MULTI-RATE HIGH-THROUGHPUT LDPC DECODER: TRADEOFF ANALYSIS BETWEEN DECODING THROUGHPUT AND AREA

Predrag Radosavljevic, Alexandre de Baynast, Marjan Karkooti, and Joseph R. Cavallaro
Department of Electrical and Computer Engineering, Rice University
{rpredrag, debaynas, marjan, cavallar}@rice.edu

ABSTRACT

In order to achieve high decoding throughput (hundreds of MBits/sec and above) for multiple code rates and moderate codeword lengths (up to 2.5K bits), several decoder solutions with different levels of processing parallelism are possible. Selection between these solutions is based on a threefold criterion: hardware complexity, decoding throughput, and error-correcting performance. In this work, we determine multi-rate LDPC decoder architecture with the best tradeoff in terms of area size, error-correcting performance, and decoding throughput. The prototype architecture of optimal LDPC decoder is implemented on FPGA.

I. INTRODUCTION

[1] Recent designs of LDPC decoders are mostly based on block-structured Parity Check Matrices (PCMs). Tradeoff between data throughput and area for structured partly-parallel LDPC decoders has been investigated in [5, 8]. However, authors restricted their study to block structured regular codes that do not exhibit excellent performance. Scalable decoder from [9] that supports three code rates is based on structured regular and irregular PCMs, but slow convergence causes only moderate decoding throughput. Although extremely fast, the lack of flexibility and relatively large area occupation is major disadvantage of fully parallel decoder from [3].

Our goal is to design high-throughput (≈ 1GBits/sec) LDPC decoder that supports multiple code rates (between 1/2 and 5/6) and moderate codeword lengths (up to 2.5K bits) as it is defined by the IEEE 802.16e and IEEE 802.11n wireless standards [7]. Implemented decoder aims to represent the best tradeoff between decoding throughput and hardware complexity with excellent error-correcting performance. Range of decoder solutions with different levels of processing parallelism are analyzed. Decoder solution with the best throughput per area ratio is chosen for hardware implementation. A prototype architecture is implemented on Xilinx FPGA.

II. LOW DENSITY PARITY-CHECK CODES

LDPC code is a linear block code specified by a very sparse PCM as shown in Fig. 1 with random placement of nonzero entries [7]. Each coded bit is represented by a PCM column (variable node), while each row of PCM represents parity check equation (check node). Log-likelihood ratios (LLRs) are used for representation of reliability messages in order to simplify arithmetic operations: R_{mj} message denotes the check node LLR sent from the check node m to the variable node j, $L(q_{mj})$ message represents variable node LLR sent from the variable node j to the check node m, and $L(q_j)$ messages ($j = 1, \ldots, n$) represent the *a posteriori* probability ratio (APP messages) for all coded bits. APP messages are initialized with the *a priori* (channel) reliability value of the coded bit j ($\forall j = 1, \ldots, n$).

Iterative layered belief propagation (LBP) algorithm is a variation of the standard belief propagation (SBP) algorithm [5], and achieves twice faster decoding convergence because of optimized scheduling of reliability messages [6]. As it is shown in Fig. 1, typical block-structured PCM is composed of concatenated horizontal layers (component codes) and shifted identity sub-matrices. Belief propagation algorithm is repeated for each component code while updated APP messages are passed between the sub-codes. For each variable node j inside 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, messages R_{mj} for all variable nodes j that participate in particular parity-check equation are computed according to:

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

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:

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

If all parity-check equations are satisfied or pre-determined maximum number of iterations is reached, decoding algorithm stops. Otherwise, the algorithm repeats from (1) for the next horizontal layer.

