IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 9, Issue 4, Ver. III (Jul - Aug. 2014), PP 19-23
www.iosrjournals.org

Low Complexity Decoder for CCSDS Turbo codes

K.V Karthik[1], Naveen I.G[2]
[1](Dept of ECE,SirMVIT,Bengaluru,India)
[2](Asst.Prof,Dept of ECE, SirMVIT,Bengaluru,India)

Abstract: *Error correction codes are a means of including redundancy in a stream of information bits to allow the detection and correction of symbol errors during transmission. In Forward Error Correction (FEC) it is desirable to have low Bit Error Rates (BER) and low decoder complexity for reliable data transmission Turbo code is a great achievement in the field of communication system. Turbo code also consists of an interleaver unit and its BER performance also depends on interleaver size. This paper presents a low complexity turbo decoder for CCSDS (Consultative Committee for Space Data Systems) telemetry channel coding standard. Simulations are done in MATLAB to assess the BER performance of the design in an AWGN channel. Log–MAP decoding algorithm is less computationally complex with respect to MAP (maximum a posteriori) algorithm, without compromising its BER performance. A register transfer level (RTL) turbo encoder is designed and simulated using Hardware Description Language.*

Index Terms: *Turbo decoder, CCSDS, Log-MAP*

I. Introduction

The CCSDS recommended turbo encoder [1] is a robust candidate for systems with near Shannon-limit [2] performance. Due to its powerful error correcting capability, reasonable complexity, and flexibility in terms of different block lengths and code rates, number of memory element etc., turbo codes are widely used in various wireless systems such as 3GPP, CDMA 2000, IEEE 802.16, DVB-RCS, UMTS and CCSDS.

As a complement to turbo encoding, there are several decoding techniques. Traditional MAP algorithm was too complex to implement in hardware, because the exponential calculus was too complex and huge memory was required for storing the metrics: forward, backward and transition. But Log MAP and its sub-optimal version, max log map can reduce its complexity to a great extent [4] [5]. In turbo code, interleaver unit is a random block that is used to rearrange the input data bits with no repetition. Interleaver unit is used in both encoder and decoder part. At the encoder side it generates a long block of data, whereas in decoder part it correlates the two SISO decoder and helps to correct the error. At the decoder side after passing the encoded data from first decoder some of the errors may get corrected, then we again interleave this first decoded data and pass through the second decoder. Here, remaining error may get corrected. The process is repeated for more number of times.Several implementations recently proposed regarding turbo decoder [4] [5] [6] are based on fixed point arithmetic. This paper extends the log map SISO algorithm for the turbo codes used in CCSDS compliant systems as an example.Encoder specifications are based on CCSDS recommendations [1] and decoding algorithm are explained in section [III] and section [IV] respectively.

II. CCSDS Turbo Encoder Structure

The turbo encoder in the CCSDS Recommended Standard [1] is shown in Fig. 1. The two convolutional encoders in the figure are identical recursive with constraint length K = 5, and are realized by feedback shift registers. The input message is a frame of k information bits. The parity symbols generated by the two component encoders are selected by a puncturing block for a particular rate and are sent along with the uncoded information bits. An interleaver permutes bit wise

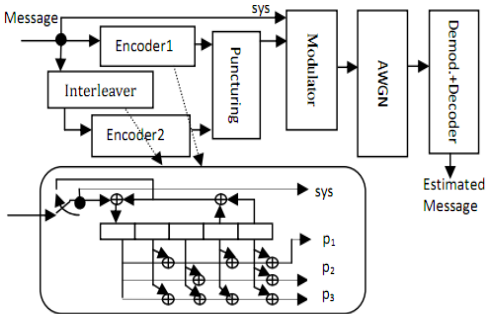

Fig1.Turbo system block diagram

The original k information bits before input to the second encoder. The turbo codeblock is terminated by running each encoder for an additional K-1 bit times beyond the end of the information bit frame.
Turbo code considered here has the following specifications [1]:

-Code rate =1/3.
-Constituent code generating function:
g= (23, 33) oct.
-Block size: 1784.
-Interleaver: CCSDS recommended.
Trellis termination:done.
-Considered modulation: BPSK over AWGN

III. Turbo Decoder

The block diagram of the turbo decoder is shown in fig.2.There are two SISO decoder corresponding to the two encoders. The inputs to the first decoder are the observed systematic bits, the parity bit stream from the first encoder and the deinterleaved systematic bit stream, the observed parity bit stream from the second RSC and the interleaved extrinsic information from the first decoder.

