Journal of Theoretical and Applied Information Technology
20th May 2014. Vol. 63 No.2
© 2005 - 2014 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN FPGA BASED OVERLAPPED QUASI CYCLIC LDPC DECODER FOR WI-MAX

[1]G.AMIRTHA GOWRI, [2]S.SUBHA RANI
Associate Professor, Department of Electronics and Communication Engineering,
Kumaraguru College of Technology, Coimbatore, India
[2]Professor, Department of Electronics and Communication Engineering,
PSG College of Technology, Coimbatore, India
E-mail: [1] amirthagowri.g.ece@kct.ac.in , [2] ssr@ece.psgtech.ac.in

ABSTRACT

In this paper, we present a partially parallel Quasi cyclic Low Density Parity Check decoder architecture for WiMAX IEEE 802.16e standard. Two phase message passing Min-sum decoding algorithm is used to decode the Low Density Parity Check codes. The decoder is designed for code rate ½ and code length of 576 bits and 2304bits. The decoder is easily configurable to support different code lengths of WiMAX IEEE 802.16e standard. In the proposed architecture, two phases of decoding are overlapped to increase the throughput and hardware utilization is increased. In this work , a novel configurable variable node processor is introduced to increase the hardware utilization. The proposed architectures are implemented on Xilinx Virtex V xc5vlx110-3-ff676 FPGA.

Keywords: *Field programmable Gate Arrays, Min-sum Decoding Algorithm, Quasi cyclic Low Density Parity check Codes.*

1. INTRODUCTION

Low Density Parity Check Codes (LDPC) are a class of linear block codes for forward error correction, first proposed by Gallager [1] in 1962. LDPC codes are capacity approaching codes [2,3]. LDPC decoder is more realizable due to inherent parallelism present in the decoding algorithm of LDPC codes. So, LDPC codes find applications in digital video broadcasting through satellite, mobile communication and magnetic storage systems.

LDPC codes are classified as random codes and structured codes. The hardware design of encoder and decoder is more complex for random codes, even though they have excellent error correcting performance. The performance of structured codes such as QC-LDPC codes are closer to random codes and hardware complexity is also reduced. It is easy to design QC-LDPC codes with multiple code rates and code lengths. Hence, these structured QC-LDPC codes are included in many communication standards such as DVB-S2, 10GBASE-T, IEEE802.11 and IEEE802.16e. The LDPC codes are decoded using soft decision iterative message passing algorithm. The two phase message passing(TPMP) algorithm presented in Gallager's work[1] has optimal error correcting performance, but it is hard to realize. Based on the TPMP algorithm several simplified algorithms have been developed. The sum product algorithm (SPA)[5] which uses log likelihood ratios and avoids exponential calculations. Even though the SPA has excellent error correcting performance, it is not suitable for VLSI design due to more mathematical computations. So, Min-sum algorithm (MSA) [6] , an approximation of SPA which reduces the hardware complexity.

Based on the requirement, the decoders may be realized as :

1) Fully parallel
2) Fully Serial
3) Partly parallel

Fully parallel architectures are desirable for high throughput applications. Fully parallel architectures suffer by routing congestion and silicon area is increased due to interconnection wires. The ASIC implementation of a 1024-b, rate1/2 fully parallel architecture presented in[8] directly maps the tanner graph into hardware and an information throughput of 500 Mbps is obtained. Certain optimization measures can be taken to alleviate routing congestion and routing delay in these fully parallel architectures. The critical path delay of the decoder[9] was reduced by dividing the longest wires into several short wires with pipeline registers and log-likelihood ratio messages are transmitted along with these pipelined paths and achieved a throughput of 13.21 Gbps. The energy per bit is also decreased. The fully parallel architectures are realized only for high data rate applications, even though several techniques such as bit serial [10], multi split row [11] are proposed to reduce routing congestion. The routing congestion of these architectures increases as code length grows. In contrast to fully parallel decoders, fully serial decoders have one check node processing unit and one variable node processing unit and large memory is required to store data. The hardware complexity is less for these serial architectures and a very low data throughput is achieved. The partly parallel architectures balance the hardware complexity and throughput. The partly parallel architectures are designed for different area/throughput requirements[12-23]. The layered decoding algorithm is used in [24] and the architecture supports irregular LDPC codes with multiple code lengths and code rates of Wi-MAX standard IEEE802.16e.In this architecture, different parallel factors are used to improve the throughput. The architectures presented in[25,26] support regular LDPC codes of Wi-MAX and WLAN standards. Modified min-sum algorithm is used in[25]. These architectures are used for multimedia communication. Different parallelism factors were used to improve the throughput.

In this paper, we propose a partly parallel architecture in which two phase message passing min-sum (TPMP-MS) decoding algorithm is used. In this TPMP-MS, decoding is done in two phases: 1.check node updating phase and 2. Variable node updating phase. These two phases are executed one after the other. In our work, these two phases are partially overlapped based on the dual diagonal structure of the base matrix. In this decoder, a configurable variable node processing unit is introduced to improve the hardware utilization.

