From Algorithm testing to ASIC input code of SOVA algorithm for TURBO-Codes

K. Koora, F. Poegel, S. Zeisberg and A. Finger
Dresden University of Technology, Communications Laboratory, 01062 Dresden, Germany
e-mail: koora@ifn.et.tu-dresden.de

Abstract

Recently a new class of codes, TURBO-codes have been introduced for channel coding in communication systems. Due to the iterative decoding scheme of these codes, it is possible to achieve results near to the Shannon limit. In order to decode these codes, implementation of a soft input and soft output decoder is essential. The realization of such type of decoders can be done either by using a maximum a posteriori (MAP) symbol estimator or by using a soft output Viterbi algorithm (SOVA). In this paper, an approach from algorithm to ASIC input of a SOVA decoder for TURBO-codes is described.

1. Introduction

Nowadays most of the communication receivers use Viterbi algorithm either to decode or to equalize the received signals. A few systems use one Viterbi decoder concatenated with a block decoder. Some coding systems use convolutional codes with Viterbi decoding to perform Forward Error Correction (FEC) decoding. The new class of codes, TURBO-Codes [1, 4, 5, 7, 8] also use at least two decoders in serial per iteration depending on the coder design. For detail description of TURBO-Codes please refer to the specified literatures. In case of concatenation, there will be a loss in the performance of error correction, if the first decoder takes the received soft values and gives hard values to the second decoder, though the second one is capable of using quantized soft values. This drawback can be overcome if the first decoder can produce relevant soft output values. Computing these values can be achieved by using maximum a posteriori (MAP) symbol estimators [1, 9], which work with the probability values of the received symbols. But due to the enormous complexity of this algorithm, it is less interesting for the realization. The other algorithm to solve this problem, Soft Output Viterbi Algorithm (SOVA) [2, 3, 4], is also capable of giving not only hard decisions but also an estimate of the probability with which the bit has been correctly detected. This is done with the help of calculated metric values in the Viterbi algorithm. This paper gives a concise but illustrative overview of the implementation of SOVA algorithm for TURBO codes.

2. The Soft Output Viterbi Algorithm

Here a brief description of SOVA algorithm is given. For thorough description of this algorithm please refer to [3].
In its basic form, the Viterbi algorithm accepts either hard or soft values of the received signals to compute the branch metrics for every state, which are added recursively to the old branch metric values giving rise to the path metric values. These operations are done for every state and the path with largest metric value is selected as the survivor among the other paths

merging in that state. This is done in the Add-Compare-Select (ACS) unit of a Viterbi decoder. The outputs of the ACS unit are then managed in the survivor memory unit. Finally the state with the largest accumulated metric is selected as the survivor at the given time point, and the output of the decoder is taken from the survivor memory of that particular state. Better results can be achieved if the survivor memory unit is very large. But for practical realization it is limited. This is also called truncation path length and depends on the memory which is used by the convolutional encoder. be The least value is thrice the memory length [11]. For the soft output algorithm, not only the hard values but also the metric difference (‘Δ’) between the survivor and concurrent paths are delivered by the ACS unit. Along with the hard values these values are also stored in the soft value memory unit. For a time point *‘t’* all the metric differences, which have been calculated till time *‘t-1’* are updated by comparing the survivor and concurrent paths of each state in case if the hard output values are different. This update is done by selecting the minimum among the old value and the newly computed ‘Δ’. In order to keep the complexity of the computations and comparisons of ‘Δ’ values acceptable, only codes with code rate = $1/x$ are considered for the practical realization, where *‘x’* represents the number of encoder outputs.

