2010 3rd International Conference on Computer and Electrical Engineering
(ICCEE 2010)
IPCSIT vol. 53 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V53.No.1.66

QC-LDPC Encoding and Decoding Algorithms for G.hn Standard

He Yi[+], Yao Rugui and Wang Ling

School of Electronics and Information
Northwestern Polytechnical University
xi'an, China

Abstract. G.hn is a standard which is recommended by ITU-T for next-generation wire-line based home networking. The transceivers use OFDM type of modulation, and support a great variety of multimedia technology. This paper presents the algorithms of QC-LDPC encoder and decoder according to the G.hn standard. A recursion encoding algorithm with low complexity is adopted; besides, a simple and practicable hardware encoder implementation is presented. As for decoding, we analyze a newly-developed Parallel Turbo-Decoding Message-Passing (P-TDMP) algorithm. Simulation results demonstrate that P-TDMP decoding algorithm has a performance advantage over classic TPMP decoding algorithm in G.hn. what's more, it improves the decoding throughput and cuts down the overhead greatly which has bright prospect in applications.

Keywords: G.hn; QC-LDPC; recursion encoding; P-TDMP

1. Introduction

LDPC code (low density parity check code) is a kind of linear block code which can be fully described by its parity-check matrix H. It was first introduced by R. G. Gallager in 1963 [1]. But as the high complexity of encoding and decoding, it didn't get much attention. In 1993, C. Berrou proposed Turbo code, and found new way to study cascade code. Based on C. Berrou's work, R. M. Neal and D. MacKay restudied LDPC code and discovered its excellent performance: close to Shannon limits, flexible structure, low error platform, high throughput, easy for hardware implementation. In recent years, LDPC code has received a great deal of interest and been widely used, for instance, it is used as channel coding in DVB-S2 [2], IEEE802.11n [3], IEEE802.16e [4], and CMMB [5] etc.

Quasi-cyclic LDPC (QC-LDPC) code is a particularly important class of LDPC code. It has been constructed based on circulant permutation matrix which is an identity matrix or a rotated identity matrix. The greatest advantage of QC-LDPC code is that it involves much easier implementation in both encoding and decoding procedure; especially the overhead of hardware implementation can be much smaller.

ITU-T Recommendation G.hn standard specifies basic characteristics of next generation home networking transceivers designed for the transmission of data over in-premises networks operating over phone line, power line or coax [6]. The transceivers use OFDM modulation and are designed to provide compatibility with many types of access technologies as well as multimedia technologies.

In this paper, a thorough study is carried out in terms of QC-LDPC encoding and decoding schemes according to the G.hn standard, besides, the encoding and decoding simulations have been done. The corresponding performance evaluation results show that the encoding and decoding algorithms adopted in this paper are excellent.

[+] Corresponding author.
E-mail address: heyi@mail.nwpu.edu.cn.

The remainder of the paper is organized as follows. Section II describes the feature of QC-LDPC in G.hn standard. Section III introduces the QC-LDPC encoding algorithm and analyses the hardware implementation schemes in G.hn. Section IV addresses the QC-LDPC decoding algorithm in G.hn. The decoding performance simulation results are presented in Section V, and the maximum decoding iterations as well as appropriate quantization bits are discussed through simulation. Concluding remarks follows in Section VI.

2. The Feature of QC-ldpc in G.HN

2.1 The construction of parity check matrix H in G.hn

The QC-LDPC encoder in G.hn supports mother codes with rates 1/2, 2/3 and 5/6. From these mother codes, codes with higher coding rates (16/18 and 20/21) shall be obtained through puncturing. The parity check matrix H are composed by an array of $c \times t$ circulant $b \times b$ sub-matrices $A_{i,j}$, by selecting different sets of c and t, different code rates can be obtained [6].

$$H = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,t} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,t} \\ \vdots & \vdots & \ddots & \vdots \\ A_{c,1} & A_{c,2} & \cdots & A_{c,t} \end{bmatrix} \tag{1}$$