III. LDPC DECODERS WITH DIFFERENT PROCESSING PARALLELISM AND DESIGNED PCMS

Processing parallelism consists of two parts: parallelism in decoding of concatenated codes (horizontal layers of PCM), and parallelism in reading/writing of reliability messages from/to memory modules. According to different

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

levels of processing parallelism, while targeting decoding throughput of hundreds of MBits/sec and above, the following decoder solutions are considered:

1. L1RW1: Decoder with full decoding parallelism per one horizontal layer (LBP algorithm), reading/writing of messages from one nonzero sub-matrix in each clock cycle.
2. L1RW2: Decoder with full decoding parallelism per one horizontal layer (LBP algorithm), reading/writing of messages from two consecutive nonzero sub-matrices in each clock cycle.
3. L3RW1: Decoder with pipelining of three consecutive horizontal layers, reading/writing of messages from one nonzero sub-matrix in each clock cycle.
4. L3RW2: Decoder with pipelining of three consecutive horizontal layers, reading/writing of messages from two consecutive nonzero sub-matrices in each clock cycle.
5. L6RW2: Decoder with double pipelining (pipelining of six consecutive horizontal layers), reading/writing of messages from two consecutive nonzero sub-matrices in each clock cycle.
6. FULL: Fully parallel decoder with simultaneous execution of all layers and simultaneous reading/writing of all reliability messages (SBP decoding algorithm).

Optimized block-structured irregular PCMs with excellent error-correcting performance have been proposed for IEEE 802.16 and 802.11n wireless standards [7]. Inspired by these codes, we propose novel design of irregular block-structured PCMs which allows twice parallel memory access while preserving error-correcting performance. An example of designed block-structured PCMs is shown in Fig. 1.

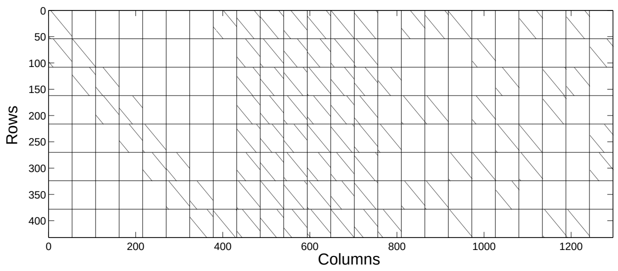

Figure 1: Example of designed block-structured irregular parity-check matrix with 24 block-columns and 8 horizontal layers. Codeword size is 1296, rate$= 2/3$, and size of each sub-matrix is 54×54.

The main constraint in proposed PCM design is that any two consecutive nonzero sub-matrices from the same horizontal layer belong to odd and even PCMs' block-columns (see Fig.1). This property is essential for achieving more parallel memory access since APP messages from two sub-matrices can be simultaneously read/written from/to two independent APP memory modules. PCMs with 24 block-columns provide the best performance due to near-optimal profile and sufficiently high parallelism degree for high-throughput decoder implementations: all proposed decoder solutions including the fully parallel architecture support newly designed PCMs.

In order to analyze different levels of processing parallelism, we first introduce different ways of memory access parallelization, as well as different ways to decode horizontal layers of PCM. Figure 2 (part labelled as RW1) shows APP memory organization composed of single dual-port RAM block where each memory location contains APP messages from one (out of 24) block-column. This memory organization allows read/write of one full sub-matrix of PCM in each clock cycle. The width of the valid memory content depends on the codeword length (size of the sub-matrix), and it can be up to 108 messages for the largest supported codeword length of 2592. Meanwhile, each check node memory location contains check node messages from single nonzero sub-matrix.

Architecture-oriented constraint of equally distributed odd and even nonzero block-columns in every horizontal layer allows read/write of two sub-matrices per clock cycle. As it is shown in Fig. 2 (part labelled as RW2) APP memory is partitioned into two independent modules. Each location of one APP memory module contains APP messages that correspond to one (out of twelve) odd block-columns. Another APP memory module contains APP messages from even block columns. Meanwhile, each check node memory location contains check node messages that correspond to two consecutive nonzero sub-matrices from the same horizontal layer. Whole content of the check node memory location is loaded/stored in a single clock cycle. It is important to observe that dual-port RAMs are replaced with single port RAMs if there is no pipelining of horizontal layers. However, memory organization is independent of the number of memory ports and remains same.

