International Journal of New Technology and Research (IJNTR)
ISSN: 2454-4116, Volume-1, Issue-2, June 2015 Pages 13-15

Non Binary Low Density Parity Check Codes Decoding Over Galois Field

Yogita Ahuja, Ramesh Bharti

Abstract—. **Conventional LDPC codes have a low decoding complexity but may have high encoding complexity. The encoding complexity is typically of the order O(n2)[5]. Also high storage space may be required to explicitly store the generator matrix. For long block lengths the storage space required would be huge. The above factors make the implementation of the Conventional LDPC codes less attractive.**

These codes are usually decoded using the sum-product algorithm, which is a message passing algorithm working on the Tanner graph of the code[5]. The sparseness of the parity check matrix is essential for attaining good performance with sum-product decoding. The time complexity of the sum-product algorithm is linear in code length. This property makes it possible to implement a practical decoder for long lengths.

***Index Terms*—LDPC, Cyclic Codes.**

I. INTRODUCTION

Linear codes use a generator matrix G to map a message vector *X* of length *k* to a transmitted codeword *Y* of length *n*. All codeword satisfy H*Y*=0, where H is the parity check matrix. Gallager defined (*n*, *p*, *q*) LDPC codes to have a block length *n* and a parity check matrix with exactly *p* ones per column and *q* ones per row, where *p* >=3. The rate of the code is *k/n = 1*. Gallager proved that, for a fixed *p*, the error probability of the optimum decoder decreases exponentially for sufficiently low noise and sufficiently long block length. The parity check matrix is typically constructed randomly while constraining the distributions of the row and column vectors as uniform as possible. Since H is not in systematic form, we perform Gaussian elimination using row operations and reordering of columns.

II. DECODER ARCHITECTURES

The different types of Decoder architecture implementation are Parallel, Serial and Semi- Parallel and are given below[5]

A. Parallel Decoder Architecture:

A fully parallel implementation of the decoder is shown in figure 2.1. The parallel implementation consists in mapping directly the symbol and check nodes in the Tanner graph to the respective symbol and check modules. The edges of the graph become physical buses of width equal to chosen precision. A fully parallel implementation, while efficient in speed point of view is demanding in terms of area, due to interconnect between the processing elements. Although the computations for calculating the check to symbol and symbol to check messages are not particularly complex and require a small area to be implemented, the massive number of interconnections in the graph lead to complex wiring. In fully parallel architecture the number and complexity of interconnects results in the implementation where almost 60% of the area is being dominated by wires. Moreover, the number of computational blocks required is in one to one relationship with the number of nodes in the Tanner graph. For medium or long code the resource demands and complexity of hardware implementation will become infeasible.

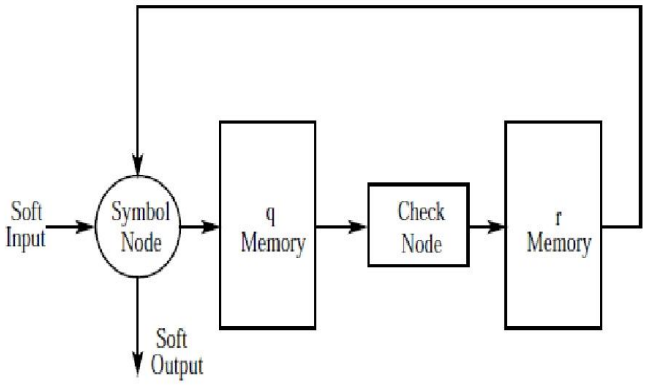

Figure 1: Parallel Decoder Architecture

B. Serial Decoder Architecture:

A serial implementation dramatically reduces the complexity of the interconnect and the total area of the design. The architecture has only one symbol module and one check module so it updates only one message at a time. Even supposing that all computations in the node modules can be executed in one clock cycle, m clocks are needed before the updating *r* phase is completed, and n clocks before the updating *q* phase is done. If codes with randomly constructed **H** matrix are considered, all symbol nodes must be updated and

Yogita Ahuja , M.Tech Scholar, JNIT University Jaipur Rajsthan
Ramesh Bharti, Associate Professor JNIT University Jaipur Rajsthan.

the messages stored in the memory before the update of the check node starts. Therefore minimum (m+n) clock is needed for each iteration of the code. This makes the decoding process very slow reducing the total throughput. A possible way to improve the serial architecture is to have as many symbol/check nodes as the number of max iterations, pipelining the updating operations, hence reducing the average time required to decode a word. The serial architecture also requires large memories to store all the messages. The addressing of the memories is another problem typical of this type of implementation. This is due to the fact that the messages must be read/written from/to the memory in the proper order to assure that the nodes that do the computation receive the proper messages. This requires a random access to the memory and a complex control unit that generates the correct addresses.

