IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 8, Issue 8, August 2020

Impact Factor: 7.488

Design and Analysis Combining Two Algorithms in One Turbo Decoder

Amer T. Ali, Dr. Dhafir A. Alneema

Master Student, Dept. of Computer Engineering, College of Engineering, University of Mosul, Nineveh, Iraq

Lecturer, Dept. of Computer Engineering, College of Engineering, University of Mosul, Nineveh, Iraq

ABSTRACT: Turbo code is one of important technology in digital communication systems. And it is widely used for many applications in these dayssystems. The (Max-Log-MAP)and (Log-MAP) algorithmsare used for turbo decoder. But a novel work was proposed in this work by combining these two algorithms in one turbo decoder. The results show that when used small interleaver size the performance in bit error rate (BER) was better than the average of the two algorithms in (BER). And the maximum latency for the proposed design was equal to the average of maximum latency of the two algorithms. And the resource utilization by the proposed design was equal to the average of resources utilization of the two algorithms.

KEYWORDS: Turbo code; (Max-Log-MAP); (Log-MAP); bit error rate; maximum latency; resources utilization.

I. INTRODUCTION

To solve the noise problem, a channel coding technique or also called Forward Error Correction (FEC) appeared. This technique has been used in digital communication systems [1]. Since bits are added to the main message, and these bits do not carry new information about the message, but are only added to help the recipient receive the message correctly [2]. This technology is not only used to protect digital data against errors when it is transmitted through a channel exposed to noise, but it is also used in the case of storing the data in an unreliable memory [3].

For (FEC) to be effective, it must have high performance through error correction, as well as a small delay through fast decoding [4]. There are many types of (FEC) techniques, for example, but not limited to: Hamming coding, parity bit,Reed-Solomon coding, and Convolutional Coding [5].

Turbo coding as well as low-density parity check coding (LDPC) are among the most important and widely used types of modern communication technologies. This is due to their ability to achieve performance close to the capacity of the channel (Shannon limits). Turbo codec has been used in telecom fields such as 3GPP LTE standard, IEEE802.16m standard and other applications [6].

Programmable Logic Gate Array (FPGA) is the most important integrated circuit for implementing turbo coding designs . It plays a major role in terms of flexibility and performance; This is because it combines reprogramming as well as the implementation of effective dedicated circuits for a specific application. These designs are implemented using traditional languages such as Very High-Speed Integrated Circuit Markup Language (VHDL), and the Record Transfer Language (RTL) is generated manually [7].

Recently, attentionshave been directed towards methods of using high level mounting tools (HLS). Algorithms written in a high-level language, such as C and C ++, are converted into a record transfer language directly and without the designers' intervention. This approach in design avoid the designers many low-level design problems, as well as enabled designers to explore the design space through early knowledge of the amount of reserved resources [8].

Designing with (HLS)tools provides a record transfer language designwithout errors, speeds up design time, and reduces overall design verification effort. By using high-level tools, it will enable designers to focus on the behavior required for design without focusing on how to conduct operations, hierarchy and clocks. Thus, the code will be reduced, which will lead to fewer errors, and thus the design will be fully and easily verified [9].

II. RELATED WORK

In [10] a new algorithm for decoding turbo coding was proposed, and it was implemented. Using networks on a chip (NoC). When using this proposed algorithm, the sequential action nature used in the traditional MAP algorithm is eliminated. Thus, a productivity of (2.13) was obtained using this proposed algorithm.

In [11] researchers proposed a full parallelism algorithm for turbo decoding compared with the standard long-term evolution model (LET standard) . This design was implemented using Programmable Logic Gate Array (FPGA) When using the proposed algorithm, the researchers obtained a better use of reserved sources, as the proposed model provided resources with a percentage (50%) difference from the standard long-term evolution model, when using a block size (720 bits) of data.

Researchers in [12] made a comparison between three types of coding, turbo coding, Polar coding and low-density parity check coding. The researchers observed that although turbo coding has lower throughput than polar coding and low-density parity check coding due to the serial nature of its operation . However, it showed better performance than the other two types in correcting errors. Also, the turbo codec is flexible in supporting different block lengths as well as different coding ratios better than the other two types.

The researchers in [13] designed a low-density parity validation decoder using the high-level compositing tool (Vivado HLS) and explored whether this tool was able to provide performance comparable to written designs. Using Hardware Description Language (HDL) . They found that when creating distinct program structures and using enhancements in the program, a written design using a high-level tool can approach the performance of a written design using Hardware Description Language.