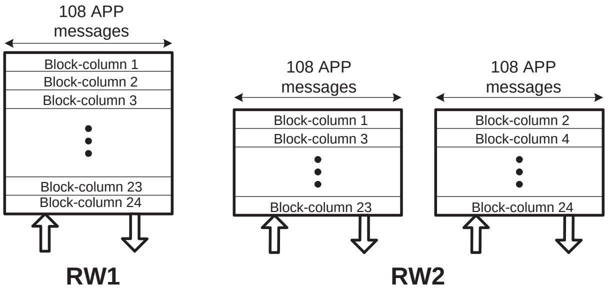

Figure 2: Organization of APP memory into single dual-port RAM module, and two independent dual-port RAM modules with messages from odd and even block-columns of parity check matrix.

By construction, all rows inside single horizontal layer are independent and can be processed in parallel without any performance loss. Every horizontal layer is processed through three stages: memory reading stage, processing stage, and memory writing stage corresponding to Eqs. (1), (2), and (3) respectively. Processing parallelism can be increased if three consecutive horizontal layers are pipelined as it is visualized in Fig. 3 and labelled as L3. Three-stage pipelined decoding introduces certain performance loss due to the overlapping between the same APP messages from pipelined layers.

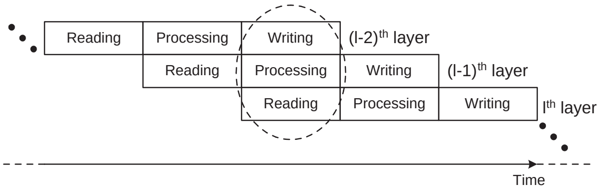

Figure 3: Belief propagation based on pipelining of three decoding stages for three consecutive horizontal layers of parity check matrix.

Processing parallelism is two times increased if pipelining of six consecutive layers is employed, as it is shown in Fig. 4 and labelled as L6. In order to avoid reading/writing collisions (simultaneous reading and writing of APP messages from the identical block-columns but from two different horizontal layers), it is necessary to postpone start of every second layer by one clock cycle (see Fig. 4).

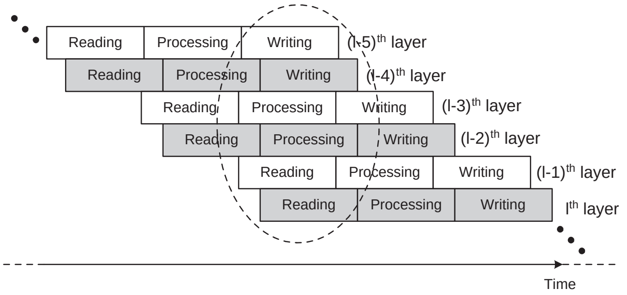

Figure 4: Belief propagation based on pipelining of three decoding stages for six consecutive horizontal layers of parity check matrix.

IV. Throughput-Area-Performance Tradeoff Analysis for Proposed LDPC Decoders

Proposed six decoder solutions with different processing parallelism levels from section III. are compared. It is assumed that each solution (except L6RW2 architecture) supports multiple code rates (from 1/2 to 5/6) and codeword lengths (648, 1296, 1944, and 2592) as well as newly designed block-structured PCMs with 24 block-columns. The L6RW2 solution supports code rates of up to 3/4 since designed PCM with 24-block-columns has only four horizontal layers for code rate of 5/6: this solution has reduced flexibility. Figure 5 shows decoding throughput and area complexity for analyzed LDPC decoders: decoding parallelism and memory access parallelism increase from left to right (same order as labelled from L1RW1 to FULL in section III.).

Hardware complexity is represented as an area size in mm^2 assuming 0.18 μm 6-metal CMOS process. Total area is computed as a summation of the arithmetic area and area of the memory. Arithmetic part of each decoder is represented as a number of standard CMOS logic gates (also shown on Fig. 5), while it is assumed typical TSMC's design rule for 0.18 μm technology of 100K logic gates per mm^2 [1]. Memory size is given as the number of bits required for a storage of APP and check messages (SRAMs) and supported PCMs (ROMs). Corresponding memory area is computed by assuming SRAM cell size of a 4.65 μm^2 which is typical size for 0.18 μm technology [1]. Dual-port SRAMs are required for L3RW1, L3RW2 and L6RW2 decoder architectures since they employ multiple pipeline stages: memory cell area is typically increased two times for the same design rule [2].