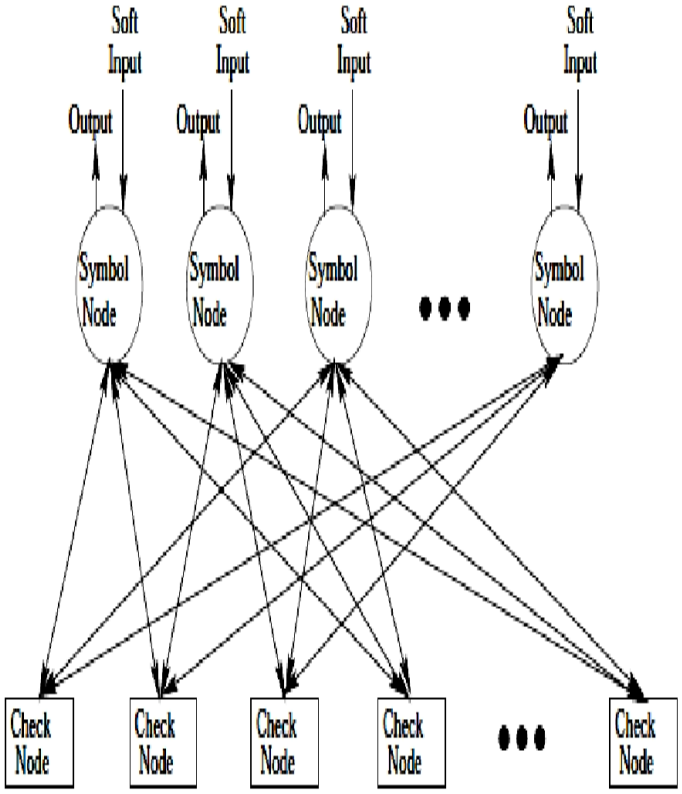

Figure 2: Serial Decoder Architecture

III. ARCHITECTURE OF NON-BINARY LDPC DECODER

Figure 3 presents the architecture of the LMMA-based non-binary LDPC decoder, which consists of two parts; namely, memories and processors. The processors can be divided into one variable node unit (VNU) corresponding to one check node unit (CNU) corresponding to and one early termination unit (ETU).

There are four types of memories used in implementation: memory for R_{cv} with size of $\gamma \; n \; q \; W_R$ stores the information from check nodes to variable nodes, memory for L_v with size of $n \; q \; W_L$ stores the initial log-likelihood ratios, memory for $\hat{\mathbf{c}}$ with size $n \; \log_2 q$ stores the decoded bits, and memories inside each CNU store the intermediate values. In discussion above, γ denotes the column weight, n is the codeword length, q is the size of Galois field, W_R and W_L represent the word-lengths for R_{cv} and L_v .

As shown in Fig. 3(b), it is obvious that CNU is the most complex part of the decoding permutator block shifts the incoming message vector cyclically. The first FIFO is used to perform the parallel–to-serial conversion as required in min-max processor. In Fig. 3(c), the min-max processor consists of one forward recursion block, one backward recursion block, two memories storing intermediate values, and one merge block. Then FIFO block is used again to perform serial-to-parallel conversion, followed by the permutator block. In order to reduce the latency of min-max processor, we adopted the bidirectional recursion technique. In conventional BCJR processor, it takes ρ cycles updating forward metric and backward metric recursively and additional ρ cycles to combine them. However, the combining process can be proceeding once half of the recursion is done, which saves up to ρ cycles. Because of high complexity of CNU design and high memory requirements of non-binary decoder than that of binary decoder, reduced-complexity architectures and selective version of MMA have been widely studied [22,23]

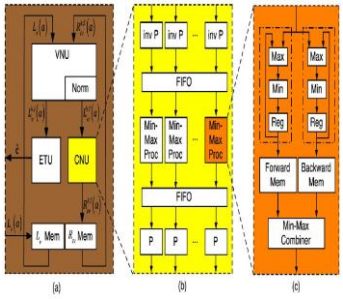

Fig. 3 Non-binary LDPC decoder architecture: (a) overall architecture, (b) architecture of CNU, and (c) architecture of BCJR-based min-max processor.

