B Srisailam* et al.
(IJITR) INTERNATIONAL JOURNAL OF INNOVATIVE TECHNOLOGY AND RESEARCH
Volume No.7, Issue No.6, October-November 2019, 9471-9474.

Efficient Hardware Implementation of Probabilistic Gradient Descent Bit Flipping

B SRISAILAM
M. Tech student, Dept of ECE, Siddhartha Institute of Engineering And Technology, Hyderabad, TS, India.

T NAGARAJU
Associate Professor, Dept of ECE, Siddhartha Institute of Engineering and Technology, Hyderabad, TS, India.

Abstract: **This paper presents a new Bit Flipping (BF) decoder, called Probabilistic Parallel Bit Flipping (PPBF) for Low-Density Parity-Check (LDPC) codes on the Binary Symmetric Channel. In PPBF, the flipping operation is preceded with a probabilistic behavior which is shown to improve significantly the error correction performance. The advantage of PPBF comes from the fact that, no global computation is required during the decoding process and from that, all the computations can be executed in the local computing units and in-parallel. PPBF provides a considerable improvement of the decoding frequency and complexity, compared to other known BF decoders, while obtaining a significant gain in error correction. One improved version of PPBF, called non-syndrome PPBF (NS-PPBF) is also introduced, in which the global syndrome check is moved out of the critical path and a new terminating mechanism is proposed. In order to show the superiority of the new decoders in terms of hardware efficiency and decoding throughput, the corresponding hardware architectures are presented in the second part of the paper. The ASIC synthesis results confirm that, the decoding frequency of the proposed decoders is significantly improved, much higher than the BF decoders of literature while requiring lower complexity to be efficiently implemented.**

Index Terms—**Low-Density Parity-Check; Probabilistic Bit Flipping Decoding; High Decoding Throughput; Low-Complexity Implementation;**

I. INTRODUCTION

Low-Density Parity-Check (LDPC) codes were introduced by Gallager in 1963 and they have been adopted as a part of several standards, such as IEEE 802.11n, 802.16a, etc. due to their outstanding error correction capability [1][2]. LDPC codes can be decoded by two classes of decoding algorithms: soft-information message passing algorithms, e.g. Min Sum (MS), Sum Product (SP) [3], or hard-decision algorithms, e.g. Bit Flipping (BF), Gallager-A,B [4]. The soft-information decoding algorithms provide a very good decoding performance, being close to the capacity, but require an intensive computation. They exhibit, therefore, very high complexity in hardware realization [3]. On the contrary, the hardware implementations of hard-decision decoders were shown to be very low complexity thanks to the simple computation units and smaller connection networks, but they are weak in error correction. The increasing demand of massive data rates in several applications, such as optical communications, high-speed Ethernet [5], data storage devices [6][7][8] or the New Radio of 5G enhanced mobile broadband (eMBB) [9], will require higher decoding throughput. In such applications, hard decision decoders become the promising candidates thanks to their simple computations and hence, low complexity and high decoding throughput, given that, the decoding performance is improved. This paper focuses on the Bit Flipping (BF) hard decision decoder to not only enhancing the decoding throughput and reducing the decoder complexity but also improving the error correction performance. Furthermore, we focus on the decoders which require only the hard information from the channel. These decoders could be used when the channel soft-information is unable to obtain or requires a long latency to be generated, such as the storage system [10][11].

2. LITERATURE SURVEY

The BF decoding concept is firstly introduced by Gallager [4], in which the binary information is iteratively passed between two groups of nodes: the Variable Nodes (VNs) and the Check Nodes (CNs), and it uses the channel hard information as the input values. The CN operation is the exclusive-OR (XOR) operations and at each iteration, a VN is flipped if the number of unsatisfied neighboring CNs is higher than a predefined threshold. This BF is a very low complexity decoder but providing a very weak error correction, compared to the soft-decision decoders. Several BF decoders have been latter proposed in literature, in which Gradient Descent Bit Flipping (GDBF) and Probabilistic Gradient Descent Bit Flipping (PGDBF), introduced by Rasheed et al. in [12], could be seen as the most promising algorithms, in term of error correction. In GDBF, the CN operation is the XOR calculation as in the standard BF decoder, while VN computes a function called inversion or energy function derived from a gradient descent formulation. A VN is flipped when its energy value is a maximum, compared to all other energies.

GDBF provides a very good error correction and it is better than the prior-introduced deterministic BF decoders. PGDBF is a variant of GDBF, introduced also in [12]. PGDBF follows precisely the decoding steps of GDBF and differs only from the VN flipping operations. All the flipping candidates in GDBF are only flipped with a probability of p (p < 1) in PGDBF. This probabilistic behavior interestingly improves the decoding performance, far better than the original GDBF and very close to MS [13]. The evolution of BF performance improvement is shown in Fig. 1 on the well-known Tanner code [14]. Another newly proposed BF decoder is introduced by J. Jung in 2017 [10], called Multi-Bit Flipping (MBF). The MBF scenario is in line with our target in which the decoder requires only hard information from the channel for decoding. The VNs of MBF also compute at each iteration the energy value basing on its neighbor CN values. The novelty comes from the fact that, at each iteration of MBF, only a constant number of VNs are allowed to flip to avoid the overcorrection. For example, only 4 VN are flipped at each iteration. This truly helps the MBF avoid VN miss-flipping and offers a considerable improvement in error correction.