Tradeoff is defined as a ratio between decoding throughput and total area and it is represented in MBits/sec/mm^2. Decoding throughput is based on the average number of decoding iterations required to achieve frame-error rate (FER) of 10^{-4} (averaged over 10^6 codeword transmissions) for the largest supported code rate and code length for clock frequency of 200 MHz. Since fully parallel solution does not have inherent flexibility arithmetic logic dedicated to 16 different rate-size combinations is necessary which significantly increases arithmetic area. On the other hand, same set of latches are utilized for the storage of reliability messages for all 16 supported cases.

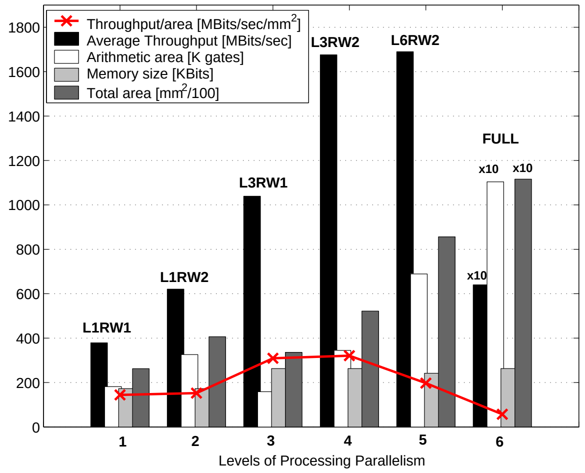

Figure 5: Average decoding throughput, hardware complexity and tradeoff ratio for LDPC decoder solutions with different levels of processing parallelism labelled same as in section III.

By employing only pipelining of three consecutive horizontal layers decoding throughput is increased by approximately three times while arithmetic area is increased only marginally. On the other hand, memory area is doubled since mirror memories are required for storage of APP and check messages. Overall, three-stage pipelining improves throughput/area ratio (see Fig. 5, L1RW1 vs. L3RW1 architecture, and L1RW2 vs. L3RW2 architecture). If memory access parallelism is doubled (from RW1 to RW2), throughput is directly increased by more than 50% while arithmetic area is twice larger and memory size is same. Overall, only by increasing memory access parallelism throughput/area ratio is slightly improved (see Fig. 5, L1RW1 vs. L1RW2 architecture, and L3RW1 vs. L3RW2 architecture).

Further increase of decoding parallelism (L6RW2 and FULL solutions) does not improve tradeoff ratio since throughput improvements are smaller than the increase of area occupation. We can expect similar effect if memory access parallelism is further increased. As an illustration, if four sub-matrices per clock cycle are loaded/stored (not shown since it requires special redesign of block-structured PCMs) arithmetic area increases two times but decoding throughput improves only by approximately 25% since processing latency inside DFUs will start to dominate compare to memory access latency.

It can be observed from Fig. 5 that the best throughput per area ratio is achieved for three-stage pipelining with memory access that allows reading/writing of two sub-matrices per clock cycle (L3RW2 solution). In the same time, performance loss due to pipelining is acceptable (about 0.1dB for FER around 10^{-4}, see Fig. 6). In order to avoid performance loss double pipelining and SBP require additional decoding iterations which decreases decoding throughput.

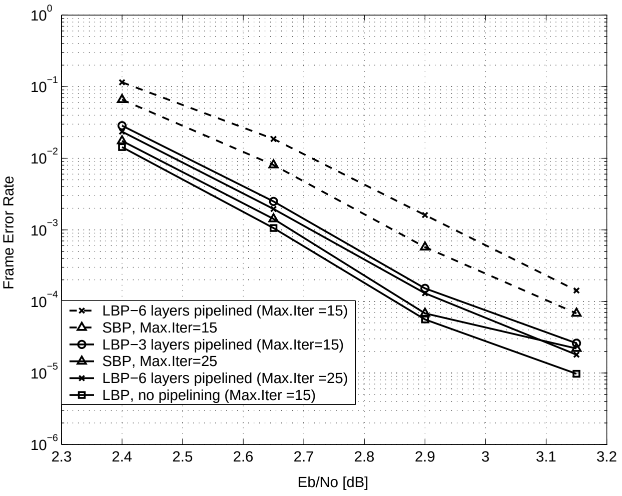

Figure 6: FER for PCM with 24 block-columns (code rate of 2/3, code size of 1296: non-pipelined LBP, pipelined LBP, double-pipelined LBP, SBP.

