AN FPGA IMPLEMENTATION BELIEF PROPAGATION DECODING ALGORITHM

M. M. Jadhav[1], Chetna N. Kharkar[2] , Dr. A. M. Sapkal[3]

Abstract — *Low Density Parity Check (LDPC) codes are one of the best error correcting codes that enable the future generations of wireless devices. In this paper we have presented an FPGA based self error checking & correction system with higher capability. We implemented the error correction algorithms such as belief propagation & bit flip algorithm. The number of error corrected is simulated using Xilinx High level Synthesis tool & Modelsim simulator. A complete self checking system is implemented on Xilinx Spartan 3E FPGA & synthesis is done using ISE13.2. The decoded codeword of the decoder is displayed on PC HyperTerminal using serial Rs232 interface. The system is capable of correcting large & different size of codeword. Self error detection & correction capability of the system is compared & verified in presence of noise.*

Index Terms — *LDPC codes, LDPC decoder, FPGA Implementation, bit flip decoding, Belief Propagation, log domain, soft decoding, hard decoding*

1. INTRODUCTION

Low-density parity-check (LDPC) codes are forward error-correction codes [1], first proposed in the 1962 PhD thesis of Gallager at MIT[2].The computational effort in implementing coder & encoder are more for these codes.LDPC codes is then rediscovered by Mackay and Neal [3].These codes are decoded iteratively by Belief propagation decoding on their associated factor graph [4]. In the mean time the field of forward error correction was dominated by highly structured algebraic block and convolution codes.

Optimization results for various channels, such as the Additive White Gaussian Noise (AWGN) channel and the Binary Symmetric Channel (BSC) have produced specific degree distributions such that the corresponding codes come very close to capacity [5].In fact, the state of art in the coding field is currently achieved by LDPC codes, which was accomplished by Chung et al. [6].It was soon recognized that these block codes were in fact a rediscovery of the LDPC codes developed years earlier by Gallager. New generalizations of Gallager's LDPC codes by a number of researchers produced new irregular LDPC codes. These codes easily outperform the best turbo codes, as well as offering certain practical advantages and cleaner setup for theoretical results. Today, design techniques for LDPC codes exist which enable the construction of codes which approach the Shannon's capacity to within hundredths of a decibel. Low density parity check (LDPC) codes have been extensively adopted in next-generation forward error correction applications. They achieve very good performance using the iterative decoding approach of the belief-propagation (BP) [7].The main contribution of this paper is implementation of self error checking system on Xilinx Spartan 3E FPGA using High level synthesis tool and Synthesis has been done for LDPC codes Construction & Bit-flipping decoding. Self error checking system is flexible to

[1] **M. M. Jadhav** is presently perusing PHD from Government college of Engineering, Pune in the field of communication. He has guided graduates & more than ten Post graduates students, His area of Interest is communication system .Email- makj123@yahoo.com

[2] ***Chetna N. Kharkar*** *is Currently* Working as a Sr. Design Engineer at Qualitat systems, Pune (India).She has a 6+ years of experience in Embedded domain. Perusing Master of Engineering from Sinhgad college of Engineering,Pune,*India,Email-****chetnakharkar@gmail.com***

[3] **Dr. A. M. Sapkal** is HOD of E&TC Department at government college of Engineering, Pune.He has 23 years of experience in teaching, 3 years Industrial & 14 years of Research experience. He has guided More than 100 post graduates students. Email -***hod.extc@coep.ac.in***

use for any length of code word (or) data word and also for any rate of code word. So the usage of this code leads to high performance. The above decoding algorithms can recover the original codeword in the presence of large amounts of noise.Decoding algorithms such as belief propagation & Bit flip algorithm are simulated, error detection and correction capability of both algorithms compared & verified

2. DESIGN OF DECODING ALGORITHM FOR ERROR CHECKING

The decoding algorithm can be stated in three steps:

i. Initialization
The decoding algorithm is fed by the log-likelihood ratio as represented in equation (2).
$r_0[i] = \log\{P(c_i = 0 \mid y_i) / P(c_i = 1 \mid y_i)\}$ -------(1)

ii. Syndrome Check
The code word simply multiplied by the transpose of H . If the result is all zero vectors then the code word is valid.
$H^T * Z = 0$ -------(2)

iii.The Iterative Decoding
The algorithms used to decode LDPC codes are the belief propagation algorithm or the sum-product algorithm, bit flipping algorithm.

