14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

HIGH-THROUGHPUT MULTI-RATE LDPC DECODER BASED ON ARCHITECTURE-ORIENTED PARITY CHECK MATRICES

Predrag Radosavljevic, Alexandre de Baynast, Marjan Karkooti, Joseph R. Cavallaro

ECE Department, Rice University
6100 Main Street, 77005, Houston, USA
phone: + (1)713-348-3579, fax: + (1) 713-348-5686
email: {rpredrag, debaynas, marjan, cavallar}@rice.edu

ABSTRACT

A high throughput pipelined LDPC decoder that supports multiple code rates and codeword sizes is proposed. In order to increase memory throughput, irregular block structured parity-check matrices are designed with the constraint of equally distributed odd and even nonzero block-columns in each horizontal layer for the pre-determined set of code rates. The designed decoder achieves a data throughput of more than 1 Gb/s without sacrificing the error-correcting performance of capacity-approaching irregular block codes. The architecture is prototyped on an FPGA and synthesized for an ASIC design flow.

1. INTRODUCTION

Low Density Parity Check (LDPC) codes optimized in [1] approach the capacity as close as 0.0045 dB. However, the optimization is not architecture-oriented since a fully random highly irregular parity-check matrix (PCM) structure is assumed. On the other hand, the tradeoffs between data throughput and area for structured partly-parallel LDPC decoders have been investigated in [2]. However, the authors restricted their study to block structured regular codes that do not exhibit excellent performance. Recently, block structured irregular LPDC codes have been proposed for the IEEE 802.11n standard [3]. Each nonzero sub-matrix in the PCMs is a randomly shifted identity matrix. The corresponding profiles are near optimal leading to excellent performance.

Block structured PCM is the key architecture-oriented constraint in the recent design of LDPC decoders. Authors in [4] offer a scalable decoder design based on the structured PCMs (regular and irregular) for supporting three code rates, but slow convergence of the belief propagation algorithm leads to only moderate decoding throughput. High decoding throughput is achieved in [5], but the partially parallel decoder design supports only regular (3,6) code. Although extremely fast, the lack of flexibility is a major disadvantage of the fully parallel decoder in [6].

In this paper, we design block structured irregular PCMs with an architecture-oriented constraint that directly allows the design of semi-parallel LDPC decoders with highly parallel memory access, while preserving the excellent error-correcting performance of irregular codes. The proposed LDPC decoder supports multiple code rates and codeword sizes with only moderate control and arithmetic logic overhead thanks to the common structure of designed PCMs. The pipelined version of the optimized belief propagation algorithm is employed in order to further increase decoding throughput. The prototype architecture of the proposed decoder is implemented on an FPGA. In addition, the low level synthesis results of an ASIC design are also discussed.

This work was supported in part by Nokia Corporation and by NSF under grants EIA-0224458 and EIA-0321266.

2. LOW DENSITY PARITY-CHECK CODES

An LDPC code is a linear block code specified by a very sparse PCM [7] where nonzero entries are typically placed at random. Each coded bit is represented by a variable node, whereas each parity check equation represents a check node. The variable node connection degree is the number of check equations in which it participates. The check node connection degree is the number of variable nodes that participate in the particular check equation. For convenience log-likelihood ratios (LLRs) are used for representation of the reliability messages. Let R_{mj} denote the check node LLR message sent from the check node m to the variable node j. Let $L(q_{mj})$ denote the variable node LLR message sent from the variable node j to the check node m. The messages $L(q_j)$ ($j = 1,\ldots,n$) represent the *a posteriori* probability ratio (APP messages) for all variable nodes (coded bits). All APP messages are initialized with the *a priori* (channel) reliability value of the coded bit (variable node) j ($\forall j = 1,\ldots,n$).