3. Simulations with SOVA-Algorithm

To test the SOVA algorithm, simulations were carried out with TURBO codes for multi-path and AWGN channels. The generator polynomial and the puncture matrix used here are as in [1]. In figure 1a, a comparison between TURBO codes and a classical coding schemes is shown. These simulations were done using measured channel impulse response with two omini-directional antennas in 60GHz environment and DQPSK modulation. A slightly modified 22x22 block interleaver was used in the TURBO codec. The applied RS code is a shortened (255,199) code with 58 information symbols and a code rate of 0.5. In case of concatenation coding scheme, a shortened (255,239) RS code with rate 0.78 and convolutional code with constraint length 7 and code rate 2/3 is used. Because of overall code rate 0.52, the achieved results can be compared with TURBO codes of code rate 0.5. It is clearly seen that TURBO codes achieve better results than the other coding schemes. Figure 1b shows the results which were achieved with the help of TURBO decoders for three iterations using AWGN channel with block inerleaver of size 128x128 and BPSK modulation. Compared to classical convolutional RSC g{37,21} code, TURBO decoder with three iterations has a gain of 1.6 dB at 10^{-3} bit error rate.

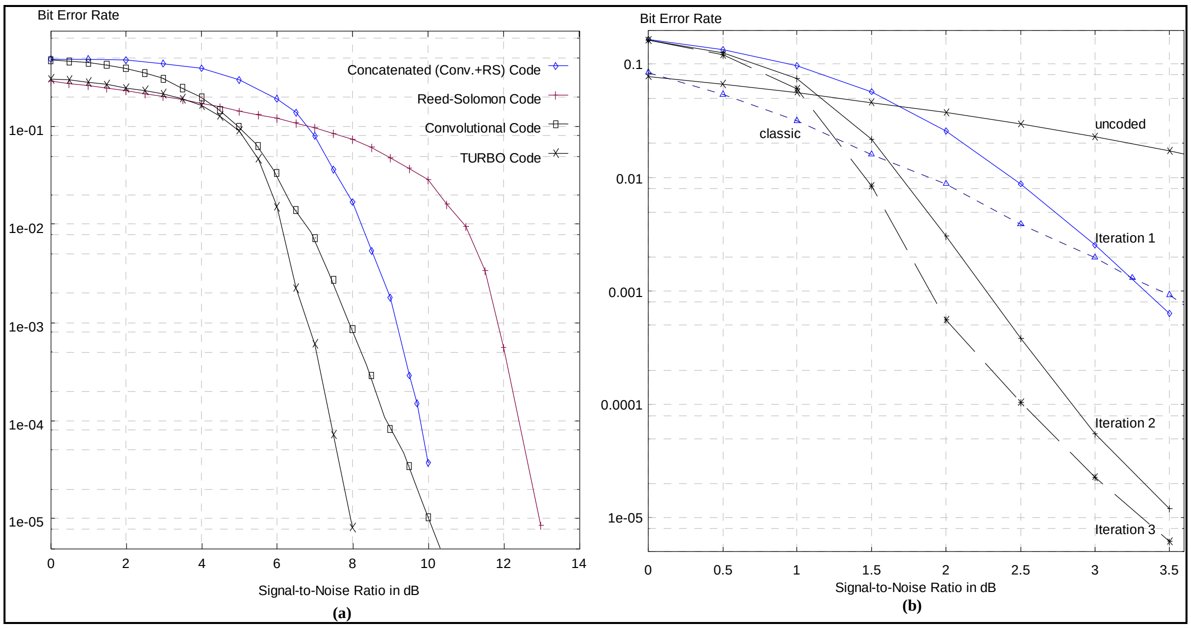

Figure 1: *Application of TURBO-Codes for AWGN and multi path channels.*

A flattening effect is also seen in the curves as the iteration number increases. This is because of the free distance of the codes. The free distance can be increased by using selected random interleavers, which improves the BER [5, 6].

In order to find out the input, output and internal results word length without any great loss on the performance, the algorithm was implemented as a bittrue model for a simulation tool. Figure 2 shows the results of the simulations, where the algorithm has been tested for a convolutional encoder with the generator polynomial g{37,21} and using the soft input decoder. With the help of those simulations, the word lengths for the signal processing have been chosen.

