Design of Fixed-Point Processing Based Turbo Codes Using Extrinsic Information Transfer Charts

Liang Li, Robert G. Maunder, Bashir M. Al-Hashimi and Lajos Hanzo
School of Electronics and Computer Science, University of Southampton, SO17 1BJ, United Kingdom

***Abstract*—The operand-width specifications in fixed-point hardware implementations of turbo code decoders is an important design issue, since this governs the trade-off between the decoder's performance and its complexity, cost, area and energy consumption. The investigation of this issue would be extremely time-consuming in the conventional approach, which relies upon Monte-Carlo simulation based Bit Error Ratio (BER) analysis. In this paper, we propose a generic design method, which uses EXtrinsic Information Transfer (EXIT) chart analysis to simplify this design process. Our method is not only an order of magnitude faster than the conventional Monte-Carlo simulation based approach, but also offers deeper insights into why performance degradations are imposed by insufficient operand-width specifications. The benefits of our generic method are demonstrated in the context of a turbo decoder, allowing accurate specifications to be obtained and compared to those suggested by previous works.**

I. Introduction

Owing to their near-capacity performance, turbo codes [1] have been adopted in various wireless communication standards, including the CCSDS [2], UMTS [3], CDMA2000 [4], WiMAX [5] and LTE systems [6]. However, turbo codes are characterised by an iterative operation of parallel concatenated Logarithmic Bahl-Cocke-Jelinek-Raviv (Log-BCJR) decoders [7], which demands a high clock frequency, a high number of computing operations and a large memory space. Since low energy consumption and low cost are desirable in wireless applications, typically fixed-point implementations of the Log-BCJR are employed to mitigate these demands in Digital Signal Processor (DSP), Field Programmable Gate Array (FPGA) and Application Specific Integrated Circuit (ASIC) implementations [8]. More specifically, compared to floating-point implementations, the fixed-point approach requires fewer clock cycles and a lower complexity per Log-BCJR operation, as well as less storage memory [9]. Furthermore, at the cost of reducing the turbo decoder's performance, these gains accrue as the operand-width of the fixed-point representation is reduced. More specifically, the hardware complexity, memory requirement and hence the energy consumption are proportional to the operand-width of the data, since this determines the width of the databus, memory and computing resources in the datapath structure [10]. Indeed, it has been shown in [11] that each additional bit in the operand-width may results in approximately a 25% increase in the decoder's area and power consumption.

Previous efforts have been focused on determining the operand-width specifications that strike an attractive trade-off between the complexity and performance of turbo decoders [12], [13]. Naturally, these studies [8], [9], [11], [14]–[18] have been conducted for the Universal Mobile Telecommunications System's (UMTS) turbo decoder [3]. However, no universal conclusion has been obtained owing to the large variation in simulation parameters that may be considered. These parameters include the choice of modulation scheme, channel conditions, the operand-width used for the integer and fraction parts of the various fixed-point Log-BCJR variables and the method used to handle fixed-point overflows [11]. In previous efforts, Bit Error Ratio (BER) analysis has been employed to investigate the performance degradation caused by limited operand-widths in fixed-point implementations. However, this approach requires lengthy simulations, preventing comprehensive investigations into the interplay of diverse parameters, in various different simulation scenarios. Furthermore, BER analysis can only reveal the magnitude of the performance degradation imposed by limited operand-widths, but it cannot reveal the true cause of this degradation.