The sub-matrices $A_{i,j}$ are either a $b \times b$ rotated identity or zero matrices. So, the parity check matrix H can be described in a compact form as follows.

$$H_c = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,t} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,t} \\ \vdots & \vdots & \ddots & \vdots \\ a_{c,1} & a_{c,2} & \cdots & a_{c,t} \end{bmatrix} \tag{2}$$

The zero sub-matrices $A_{i,j}$ are defined as $a_{i,j} = -1$. When $a_{i,j}$ is a positive integer number, $A_{i,j}$ is a rotated identity sub-matrices which right shifts $\left\lfloor a_{i,j} \cdot \frac{b}{96} \right\rfloor$ module b column of the identity matrix. ($\lfloor x \rfloor$ represents the integer part of variable x). Using the compact form, H matrix can be more concisely described.

2.2 A-A (architecture-aware) LDPC codes

A-A LDPC codes have embedded structure that enable low decoding complexity [7]. Figure 1 shows the structure of an architecture-aware low density parity check matrix H. It can be portioned into information part H^i and parity check part H^p, and both of them are composed of sub-matrices I_{pi}, where I is the $b \times b$ identity matrix and I_{pi} is I with its columns permuted according to the permutation pi. There are at most t sub-matrices in each block row and c sub-matrices in each block column. So the H matrix has a size of $(c \times b) \times (t \times b)$, and the b rows of ones in each block row are non-overlapping.

The QC-LDPC code in G.hn is A-A LDPC code. Its high-structured character involves efficient encoding and decoding implementation with good performance.

$$H_{cb \times tb} = \begin{array}{c} \overbrace{\phantom{H^i_{cb\times(t-c)b}}}^{H^i_{cb\times(t-c)b}} \quad \overbrace{\phantom{H^p_{cb\times cb}}}^{H^p_{cb\times cb}} \\ \left[\begin{array}{cc|cc|cccc} I_{p1} & I_{p2} & & & I & & & & \\ & & I_{p3} & I_{p4} & I & I & & & \\ & I_{p5} & & I_{p6} & & I & I & & \\ I_{p7} & & I_{p8} & & & & I & I & \\ I_{p9} & & & & & & & I & I \\ & I_{p10} & & I_{p11} & & & & & I & I \end{array}\right] \\ \underbrace{\phantom{\text{Information}}}_{\text{Information}} \quad \underbrace{\phantom{\text{Parity}}}_{\text{Parity}} \end{array}$$

Figure 1. The structure of A-A LDPC H

3. QC-ldpc Encoder in G.HN

3.1 General LDPC encoding algorithm

H is a classic spare matrix, only few elements of H are nonzero, however, the corresponding generation matrix is dense. Implementation the encoder through dense generation matrix shall result in large complexity. In general, the following equation can be used for encoding.

$$H \cdot v^T = 0 (\mathrm{mod}\, 2) \tag{3}$$

The parity check matrix H has a size of $(n-k)\times n$, it can be divided into H_1 and H_2. $H = [H_1 \quad H_2]$, where H_1 has a size of $(n-k)\times k$ while H_2 has a size of $(n-k)\times(n-k)$. Partition the codeword v, in the same way, into k information bits, S, and $n-k$ check bits, P. The parity check equitation(3) becomes

$$[H_1 \quad H_2][S \quad P]^T = 0 (\mathrm{mod}\, 2) \tag{4}$$

From this, we get

$$H_1 \cdot S^T + H_2 \cdot P^T = 0 (\mathrm{mod}\, 2) \tag{5}$$

And hence

$$P^T = H_2^{-1} H_1 \cdot S^T (\mathrm{mod}\, 2) \tag{6}$$

According to (6), in order to get P, $H_2^{-1}H_1$ can be pre-computed, and then multiplying the information bits S by this matrix.

As the computation of H_2^{-1} results in high complexity and the QC-LDPC code in G.hn is highly structured, some more efficient encoding algorithm for QC-LDPC in G.hn has been explored as follows.