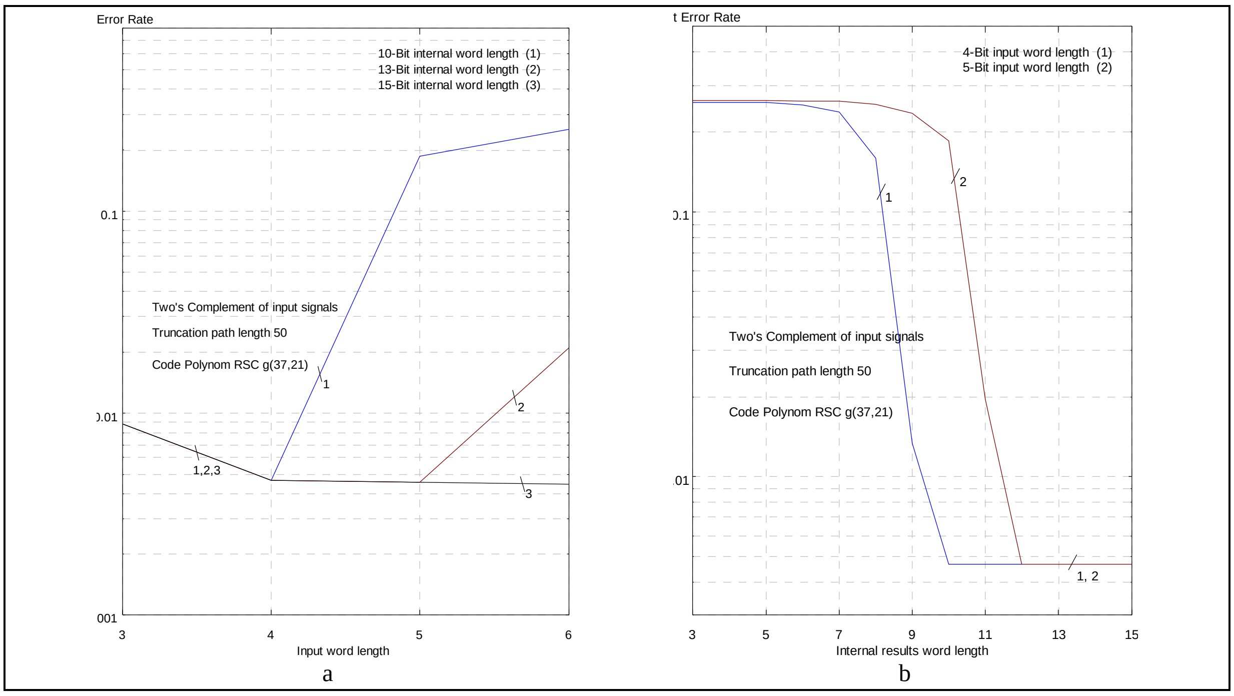

Figure 2: *Effect of (a) input and (b) internal results word length on the performance at SNR = 2.5 dB.*

By a fixed internal word length, though the input word length is increased there is a degradation in the performance of the algorithm. This is because, by large input word lengths the internal results reaches the maximum fixed limit very quickly and many of the accumulated values are manipulated, for example by clipping. The algorithm, which is used here, selects a path as the survivor, if its accumulated value is greater than that of the other path values. In case of clipping the results, many of the paths have the same maximum value. This increases the possibility for the selection of a wrong path as the survivor. Figure 3 shows the histograms of the soft values, also called extrinsic informations, which are generated and quantized (4- and 5- bit quantization) by SOVA decoder, when only 1's are taken as input of the encoder. Due to less number of signal levels at 4-bit quantization compared to 5-bit, there are many ‘sure’ events. This may degrade the performance of the next decoding stage, as all the events are not sure events at low SNR values.