In this paper, we propose a method based on EXtrinsic Information Transfer (EXIT) chart [19] analysis for determining attractive operand-width specifications for fixed-point turbo decoders. A particular advantage of our method is that EXIT chart analysis only requires the consideration of a single Log-BCJR decoder, at a single channel Signal to Noise Ratio (SNR); it does not require the iterative operation of two decoders to be simulated over a range of SNRs, which is in contrast to the corresponding BER analysis. As a result, EXIT charts can be obtained using an order of magnitude less simulations than BER plots [19]. Owing to its lower simulation complexity, our approach allows comprehensive studies into the attractive operand-widths of the Log-BCJR variable to be readily conducted. Moreover, EXIT chart reveals the convergence behaviour of the iterative decoder [20]. By observing the influence of the different fixed-point specifications on EXIT chart, our method indicate the reasons why performance degradations are caused by limiting the operand-width of the Log-BCJR variables, as will be discussed in Section III. This can further expedite the process of finding attractive operand-widths by avoiding a comprehensive simulation of all the possible specifications, which is again, in contrast to the conventional approach. Since a simulation for generating an EXIT chart requires significantly less computing complexity than generating a BER chart, and our method allows to avoid enumerating all the possible specifications in experiments, only a small percentage of simulation time of conventional approach could be expected for our method. Note that EXIT charts were first used to characterise the performance of fixed-

978-1-4244-3574-6/10/$25.00 ©2010 IEEE

point implementations of turbo codes in [21]. However, this was not in the context of finding attractive operand-widths, as we propose here.

The rest of this paper is organised as follows. In Section II, our simulation scenario, its parameter values and the corresponding EXIT charts are briefly introduced. In Section III, we demonstrate our process for determining attractive operand-width specifications for the UMTS turbo decoder. Finally, we offer our conclusions in Section IV.

II. EXIT CHART AIDED TURBO DECODER DESIGN EXAMPLE

Figure 1 provides the schematic of the 1/3-rate UMTS turbo code [3], as a design example. The turbo encoder comprises a parallel concatenation of two identical terminated Recursive Systematic Convolutional (RSC) codes, which are separated by the prime block interleaver π [3]. This rearranges the order of the bit sequence $\mathbf{a}$ in order to generate the sequence $\mathbf{b}$ of Figure 1. These sequences provide the inputs of the convolutional encoder shift registers. The resultant parity bit sequences, $\mathbf{c}$ and $\mathbf{d}$ respectively, are multiplexed with the systematic sequence $\mathbf{a}$, as shown in Figure 1.

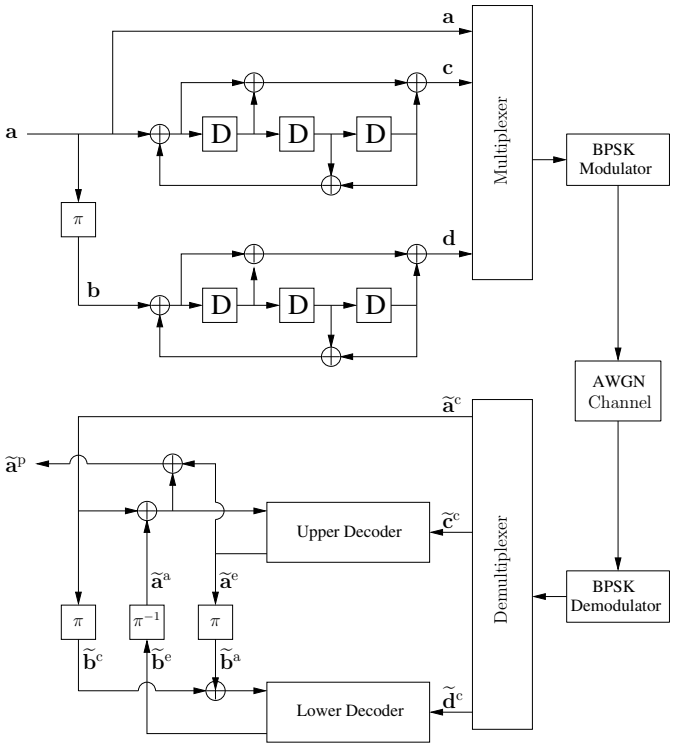

Fig. 1. Schematic of the UMTS turbo decoder. Note that unlike those of the turbo decoder, the additions performaned in the encoder are module-2.

