Journal of Engineering Science and Technology
Vol. 14, No. 5 (2019) 2808 - 2823
© School of Engineering, Taylor's University

A STRATEGICALLY ACCURATE AND OPTIMISED FPGA ARCHITECTURE FOR PERFORMANCE GAIN IN TURBO DECODERS IN DEEP SPACE NETWORK

LI LI LIM[1,2], DAVID WEE GIN LIM[1,*],
NAFIZAH GORIMAN KHAN[1], SOO YONG LIM[1]

[1]The University of Nottingham Malaysia Campus 43500 Semenyih, Selangor, Malaysia
[2]Tunku Abdul Rahman University College 53300 Setapak, Kuala Lumpur, Malaysia
*Corresponding Author: lim.wee-gin@nottingham.edu.my

Abstract

Motivated by the continual endeavour for improved coding gains in Deep Space Network (DSN) this paper re-examines the performance of suboptimal Logarithmic-Maximum A Posteriori (Log-MAP) Turbo decoders under poor Signal to Noise Ratio (SNR) and SNR mismatch conditions. Traditionally, the Log-MAP Turbo decoding forsakes the correction function computation in favour of hardware simplicity using only maximum operators known as the Max Log-MAP. However, this comes at the cost of decoder performance loss. Improved decoders reported in literature proposed various approximations to the correction function with limited reports on architectures to mitigate performance loss. In this paper, the link between the accuracy of the correction function to the Turbo decoder's performance is highlighted, and a proposal of an optimised Hybrid Log-MAP (HLM) architecture suitable for Software Defined Radio (SDR) based on recent Field Programmable Gate Array (FPGA) technology is presented. It is shown that with strategic computational accuracy of the suboptimal Turbo decoder offered by HLM, performance gain with close adherence to the ideal Log-MAP decoder can be achieved especially in poor channel conditions and errors in SNR estimations. The implementation utilises built-in high speed and optimised computation blocks. Comprehensive simulation results show that the proposed HLM algorithm performance borders on the ideal Log-MAP performance especially in very low SNR and in SNR mismatch conditions.

Keywords: Correction function, Log-MAP, SNR mismatch, Turbo codes.

1. Introduction

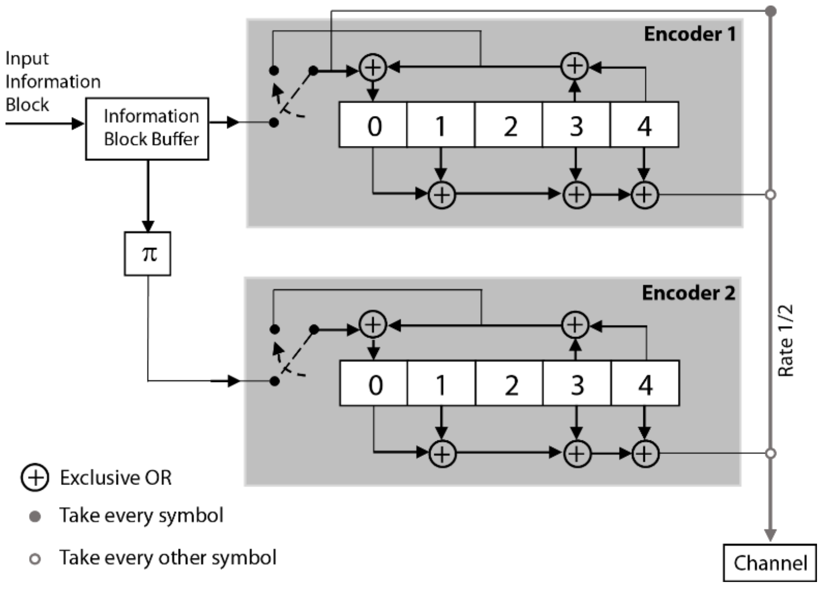
Turbo code has been touted as one of the best code for communication technology. Once, too complex for any practical application, but now the standard code for a reliable communication protocol. This development has been made possible due to advancement in Application Specific Integrated Circuit (ASIC) technology, which are applied to Digital Signal Processors (DSP) and Field Programmable Gate Array (FPGA), allowing smaller footprint hardware, and made significantly more affordable in recent years. The use of Software Defined Radio (SDR) leveraging FPGAs is quickly becoming commonplace as it is reconfigurable and adopts highly optimised logic and computational blocks with flexible interconnect.

The motivation of this work is driven by enterprising outer space missions that require for cutting edge, high-performance telecommunication channel to establish a reliable virtual presence throughout an outer space assignment. Deep Space Network (DSN) application is known to invest intensively on codes that could return even a slight additional coding gain to exploit the telecommunication link between probes and rovers in space missions. Since 2004, Turbo codes are implemented on the Messenger, Stereo, Mars Reconnaissance Orbiter and other missions up to present-day [1].

