{1p}, \vec{x}^{2p}_{int}$). For UMTS [2], the Turbo-Encoder consists of two recursive systematic convolutional (RSC) encoders with constraint length $K_c = 4$. Both encoder work on the same block of information bits; for UMTS the blocklength (K) is in the range from 40 to 5114.

One RSC encoder works on the block of information in its original sequence. The second encoder has the same structure but works on the original data in a different order (an interleaved sequence). In a puncturing unit the code rate R_c can be adjusted.

The Turbo-Decoder receives three sequences of *logarithmic likelihood ratios* (LLRs) Λ^s_k, Λ^{1p}_k, and $\Lambda^{2p}_{k,\mathrm{int}}$ according to $\vec{x}^s, \vec{x}^{1p}, \vec{x}^{2p}_{int}$. For every RSC encoder a corresponding component decoder exists, see Figure 1. Decoding is an iterative process with the exchange of reliability information. In every iteration each decoder calculates for every received bit a LLR as soft-output (reliability information). The soft-output of each component decoder ($\vec{\Lambda}$) is modified to reflect only its own confidence ($\vec{\Lambda}^e$) in the received information bit. The sign of each LLR indicates the received information bit of being sent either



DIL
IL
IL
MAP1
MAP2
DIL
$\vec{\Lambda}^{1a}$
$\vec{\Lambda}^{2e}_{\text{int}}$
$\vec{\Lambda}^{1e}$
$\vec{\Lambda}^{2a}_{\text{int}}$
$\vec{\Lambda}^{1}$
$\vec{\Lambda}^{2}_{\text{int}}$
$\vec{\Lambda}^{1p}$
$\vec{\Lambda}^{2p}_{\text{int}}$
$\vec{\Lambda}^{s}$
$\vec{\Lambda}^{s}_{\text{int}}$
$\vec{\Lambda}^{2}$

Fig. 1. Turbo-Decoder

as “0” or “1”, the absolute values are measures of confidence in the respective 0/1-decision. The *maximum a posteriori* (MAP) decoder has been recognized as the component decoder of choice as it is superior to the *Soft-Output Viterbi Algorithm* (SOVA) in terms of communications performance and implementation scalability [7]. The first MAP-Decoder (MAP1) works on the received bits in the original sequence, the second decoder (MAP2) works on the received bits in the interleaved order. In Figure 1 interleaving and deinterleaving is performed by the blocks IL and DIL respectively. The exchange continues until a stop criterion is fulfilled. The last soft-output is not modified and becomes the soft-output of the Turbo-Decoder ($\vec{\Lambda}^2$).

For the following investigations, we used a simulation model implemented in Synopsys CoCentric System Studio to simulate the Turbo-Decoder performance. The used Turbo coding system complies with the UMTS standard [2]. The encoder consists of two identical RSC encoders with constraint length $K_c = 4$, $M_c = 2^{K_c-1} = 8$ states, 13/15 generator polynoms, and an UMTS-compliant interleaver. The total code rate of the used Turbo-Encoder is $R_c = \frac{1}{3}$.

The Turbo-Decoder is implemented using fixed-point arithmetic. The notation $(q,\ f)$ describes the fixed-point representation of a number: q represents the total bitwidth of the fixed-point number, f is the bitwidth of the fractional part. For the input signal $\vec{\Lambda}^s$, $\vec{\Lambda}^{1p}$, and $\vec{\Lambda}^{2p}_{\text{int}}$ a (6,2) quantization scheme is used. From these input signals, the branch metrics are calculated exactly. The calculation of the state metrics is implemented with a (12,2) quantization and modulo arithmetic [6] as a renormalization scheme to avoid overflows. Finally, the extrinsic information $\vec{\Lambda}^{1e}$, $\vec{\Lambda}^{2e}$ and the LLRs $\vec{\Lambda}^1$, $\vec{\Lambda}^2$ are implemented with a (7,2) quantization and saturation arithmetic.