For simplicity and compatibility with [8], [9], [11], [14]–[18], we consider the use of Binary Phase Shift Keying (BPSK) modulation and an Additive White Gaussian Noise (AWGN) channel in our simulations. The soft-decision BPSK demodulator provides the three Logarithmic-Likelihood Ratio (LLR) sequences, $\tilde{\mathbf{a}}^{c}$, $\tilde{\mathbf{c}}^{c}$ and $\tilde{\mathbf{d}}^{c}$, which pertain to the bit sequences having the corresponding notation. Turbo decoding proceeds with an iterative exchange of increasingly reliable extrinsic LLR sequences between the concatenated Soft-Input Soft-Output (SISO) Log-BCJR decoders, until convergence is achieved. More specifically, the extrinsic LLR sequence $\tilde{\mathbf{a}}^{e}$ generated by the upper decoder is interleaved and used as the *a priori* LLR sequence $\tilde{\mathbf{b}}^{a}$ during the operation of the lower decoder of Figure 1. The resultant extrinsic LLR sequence $\tilde{\mathbf{b}}^{e}$ is deinterleaved and used as the *a priori* LLR sequence $\tilde{\mathbf{a}}^{a}$ by the upper decoder in the next decoding iteration, as shown in Figure 1. Upon achievement of iterative decoding convergence, the *a posteriori* LLR sequence $\mathbf{a}^{p}$ is obtained by summing the *a priori* and extrinsic LLR sequences $\mathbf{a}^{a}$ and $\mathbf{a}^{e}$, respectively.

The EXIT functions [19] of the UMTS turbo decoder using the conventional floating-point Log-BCJR and a channel SNR of -4.83 dB are shown using solid lines in Figure 2. These functions characterise the relationship between the reliability of the extrinsic LLR sequences that are iteratively exchanged by the upper and lower decoders of Figure 1. Here, the reliability is quantified by the Mutual Information (MI) between the corresponding sequences of bits and LLRs, where higher values in the range of [0,1] indicate a higher reliability. The upper EXIT function in Figure 2 plots the MI $I(\tilde{\mathbf{a}}^{e}, \mathbf{a})$ of the extrinsic LLRs $\tilde{\mathbf{a}}^{e}$ produced by the upper decoder as a function of the MI $I(\tilde{\mathbf{a}}^{a}, \mathbf{a})$ of the *a priori* LLRs $\tilde{\mathbf{a}}^{a}$. Similarly, the lower EXIT function plots $I(\tilde{\mathbf{b}}^{e}, \mathbf{b})$ versus $I(\tilde{\mathbf{b}}^{a}, \mathbf{b})$ for the lower decoder of Figure 1, but on interchanged axes. Note that the two EXIT functions are mirror images of each other, since the two decoders in Figure 1 are identical. Also note that the position of the intersection point of the curves quantifies the decision reliability and hence the BER that may be achieved upon convergence, while the width of the tunnel between the curves determines the number of decoding iterations that are required to achieve this [19].

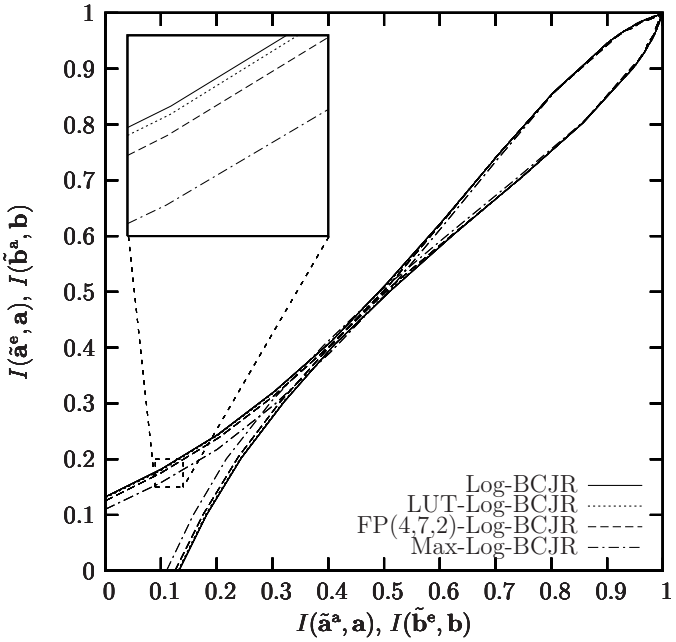

Fig. 2. EXIT functions for various approximations of the Log-BCJR, using a 453-bit UMTS turbo code and a channel SNR of -4.83 db.