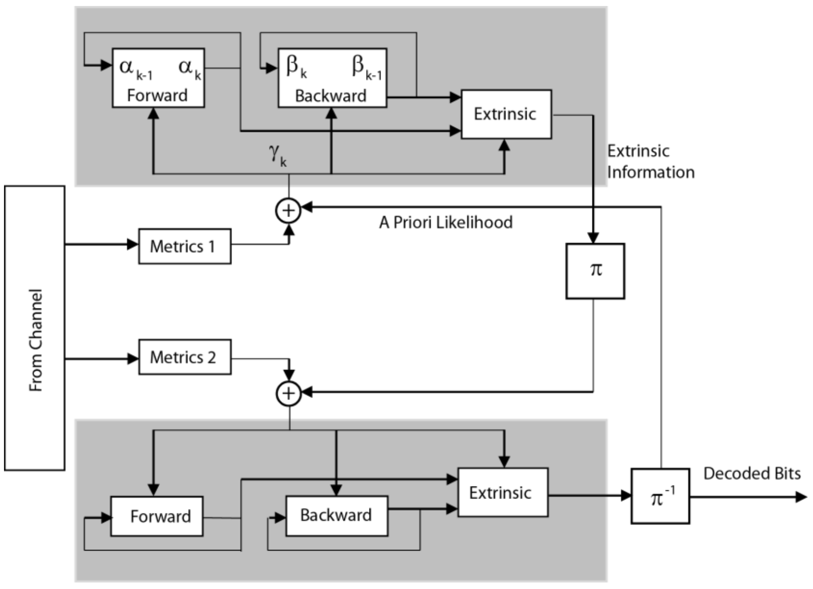
In this paper, an investigation was conducted on the feasibility of a suboptimal Hybrid Log-MAP implementation on FPGAs with improvement in coding gains, higher tolerance toward SNR estimation with optimisation in terms of speed and complexity. The Consultative Committee for Space Data System (CCSDS) standard for Turbo coding and decoding [2] as illustrated in Figs. 1(a) and (b) respectively, are used throughout this article.

Turbo codes, which were first developed by Berrou et al. [3] achieves near Shannon limit performance. However, this superior performance comes at the expense of complex computation such as the decoder metric value computation, representation of probabilities, and a high dependency on nonlinear exponential functions. Fortunately, these problems are mitigated when computations are shifted to the logarithmic domain. Multiplications diminish to addition, and complex exponential functions reduce to linear computation. However, not all exponential functions are simplified. The log of exponential sums kernel within the correction function of the max* operation remains very much needed and can be unacceptably complex on hardware. An overly simplified solution is to omit the correction term entirely, but this inevitably degrades the Bit-Error-Rate (BER) performance by up to 10% [4, 5]. The performance degradation is exacerbated under SNR mismatch, which often occurs in practice [6]. To improve performance, typically a lookup-table (LUT) with 8 to 16 values of the correction function is used [5]. Although simple in implementation, the LUT is inaccurate for a smaller argument of $|x|$ to the max* operator.

In the case of low SNR and SNR underestimation circumstances, the accuracy of the correction function is crucial for performance [7]. Furthermore, the correction function is an exponentially decaying function. Hence, computation accuracy for small values of $|x|$ is imperative for improved performance particularly in higher state Turbo decoders such as the CCSDS standard with 16 states [7]. Based on studies by Gross and Gulak [8], Talakoub and Shahrrava [9], Lim and Lim [10], in order to close the gap between the exact or ideal Log MAP and the Max-Log-MAP algorithm, other solutions in the literature attempt to approximate the correction function with robust implementation. This study focuses particularly on literatures with an accurate approximation for small values of $|x|$ with linear and reduced hardware complexity implementation. It is also observed that there is a gap in existing literatures for the implementation of these algorithms specifically on FPGA technology, which are used extensively in aerospace, broadcast, and consumer electronics due to its highly configurable, and hardware optimised properties for SDR.



(a) CCSDS turbo encoder.



(b) CCSDS turbo decoder.

Fig. 1. CCSDS turbo decoding architecture.

In this paper, this gap is addressed through the proposal of a new architecture of a Hybrid Log-MAP (HLM) algorithm, which approximates the correction term for strategically accurate and robust computation. It is shown that the HLM, albeit hardware efficient, achieves near-exact Log-MAP performance even at SNR underestimation, a feat that is not commonly accomplished by other linear approximation methods. An FPGA implementation of the HLM algorithm on the latest Xilinx's FPGA family of Virtex, Kintex, and Artix UltraSCALE$^+$ devices featuring the highly optimised DSP48E2 block [11] are also presented. Section 4 describes how the DSP48E2 block can be configured as the max* module within the Turbo decoder. Considering the vast availability of the DSP48E2 block even on the lower end FPGAs, the utilisation of this block can contribute a considerable significance for optimised speed, cost, and overall improved performance of the Turbo decoder. The rest of the paper is organised as follows. Section 2 details the problem statement and Section 3 reviews methods ranging from commonly used approximations to accurate implementation of the Log-MAP approximation algorithms in the literature. Section 4 describes the proposed Hybrid Log-MAP (HLM) approximation, and Section 5 presents the simulation results of the HLM in comparison with other approximation methods both with and without SNR mismatch. The paper is finally concluded in Section 6.

