Quantization and Quantization Sensitivity of Soft-Output Product Codes for Fast-Speed Applications

Ruiyuan Hu
Electrical & Computer Engineering Dept
Lehigh University
Bethlehem, PA 18015
ruh2@ece.lehigh.edu

Jing Li (Tiffany)
Electrical & Computer Engineering Dept
Lehigh University
Bethlehem, PA 18015
jingli@ece.lehigh.edu

Erozan Kurtas
Seagate Research Laboratory
1251 Waterfront Place
Pittsburgh, PA 15222
erozan.m.kurtas@seagate.com

***Abstract*— This paper investigates two quantization schemes on three soft-output message-passing decoding algorithms for 2-dimensional product codes.Quantization sensitivity on code rates and channel conditions is also investigated. The surprising yet encouraging result is that a simple (3,1) uniform quantization scheme on the min-sum algorithm results in the best overall quality in terms of space, performance and complexity.**

***Index Terms*— product codes, quantization, sum-product algorithm, min-sum algorithm, message-passing, soft decoding**

I. Introduction

The phenomenal performance of turbo codes, low-density parity-check (LDPC) codes and product codes has revolutionized the coding research with new concepts like code concatenation and iterative decoding. It has now been widely accepted that short and simple (almost useless) codes like single-parity check (SPC) codes can be combined to form a long and powerful code. In particular, multi-dimensional product codes based on single-parity check codes, also termed single-parity check turbo product codes or TPC/SPC codes [1][2], array codes [3] or hyper codes [4], have received considerable interests lately due to the intrinsic high rate, regular code structure, linear-time encodability, linear-time soft decodability, and a highly parallelizable encoding/decoding procedure [2]. Studies reveal that these codes perform on par with turbo or LDPC codes in a number of applications including the high-density digital recording systems [2].

This work is motivated by the need to understand how quantization, a necessary step toward fast VLSI hardware implementation, affects the performance of the soft-output decoder. For ease of VLSI implementation, we consider fixed-point arithmetics only. We investigate two quantization schemes, the uniform quantization and the hybrid quantization, on three soft-output decoding algorithms, the sum-product algorithm, the min-sum algorithm and the modified min-sum algorithm. The goal is to shed light on the optimal strategy that balances space, complexity and performance.

We start with a brief analysis on the structure of TPC/SPC codes and the three decoding algorithms in Section II. Section III discusses in detail different quantization strategies along with simulation results. Quantization sensitivity is also studied. Section IV summarizes the paper.

This project was supported in part by a grant from Seagate Technology and a grant from the Commonwealth of Pennsylvania, Department of Community and Economic Development, through the Pennsylvania Infrastructure Technology Alliance (PITA).

II. Background

A. TPC/SPC Codes

A multiple dimensional product code can be constructed in the following way [1]. The data to be transmitted are arranged in a hypercube of dimension d with the length in each dimension defined by $k_1, k_2, ..., k_d$. The ith dimension is encoded with a (n_i, k_i, d_{min_i}) systematic linear block code, and this is repeated for all $i = 1, ..., d$ dimensions. The resulting code has parameters $(\prod_{i=1}^{d} n_i, \prod_{i=1}^{d} k_i, \prod_{i=1}^{d} d_{min_i})$ (hence the name *product* code). Since a product code is typically decoded using an iterative decoder (which is composed of sub-decoders for each component code), it is also known as *turbo product code* (TPC). This paper investigates product codes that are formed from single-parity check codes, namely, TPC/SPC codes. Since high rates are desirable in a number of applications (e.g. the data storage system), and since a high dimensionality in product codes reduces the code rate as well as detriments the efficiency of iterative decoding[1], we focus on 2-dimensional TPC/SPC codes in this paper.

B. Decoder Analysis

Although a product code is generally decoded via an iterative constraint search algorithm known as the Chase algorithm, TPC/SPC codes can be efficiently decoded by an iterative message-passing (MP) algorithm. Assuming even-parity check codes in all dimensions, antipodal signaling ($0 \rightarrow +1$, $1 \rightarrow -1$) and additive white Gaussian noise (AWGN) channels, a 2-D TPC/SPC code formed from row code $\mathcal{C}_1] \sim (N_1, N_1-1)$ and column code $\mathcal{C}_2 \sim (N_2, N_2-1)$ has the following soft-input soft-output (SISO) decoding algorithm (see Tab. I) [2]. We use L to denote the log-likelihood ration (LLR) information. Subscript ch, o, e, and c refer to the LLR information obtained from the channel observation, the *a priori* LLR, the extrinsic LLR, and the overall LLR of the code, respectively. Superscript $(1, \tau)$ and $(2, \tau)$ refer to the quantities associated with row code and column code in the τth decoding iteration.