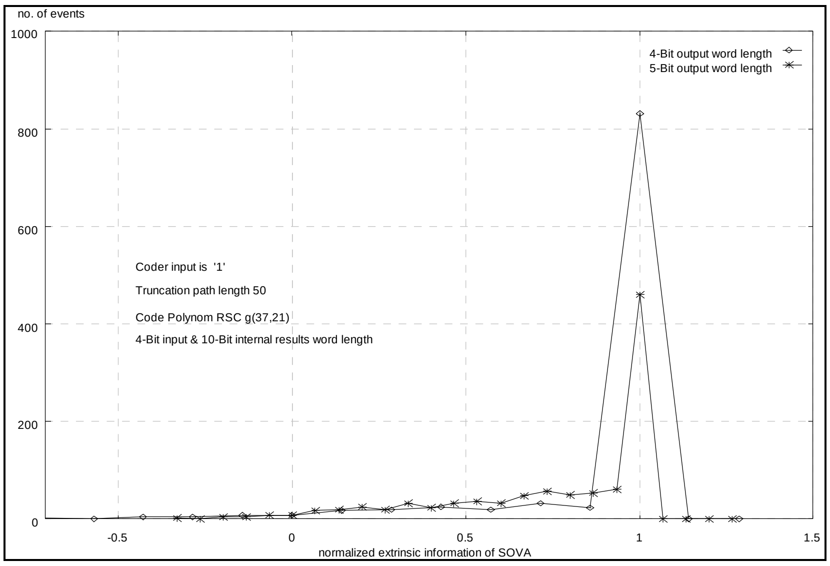

Figure 3: *Histogram of extrinsic information for given output word length at SNR = 2.5 dB.*

These simulations helps the designer to choose the word lengths for the practical realization. Through the achieved results one can select 4-bit word length for input and output ports and 10-bit word length for internal results without a great loss in the precision of the algorithm.

4. Realization of the SOVA-Algorithm

In order to realize this code using ASIC tools an input for a synthesis program was prepared in VHDL language. During this phase the algorithm was divided into many small processes and a synchronization between them was fixed. The figure 4 shows the division of the algorithm.

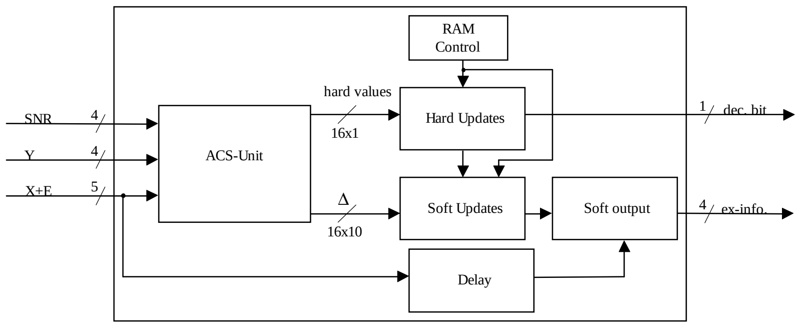

Figure 4: *Process division of SOVA algorithm.*

Along with the received systematic and redundancy symbols, an estimate of the channel state and an extrinsic information is fed into the ACS unit, with which the branch metric of the states are computed and are added to the path metrics. In order to prevent an overflow of the accumulated values, the maximum of all is substituted from the metric values. At the same time the metric differences 'Δ' are also calculated. The hard and soft decision values are then fed into the respective update units. The path management was realized using register exchange method. Though this method has a large demand on the memory, it has been selected as it allows the update of the soft values parallel to hard values. Unlike the architectures presented in [2, 10], there is no further latency than the desired truncation path length. If the truncation path length of the decoder is large enough, there is no need for the updates of the soft values at the end of the path. This is because of the merging property of the register exchange method, due to which all the paths have the same outputs. In order to keep the computed soft values uncorelated with the received values, the input values are delayed and are subtracted in the soft output unit. The registers are supervised with the help of the logic in the RAM control unit. After successful tests with the VHDL code, Register Transfer Level (RTL) codes of the processes were created using the Synopsys-Design Analyzer tool (version 3.4a). The created codes have been further optimized to achieve same results using less hardware components and to generate a gate level net list.
A rough estimation of the gate number has also been done to calculate the chip size without taking the wire area into consideration. Table 2 shows the gate count for four major units.