2. Problem Description

The Log-MAP implementation of the Turbo decoder addresses several issues but introduces the problem of log exponential sum computation.

$$\ln\left(\sum_{i=1}^{N} e^{x_i}\right), \quad N \in Z^+ \geq 2. \tag{1}$$

Implemented with the Jacobi logarithm, and considering only two variables involved Eq. (1) can be conveniently expressed as the max* operation, which consists of a maximum operation and a correction function, f_c.

$$\ln\left(e^{x_1} + e^{x_2}\right) = \max{}^*(x_1, x_2)$$

$$= \max(x_1, x_2) + \ln\left(1 + e^{|x_1 - x_2|}\right) \tag{2}$$

$$= \max(x_1, x_2) + f_c\left(|x_1 - x_2|\right) \tag{3}$$

The simplest solution to avoid the computational cost in calculating f_c in Eq. (3) is to simply omit the correction function, i.e., $f_c(x) = 0, \forall |x|$. This solution is known as the Max-Log-MAP solution, which is the simplest to implement but sacrifices performance gain for complexity due to its crude approximation.

where $f_c(x)$ in Eq. (3) is known as the correction term with $x = |x_1 - x_2|$, which as shown, retains the complexity of a log exponential sum. It can be shown that through a recursion of two variables, a longer chain of log exponential sum as in Eq. (1) can be easily computed as:

$$\ln\left(\sum_{i=1}^{N} e^{x_i}\right) = \max{}^*\left(x_N, \max{}^*\left(x_{N-1}, \cdots \max{}^*\left(x_3, \max{}^*\left(x_2, x_1\right)\right)\right)\right) \tag{4}$$

Even though the computation of $f_c(x)$ in Eq. (2) incurs the most cost, it should not be simply omitted if the BER is not to be compromised. The max* operation has to be computed in all nodes of the forward, α as well as backward, β metric computation. As such, it constitutes a large role in the decoder's hardware complexity. This is exacerbated with the increase in the number of decoder state. Thus, the implementation of the max* operation is crucial to the complexity and performance of the decoder and several methods of approximating the correction function are reviewed in the next section.

3.Review of Log-MAP

In this section, several linear approximations to the exact Log-MAP is reviewed. A graph depicting the ideal Log-MAP is shown in Fig. 2 and superimposed with the approximations are reviewed here.

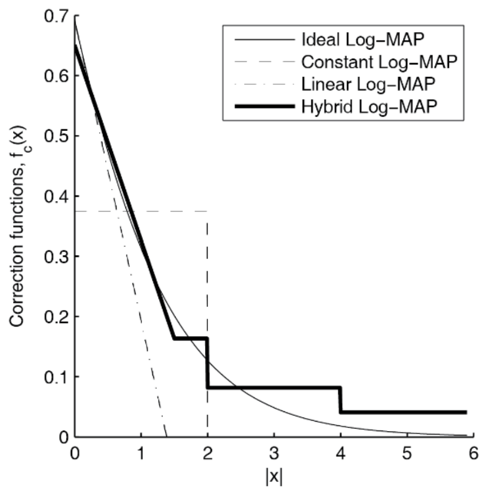


Fig. 2. Approximations to correction function.

Gross and Gulak [8] mentioned that one of the commonly used, Constant Log-MAP (CLM) algorithm that was first proposed in and implemented more recently in [4, 12-14]:

$$f_c(x) = \begin{cases} 3/8, & \text{for } 0 \le x < 2 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

CLM is simple to implement in hardware with a simple comparator but lacks accuracy, which in turn affects the BER performance of the decoder.

According to Talakoub and Shahrrava [9], another approximation, also known as the Linear-Log-MAP (LLM) extracts the first term of the MacLaurin Series expansion to yield a positive straight line as first introduced and used by Nguyen and Nguyen [4], Ivanov et al. [13] and Foda et al. [14]:

$$f_c(x) = \max\left(0, \ln 2 - \frac{x}{2}\right) \tag{6}$$

The Linear-Log-MAP is an attractive solution as it provides a closer approximation to the ideal Log-MAP. However, a much closer approximation is desirable, especially in the smaller arguments of $x = |x_1 - x_2|$ where $f_c(x)$ is more influential to accuracy. This is especially beneficial when the decoder is operating in low SNR conditions as well as when there are SNR mismatch.

Ivanov et al. [13] introduced a more recent linear approximation to the Log-MAP is the Piecewise Linear Log-MAP (PWL) and applied by Zhilyaev and Gurova [15] and Fedorenko et al. [16]. The PWL approximation divides the ideal Log-MAP into five equal proportioned sections within the range of $0 \leq |x| \leq 5$, which are in turn, approximated by five linear equation described in Table 1.

Table 1. Piecewise Linear Log-MAP (PWL) approximation [13].

Sub Region of $\|x\|$	**$f_c(x)$ Approximations**
0:1	$f_c(x) = -0.3792x + 0.6754$
1:15	$f_c(x) = -0.2229x + 0.5327$
1.5:2	$f_c(x) = -0.1483x + 0.4213$
2:3	$f_c(x) = -0.0773x + 0.2758$
3:4	$f_c(x) = -0.0300x + 0.1362$

It is noted that the PWL offers consistently good approximation to the ideal Log-MAP with the goodness of fit R-Squared measure of over 0.99 for each section. This work, however, did not offer many suggestions on hardware implementation aspects. By dividing the region for approximation into five sections, an increased number of serially cascaded comparators are required before the respective multiplication and addition can be performed. Additionally, the computational results are aided through the use of multiplexers, which incurs higher complexity, cost and delay for computation.

Liu et al. [17] presented another recent development in the effort to approximate the correction function using Multivariable Taylor Series (MTS). In this work, up to four arguments of $|x|$ are ranked, and considered for input to the correction function computation. For brevity, consider four $|x|$ argument as follows: $x_1 > x_2 > x_3 > x_4$. The minimum value, x_4 is simply discarded and the correction function reduces to [17]:

$$f_c(x) = \max{}^*(x_1, x_2, x_3) \tag{7}$$

$$= x_1 + \ln\left(1 + e^{x_2 - x_1} + e^{x_3 - x_1}\right) \tag{8}$$

with three or five multivariable Taylor series expansion points (EP) based on $|x_2 - x_1|$ and $|x_3 - x_1|$ as defined in Table 2.