3.2 QC-LDPC encoding algorithm in G.hn

The QC-LDPC H in G.hn can also be divided into corresponding H_1 and H_2, which are composed by an array of $c\times(t-c)$ and $c\times c$ circulant $b\times b$ sub-matrices respectively, where $b=(n-k)/c$. In G.hn, the deterministic structure of H_2 enables the encoding procedure recursively done. The first column sub-matrix h_i of H_2 is either a $b\times b$ zero matrix or a $b\times b$ rotated identity matrix, and satisfies that $\sum_{i=0}^{c-1} h_i = I_{b\times b} (\mathrm{mod}\, 2)$. The other column in H_2 each has two identity matrices. Figure 2 shows the structure of H_2.

$$\begin{bmatrix} h_0 & I & & & \\ h_1 & I & I & & \\ h_2 & & I & I & \\ \vdots & & & I & \ddots \\ \vdots & & & & \ddots & I \\ h_{c-1} & & & & & I \end{bmatrix}$$

Figure 2. Structure of H_2

h_i is $b\times b$ rotated identity matrix and I is $b\times b$ identity matrix

Based on the character of H in G.hn, we can decompose the parity check bits P into $1\times c$ array of $1\times b$ sub-matrices, and partition the incoming information bits S into $1\times(t-c)$ array of $1\times b$ sub-matrices. For calculating P, as $\sum_{i=0}^{c-1} h_i = I_{b\times b} (\mathrm{mod}\, 2)$, the rows from the second to the last are added into the first row in H_2, implement the same elementary row transformation on H_1. According to (5), $I.p_0^T = \sum_{i=0}^{c-1} H_1^{(rowi)} \cdot s^T (\mathrm{mod}\, 2)$.

Hence, the P_0 can be calculated by $p_0^T = \sum_{i=0}^{c-1} H_1^{(rowi)} \cdot s^T (\mathrm{mod}\, 2)$. Then, through simple derivation, $P_{i,(i=1,\cdots,c-1)}$ can be found recursively as follows.

$$p_1^T = H_1^{(row0)} \cdot s^T + h_0 \cdot p_0^T (\mathrm{mod}\, 2)$$

$$p_2^T = H_1^{(row1)} \cdot s^T + h_1 \cdot p_0^T + p_1^T \,(\mathrm{mod}\, 2)$$

…

$$p_{c-1}^T = H_1^{(rowc-2)} \cdot s^T + h_{c-2} \cdot p_0^T + p_{c-2}^T \,(\mathrm{mod}\, 2)$$

Consider the hardware implementation, as H_1 is composed of either rotated identity sub-matrices or zero sub-matrices, we can compute $H_1^{(rowi)}s^T$ by the circuit shown in Figure 3 with a b-bit register, a b-bit XOR and a b-bit barrel shifter [8]. With all the $H_1^{(rowi)}s^T$ known, P can be determined recursively.

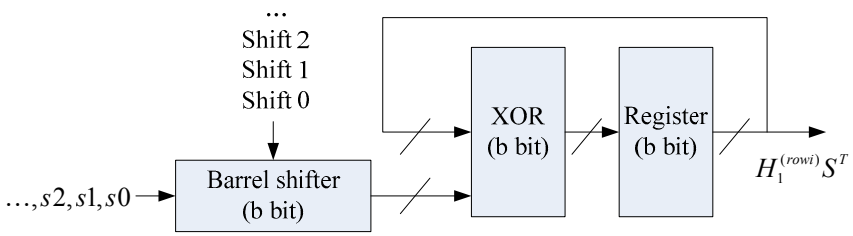

Figure 3. Circuit to compute $H_1^{(rowi)}s^T$

With this scheme, the QC-LDPC encoder in G.hn is quite easy to implement, what's more, the encoding efficiency can be greatly improved.

4. QC-ldpc Decoder in G.HN

4.1 The algorithm selection of QC-LDPC decoding in G.hn

The classic LDPC decoding algorithm is two-phase message passing (TPMP) algorithm or the so called belief propagation (BP) algorithm. The decoding procedure revolves around the two-phase transmission of extrinsic information between check nodes and bit nodes in Tanner Graph, message passing decoding algorithm utilizes the bit-to-bit dependence required in check-sum equations of LDPC code to adjust the probability of each bit until obtaining a valid codeword [9].