The core operation in the decoding algorithm is the so-called *check operation*, $L(u_1) \uplus L(u_2) = L(u_1 \oplus u_2) \triangleq \ln \frac{P(u_1 \oplus u_2 = 0)}{P(u_1 \oplus u_2 = 1)}$, which is also where the majority of the complexity stems.

[1]Our experience with concatenated/compound codes reveals that an iterative decoder can approximate the optimal decoder quite well when there are only a few component codes, but the performance soon deteriorates as the number of component codes increases [7].

TABLE I

2-D TPC/SPC CODES DECODING ALGORITHM

Definitions:
$L(u_1) \uplus L(u_2) = L(u_1 \oplus u_2) \triangleq \ln \frac{P(u_1 \oplus u_2 = 0)}{P(u_1 \oplus u_2 = 1)}$
Initialization:
for i $=$ 1 to N_2, for j $=$ 1 to N_1,
$Lch_{i,j} = \frac{2}{\sigma^2} r_{i,j}$, $Le_{i,j}^{(1,0)} = Le_{i,j}^{(2,0)} = 0$,
The τth Iteration:
Decoding $\mathcal{C}_1$: *for* $i = 1$ *to* N_2, *for* $j = 1$ *to* N_1,
$Lo_{i,j}^{(1,\tau)} = Lch_{i,j} + Le_{i,j}^{(2,\tau-1)}$,
$Le_{i,j}^{(1,\tau)} = \sum_{1 \le t \le N_1, t \ne j}^{\uplus} Lo_{i,t}^{(1,\tau)})$,
Decoding $\mathcal{C}_2$: for j $=$ 1 to N_1, for i $=$ 1 to N_2,
$Lo_{i,j}^{(2,\tau)} = Lch_{i,j} + Le_{i,j}^{(1,\tau)}$,
$Le_{i,j}^{(2,\tau)} = \sum_{1 < t < N_2, t \ne j}^{\uplus} Le_{t,j}^{(2,\tau)})$,
Soft Output and Decisions after M iterations:
for i $=$ 1 to N_2, for j $=$ 1 to N_1,
$Lc_{i,j} = Lo_{i,j} + Le_{i,j}^{(1,M)} + Le_{i,j}^{(2,M)}$,
$\hat{u}_{i,j} = Lc_{i,j} > 0 \ ? \ 0 : 1;$

Depending on how the check operation is implemented, several different forms of message-passing decoding result.

The Sum-Product Algorithm: The sum-product algorithm is the best MP algorithm where the check operation is optimally implemented using either the f-function or the $tanh$-rule [2]:

$$L(u_1) \uplus L(u_2) \triangleq 2tanh^{-1}\left(tanh\frac{L(u_1)}{2}\tanh\frac{L(u_2)}{2}\right) \quad (1)$$

$$= \prod_{i=1}^{2} \mathrm{sgn}(L(u_i)) \cdot \ f\left(\sum_{i=1}^{2} f\big(|L(u_i)|\big)\right) \quad (2)$$

$$= \mathrm{sgn}(L(u_1))\ \mathrm{sgn}(L(u_2))\ \min(|L(u_1)|, |L(u_2)|) + \underbrace{\log \frac{1+e^{-|L(u_1)+L(u_2)|}}{1+e^{-|L(u_1)-L(u_2)|}}}_{\text{correction factor}}, \quad (3)$$

$$\text{where} \quad f(z) = \ln \frac{e^z+1}{e^z-1}, \qquad z > 0,$$

and the sign operation is defined as $\mathrm{sgn}(u) = +1$ if $u > 0$, $\mathrm{sgn}(u) = 0$ if $u = 0$, and $\mathrm{sgn}(u) = -1$ if $u < 0$.

