Mapper and Demapper for LDPC-Coded Systems

U.Kiran
Dept. of ECE
Vaagdevi College of Engineering, Warangal.
Email: kiranuday411@gmail.com

V.Ugendar
Dept.of ECE
Vaagdevi College of Engineering, Warangal.
Email: ugi405@gmail.com

***Abstract*—Low Density Parity Check (LDPC) codes are state-of-art error correcting codes, included in several standards for broadcast transmissions. Iterative soft-decision decoding algorithms for LDPC codes reach excellent error correction capability; their performance, however, is strongly affected by finite-precision issues in the representation of inner variables. Great attention has been paid, in recent literature, to the topic of quantization for LDPC decoders, but mostly focusing on binary modulations and analyzing finite precision effects in a disaggregrated manner, i.e., considering separately each block of the receiver. Modern telecommunication standards, instead, often adopt high order modulation schemes, e.g. M-QAM, with the aim to achieve large spectral efficiency. This puts additional quantization problems, that have been poorly debated in previous literature. This paper discusses the choice of suitable quantization characteristics for both the decoder messages and the received samples in LDPC-coded systems using M-QAM schemes. The analysis involves also the demapper block, that provides initial likelihood values for the decoder, by relating its quantization strategy with that of the decoder. A signal label for a signal in a 2*m*-ary modulation scheme is simply the *m*-bit pattern assigned to the signal. A mapping strategy refers to the grouping of bits within a codeword, where each *m*-bit group is used to select a 2*m*-ary signal in accordance with the signal labels. The most obvious mapping strategy is to use each group of *m* consecutive bits to select a signal. . We will call this the consecutive-bit (CB) mapping strategy. An alternative strategy is the bit-reliability (BR) mapping strategy which will be described below. A new demapper version, based on approximate expressions, is also presented, that introduces a slight deviation from the ideal case but yields a low complexity hardware implementation.**

Keywords- Low Density parity Check, Mapping, Demapping, Quantization, Quadrature Amplitude Modulation.

I. INTRODUCTION

LDPC codes (also known as Gallager codes) have recently received much attention from the communications industry because of their excellent error-correcting performance as well as having a highly parallelizable decoding algorithm even though they were developed half century ago. In 2003, the LDPC code beat six alternative turbo codes to become the error correcting code in the second generation standard for satellite transmission of digital television (European Telecommunication Standards Institute (ETSI)) and has already been proposed for the next generation digital terrestrial television standards (Digital Video Broadcasting (DVB))[1]. In designing the LDPC code the following design properties should be observed in order to obtain good code performance; first the code should be long enough, as performance improves with the code length. Second, few small cycles in the code bipartite graph since too many of them will seriously degrade the error-correcting performance. Finally, using carefully designed LDPC codes with irregular node degree distributions proved to remarkably outperform regular ones.

II. OVERVIEW OF PREVIOUS WORKS

A. Convolutional Coding

The information bits are input into shift registers and the output encoded bits are obtained by modulo-2 addition of the input information bits and the contents of the shift registers. The connections to the modulo-2 adders were developed heuristically with no algebraic or combinatorial foundation. The code rate r for a convolutional code is defined as $r = k/n$. The constraint length K for a convolutional code is defined as $K=m+ 1$, where m is the maximum number of stages (memory size) in any shift register. The shift registers store the state information of the convolutional encoder and the constraint length relates the number of bits upon which the output depends.

B. Duo-Binary Turbo Codes

Duo-Binary Turbo Codes (DBTC) differ from classical Convolutional Coding (CC) by the fact that the information bits are encoded pair wise [2]. The use of Circular Recursive