2.1 Belief propagation decoding algorithm

Decoding algorithm consists of three steps computed iteratively
1. The data node process (data update) computes $M_{j,i}$ messages from $E_{j,i}$ messages and log-likely hood ratios. For each data node, *i* being the degree of the node and r_i is the log-likelihood ratio entering this node:

$$M_{j,i} = \sum_{j' \in Ai, j' \neq j} E_{j',i} + r_i$$

2. The check node process (check update) computes $E_{j,i}$ messages. For each check node (j being the degree of the node):

$E_{j',i} = (\prod \alpha_{j,i'}) \, g \, (\Sigma \, f(\beta_{j,i'}))$
where $\alpha_{j,i'} = \text{sign}\,(M_{j,i})'$ and $\beta_{j,i'} = | M_{j,i} |'$

3. Two math functions are used in step 2 for transposing the initial needed convolution in the log-Fourier domain. These functions are defined as,
$f(x) = \log [\tanh (|x|/2)]$

$g(x) = 2 \text{ x } \tanh^{-1} (\exp(x))$

4. The hard decision process that computes the estimated value of each symbol of the code. For each data node:
$L_i = r_i + \Sigma E_{j,i}$

2.2 Bit flip Decoding Algorithm

Algorithm: Bit-flipping Decoding
Procedure Decode(y)
for i = 1 : n **do**
$M_i = y_i$
end for
repeat
for j = 1 : m **do**
for i = 1 : n **do**
$E_{j,i} = \sum_{i' \in Bj, i', i \neq i'} (M_{i'} \bmod 2)$
end for
end for

for i = 1 : n **do**
if the messages $E_{j,i}$ disagree with y_i **then**
$M_i = (r_i + 1 \bmod 2)$
end if
end for

for j = 1 : m **do** $L_j = \sum_{i' \in Bj} (M_{i'} \bmod 2)$
end for

if all $L_j = 0$ or $I = I_{max}$ **then** Finished
else
$I = I + 1$
end if
until Finished
end procedure

3. EXPERIMENTAL SETUP

We have Implemented Bit flip decoder & belief propagation decoder algorithm on Xilinx Spartan 3E starter Board .The output of the decoder is shown on PC (computer) HyperTerminal. Figure 1 shows the experimental setup & figure 2 shows

Decoder codeword on PC HyperTerminal the hardware for hardware implementation of the decoder algorithm. The results of bit flip decoder & belief propagation algorithm is compared in terms of number of error recovered. The results show that belief propagation algorithm has a better error correction capability than the bit flip decoder.

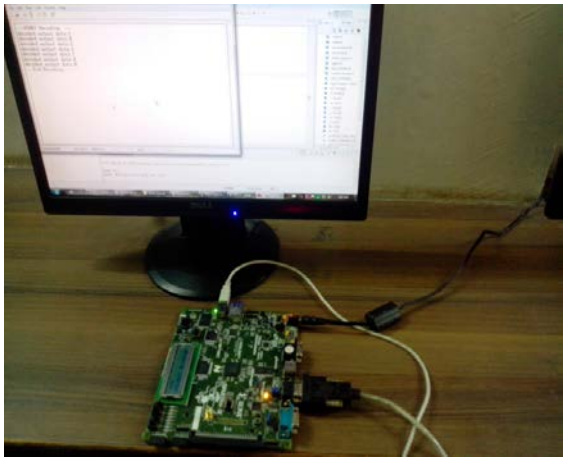

Figure 1: Experimental setup

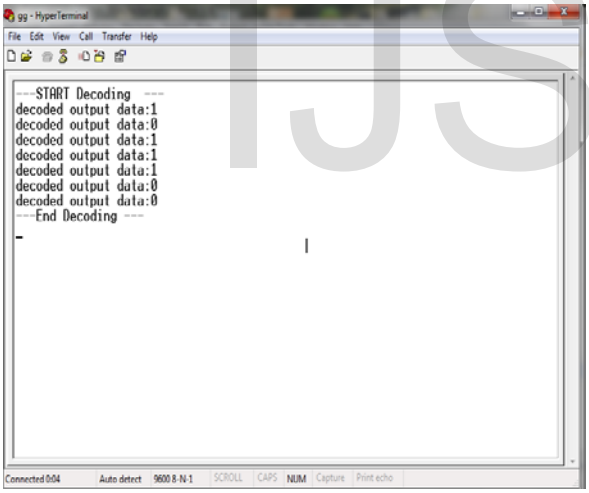

Figure 2: Decoded codeword on serial Port HyperTerminal

4. FPGA IMPLEMENTATION OF ERROR CHECKING DECODER

FPGA implementation of the decoder is done for large H matrix & results are simulated using High level Simulator. The design is synthesized and implemented using Xilinx ISE 13.2. The Xilinx High level synthesis Tool converts the High level c language code into HDL. The Synthesis is done using the High level synthesis & Xilinx ISE 13.2, built in synthesis tool. Table 1, shows synthesis report which gives information of the Slice flip flops, LUTs usage, device utilization constraint, timing summary. Figure 3, shows the simulation results which show the working of the decoder. The reset & clock activates the other signals ,when reset is '1' ,the decoder is in Idle condition, when reset'0' ,decoder operation starts on the clock tick event. Figure1 shows the H-matrix used for decoding& the output Z required decoded codeword

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	1224	4656	26%
Number of Slice Flip Flops	1793	9312	19%
Number of 4 input LUTs	1355	9312	14%
Number of bonded IOBs	38	232	16%
Number of BRAMs	5	20	25%
Number of MULT18X18SIOs	2	20	10%
Number of GCLKs	1	24	4%

Table 1: Xilinx device utilization summary

5. FPGA IMPLEMENTATION RESULT

The decoding performance of belief propagation and bit flip algorithms in recovering original codeword, shown in the table Table 2.