V. Three-stage Pipelined LDPC Decoder with Double-Parallel Memory Access

LDPC decoder with three-stage pipelining and memory organization that supports access of two sub-matrices during the single clock cycle is chosen for the hardware implementation as the best tradeoff between throughput and area. High-level block diagram of decoder architecture is shown in Fig. 7. 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, which is compatible with IEEE 802.16 and 802.11n standards [7].

Four identical permuters are required for block-shifting of APP messages after loading them from each original and mirror memory module. Scalable permuter (permutation of blocks of sizes: 27, 54, 81, and 108) is composed of 7-bit input 3:1 multiplexers organized in multiple pipelined stages. Modified min-sum approximation with correcting offset [4] is employed as a central part of decoding function unit (DFU). Single DFU is responsible for processing one row of PCM through three pipeline stages according to Eqs. (1), (2), and (3) as it is shown in Fig 8.

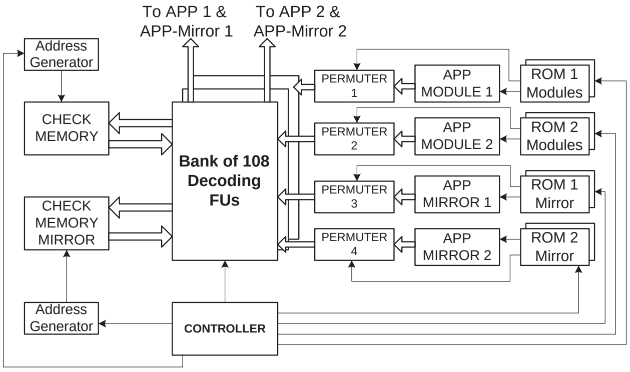

Figure 7: Block diagram of decoder architecture for block structured irregular parity-check matrices with 24 block-columns.

Support for different code rates and codeword lengths implies usage of multiple PCMs. Information about each supported PCM is stored in several ROM modules. Single memory location of ROM module contains block-column position of nonzero sub-matrix as well as associated shift value. Block-column position is reading/writing address of appropriate APP memory module. In order to avoid permutation of APP messages during the writing stage, relative shift values (difference between two consecutive shift values of the same block-column) are stored instead of original shift values.

Support of multiple code rates is defined by control unit. The number of nonzero sub-matrices per horizontal layer significantly varies with the code rate which affects memory latency, as well as processing latency inside DFUs. Control logic provides synchronization between pipelined stages with variable latencies. Controller also handles an exemption which occurs if the number of nonzero sub-matrices in horizontal layer is odd. In that case, only one block-column per clock cycle is read/written from/to odd or even APP memory module. Full content of corresponding check node memory location (width of two sub-matrices) is loaded while second half of it is not valid. Control unit then disables part of arithmetic logic inside DFUs.

A. *Error-Correcting Performance and Hardware Implementation*

Figure 9 shows FER performance of implemented decoder for $2/3$-rate code and length of 1296. It can be noticed that