Systematic Constituent (CRSC) encoders, which rely on a tail-biting strategy, allows matching the ending and starting state of the encoder without any reduction of the code rate. The internal interleaver is based on an algorithmic permutation, defined by a single equation. This improved interleaver not only enhances the code's performance at low error rates, but also enables the adjustment of the frame size of this type of CC by modifying only four values which parameterize the internal interleaver. As the mother code rate of DBTC is 1/2 instead of 1/3 for classical CC, it is also inherently more robust to puncturing. DBTC do also perform well at smaller block sizes and have been standardized in DVB with block lengths as small as 128 bits. Another particularity of DBTC comes from the number of Log Likelihood Ratio (LLR) that should be propagated during iterative decoding. Dealing with duo-binary symbols implies handling only 3 LLRs per symbol, since each couple of bits (A_k, B_k) can take only the values (0,0), (0,1), (1,0) or (1,1). As a consequence:

$$LLR(A_k, B_k) = A_{a,b}(A_k, B_k)$$
$$= \log\left[\frac{P(A_k = a, B_k = b)}{P(A_k = 0, B_k = 0)}\right] \quad (1)$$

Where (a,b) $\neq$ (0,0).

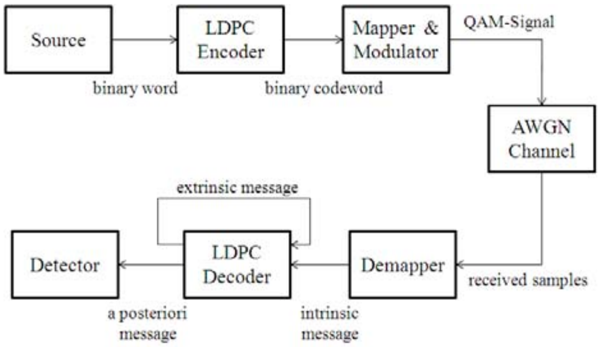

Figure 1. Block diagram of a LDPC-coded system.

III. SYSTEM MODEL

The LDPC encoder maps each *k*–bit word produced by the source into an *n*-bit LDPC codeword. Each codeword is then passed to the mapper and modulator block, that transforms groups of $t=log_2M$ code bits into a symbol of the bi-dimensional *M*-QAM constellation. The modulated signal is then transmitted over an Additive White Gaussian Noise (AWGN) channel. At the receiver side, the demapper block is a maximum a posteriori (MAP) symbol-to-bit metric calculator, that is able to produce an initial likelihood value for each received bit (such values are denoted as *intrinsic* or channel messages). These messages serve as input for the Sum-Product Algorithm (SPA), that starts iterating and, at each iteration, produces updated versions of the *extrinsic* and the *a posteriori* messages [3] are used as input for the subsequent iteration (if needed), while the latter represent the decoder output, and serve to obtain an estimated codeword that is subject to the hard decision and the parity-check test. This is as show in the above *Fig. 1.*

IV. LOW DENSITY PARITY CHECK CODES

A. Construction of G

A generator matrix G is used for constructing the code. The generator matrix may be found from the parity check matrix H.

First we note that

$$H_X = X^T H^T \quad (2)$$

The code word x may be split into one information part *I* and one parity check part c. The code word may then be
Written as

$$X^T = [i \mid c] \quad (3)$$

Correspondingly, the parity check matrix may be split into two matrices:

$$H = [A \mid B] \quad (4)$$

From (2), we note that vector i is multiplied with matrix A, whereas vector c is multiplied with matrix B.

On the basis of (3) and (4), written as

$$Ai + Bc = 0 \quad (5)$$

If the matrix B is non-singular, (5) may be inverted and the check bits c may be found from (6)

$$c = B^{-1}Ai \quad (6)$$

In practice, it may be necessary to swap over some of the columns in H in order for B to become non-singular. The product $B^{-1}A$ makes out the generator matrix G. This matrix is calculated only once, and is used for all encoding. The parity check matrix is used for constructing a graph structure in the decoder.

B. Graph Structure

The decoding of LDPC codes may be efficiently performed through the use of a graph structure. In this work, Tanner graphs will be used for the decoding [4]. The graph is constructed from the parity check matrix H. Each row in the matrix is represented by a check node, whereas each 1 in the row is represented by an edge into a bit node. Each column is represented by a bit node, and each 1 in the column corresponds to an edge into a check node. This is illustrated in *Fig. 2* and *Fig. 3*. In this manner, a graph is constructed which contains a total of N bit nodes and M check nodes. The number of edges is decided by the number of 1's in the parity check matrix. All edges are connected to a check node and to a bit node. The number of edges connected to a node denotes the degree of the node.