The newly developed LDPC decoding algorithm is turbo-decoding message passing. The TDMP algorithm outperforms the standard two-phase decoding algorithm in its faster convergence speed by roughly a factor of two in terms of decoding iterations, and in the more than 50% memory saving [10]. In TDMP scheme, both variable and check messages collapse into a single type of messages [7], hence, compared to TPMP, the TDMP algorithm complexity is lower and more suitable for hardware implementation. With so many advantages, TDMP algorithm is used in The QC-LDPC decoding in G.hn standard, it can offer a better performance-complexity tradeoff when the number of iterations is small.

The QC-LDPC code in G.hn is A-A LDPC code, in A-A LDPC H, the ones in each block row (every b rows) are not overlapped. Utilizing this character, decoding can be processed for b rows at every iteration. So, the TDMP algorithm in G.hn can be modified into a parallel version named as P-TDMP [7]. P-TDMP provides an improvement in decoding throughput over ordinary TDMP algorithm, which is attractive for high speed hardware application.

4.2 TDMP algorithm

The TDMP algorithm is proposed by Mansour, M.M, it is base on decoding the rows of H sequentially [7, 11]. In order to give a full description of algorithm, we can take a parity check matrix H shown in Figure 4 for example to give some intuitive explanations. For row i in H, I_i denotes the set of indexes of ones in row i, $\lambda^i = \lambda_1^i, \ldots, \lambda_{n_i}^i$ represents the extrinsic messages corresponding to the ones in row i, where n_i is the number of ones in row i. $\gamma(I_i)$ represents the posterior messages of row i. So, in Figure 4, for the highlighted row 2, $n_2 = 4$ while $I_2 = \{1,4,6,7\}$ and $\lambda^2 = [\lambda_1^2, \lambda_2^2, \lambda_3^2, \lambda_4^2]$. λ_1^2 corresponds to bit 1, λ_2^2 corresponds to bit 4…the extrinsic messages λ^i are associated to the extrinsic estimates about the bits from all parity check equations that these bits participate. Moreover, let $\gamma = [\gamma_1, \cdots, \gamma_N]$ denote posterior messages that stores the sum of all

messages generated by the rows in which each bit participates, as in Figure 4, $N = 8$, $\gamma_1 = \lambda_1^2 + \lambda_1^3$, $\gamma_2 = \lambda_1^1 + \lambda_2^3 \ldots$

$$\begin{array}{c} \\ \lambda^1 \\ \lambda^2 \\ \lambda^3 \\ \lambda^4 \end{array} \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Figure 4. Parity check matrix H

Decoding row i can be implemented as the follows [7], The 4 steps constitute a decoding subiteration.

1) *read: λ^i and $\gamma(I_i)$ are read for row i*
2) *subtract: λ^i are subtracted from $\gamma(I_i)$ to generate prior message $\rho = [\rho_1, \ldots, \rho_{n_i}]$*
3) *decode: Decode row i using a SISO algorithm [7,11,12] with ρ as input and $\Lambda^i = [\Lambda_1^i, \ldots, \Lambda_{n_i}^i]$ as output*
4) *write back: Replace the original extrinsic message λ^i with Λ^i, and update $\gamma(I_i)$ by $\gamma(I_i) = \rho + \lambda^i$*

The whole decoding procedure is made up of multiple subiterations. A decoding iteration is completed when the subiterations for all rows of H are over.

The TDMP algorithm utilizes the most recent updated message directly within an iteration to compute a new message, which speeds up the convergence speed in terms of iteration. Furthermore, the two phase message computations are substituted by a single type of computation, hence, the memory advantages is significant. Besides, the throughput and improvement in coding gain over the TPMP algorithm is obvious [11].