In practice, a sub-optimal approximation of the conventional floating-point Log-BCJR can be employed in order to reduce

the turbo decoder's computational complexity. For example, the LUT-Log-BCJR [7] uses an eight-entry Look-Up-Table (LUT) to approximate the Jacobian logarithm that is employed in the conventional Log-BCJR. Additionally, the Max-Log-BCJR achieves an even more dramatic decoding complexity reduction by further approximating the Jacobian logarithm by considering only its most influential, ie. maximum contributor, as detailed in [7]. The channel SNR required for the scheme of Figure 1 to achieve a BER of 10^{-5} is up to 0.54 dB higher when the Max-Log-BCJR is employed instead of the Log-BCJR [22], which may be considered to be significant. By contrast, the SNR degradation imposed by the LUT-Log-BCJR is considered to be insignificant. These degradations are reflected by the narrower widths of the corresponding EXIT chart tunnels shown in Figure 2.

For the reasons described in Section I, fixed-point implementations of the Log-BCJR are capable of providing even more significant complexity reductions than the above-described floating-point approximations. Note that fixed-point implementations can be considered to naturally extend these floating-point approximations. This is because the quantisation imposed by fixed-point implementations effectively decomposes the Jacobian logarithm into a LUT or maximum operation. In this paper, we consider the signed two's complement fixed-point representation since this can naturally mitigate the problems associated with overflow in fixed-point Log-BCJR (FP-Log-BCJR) decoders[1] [11]. More specifically, the Log-BCJR decoding process is only concerned with the *difference* between pairs of its intermediate variables, not their absolute values [23]. In the two's complement representation, an overflow will not affect the difference between two intermediate variables that is obtained by two's complement subtraction, provided that the true difference is not greater than the largest value that can be represented by the specified fixed-point operand-width. In this event, overflow will cause corruption and degrade the performance of the Log-BCJR decoder. For this reason, we additionally clip the values of the LLRs as they are input to the BCJR decoders of Figure 1, since this can further mitigate the overflow problem[2] [16], as we shall show in Section III-B.

III. Results

In this section, we detail the process, the simulation results and the related analysis that we used to identify attractive operand-width parameters for fixed-point implementations of the UMTS turbo decoder. The performance of fixed-point implementations of the Log-BCJR is affected by two key issues, namely the precision and the dynamic range of the fixed-point representation. The precision is determined by the operand-width of the fraction part of the fixed-point two's complement representation, while the dynamic range is determined by the operand-width of the integer part. In the two's complement representation, the integer and the fraction parts separated by an imaginary point of fractionation. For example, if the binary sequence 101011 has an integer operand-width of 4 bits and a fraction operand-width of 2 bits, then it represents 1010.11, which is -5.25 in the decimal domain.

We commence our investigations, by independently considering the performance degradation associated with limited accuracy fraction and integer operand-widths in Section III-A and Section III-B, respectively. By investigating the effects of finite-precision number-representation on the corresponding EXIT chart, we separately find the shortest fraction and integer operand-widths that do not impose any significant performance degradation. In Section III-C the combined effect of limited fraction and integer operand-widths is investigated in order to validate our arguments.

In the following discussions, we use the notation FP(x, y, z)-Log-BCJR to specify the parameterisation of the fixed-point approximation of the Log-BCJR decoder. In the notation, z is the fraction's operand-width of our fixed-point representation, while y is the integer operand-width, including the sign bit. The parameter $x \leq y$ identifies the number of bits that the integer part of the fixed-point LLRs is clipped to, when they are input into the Log-BCJR decoder. As described in Section II, this is necessary to mitigate the effects of overflow. For example, in the FP(3, 4, 2)-Log-BCJR, an a priori LLR having the value 1011.11 would be clipped to 1100.00, which is -4 in its decimal representation.