III. Related Work

Early iteration control criteria are based on the principle of cross entropy minimization between the probability- or LLR distributions. Hagenauer used in [8] a threshold value on the cross entropy between the first and second component decoder's soft outputs to stop iterations. Shao *et al.* devised in [9] two more simple criteria which are related to cross entropy minimization. Both are based on monitoring the number of sign changes of the soft-output Λ^2 and extrinsic information Λ^{2e} respectively of subsequent iterations and compare them to a threshold. A similar criterion was proposed by Wu and Woerner in [10], there the number of different signs between *a priori* and extrinsic information of one component decoder is calculated and compared to a threshold.

Cyclic Redundancy Check (CRC) was employed in [11] to reduce the average number of iterations. Transmitting the CRC slightly decreases the rate R_c. Furthermore, the CRC polynomial has to be chosen carefully, as the residual error rate of the CRC influences the decoding performance.

Wang and Parhi proposed in [12] so-called *decoding metrics*, comprising the minimum of the extrinsic information $|\Lambda^{2e}|$, the minimum of the soft-output $|\Lambda^2|$, the number of non-matching signs between extrinsic information and soft-output, and the comparison of the number of decoded ‘1’ in datablock of successive iterations. Matache *et al.* presented in [13] a thorough analysis of different stopping rules for iteration control. They distinct between *hard rules*, which are based on comparing decoded bits (hard bit decisions), and *soft rules*, which are based on comparing reliabilities (soft-output) with a threshold. Soft rules comprise among others of the mean and the minimum of the soft-output $|\Lambda^2|$.

We proposed in [6], [3] for the first time a criterion that can be used to detect - with high reliability - both successfully decoded and undecodable datablock within a reasonable number of iterations. We calculate the mean absolute value μ_i (Equation 6) of iteration i and monitor its value over the iterations. We observed that μ_i increases from iteration to iteration as the decoding process is improving its results. If the datablock has been decoded, or further iterations cannot correct the remaining errors, μ_i saturates. Therefore we can use the μ_i as an progress indicator of the decoding process. This criterion was independently proposed in [13], [14], [15] and in [15] named as *mean-reliability*.

IV. Iteration Control Criteria

In this section we define our proposed stopping rules for iteration control: Sum-Reliability and Combined Minimum LLR and Sum-Reliability. Furthermore, different stopping rules from literature are described as references.

Some of the following rules use a threshold θ. This threshold is rule dependant, even if the same symbol is used for all rules. In the results presented in Section V the used value for θ will be explicitly stated. The chosen thresholds were determined by simulation, with the objective of covering a reasonable range of undetected frame-error-rates for each rule.

A. New Iteration Control Criteria Definition

Sum-Reliability (Sum). We propose to calculate the *sum-reliability* instead of the mean-reliability to *avoid a costly division* operation. After each iteration i the sum of the absolute values of the LLRs is calculated:

$$S_i = \sum_{k=1}^{K} |\Lambda_k^{2,i}|. \tag{1}$$

The decoding process is stopped after iteration i for $i \geq 2$, if

$$S_i - S_{i-1} \leq 0. \tag{2}$$

Combined Minimum LLR and Sum-Reliability (Comb). After each iteration i the sum of the absolute values of the LLRs is calculated as in Equation 1. The decoding process is stopped after iteration i for $i \geq 2$, if

$$S_i - S_{i-1} \leq 0 \quad \text{or} \quad \min_{1\leq k\leq K} |\Lambda_k^{2,i}| > \theta. \tag{3}$$

The threshold $\theta = 7.75$ used in our simulations was chosen as $\frac{1}{2}$ of the dynamic range of the LLRs' absolute values.

B. Reference Iteration Control Criteria

Cross Entropy (CE). As presented in [8], the cross entropy can be used to stop the iteration process. Based on the assumptions presented in [16] the CE of the iteration i can be approximated by:

$$T(i) \approx \sum_{k=1}^{K} \frac{(\Lambda_k^{2e,i} - \Lambda_k^{2e,i-1})^2}{e^{|\Lambda_k^{1,i}|}}. \tag{4}$$

The decoding process is stopped after iteration i for $i \geq 2$, if

$$\frac{T(i)}{T(1)} < \theta, \tag{5}$$

where $T(1)$ is the approximated CE after the first iteration.

