Parallel Decoding Architectures for Low Density Parity Check Codes

C. Howland and A. Blanksby

High Speed Communications VLSI Research Department
Agere Systems
101 Crawfords Corner Road, Holmdel NJ 07733.
Email: {howland,blanksby}@agere.com

ABSTRACT

A parallel architecture for decoding Low Density Parity Check (LDPC) Codes is proposed that achieves high coding gain together with extremely low power dissipation, and high throughput. The feasibility of this architecture is demonstrated through the design and implementation of a 1024 bit, rate-1/2, soft decision parallel LDPC decoder.

1. INTRODUCTION

Error correcting codes are a critical part of modern communications systems where they are used to detect and correct errors introduced during transmission [1]. In many applications a substantial portion of the baseband signal processing is dedicated to the forward error correction (FEC) encoding and, particularly, decoding of data signals. As system designers can trade coding gain for lower transmit power, longer reach and/or higher data throughput there is an ongoing effort to incorporate increasingly more powerful coding techniques into communications systems.

Recently a new family of codes known as turbo codes have been developed that approach the Shannon limit, the theoretical maximum achievable coding gain for a given channel [2]. Turbo codes are based on concatenated, interleaved convolutional codes and provide excellent coding gain at low code rates[1]. A related family of codes are block turbo codes which are based on linear product block codes and yield very high coding gain at high code rates [3]. However, there exist significant implementation challenges for turbo codes and block turbo codes due to the iterative nature of their respective decoding algorithms. Both turbo and block turbo codes are decoded using sophisticated algorithms that iterate multiple times across the data block. Each pass through the data block requires the fetching, computation, and storage of large amounts of state information. Performing multiple iterations to achieve high coding gain necessarily reduces throughput and increases power dissipation. Furthermore the design time of turbo and block turbo decoders is impacted by the need to integrate multiple memories and develop and verify complex state machines for sequencing the decoding process.

In general parallel architectures for a given algorithm are attractive from an implementation perspective giving low power, high throughput, and simple control logic. Parallel architectures are even more favorable for iterative algorithms if the data converges and the switching activity decays. However, no such architectures can be found for turbo and block turbo codes due to the inherent sequential nature of their respective decoding algorithms. An alternative approach is to consider codes that can be iteratively decoded in a block parallel fashion. One such family of codes that has recently been rediscovered is Low Density Parity Check (LDPC) codes [4, 5].

LDPC codes are linear block codes with a sparse parity check matrix. Above a code rate dependent minimum block size, powerful LDPC codes and decoding algorithms can be found that either match or exceed the coding gain of turbo and block turbo codes. The most powerful code currently known is a 1 million bit, rate 1/2, LDPC code, achieving a capacity which is only 0.13dB from the Shannon limit for a bit error probability (BER) of 10^{-6} [6]. In this paper we propose a parallel decoder architecture for LDPC codes that achieves both very high throughput and extremely low power dissipation. This architecture is demonstrated through the design and implementation of a 1024 bit, rate-1/2, soft decision, LDPC decoder.

1. The rate of a code is defined as the ratio of the information bits to the sum of the information and parity bits. A low-rate code has a large redundancy overhead while a high-rate code has a small overhead.

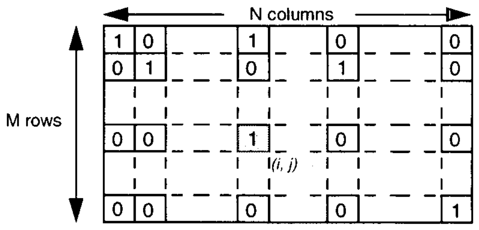

Figure 1. General structure of a low-density parity check matrix ***H***.

2. LOW DENSITY PARITY CHECK CODES

Low density parity check codes are linear block codes thus the set of all codewords, x, span the null space of a parity check matrix $\boldsymbol{H}$:

$$H \cdot x = 0 \qquad (1)$$

The parity check matrix $\boldsymbol{H}$ for LDPC codes is a sparse binary matrix where the set row and column elements are chosen to satisfy a desired row and column weight profile [6]. The set elements in the graph are further constrained by the requirement that the row and column overlap be minimized. The constraints on the matrix structure enable efficient decoding and produce a powerful code.