A. Fraction operand-width

In order to investigate the degradation caused by a limited fraction operand-width z in isolation, initially, we considered a long integer operand-width, comprising $y = x = 32$ bits. This 32-bit integer operand-width may be considered to offer an unimpaired dynamic range, since it is sufficiently high to eliminate overflow in the fixed-point approximation of the Log-BCJR. Therefore, we use the notation FP(∞, ∞, z)-Log-BCJR in this section. Note that when z tends to infinity, the performance of the FP(∞, ∞, z)-Log-BCJR approaches that of the Log-BCJR. When $z = 3$, the FP(∞, ∞, z)-Log-BCJR represents a fixed-point version of the LUT-Log-BCJR, since this uses an $2^3 = 8$-entry LUT. Finally, when $z = 0$, the FP(∞, ∞, z)-Log-BCJR represents a fixed-point version of the Max-Log-BCJR.

Figure 3 compares the EXIT charts of the Log-BCJR and FP(∞, ∞, z)-Log-BCJR schemes, where $z \in [0, 3]$. Here, we consider a channel SNR of -4dB, since the resultant open EXIT chart tunnel allows performance degradations to be readily characterised, and a block length of 453 bits, since this is the geometric average of the minimum and maximum

[1]Other techniques for handling overflows in fixed-point implementations of the Log-BCJR include subtractive normalisation and saturation [11], [13]. These techniques facilitate even lower operand-width requirements than the two's complement approach, but they require additional operations in the Log-BCJR and potentially increase its complexity. The EXIT chart analysis proposed in this paper can be readily applied to determine the associated operand-width requirements.

[2]Some previous works have achieved clipping by using reduced operand-widths for the fixed-point representation of LLRs, as well as for some of the Log-BCJR's internal variables [14], [15]. This approach can reduce the memory requirements in principle. However, in practice, a constant operand-width specification is typically required for the hardware implementation of the datapath and memory. This is because complicated operand-width management would otherwise be required to achieve the described reductions, which would increase the turbo decoder's complexity. While these techniques are not considered in this paper, the associated operand-width requirements can be readily determined using our EXIT chart approach.

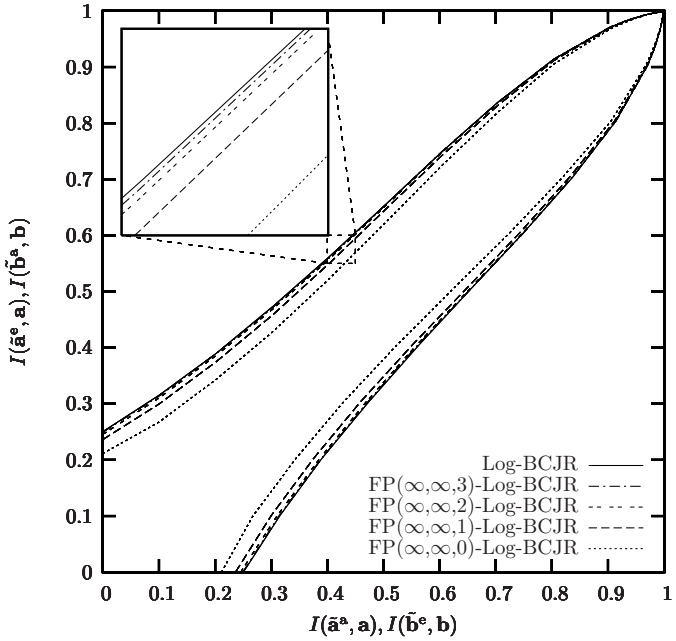

Fig. 3. EXIT functions for various fixed-point approximations of the Log-BCJR, using a variety of operand-widths for the fraction part of the fixed-point representation, a 453-bit UMTS turbo code and a channel SNR of -4 dB.