The proposed LDPC decoder utilizes the iterative layered belief propagation (LBP) algorithm as defined in [2]. This algorithm is a variation of standard belief propagation [7], and achieves about two times faster decoding convergence due to optimized scheduling of updated reliability messages [8]. The PCM can be viewed as a group of concatenated horizontal layers as shown in Fig. 1, where every layer represents the component code. The belief propagation algorithm is repeated for each horizontal layer and updated APP messages are passed between them. For each variable node j inside the current horizontal layer, messages $L(q_{mj})$ that correspond to all check nodes neighbors m are computed according to:

$$L(q_{mj}) = L(q_j) - R_{mj} \tag{1}$$

For each check node m, the messages R_{mj}, corresponding to all variable nodes j that participate in a particular parity-check equation, are computed according to:

$$R_{mj} = \prod_{j' \in N(m)\setminus\{j\}} \operatorname{sign}\left(L(q_{mj'})\right) \Psi\left[\sum_{j' \in N(m)\setminus\{j\}} \Psi\left(L(q_{mj'})\right)\right], \tag{2}$$

where $N(m)$ is the set of all variable nodes from parity-check equation m, and $\Psi(x) = -\log\left[\tanh\left(\frac{|x|}{2}\right)\right]$. The *a posteriori* reliability messages in the current horizontal layer are

updated according to:

$$L(q_j) = L(q_{mj}) + R_{mj}. \quad (3)$$

Hard decisions can be made after every horizontal layer based on the sign of $L(q_j)$, $j = 1, \ldots, n$. If all parity-check equations are satisfied or a pre-determined maximum number of iterations is reached, then the decoding algorithm stops. Otherwise, the algorithm repeats from (1) for the next horizontal layer.

3. ARCHITECTURE-ORIENTED DESIGN OF BLOCK-STRUCTURED IRREGULAR CODES

Optimized block structured PCMs have been proposed in the IEEE 802.11n standard [3]. As shown in Fig. 1, the PCM is partitioned into square sub-matrices with at most one nonzero entry per row/column. Each nonzero sub-matrix is a shifted identity matrix with a random shift value. We propose

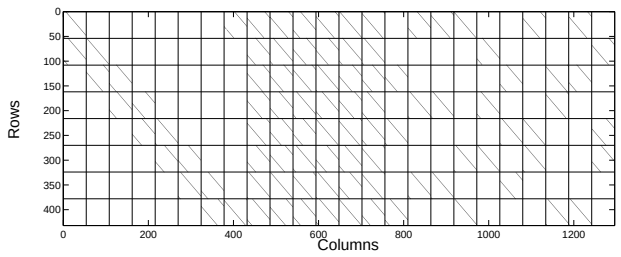

Figure 1: An example of novel block structured irregular parity-check matrix with 24 block-columns and 8 horizontal layers. Codeword size is 1296, rate$= 2/3$, and size of each sub-matrix is 54×54.

a new design of block-structured PCMs to achieve decoding throughput of approximately 1Gb/s by increasing the memory throughput while preserving excellent error-correcting performance.

The profile of the code is optimized through density evolution analysis (DEA) [7] such that the maximum variable node degree does not exceed the number of layers. For example, if eight layers are used for 2/3-rate PCMs, then the maximum variable node degree is also eight.

In order to parallelize the memory access, PCMs are designed with the constraint that any two consecutive nonzero sub-matrices from the same layer can belong to two independent APP memory modules. This property is essential to achieve high data throughput since the APP messages of two sub-matrices can be read/written from/to two separate memory modules in the same clock cycle. As shown in Fig. 2, two memory modules (referred as *A* and *B*) are assumed for the storage of APP messages per block-column. For illustration, it is supposed that the *i*th block-column is currently designed. Since the previous nonzero (*i*-4)th block-column belongs to the memory module *A*, the sub-matrix of the *l*th layer and *i*th block-column has to be an all zero matrix. However, the sub-matrix in the (*l*+1)th layer can be nonzero since the previous nonzero sub-matrix belongs to the memory module *B*.