Mean-Reliability (Mean). In this work, we refer to stopping the iteration with the mean-reliability as described in [13] and similarly presented in [14]. After each iteration i the mean of the absolute values of the LLRs is calculated:

$$\mu_i = \frac{1}{K} \sum_{k=1}^{K} |\Lambda_k^{2,i}|. \tag{6}$$

The decoding process is stopped after iteration i for $i \geq 1$, if

$$\mu_i > \theta. \tag{7}$$

Minimum LLR (Min). Stopping the iteration based on evaluating the minimum LLR was described in [13], [12]. The decoding process is stopped after iteration i for $i \geq 1$, if

$$\min_{1 \leq k \leq K} |\Lambda_k^{2,i}| > \theta. \tag{8}$$

Sign-Change Ratio (SCR). The SCR criterion is related to cross entropy as described in [16]. SCR evaluates a decision function f^{SCR} after each iteration i for all $k \in \{1 \ldots K\}$, which evaluate the sign change of the extrinsic information of successive iterations:

$$f_i^{SCR}(k) = \begin{cases} 0, \text{ if } sign(\Lambda_k^{2e,i}) = sign(\Lambda_k^{2e,i-1}) \\ 1, \text{ else} \end{cases}. \tag{9}$$

The decoding process is stopped after iteration i for $i \geq 2$, if

$$\frac{1}{K} \sum_{k=1}^{K} f_i^{SCR}(k) < \theta, \tag{10}$$

where $\sum f_i^{SCR}(k)$ corresponds with the number of sign changes between $\Lambda^{2e,i}$ and $\Lambda^{2e,i-1}$.

Sign-Difference Ratio (SDR). In [10] the SDR is calculated as the number of different signs between *a priori* and extrinsic information of one component decoder. SDR evaluates a decision function f^{SDR} after each iteration i for all $k \in \{1 \ldots K\}$:

$$f_i^{SDR}(k) = \begin{cases} 0, \text{ if } sign(\Lambda_k^{2a,i}) = sign(\Lambda_k^{2e,i}) \\ 1, \text{ else} \end{cases}. \tag{11}$$

The decoding process is stopped after iteration i for $i \geq 1$, if

$$\frac{1}{K} \sum_{k=1}^{K} f_i^{SDR}(k) < \theta. \tag{12}$$

Hard-Decision Aided (HDA). This criterion proposed by [16] compares the decoded bits of two successive iterations. The decoding process is stopped after iteration i for $i \geq 2$, if

$$sign(\Lambda_k^{2,i}) = sign(\Lambda_k^{2,i-1}) \quad \forall\, k \in \{1 \ldots K\}. \tag{13}$$

CRC Rule (CRC). A separate error-detection code, such as a CRC, can be concatenated as an outer code with an inner Turbo-Code in order to flag erroneous decoded sequences. The decoding process is stopped after iteration i whenever the syndrome of the CRC is zero. The 16 bit CRC code used detects at least 99.9985 % of all frame errors.

"Magic Genie" (M). For lower bounds on the required iterations, we add an additional unrealizable stopping rule. For this rule, the magic genie immediately recognizes the correct decoded word, based on foreknowledge of the transmitted bit sequence, and stops the iterative process in exactly the minimum number of iterations required. If a datablock can not be decoded with the maximum number of iterations, magic genie does not start any iteration at all.

To prevent an endless loop, if the stopping rules are never satisfied, decoding is finally stopped after a maximum of 10 iterations. After decoding has stopped (in any case), the final hard-decisions are compared to the original bit sequence for Frame-error-rate (FER) computation.