The remainder of the paper is organized as follows. A brief review of LDPC codes and decoding algorithm is presented in section2, The proposed decoder is presented in section 3. FPGA implementation results are discussed section 4. Section 5 concludes the paper.

2. LDPC CODES AND DECODING ALGORITHM

2.1 LDPC Codes

LDPC codes are represented by M×N sparse parity check matrix ‘H’ where ‘N’ represents code length, ‘M’ represents number of parity bits , K = N – M represents message length and K/N is code rate. The number of 1’s present in each row and column is defined as row weight and column weight respectively. The LDPC codes may be regular if row and column weights are constant or irregular otherwise.

The PCM ‘H’ is graphically represented by tanner graph which has two sets of nodes: N number of variable nodes corresponding to each column of the PCM and M number of check nodes corresponding to each row of the PCM. An edge is connected between a variable node and a check node if the element H_{ij} of the PCM is ‘1’. The example ‘H’ matrix and its corresponding tanner graph are shown in Figure 1 and Figure 2.

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Figure 1. Example ‘H’ Matrix

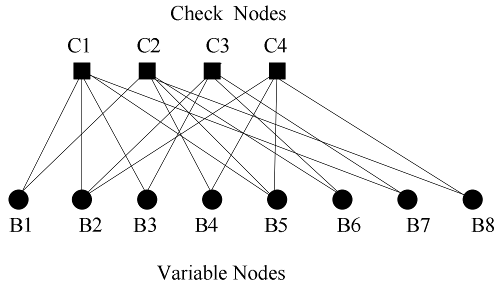

Figure 2: Tanner Graph of the example ‘H’Matrix

The structured QC-LDPC codes are defined by a base matrix H_b of size $m_b \times n_b$ which is an array of circulant matrices. The ‘H’ matrix of QC-LDPC codes are constructed from the base matrix H_b by replacing each element of the base matrix by z × z circulant matrix. The circulant matrix is a zero matrix if the element of the base matrix is a negative number or identity matrix if the base matrix has ‘0’ entry or shifted identity matrix if the entry is a positive integer which specifies the shift values of the identity matrix.

2.2 IEEE 802.16e Standard QC-LDPC Codes

IEEE 802.16e standard for Wi-MAX defines four code rates 1/2, 2/3, 3/4 and 5/6 and 19 code lengths. The base matrix of sizes of 12 x 24, 8 x

Journal of Theoretical and Applied Information Technology
20th May 2014. Vol. 63 No.2
© 2005 - 2014 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

24, 6 x 24 and 4 x 24 support the above code rates respectively. The same base matrix is used as a platform for different code lengths related to a selected code rate and different code lengths can be achieved by having different values of z. The value of z is increased from 24 to 96 in steps of 4. So, 19 different code lengths of size 576 bits (with z = 24) to 2304 bits (with z = 96) is obtained. The base matrix of size 12 x 24 is shown in Figure 3in Appendix-I which has dual diagonal structure. The diagonal elements are identity matrices which are marked in red colour in the Figure 3.

2.3 Min-Sum Decoding Algorithm

Iterative message passing algorithms are used to decode the LDPC codes. Soft belief propagation(BP) algorithm gives the good decoding performance. Binary phase shift keying (BPSK) modulation is used to modulate the coded information bits. The modulated information is transmitted over AWGN channel with mean '1' and variance σ^2. The reliability messages are represented as logarithmic likelihood ratios. The general BP algorithm is composed of two phases of message passing i.e. from variable to check nodes and check to variable nodes.

The sum product algorithm is regarded as the standard algorithm for decoding LDPC codes. This algorithm suffers performance loss when the processing data has wide dynamic range, since a trade off must be made between hardware resources and data precision. This problem is overcome by min- sum decoding algorithm which achieves a comparable performance to SPA and complexity in check node process is simplified. The decoding steps are:

1. The belief propagation algorithm computes the posterior probability that a given bit in the transmitted code word $C = [C_0, C_1, \ldots, C_{N-1}]$ equals '1', given the received word $Y = [Y_0, Y_1, \ldots, Y_{N-1}]$. The messages at variable nodes are initialized with intrinsic messages (η_i) which are defined as:

$$\eta_i = \log \frac{p(c_i=1)}{p(c_i=0)} = 2Y_i/\sigma^2 \qquad (1)$$

2. The outgoing messages from each check node are computed by using the incoming messages from the neighbouring variable nodes except the destination one. The number of neighbouring variable nodes is equal to the column weight of that particular row.

The check to variable message is computed as

$$\gamma_{mn} = \left\{\prod_{n' \epsilon N(m), m \neq n} sign(\beta_{n'm})\right\} \underbrace{Min}_{n'} |\beta_{n'm}| \quad (2)$$

Where γ_{mn} represents outgoing message from the check node 'm' to the neighbouring variable node 'n', while $\beta_{n'm}$ represents the incoming messages gathered at the check node from its neighbouring variable nodes except the node n, in the Tanner graph. In the first round of the iteration, the intrinsic messages are used as messages from the variable node.