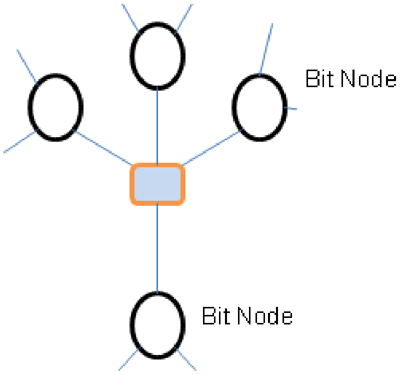

Figure 2. Check Nodes.

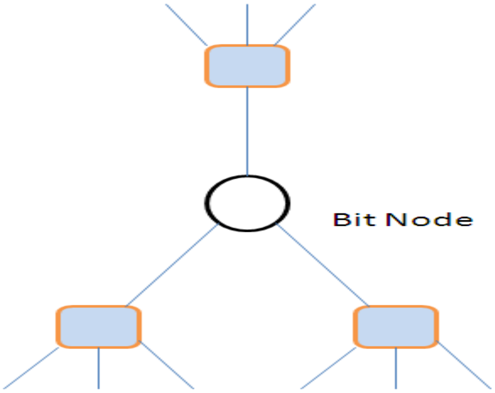

Figure 3. Bit Nodes.

C. *Decoding*

In this context, the decoder is a soft-decision input decoder, implying that it operates on the channel symbols, denoted by

$$r = 2x - 1 + n \tag{7}$$

where *n* is the AWGN noise vector added in the channel and *x* is the code word.

Finding the probability of the parity of a vector is a central concept in the decoding of LDPC codes. Each parity check may be regarded a vector of even parity [5]. First, we define the Likelihood Ratio (LR) as the ratio between the two probabilities P(x = 1) and P(x = 0):

$$LR = \frac{P(x=1)}{P(x=0)} \tag{8}$$

The symbol _ is used for the Log Likelihood Ratio,

$$\lambda = \frac{P(x=1)}{P(x=0)} \tag{9}$$

If x is a vector of bits, the LLR of a bit i in that vector is given by λ_i

$$\lambda_i = \frac{P(x_i=1)}{P(x_i=0)} \tag{10}$$

The notation $\Phi(x)$ is used for the vector parity. The LLR of the parity of the vector x is then given by:

$$\lambda_{\Phi(x)} = \frac{P(\Phi_x = 1)}{P(\Phi_x = 0)} \tag{11}$$

In order to compute $\lambda_{\Phi(x)}$, (12) is used:

$$\tanh\left(\frac{-\lambda_{\Phi(x)}}{2}\right) = \prod_{i=1}^{n} \tanh\left(\frac{-\lambda_i}{2}\right) \tag{12}$$

Equation (12) is solved with respect to $\lambda_{\Phi(x)}$:

$$\lambda_{\Phi(x)} = -2\tanh^{-1}\left(\prod_{i=1}^{n} \tanh\left(\frac{-\lambda_i}{2}\right)\right) \tag{13}$$

The a posteriori LLR of a bit n is given by:

$$\lambda_n = \log \frac{P(x_n = 1 \mid r)}{P(x_n = 0 \mid r)} \tag{14}$$

The vector r may be split into two parts: rn, which refers to the systematic part of the code word, and $\{r_{i\neq n}\}$, which refers to the parity bits of the code word. (15) may then be expressed as

$$\lambda_n = \log \frac{P(x_n = 1 \mid r_{n,} \{r_{i\neq n}\})}{P(x_n = 0 \mid r_{n,} \{r_{i\neq n}\})} \tag{15}$$