The number of short cycles in these new PCMs should be small in order to preserve excellent decoding performance (typically to achieve a frame-error rate of 10^{-4}). In order to remove short cycles and to lower the error floor, we proceed

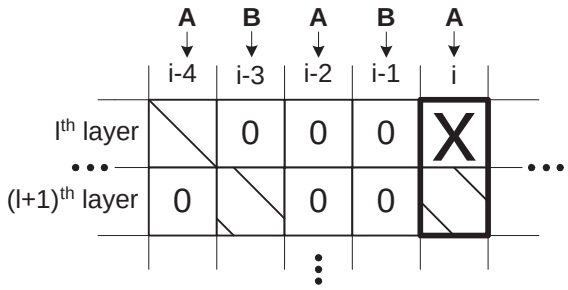

Figure 2: The design-constraint of equally distributed odd and even nonzero block-columns in PCMs is applied in order to achieve highly parallel memory access.

as in [9]. The position and the shift value of each nonzero sub-matrix is selected according to the cycle distribution. Table 1 lists the number of cycles for new PCMs with different numbers of block-columns (18, 24, 48, 72).

Table 1: The number of short cycles for different PCMs ($2/3$-rate code of length 1296). Maximum variable node and check node degrees are 8 and 13, respectively. Results for random matrix construction are also shown for comparison.

# Block-columns		18	24	48	72	rand
cycles	4	0	0	54	0	570
	6	1,008	6,911	7,990	791	11,921
	8	81K	304K	375K	58K	320K

The PCMs with 18 and 72 block-columns have the smallest number of short cycles. However, both are sparser since a sub-optimal profile with maximum variable node degree of six is used. For 18 block-columns, the maximum degree is limited to the number of layers; for 72 block-columns, it is not possible to remove all cycles of length 4 with maximum variable node degree of eight. The PCMs with 24 and 48 block-columns have identical profiles while the PCMs with 24 block-columns have a smaller number of cycles because the number of possible shift values is twice larger.

The frame error rate (FER) performance for PCMs designed with different number of block-columns are compared in Fig. 3. The PCMs with 24 block-columns outperform PCMs with 18 and 72 block-columns due to a better profile (about 0.4 dB performance loss). As is expected, the larger number of short cycles for PCMs with 48 block-columns introduces some performance loss compares to PCMs with 24 block-columns. Because of the reduced number of short cycles, no error-floor is observed until very low FER. PCMs with 24 block-columns provide the best performance and sufficiently high parallelism degree for high-throughput decoder implementations. Due to the near-optimal profile, the corresponding LDPC decoder exhibits excellent error-correcting performance.

4. DESIGN OF PIPELINED MULTI-RATE DECODER

The proposed LDPC decoder is based on the architecture-oriented block structured irregular PCMs with 24 block-

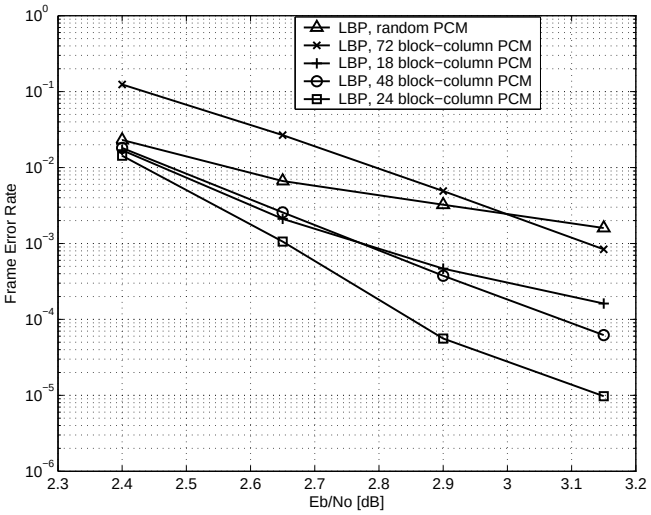

Figure 3: FER for different parity-check matrix (PCM) structures (code rate of 2/3, code size of 1296, maximum number of iterations is 15): random (same profile as 24 block-column PCMs), new block-structured irregular PCMs with 18, 24, 48, and 72 block-columns.

columns designed in Section 3. The main target is to achieve decoding throughput of approximately 1 Gb/s, while supporting multiple codeword sizes and code rates with only moderate area, memory and control overhead for next generation wireless systems. Single decoder architecture supports codeword sizes of: 648, 1296, 1944, and 2592, and code rates of: 1/2, 2/3, 3/4, and 5/6, to be compatible with IEEE 802.11n standard [3].