3. Similarly, variable nodes perform the updating process by collecting messages from the connected check nodes. The message from each variable node is updated as follows:

$$\beta_{mn} = \eta_n + \sum_{m' \epsilon M(n), m' \neq m} \gamma_{m'n} \qquad (3)$$

The outgoing messages β_{mn} of a variable node are updated by using addition of the intrinsic message η_n and the incoming extrinsic messages $\gamma_{m'n}$ which traversed from its neighbouring check nodes M(n) except the target one.

4. After one round of check node and variable node updates,(step2 and step3), the termination condition (the maximum number of iterations) should be checked. If the termination condition is not satisfied, the decoding operation would be performed iteratively until the termination condition is met. If the termination condition is met, the reliability message is computed as:

$$L_n = \eta_n + \sum \gamma_{mn} \qquad (4)$$

Based on the reliability information, the code word bit C_n is estimated as

$$C_n = \begin{cases} 0, & x \leq 0 \\ 1, & x > 0 \end{cases} \qquad (5)$$

3. ARCHITECTURE OF THE PROPOSED DECODER

In the two phase iterative message passing LDPC decoding algorithm, the check node messages are updated in the first phase and variable node messages are updated in the second phase. These two phases are repeated for several iterations till the termination condition is met. The variable node message updating phase depends on the updated messages from check node processing unit and vice versa. So, the variable node updating process starts only after the completion of the check node updating phase as shown in Fig.4.

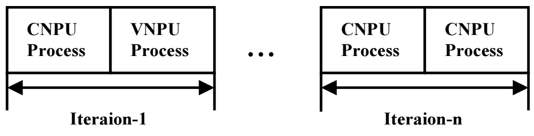

Figure 4: Non Overlapped Decoding phases

If each phase requires 'n' clock cycles, '2n' clock cycles are required to complete one iteration. But, if these two phases of the decoding process are overlapped, the number of clock cycles required for decoding can be reduced and hardware utilization is also improved. The overlapping is done by determining the data dependencies between the rows and columns.

The proposed architecture is designed for 12 x 24 H_b base matrix(rate ½). The architecture is designed to support the code lengths 576 bits (Z = 24) and 2304 bits (Z = 96) and ½ code rate. Since, the given 'H' matrix has different row and column weights, the architecture is designed for irregular QC–LDPC codes. Based on the dual diagonal structure of the base matrix, the variable node process is partially overlapped with check node process and hardware utilization is improved. The overlapping of two phases is shown in Fig.5. The throughput of the proposed decoder is compared with the non-overlapped decoder and existing decoders and throughput of the decoder is given by,

$$Throughput = \frac{Rate \times N \times Max.\ Frequency}{No.of\ clock\ cycles\ per\ iteration \times N_T} \quad (6)$$

where N_T is total number of iterations.

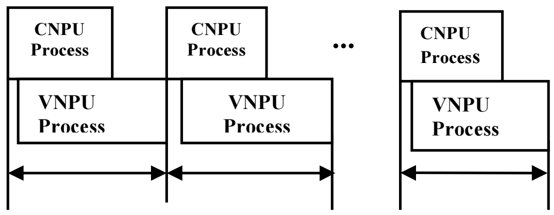

Figure 5: Overlapped Decoding phases

The block diagram of the architecture of overlapped decoder is shown in Figure 6 in Appendix- I. The iterative decoding process for one iteration is carried in two phases: 1. Check node computation 2. Variable node computation. Each row of the matrix corresponds to a check node and each column corresponds to a variable node. In the check node computation phase, the check node processing units compute the extrinsic messages for variable nodes as defined in equation(2). In the variable node computation phase, the VNPU computes the messages for check nodes as defined in equation(3). This process of updating check and variable nodes is continued to a maximum number of iterations defined. At the end of the final iteration, the VNPU computes the message as in equation (4). The code word bit is estimated as given in equation (5) at the end of final iteration.

There are two different check node processing units based on the row weights.CNPU_6 which finds minimum, sub minimum and index of the minimum of messages from 6 neighbouring nodes (i.e. row weight is 6) and CNPU_7 which finds minimum, sub minimum and index of the minimum of messages from 7 neighbouring nodes (i.e. row weight is 7). In the base matrix, four rows have row weight of 7 and eight rows have row weight of 6. So, there are 8 numbers of CNPU_7 and 16 numbers of CNPU_6.

There are three different functional units for variable node processing based on the different column weights: VNPU_3 which adds two check to variable node updates from two neighbouring nodes and intrinsic value of that node (i.e. if column weight is 2). VNPU_4 which adds three check to variable node updates from three neighbouring nodes and intrinsic value of that node (i.e. if column weight is 3). VNPU_7 which adds six check to variable node updates from six neighbouring nodes and intrinsic value of that node (i.e. if column weight is 6).