V. Results

Frame-error-rate (FER) and average iterations were simulated with a FDD-Downlink chain according to the UMTS standard [2]: 3GPP TS 25.101 V5.3.0 (2002-06). It is implementing a complete wireless communications environment. Some of the reference measurement channels, representing example configurations of radio access bearers for different data rates, were simulated as examples of a realistic communications environment. In the following, we present results based on a multi-path fading channel (Case 3, Test 12). This configuration simulates a 384 kbit/s data service with a blocklength of 3840 data-bits and additional 16 bit CRC, leading to a total blocklength of 3856 bit. Note, that similar results were achieved by simulation of an *Additive White Gaussian Noise* (AWGN) channel model.

The communications performance of the *M* and the *CRC* criteria is identical to a 10 iterations decoder in our simulations. In the following, for the comparison of the communications performance we therefore use a 10 iterations decoder as a reference. For comparison of the average iterations, we calculate the normalized iterations in relation to 10 iteration (= 100 %). The normalized iteration of the *M* criterion is depicted as a lower bound. Finally, the computational complexity of the stopping rules is compared.

A. Communications Performance

Our simulations confirm the iteration control criteria reported in literature. The communications performance of all criteria (see Figure 2) are nearly identical. Note, that Figure 2 only shows a subset of the iteration control criteria: the communications performance of $CE_{\theta=0.001}$, $Min_{\theta=7.75}$, and $SDR_{\theta=0.0001}$ is comparable to 10 iterations in Figure 2; $Comb_{\theta=7.75}$ and *HDA* is comparable to *Sum* in Figure 2. Additionally, the requirements of the UMTS standard are depicted.

There is one exception to the results in literature: the *Mean* criterion does not show the in [13], [14] described behavior, due to our fixed-point Turbo-Decoder model. The LLR calculation is implemented with a (7,2) quantization and saturation arithmetic. Therefore, all LLRs are in the range from -16 to 15.75. This leads to a mean value ranging from 15.75 to 16 even for perfectly decoded datablocks, depending on the number of "0" and "1" in the original bit sequence.

B. Average Iterations

The iteration control criteria presented in literature are able, as expected, to reduce the average iterations for high SNR in our simulation environment (see Figure 3). The average iterations of $CE_{\theta=0.001}$, $\text{SCR}_{\theta=0.005}$, and *HDA* are nearly identical. *Min* shows the best results for high SNR, as it requires on average only $\frac{1}{2}$ iterations more than the *M* criterion. For low SNR, only the *Sum* criterion based on the sum-reliability reduces the average iterations. For high SNR, however, it needs an additional iteration than the other iteration control criteria. *Comb* combines the advantages of *Sum* and *Min*:for low SNR the sum-reliability detects with a high probability undecodable datablocks within the first four iterations. For high SNR, successful decoding is reliable detected by the minimum LLR.