4.3 SISO decoding message computation

SISO (soft input soft output) algorithm is adopted in message computation of TDMP. Unlike other schemes as $\Lambda_j^i = \psi^{-1}(\sum_{l:l \neq j} \psi(\rho_l)), i = 1, \ldots, n; j = 1, \ldots, n_i$ [7], SISO decoder needn't use lookup tables, and can be implemented using simple logic gates. Moreover, the SISO circuit can be designed into the form of dual stack, resulting a memory saving and efficiency improvement for hardware implementation [12].

The description of SISO algorithm [7] is listed as below.

$\Lambda = SISO(\rho)$
//Input: $\rho = [\rho_1, \rho_2, \ldots, \rho_c]$
//Output: $\Lambda = [\Lambda_1, \Lambda_2, \ldots, \Lambda_c]$
//Initializations
$\alpha_1 \leftarrow \rho_1$
$\beta_c \leftarrow \rho_c$
//Forward-backward recursion
for $i = 1$ to $c - 2$ do
  $j = c - (i - 1)$
  $\alpha_{i+1} \leftarrow Q(\alpha_i, \rho_{i+1})$
  $\beta_{j-1} \leftarrow Q(\rho_{j-1}, \beta_j)$
  if $c/2 \le i \le c - 2$ then
    $\Lambda_{i+1} \leftarrow Q(\alpha_i, \beta_{i+2})$
    $\Lambda_{j-1} \leftarrow Q(\alpha_{j-2}, \beta_j)$
  end if
end for
$\Lambda_1 \leftarrow \beta_2$
$\Lambda_c \leftarrow \alpha_{c-1}$

Where $Q(x, y) = \max(x, y) + \max(5/8 - |x - y|/4, 0)$
$- \max(x + y, 0) - \max(5/8 - |x + y|/4, 0)$ [12]

The SISO message computation process starts with the initialization of α_1 and β_c with ρ_1 and ρ_c respectively, assuming c is even, then the forward and backward recursion can be handled at the

same time. For forward recursion, $Q(\rho_1,\rho_2)=\alpha_2$, then $Q(Q(\rho_1,\rho_2),\rho_3)=Q(\alpha_2,\rho_3)=\alpha_3$ … until $Q(\alpha_{j-2},\rho_{j-1})=\alpha_{j-1}$ .while in the backward, first $Q(\rho_{n-1},\rho_n)=\beta_{n-1}$,when β_{n-1} is known ,the recursion continues $Q(\rho_{n-2},Q(\rho_{n-1},\rho_n))=Q(\rho_{n-2},\beta_{n-1})=\beta_{n-2}$ …up to $Q(\rho_{j+1},\beta_{j+2})=\beta_{j+1}$ .when α_{j-1} and β_{j+1} are known, the first output $\Lambda_j=Q(\alpha_{j-1},\beta_{j+1})$ can be produced. Then, with the forward and backward recursion updating, the output messages can be carried out in pairs. Finally, β_2 and α_{c-1} are assigned to the leftmost output Λ_1 and the rightmost output Λ_c respectively. We can illustrate the SISO algorithm in [7], assuming c is 6. There will be some minor modifications in case c is odd.

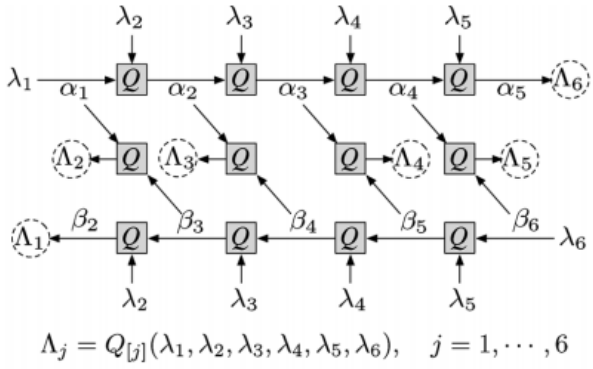

Figure 5. SISO algorithm

4.4 P-TDMP algorithm

As described in section II, the QC-LDPC in G.hn standard is A-A LDPC code, hence, the parallel version turbo decoding message passing (P-TDMP) algorithm is adopted. The detail of P-TDMP algorithm [7] is shown as follows.