The partly parallel architecture has a $2 \times m_b(24)$ Check Node Processor units (CNPU) which work in parallel. Each block row having 'z' sub rows is processed by 2 CNPUs and each CNPU process z/2 rows serially. Hence, at the end of z/2 clock cycles all the rows ($m_b \times z$) of the 'H' matrix are processed. In the non overlapped decoder, the variable nodes are updated after the completion of processing all the rows. But, in the overlapped decoder, the check and variable node updating phases are partially overlapped. At the end of first clock cycle, the updated check node messages(check node messages) for the first and thirteenth sub columns of block columns 14 to 24 are ready to process by VNPUs. So, at the second clock cycle, the CNPUs process next subsequent rows and VNPUs process the first and 13th columns of block columns 14 to 24. At z/2th clock cycle, the CNPUs complete the processing all the rows. At (z/2 + 1)th clock cycle, all the column updates for block columns 14 to 24 are ready. The processing of block columns 1 to 13 starts at (z/2 + 1)th clock cycle. At (z+1)th clock cycle all the columns are processed and check node updating phase of next iteration starts. The process continues until the maximum number of iterations specified.

To process 'N' columns in z/2 clock cycles, 2 × n_b (48) VNPUs are required. Since irregular codes are used , three different variable node processing units are required (22 VNPU_3s, 10 VNPU_7s and 16 VNPU_4s). Two VNPUs are assigned to process each block column and each VNPU process z/2 columns serially. Hence, 2 × 11(22) VNPUs are required to process the block columns 14 to 24. These 22 VNPUs perform their operation in parallel with 24 CNPUs in the first phase of the decoding cycle. In the second phase, block columns 1 to 13 are to be processed. To process these columns 26 VNPUs are required. The 22 VNPUs assigned for block columns 14 to 24 will be idle in the second phase. So, to improve the hardware utilization, these 22 VNPUs are used in the second phase also.

The number of check node messages required for block columns 14 to 24 are 2. But, in block columns 1 to 13, the number of check node messages required are 6 and 3. Hence, different functional units are required for VNPUs to process 6 messages and 3 messages and 2 messages. The three different VNPUs are designated as VNPU7, VNPU4 and VNPU3 respectively. The number of VNPU3s to process block columns 14 to 24 are 22. These 22 VNPU3s are configured to function as 7 VNPU7s in the second phase. Additionally, 16 VNPU4s and 3 VNPU7 are required to process the block columns 1 to 13. So, a total of 27 VNPUs (7 VNPU3-7s, 16 VNPU4s and 3 VNPU7s and 1 VNPU3s) are required.

The architecture has m_b (24) numbers of CNURAM-A memory units, m_b number of CNURAM_B memory units, m_b number of VNURAMS which stores messages for CNPUs and VNPUs, m_b Index ROM memories to store column addresses and 2 VNUROM memories to store LLR values.

The messages required for CNPUs are stored in CNURAM_A memory banks and CNURAM_B memory banks. The non zero entries of columns 1-13 in each row of a block row are stored in CNURAMA bank. The non zero entries of columns 14-24 are stored in CNURAMB. For each block row, there is one CNURAMA and one CNURAMB. The inputs for all 24 CNPUs are accessed at the same time and processed concurrently. The rows of each block row are processed sequentially. At z/2 clock cycles all the rows are processed by 24 CNPUs and the outputs of the CNPUs (for columns 1 to 313) are stored in the same memory locations of VNURAM_As and outputs of the CNPUs(for columns 313 to 576) are stored in the same memory locations of CNURAM_Bs. The VNPUs process the data column wise. The column addresses are stored in INDEXROM. The output lines of the INDEXROM are connected to the read address lines of the VNURAM. There is one INDEXROM for each VNURAM. The data to be processed by VNPUs are read from VNURAMs. The processed data is stored in CNURAMs. CNURAMs and VNURAMs have dual ports.

To increase the throughput, the frequency of operation is to be increased. The frequency of operation is increased by introducing pipeline stages in the check and variable node process.

3.3 Check Node Processing Unit (CNPU)

The check node to variable node messages are computed in check node processing units. The architecture of check node processing unit is shown in Figure 7 in Appendix I. The inputs to the CNPU are stored in CNURAM_A and CNURAM_B in the sign magnitude form. The messages are sent to the CNPUs simultaneously. The CNPU computes the minimum, sub minimum and index of the minimum of the absolute values of the incoming messages from the neighbouring variable nodes. The XOR logic in the CNPU computes the products of all signs and signs of all individual messages. The minimum, sub minimum and index of the minimum, product of all signs and signs of all individual messages are given to the data distributor. The sub minimum value is used as the magnitude of the updating message of the neighbouring node when the index of the minimum equals the index of the neighbouring node and all other neighbouring nodes are assigned the minimum value. Then the output in the sign magnitude form is converted into the two's complement form. To reduce critical path delay, pipeline registers are introduced in CNPU. In Figure 10, CNPU computes minimum, subminimum and index of minimum for 4 incoming messages and the same logic is used for 6 and 7 messages. The first phase of decoding process(check node phase) is shown in Fig.8.