The Min-Sum Algorithm: The min-sum algorithm approximates the sum-product algorithm by ignoring the non-linear *correction factor* in (3), i.e.

$$L(u_1) \uplus L(u_2) \approx \prod_{i=1}^{2} \mathrm{sgn}(L(u_i)) \cdot \min(|L(u_1)|, |L(u_2)|). \quad (4)$$

The Modified Min-Sum Algorithm: The min-sum algorithm is extremely simple, but may result in a noticeable performance degradation. An easy enhancement is, instead of completely ignoring the correction factor, to compensate the signed-min operation with a simple, fixed-value correction term. Let $x_1 = L(u_1)+L(u_2)$, $x_2 = L(u_1)-L(u_2)$ and $g(x) = \log(1+e^{-|x|})$, then the original correction factor can be approximated as [6]:

$$g(x_1) - g(x_2) = \begin{cases} c & : \text{if } |\mathrm{x_1}| < 2 \text{ and } |\mathrm{x_2}| > 2|\mathrm{x_1}|, \\ -c & : \text{if } |\mathrm{x_2}| < 2 \text{ and } |\mathrm{x_1}| > 2|\mathrm{x_2}|, \\ 0 & : \text{otherwise.} \end{cases} \quad (5)$$

Simulation experiments show that the optimal value of c varies slightly with different SNR values, but the range of $c \in [0.4, 0.6]$ tends to yield good performance. For ease of implementation, we fix $c = 0.5$. This translates to a 1-bit cost of the correction term in hardware implementation.

To give an idea of how the different decoding algorithms perform with TPC/SPC codes, we show in Fig. 1 the simulation results of a non-quantized $(16, 15)^2$ TPC/SPC code (this code will also be used throughout the paper unless otherwise stated). BER performances after 1, 2, 3 and 4 iterations are examined. Two things are immediately observable. First, sum-product and min-sum algorithms, also differ in complexity, appear to yield similar BER performance. Second, 3 decoding iterations seem to be sufficient to exploit most of the coding gain. Hence, only 3 decoding iterations will be considered hereafter.

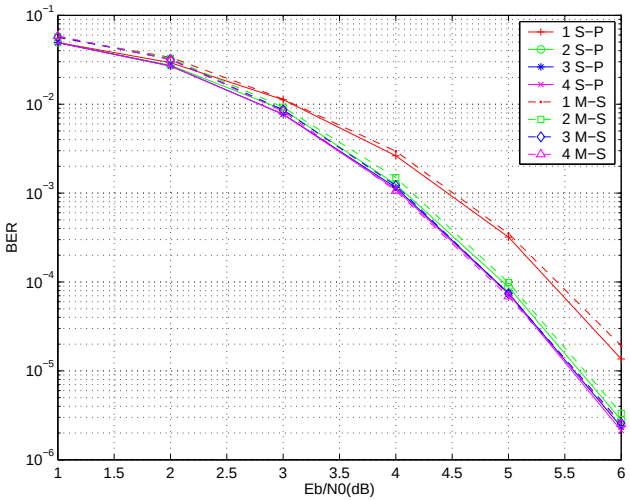

Fig. 1. Non-quantized Sum-Product & Min-Sum TPC/SPC Decoder ("S-P" stands for sum-product decoding and "M-S" stands for min-sum decoding)

III. QUANTIZATION OF TPC/SPC DECODER

Toward practical fast-speed hardware implementation, quantization and quantization sensitivity are indispensable issues that need to be resolved. Among the various consideration, the word length M, i.e. the number of bits or the number of quantization levels needed to represent an internal/external data, is of particular importance. Clearly, an excessively large M incurs unnecessary hardware cost for the buffer as well as the decoding computation, whereas an overly small M may significantly degrade the performance, since quantization and fixed point arithmetics inevitably introduce noise to the decoder (termed *self-noise*). In this section, we investigate both the uniform and the non-uniform (hybrid) quantization schemes (with various word lengths) on the three message-passing algorithms discussed before. The goal is to find the best trade-off between the hardware complexity and the performance.

A. Sum-Product TPC/SPC Decoder

We start with the uniform quantization scheme. A uniform quantizer is simple and fast. It makes linear operations like additions easy to implement, but may not be able to make optimal use of the available bits. Let (p, q) represent a fixed-point number with p integer bits and q fractional bits. An additional

sign bit is also allocated for each data. This corresponds to limiting the range of an LLR value (i.e. internal/external data) to $[-2^{p+1}+2^{-q},\ 2^{p+1}-2^{-q}]$ with precision of 2^{-q}. When p is small, chopping off the absolute LLR values to no more than 2^{p+1} tends to impair the performance. Hence, a scaling factor is used. The value of the scaling factor is tuned with respect to the specific quantization scheme and the specific SNR value under investigation. This will be discussed in more detail later.