$\hat{u}=P-TDMP(H,\delta,c,b,T)$
//Input: parity check matrix H
//Input: channel value $\delta=[\delta_{u1.,},...,\delta_{uk}]$
//Input: c =The number of submatrices contained in each block column of H
//Input: b =the dimension of submatrices I_{pi}
//Input: Iterations T
//Output: hard decision estimates $\hat{u}$
//Storage: $c\times b$ extrinsic message vectors $\lambda^{i,j}$
//Storage: vector of posterior messages γ
//Initialization
$\lambda^{i,j}\leftarrow 0, i=1,..c, j=1,...,b$
$\gamma\leftarrow\delta$
// Iterative decoding
for t=1 to T do
  for $i=1$ to c do for all j
    $\rho^j\leftarrow\gamma(I_{(i-1)b+j})-\lambda^{i,j}$ // read and subtract
    $\Lambda^j\leftarrow SISO(\rho^j)$ // decode
    $\lambda^{i,j}\leftarrow\Lambda^j$ // write back
    $\gamma(I_{(i-1)b+j})\leftarrow\rho^j+\Lambda^j$ // write back
  end for
end for
//Hard decisions
$\hat{u}\leftarrow\frac{1}{2}(\mathrm{sgn}(\gamma_j)+1), j=1,...,k$

The P-TDMP algorithm is based on that the ones in every b rows of submatrices are not overlapped, so the decoding procedure can be handled in parallel using b SISO decoders. The b decoders constitute a decoder group and work simultaneously. Decoder i processes row i in each submatrices and maintains the extrinsic messages $\lambda^{i,j}$ in local memory, while posterior messages are passed to the decoders, b messages at a time, from a global memory. The decoding procedure runs over the rows of H in c iterations, and processing b rows simultaneously in every iteration which improves the decoding throughput by a factor of b over TDMP algorithm.

5. Simulation Results

5.1 The BER performance comparisons of three decoding algorithm

The BER performance of P-TDMP algorithm is compared with LLR BP algorithm and UMP BP algorithm by simulating the QC-LDPC code with code length 960, code rate 1/2 and iterations 10. LLR (Log Likelihood Ratios) BP algorithm is BP algorithm in Logarithm domain, which performs additions instead of multiplications. UMP (Uniformly Most Powerful) BP algorithm is an approximation of LLR BP algorithm, in which the non-linear function $\phi(x) = 2\arctan h(\tanh(x/2))$ is approximated by a minimum function. Figure 6 shows the BER plots.

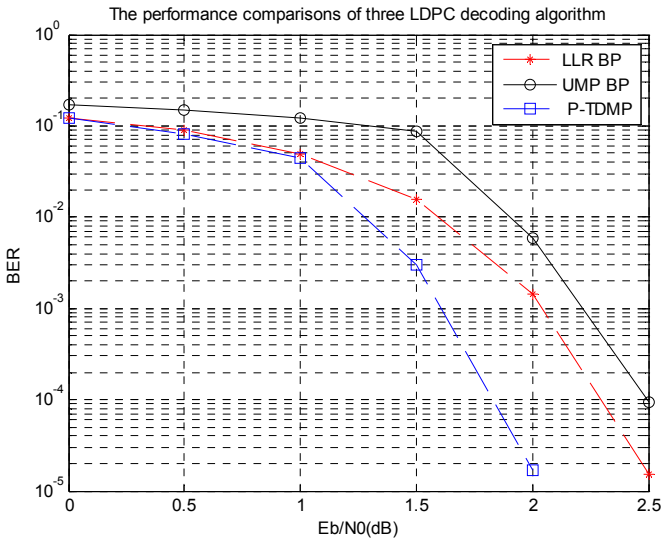

Figure 6. BER performance comparisons: P-TDMP versus LLR BP ,UMP BP(code length =960, code rate=1/2, iterations=10)

Figure 6 demonstrates that for the same iterations and at the same SNR, P-TDMP algorithm achieves much better BER than the other twos. It is quite suitable for QC-LDPC decoding in G.hn.