Bayes rule is given by:

$$p(a \mid b) = \frac{P(b,a)}{P(b)} \tag{16}$$

We use this rule in order to re-express the numerator of (15)

$$P(x_n = 1 \mid r_{n,} \{r_{i\neq n}\}) = \frac{f(r_{n,} x_n = 1 \{r_{i\neq n}\})}{f(r_{n,} \{r_{i\neq n}\})} \tag{17}$$

Furthermore, using the equality $p(a \mid b) = \dfrac{P(b,a)}{P(b)}$

$$\begin{aligned} &P(x_n = 1 \mid r_{n,} \{r_{i\neq n}\}) \\ &= \frac{f(r_{n,} \mid x_n = 1 \{r_{i\neq n}\}) f(x_n = 1, \{r_i \neq n\})}{f(r_{n,} \mid \{r_{i\neq n}\}) f(\{r_i \neq n\})} \end{aligned} \tag{18}$$

Given x_n, *rn* is statistically independent of $\{r_{i\neq n}\}$:

$$\begin{aligned} &P(x_n = 1 \mid r_{n,} \{r_{i\neq n}\}) \\ &= \frac{f(r_{n,} \mid x_n = 1) P[x_n = 1, \{r_i \neq n\}]}{f(r_{n,} \mid \{r_{i\neq n}\}) f(\{r_i \neq n\})} \end{aligned} \tag{19}$$

Following the same line of reasoning, the denominator of (15) is expanded. (15) then becomes:

$$\lambda_n = \log \frac{f(r_{n,} \mid x_n = 1)}{f(r_{n,} \mid x_n = 0)} + \log \frac{P[x_n = 1, \{r_i \neq n\}]}{P[x_n = 0, \{r_i \neq n\}]} \tag{20}$$

In an AWGN channel the conditional probability $f(r_n|x_n)$ is given by (21).

$$f(r_n \mid x_n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(r_n - 2x_n + 1)^2}{2\sigma^2}} \tag{21}$$

Substituting (21) into (20) we get (22).

$$\lambda_n = \frac{2}{\sigma^2} r_n + \log \frac{P[x_n = 1, \{r_i \neq n\}]}{P[x_n = 0, \{r_i \neq n\}]} \tag{22}$$

The first element of the right hand side of the equal sign is denoted the intrinsic part, whereas the second element is denoted the extrinsic part. Note that if it is known that the parity of a vector x is 0 (even parity), the probability that a bit x_n is 1, given the received values of the rest of the vector $r_i \neq n$, is the same as the probability that the rest of the vector $\{r_i \neq n\}$ has odd parity.

$$\lambda_n = \frac{2}{\sigma^2} r_n + \log \frac{P[\Phi x_{(i)} = 1 fori = 1 \ldots j \mid \{r_{i \neq n}\}]}{P[\Phi x_{(i)} = 0 fori = 0 \ldots j \mid \{r_{i \neq n}\}]} \tag{23}$$

If the graph is free of cycles, the vectors x1,x2..xj are independent, given $\{r_i \neq n\}$, and if the elements within the vector are independent, (23) may be written as (24) and (25).

$$\lambda_n = \frac{2}{\sigma^2} r_n + \log \frac{\prod_{i=1}^{j} P[\Phi x_{(i)} = 1 \mid \{r_{i \neq n}\}]}{\prod_{i=1}^{j} P[\Phi x_{(i)} = 0 \mid \{r_{i \neq n}\}]} \tag{24}$$

$$= \frac{2}{\sigma^2} r_n + \sum_{i=1}^{j} \log \frac{P[\Phi x_{(i)} = 1 \mid \{r_{i \neq n}\}]}{P[\Phi x_{(i)} = 0 \mid \{r_{i \neq n}\}]}$$

$$\lambda_n = \frac{2}{\sigma^2} r_n + \sum_{i=1}^{j} \lambda_{\Phi}(x_{(i)}) \tag{25}$$

In the graph, $\lambda_{i,l}$ is the message (contribution) from bit node *i* to check node l:

$$\lambda_{i,l} = \log \frac{P[\Phi x_{(i)} = 1 \mid \{r_{i \neq n}\}]}{P[\Phi x_{(i)} = 0 \mid \{r_{i \neq n}\}]} \tag{26}$$