Among the various ways of implementing the check operation (Eqn. (1)-(3)), we opt for the f-function implementation in (2). The key advantage is that the function $f(z)$ rapidly approaches zero as z increases as shown in Fig. 2.

Suppose we could quantize this term and store the value of z say for 2^{p+q} levels, and assume it is zero for $z > z_{thresh}$, then we get an approximation for the nonlinear function $f(z)$. The number of quantization levels shall be determined through evaluating the values of LLR information and the values of the $f(z)$ operation. Observing that the inverse of $f(z)$ is exactly itself, and that $f(1/2^5) \approx 4.16$, $f(1/2^4) \approx 3.47$ and $f(1/2^3) \approx 2.78$, we decide that quantization schemes (3,5), (3,4) and (2,3) are reasonable choices. Our experiments show that the (3,5) scheme yields the best performance, and the $(2,3)$ scheme incurs a significant performance loss compared to the other two. Hence, for the sake of clarity, we present in Fig. 3 the performances of the (3,5) and (3,4) schemes only.

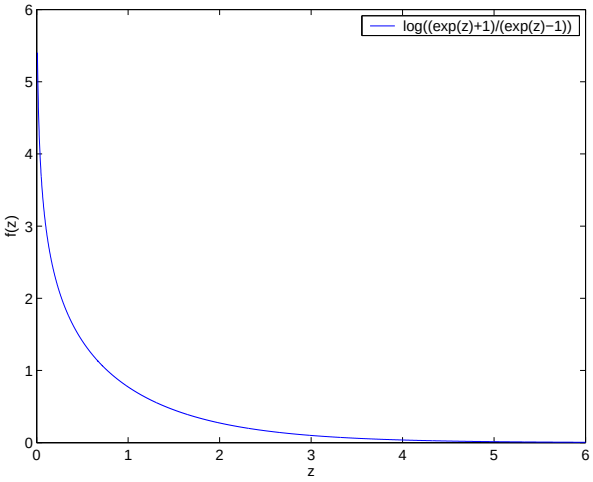

Fig. 2. Plot of $f(z) = \ln \frac{e^z+1}{e^z-1},\ z > 0$

While the $(3,5)$ scheme sees hardly any performance loss in comparison with the non-quantized decoder (Fig. 3), the need for 9 bits (1 sign bit) to represent a data imposes a high cost in the fast VLSI implementation. Is it possible to use fewer bits to achieve approximately the same performance?

Before we answer this, let us revisit the nonlinear function $f(z)$ in the sum-product decoder (Fig. 2). Intuitively, the optimal quantizer needs to have varying quantization steps that are matched to the slope of the curve. This suggests the use of a non-uniform quantizer. On the other side, concerns for high complexity hold us back from non-uniform quantizers, since they make (linear) operations tricky. In this work, we circumvent the issue by introducing a *hybrid quantizer*, which is formed from a combination of two uniform quantizers. This is motivated by the observation that the $f(z)$ curve experiences roughly two slope regions: the water-fall region and the floor region (Fig. 2). At the water-fall region, the steep slope calls on a quantizer with a small quantization step, i.e. a large q, since the quantization step is $\propto 1/q$). At the floor region, the shallow slop but the long span of the curve calls on a quantizer with a large quantization region, i.e. a large p, since the quantization region is $\propto 2^{p+1}$. Specifically, the following quantizing rule is used in the hybrid quantizer that we investigate, and similar techniques have also been used in [8]:

- If the magnitude of $f(z)$ is smaller than 1, then the output is represented by a $(1, q_1)$ uniform quantization scheme, where the integer part is always set to 0;
- Otherwise a different quantization scheme (p_2, q_2) will be adopted, where the most significant bit (MSB) of the integer part is always set to 1.

We experimented on different combinations of $(p_2, q_2)/(1, p_1)$ hybrid schemes. Among them, the $(4,2)/(1,5)$ scheme stands out by exhibiting essentially the same performance as the $(3,5)$ uniform quantization scheme (but using fewer bits). The $(4,1)/(1,4)$ scheme can save an additional bit with a slightly performance compromise. The performances of all these schemes (after three decoding iterations) are shown and compared in Fig. 3.

