WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

LDPC Decoders for the Design and Implementation of High Performance and Low Cost Techniques

A.Swapna[1], R. Vinay Kumar [2]

Research Scholar, Department of E.C.E, International School of Technology and Sciences, East Godavari, Andhra Pradesh, India.
Assistant Professor, Department of E.C.E, International School of Technology and Sciences,
East Godavari, Andhra Pradesh, India.

ABSTRACT

There may be fluctuations in the parameters and noise levels as a result of technology scaling and increasing integration densities, which would increase error rates at different levels of the computations. Soft mistakes and single event upsets are a constant source of trouble for memory applications. The primary focus of the work is on the design of an effective Multi Detector/Decoder (MLDD) for fault detection and correction for memory applications, which significantly cuts down on the time required for fault detection. For Euclidean Geometry Low Density Parity Check Codes, the error detection and correction approach uses one-step majority logic decoding (EG-LDPC). Although the majority of decodable codes can discover and rectify a huge number of faults, doing so requires a lengthy decoding process. Decoding techniques may take the same amount of time to find faults for both incorrect and error-free code words, which slows down memory performance. The suggested fault-detection method requires less decoding cycles to find the error (almost in three). It is evident that reading error-free data can speed up memory access. For big code wordsizes, the method keeps area overhead to a minimum and power usage to a minimum..

Introduction

Low-density parity-check (LDPC) codes are a type of linear error correcting code used to transfer data over noisy transmission channels in information theory. A sparse bipartite graph is used to build an LDPC. Because LDPC codes are capacity-approaching codes, it is possible to put the noise threshold very close to the theoretical upper limit (the Shannon limit) for a symmetric memoryless channel, or even arbitrarily close to the BEC. The noise threshold establishes a limit for the channel noise at which the chance of information loss can be reduced to the desired level. Iterative belief propagation algorithms enable the time-linear block-length decoding of LDPC codes.

.

Applications needing dependable and highly effective data transfer over bandwidth or return channel-constrained networks in the presence of corrupting noise are increasingly using LDPC codes. LDPC code implementation has lagged behind that of other codes, particularly turbo codes. On August 29, 2013, the primary patent for Turbo Codes expired.

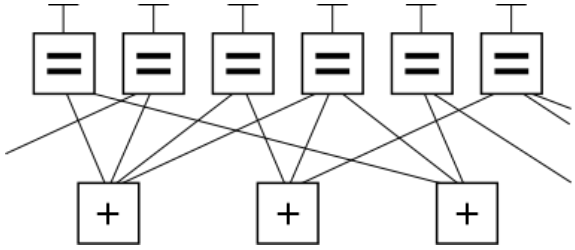
In honour of Robert G. Gallager, who created the LDPC concept in his PhD dissertation at the Massachusetts Institute of Technology in 1960, LDPC codes are sometimes known as Gallager codes..

Implementation of LDPC inVHDL

a. **Basic Terminology in Communication Systems:** A sparse parity-check matrix serves as the definition of LDPC codes. The development of the LDPC code is detailed below. This sparse matrix is frequently randomly generated, according to the sparsity limitations. Robert Gallager created these codes for the first time in 1962.

An example LDPC code utilising Forney's factor graph notation is shown in the graph fragment below. In this graph, the (nk) constraint nodes at the bottom are connected to the (n) variable nodes at the top. This is a common method for displaying a (n, k) LDPC code graphically. The bits of a valid message fulfil thegraphical restrictions when they are arranged on the Ts at the top of the graph. To be more precise, all lines connecting to a variable node (box with a

"=" sign) must have the same value, and all values linking to a factor node (box with a "+" symbol) must total to zero, modulo two (i.e., they must sum to an even number, or there must be an even number of odd values).



Eight six-bit strings that correspond to legal codewords are possible, ignoring any lines that leave the frame: (i.e., 000000, 011001, 110010, 101011, 111100, 100101, 001110, 010111). This six-bit LDPC code fragment represents a three-bit message. Here, redundancy is employed to boost the likelihood of recovering from channel faults. With n = 6 and k = 3, this is a linear code of the form (6, 3).
The parity-check matrix encoding this graph fragment, once more excluding lines leaving the frame, is

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}_1 \sim \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix}_2 \sim \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}_3 \sim \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}_4 .$$

In this matrix, each row represents one of the three parity-check constraints, while each column represents one of the six bits in the received codeword. In this example, the eight codewords can be obtained by putting the parity-check matrixH into this form $[-P^T|I_{n-k}]$through basic row operations in$GF(2)$:

Step 1: H.