The architecture-oriented constraint of equally distributed odd and even nonzero block-column positions in every horizontal layer of the PCM provides highly parallel memory access. The PCMs designed in Section 3 (for all supported code rates and codeword sizes) allow reading/writing of APP messages that belong to two consecutive nonzero sub-matrices of the same horizontal layer from/to two independent memory modules in a single clock cycle. Each location in one APP memory module contains APP messages that correspond to one (out of twelve) odd block-columns (depth of the memory module is twelve). Another APP memory module contains APP messages from even block columns. Fig. 4 shows the partitioning of APP memory into two independent modules. The width of the valid memory content depends on the size of the square sub-matrix, and it can be up to 108 messages for the largest supported codeword size of 2592 (in this case the number of APP messages in one sub-matrix is 108).

Each check node memory location contains check node messages that correspond to two consecutive nonzero sub-matrices from the same horizontal layer. The entire content of the check node memory location is loaded/stored in a single clock cycle and accompanied (after loading) with APP messages from two memory architecture blocks in order to update variable node messages according to (1). An arithmetic precision of seven bits is chosen for representation of reliability messages (two's complement with one bit for the fractional part).

By construction, all rows inside a single horizontal layer

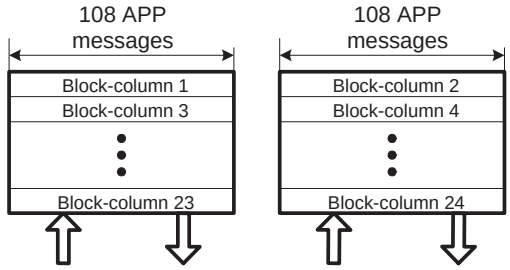

Figure 4: Organization of APP memory (identical for all code rates) into two dual-port memory blocks with messages from odd and even block-columns of the parity check matrix.

are independent and can be processed in parallel without any performance loss. Furthermore, every horizontal layer is processed through three pipeline stages as it is visualized in Fig. 5: memory reading stage, processing stage, and memory writing stage corresponding to equations (1), (2), and (3), respectively. The pipelining of layers assumes simultaneous reading and writing of messages from different memory addresses: dual-port RAMs are employed for APP memory modules (see Fig. 4), as well as for check node memory. The design of control logic unit ensures that there is no simultaneous reading and writing of reliability messages from the same memory address. The three-stage pipelined decoding introduces performance loss that is evaluated in Section 5.1 due to the overlapping between APP messages from different pipelined layers.

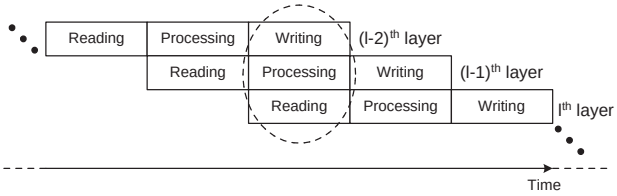

Figure 5: Belief propagation based on pipelining of three decoding stages (reading, processing, writing) for three consecutive horizontal layers of the parity check matrix.

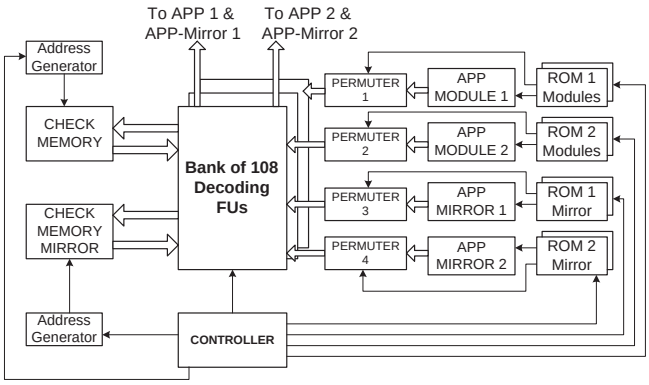

Figure 6: Block diagram of decoder architecture for block structured irregular parity-check matrices with 24 block-columns.