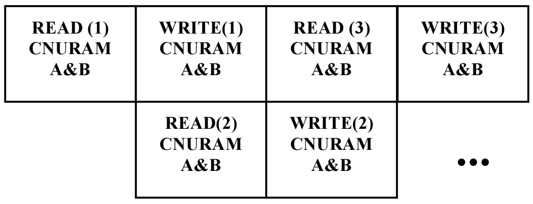

Figure 8: Check node process

3.4 Variable Node Processing Unit(VNPU)

The variable node processing unit(VNPU) is shown in Fig. 9 which adds the extrinsic messages from the neighbouring check nodes connected to the variable node and intrinsic message of that variable node as given in the equation (3). The row weights of the 'H' matrix are 2, 3 and 6. So, three different VNPU functional units are required to add 3 inputs, 4 inputs and 7 inputs. The configurable variable node processing is shown in Figure 10 in Appendix-I. The second phase of decoding process(variable node phase) is shown in Figure 11 in Appendix-I.

4. FPGA IMPLEMENTATION RESULTS

The proposed architecture is modeled in VHDL and simulated using Modelsim. The decoder is synthesized and implemented using Xilinx ISE version 13.2 software. The target FPGA is Xilinx Virtex V XC5VLX-110. The decoder is designed to support the code lengths 576 bits and 2304 bits at code rate ½ of WiMAX IEEE802.16e standard. The maximum number of iterations is set at 10.

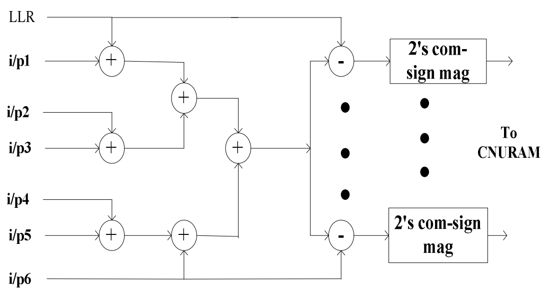

Figure 9: Variable node processing unit

The messages are quantized to 6 bits. The comparison of performance and device utilization of proposed architectures with the exiting architectures is shown in Table 1. The performance is compared with multi rate decoder[24]for irregular codes of WiMAX IEEE802.16e standard which uses layered decoding algorithm and has 48 parallel functional units. The throughput of the proposed architecture is 23.7% higher than the multi rate decoder. Since, distributed RAMs are used in the proposed architecture, larger number of logic slices are utilized in comparison with the other architectures. If block RAMs are used, number of slices will be reduced and the number of block RAMs will be increased. The throughput of the proposed architectures is less in comparison with the decoder presented in[25], because the number of parallel CNPUs are lesser in the proposed architecture. Hence, the number of clock cycles required for check node process in the proposed architecture is higher in comparison with the architecture[25]. The throughput of the proposed architectures is much higher than the decoder[26] which supports regular QC-LDPC codes of WLAN. The performance comparison of proposed architecture for code lengths 576 bits and 2304 bits is shown Table 2. The decoder with configurable VNPU has higher throughput for code length of 2304 bits. The throughput of proposed architecture is increased by 5.2% in comparison with non overlapped architecture for 2304 bits.

5. CONCLUSION

In this paper, we proposed a partially parallel decoder architecture in which check and variable node processes are partially overlapped . The throughput of the decoder with configurable VNPU is higher than the non overlapped decoder. A reconfigurable variable node unit (using multiplexers and de-multiplexers), a novel idea is introduced in the decoder architecture. Our implementations shows that the throughput of the proposed architecture is higher. The number of logic slices utilized are larger in comparison with the other architectures, because distributed RAMs are used to store the messages for CNPU and VNPU. The proposed architecture can be configured to adopt various code lengths of WiMAX IEEE 802.16e standard(576 bits to 2304 bits in steps of 24) .

REFERENCES:

[1] Gallager. R, "Low - density parity – check Codes " , IRE Trans. on Information Theory vol. 8, 1962, pp. 21– 28.

[2] MacKay. D, & Neal. R, "Near Shannon limit Performance of low density parity Check codes", in Electronic Letters vol. 32, 1996, pp. 1645–1646.

[3] Chung, S., Forney, G., Richardson, T., & Urbanke, R., "On the design of low- density parity- check codes within 0.0045 dB of

the Shannon limit", IEEE Communication Letters 5 2001, pp. 58–60.

[4] R.M.Tanner, "A recursive approach to low Complexity codes",IEEE Transactions on Information Theory, IT-27, 1981, pp. 533-547.

[5] Chung, S., Richardson, T., & Urbanke, R., "Analysis of sum-product decoding of Low – density parity-check codes using a Gaussian approximation", IEEE Trans. on Information Theory vol. 47, 2001, pp. 657–670.

[6] Jinghu Chen, Ajay Dholakia, Evangelos Eleftheriou, Marc P. C. Fossorier and Xiao – Yu Hu, "Reduced - Complexity Decoding of LDPC codes", IEEE Transactions on Communications, vol. 53, 2005, pp. 1288 – 1299.

[7] H. Zhong and T. Zhang, "Design of VLSI implementation oriented LDPC codes", IEEE 58th Vehicular Technology Conf. vol. 1, 2003, pp. 670-673.