Table 2. MTS approximation for three and five EPs [17].

Number of EPs	**$f_c(x)$ Approximations**
3	$\max(x_1, 1.0986 + 0.3333 * (x_1 + x_2 + x_3), 0.8853 + 0.4683 * (x_1 + x_2) + 0.0634 * x_3, 0.0634 * x_3, 0.6655 + 0.787 * x_1 + 0.1065 * (x_2 + x_3)$
5	$\max(x_1, 1.0986 + 0.3333 * (x_1 + x_2 + x_3), 0.8853 + 0.4683 * (x_1 + x_2) + 0.0634 * x_3, 0.7389 + 0.4954 * (x_1 + x_2) + 0.0092 * x_3, 0.6655 + 0.787 * x_1 + 0.1065 * (x_2 + x_3), 0.4411 + 0.8668 * x_1 + 0.1173 * x_2 + 0.0159 * x_3$

Complexity analysis of reviewed solutions

In general, each of the reviewed approximation shows a trade-off between complexity and accuracy for improved BER performance. Table 3 shows a summary of the hardware resources needed based on each mathematical equation given for each correction function approximation.

Table 3. Number of basic hardware resources for correction function computation.

Approximations	Look up table	Comparator	Multiplier	Addition/ Subtraction
CLM	1	1	0	0
LLM	1	1	1	1
PWL	1	4	5	5
MTS (3 EP)	1	3	5	9
MTS (5 EP)	1	5	10	15

It is observed that the CLM and LLM are the least complex suboptimal solutions for hardware implementation and that the computational cost increases with the PWL and MTS solutions. The effect of complexity indirectly affects the area, power utilisation, and latency of the Turbo decoder. Despite the PWL offering excellent approximation in terms of accuracy, the multi-region approximation introduces significant delay as the identification of regions can only be decided once the input $|x|$ has gone through a series of comparators before the respective linear computation can be performed. The MTS (3 EP) in comparison, does not differ far from the PWL and the MTS (5 EP) demands the highest computational resource in this review. Due to this, further comparison work with the PWL and MTS are omitted for an optimised hardware solution.

The following section details the development of a strategically accurate correction function approximation with optimised hardware for a low complexity solution.

4. Hybrid Log-MAP (HLM) Algorithm

Hardware efficient yet accurate approximation of the exact $f_c(x)$ is proposed in this section. As will be shown in Section 0, the HLM shows commendable tolerance to SNR underestimation compared to other algorithms. The HLM approximation is:

$$f_c(x) = \begin{cases} a - bx, & \text{for } x < 1.5 \\ c \cdot 2^{-\lfloor 0.5x \rfloor}, & \text{otherwise} \end{cases} \tag{9}$$

where $a = 0.6512$, $b = 0.3251$, and $c = 0.1635$ [10]. The values of a and b were obtained using a linear curve fitting method, specifically by determining a linear approximation with the least Sum of Squared Error (SSE). While the value of c was determined upon continuation from the linear function. The algorithm is a hybrid between a linear graph for $x < 1.5$ and a multistep function for values of $x \geq 1.5$. Before moving on, it is stressed that the accuracy of the approximation in this region of $x < 1.5$ is crucial when the SNR is low or when the SNR has been underestimated by the decoder. This is due to the fact that at low SNRs, the likelihood of $x < 1.5$ is much higher and that the value of $f_c(x)$ is more significant for $x < 1.5$. The goodness of fit statistics for the linear, as well as the overall region for all algorithms are reviewed in Tables 4 and 5 respectively alongside other algorithms.

Table 4. Goodness of fit statistics of HLM's linear graph portion for $x < 1.5$.

Goodness of fit parameters	Measure for various algorithms		
	HLM	Linear log-MAP	Constant log-MAP
Sum-of-Squared-Error (SSE)	0.05058	195.0223	317.6478
***R*-square**	0.9836	0.3546	-0.0512
Root-Mean-Squared-Error (RMSE)	0.01842	0.1140	0.1455

Table 5. Goodness of fit statistics of correction function approximates in overall range $0 \leq x < 6$ for various algorithms.

$f_c(x)$ Approximations	SSE measures for range $0 \leq x < 6$
Constant log-MAP	631.78
Linear log-MAP	411.77
Hybrid log-MAP	61.01

The linear curve fits the exact correction function with a 95% confidence bound, which is sufficiently precise even from the naked eye (see Fig. 2), especially when it is compared to the other approximations.

The region of $0 \leq |x| < 1.5$, and not any wider, is chosen so that the linear curve fit is able to yield acceptably accurate goodness of fit parameters before he slope flattens out at larger values of $|x|$. The threshold of $x = 1.5$ was selected to ease the comparator's function with reduced fixed-point arithmetic complexity.

Besides being adequately accurate, this linear approximation is also simple to implement in hardware with the highly optimized DSP48E2 slice. The second region of the HLM algorithm lies in the $x \geq 1.5$ range. In this region, a "stair-case /multi-step" type approximation is employed where the ending value of $f_c(x = 1.5)$ of the linear region is calculated to be 0.1635 from Eq. (6) and used as the starting value, *c* of the second region, which features a multi-step like approximation. The multi-step approximation is limited to only three steps to simplify hardware implementation and each step level is related by a division of *c* = 0.1635 by 2 for *n* times.