Unit	**no. of gates**
calculation unit (ACS + RAM control)	9347
hard values management	5600
soft values management	110000
signal delay	4000
Total no. of gates	128947

Table 1: *Gate count for the SOVA block.*

Using 0.35 µm technology the number of gates which can be brought on 1 mm² of chip area is roughly 8000. If this technology is used for the implementation, then the total chip area is about 16.12 mm². In case of 1µm technology the required chip area is round 47 mm².
Since the simulations are done taking worst case into consideration, the number of gates for soft values management is large. In general, if the noise in channel is acceptable, the internal results word length can be decreased. Every bit, which has been reduced, helps to save memory elements which are equal to the number of states times the length of soft value management. By large SNR values, due to the merging property used by the register exchange method, all the states have the same outputs. In this case, the soft value management length can be kept smaller than the truncation path length. This helps to reduce the gate count and to keep the chip area small.

5. Conclusions

In this paper, results which have been achieved with TURBO codes using the soft output Viterbi algorithm for AWGN and for multipath channels are presented. Compared to classical coding schemes these codes achieve same bit error rates at lower SNR vlaues. The design of the algorithm has been further optimised for a TURBO code with a 16 state recursive systematic convolutional code to allow the parallel updates of soft and hard values. Using bittrue models of the TURBO decoder, simulations were carried out to select optimum input, output and internal results word lengths. Using VHDL code of SOVA algorithm as an input of synthesis tools, an estimate has been done for the required chip area. This process can also be applied for the design of a complete system.

Acknowledgements

We like to thank the staff members of the DSP and the EDA groups at VTT Electronics Oulu, Finland for their guidance, support and for providing an access to modern tools. Propagation measurements of the “Forschungszentrum der Deutschen Telekom AG” were greatfully acknowledged.

References

[1] Claude Berrou, Alain Glavieus, Punya Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding (TURBO-Codes)”, International Conference on Communication (ICC), Geneva Switzerland, pp. 1064 - 1070, May 1993

[2] Claude Berrou, Patrick Adde, Ettiboua Angui and Stéphane Faudeil, “A Low Complexity Soft-Output Viterbi Decoder Architecture”, International Conference on Communication (ICC), Vol. 2/3, pp. 737 - 740, May 1993

[3] Joachim Hagenauer, Peter Hoeher, “A Viterbi Algorithm with Soft Decision Outputs and its applications”, Proceedings of IEEE GLOBECOM, pp. 1680 - 1686, November 1989

[4] Joachim Hagenauer, Patrick Robertson and Lutz Papke, “Iterative (TURBO) Decoding of Systematic Conovolutional Codes with the MAP and SOVA Algorithms”, Informationstechnische Gesellschaft (ITG) Fachbericht, pp. 21 - 29, October 1994

[5] Patrick Robertson, “Illuminating the Structure of Code and Decoder of Parallel Concatenated Recursive Systematic (Turbo) Codes”, Proceedings IEEE GLOBECOM'94, San Francisco, pp. 1298 - 1303, 1994

[6] Peter Jung and Markus Nasshan, “Performance evaluation of turbo codes for short frame transmission systems”, Electronics Letters, Vol. 30, No. 2, pp. 111 - 113, January 1994

[7] Peter Jung, “Novel low complexity decoder for Turbo-codes”, Electronic Letters, 19th, Vol. 31, No. 2, pp. 86 - 87, January 1995

[8] Peter Jung, Markus M. Nasshan, “Comprehensive Comparison of Turbo-code decoders”, IEEE Vehicular Technology Conference (VTC), Chicago, July 1995

[9] L. R. Bahl, J. Cocke, F. Jeinek and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate”, IEEE Transactions on Informations Theory, Vol. IT-20, pp. 248 - 287, March 1974

[10] O. J. Joeressen, M. Vaupel and H. Meyr, “VLSI Architectures for Soft-Output Viterbi Decoding”, Proceedings of the International Conference on Application Specific Array Processors, IEEE-Computer Society Press, pp. 373 - 384, August 1992

[11] Ivan M. Onyszchuk, “Truncation Length for Viterbi Decoding”, IEEE Transactions on Communications, Vol. 39, No. 7, pp. 1023 - 1026, July 1991