A high-level block diagram of the proposed decoder that supports sixteen PCMs (four code rates with four different codeword sizes) is shown in Fig. 6. Mirror memories (identical content as original memories) are required to support the pipelining of multiple layers and to avoid the buffering of large amounts of messages. Parity checking is based on the content of the APP modules and is performed after every decoding iteration. Four identical permuters are required for block-shifting of the APP messages after loading them from each original and mirror memory module. The permuter is composed of 7-bit input 2:1 multiplexers organized in multiple pipelined stages. Additional multiplexers (only 8.5% of the total permuter size) are required to support permutation of blocks of multiple sizes: 27, 54, 81, and 108 which correspond to the four supported codeword lengths.

The modified min-sum approximation with correcting offset [10] is employed as a part of the central processing stage inside the decoding function unit (DFU) shown in Fig. 7. A single DFU is responsible for decoding one row of the PCM through three pipeline stages according to equations (1), (2) and (3): reading, processing and writing stage, respectively as shown in Fig. 7. It is important to note that (2) is modified to represent the min-sum approximation with correcting offset. Three min-sum units inside the DFU serially search for the two smallest variable node messages (in absolute sense) in the current row of the PCM. There is no hardware overhead inside a single DFU to support multiple code rates and codeword sizes because of the serial nature of processing.

Support for different code rates and codeword sizes implies the usage of multiple PCMs. Information about each supported PCM is stored in four ROM modules (total of 64 ROM modules for 16 PCMs). Two ROM modules correspond to two APP memory modules: a single memory location in a ROM module contains the block-column position of a nonzero sub-matrix as well as its associated shift value. The block-column position is the reading/writing address of the odd/even APP memory module. Two additional ROM modules are required for storage of relative shift values (difference to the previous shift value of the same block-column) of odd and even block-columns. These modules are used after the first decoding iteration in order to avoid permutation of APP messages during the writing stage. A total of about 10 KBytes (all ROMs that correspond to original and mirror APP modules) is required to support four codeword sizes with four different code rates.

The control unit is designed to support a set of codeword sizes and code rates. The codeword size is not a critical issue because of the block-serial decoding principle that does not depend on the size of the sub-matrix: only a small control logic overhead is required. On the other hand, the number of nonzero sub-matrices per horizontal layer significantly varies with the code rate which affects latency of the reading/writing stages, as well as latency of the serial min-sum processing inside DFUs. Control logic needs to provide proper synchronization between pipelined decoding stages with variable latencies. The controller also needs to handle an exception which occurs if the number of nonzero sub-matrices in a horizontal layer is odd. In that case, only one block-column per clock cycle is read/written from/to an odd or even APP memory module. The full contents of the corresponding check node memory location (width of two sub-matrices) is loaded even though the second half of it is not valid. The control unit knows when this happens and disables the appropriate part of the arithmetic logic inside DFUs.

5. HARDWARE IMPLEMENTATION OF LDPC DECODER

A prototype architecture has been implemented in Xilinx System Generator and targeted to a Xilinx VirtexIV-fx60 FPGA. Table 2 shows the utilization statistics. Based on the XST synthesis tool report and full place and route statistics, a maximum clock frequency of 95.8 MHz can be achieved.