In hardware, a division by 2 is a simple right bit shift. The multi-step algorithm is designed deliberately to exploit this property. Additionally, the approximation precision of the HLM algorithm for $1.5 \leq x < 6$ is improved by adjusting the number of integer shifts with $\lfloor 0.5x \rfloor$. Exploiting again the simplicity of a division by two and flooring, which incurs the least complexity in quantization methods on hardware. It is stressed now that the integer shift operation can be implemented with the Reinterpret Block, which does not incur any hardware resources in hardware implementation as elaborated in the next section.

As observed, graphically form Fig. 2, the resulting approximation yields a better agreement to the ideal Log-MAP. The next section presents the HLM correction function algorithm applied on to FPGA.

Implementation of algorithm in FPGA

The Xilinx DSP48E2 primitive is an embedded DSP block in the 7 series and more recently, the UltraSCALE+ series. Designed for high-speed DSP computation, it consists of a multiplier, arithmetic logic unit (ALU) and is capable of various operations such as multiply-add, multiply-accumulate, pattern detection, and logic units to name a few [11].

The HLM algorithm is designed in Xilinx's System Generator (XSE), a high-level tool for designing high-performance DSP systems. The design schematic is shown in Fig. 3 with the following key DSP blocks featured in the design [18].

- **Reinterpret block:** a basic element in System Generator that implements the division by 2. The block reinterprets the output fixed point such that a single right bit shift is implemented. This block costs virtually nothing in hardware [18].
- **Convert block:** a basic element that also costs nothing in hardware [18]. This block converts the input fixed point into a suitable binary output that drives the multiplexer in selecting the appropriate successive division by 2 for the region of $x \geq 1.5$ in Eq. (7).
- **Relational block:** implements a comparator in determining the regions of $|x|$ as described in Eq. (7) triggering the select of Mux2 in choosing the correct computation for either $|x| < 1.5$ or otherwise.
- **DSP48E2 block:** performs the multiply-add operation with the data flow of $p = a \times b + c$. The multiply-add operation requires only one clock cycle and is capable of executing at a maximum speed of 741 MHz [11].

Table 6 details the HLM resource consumption post place and route implementation excluding resources such as the reinterpret block that does not affect any cost. The proposed design leverages the reinterpret and convert block extensively in order to minimise resources on the FPGA. The distributive nature of these resources also minimises path delays and ensures good performance speed.

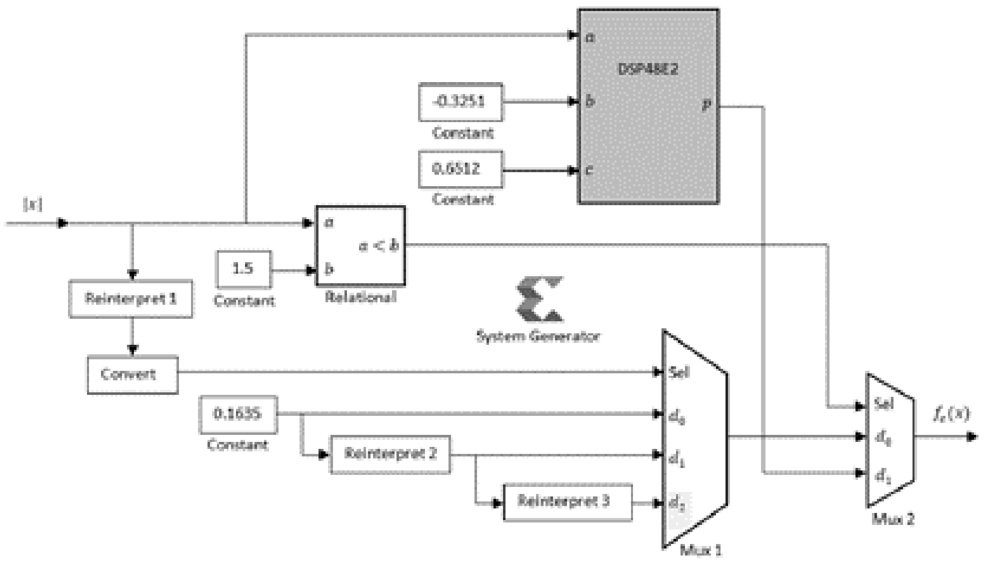


Fig. 3. HLM hardware implementation.

Table 6. Resource consumption of HLM approximation post place and route implementation along with other approximations.

Resources	HLM
DSP48E2	1
Relational	1
Constant block	4
Multiplexer	2
Adder	1

5. Results and Discussion

In this section, the performance of the HLM decoder under perfect SNR estimation and SNR mismatch are presented against the other reviewed approximations. The simulation parameters used are as shown in Table 7.

Table 7. Simulation parameters [2].

Parameters	Specifications
Decoder generator polynomial	G [23, 33]
Number of iterations	5
Rate	$1/2$
Interleaver	pseudorandom
Interleaver size	1024
Channel	AWGN

5.1. Performance comparison under perfect SNR estimation

The proposed HLM approximation has the closest performance to the exact Log-MAP algorithm when compared with other approximations especially at low SNR as shown in Fig. 4. Due to the good fit of the HLM algorithm for small values of $|x|$ the BER performance of the HLM algorithm is better than other approximations in low SNR values. Note that the sensitivity of the Log-MAP algorithm to SNR is more pronounced in encoders with more memory elements [7].