The general structure of $\boldsymbol{H}$ is illustrated in Fig. 1. Each row of $\boldsymbol{H}$ corresponds to a parity check and a set element (i,j) indicates that data symbol j participates in parity check i. In a block of N bits or symbols, there are M redundant parity symbols and the code rate R is given by:

$$R = (N-M)/N \qquad (2)$$

2.1 Graph Representation of LDPC Codes

Low density parity check codes can also be represented using bipartite graphs where one set of nodes represents the parity check constraints and the other set represents the data symbols or variables as illustrated in Fig. 2. A variable node v_j, corresponding to column j in $\boldsymbol{H}$, is connected to check node c_i, corresponding to row i in $\boldsymbol{H}$, if the entry (i,j) in $\boldsymbol{H}$ is set, i.e. non zero. When considering the graph form of the parity check matrix, minimizing the row and column overlap of $\boldsymbol{H}$ is equivalent to maximizing the length of cycles in the graph.

0-7803-6685-9/01/$10.00©2001 IEEE

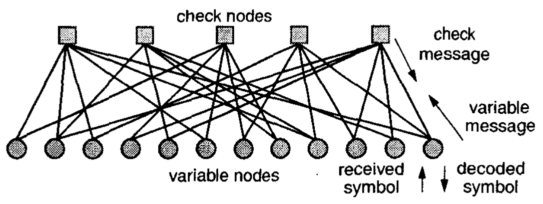

Figure 2. Example of the bipartite graph representation for a LDPC code and information flow in the message passing algorithm.

2.2 The Message Passing Algorithm

The message passing algorithm is an iterative algorithm for decoding LDPC codes best understood with reference to the graph representation of an LDPC code [5]. It can be summarized as follows:

1. Initialize all variable nodes and outgoing messages to the value of the corresponding received bit.
2. Propagate messages from the variable nodes to the check nodes along the edges of the graph.
3. Perform a parity check (XOR) on the incoming messages at the check nodes. Assign to each connected edge of a check node the XOR of the incoming message and the parity check result. This is the value all other connected variables imply the variable corresponding to each edge should take.
4. Pass messages from the check nodes back to the variable nodes along the edges of the graph.
5. At the variable nodes update estimates of the decoded bit and outgoing messages for each edge connected to the variable using a weighted majority function or summation.
6. Repeat steps 2-5 until a termination condition is met. Possible iteration termination conditions include:
   - The estimated decoded block x satisfies (1).
   - The current messages passed to the parity check nodes satisfy all of the parity checks. This does not guarantee that (1) is satisfied but is almost sufficient and is simple to test.
   - Stop decoding after a fixed number of iterations.

2.3 Hard Decision Decoding

Hard decision decoding involves the propagation of only one bit per message or graph edge, corresponding to the current estimate of the parity for that variable or message. Check nodes need only perform the simple parity check and message update with an XOR network. Variable nodes perform a weighted majority function or addition of the input messages and the received bit. The sign of the majority or summation determines the current iterations estimate of the decoded bit. Outgoing messages from the variable nodes are the sign of the majority or summation of all inputs except the current edges input. This corresponds to the implied value for that variable message by all other connected checks and the received value.

2.4 Soft Decision Decoding

Soft decision decoding requires the propagation of reliability information in addition to the parity information. This extra information requires additional bits to be propagated for each message or edge in the graph. Check nodes perform a reliability update along with the parity update to combine the reliability of the other incoming messages and produce an estimated reliability for each outgoing message. These reliabilities are generally the log likelihoods for each message and the update involves the hyperbolic tangent function similar to MAP decoding introduced by Bahl et. al [8]. The update of the decoded value and message values associated with each variable node is almost unchanged from a hard decision decoder except that the reliabilities of the received bit and messages act as scaling values for their associated weighting in a majority function or adder.

2.5 Graph Cycles

While the graph remains cycle free all information in updating the decoded bits or messages is unrelated having originated from distinct received bits. Under these conditions the decoding algorithm is optimal [9]. However, once a cycle is encountered the algorithm becomes suboptimal. Hence, the performance of the decoder is improved by maximizing the cycle lengths during the design of the parity check matrix $\boldsymbol{H}$ subject to the constraints of the block size and required column/row weight profiles.

3.ARCHITECTURES FOR LDPC DECODERS

The main challenge when implementing the message passing algorithm for decoding LDPC codes is managing the passing of the messages. As the functionality of both the check and variable nodes is very simple their respective realizations are straightforward and involve only a small number of gates. Implementing the message passing between the nodes results in very different challenges depending on whether a hardware sharing or parallel decoder architecture is pursued.