5.2 The influence of maximum iterations on P-TDMP decoding performance

The maximum iterations have great influence on decoding performance. Large iterations will lower down decoding speed while too small iterations results worse BER performance. Through simulation, the most appropriate iterations can be found according to various code length and code rate. Figure 7 shows the P-TDMP decoding performance of QC-LDPC code with code length 960, code rate 1/2 under iterations 5, 8, 10 respectively.

As shown below, when iterations are 8 and 10, there exists a significant performance gain over the case iteration is 5. Considering 8 and 10, 10 performs a little better, but ensues much lower decoding speed, so the optimal iterations is 8 in case that code length 960 and code rate 1/2.

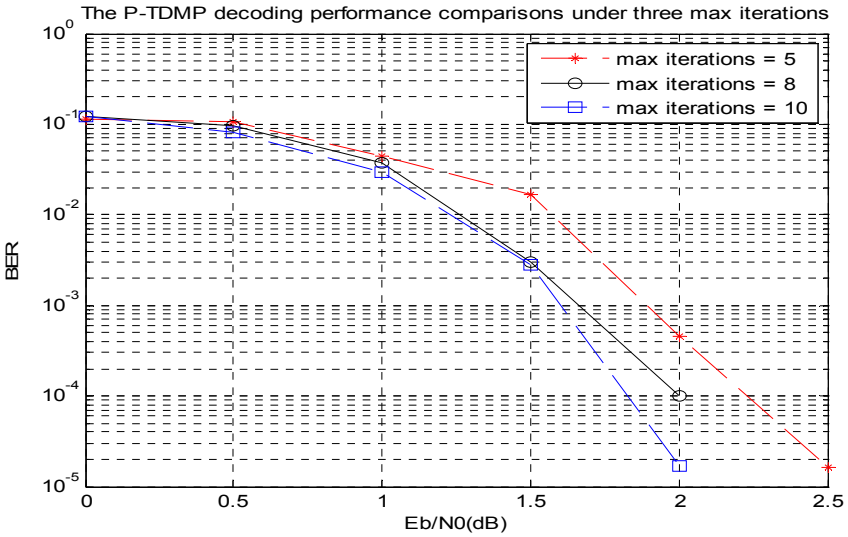

Figure 7. the comparisons of max iterations on P-TDMP decoding performance (code length =960, code rate=1/2)

5.3 The influence of quantization scheme on P-TDMP decoding performance

All the variables and constants are decimal numbers when we do simulation using MATLAB, they have infinite precision. However, the digital system is with finite precision in hardware implementation, hence, quantization should be taken into consideration. Usually, quantization scheme is determined through simulation as it is a tough problem for theoretical analysis.

In this paper, for further reduction of the overhead, different variables are quantized with different bits. Three different quantization schemes are listed in TABLE I. The quantization bits are in the form of (QI, QF), where QI and QF represent the integer part and decimal part respectively.

Table 1 Quantization schemes

	Soft input	ρ	λ	γ	SISO
Case1	(4,1)	(5,1)	(3,1)	(6,1)	(9,1)
Case2	(3,2)	(5,2)	(3,2)	(6,2)	(9,2)
Case3	(3,3)	(5,3)	(3,3)	(6,3)	(9,3)

Figure 8 plots the BER performance of QC-LDPC code under three quantization schemes with code length 960, code rate 1/2, and iteration 5. From Figure 8, we can see, case 2 offers a better performance/overhead tradeoff.