In [14] three different designs were accelerated for decoding the turbo code and by using the capabilities of high-level synthesis tools in the implementation of these designs. The researcher implemented a simple sliding window for the first design, a second design implemented the parallel technology, and a third design for pipeline technology, and he concluded that it is possible to obtain good designs using (HLS) tools when writing programs through knowledge of how the devices used in the application work.

In [15] (46) articles on the quality of results and design efforts were studied. Designs implemented with (HLS) tools were compared with manual (RTL) designs. The researchers found that (40%) of the cases studied, (HLS) tools equal or exceed (RTL) designs in terms of performance and better resource use. They also studied whether the size of the design affected the quality of the performance, but did not find any correlation. Accordingly, they concluded that (HLS) tools are suitable for both large and small designs. They also conclude that the (HLS) tools provide enormous savings in time when making architectural changes to an existing design.

III.THEORETICAL BACKROUND

Figure (1) shows the general structure of the turbo-iterative decoder. Two Soft In Soft Out (SISO) decoders are connected. Each decoder takes three inputs: the systematic bit and parity bits, all received from the channel plus information from the other decoder about the possible values of the decoded bits. This information from another decoder is referred to as extrinsic information. The decoders must exploit both the input from the channel and this preset information. It should also provide what is known as soft output for the decoded bits. Usually the flexible output is represented by what is called the logarithmic probability ratio (LLR) as the polarity of this ratio determines the bit sign whether it is negative or positive, and from this the decision is made that the bit is (0) or (1), while the amplitude of the value,this ratio determines the probability of a correct decision being taken, so that the greater the value, the greater the confidence in the decision [16].

The SISO1 decoder receives the systematic bits (y) and parity bits (p1) from the channel, and here the extrinsicinformation for the first decoder at the beginning is equal to (0), and this is only the case at the beginning . After the first decoder extracts the LLR values, the bits are interleaved by the interleaver after the information received from the channel and the preset information from the previous decoder are subtracted. Then they are sent to the second decoder (SISO2), which in turn relies on the information received from the channel the systematic bits (y), but they are interleaved according to the interleaver used in the encoder, as well as the parity bits (p2) in addition to the present information sent by (SISO1) . After the second decoder calculates the values of (LLR), it sends the preset information to the decoder (SISO1) and the process is repeated iteratively. The final result is obtained by the decoder (SISO2) and then a decision is made whether the bit is (0) or (1) . A good result is obtained with a large number of iterations and the improvement in performance is weak after 20 iterations. For shorter blocks, the required number of iterations tends to be reduced, and turbo decoders usually carry out the process from 6 to 10 iterations[17].

The maximum a posterior probability algorithm (MAP) is much more complex than the Viterbi algorithm (used in the convolutional decoder), so it has been ignored for nearly twenty years. However, the concept of turbo coding has renewed interest in this algorithm, and it has been realized that its complexity can be greatly reduced without affecting its performance. The roots of the use of the (Max-Log-MAP) algorithm, based on the 1989 Jacobian logarithmic approximation, go back to (Max-Log-MAP) simplifies the (MAP) algorithm by transferring arithmetic operations to the logarithmic domain and calling an approximation in order to reduce implementation complexity. Because of this approximation, the performance provided by (Max-Log-MAP) algorithms is sub-optimal. However, in 1995 the (Log-MAP) algorithm was proposed, which corrected the approximation used in the (Max-Log-MAP) algorithm and then achieved almost identical performance to that of the (MAP) algorithm in part of its complexity [1].

Thus, the equations used in the decoders are as explained by the following equations:

$$L(u_k|y) = max^*_{u_k=+1}[A_{k-1}(S)+\Gamma_k(S',S)+B_k(S)] - max^*_{u_k=-1}[A_{k-1}(S)+\Gamma_k(S',S)+B_k(S)] \quad \dots(1.2)$$

$$\Gamma_k(S',S) = \ln C_k + \frac{u_k * l(u_k)}{2} + \frac{L_c}{2} * \sum_{l=1}^{n} x_{kl} * y_{kl} \dots(2.2)$$

$$A_k(S) = max^*_{S'}[A_{k-1}(S)+\Gamma_k(S',S)] \quad \dots(3.2)$$

$$B_{k-1}(S') = max^*_{S'}[B_k(S)+\Gamma_k(S',S)] \quad \dots(4.2)$$

Where $L(u_k|y)$is represented the bit value,(S) is the current state, (S') is the next state, (Γ) is the transition probability, (A) is the forward recursion, (B) is the backward recursion, (L_c) is the channel reliability, the value of (C_k) is not important and it will be cancelled.