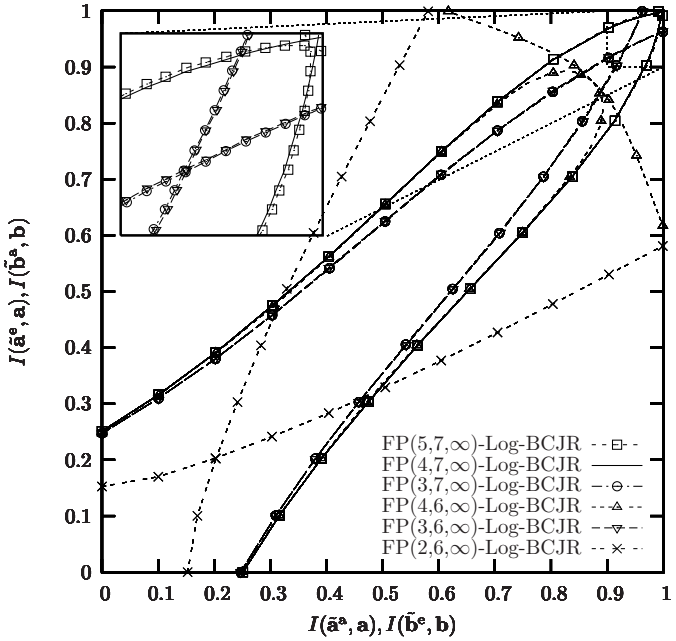

Fig. 4. EXIT functions for various fixed-point approximations of the Log-BCJR, using a variety of operand-widths for the integer part of the fixed-point representation, a 453-bit UMTS turbo code and a channel SNR of -4 dB.

blocks lengths in the UMTS standard. The simulation results of Figure 3 show that the open EXIT chart tunnel is narrowed as the fraction operand-width z is reduced. Like the LUT-Log-BCJR, the FP(∞, ∞, 2)-Log-BCJR does not impose a significant performance degradation compared to the Log-BCJR, as shown in Figure 3. For this reason and considering the trade-off between complexity and performance, we recommend the FP(∞, ∞, 2)-Log-BCJR specification. This conclusion agrees with those of [16] and [9]. By contrast, [16] and [18] recommended fraction operand-widths of $z = 1$ and $z = 3$, respectively.

B. Integer operand-width

Similar to the fraction operand-width investigation of Section III-A, the integer operand-width was investigated by using a $z = 32$ fraction operand-width, which may be considered to offer an unconstrained precision. We therefore use the notation FP(x, y, ∞) in this section. The simulation results of Figure 4 show that instead of just narrowing the EXIT chart tunnel, insufficient integer width specifications cause the tunnel to close completely, preventing iterative decoding convergence to a low BER. Furthermore, Figure 4 reveals that there are two distinct mechanisms that cause the tunnel to close, namely the "over-clipping" and "under-clipping" of the LLRs, when they are input to the Log-BCJR decoders.

1) Over-clipping: In the case of the FP(2, 6, ∞)-Log-BCJR, FP(3, 6, ∞)-Log-BCJR and FP(3, 7, ∞)-Log-BCJR, the gradient of the EXIT functions is reduced, causing them to intersect each other before the $(1, 1)$ point in the top-right hand corner of the EXIT chart, as shown in Figure 4. This performance degradation may be attributed to the information loss that results from the over-clipping of the LLRs as they are input into the Log-BCJR decoders. Hence, reduced EXIT function gradients indicate that x should be increased in order to reduce the associated information loss. Indeed, Figure 4 reveals that using $x \geq 4$ is sufficiently accurate in the case of the UMTS turbo decoder.