We substitute (13) into (25)

$$\lambda_n = \frac{2}{\sigma^2} r_n - 2 \sum_{i=1}^{j} \tanh^{-1}\left(\prod_{l=2}^{k} \tanh\left(\frac{-\lambda_{i,l}}{2}\right)\right) \tag{27}$$

Equation (27) is the expression of the LLR of bit n. The first part is the intrinsic information, whereas the second part is the extrinsic information from the j vectors that contain bit n.

V. THE BIT RELIABILITY MAPPING STRATEGY

In general, an irregular LDPC code performs better than a regular LDPC code. An irregular LDPC is characterized by the degree distribution pair (*λi, ρj*), where *λi* is the fraction of edges connected to variable nodes of degree *i* and *ρj* is the fraction of edges connected to check nodes of degree *j*. Different variable node degrees imply different reliabilities after decoding. The larger the degree, the higher the average reliability. One way to explain this is to first note that the degree of a variable node is equal to the number of ones in the corresponding column of the code's parity-check matrix *H*. The column of a parity-check matrix can be considered to be a repetition code with the number of ones corresponding to the number of repetitions . The more repetitions, the more reliable the decoded bit; that is, the larger the node degree, the more reliable the decoded bit.

For *M*-ary modulation, we transmit *m* bits, (*cm*−1*, ...,* *c*1*,* *c*0)*,* in different levels (or "bit planes"),where *c*0 is in the least-significant-bit (LSB) level and *cm*−1 is in the most-significant-bit (MSB) level. Bits transmitted at different levels are protected differently. The LSB level has the weakest protection and the MSB level has the strongest protection. Based on this knowledge, we propose a bit-reliability mapping strategy. In the bit-reliability mapping strategy, we map the less reliable LDPC code bits to the lower level modulation bits and the more reliable bits to the higher level bits.

VI. DEMAPPER BASED ON APROXIMATION EXPRESSION

A. Second Order Approximation

When the value of SNR (and then of E_b / N_o) is sufficiently high, can be greatly simplified by considering, in each sum, the leading term only. This dominant contribution is due to the signals $s^0 = s^0 + js_y^0 \in A_k$ and $s^1 = s^1 + js_y^1 \in B_k$ that, for each *k*, are at minimum distance from the received sample. This technique coincides with the log-sum approximation and has been successfully applied for both product codes [6] and convolutional codes [7]. Actually, by imposing this simplification and taking into account becomes:

$$\mid \Delta z_k \approx \frac{SNR}{5a^2}(s_x^0 - s_x^1) + (s_x^0 - s_y^1)\frac{d_s}{2} \tag{28}$$

This relationship is very simple and more expressive than: first of all we notice a linear dependence on the SNR (such dependence is necessarily more involved in the rigorous expression). Moreover, in general, it can be further simplified. It is easy to see that s^0 and s^1 have always in common the in-phase component (i.e., $s_x^0 = s_x^1$) or the quadrature component (i.e., $s_y^0 = s_y^1$) and that the maximum difference between the unequal components is 4*a*.Together with the highlighted maximum value, with simple algebra we find:

$$m_s \geq \left\lceil \log_2^{\frac{SNR.T_s}{5ad_e}} \right\rceil + 3 \tag{29}$$

Where [*x*] is the smallest integer greater than *x*.

A function of E_b / N_o, and compared to the exact one (for bits 1 and 2). Both the exact and approximate curves exhibit, as obvious, a staircase behavior. Small regions usually exist, for low/medium signal-to-noise ratios, where the approximate formula can provide a value m_s of one bit higher than that given by the exact formula. Actually, these regions are practically indistinguishable, in the E_b / N_o range considered, for the first bit, whilst they are evident for the second bit. This is due to the fact that, when the second bit is considered, the maximum difference between the dominant contributions in $|\Delta z_2|$ is smaller than 4*a*. So, in principle, an adaptive quantization can be conceived, that varies the value of m_s according with the bit position. Anyway, it is clear that such a procedure would be difficult to manage in a practical implementation.