3.1 Hardware Sharing Decoder Architecture

A hardware sharing architecture consists of a small number of units implementing either the check or variable node functionality and a memory fabric to store the messages and realize the graph connectivity. This approach is illustrated in Fig. 3. While the hardware sharing architecture minimizes area the maximum throughput is limited by the need for the functional units to be re-used and the memory fabric accessed multiple times to perform each decoder iteration. Furthermore, it can be expected that the power dissipation of this architecture would be relatively high due to the significant memory bandwidth needed. Another major issue is the complexity of the control logic required for the representation of the graph connectivity and the corresponding address generation needed for fetching and storing the messages.

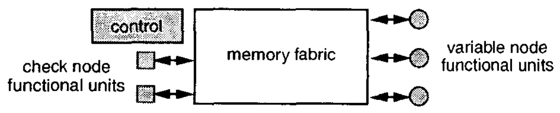

Figure 3. Hardware sharing LDPC decoder architecture.

3.2 A Parallel Decoder Architecture

The message passing algorithm maps extremely well to a parallel decoder architecture in which the graph, see Fig. 2, is directly instantiated in hardware. The graph representation of an LDPC code shows that the computational dependencies for any node depend only on nodes of the opposing type. This allows all variable nodes or check nodes to be updated in a block-parallel manner enabling very high throughput. This is in stark contrast to the block-serial dependencies inherent in turbo decoding. Furthermore, as illustrated in Fig. 4, it can be shown that as the message passing algorithm iterates the

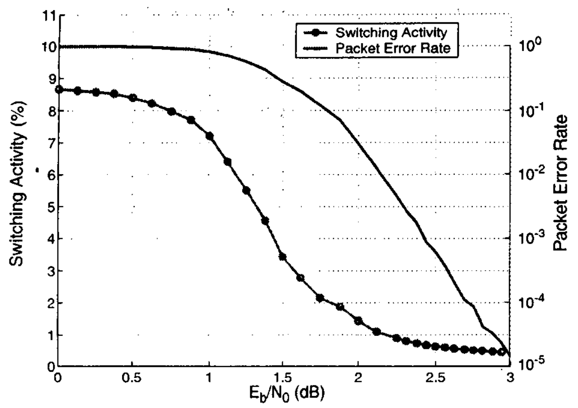

Figure 4. Switching activity of message bits in a 1024-bit, R=1/2 soft decision LDPC decoder with 4-bit messages.

percentage of messages changing rapidly converges to a very small value, dependent on the input signal to noise ratio (SNR).While the parallel decoder architecture is necessarily larger than that of the hardware sharing architecture, the activity factor for the parallel architecture is very small resulting in extremely low power dissipation. Very little control logic is needed for the parallel architecture when compared to the hardware sharing architecture because the LDPC code graph is directly instantiated by the interconnection of the functional nodes. The parallel architecture also scales well with process technology compared to a hardware sharing or serial architecture. Higher throughput with a parallel decoder can be achieved by simply implementing a code with a larger block size and maintaining the same clock frequency. To increase throughput in a hardware sharing decoder it is necessary to run the decoder at a higher clock frequency, or in the case of turbo decoding to develop sliding window techniques to address the block-serial dependencies [10].

The main challenge implementing a parallel decoder architecture for LDPC codes is the interconnection of the functional units at the top level. For an LDPC code to provide good coding performance the check nodes must necessarily connect to variable nodes distributed across a large fraction of the data block length. This results in a large number of long routes at the top level. However, by careful management of the physical design process it is possible to avoid routing congestion and timing closure problems.

4. A PARALLEL 1024 BIT, RATE-1/2 SOFT DECISION LDPC DECODER

To demonstrate the feasibility of a parallel architecture for decoding LDPC codes, a prototype soft decision decoder was implemented based on a 1024 bit, rate-1/2 LDPC code. This corresponds to one of the block sizes and code rates proposed for 3G wireless turbo codes [11]. The prototype LDPC decoder performs 64 decoder iterations on every received packet, executing one decoder iteration per clock cycle. Thus, to reach the maximum target throughput for 3G wireless of 2 Mbit/s the prototype decoder requires a clock frequency of only 128 KHz. However, to demonstrate the massive throughput potential of the parallel architecture the decoder operates at a maximum clock frequency of 64 MHz giving a throughput of 1 Gbit/s.