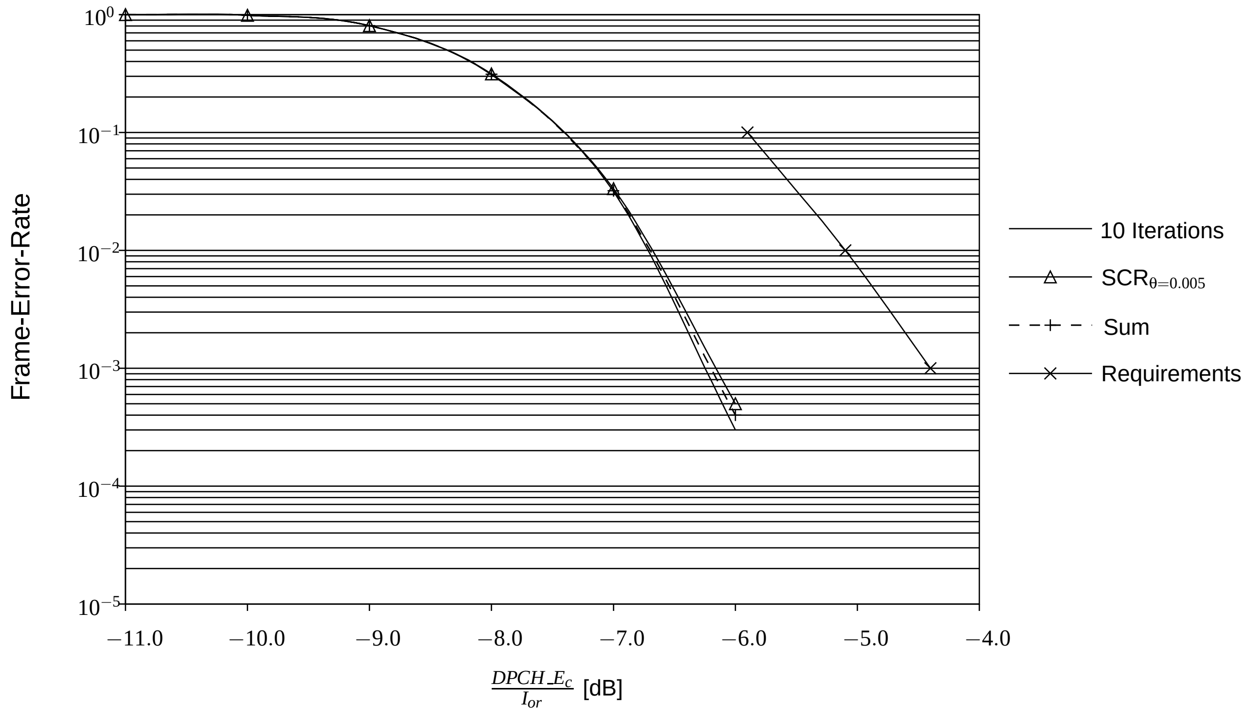


Fig. 2. Communications performance of iteration control criteria compared to 10 iterations fixed