IV.METHOLODOGY

The turbo decoder was implemented by using three different design methods. The (Log-MAP) algorithm was used in both (SISO1) and (SISO2) decoders in the first design (Design1) . In the second design (Design2), the (Max-Log-MAP) algorithm was used in both (SISO1) and (SISO2) decoders. Where, in the third design (Design3) proposed by this work, the combination of the two algorithms was used. In the third design, the (Max-Log-MAP) algorithm was used in the (SISO1) decoder, and in the (SISO2) decoder the algorithm was used (Log-MAP).

For calculating the Bit Error Rate (BER) for all three designs, the general model shown in Figure (2) was designed, which represents how the message is generated and encoded, then using (BPSK) to represent bits, then (AWGN) noise is generated and combined with the signal to represent the transmission of the signal through the transport channel, then the message is decoded, and then the bits are returned to their numerical values 0 and 1 . Then the decoded bits are compared with the original message bits and calculate the (BER) values.The block size was considered (5476 bits) and this value were repeated for (100000) block.

Also, in this work one memory element was used for the recursive systematic convolutional encoder which used in the turbo encode with (1/3) coding rate, the zero-termination method was used for turbo encoder termination, and the block interleaver with square array was used for the interleaver type. Two different interleaver size were designed (16, and 5476),three different number of iterations were used in the turbo decoder (1, 2, and 6), and seven different value of (E_b/N_o) (0-3 db). All these designs were done using (Vivado HLS), and implemented using (ZYNQ UltraScale$^+$ ZCU102 Evaluation Platform). The clock was specified at a value (10 ns).

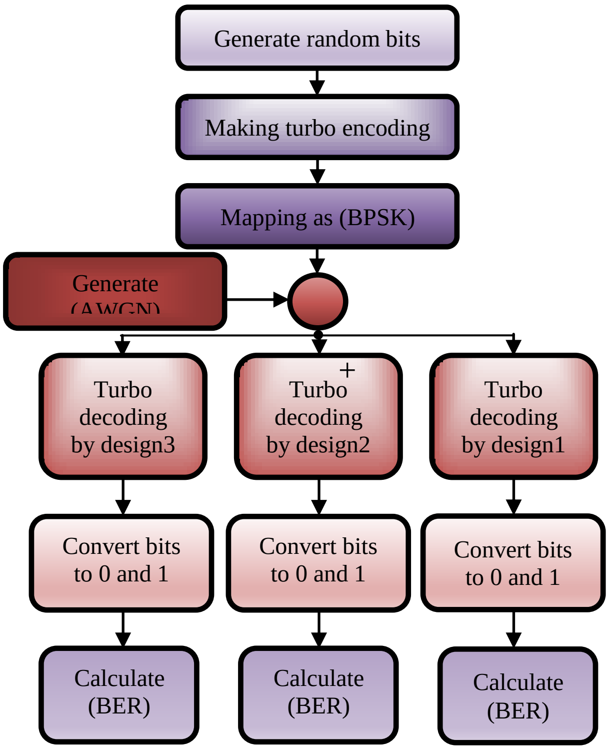

Fig.2. the system for calculating (BER)

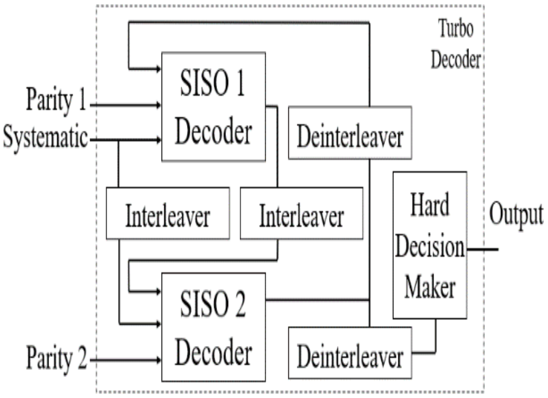

Fig.1. Block diagram of turbo decoder [18]

V. SIMULATION RESULTS

The (BER) values were calculated for the 16-bit interleaver size and the interleaver size 5476 bits. And were calculated for the number of iterations (1, 2, and 6) and the (E_b/N_o) values mentioned previously. The values can be observed in table (1). And for finding the gain for proposed design (design 3), the average value of (BER) for the designs 1 and 2 is subtracted from the (BER) value of design 3. If the gain is equals to zero that mean there is no benefit, if the gain is negative value that will mean the performance of the design is not good, and if the gain is positive value that mean the performance of the proposed design is better. Table (2) show the gain of proposed design.