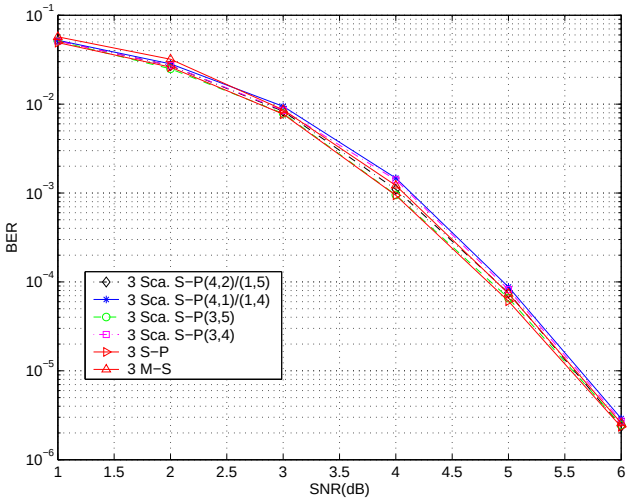

Fig. 3. Uniform & Hybrid Quantization Schemes on the Sum-Product Decoder (3 decoding iterations; "Sca" means that an optimal scaling factor is applied; the performance of non-quantized decoders is provided as a reference)

B. Min-Sum TPC/SPC Decoder

Although the $(4,2)/(1,5)$ and $(4,1)/(1,4)$ hybrid quantizers on the sum-product decoder use fewer bits than the $(3,5)$ scheme, the nonuniformity in the structure nevertheless incurs computational inconvenience. In seeking (possibly) better solutions, we turn to the min-sum and the modified min-sum decoders. Since these two algorithms involve linear operations only, it is sufficient to look at the uniform quantizer only. Various schemes are examined, are the best results are summarized in Fig. 4. We observe that the (3,1) uniform quantizer on the min-sum decoder performs slightly worse than the (3,5) quantizer on the sum-product decoder, but requires

significantly less hardware. The introduction of the one-bit correction term in the modified min-sum algorithm yields very little performance improvement, which seems unworthy of the extra bit that is used.

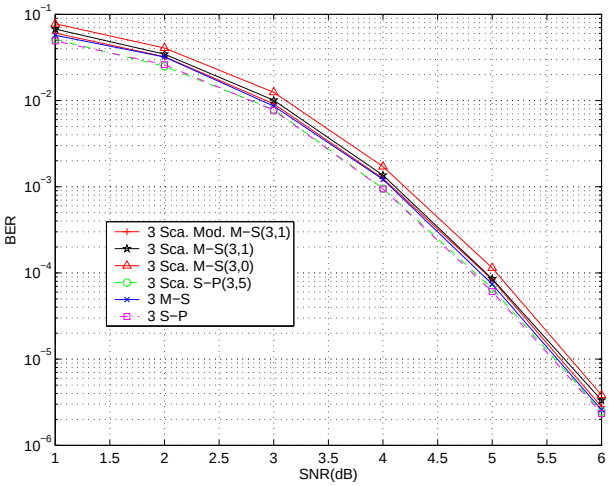

Fig. 4. Uniform Quantization on the Min-Sum Decoder and the Modified Min-Sum Decoder ("Mod. M-S" stands for modified min-sum decoder)