At last, we present in Table 1 the utilization results for hardware implementation of the QC (3,15)-regular, girth-8 (16935, 13550, 0.8) LDPC code over GF(2) and GF(4). In order to make fair comparison, we adopt the 6 bits precision (including the sign bit) for both decoders, the maximum number of iterations is set to 15, 15 variable node units and one check node processor are employed. One can clearly notice that LMMA consumes 3.6 times larger memory than layered attenuated min sum algorithm (LAMSA) because of large field size, while occupied number of slices for LMMA is five times of that of LAMSA because of higher complexity

involved in CNU. On the other hand, Max-Log algorithm has larger logic usage than Min-Max algorithm as they are both based on the BCJR algorithm while the min operation is replaced by addition

Table 1 Utilization Summary of LDPC Decoders and Reed-Solomon Decoder.

Resources	LAMS A	Log-FFT	Max-Log	Min-Max	RS
Occupied Slices	3,086 (4%)	12,020 (16%)	18,070 (24%)	13,832 (18%)	12,336 (6%)
RAMB36E1	38 (3%)	38(3%)	38(3%)	38(3%)	38 (3%)
RAMB18E1	89 (4%)	782(32%)	512(24%)	512(24%)	84 (3%)

IV. CONCLUSION AND FUTURE WORK

We proposed a novel SD-FEC employing the concatenation of a girth-10 non-binary QC-LDPC code and a RS code with overall 27% OH for high-speed optical transmission systems. The BER performance was verified through FPGA emulation system. Superior waterfall and error floor performance is demonstrated at a post-FEC BER of 10−15. No error floor has been found and 5.05dB in Q-limit is achieved, corresponding to NCG of 11.91dB. To the best of our knowledge, this is the first implementation of concatenated non-binary LDPC code + shortened- RS code. We believe that the proposed non-binary QC-LDPC code is one of the promising candidates for the next generation optical communication systems.

The performance analysis and hardware implementation of the Non Binary LDPC codes have been done and it has been concluded that the codes perform better for medium codes as compared to long codes and is thus useful for short and medium packet transmission. If we increase the packet size the number of check node and bit node increases, again the decoding complexity increases and large number of hardware as well as memory is required. The advantage of using non-binary LDPC codes over Galois field is that the equivalent binary weight of parity check matrix is increased, while the number of short cycles may remain low.

It can also outperform Reed Solomon codes even for burst error channels. The Hardware implementation of the decoding algorithm will be performed on the parity check matrix [H]. The future work can be hardware implementation of Non Binary LDPC codes using FFT method devised by Bernault, Declercq and Fossorier which reduces the number of operations and hence decoding fast.

REFERENCES

[1] R.G. Gallagher, "Low-density parity-check codes", *MIT Press*, Cambridge, MA 1963.
[2] David J.C Mackay, "Good Error- Correcting codes Based on Very Sparse Matrices", *IEEE Transactions On Information Theory*, Vol. 45, No. 2, March 1999.
[3] Amir Bennatan and David Burshtein, "Design and Analysis of Nonbinary LDPC Codes for Arbitrary Discrete-Memoryless Channels", *IEEE Transactions on Information Theory*, Vol. 52, No. 2, February 2006.
[4] M. Davey and D. MacKay, "Low-density parity check codes over GF (q)" *IEEE Communications Letters*, vol. 2, 6, pp. 165 .167, June 1998.
[5] "Aspects of LDPC codes for Hardware Implementation", *PhD Thesis by Christian Spagnol*, 26th January 2009.
[6] Christian Spagnol, Emanuel Mihai Popovici, William Peter Marnane, "Hardware Implementation of GF(2^m) LDPC Decoders", *IEEE Transactions on Circuits and Systems—I*: Regular Papers, Vol. 56, No. 12, December 2009.
[7] PhD thesis of Deepak Girla on PA code for non binary LDPC codes, May 2003.
[8] "Non Binary LDPC Decoding and its Implementation", Jie Huang, November 5 2008.
[9] Zhongfeng Wang, Zhiqiang Cui, and Jin Sha ,"VLSI Design for Low-Density Parity- Check Code Decoding" , *IEEE Circuits and Systems Magazine*, 18th February 2011.
[10] L.Bernault, D.Declercq ,"Fast Decoding Algorithm for LDPC codes over GF(2^q)", *ITW2003*, Paris, France, March 31 - April 4, 2003.
[11] Christian Spagnol, William Marnane ,"A Class of Quasi-Cyclic LDPC codes over GF(2^m)" , *Transaction On Communications*, January 2000.
[12] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379-423, 1948.
[13] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE Convention Record*, vol. 4, pp. 142-163, 1959.
[14] R. W. Hamming, "Error detecting and error correcting codes," *Bell System Technical Journal*, vol. 29, pp. 147-160, 1950.
[15] K. Yang and T. Helleseth, "On the minimum distance of array codes as LDPC codes," *IEEE Trans. Inform. Theory*, vol. 49, no. 12, pp. 3268-3271, 2003.