2) Under-clipping: By contrast, in the case of the FP(4, 6, ∞)-Log-BCJR and FP(5, 7, ∞)-Log-BCJR, the EXIT functions intersect each other before reaching the $(1, 1)$ point, because their gradients become negative at high values of $I(\tilde{\mathbf{a}}^{\mathrm{a}}, \mathbf{a})$. These high mutual informations are associated with *a priori* LLRs having high magnitudes, which may cause excessive overflow during the Log-BCJR, if their values are under-clipped relative to the integer operand-width y, as described in Section II. Hence, non-increasing EXIT functions indicate that $(y - x)$ should be increased in order to mitigate the effects of overflow. Indeed, Figure 4 reveals that $y - x \geq 3$ is sufficient in the case of the UMTS turbo code.

3) Desirable clipping: The conclusions of Sections III-B1 and III-B2 imply that the FP(4, 7, ∞)-Log-BCJR has the minimal integer operand-widths that avoid both the information loss of over-clipping and the overflow corruption of under-clipping. This is confirmed by the corresponding EXIT functions of Figure 4, which do not intersect before reaching the $(1, 1)$ point, where iterative decoding convergence to a low BER is facilitated. While this conclusion agrees with that of [15], the other previous works recommended different specifications, because they employ different fixed-point processing techniques, as described in the footnotes of Section II.

C. Combined fraction and integer operand-widths

The conclusions of Section III-A and III-B suggest that the FP(4, 7, 3)-Log-BCJR has the minimal fraction and integer operand-widths that do not impose a significant performance

degradation. This conclusion is confirmed by the corresponding EXIT functions of Figure 2. These were obtained for a channel SNR of -4.83 dB, which is the lowest SNR at which the corresponding EXIT chart tunnel remains open. By contrast, the floating-point Log-BCJR, LUT-Log-BCJR and Max-Log-BCJR respectively require SNRs of -4.87, -4.86 and -4.73 dB to achieve open tunnels. Since the performance degradation imposed by the FP(4, 7, 2)-Log-BCJR is significantly lower than that of the Max-Log-BCJR, we suggest that the FP(4, 7, 2)-Log-BCJR offers the most attractive trade-off between the turbo decoder's complexity and performance.

IV. Conclusion

In this paper, we have proposed the employment of EXIT chart analysis to find attractive parameterisations for fixed-point implementations of turbo decoders. In contrast to the conventional BER analysis based approach, our method does not require the iterative operation of both Log-BCJR decoders to be simulated over a range of channel SNRs. Instead, our simulations consider only a single decoder and the channel SNR. Furthermore, in addition to expediting the analysis of the performance degradation that is imposed by fixed-point implementations, our approach reveals the cause of this degradation. More specifically, EXIT chart analysis can reveal whether a fixed-point implementation of a turbo decoder is suffering from over-clipping, under-clipping or a lack of fixed-point precision, as discussed in Sections III-B1, III-B2 and III-A, respectively. This is possible because these problems affect the EXIT functions in different ways. By contrast, they all have the same effect on the BER, namely increasing it. While we have focused on fixed-point approximations of the Log-BCJR in this paper, the proposed EXIT chart approach may be used to analyse any practical fixed-point implementation of the Log-BCJR.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo codes," in *Proceedings of the IEEE International Conference on Communications*, vol. 2, Geneva, Switzerland, May. 1993, pp. 1064–1070.

[2] CCSDS, "Recommendation for space data system standards, TM synchronization and channel coding," National Aeronautics and Space Administration, Tech. Rep., Sep. 2003.

[3] *Universal Mobile Telecommunications System (UMTS); Multiplexing and Channel Coding (FDD)*, European Telecommunications Standards Institute Std., 1999.

[4] *Third generation partnership project 2 (3GPP2), Physical Layer Standard for CDMA2000 spread spectrum systems, Release D*, Telecommunications Industry Association Std., Feb. 2004.

[5] *IEEE Standard for Local and Metropolitan Area Networks. Part 16: Air Interface for Fixed Broadband Wireless Access Systems, IEEE802.16-2004*, IEEE Standards Association Std., Nov. 2004.