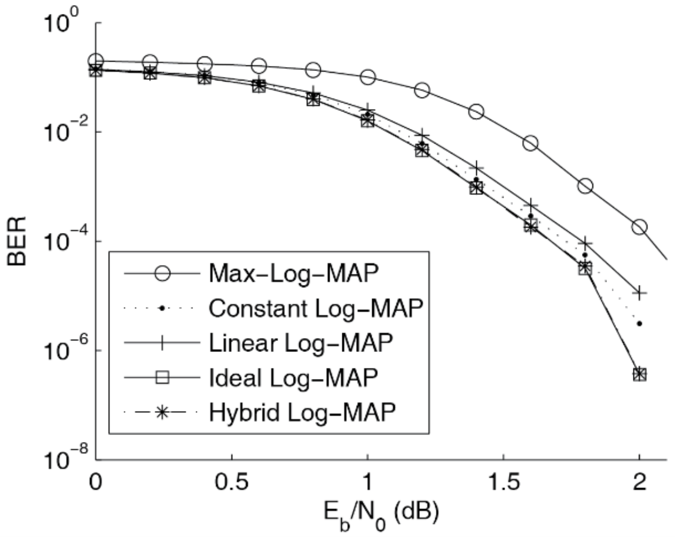


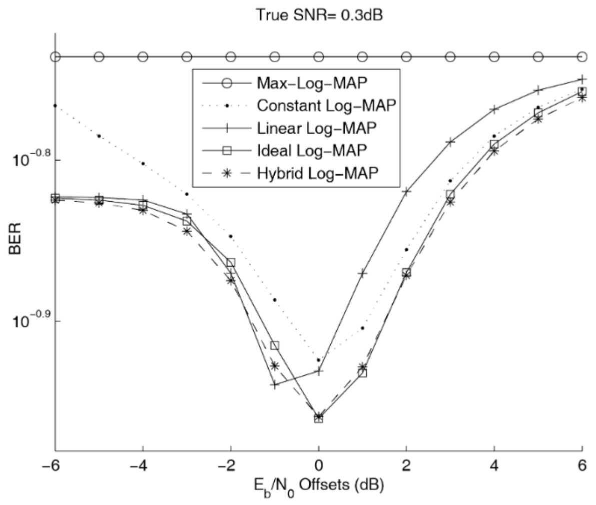
Fig. 4. BER performance under AWGN channel for CCSDS turbo decoder with different correction functions.

5.2. Performance comparison under SNR mismatch

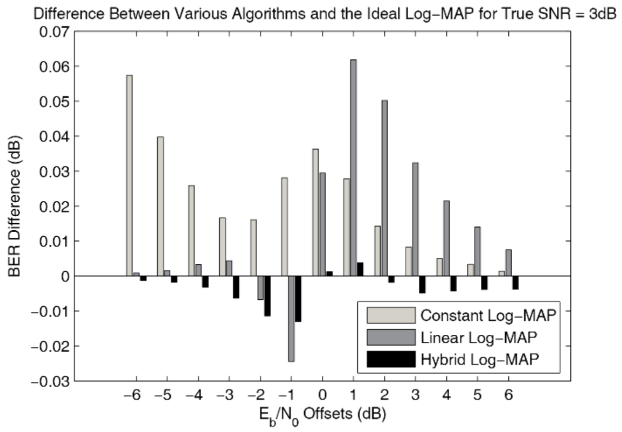
In this section, a performance comparison of the HLM against other algorithms when the SNR has not been correctly estimated is presented. Simulated results show that the HLM algorithm is close to and at times, better than the exact Log-MAP for negative SNR offsets in comparison with other algorithms under SNR mismatch.

This is useful in practical mobile communications where channel conditions may vary rapidly, causing severe SNR underestimation. The same simulation parameters in the previous section were used with the conditions of SNR offset. Since the Max-Log-MAP algorithm is insensitive towards SNR mismatch [7, 19, 20], it is therefore omitted in our analysis.

Figures 5(a) and (b) depict the BER performance of various Log-MAP approximations under SNR mismatch ranging from -6 dB (underestimation) to +6 dB (overestimation) of the true SNR values of 0.3 dB and 1 dB, respectively.



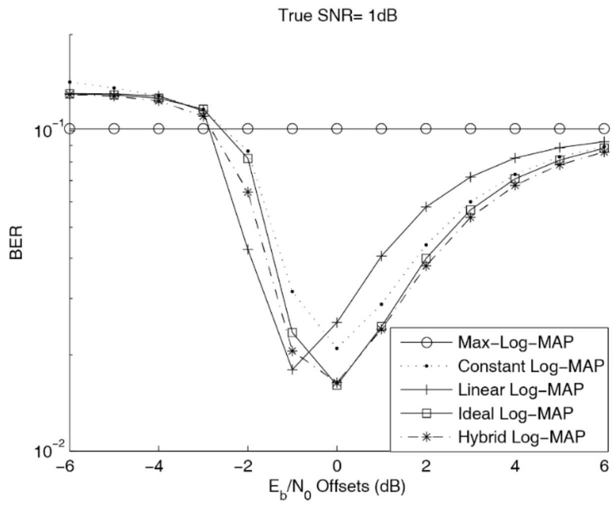
(a) BER versus *E_b/N_0* estimation offset for *E_b/N_0* = 0 .3 dB.