3. EXISTING METHOD:

The first proposed hardware architecture to implement PPBF decoder. Similarly to the architecture of PGDBF decoder, the CN operations are realized by the dc-input XOR gates. The VN energy value is computed by the summation block whose inputs are the CN computation results and the XOR1 (computing the term v (k) n ⊕ yn). The key feature of this architecture is that, the computed energy values will control the threshold input of a random generator (RG). This threshold input is used to control the probability of the generated bit and by doing that, the probability of the generated bit is adapted for each VN and each iteration. One RG method could be used is the Linear Feedback Shift Register (LFSR) as in [18]. The VN value for the next iteration, v (k+1) n , is computed by the XOR2 gate basing on the generated bit and the VN current value v (k) n . The v (k+1) n is an inversion of v (k) n if the generated bit R (k) n = 1 since X ⊕ 1 = NOT(X), ∀X. When the generated bit R (k) n = 0, v (k+1) n = v (k) n since X ⊕ 0 = X, ∀X. The SC module is implemented to stop the decoder when the correct codeword is found. This architecture can fully execute the operations of the PPBF decoder and can also be adapted for the NS-PPBF. However, the results reported in [18] revealed that, this method may be costly since each VN requires an LFSR module and the decoder complexity may be increased due to these RGs, especially when the LDPC code with large N is used. We leave aside this architecture and focus on a low-complexity solution to generate the binary random signals, introduced in [13], called Cyclic Shift Truncated Sequence (CSTS).

4. PROPOSED PGDBF ARCHITECTURE:

The global architecture of the PGDBF decoder is shown in Fig. 1. The organization of the blocks and the data flow follow precisely the organization of the GDBF decoder, with the difference being in the additional block which produces the random signals R(k) .

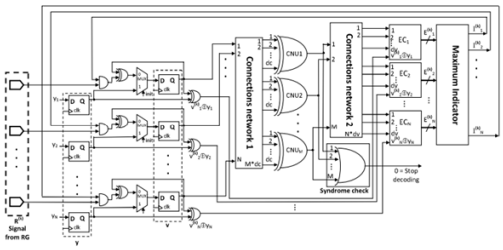

Fig. 1. The global architecture of the PGDBF

The PGDBF follow precisely the data flow of GDBF with difference coming from the random generator and the AND-gates Let us first briefly present how the decoder described in Algorithm 1 operates on this hardware architecture. Since we focus in this paper on the BSC channel, two register arrays represented as two sequences of D-Flip Flops (D-FFs) are required to store the noisy codeword y and the estimated codeword at the current iteration v(k) . At the initialization of the decoder, the signal init triggers the copy of y into v(0) . Then, the CNUs compute the parity of their neighboring bits in v(k) , after properly driven by a first connection network. The second connection network drives the CN values to the energy computation blocks, for each VN. The maximum indicator module is composed of a maximum finder component and comparators which outputs I (k) n = 1 whenever the corresponding energy is equal to the maximum, and I (k) n = 0 otherwise. Indicator values I (k) n are propagated to the AND gates, and combined with the RS sequence. This series of AND gates highlights the difference between PGDBF and GDBF. In the GDBF decoder, all bits with I (k) n = 1 are flipped, while in the PGDBF algorithm, only the bits with I (k) n = 1 and R(k) n = 1 are flipped. New values of the bits stored in v(k) are used for the next decoding iteration.

At each iteration, the syndrome check module performs an OR operation on the CNs values to verify whether the intermediate sequence v(k) is a codeword, in which case the decoding process is halted. Another instance when the decoding halted is when no codeword v(k) has been found, and a predetermined maximum number of iterations Itmax has been reached, in which case a decoding failure is declared. Note that all components in the

decoder architecture are combinational circuits except the registers v(k) and y. Therefore, new values of the bits in v(k) are updated after each clock cycle. In order to optimize the proposed architecture, we focus on the following two important issues. First, as identified in [16], the hardware overhead induced by the RS is not negligible with a naive implementation, and we propose in Section III-B different low complexity methods to generate the sequences R(k) . Second, we optimize the architecture of the maximum indicator module in order to maximize the decoding throughput

the performance gain in the PGDBF algorithm comes from the introduction of the random sequence R(k) which makes a perturbation in bit flips. In our theoretical description of the PGDBF, for each and every codeword, the sequences at different iterations (R(0) ,..., R(I tmax)) are independent and identically distributed. However, a direct and naive generation of the sequences R(k) with linear feedback shift register (LFSR) random generators is costly, and requires many times more registers than the non-probabilistic GDBF [16]. We propose in this section an approach to reduce the hardware overhead required to generate the RS sequences.