[6] J. Zyren, "Overview of the 3GPP long term evolution physical layer," Freescale Semiconductor, Tech. Rep., Jul. 2007.

[7] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proceedings of IEEE International Conference of Communication*, vol. 2, Seattle, WA, USA, June. 1995, pp. 1009–1013.

[8] H. Michel and N. Wehn, "Turbo-decoder quantization for UMTS," *IEEE Communication Letters*, vol. 5, no. 2, pp. 55–57, Feb. 2001.

[9] M. A. Castellon, I. J. Fair, and D. G. Elliott, "Fixed-point turbo decoder implementation suitable for embedded applications," in *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, Saskatoon, Canada, May. 2005, pp. 1065–1068.

[10] E. Boutillon, C. Douillard, and G. Montorsi, "Iterative decoding of concatenated convolutional codes: Implementation issues," in *Proceedings of the IEEE*, vol. 95, no. 6, 2007, pp. 1201–1227.

[11] A. Worm, H. Michel, F. Gilbert, G. Kreiselmaier, M. Thul, and N. Wehn, "Advanced implementation issues of turbo-decoders," in *Proceedings of the International Symposium on Turbo-Codes and Related Topics*, Brest, France, Sep 2000, pp. 351–354.

[12] Y. Wu, B. D. Woerner, and T. K. Blankenship, "Data width requirements in SISO decoding with modulo normalization," *IEEE Transactions on Communications*, vol. 49, no. 11, pp. 1861–1868, Nov. 2001.

[13] G. Masera, *Turbo Code Applications: a journey from a paper to realization*, K. Sripimanwat, Ed. Springer Netherlands, 2005.

[14] J. Hsu and C. Wang, "On finite-precision implementation of a decoder for turbo codes," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 4, Orlando, FL, USA, Jul. 1999, pp. 423–426.

[15] T. K. Blankenship and B. Classon, "Fixed-point performance of low-complexity turbo decoding algorithms," in *Proceddings of the IEEE Vehicular Technology Conference*, vol. 2, Rhodes, Greece, May. 2001, pp. 1483–1487.

[16] G. Montorsi and S. Benedetto, "Design of fixed-point iterative decoders for concatenated codes with interleavers," in *Proceedings of the Global Telecommunications Conference*, vol. 2, San Francisco, CA, USA, Nov. 2000, pp. 801–806.

[17] R. Hoshyar, A. R. S. Bahai, and R. Tafazolli, "Finite precision turbo decoding," in *Proceedings of the International Symposiumon on Turbo Codes and Related Topics*, Brest, France, Sep. 2003, pp. 483–486.

[18] A. Morales-Cortes, R. Parra-Michel, L. F. Gonzalez-Perez, and T. G. Cervantes, "Finite precision analysis of the 3GPP standard turbo decoder for fixed-point implementation in FPGA devices," in *Proceedings of the International Conference on Reconfigurable Computing and FPGAs*, Cancun, Mexico, Dec. 2008, pp. 43–48.

[19] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[20] L. Hanzo, O. R. Alamri, M. El-Hajjar, and N. Wu, *Near-Capacity Multi Functional MIMO Systems*. John Wiley and IEEE Press, 2009, ch. Turbo Detection of Channel-Coded STBC-SP Schemes, p. 100.

[21] B. Riaz and J. Bajcsy, "Impact of finite precision arithmetics on EXIT chart analysis of turbo codes," in *Proceedings of the IEEE Consumer Communications and Networking Conference*, Las Vegas, NV, USA, Jan. 2008.

[22] M. C. Valenti and J. Sun, "The UMTS turbo code and an efficient decoder implementation suitable for software-defined radios," *International Journal of Wireless Information Networks*, vol. 8, no. 4, pp. 203–215, Oct. 2001.

[23] A. Hekstra, "An alternative to metric rescaling in Viterbi decoders," *IEEE Transcations on Communications*, vol. 37, no. 11, pp. 1220–1222, Nov. 1989.