4.1 Code Design and Performance

A 1024 bit, rate-1/2, soft decision LDPC code was designed based

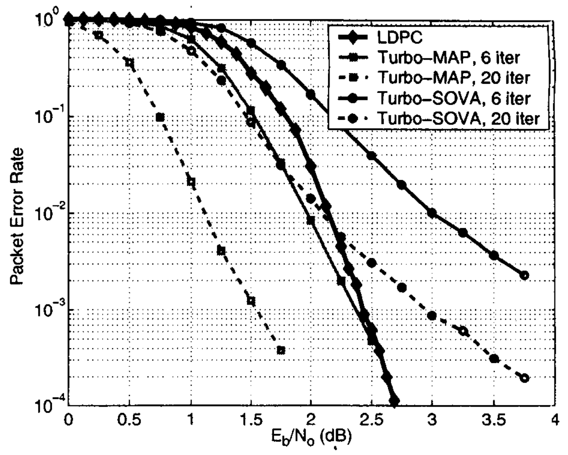

Figure 5. Comparison of 1024-bit, R=1/2, LDPC Code (4-bit messages) and 3G wireless turbo code using MAP and SOVA decoders (full floating point precision)[1].

on an irregular graph with an average column weight of 3.25 set elements per column, corresponding to an average row weight of 6.5 set elements per row. The actual weight of the columns implemented was 3, 6, 7 and 8. The check nodes were implemented as 256 weight 6 and 256 weight 7 nodes. Based on this weight profile a systematic code was constructed using an algorithm to maximize cycle lengths. All decoder messages consist of one parity and 3 reliability bits.

Although a 1024 bit, rate 1/2 LDPC code is theoretically inferior in terms of coding gain compared to a turbo code [6], implementation limitations of turbo codes do not necessarily enable the full realization of this difference. The coding gain of this decoder is compared to the 1024 bit rate 1/2 turbo code proposed for the 3G standard in Fig. 5 [11]. This turbo code has higher coding gain than the LDPC decoder when decoded using 20 iterations of the MAP algorithm. However, physical implementations of turbo decoders published so far are limited to only a few decoder iterations. With the number of decoder iterations is restricted to 6 the performance of the turbo code and LDPC code are almost identical as shown in Fig. 5.

4.2 Physical Design

The decoder was implemented in a 0.16μm, 1.5V CMOS process with 5 levels of metal using standard cells. A description for each of the functional units was written in VHDL and synthesized. To simplify the chip I/O and clock distribution the 1024 variable nodes were grouped into 16 macros labelled *vgrp0* to *vgrp15*. Macros were also created for the weight 6 and weight 7 check nodes. The main implementation challenge was the placement of the macros and routing of more than 26000 wires representing the messages between the macros. Custom algorithms were developed to place the macros and insert buffers to reduce the route lengths, reduce routing congestion, and achieve timing closure. The decoder was designed for a maximum clock frequency of 64MHz under worst case slow operating conditions. The layout of the decoder is shown in Fig. 6. The chip area is 7.5mm × 7.0mm and the utilization of 50% was limited by routing congestion. A higher utilization could have been achieved in a technology with 6 or 7 metal layers.

1. One decoder iteration for the turbo codes is taken to mean one constituent codes trellis decoding.

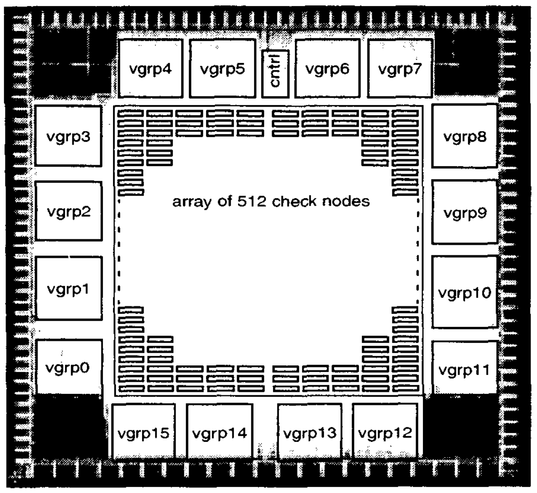

Figure 6. Layout of a parallel 1024 bit, rate-1/2, soft decision LDPC decoder.