Table 2: Xilinx VirtexIV-fx60 FPGA utilization statistics

Resource	*Used*	*Utilization rate*
Slices	19,595	77%
LUTs	36,452	72%
Block RAMs	140	60%

The proposed decoder architecture using 7-bit arithmetic precision was synthesized for a Chartered Semiconductor 0.13μm, 1.2 V, 6 metal level standard cell CMOS technology using the BEE/Insecta design flow [11] and Synopsys tools. A system clock frequency of at least 200 MHz is targeted which is compatible with a decoding throughput of more than 1 Gb/s. The Chartered memory compiler was used to generate efficient RAM and ROM blocks. The synthesis results provide an estimate for the decoder core area of $5.21mm^2$ and the highest operational clock frequency of 412 MHz.

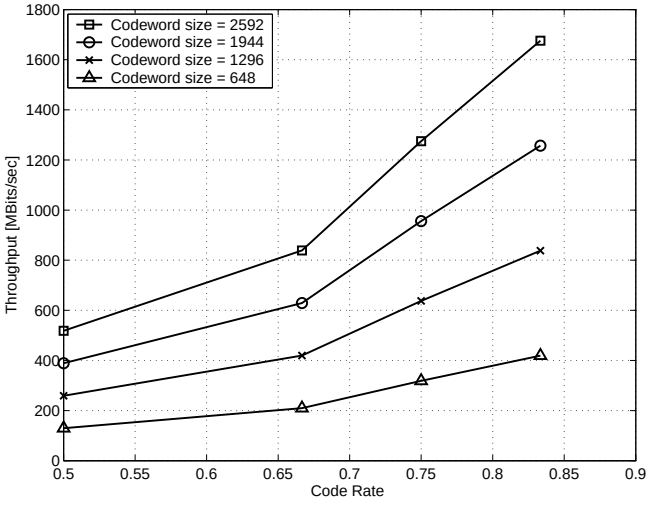

Figure 8: Decoding throughput for different code rates and codeword lengths based on average number of decoding iterations.

5.1 Decoding Throughput and Performance Results

An estimation of decoding throughput (information bits are considered) is based on the average number of decoding iterations required to achieve a frame error rate of 10^{-4}, with a maximum number of iterations set to 15 with a moderate clock frequency of 200 MHz. The achievable decoding

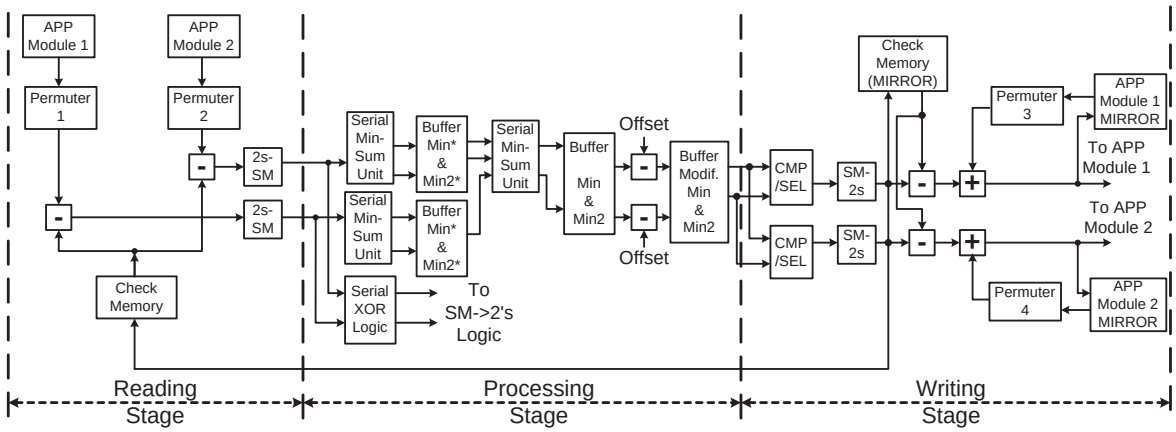

Figure 7: Block diagram of single decoding function unit (DFU) with three pipeline stages and block-serial processing. The interface to the APP and check node memories is also included.

throughput as a function of code rate and codeword size is shown in Fig. 8.