Table (1): the value of (BER)*10^{-3} for the three designs

Interleaver size	16									5476								
Number of iterations	1			2			6			1			2			6		
Design type (E_b/N_o)	Design 1	Design 2	Design 3	Design 1	Design 2	Design 3	Design 1	Design 2	Design 3	Design 1	Design 2	Design 3	Design 1	Design 2	Design 3	Design 1	Design 2	Design 3
0 db	101.717	105.195	103.092	87.536	95.165	91.401	85.912	89.214	88.033	106.352	110.363	107.842	92.352	100.309	95.774	90.462	100.570	94.476
0.5 db	83.722	85.879	84.293	69.624	75.352	72.569	67.669	69.903	69.123	86.832	89.279	87.404	70.939	76.747	73.441	68.360	76.072	71.521
1 db	67.005	68.258	67.076	53.578	57.700	55.707	51.477	52.948	52.440	68.590	69.880	68.557	51.502	55.488	53.221	48.357	53.822	50.687
1.5 db	51.949	52.588	51.746	39.745	42.558	41.192	37.650	38.564	38.242	52.047	52.588	51.699	34.970	37.489	36.063	31.744	35.228	33.294
2 db	38.877	39.147	38.558	28.338	30.142	29.251	26.435	26.981	26.781	37.708	37.761	37.213	22.019	23.487	22.662	19.230	21.158	20.116
2.5 db	28.020	28.088	27.695	19.370	20.463	19.909	17.753	18.063	17.940	25.898	25.732	25.416	12.808	13.584	13.149	10.847	11.753	11.267
3 db	19.378	19.360	19.107	12.661	13.276	12.951	11.400	11.567	11.494	16.710	16.487	16.324	6.911	7.286	7.076	5.705	6.056	5.868

Table (2): the gain of design3 in (BER)*10^{-3}

Interleaver size	16			5476		
Gain for iteration (E_b/N_o)	Gain for (1iteration)	Gain for (2iterations)	Gain for (6iterations)	Gain for (1iteration)	Gain for (2iterations)	Gain for (6iterations)
0 db	-0.3640	0.0505	0.4700	-0.5155	-0.5565	-1.0825
0.5 db	-0.5075	0.0810	0.3370	-0.6515	-0.4020	-0.8235
1 db	-0.5555	0.0680	0.2275	-0.6780	-0.2740	-0.5270
1.5 db	-0.5225	0.0405	0.1350	-0.6185	-0.1665	-0.2485
2 db	-0.4540	0.0110	0.0730	-0.5215	-0.0910	-0.0805
2.5 db	-0.3590	-0.0075	0.0320	-0.3990	-0.0470	-0.0230
3 db	-0.2620	-0.0175	0.0105	-0.2745	-0.0225	-0.0035

From table (1) we notice that when the number of iterations is equal to (1) the performance is better when the interleaver size small; but the performance is better for large size of interleaver when the number of iterations is increases.

We notice from the table (2) that the performance is better when the size of the interleaver is equal to (16 bits) and it is better than the average of designs 1 and 2, especially when the number of iterations increases. From table (3) we notice that the maximum latency for design3 is equal to the average maximum latency of design1 and 2. And from table (4) can notice that the resource utilization of design3 equal to the average resource utilization of design 1 and 2.