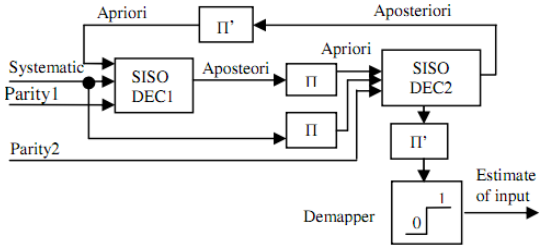

Fig2 Turbo Decoder

An iterative decoding procedure,in each component decoder is an algorithm that computes the a posteriori probability(APP) of the information symbols which is the reliability value for each information symbol.The sequence of reliability values generated by a decoder is passed to the other one.To improve the correctness of its decisions ,each decoder has to be fed with information that does not originate from itself.The concept of extrinsic information was introduced to identify the component of the general reliability value,which depends on reduntant information introduced to identify the component of the general reliability value,which depends on redundant information was introduced by the considered constituent code.A natural reliability value,in the binary case is the logarithm likelihood ratio(LLR).Each decoder has a number of compute intensive tasks to be done during decoding .there are five main computations to be performed during each iteration in the decoding stage as shown in Fig.2 The computations are as follows (computations in one decoder per iteration)

A. Branch Metric Unit (BMU)

In the algorithm for turbo decoding the first computational block is the branch metric computation. The branch metric is computed based on the knowledge of input and output associated with the branch during the transition from one state to another state.There are sixteen states and each state has two branches,which gives a total of thirty two branch metrics.The computation of branch metric is done using below equation:

$$\gamma[k] = x_s[k].z[k] + [k].Lc.y_s[k] + x_p[k].y_p[k] \quad (1)$$

only two values are sufficient to derive branch metrics for all the state transitions Where $\gamma[k]$ is the branch metric at time k,$x_s[k]$ are the systematic bits information with frame length N,is the information that is fed back from one decoder to other decoder,z[k] is the channel estimate which corresponds to the maximum signal to distortion ratio, $x_p[k]$ is the encoded parity bits of the encoder, $y_p[k]$ is the noisy observed values of the encoded systematic bits.The γ unit takes the noisy systematic bit stream,the parity bits from encoder 1 and encoder 2 to decoder 1 and decoder 2 respectively and the a priori information to compute the branch metrics.The branch metrics $\gamma[k]$ for all branches in the trellis are computed and stored.

B. State Metric Unit(SMU)

The forward metric α is the next computation in the algorithm,which represents the probability of a state at time k,given the probabilities of states at previous time instance α is calculated using equation(2)

$$\alpha_s[k] = \sum_{s' \in s} \alpha_{s'}[k-1].\gamma_{s',s}[k] \quad (2)$$

Where the summation is over all the state transitions s' to s.α is computed at each node at a time instance k in the forward direction traversing through the trellis.α is computed for states 16 states.

C. Backward State Metric –β unit

The backward state probability being in each state of the trellis at each time k,given the knowledge of all the future received symbols,β is recursively calculated and stored.The backward metric β is computed using equation(3)in the backward direction going from the end to the beginning of the trellis at time instance k-1,given the probabilities at time instance k,

$$\beta_{s'}[k-1] = \sum_{s' \in s} \beta_{s'}[k].\gamma_{s',s}[k-1] \qquad (3)$$

Where the state transition is from s'.β is computed for 16 states .Backward metric computation can start only after the completion of the computation by the γ unit. Observing the trellis diagram ,there are four β values to be computed,but now in backward order,from (N-1 down to 0).A backward recursion on the trellis is performed by computing β[k] for each node in the trellis.Thecomputation is the same as for α,but starting at the end of the trellis and going in the reverse direction.

D. LLR Unit (LU)

The LLR unit will compute the log likelihood ratios and is in fact an extended version of the state metric unit in hardware where three inputs(a,P,y) are applied to each input adder and the maximum of sixteen inputs are calculated instead of two as in BMU. Log Likelihood Ratio (LLR) is the output of the turbo decoder.This output LLR for each symbol at time k is calculated as

$$LLR[k-1]=\ln\frac{\sum u_k=1\ \alpha'[k-1]\beta_{s'}[k]\gamma_{s',s}[k]}{\sum u_k=0\ \alpha'[k-1]\beta_{s'}[k]\gamma_{s',s}[k]} \qquad (4)$$

Where numerator is summation over all the states s' to s in γ[k] and input message bit u[k]=1.The denominator is summation over all the states s' to s in γ[k] and input message bit u[k]=0.The γ values unit output and the β values obtained from the above steps are used to compute the LLR values. The main operators are comparison, addition and subtraction. Finally,these values are de-interleaved at the second decoder output, after the required number of iterations to make the hard decision,inorder to retrieve the information that is transmitted. The log likelihood ratio LLR[k] for each k is computed.The LLR is a convenient measure since it encapsulates both soft and hard bit information in one number.The sign of the number corresponds to the hard decision while the magnitude gives a reliability estimate.