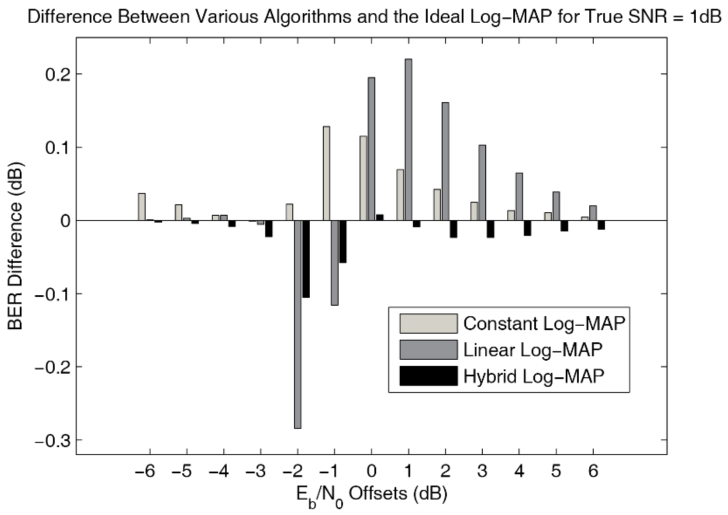
(b) BER difference of HLM and multistep with respect to exact Log-MAP for *E_b/N_0* = 0 .3 dB.

Fig. 5. SNR mismatch performance for 0.3 dB true SNR.

Figures 6(a) and (b) show the BER differences of various algorithms with respect to the exact Log-MAP performance. A negative BER difference indicates that the approximate algorithm has outperformed the exact Log-MAP solution. It is observed that the HLM outperforms other algorithms, especially under SNR underestimation. This characteristic is more evident at low (true) SNR levels since the HLM's approximation profile follows the exact Log-MAP closely at a smaller region of $|x|$. Comparatively, the Linear Log-MAP algorithm does outperform the HLM and even the ideal Log-MAP in SNR underestimation conditions. However, the Linear Log-MAP performance suffers in SNR overestimation, performing worse than the Constant Log-MAP algorithm.



(a) BER versus *E_b/N₀* estimation offset for *E_b/N₀* = 1 dB.



(b) BER difference of HLM and multistep with respect to exact Log-MAP for E_b/N_0 = 1 dB.

Fig. 6. SNR mismatch performance for 1 dB True SNR.

5.3. Comments on HLM's superior performance in SNR underestimation

It has been shown in many works of literature that approximate Log-MAP algorithms are more tolerant towards SNR overestimation than it is towards SNR underestimation [7, 19, 20]. This behaviour is mainly caused by the nonlinearity of the correction function and the dependency of $|x|$ on the channel reliability value, L_c. In the event of SNR overestimation or equivalently a positive SNR offset, the L_c estimate is larger than its true value. In over scaling L_c, the argument $|x|$ is usually large where the correction function is not effective. The correction function is thus de-emphasized, and the performance approaches to that of the Max-Log-MAP algorithm. For the case of underestimation, the opposite occurs, and the L_c estimate is lesser than its true value and thus the argument $|x|$ takes on smaller values. Because the correction function grows faster when the argument of $|x|$ is smaller, the correction term is overly emphasized. Thus, it is crucial for the approximate of the correction function to be acceptably accurate when $|x|$ is small. This is accomplished by the HLM algorithm, which fits closely to the exact Log-MAP curve.

6. Conclusions

The exact Log-MAP algorithm involves computationally intensive operations. Our proposed suboptimal hybrid Log-MAP (HLM) algorithm achieves nearly identical performance. Despite approximating the exact Log-MAP curve with pronounced accuracy, the HLM algorithm is designed to be hardware efficient from the start, and targeting SDR based platforms. Only simple bit-shifts (reinterpret) and other basic multiplication, addition and comparator operations are required. Simulations that were carried out verified the effectiveness of the HLM algorithm with respect to the exact solution, both with and without SNR mismatch. The performance of the HLM algorithm has shown greater tolerance to SNR mismatch be it overestimation or underestimation, which is ideal in practical situations where channel conditions are uncertain. The following itemises the contribution of this paper:

- The need for a strategically accurate approximation for the correction function and show its correlation to improved performance.
- A review of recent publications with a focus on linear and accurate approximations at small values of $|x|$. The lack of literature addressing the hardware implementation aspect on FPGAs were also highlighted.
- A new architecture for HLM algorithm leveraging rapid prototyping with minimal hardware resources was proposed as the primary design feature with details of resource block utilised.
- The performance result and analysis of the HLM algorithm against the competing algorithm in AWGN channel and SNR mismatch conditions were presented.