The same simplification used in (28) can be also introduced in the LLR expression. This looks like the classic max-log approximation. Under the same hypotheses, becomes:

$$\begin{aligned} &L(b_k) \approx L'(b_k) \\ &= SNR\frac{(x-s_x^1)^2 + (y-s_y^1)^2 - (x-s_x^0)^2 - (y-s_y^0)^2}{10a^2} \qquad (30) \\ &= f'_k(x,y,\sigma) \end{aligned}$$

The residual difference between $L(b_k)$ and $L'(b_k)$, that is due to the approximation, is appreciable for small signal-to-noise ratios. An example is shown in Fig. 4, for E_b / N_o =0 dB, where $L(b_1)$ and $L(b_2)$ are plotted as a function of *x*, for an arbitrary *y*. The difference becomes smaller and smaller for increasing signal-to-noise ratios and, at the values of E_b / N_o of interest (i.e., those required to have low error rates), it is usually acceptable for all bits. An example is shown in Fig. 5 for E_b / N_o =8dB; in this case the exact and approximate curves are almost overlaid. In comparison with Fig. 4, it is interesting to observe the very different LLRs dynamics.

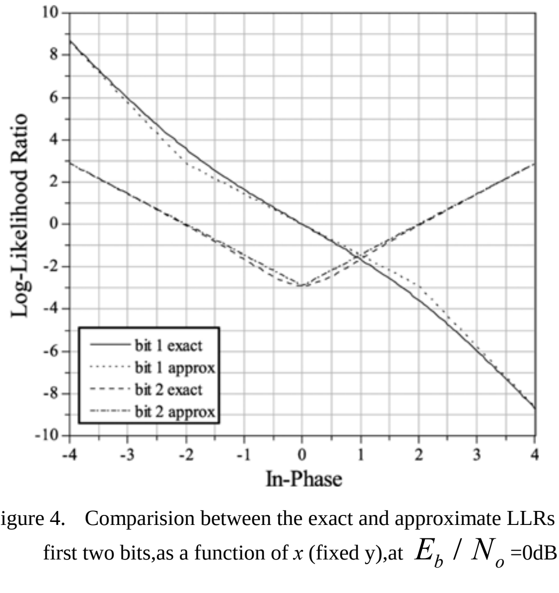

Figure 4. Comparision between the exact and approximate LLRs for the first two bits,as a function of *x* (fixed y),at E_b / N_o =0dB.

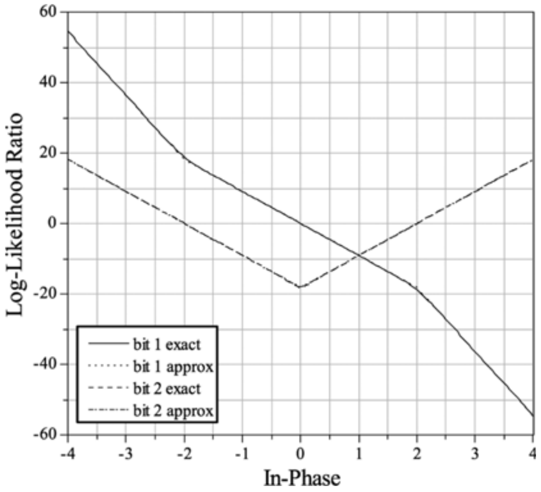

Figure 5. Comparision between the exact and approximate LLRs for the first two bits,as a function of *x* (fixed y),at E_b / N_o =8dB.