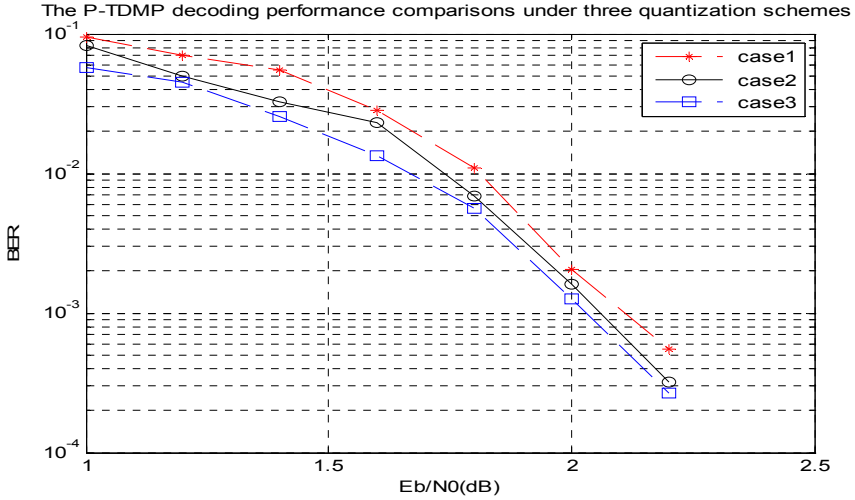

Figure 8. The comparisons of quantization schemes on P-TDMP performance(code length =960, code rate=1/2, iterations=5)

6. Concludes

In this paper, the algorithms of QC-LDPC encoder and decoder in G.hn are discussed. According to the structural character of H in G.hn, the encoder uses a recursion algorithm which has high efficiency and low complexity, the hardware implementation scheme is proposed as well. The decoder adopts the P-TDMP algorithm proposed by M.Mansour. The P-TDMP algorithm outperforms the classic BP algorithm not only in BER performance but also in throughput. Through simulation, the optimal iterations and quantization schemes for hardware implementation are explored.

In all, the recursion encoder and highly parallel decoder in G.hn have excellent performance.

7. Acknowledgements

This work is sponsored by the special assistance graduation project of Northwestern Polytechnical University, 2009. The innovation fund of science and technology of Northwestern Polytechnical University. The National Natural Science Foundation of China (No. 60802083) and the NPU Foundation for Fundamental Research (NPU-FFR-JC200817).

8. Reference

[1] R. G. Gallager, “Low-Density Parity-Check Codes,” Cambridge, MA:M.I.T. Press, 1963.

[2] Draft DVB-S2 Standard, [Online]. Available: http://www.dvb.org.

[3] IEEE 802.11n Wireless LAN Medium Access Control MAC and Physical Layer PHY specifications. IEEE 802.11n-D1.0, 2006.

[4] IEEE Std 802.16e, “Air interface for fixed and mobile broadband wireless access systems,” [Online]. Available:http://standards.ieee.org/getieee802/download/802.16e-2005.pdf.

[5] GY/T 220.1-2006, Mobile Multimedia Broadcasting Part 1: Framing Structure, Channel Coding and Modulation for Broadcasting Channel.

[6] ITU-T G.9960, Unified high-speed wire-line based home networking transceiver.

[7] Mohammad M.Mansour, “A Turbo-Decoding Message-Passing Algorithm for Sparse Parity-Check Matrix Codes,” IEEE Trans on Signal Processing, 2006, 54(11):4376-4392 .

[8] Yang Sun, Marjan Karkooti and Joseph R. Cavallaro, “ High Throughput, Parallel, Scalable LDPC Encoder/Decoder Architecture for OFDM Systems,” 2006 IEEE Dallas/CAS Workshop on Design, Applications, Integration and Software, Oct 2006, 39-42.

[9] Zhixing Yang, Nan Jiang, Kewu Peng, and Jintao Wang, “High-Throughput LDPC Decoding Architecture,” International Conference on Communication, Circuits and Systems, ICCCAS, 2008, 1240-1244.

[10] Dan Bao, Bo Xiang, Rui Shen, Zui chen, Xiaoyang Zeng, “VLSI Implementation of QC-LDPC Decoder Using Optimized TDMP Algorithm,” Journal of Computer Research and Development, 2009,46(2):338-344.

[11] Mohammad M.Mansour, Naresh R. Shanbhag, “High Throughput LDPC Decoders,” IEEE Transactions on very large scale integration systems, 2003, vol. 11, NO.6:976-996.

[12] Yuyang Zhang, Jianhao Hu, Feng Li, “A TDMP-LDPC Decoder Designed for DMB-T Standard,” China Integrated Circuit 2009,vol. 18, NO.3:26-35.