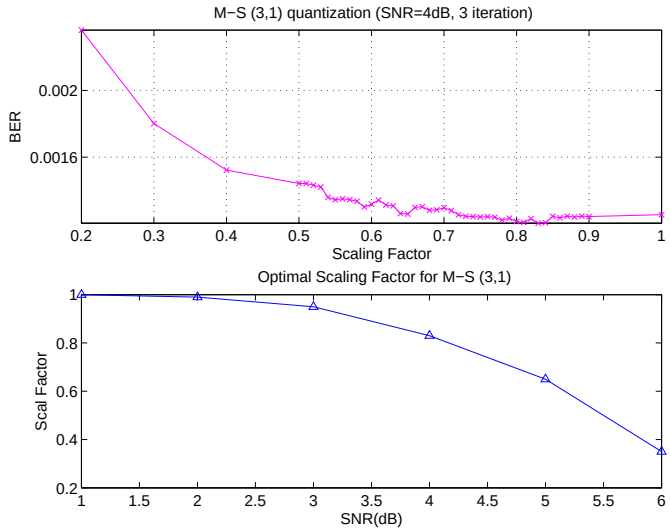

Fig. 5. Optimal Gain/Scaling Factor for (3,1) Quantized Min-Sum Decoder

C. Optimal Gain and Quantization Sensitivity

To achieve the best signal-to-distortion ratio at the quantizer, the amplitude of the signal needs to be adjusted by a gain control or a scaling factor. There is an optimal value to scale the received signal (and consequently the LLR values) before it is fed into the quantizer. When the scaling factor is too small, quantization noise will distort the signal; when it is too large, saturation will occur. Both cases result in an undesirable performance degradation. As we mentioned earlier, the optimal value is dependent upon the specific quantizer and the specific SNR value under investigation.

To demonstrate how sensitive the performance of the quantizer/decoder is to the scaling factor, we take the (3,1) quantizer on the min-sum decoder as an example. We first fix an SNR value and examine the impact of the scaling factor on the BER performance. As shown in the upper sub-plot of Fig. 5, different values of the scaling factor result in different BER performances. For the specific case examined, i.e. SNR$=4$dB, a scaling factor of around 0.8 yields the best BER performance. We note that the optimal scaling factor (or the optimal gain) is also a function of the SNR value. As illustrated in the lower sub-plot of Fig. 5, the optimal scaling factor tends to be large (approaches 1) at low SNR values, but decreases as SNR increases.

We have also investigated the sensitivity of the quantization schemes to the code rate. For the codes that are of interest, that is, having high rates like $(16, 15)^2$ and $(32, 31)^2$ TPC/SPC codes, we find that the value of the scaling factor (optimized for a given quantization scheme and a given SNR value) to be relatively stable for different code rates.

IV. Conclusion

We have investigated quantization strategy for fixed-point arithmetics on three soft-output decoding algorithms for TPC/SPC codes. The key issue is a balanced trade-off between complexity and performance (see Tab. II). With the goal for simplicity and speed in mind, we opt for the (3,1) uniform quantizer on the min-sum decoder over all other quantizer/decoder schemes.

Quantization sensitivity is another important issue that we have investigated. The observation is that the scaling factor, which is required in order to maximize the performance, is relatively sensitive to the channel condition or the SNR values, but much less so to the code rate.

TABLE II

Performance & Complexity Comparison

	Performance	Complexity
High	Hybrid, S-P	Hybrid, S-P
	Uniform, S-P	Uniform, S-P
	Uniform, Modified M-S	Uniform, Modified M-S
Low	Uniform, M-S	Uniform, M-S

References

[1] D. M. Rankin and T.A. Gulliver, "Single parity check product codes," *IEEE Trans. Communications*, pp. 1354-1362, August 2001.

[2] J. Li, K. R. Narayanan, E. Kurtas and C.N. Georghiades, "On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels," *IEEE Trans. Commun.* vol. 50, pp. 723-734, pp. 2176-2183, May 2002.

[3] M. Blaum, P.G. Farrell and H.C.A. van Tilborg, "Array codes," *Handbook of Coding Theory*, V.S. Pless and W.C. Huffman, North-Holland, pp. 1855-1909, 1998.

[4] P.-P. Sauve, A. Hunt, S. Crozier and P. Guinand, "Hyper-Codes: high-Performance, low-complexity codes," *Proc. 2nd Intl. Symp. on Turbo Codes and Related Topics*, France, Sept. 2000.

[5] J. Hagenauer, E. Offer and L. Papke, "Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Trans. Inform. Theory* pp. 429-445, March 1996.

[6] E. Eleftheriou, T. Mittelholzer and A. Dholakia, "Reduced-complexity decoding algorithm for low-density parity-check codes," *IEEE Electronics Letters*, vol. 37, pp. 102-104, Jan. 2001.

[7] J. Li, K. R. Narayanan, and C. N. Georghiades, "Generalized product accumulate codes: analysis and performance," *Proc. of IEEE GLOBECOM*, San Antonio, Nov. 2001.

[8] T. Zhang, Z. Wang, K.K. Parhi, "On finite precision implementation of low density parity check codes decoder," *The 2001 IEEE International Symposium on Circuits and Systems*, vol.4, pp. 202-205, May 2001.