Nomenclatures

a, b, c	Coefficients to the HLM approximation
$f_c(\|x_1 - x_2\|)$	Correction function, $f_c(\|x_1 - x_2\|) = \ln\left(1 + e^{\|x_1 - x_2\|}\right)$
L_c	Channel reliability value

max*	Log exponential sum operator $\max^*(x_1, x_2) = \max(x_1, x_2) + \ln\left(1 + e^{\lvert x_1 - x_2 \rvert}\right)$
$\lvert x \rvert$	Generalised input to the max* operator $x = \lvert x_1 - x_2 \rvert$
$\lfloor x \rfloor$	Integer truncation of argument *x*
x_n	Individual input to the max* operator
Greek Symbols	
α	Forward metric in Turbo decoder
β	Backward metric in Turbo decoder
Abbreviations	
ASIC	Application Specific Integrated Circuit
BER	Bit Error Rate
CCSDS	Consultative Committee for Space Data System
DSN	Deep Space Network
DSP	Digital Signal Processors
FPGA	Field Programmable Gate Array
HLM	Hybrid Log-MAP
LUT	Lookup Table
MAP	Maximum A Posteriori
RMSE	Root Mean Squared
SDR	Software Defined Radio
SNR	Signal to Noise Ratio
SSE	Sum of Squared Error
XSG	Xilinx System Generator

References

1. Andrews, K.S.; Divsalar, D.; Dolinar, S.; Hamkins, J.; Jones, C.R.; and Pollara, F. (2007). The development of Turbo and LDPC codes for deep-space applications. *Proceedings of the IEEE*, 95(11), 2142-2156.
2. CCSDS 131.0-B-3. (2017). Recommendation for space data system standards: TM synchronization and channel coding. Retrieved March 24, 2019, from https://public.ccsds.org/Pubs/131x0b3e1.pdf.
3. Berrou, C.; Glavieux, A.; and Thitimajshima, P. (1993). Near Shannon limit error-correcting coding and decoding: Turbo-codes. *Proceedings of the IEEE International Conference on Communications*. Geneva, Switzerland, 1064-1070.
4. Nguyen, D.-H.; and Nguyen, H. (2015). An improved Log-MAP algorithm based on polynomial regression function for LTE Turbo decoding. *Proceedings of the IEEE International Conference on Communication Workshop (ICCW)*. London, United Kingdom, 2163-2167.

5. Robertson, P.; Villebrun, E.; and Hoeher, P. (1995). A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain. *Proceedings of the IEEE International Conference on Communications* (ICC) Seattle, Washington, United States of America, 1009-1013.
6. Zhou, Y.; and Shahrrava, B. (2009). Effect of SNR mismatch on the performance of turbo decoding algorithms. *Proceedings of the IEEE International Conference on Electro/Information Technology*. Windsor, Ontario, Canada, 15-18.
7. Khalighi, M.A. (2003). Effect of mismatched SNR on the performance of log-MAP turbo detector. *IEEE Transactions on Vehicular Technology*, 52(5), 1386-1397.
8. Gross, W.J.; and Gulak, P.G. (1998). Simplified MAP algorithm suitable for implementation of turbo decoders. *Electronic Letters*, 34(16), 1577-1578.
9. Talakoub, S.; and Shahrrava, B. (2004). A linear Log-MAP algorithm for turbo decoding over AWGN channels. *Proceedings of the IEEE Electro/Information Technology Conference*. Milwaukee, Wisconsin, United States of America, 293-296.
10. Lim, L.L.; and Lim, D.W.G. (2011). Hybrid Log-MAP algorithm for Turbo decoding over AWGN channel. *Proceedings of the Seventh International Conference on Wireless and Mobile Communications*. Luxembourg, 211-214.
11. Xilinx UG 579. (2018). UltraScale architecture DSP slice. Retrieved March 24, 2019, from https://www.xilinx.com/support/documentation/user_guides/ug579-ultrascale-dsp.pdf.
12. Wang, H.; Yang, H.; and Yang, D. (2006). Improved Log-MAP decoding algorithm for turbo-like codes. *IEEE Communications Letters,* 10(3), 186-188.
13. Ivanov, Y.Y.; Romanyuk, A.N.; Kulyk, A.I.; and Stukach, O.V. (2015). A novel suboptimal piecewise-linear-log-MAP algorithm for turbo decoding. *Proceedings of the International Siberian Conference on Control and Communications (SIBCON).* Omsk, Russia, 1-8.
14. Foda, M.A.; El Ghany, M.A.A.; and Hoffman, K. (2017). Efficient polynomial regression algorithm for LTE turbo decoding. *Proceedings of the IEEE International Conference on Electronics, Circuits and Systems (ICECS).* Batumi, Georgia, 264-269.
15. Zhilyaev, E.; and Gurova, E.B. (2018). On the question of the authentication tag length based on Reed-Solomon codes. *Proceedings of the Moscow Workshop on Electronic and Networking Technologies (MWENT).* Moscow, Russia, 1-5.
16. Fedorenko, V.V.; Aldushchenko, D.V.; Listova, N.V.; Samoylenko, I.V.; and Samoylenko, V.V. (2018). The signal code structure selection in the communication channels in the wireless sensor networks. *Proceedings of the Workshop on Electronic and Networking Technologies (MWENT).* Moscow, Russia, 1-4.
17. Liu Z.; Wu B.; and Ye, T.C. (2018). Improved Turbo decoding with multivariable Taylor series expansion. *IEEE Communications Letters*, 22(1), 37-40.
18. Xilinx UG 912. (2018). Vivado Design suite properties reference guide. Retrieved March 24, 2019, from https://www.xilinx.com/support/documentation/sw_manuals/xilinx2018_2/ug912-vivado-properties.pdf.

19. Cheng, J.-F; and Ottosson, T. (2000). Linearly approximated log-MAP algorithms for turbo decoding. *Proceedings of the IEEE 51st Vehicular Technology Conference*. Tokyo, Japan, 5 pages.
20. Worm, A.; Hoeher, P.; and Wehn, N. (2000). Turbo-decoding without SNR estimation. *IEEE Communications Letters*, 4(6), 193-195.