5. SIMULATION RESULTS

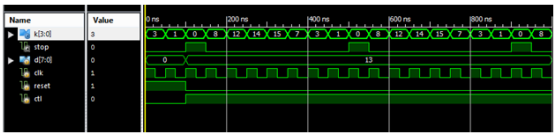

Fig 2. PGDBF output

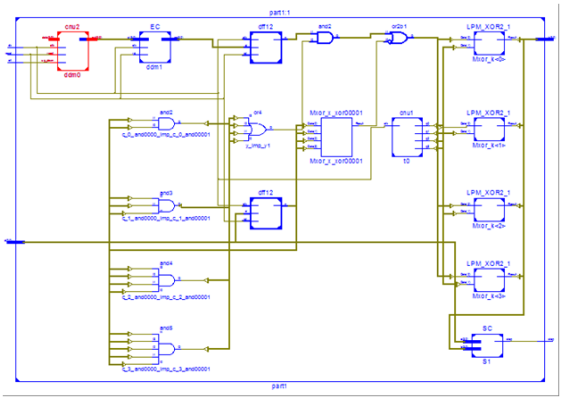

Fig 3.RTL schematic

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	14	960	1%
Number of Slice Flip Flops	20	1920	1%
Number of 4 input LUTs	16	1920	0%
Number of bonded IOBs	16	108	14%
Number of GCLKs	1	24	4%

Fig 4. Design summary

```
Timing constraint: Default path analysis
  Total number of paths / destination ports: 4 / 1
--------------------------------------------------------------------------------
Delay:               7.535ns (Levels of Logic = 4)
  Source:            d<4> (PAD)
  Destination:       stop (PAD)

  Data Path: d<4> to stop
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   1.218   0.622  d_4_IBUF (d_4_IBUF)
     LUT4:I0->O            1   0.704   0.595  stop426 (stop426)
     LUT2:I0->O            1   0.704   0.420  stop454 (stop_OBUF)
     OBUF:I->O                 3.272          stop_OBUF (stop)
    ----------------------------------------
    Total                      7.535ns (5.898ns logic, 1.637ns route)
                               (78.3% logic, 21.7% route)
```

Fig 5. Time delay

6. CONCLUSION

In this paper, we propose a new high-throughput, lowcomplexity Bit Flipping decoder for Low-Density ParityCheck (LDPC) codes on Binary Symmetric Channel (BSC), called Probabilistic Parallel Bit Flipping (PPBF). The PPBF is more advantageous than other state-of-the-art BF decoders such as the Gradient Descent Bit Flipping and Probabilistic Gradient Descent Bit Flipping decoders, in terms of operating frequency and decoder complexity, thanks to the elimination of the global operation which helps shorten the critical path and reduce the decoder hardware cost. Furthermore, in PPBF, the probabilistic flipping of the Variable Node is introduced for all levels of its energy value rather than only on the maximum values as in PGDBF.

REFERENCES

[1] "Wireless lan medium access control and physical layer specications: Enhancements for higher throughput, p802.11n/d3.07," IEEE-802.11n, Mar. 2008.

[2] "Local and metropolitan area networks - part 16, ieee std 802.16a-2003," IEEE-802.16.

[3] D. Declercq, M. Fossorier, and E. Biglieri, "Channel coding: Theory, algorithms, and applications," Academic Press Library in Mobile and Wireless Communications, Elsevier, ISBN: 978-0-12-396499-1, 2014.

[4] R. G. Gallager, "Low density parity check codes," MIT Press, Cambridge, 1963, Research Monograph series, 1963.

[5] "Ieee standard for information technology - corrigendum 2: Ieee std 802.3an-2006 10gbase-t correction," IEEE Std 802.3-2005/Cor 2-2007 (Corrigendum to IEEE Std 802.3-2005), Aug. 2007.

[6] S. Jeon and B. V. K. V. Kumar, "Performance and complexity of 32 k-bit binary ldpc codes for magnetic recording channels," IEEE Transactions on Magnetics, vol. 46, no. 6, pp. 2244–2247, June 2010.

[7] K. C. Ho, C. L. Chen, Y. C. Liao, H. C. Chang, and C. Y. Lee, "A 3.46 gb/s (9141,8224) ldpc-based ecc scheme and on-line channel estimation for solid-state drive applications," in 2015 IEEE International Symposium on Circuits and Systems (ISCAS), May 2015, pp. 1450–1453.

[8] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision errorcorrection codes in nand flash memory," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 58, no. 2, pp. 429–439, Feb 2011.

[9] G. document, "Ldpc codes for the enhanced mobile broadband (embb) data channel in the 3gpp 5g new radio," [Online]. Available: http://www.3gpp.org/ftp/tsg ran/WG1 RL1/TSGR1 88/Report/.

[10] J. Jung and I. C. Park, "Multi-bit flipping decoding of ldpc codes for nand storage systems," IEEE Communications Letters, vol. PP, no. 99, pp. 1–1, 2017.