[8] A. Blanksby and C. Howland, "A 690 – mW 1-Gbps 1024-b, rate - 1/2 low-density parity-Check code decoder", IEEE J. Solid State Circuits vol. 37 2002, pp. 404–412.

[9] Naoya Onizawa, Takahiro Hanyu and Vincent C. Gaudet, "Design of High- Throughput Fully Parallel LDPC Decoders Based on Wire Partitioning", IEEE Transactions. on Very Large Scale Integration (VLSI) Systems vol. 18, 2010, pp. 482-489.

[10] A. Darabiha, A. C. Carusone and F. R. Kschischang, "A 3.3 – Gbps bit – serial Block interlaced min-sum LDPC decoder in 0.13µm CMOS", IEEE CICC 2007, pp. 459-462.

[11] Tinoosh Mohsenin, Dean N.Truong and Bevan M.Bass, "A Low Complexity Message Passing Algorithm for Reduced Routing Congestion in LDPC Decoders", IEEE Transactions On Circuits and Systems-I: Regular papers vol. 57, 2010, pp. 1048-1061.

[12] T. Zhang and K. K. Parhi, "A 54Mbps (3,6) – regular FPGA LDPC decoder", IEEE SIPS, 2002, pp. 127-132.

[13] Y. Chen and D. Hocevar, "A FPGA and ASIC implementation of rate ½ 8088-b irregular low density parity check decoder", IEEE lobal Telecom Conference 2003, pp.113–117.

[14] M. Mansour and N. Shanbhag, "Low power VLSI decoder Architectures for LDPC codes", Int. Symposium on Low Power Electronics and Design, 2002, pp. 284-289.

[15] M. Karkooti and J.R. Cavallaro, "Semi-Parallel reconfigurable architectures for Real - time LDPC decoding", IEEE International Conference on Information Technology and computing ITCC 2004, pp. 579–585.

[16] Zhongfeng Wang, Zhiqiang Cui, "Low Complexity High Speed Decoder Design for Quasi Cyclic LDPC codes", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 15, 2007, pp.104-114.

[17] Yongmei Dai, Zhiyuan Yan, and Ning Chen, "Optimal Overlapped Message Passing Decoding of Quasi – Cyclic LDPC Codes", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 16 2008, pp. 565-578 .

[18] Yongmei Dai, Ning Chen and Zhiyuan Yan, "Memory Efficient Decoder Architectures for Quasi-Cyclic LDPC Codes", IEEE Trans. On Circuits and Systems-I: Regular papers, vol.55 2008, pp. 2898-2911.

[19] Daesun Oh and K. K. Parthi , " Min-Sum Decoder Architectures with reduced word Length for LDPC codes", IEEE Trans. On Circuits and Systems-I: Regular papers, vol.57 2010, pp. 105-115.

[20] Xiaoheng, Jingyu Kang, Shu Lin and Venkatesh Akella, "Memory System Optimization for FPGA - based Implementation of Quasi-Cyclic LDPC codes Decoders",IEEE Transactions On Circuits and Systems-I: Regular papers, vol.58, 2011, pp.98-111.

[21] Marjan Karkooti, Predrag Radosavljevic, Joseph R. Cavallaro, "Configurable LDPC Decoder Architectures for Regular and Irregular Codes", J. Sign Process Syst vol. 53, 2008, pp. 73-88.

[22] Xiaoheng Chen, Shu Lin ,Life Fellow, IEEE, and Venkatesh Akella, "QSN— A Simple Circular – Shift Network for Reconfigurable Quasi – Cyclic LDPC Decoders", IEEE Transactions on Circuits and Systems—II: express briefs, vol. 57, 2010, pp.782 – 786.

[23] Yong Ki Byun, Jong Kang Park, Soongyu Kwon, and Jong Tae Kim, "An Efficient Overlapped LDPC Decoder with Upper Dual - diagonal Structure", Journal of Semi Conductor Technology and Science, 2013, pp. 8-14.

[24] Kiran K. Gunnam, Gwan S. Choi, Mark B. Yeary and Mohammed Atiquzzaman, "VLSI Architectures for Layered Decoding for Irregular LDPC codes of WiMAX", Proc.

IEEE International Conference on Communication(ICC) , 2007, pp.4542-4547.

[25] Vikram Arkalgu Chandrasetty and Syed Mahfuzul Aziz, "Resource Efficient LDPC Decoders for Mutimedia Communication", Electronic edition @aXiv.org:1305.6216.

[26] Vikram Arkalgu Chandrasetty and Syed Mahfuzul Aziz, "A Highly Flexible LDPC Decoder using Hierarchical Quasi - Cyclic Matrix with Layered Permutation", Journal of Networks vol. 7, 2012, pp. 441- 449.