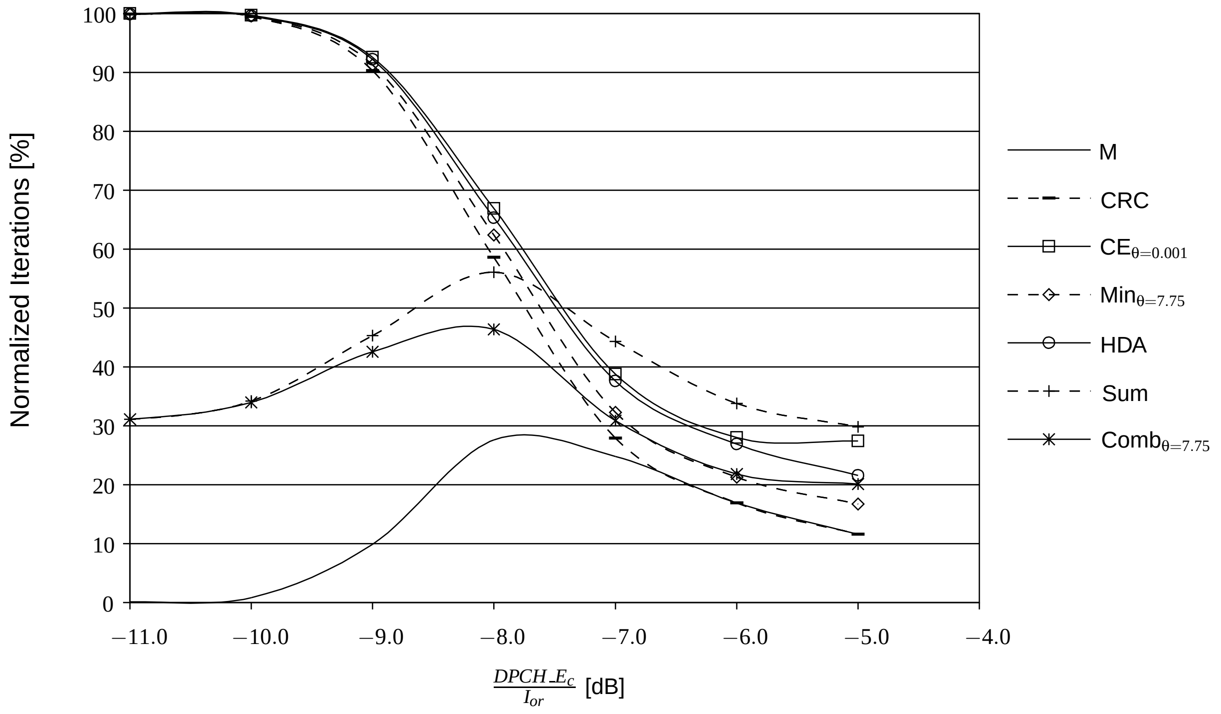


Fig. 3. Normalized iterations of iteration control criteria compared to the CRC and M rule (10 iterations = 100 %)

Criterion	Addition	Mult.	Memory	Impl. Complexity
CE	$3K$	$3K$	$2K+\mathrm{LUT}$	very high
Mean	$2K$	1	-	low
Min	$2K$	-	-	very low
SCR	$2K$	1	$1K$	high
SDR	$2K$	1	-	low
HDA	$2K$	-	$1K \cdot 1\,\mathrm{bit}$	medium
CRC	$3K$	-	-	very low
Sum	$2K$	-	1	very low
Comb	$3K$	-	1	very low

TABLE I

COMPARISON OF IMPLEMENTATION COMPLEXITY OF STOPPING RULES, WHERE K IS THE BLOCKLENGTH AND LUT A LOOKUP-TABLE FOR THE VALUES OF $\frac{1}{e^x}$

In [10], for *CE* also a reduction of the required iterations for low SNR is reported. It is argued, that Λ_k^s and Λ_k^p dominate the computation as Λ_k^a is usually small for low SNR. Therefore, the extrinsic information should stabilize quickly, leading to a termination of the decoding by the *CE* rule. However, in our simulations the extrinsic information shows no convergence for most of the datablocks in a low SNR region. The extrinsic information floats around zero and does not stabilize over iterations, therefore the *CE* rule never stops the decoding. We consider this as an effect of the limited accuracy of our fixed-point model, due to quantization.

C. Implementation Complexity of Stopping Rules

We assume an efficient implementation in hardware or software, *i.e.* a fixed-point implementation, where each component decoder has only access to its input *a priori* information and its calculated extrinsic information and LLR. Any further information required by any stopping rule leads to additional memory demand.

We analyze the number of operations similar to an addition (addition, subtraction, increment, comparison, shift, XOR, minimum, and absolute value), considering them as very low complexity operations. More complex operation, such as multiplication or division, should be avoided. If occurring only once per iteration, we consider them as low complexity. Furthermore, we analyze the additional memory for the storage of information between successive iterations. Storing a single value is neglectable, if a block of values is required we consider this as medium to high complexity, depending on the size of the memory. The memory demand for intermediate values calculating the rules are neglected.

Table I illustrates the implementation complexity and rates the rules from "very low" to "very high" complexity. The most complex iteration control criterion is *CE* followed by *SCR* and *HDA* as they require additional memory. *Min*, *SDR*, *Sum*, and *Comb* have the lowest implementation complexity.

CRC also has a low implementation complexity, especially in hardware (a simple shift register). However, there are some implementation related problems. First, the *CRC* has to be computed after the second component decoder, but MAP2 works on the interleaved data sequence. As the *CRC* depends on the order of the input bits, it can not be calculated on-the-fly during decoding but must be calculated after deinterleaving. Second, *CRC* calculation is serial by nature and therefore not suited for implementation in parallel decoding architectures.