B. Simplified Demapper

The acceptability of the approximation suggests a simple solution to reduce considerably the complexity of the demapper block. The exact expression for $L(b_k)$, in fact, requires the implementation of a processor able to calculate $f_k(x,y,\sigma)$, given its inputs. An alternative solution would be to store the values of $f_k(x,y,\sigma)$ in a LUT indexed on x_q, y_q, σ_q (i.e. the quantized versions of *x,y,* σ , respectively). Looking at (30), instead, a smarter solution is possible. Due to the linearity in the SNR ,the m_c-bit level indexes for the quantized version of $f_k(x,y) = (x-s_x^1)^2 + (y-s_y^1)^2 - (x-s_x^0)^2 - (y-s_y^0)^2$ Can be stored in the LUT, in place of those of $L'(b_k)$. This way, the dependence on the SNR is eliminated, and the m_c-bit output words only depend on the m_s-bit input words, regardless of the channel. To reconstruct the value of $L'(b_k)$ from each m_c-bit value, if needed, the circuit shown in Fig. 6 can be adopted. It multiplies each level index by the fixed point representation of $SNR/(10a^2)$. This circuit uses an SNR value that is continuously estimated at the receiver side, for example by using the signal-mean square error (S/MSE) ratio. When multiplication is performed, it is easy to show that, if *l* is the number of bits used to represent (the always positive quantity) $SNR/(10a^2)$ and the m_c-bit index includes one sign bit, the output value of $L'(b_k)$ can be represented through $m' = m_c + l$ bits, at the most.

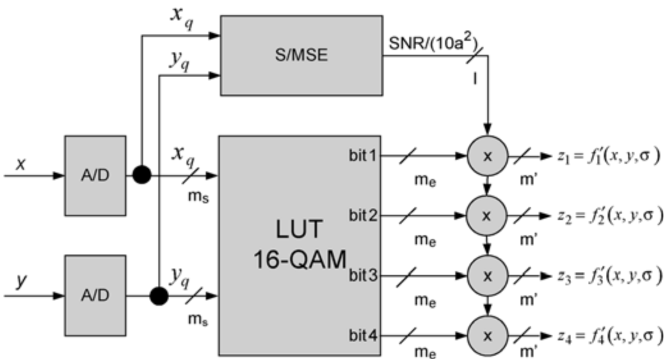

Figure 6. Circuit for the evaluation of $L'(b_k)$.

VII. CONCLUSION

We studied the performance of LDPC-coded modulation systems with 8PSK and 16QAM. With the proposed BR and second order approximation demapper strategy, a 0.15 dB - 0.2 dB performance improvement over the conventional mapping method is achieved. The performance of LDPC-coded modulation systems with Gray and natural labeling are studied. For natural labeling, iterative decoding/demodulation is required whereas demodulating once is all necessary for Gray labeling. We showed that mapper and demapper involved systems are always superior to systems.

REFERENCES

[1] Digital Video Broadcasting (DVB); Second Generation Framing Structure,Channel Coding and Modulation Systems for roadcasting, Interactive Services, News Gathering and Other Broadband Satellite Applications, ETSI EN Std. 302 307 (v1.1.2), Jun. 2006, Rev. 1.1.1.

[2] Douillard and C. Berrou, “Turbo Codes with rate-m/(m+1) constituent convolutional codes”, IEEE Trans. On Communications,vol. 53, no. 10, pp. 1630-1638, Oct. 2005.

[3] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of binary block and convolutional codes,” IEEE Trans. Inform. Theory, vol. 42, no. 2, pp. 429–445, Mar. 1996.

[4] R. M. Tanner, A recursive approach to low complexity codes, IEEE Transactions on Information Theory, vol. 27, IEEE, 1981, pp. 533–547.

[5] John R. Barry, Low-density parity-check codes, Tech. report, Georgia Institute of Technology, 2001.

[6] R. Pyndiah, A. Picart, and A. Glavieux, “Performance of block turbo coded 16-QAM and 64-QAM modulations,” in Proc. IEEE Global Telecommunications Conference (GLOBECOM’95), Singapore, Nov. 1995, vol. 2, pp. 1039–1043.

[7] F. Tosato and P. Bisaglia, “Simplified soft-output demapper for binary interleaved COFDM with application to HIPERLAN/2,” in Proc. IEEE ICC 2002, New York, May 2002, vol. 2, pp. 664–668.