Table (3): the maximum latency

Interleaver size	16			5476		
latency Number of iterations	Max $*10^6$for design1	Max $*10^6$ for design2	Max $*10^6$ for design3	Max $*10^6$ for design1	Max $*10^6$ for design2	Max $*10^6$ for design3
1	3.884977	0.844561	2.364769	3.373295	0.711959	2.042627
2	7.695663	1.614831	4.655247	6.680881	1.358209	4.019545
6	22.935279	4.692783	13.814031	19.911217	3.943201	11.927209

Table (4): the resource utilization

Interleaver size	16						5476					
Design type	Design 1		Design 2		Design 3		Design 1		Design 2		Design 3	
Resources Number of iterations	LUT	FF	LUT	FF	LUT	FF	LUT	FF	LUT	FF	LUT	FF
1	42167	26145	18631	12033	30399	19089	42533	26836	18997	12708	30765	19772
2	42198	26150	18662	12038	30430	19094	42564	26841	19028	12713	30796	19777
6	42200	26152	18664	12040	30432	19096	42566	26843	19030	12715	30798	19779

VI. CONCLUTION AND FUTURE WORK

After the discussion of the simulation results,we conclude that the proposed design can be used in turbo decoder for the systems that require medium resource utilization as well as medium latency. The results of (BER) showed that the proposed design made performance better than the average of the other two designs when the interleaver size was small (16 bits). And for future work find if we can do another combination between other algorithms. And study what happened if we use this technique in reverse.

REFFRENCES

1. Hanzo, Lajos, et al. “Turbo coding, turbo equalisation and space-time coding: EXIT-chart-aided near-capacity designs for wireless channels.” vol. 22. John Wiley & Sons, 2011.
2. Rao, K. Deergha. “Channel coding techniques for wireless communications.” Springer India, 2015.
3. Johannesson, Rolf, and Kamil Sh Zigangirov. “Fundamentals of convolutional coding.” John Wiley & Sons, 2015.
4. Cancellieri, Giovanni. “Polynomial theory of error correcting codes.” Springer International Publishing, 2015.
5. Chiueh, Tzi-Dar, et al. “Baseband receiver design for wirelessMIMO-OFDM communications.” Singapore: Wiley, 2012.
6. Giulietti, Alexandre, Bruno Bougard, and Liesbet Van Der Perre. “Turbo Codes: Desirable and Designable.” Springer Science & Business Media, 2012.
7. Winterstein, Felix. “Separation Logic for High-level Synthesis.” Springer, 2017.
8. Kundu, Sudipta, Sorin Lerner, and Rajesh K. Gupta. “High-Level Verification: Methods and Tools for Verification of System-Level Designs.” Springer Science & Business Media, 2011.
9. Fingeroff, Michael. “High-level synthesis: blue book.” Xlibris Corporation, 2010.
10. Li, An, et al. “A scalable turbo decoding algorithm for high-throughput network-on-chip implementation.” IEEE Access, vol. 4, pp. 9880-9894, 2016.
11. Li, An, et al. “1.5 Gbit/s FPGA implementation of a fully-parallel turbo decoder designed for mission-critical machine-type communication applications.” IEEE Access, vol. 4, pp. 5452-5473, 2016.
12. S. Shao et al. “Survey of Turbo, LDPC, and Polar Decoder ASIC Implementations.” in IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2309-2333, 2019.
13. J. Andrade et al. “From low-architectural expertise up to high-throughput non-binary LDPC decoders: Optimization guidelines using high-level synthesis.” 25th International Conference on Field Programmable Logic and Applications (FPL), London, pp. 1-8, 2015.
14. B. E. Conn. “Exploring High Level Synthesis to Improve the Design of Turbo Code Error Correction in a Software Defined Radio Context.” Master’s thesis, pp. 109, 2018.
15. S. Lahti, P. Sjövall, J. Vanne and T. D. Hämäläinen. “Are We There Yet? A Study on the State of High-Level Synthesis.” in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 38, no. 5, pp. 898-911, 2019.
16. Shu Lin and Daniel J. Costello. “Error Control Coding, Second Edition.” Prentice-Hall, 2004.
17. Declercq, David, Marc Fossorier, and Ezio Biglieri. “Channel Coding: Theory, Algorithms, and Applications: Academic Press Library in Mobile and Wireless Communications.” Academic Press, 2014.
18. Amer T. Ali, Dhafir A. Alneema. "Design Analysis of Turbo Decoder Based on One MAP Decoder Using High Level Synthesis Tool." Al-Rafidain Engineering Journal (AREJ), Vol.25, No.1, pp.70-77, 2020.

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

☐ 9940 572 462 ☎ 6381 907 438 ✉ ijircce@gmail.com

www.ijircce.com

Scan to save the contact details