Step 2: Row 1 is added to row 3.

Step 3: Row 2 and 3 are swapped.

Step 4: Row 1 is added to row 3. From this, the generator matrixG can beobtainedas $[I_K|P]$ (noting that in the special case of this being a binary code P=-P), orspecifically.

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

:

Finally, by multiplying all eight possible 3-bit strings by G, all eight valid codewords are obtained. For example, the codeword for the bit-string '101' is obtained by:

$$\begin{pmatrix} 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

As a check, the row space of G is orthogonal to H suchthat $GH^T = 0$

b. Encoding:

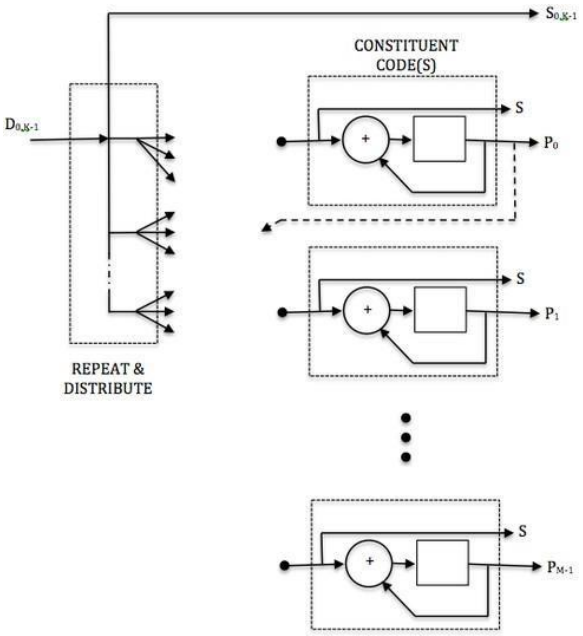


Fig. 1. LDPC Encoder

The input data bits (D) are repeated and distributed to a group of individual encoders during the encoding of a frame. Each of the constituent encoders, which are often accumulators, is utilised to produce a parity symbol. The parity bits (P) and the original data (S0,K-1) are transferred together to form the coding symbols. Each constituent encoder's S bits are discarded.
Another constituent code may make use of the parity bit..

The encoded block size in an example utilising the DVB-S2 rate 2/3 coding is 64800 symbols (N=64800), 43200 data bits (K=43200), and 21600 parity bits (M=21600). Except for the initial parity bit, which encodes 8 data bits, each constituent code (check node) encodes 16 data bits. While the remaining data bits are utilised in 3 parity codes, the first 4680 data bits are repeated 13 times (used in 13 parity codes) (irregular LDPC code).

In contrast, conventional turbo codes often use two constituent codes that are configured in tandem and each of which encrypts the entire input block (K) of data bits. These constituent encoders are made up of recursive convolutional codes (RSC) of intermediate depth (8 or 16 states), and they are separated from one another by a code interleaver that interleaves one copy of the frame..

A large number of low depth constituent codes (accumulators) are used concurrently by the LDPC code, although each one only encrypts a small amount of the input frame. It is possible to think of the several constituent codes as a collection of low depth (2 state) "convolutional codes" that are linked together by repeat and disperse operations. In the turbo code, the repeat and distribute operations serve as the interleaver..

LDPC codes can sometimes perform better than turbo codes because of the ability to more precisely manipulate the connections between the multiple constituent codes and the degree of redundancy for each input bit. At low code rates, TurboCodes still appear to perform better than LDPCs, or at the very least, designing high-performing low rate codes is simpler with TurboCodes..

Practically, during the encoding process, the hardware that makes up the accumulators is reused. In other words, the same accumulator hardware is utilised to generate a second set of parity bits after a first set of parity bits has been generated and stored.

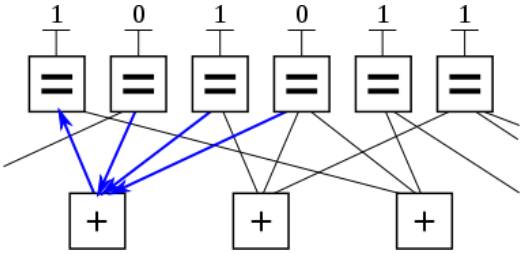
.

c. Decoding: The maximum likelihood decoding of an LDPC code on the binarysymmetric channel is an NP-complete issue, as it is with other codes. It is impractical to carry out optimal decoding for an NP-complete code of any useful size.