VI. CONCLUSION

The iteration control criteria *Min*, *Sum*, and *SDR* are best suited for implementation, as they have a low complexity, reduce the average iteration considerably with a low degradation in the communications performance. Only the *Sum* criterion, based on the sum-reliability, reduces the iteration over the complete SNR range. However, this criterion requires more iterations for high SNR, otherwise it would be ideal for iteration control. For high SNR, *Min* is best to reduce the average iterations. Consequently, the combined criterion *Comb* gives the best results of all iteration control criteria with minimal implementation overhead.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes," in *Proc. 1993 International Conference on Communications (ICC '93)*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] Third Generation Partnership Project, "3GPP home page," `www.3gpp.org`.

[3] F. Gilbert, A. Worm, and N. Wehn, "Low Power Implementation of a Turbo-Decoder on Programmable Architectures," in *Proc. 2001 Asia South Pacific Design Automation Conference (ASP-DAC '01)*, Yokohama, Japan, Jan. 2001, pp. 400–403.

[4] F. Gilbert and N. Wehn, "Voltage Scheduling Algorithms for UMTS Turbo-Decoding on Variable Supply Voltage Processors," in *Kleinheubacher Berichte*, vol. 45, Oct. 2002, pp. 215–218.

[5] A. F. J. Vogt, "Inreasing Troughput of Iterative Decoders," *IEEE Electronic Letters*, vol. 37, no. 12, pp. 770–771, June 2001.

[6] A. Worm, H. Michel, F. Gilbert, G. Kreiselmaier, M. J. Thul, and N. Wehn, "Advanced Implementation Issues of Turbo-Decoders," in *Proc. 2nd International Symposium on Turbo Codes & Related Topics*, Brest, France, Sept. 2000, pp. 351–354.

[7] J. Vogt, K. Koora, A. Finger, and G. Fettweis, "Comparison of Different Turbo Decoder Realizations for IMT-2000," in *Proc. 1999 Global Telecommunications Conference (Globecom '99)*, vol. 5, Rio de Janeiro, Brazil, Dec. 1999, pp. 2704–2708.

[8] J. Hagenauer, E. Offer, and L. Papke, "Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 429–445, Mar. 1996.

[9] R. Y. Shao, M. C. P. Fossorier, and S. Lin, "Two Simple Stopping Criteria for Turbo Decoding," in *Proc. 1998 IEEE International Symposium on Information Theory (ISIT)*, Cambridge, Massachusetts, USA, Aug. 1998, p. 279.

[10] Y. Wu, B. D. Woerner, and W. J. Ebel, "A Simple Stopping Criterion for Turbo Decoding," *IEEE Communications Letters*, vol. 4, no. 8, pp. 258–260, Aug. 2000.

[11] A. Shibutani, H. Suda, and F. Adachi, "Reducing Average Number of Turbo Decoding Iterations," *Electronic Letters*, vol. 35, no. 9, pp. 701–702, Apr. 1999.

[12] Z. Wang and K. K. Parhi, "Decoding Metrics and their Applications in VLSI Turbo Decoders," in *Proc. 2000 Conference on Acoustics, Speech, and Signal Processing (ICASSP '00)*, Sept. 2000, pp. 3370–3373.

[13] A. Matache, S. Dolinar, and F. Pollara, "Stopping Rules for Turbo Decoders," *TMO Progress Report 42–142*, Aug. 2000, `http://tda.jpl.nasa.gov/progress_report/`, Jet Propulsion Laboratory, Pasadena, California.

[14] F. Zhai and I. J. Fair, "New Error Detection Techniques and Stopping Criteria for Turbo Decoding," in *Proc. 2000 IEEE Canadian Conference on Electrical and Computer Engineering*, Halifax, Canada, Mar. 2000, pp. 58–62.

[15] I.Land and P. Hoeher, "Using the Mean Reliability as a Design and Stopping Criterion for Turbo Codes," in *Proc. Information Theory Workshop (ITW)*, Cairns, Australia, Sept. 2001.

[16] R. Y. Shao, S. Lin, and M. C. P. Fossorier, "Two Simple Stopping Criteria for Turbo Decoding," *IEEE Transactions on Communications*, vol. 47, no. 8, pp. 1117–1120, Aug. 1999.