Figure 9 shows the FER performance of the implemented decoder for $2/3$-rate code of length 1296. A small loss (about 0.1 dB for FER of 10^{-4}) is introduced due to pipelining of layers. It can be noticed that 7-bit arithmetic precision is sufficient for accurate decoding.

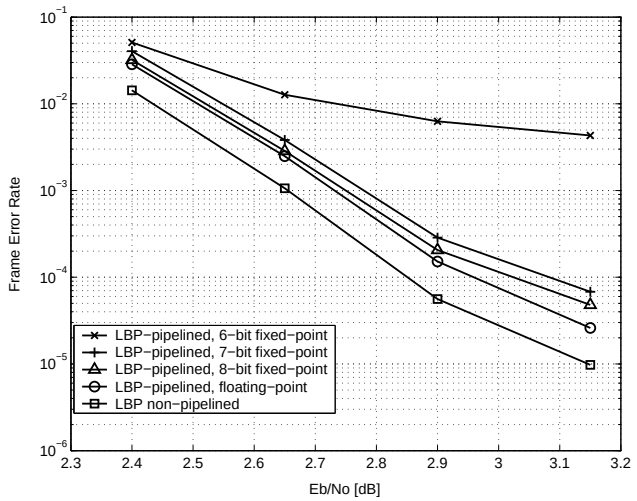

Figure 9: FER for a PCM with 24 block-columns (code rate of 2/3, code size of 1296, maximum number of iterations is 15): non-pipelined LBP, and pipelined LBP (floating and fixed-point).

6. CONCLUSION

A pipelined semi-parallel LDPC decoder is proposed to support multiple code rates and codeword sizes. It is based on newly optimized block structured irregular codes with an architecture-oriented constraint that allows reading/writing of reliability messages from two sub-matrices in each clock cycle. The excellent error-correcting performance of irregular codes is preserved while high decoding throughput is achieved. The decoder is prototyped on a Xilinx FPGA and synthesized for ASIC implementation.

REFERENCES

[1] Sae-Young Chung, G.D. Forney, T.J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *Communications Letters, IEEE*, vol. 5, pp. 58–60, Feb. 2001.

[2] M.M. Mansour and N.R. Shanbhag, "High-throughput LDPC decoders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems.*, vol. 11, pp. 976–996, Dec. 2003.

[3] *IEEE 802.11 Wireless LANsWWiSE Proposal: High Throughput extension to the 802.11 Standard*. IEEE 11-04-0886-00-000n.

[4] L. Yang, H. Lui, and C.-J.R. Shi, "Code construction and FPGA implementation of a low-error-floor multi-rate low-density parity-check code decoder," *to appear in IEEE Transactions on Circuits and Systems*.

[5] Se-Hyeon Kang and In-Choel Park, "Loosely coupled memory-based decoding architecture for low density parity check codes," *to appear in IEEE Transactions on Circuits and Systems*.

[6] A. Darabiha, A.C. Carusone, and F.R. Kschischang, "Multi-Gbit/sec low density parity check decoders with reduced interconnect complexity," in *ISCAS 2005. IEEE International Symposium on Circuits and Systems*, May 2005.

[7] S.Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657–670, Feb. 2001.

[8] P. Radosavljevic, A. de Baynast, and J.R. Cavallaro, "Optimized message passing schedules for LDPC decoding," in *39th Asilomar Conference on Signals, Systems and Computers, 2005*, pp. 591–595, Nov. 2005.

[9] K.S. Kim, S.H. Lee, Y.H. Kim, and J.Y. Ahn, "Design of binary LDPC code using cyclic shift matrices," *Electronics Letters*, vol. 40, pp. 325–326, March 2004.

[10] Jinghu Chen, A. Dholakai, E. Eleftheriou, M.P.C. Fossorier, and Xiao-Yu Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, pp. 1288 – 1299, Aug. 2005.

[11] Chen Chang, B. Nikolic, and et. al., "Rapid design and analysis of communication systems using the BEE hardware emulation environment," in *Proceedings of the 14th IEEE International Workshop on Rapid Systems Prototyping*, June 2003.