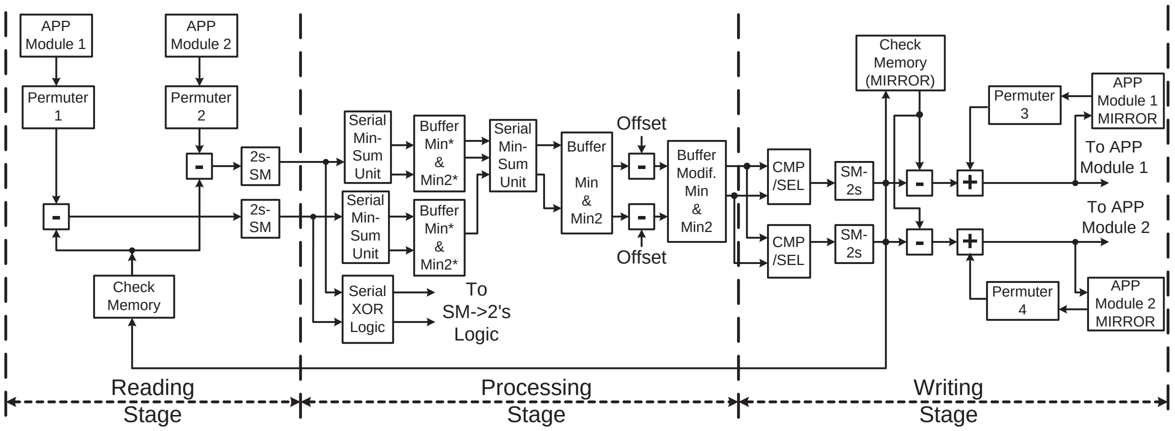

Figure 8: Block diagram of single decoding function unit (DFU) with three pipeline stages and block-serial processing. Interface to APP and check node memories is also included.

7-bit arithmetic precision is sufficient for accurate decoding. Arithmetic precision of seven bits is chosen for representation of reliability messages (two's complement with one bit for the fractional part) as well as for all arithmetic operations.

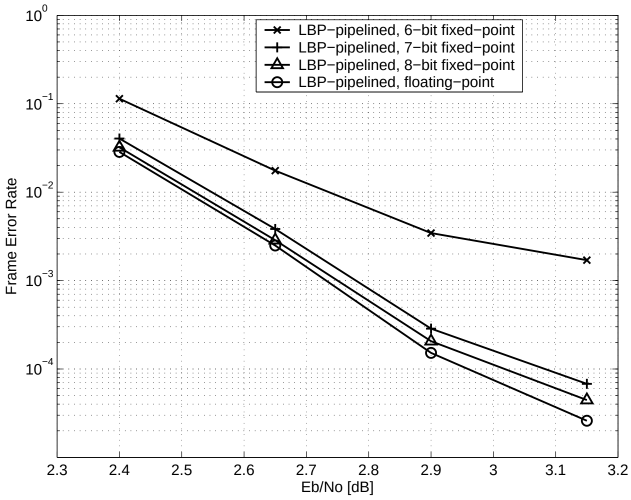

Figure 9: FER for implemented decoder (code rate of 2/3, code length of 1296, maximum number of iterations is 15): pipelined LBP (floating vs. fixed-point).

A prototype architecture has been implemented in Xilinx System Generator and targeted to a Xilinx Virtex4-XC4VFX60 FPGA. Table 1 shows the utilization statistics. Based on XST synthesis tool report, the maximum clock frequency of 130 MHz can be achieved which determines average throughput of approximately 1.1 GBits/sec

Table 1: Xilinx Virtex4-XC4VFX60 FPGA utilization statistics.

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

VI. Conclusion

In this paper multi-rate decoder architecture that represents the best tradeoff in terms of throughput, area and performance is found and implemented on FPGA. Identical tradeoff analysis can be applied for a wide range of code rates and codeword lengths. We believe that pipelining of multiple horizontal layers (blocks of PCM) combined with sufficiently parallel memory access is general tradeoff solution that can be applied for other block-structured LDPC codes.

References

[1] http://www.amis.com/asics/standard_cell.html.

[2] http://www.taborcommunications.com/dsstar/04/0224/107497.html.

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

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

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

[6] 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*, pages 591–595, Nov. 2005.

[7] Robert Xu, David Yuan, and Li Zeng. High girth LDPC coding for OFDMA PHY. Technical Report IEEE C802.16e-05/031, IEEE 802.16 Broadband Wireless Access Working Group, 2005.

[8] 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*.

[9] 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*.