CODE DESIGN AND DECODER IMPLEMENTATION OF LOW DENSITY PARITY CHECK CODE

Mong-Kai Ku, Huan-Sheng Li, Yi-Hsing Chien

Department of Computer Science and Information Engineering
National Taiwan University, Taiwan
{mku,r92084,r92049}@csie.ntu.edu.tw

***Abstract*—Low-Density Parity-Check (LDPC) codes have been widely considered as error-correcting codes for next generation communication systems. A good LDPC decoder design requires both implementation friendly LDPC codes and efficient decoder architectures. The quality of LDPC code is crucial in determining the coding gain and implementation complexity of LDPC hardware decoders. This paper presented a genetic algorithm (GA) based LDPC code search algorithm with hardware considerations. Regular quasi-cyclic LDPC codes are used due to its friendliness to hardware implementation. Our hardware architecture design schedules pipeline LDPC decoding operation to boost the hardware utilization efficiency (HUE) of LDPC decoder. A LDPC decoder with a block size of 12288 is implemented in FPGA to validate our code and architecture design.[1]**

I. INTRODUCTION

Low Density Parity-Check (LDPC) code was invented by Gallager nearly 40 years ago [1]. It has received a lot of attention lately due to its excellent error-correcting capability and implementation friendly decoding algorithms [2]. With coding gain approaching Shannon limit and lower hardware complexity than Turbo codes, LDPC has been considered by a number of next generation communication standards. LDPC codes with block size 16200 to 64800 are chosen as the forward error correction code by the satellite TV standard DVB-S2. The upcoming wireless LAN standard 802.11n is also considering LDPC codes. By changing the block size and rate of LDPC code, it is easy for system designers to make trade-offs between error correcting performance and hardware complexity, making the LDPC code suitable for a wide range of applications. Parity check matrix construction is one of the most critical factors that determine LDPC coding gain performance and decoder hardware complexity. Efficient VLSI implementation of LDPC decoders requires close integration between LDPC code and hardware architecture designs. In this paper, first we presented our hardware-aware LDPC code search algorithm, followed by the high utilization LDPC decoder architecture, and we conclude this paper with results of our LDPC decoder implementation on FPGA.

[1] This work was partially supported by the National Science Council of Taiwan under Grant No. NSC 93-2215-E-002-033 and NSC 93-2752-E-002-008-PAE.

II. HARDWARE-AWARE GA-BASED LDPC CODE SEARCH

LDPC codes with short to medium block sizes shows favorable coding gain performance than the conventional convolutional code. The goal of our LDPC code search tool is to generate short-to-medium sized parity check matrix that is optimized for hardware implementation. High performance LDPC codes can be constructed by observing the properties of corresponding Tanner graphs. In the meantime, implementation friendly code designs need to minimize hardware cost and reduce decoding complexity. Based on these two goals we choose the regular quasi-cyclic LDPC code [3]. As shown in Fig 1, the regular quasi-cyclic LDPC code has parity check matrix that is composed of cyclically shifted identity matrices. Each cyclically shifted identity matrix is made from identity matrix with cyclically shift right certain steps according to its offset.

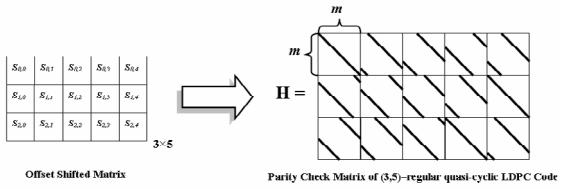

Fig. 1. Quasi-cyclic parity check matrix for LDPC code parity check matrix

The structure of quasi-cyclic LDPC code is very regular and block based such that the memory address generation in decoding can be simplified. Researches [4] and our own simulations have verified that quasi-cyclic LDPC codes can achieve excellent performance for short to moderate code length. We proposed a code search method based on Genetic Algorithm (GA) as code search engine. GA is a fast search and optimization algorithm

0-7803-9329-5/05/$20.00 ©2005 IEEE

that can have multiple screening criterions. Algebraic-based code construction method, code performance simulation and hardware architecture specific constraints are used as screening criterions in our GA search algorithm. By defining evaluation functions (constraints) for performance evaluation and crossover, mutation, selection strategies for GA operations, we can effectively search the LDPC code that meet our requirements. The codes found by our algorithm will possess the good properties of the algebraic-based code with coding gain performance verified in the same time.

Fig. 2 shows the simulation results of GA search codes, algebraically constructed codes and Neal's random generated codes. We compared (3,5)-regular LDPC code with code length 305, 1655 and 3005 respectively. The size of sub-matrix of quasi-cyclic codes is 61, 331, and 601 respectively. The parameters of GA operation are 50 generations, population size is 30, crossover rate is 0.8, and mutation rate is 0.2. Number of decoding iteration is 100. The performance of our GA search codes equals the algebraically constructed codes and randomly generated codes while satisfying constraints specific to our hardware architectures.