However, imperfect methods based on iterative belief propagation decoding produce good outcomes and are practically applicable. Each parity check that makes up the LDPC is seen as a separate single parity check (SPC) code by the sub-optimal decoding algorithms. Using soft-in-soft-out (SISO) algorithms like SOVA, BCJR, MAP, and other variations, each SPC code is decoded separately. Each duplicate SPC decoding of the same information bit cross-checks and updates the soft decision information from each SISO decoding. The modified soft decision information is then used to decode each SPC code once more. This approach is repeated until a workable code word is found or all possibilities for decoding have been explored. Sum-product decoding is a common name for this sort of decoding.
The cross-checking of the variables is frequently referred to as the "variable-node" processing, and the decoding of the SPC codes is frequently referred to as the "check node" processing.
Sets of SPC codes are decoded simultaneously in a practical LDPC decoder implementation to boost throughput.
Contrarily, belief propagation on the binary erasure channel, where it consists of iterative constraint satisfaction, is relatively straightforward..

The valid codeword 101011, for instance, might be sent over a binary erasure channel and received with the first and fourth bits erased, resulting in? 01? 11. The message can be expressed by writing the received message at the top of the factor graph because the transmitted message had to adhere to the code requirements..

Because all of the constraints associated to the first bit in this case have more than one unknown bit, the first bit cannot yet be recovered. Constraints connecting to just one of the erased bits must be found in order to continue decoding the message. Only the second constraint is required in this instance. The fourth bit had to be zero in order to satisfy the second condition because only a zero in that location would do so.
Then, this process is repeated. As seen below, the first bit may now be recovered using the new value for the fourth bit in conjunction with the first restriction. In order to satisfy the leftmost restriction, the first bit must be a one.



As a result, the message may be decoded repeatedly. Other channel models use real numbers to communicate probabilities and likelihoods of belief between the variable nodes and check nodes.

The revised codewordr can be used to verify this result by multiplying it by the parity-check matrix H:

$$\mathbf{z} = \mathbf{Hr} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

The generated codewords are correctly validated because the operation's z (the syndrome) output is the vector with the values three and one zero.

This erasure example, while instructive, does not demonstrate the usage of soft-decision decoding or soft-decision message transmission, both of which are present in almost every commercial LDPC decoder..

d. Updating NodeInformation: Additionally, a lot of time and effort have been put into exploring the implications of various schedules for variable-node and constraint-node update in recent years. Decoding LDPC codes was first accomplished using a method known as flooding. This sort of update requires that all constraint nodes be modified before changing any variable nodes, and vice versa. Later research by Vila Casado et al. examined other update strategies in which variable nodes are updated with the most recent check-nodeinformation available..
These algorithms operate on the premise that the variable nodes whose values vary the most should be updated first. Nodes that are very trustworthy do not require updates as frequently as other nodes whose sign and magnitude fluctuate more widely. These nodes have large log-likelihood ratios (LLR) and do not change dramatically from one update to the next. Compared to scheduling algorithms that employ flooding, these ones exhibit faster convergence and lower error levels. Because the Informed Dynamic Scheduling (IDS)[15] algorithm is able to avoid trapping sets of nearcodewords, these reduced error floors are made possible. [16]
An alternate concept of iteration is utilised in nonflooding scheduling algorithms. Regardless of the sequence in which

they were updated, a full iteration for a (n, k) LDPC code of rate k/n happens once n variable and n k constraint nodes have been updated.