E . Extrinsic unit

Extrinsic information computation uses the LLR outputs,the systematic bits and the a priori information to compute the extrinsic value.This is the value that is fed back to the other decoder as the priori information. This sequence of computations is repeated for each iteration by each of two decoders.After all iterations are complete,the decoded information bits is a zero.This is because the LLR is defined to be the logarithm of the ratio of the probability that the bit is a one to the probability that the bit is a zero.

IV. Decoding Algorithm

This section will not try to discuss the derivation of the Log-MAP algorithm, which has been described in [7] [8] [10]. This section is intended to present a simplified practical computation method to be implemented efficiently in the turbo decoder. A simplified block diagram of the turbo decoder is depicted in Fig. 2. [8] [9].

Decoding process for turbo decoder starts with the estimation of a-posteriori probability(APP) for each transmitted bit that corresponds to the MAP probability of that bit [9] [11]. Whenever a demodulated sequence is received, which was corrupted by AWGN channel; the APP decision process allows the MAP algorithm to determine the most likely transmitted bit on each time. The turbo decoding process is performed in an iterative manner, and its basic idea is to use two

MAP decoders to decode each turbo code component and obtain soft estimations of the received bits. Obtained soft estimations are interleaved and used by the second decoder to improve the estimation of the information bit. New estimations are sent to the first decoder to complete an iterative sequence after being deinterleaved.

LOG MAP algorithm consists of four main operations: branch metric computation, forward recursion, backward recursion and log likelihood ratio (LLR) calculation. The algorithm takes a block of N received symbols that corresponds to N trellis stages. When the received signal values y_k are assumed to be got after BPSK (binary phase shift keying) 1-2c mapping and corrupted by an AWGN channel, the branch probability (gamma) of each transition from state s' to state s has been calculated first. After that, forward recursion (alpha), backward recursion (beta) and LLR calculations have been done [8] [9].

The decoding process in MAP algorithm performs calculations of the forward and backward state metric values to obtain the log likelihood ratio (LLR) values which have the decoded bit information and reliability values. The LLR values are represented by the following equation.

$$LLR=\ln\frac{\sum_{S_k}\sum_{S_{k-1},}\gamma_1(S_{k-1},S_k)\alpha(S_{k-1})\beta(S_k)}{\sum_{S_k}\sum_{S_{k-1},}\gamma_0(S_{k-1},S_k)\alpha(S_{k-1})\beta(S_k)}=L_1\text{-}L_2 \qquad (5)$$

$$L_1=\ln\sum_{S_k}\sum_{S_{k-1,}}\gamma_1(S_{k-1},S_k)\alpha(S_{k-1})\beta(S_k) \quad (6)$$

$$L_2=\sum_{S_k}\sum_{S_{k-1.}}\gamma_0(S_{k-1},S_k)\alpha(S_{k-1})\beta(S_k) \quad (7)$$

Where γ,α and β represent the branch, forward and backward state metric values,respectively.The subscript k and S denote time and state. The LLR value (LLR)is calculated by the metric values at all states(S) of time k andl-1.The equation of γ,α and β can be represented to logarithm form below.

$$\ln\gamma(k)=\frac{1}{2}(L_e u_k^p+L_c x u_k^s+L_c y_1 u_k^p) \quad (8)$$

$$\ln\alpha_s[K]=\ln\sum_{S_{k-1}}\exp(\ln\gamma+\ln\alpha(S_{k-1})) \quad (9)$$

$$\ln\beta(S_k)=\ln\sum_{S_{k+1}}\exp\left(\ln\gamma(S_k,S_{k+1})+\ln\beta(S_{k+1})\right) \quad (10)$$

Where the branch metric (γ)is calculated by the a priori information (L_e),channel reliability value (L_c),input symbols(x and y1),the systematic bit (u_k,b) and the parity bit (u_k,s) As described in previous section,a priori information is obtained from the LLR value computed in previous decoding process after subtracting the input symbol data and a priori values from the LLR value. In order to make the logarithm function sum functionable, the well known approximation called Jacobi logarithm function is used. Which is given as: $\ln(e^x+e^y)=\text{Max}(x,y)+\ln(1+e^{-|x-y|})$. A lookup table can implement the corrective term $\ln(1+e^{-|x-y|})$.