APPENDIX –I

$$H_b = \begin{bmatrix}
-1 & 94 & 73 & -1 & -1 & -1 & -1 & -1 & 55 & 83 & -1 & -1 & 7 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & 27 & -1 & -1 & -1 & 22 & 79 & 9 & -1 & -1 & -1 & 12 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & 24 & 22 & 81 & -1 & 33 & -1 & -1 & -1 & 0 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
61 & -1 & 47 & -1 & -1 & -1 & -1 & -1 & 65 & 25 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & 39 & -1 & -1 & -1 & 84 & -1 & -1 & 41 & 72 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & -1 & -1 & 46 & 40 & -1 & 82 & -1 & -1 & -1 & 79 & 0 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 & -1 \\
-1 & -1 & 95 & 53 & -1 & -1 & -1 & -1 & -1 & 14 & 18 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 & -1 \\
-1 & 11 & 73 & -1 & -1 & -1 & 2 & -1 & -1 & 47 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\
12 & -1 & -1 & -1 & 83 & 24 & -1 & 43 & -1 & -1 & -1 & 51 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 & -1 \\
-1 & -1 & -1 & -1 & -1 & 94 & -1 & 59 & -1 & -1 & 70 & 72 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & -1 \\
-1 & -1 & 7 & 65 & -1 & -1 & -1 & -1 & 39 & 49 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 \\
43 & -1 & -1 & -1 & -1 & 66 & -1 & 41 & -1 & -1 & -1 & 26 & 7 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 0
\end{bmatrix}$$