Lookup table decoding

Lookup tables can be used to decode LDPC codes on a CPU that consumes relatively little power.
There are applications that use lower-power processors and small block lengths, despite the fact that codes like the LDPC are often implemented on high-power processors with long block lengths (1024).
Consequently, it is possible to precalculate the output bit based on input bits that have been predetermined. A table is created that has n entries and all potential entries for various input states. This table would be 1024 bits long for a block length of 1024 bits (both errored and nonerrored).
The value of the FIFO register is then used to seek up in the table the relevant output from the precalculated values as a bit is input and added to a FIFO register.
By using this technique, very high iterations can be employed with minimal processing overhead, with the memory for the lookup table being the only expense. As a result, LDPC decoding is doable even on a 4.0 MHz PIC microcontroller.

e. CodeConstruction: LDPC codes are frequently created for big block sizes by first analysing the behaviour of decoders. It can be demonstrated that LDPC decoders have a noise threshold below which decoding is consistently done and above which decoding is not achieved, a phenomenon known as the cliff effect. This noise threshold exists as the block size approaches toward infinity. By determining the ideal ratio of arcs from check nodes and arcs from variable nodes, this threshold can be improved. An EXITchart is a rough graphical representation of this barrier.
The construction of a specific LDPC code after this optimization falls into two main types of techniques:

- Pseudorandomapproaches
- Combinatorialapproaches

Theoretical findings that a random design provides acceptable decoding performance for big block sizes are the foundation for creation using a pseudo-random approach. [7] The finest decoders for pseudorandom codes can have simple encoders, although most pseudorandom codes have complex encoders. To ensure that the desired properties anticipated at the theoretical limit of infinite block size exist at a finite block size, a variety of constraints are frequently used.
Combinatorial methods can be used to improve the characteristics of LDPC codes with short blocks or to build codes with straightforward encoders..

Some LDPC codes, such the RS-LDPC code used in the 10 Gigabit Ethernet standard, are based on Reed-Solomon codes. Structured LDPC codes, such as the LDPC code used in the DVB-S2 standard, have simpler and thus less expensive hardware than randomly produced LDPC codes. codes created specifically so that the H matrix is a circulant matrix

Experiments and Results

a. Simulation result: To ensure that the code is proper, simulation is used. The Verilog coding technique is employed in this paper. Verilog code is intended to check for flaws such as logical and syntax issues. ModelSim is used for simulation. The simulation results for decoding using Verilog in Xilinx are provided below..

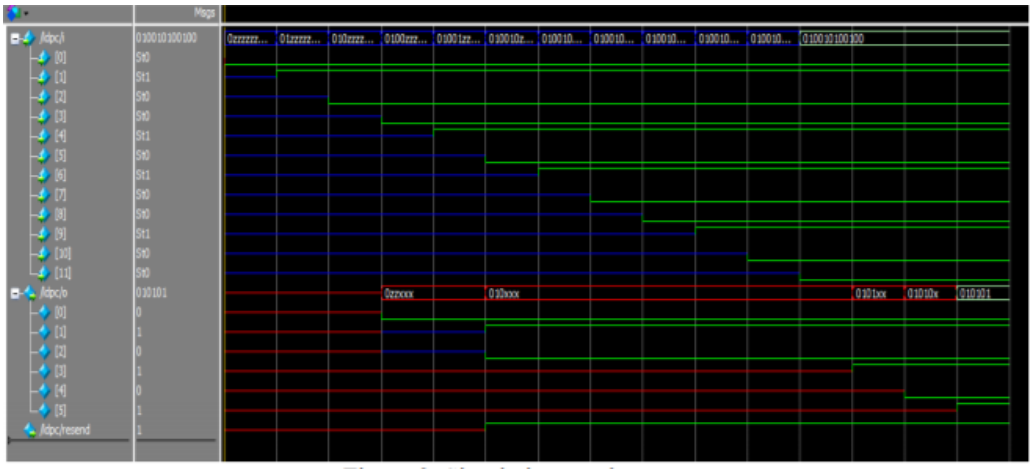


Fig 6.2 : simulation Result

b. Synthesis: The software mentor graphics Leonardo spectrum is used for synthesis. And Virtex IV technology is employed. The outcome of synthesis is shown in things like chip area, latency, etc. The number of lookup tables indicates the size of the chip area (LUTs). Propagation delay is caused by the delay. The results of the code synthesis used to create the LDPC decoder are displayed in the table..

Pass	Area (LUTs)	Delay (in ns)	DFF	PIs	POs
1	15	1	0	12	7

Conclusion and Future Scope

a. Conclusions: In this study, we thoroughly examined the LDPC code and investigated several encoding and decoding techniques. We discovered that the LDPC code is a good block error correction code. When we constructed this LDPC coded BPSK system in Matlab and then this was implemented in FPGA for testing, its error performance was found to be close to Shannon's limit, and the decoder Modified Sum-ProductAlgorithm was shown to be efficient for decoding as well. Below is a summary of this project's completion..