V. Simulation Results

Performance of the log map decoder can be evaluated by calculating its bit error rate(BER) corresponding to each Eb/No value. The decoder using Log Map algorithm was simulated using MATLAB platform,from the curve for 1784 input bits the decoder is achieving close to capacity performance.The obtained result is shown below.A RTL Turbo encoder simulated on Verilog is also included .

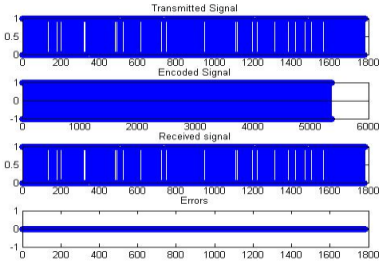

Fig3 Performance of the turbo decoder at N=1784

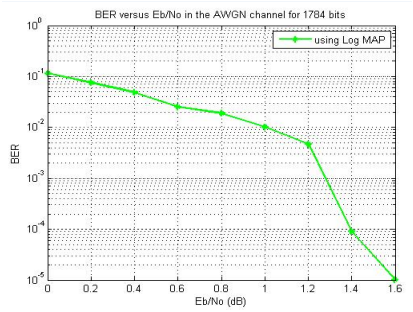

Fig4 BER performance of the CCSDS compliant turbo decoder in AWGN Channel

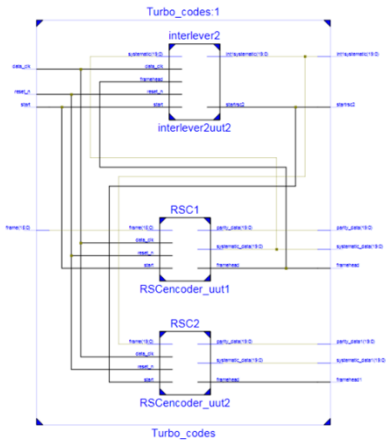

Fig5 RTL Turbo encoder based on the CCSDS Standard

VI. Conclusion

The iterative nature of decoding process improves the efficiency of transmission. The data transmission can be made error free by correct design of the decoder. The Turbo decoder can be implemented using LOG Map Algorithm as it is less computationally complex with respect to MAP Algorithm although the Log MAP algorithm is complex it can still achieve a good BER performance. The BER performance can be improved easily by increasing the number of iterations.

VII Acknowledgements

The authors are thankful to IOSR Journal for their support to develop this document.

References

[1] NASA, 'Telemetry channel coding. CCSDS 101.0-B-6 recommendation for apace," U.SA., October 2002.
[2] C. Berrou, A. Glavieux, P. Thitimajshima, 'Near Shannon limit error- correcting coding and decoding: Turbo codes " Proceedings of the 1993 International Conference on Communications, pp. 1064-1070, 1993.
[3] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," IEEE journal on Selected Areas in Communications, VOL. 16, NO. 2, Feb. 1998, pp.260-264.
[4] Indrajit Atluri, Tughrul Arslan, "Area/ Power Efficient Implementation of a Log-MAP Decoder for Turbo Codes through Memory Optimizations." Xiao-Jun Zeng and Zhi-Liang Hong, "Design and Implementation of a Turbo Decoder for 3G W-CDMA Systems". IEEE Transactions on Consumer Electronics, Vol. 48, No. 2, May 2002.
[5] J.M.Mathana, Dr.P.Rangarajan, "FPGA implementation of high speed architecture for Max Log MAP Turbo SISO decoder, International Journal of Recent Trends in Engineering, Vol 2, No. 6, November 2009.
[6] Shivani Verma and Kumar S, An FPGA realization of simplified turbodecoder architecture, International journal of the physical sciences, May 2011, pp.2338-2347.
[7] Abrantes S. A, From BCJR to turbo decoding:MAP algorithms made easier, April 2004.
[8] (2007) The website. Available: http://www. complextoreal.com/
[9] Prof M. Srinivasa Rao, Dr P. Rajesh Kumar and K. Anitha, "Modified Maximum Aposteriori algorithm for iterative decoding of Turbo codes," International Journal of Engineering Science and Technology, Vol. 3 No. 8 August 2011.
[10] Shu Lin/ Daniel J. Costello, "Error Control Coding: Fundamentals and Applications," Prentice Hall, Inc., Englewood Cliffs, 1983.
[11] Roberto Ramirez Marin, Andres David Garcia Garcia and Luis Fernando Gonzalez Perez, "Hardware architecture of MAP algorithmfor Turbo codes implemented in a FPGA " Proceedings of the 15th International Conference on Electronics, Communications and Computers, pp. 0-76952283, 2005.