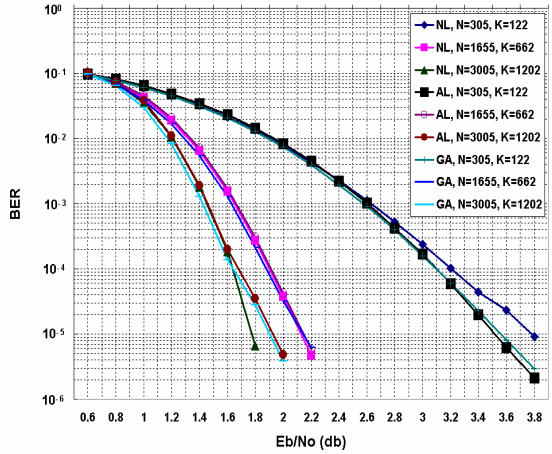

Fig. 2. Simulated results of Neal's random generated codes (NL) , algebraic constructed code (AL) and GA search codes (GA)

III. HARDWARE DECODER DESIGN

Iterative LDPC decoding algorithms such as sum-product or min-sum algorithms passes information back and forth between the check node and variable nodes in LDPC Tanner graph until correct codeword is found. Some of the decoder designs use modified decoding algorithm to reduce algorithm complexity and minimizing coding gain degradation [5]. Optimized decoding operation sequence can eliminate or reduce waiting time of data dependency [6]. We proposed a two-phase overlapped pipelined architecture shown in figure 3. The check node function unit (CNFU) and variable node function unit (VNFU) updates the check nodes and variable nodes respectively. The core of our architecture is the Jump-Reset scheduling algorithm that allows CNFU and VNFU to access the dual port memory without conflict. The memory schedule places constraint on the parity check matrix generation. The time offset value w for our overlapped pipelined LDPC decoder architecture is determined by the parity check matrix. The constraint $w<½m$ is used as evaluation function. The offset value need to be less than half of the size of the circulant sub-matrix for our decoder architecture to function correctly. Lower offset value w also reduces the latency of decoders.

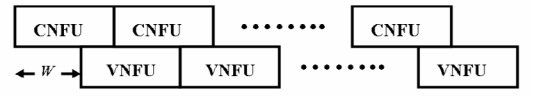

Fig. 3. Overlapped Pipelining of check node's and variable node's function units

The block diagram of the LDPC decoder is shown in Fig. 4. Our semi-parallel architecture has multiple CNFU and VNFU working at the same time to improve throughput. There is no bubbles in the scheduling results in very high hardware utilization efficiency.

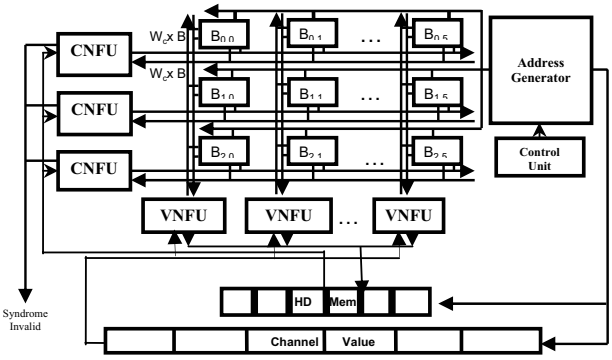

Fig. 4. Block Diagram of LDPC Decoder

We have implemented our LDPC decoder design on Altera Excalibur EPXA10 development board. It shows that Jump-reset scheduling algorithm can offer high hardware utilization with minimal BER performance loss. The results show that our architecture can support very large parity check matrices. The hardware cost and performance is summarized in the following table.

Code Rate	1/2
Code Length	12288 bits
BRAM	307200 bits
Logic Cell	3064 LCs
Clock Frequency	32.64 MHz
Throughput	19.11 Mbps

Table 1: LDPC Decoder Implementation Summary

IV. CONCLUSION

In this paper, we proposed a complete LDPC code and decoder design platform. A hardware-aware GA-based regular quasi-cyclic LDPC code search algorithm is presented. Our algorithm incorporates average girth, BER simulation and architecture specific constraints to find LDPC code suitable for efficient LDPC decoder implementation. The regular quasi-cyclic LDPC parity check matrix also simplify the hardware architecture, and make overlapping pipelining and multithread decoding easier to handle. Simulation shows that our LDPC code exhibits equal or better performance to randomly generated LDPC codes with short to medium block sizes. The overlapped pipelining architecture with Jump-Reset scheduling provides very high hardware utilization with efficient memory usage. The FPGA result shows promise for future ASIC implementation for the use in next generation communication systems.

REFERENCES

[1] R.G. Gallagher, "Low-density parity-check codes," IRE Trans. on Info. Theory, vol. IT-8, pp. 21-28, Jan. 1962.

[2] D.J.C. Mackay and R.M. Neal, "Near Shannon limit performance of low density parity check codes," IEE Electronics Letters, vol. 33, no.6, pp. 457-458

[3] Fossorier, M.P.C., "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," IEEE Transactions on Information Theory, vol. 50, no. 8, pp.1788 – 1793, Aug. 2004

[4] D. Sridhara, T. E. Fuja, and R. M. Tanner, "Low density parity check codes from permutation matrices," in Proc. Conf. Information Sciences and Systems, Baltimore, MD, pp. 142, Mar. 2001

[5] Marjan Karkooti and Joseph R. Cavallaro, "Semi-Parallel Reconfigurable Architectures for Real-Time LDPC Decoding," ITCC Proceeding of International Conference, Information Technology: Coding and Computing, vol.1, pp.579-585, April 2004

[6] Yanni Chen and Keshab K. Parhi, "Overlapped Message Passing for Quasi-Cyclic Low-Density Parity Check Codes", IEEE Trans., Circuits and Systems, vol. 51, pp.1106-1113, June 2004