MSG Bits	H Matrix	Codeword length	No. of errors recovered by BPD	No. of errors recovered by BFD
64	64X128	128	7	6
128	128X256	256	9	8
256	256X512	512	14	7
512	512X1024	1024	18	16
2048	2048X4096	4096	46	34

Table 2: Comparison of Number of errors recovered by Belief propagation & Bit flip Algorithm

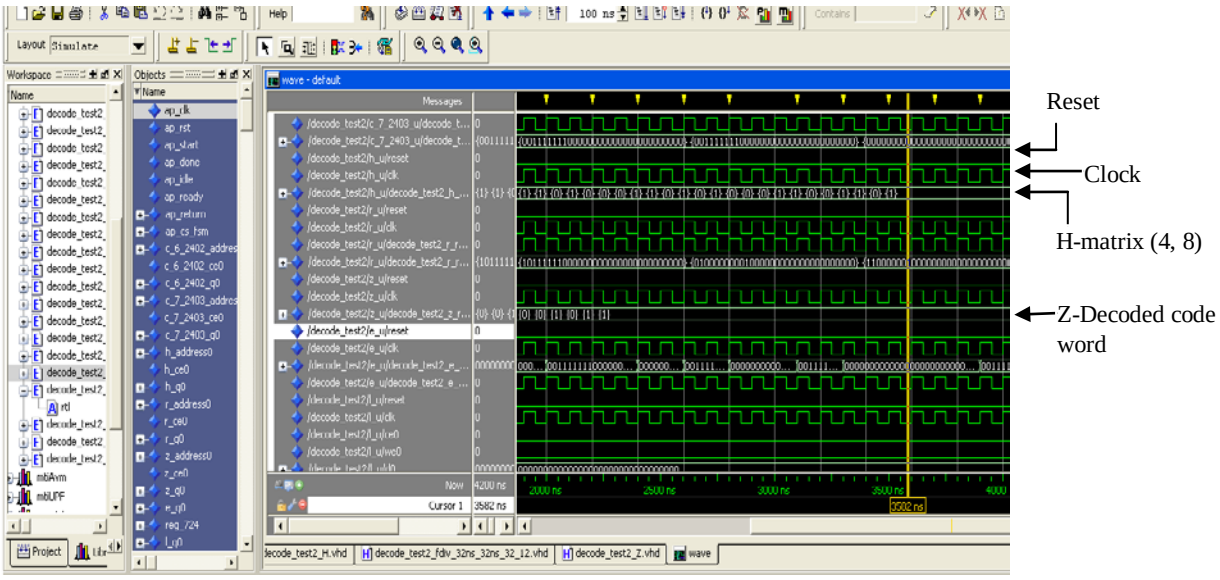

Figure 3: Simulation Result for H-matrix 4X8 on Modelsim simulator

6. CONCLUSION

The decoder was designed using High level Language & synthesized using Xilinx High level synthesis tool to Implemented on Hardware. The simulation results shows how the clock & reset signal controls the decoder operation. The implementation of self checking system on FPGA for belief propagation algorithm has resulted in correction & detection of errors on large size codeword. Belief propagation & bit flip algorithm results are compared & verified using Simulation & implementation. From the performance parameter it is concluded that Belief propagation decoding algorithm improves the performance of system in terms of error correction rate. Error recovery rate of belief propagation algorithm is better than bit flip decoding algorithm. The detailed decoder design and architecture are presented and the results are explained.

REFERENCES

[1]. Susmitha Remmanapudi and Balaji Bandaru, "An FPGA Implementation of low density Parity-check codes construction & Decoding"Devices ,Circuits & Systems International conference,pp .216-220,April 2012

[2]. R. Gallager, Low-Density Parity-Check Codes. Cambridge, M.I.T,Press, 1963.

[3]. D.I.C. MacKay, and R.M. Neal, "Near Shannon limit performance of low density parity check codes," lEE Electronics Letters, vol. 32, pp. 1645-1655, Aug. 1996.

[4]. F. Kschischang, B. Frey, and H.A. Loeliger, "Factor Graphs and the Sum Product Algorithm," IEEE Trans. Inform. Theory, vol. 47, pp. 498-519, Feb, 2001.

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

[6]. S.Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the Design of Low-Density Parity-Check Codes within 0,0045 dB of the Shannon Limit," IEEE Trans. Inform. Theory, vol. 47, pp. 638-656, Feb. 2001.

[7]. R. Lucas, M. P. C. Fossorier, Y. Kou and S. Lin 2000 Iterative decoding of one-step majority logic decodable codes based on belief propagation. IEEE Transaction communication, 48: 931–937