1. LDPC Code's error correcting performance is near to Shanon's error correctionlimit.
2. LDPC Code's performance depends on various characteristics of the parity matrix such as matrix dimension, girth of the matrix,regularityetc.
3. Cyclic codes are easy to implement due to its parallel structurenature.
4. Sum-product Algorithm (SPA) have good error performance in logdomain as compared to probabilitydomain.
5. Performance of MSPA (Modified SPA) is also near to SPA performance and easy to implement inFPGA.
6. Decoder behavior of various decoding algorithm was studied including Min-Sum and Message passing and their behavior was found near toShanons.
7. The FPGA implementation of Encoder and decoder was studied on Spartan 3E respectively and found device utilization permissible i.e.synthesizable.

b. Future Scope:

LDPC: Code is discovered to be an effective error-correcting code, and its performance close to Shanon's limit makes it a strong contender for a variety of modulation schemes, including OFDM, DVB-S2, and 802.3 a (Wi-Max). Studying the implementation of the LDPC-based OFDM system is a future objective. Future communication methods that require a lot of bandwidth, like 4G and 5G, and which require error-correcting codes are candidates for OFDM. Since OFDM has some issues, including PAPR and ISI, we can use LDPC to try to solve them before moving on to their implementation. When combined with other Block error correcting codes, such as RS codes, LDPC can sometimes be employed to achieve higher performance. For upcoming developments, we'll also take RS-codes and LDPC codes into account. Finally, we will attempt to reduce implementation complexity by either altering the decoder method or enhancing the decoder architecture.

References:

[1] M. Karkooti and J. R. Cavallaro, "Semi-parallel reconfigurable architectures for real-time LDPC decoding," in Proc. of Int. Conf. on Inf. Technology: Coding and Computing (ITCC), vol. 1, 2004, pp. 579–585.

[2] X. Chen, J. Kang, S. Lin, and V. Akella, "Memory system optimization for FPGA-based implementation of quasi-cyclic LDPC codes decoders," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 58, no. 1, pp. 98–111,2011.

[3] V. A. Chandrasetty and S. M. Aziz, "Resource efficient LDPC decoders for multimedia communication," INTEGRATION, the VLSI journal, vol. 48, pp. 213–220,2015.

[4] K. Zhang, X. Huang, and Z. Wang, "High-throughput layered decoder implementation for quasi-cyclic LDPC codes," IEEE Journal on Selected Areas in Communications, vol. 27, no. 6, pp. 985–994,2009.

[5] X. Peng, Z. Chen, X. Zhao, D. Zhou, and S. Goto, "A 115mW 1Gbps QC-LDPC decoder ASIC for WiMAX in 65nm CMOS," in IEEE Asian Solid State Circuits Conference (A- SSCC), 2011, pp.317–320.

[6] B. Xiang, D. Bao, S. Huang, and X. Zeng, "An 847–955 Mb/s 342–397 mW dual-path fully-overlapped QC-LDPC decoder for WiMAX system in 0.13 µm CMOS," IEEE Journal of Solid-State Circuits, vol. 46, no. 6, pp. 1416–1432,2011.

[7] E. Boutillon and G. Masera, Channel coding: Theory, algorithms, and applications. Elsevier, 2014, ch. Hardware Design and Realization for Iteratively Decodable Codes, pp. 583–642.

[8] S. K. Planjery, S. K. Chilappagari, B. Vasic, D. Declercq, and L. Dan- ´ jean, "Iterative decoding beyond belief propagation," in IEEE Information Theory and Applications Workshop (ITA), 2010, pp.1–10.

[9] S. K. Planjery, D. Declercq, L. Danjean, and B. Vasic, "Finite alphabet iterative decoders for LDPC codes surpassing floating-point iterative decoders," IET Electronics Letters, vol. 47, no. 16, pp. 919–921,2011.

[10] "Finite alphabet iterative decoders – part I: Decoding beyond belief propagation on the binary symmetric channel," IEEE Transactions on Communications, vol. 61, no. 10, pp. 4033–4045, 2013.

[11] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X. Hu, "Reduced-complexity decoding of LDPC codes," IEEE Trans. on Communications, vol. 53, no. 8, pp. 1288– 1299,2005

[12] V. Savin, Channel coding: Theory, algorithms, and applications. Elsevier, 2014, ch. LDPC Decoders, pp.211–260.