4.3 Power Dissipation

Simulation of the decoder gives a message switching activity of 1% at a packet error rate of 1%, considered to correspond to average decoder power dissipation. The worst case power dissipation occurs if completely random data is applied to the decoder giving a packet error rate of 100% in which case the activity factor is 9%. Using the extracted parasitics for the decoder an average total power and worst case total power of the decoder were estimated as a function of throughput and are shown in Fig. 7. At the maximum throughput of 1 Gbit/s, the average total power dissipation of the decoder is only 220mW with a worst case value of 500mW for completely random received data. If a throughput of only 1 Mbit/s is required, e.g. wireless data comparable to 3G standard proposals, the clock frequency is 64KHz and the power dissipation is a miserly 220μW average and 500μW worst case. This compares extremely favorably with 170mW reported for a turbo decoder performing 3 iterations with a block size of 256 bits and throughput of 1Mbit/s implemented in 0.6μm, 3.3V CMOS technology [7].

4.4 Implementation Discussion

The dramatic throughput and power dissipation performance of the decoder is entirely due to the parallel architecture and the convergence properties of the algorithm. Further incremental gains in throughput, power dissipation, and utilization can be expected by applying known circuit techniques, logic families, and/or adopting a full custom design style

5. SUMMARY

A parallel architecture has been proposed for decoding Low Density Parity Check (LDPC) codes that achieves extremely low power dissipation and high throughput when compared to turbo and block turbo code decoders. A 1024 bit, rate-1/2 soft decision decoder has been implemented to demonstrate the feasibility of this approach.

The feasible design space for parallel LDPC codecs spans a wider range of code rates and throughputs than turbo and block turbo codes. Possible applications of parallel LDPC codecs include disk drive read channels, wireless and satellite communications.

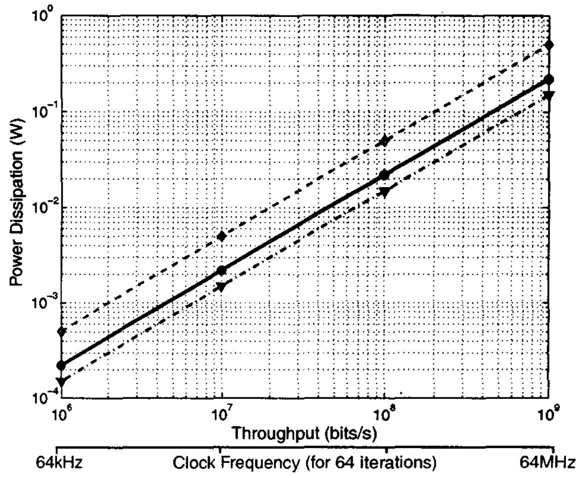

Figure 7. Power dissipation as a function of throughput and clock frequency for the parallel 1024 bit, rate-1/2 LDPC decoder.

6. REFERENCES

[1] M. Bossert, *Channel Coding for Telecommunications*, John Wiley & Sons, 1999.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting codes and decoding", *Proc. Int. Conf. Comm. '93*, pp 1064-1070, May 1993.

[3] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, "Near optimum decoding of product codes", *Proc. IEEE GLOBECOM '94*, pp 339-343, 1994.

[4] R. Gallager, "Low density parity check codes", *IRE Trans. Info. Theory*, Vol. IT-8, pp 21-28, Jan. 1962.

[5] D. MacKay and R. Neal, "Near Shannon limit performance of low density parity check codes", *Electron. Lett.*, Vol. 32(18), pp 1645-1646, Aug. 1996.

[6] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of provably good low-density parity check codes", submitted to *IEEE Trans. Inf. Theory*, March 1999.

[7] S. Hong and W. Stark, "Design and implementation of a low complexity VLSI turbo-code decoder architecture for low energy mobile wireless communications", *J. VLSI Sig. Proc.*, Vol. 24, pp 43-57, 2000.

[8] L.R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate", *IEEE Trans on Inf. Theory*, pp 363-77, March 1974.

[9] F.R. Kschischang and B.J. Frey, "Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models", *IEEE Journal on Selected Areas in Communications*, Vol. 16. No. 2, pp 219-30, Feb. 2000.

[10] P.J. Black and T. H. Y. Meng, "A 1-Gb/s, Four-State, Sliding Block Viterbi Decoder", *IEEE Journal of Solid State Circuits*, Vol. 32 No. 6, pp 797-805, June 1997.

[11] "*3rd Generation Partnership Project (3GPP); Technical Specification Group Radio Access Network Multiplexing and channel coding (TDD)*", available at http://www.3gpp.org