Figure 3: Base Matrix 'H_b' for IEEE 802.16e standard for code rate ½

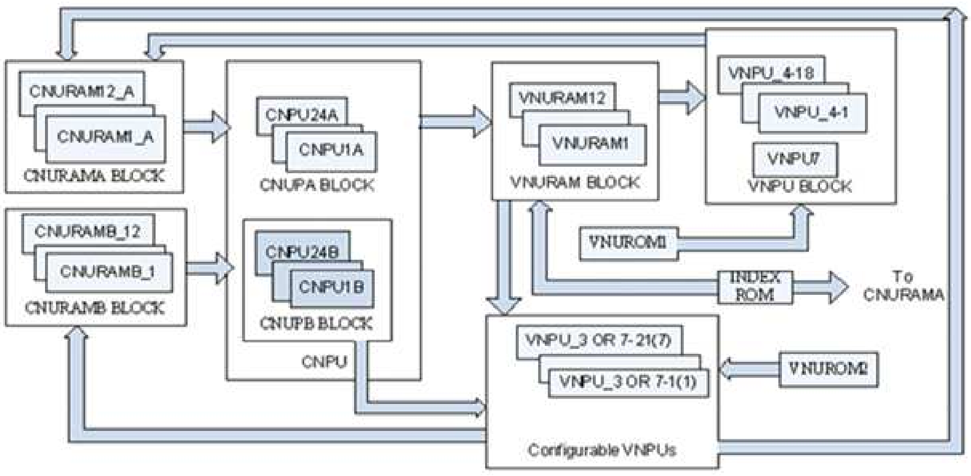

Figure 6: Proposed LDPC Decoder with Reconfigurable VNPU

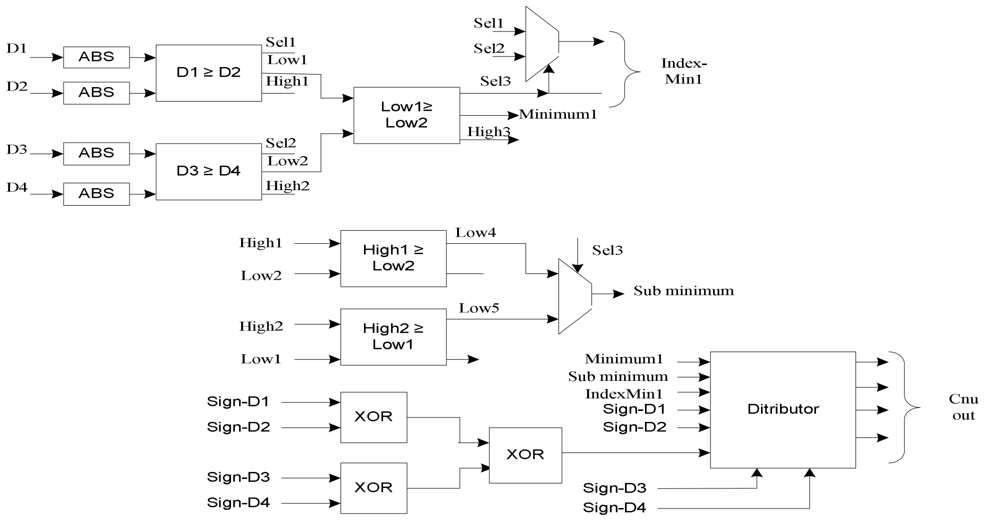

Figure 7: Check Node Processing Unit

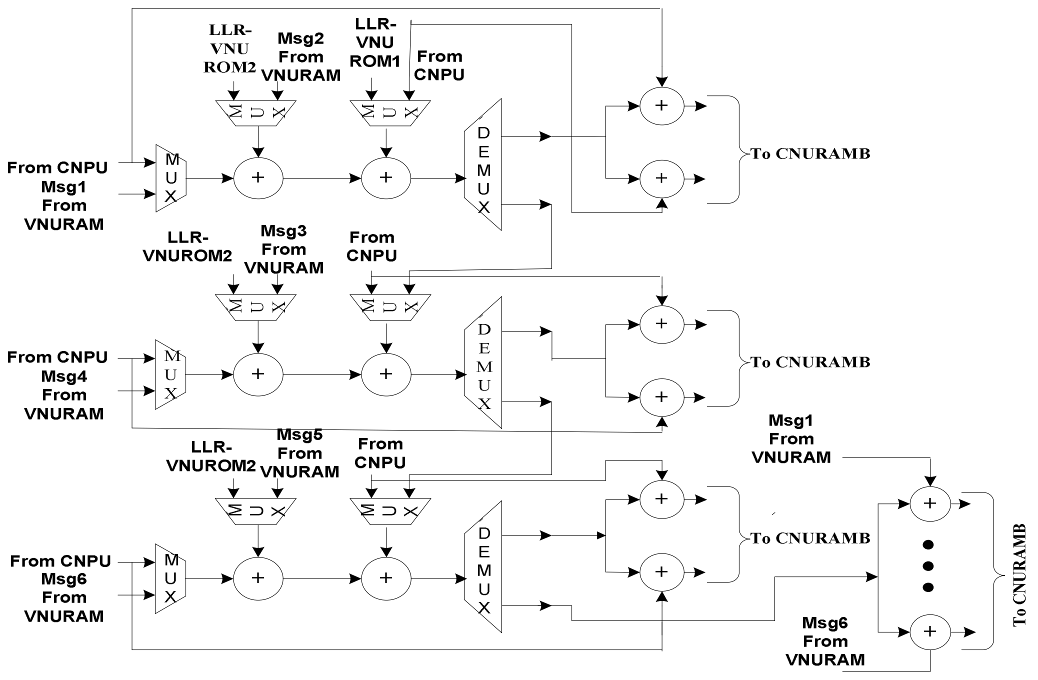

Figure 10: Reconfigurable variable node processing unit

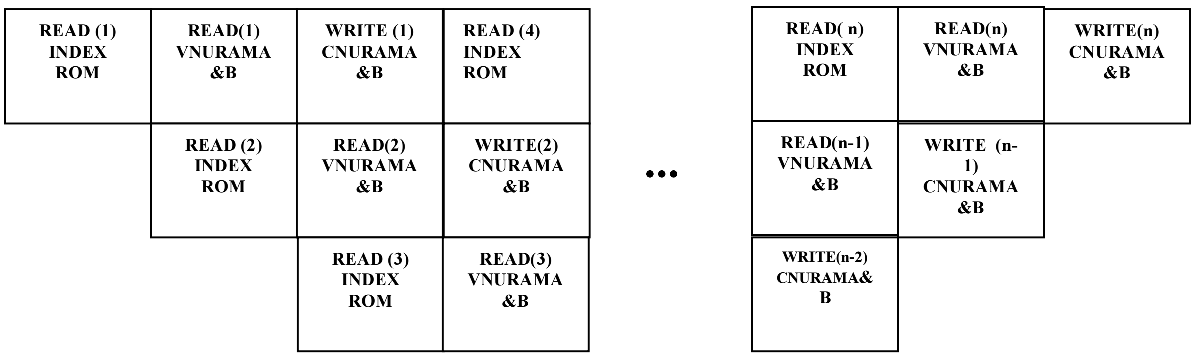

Figure 11: Variable node phase

Table 1: Comparison of Performance and resource utilization with existing architectures

	Decoder Without overlap	Proposed Architecture	[25]	[26]	[24]
Code length/Code rate	2304 / 1/2 WiMAX IEEE806.16e		2304/ 1/2	648,1296, 1944 /1/2 (WLAN)	2304 Multiple rates
Code Type	QC Irregular	QC Irregular	QC Regular	QC Regular	QC Irregular
Algorithm	Min-Sum	Min-Sum	Modified Min-Sum	Min-sum	TDMP
Parallelism	24 CNUs 48 VNUs	24 CNUs 48 VNUs	48 CNUs 48 VNUs	18 CNUs 18 VNUs	48
Slices	3754	5272	3141	5908	3239
Flip Flops	34670	36271	2024	10816	3165
LUTs	20391	19437	9547	1604	5664
Block RAMs	20	18	87	47	73
Frequency (MHz)	136.835	145.229	144	128	110
Throughput (Mbps)	163.5	172	266	54,45,32	57~139
Target FPGA	Xilinx Virtex V XC5VLX-110		Xilinx Virtex V XC5VLX-110T	Xilinx Virtex II XC2V8000-5FF152	

Table 2: Comparison of Proposed LDPC Decoders with N = 576 and N = 2304

	Decoder Without overlap		Proposed Architecture	
Code length	576	2304	576	2304
Slices	4096	3754	4976	5272
Slice Flip Flops	11157	34670	13110	36271
LUTs	11454	20391	10542	19437
Block RAMs	8	20	9	18
Frequency (MHz)	133.082	136.835	145.229	145.229
Throughput (Mbps)	157	163.5	164.5	172