Turkish Journal of Electrical Engineering and Computer Sciences

Volume 22 | Number 1 Article 18

1-1-2014

A low-memory intensive decoding architecture for double-binary convolutional turbo code

MING ZHAN

LIANG ZHOU

JUN WU

Follow this and additional works at: https://journals.tubitak.gov.tr/elektrik

 Part of the Computer Engineering Commons, Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation
ZHAN, MING; ZHOU, LIANG; and WU, JUN (2014) "A low-memory intensive decoding architecture for double-binary convolutional turbo code," *Turkish Journal of Electrical Engineering and Computer Sciences*: Vol. 22: No. 1, Article 18. https://doi.org/10.3906/elk-1203-86
Available at: https://journals.tubitak.gov.tr/elektrik/vol22/iss1/18

This Article is brought to you for free and open access by TÜBİTAK Academic Journals. It has been accepted for inclusion in Turkish Journal of Electrical Engineering and Computer Sciences by an authorized editor of TÜBİTAK Academic Journals. For more information, please contact academic.publications@tubitak.gov.tr.

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2014) 22: 202 – 213
© TÜBİTAK
doi:10.3906/elk-1203-86

A low-memory intensive decoding architecture for double-binary convolutional turbo code

Ming ZHAN[1,2,*], **Liang ZHOU**[1], **Jun WU**[3]

[1]National Key Laboratory of Science and Technology on Communications, University of Electronic Science and Technology of China, Chengdu, China

[2]School of Electronics and Information Engineering, Southwest University, Chongqing, China

[3]Global Information and Telecommunication Institute, Waseda University, Tokyo, Japan

Received: 19.03.2012 • **Accepted:** 31.08.2012 • **Published Online:** 20.12.2013 • **Printed:** 20.01.2014

Abstract: Memory accesses take a large part of the power consumption in the iterative decoding of double-binary convolutional turbo code (DB-CTC). To deal with this, a low-memory intensive decoding architecture is proposed for DB-CTC in this paper. The new scheme is based on an improved maximum a posteriori probability algorithm, where instead of storing all of the state metrics, only a part of these state metrics is stored in the state metrics cache (SMC), and the memory size of the SMC is thus reduced by 25%. Owing to a compare-select–recalculate processing (CSRP) module in the proposed decoding architecture, the unstored state metrics are recalculated by simple operations, while maintaining near optimal decoding performance.

Key words: Branch metrics, computational complexity, MAP algorithm, state metrics cache

1. Introduction

In 1999, nonbinary convolutional turbo codes were introduced by Berrou and Jezequel [1], and they have been demonstrated to give better performance than classical single-binary turbo codes [2]. Due to these advantages, double-binary convolutional turbo code (DB-CTC) has been adopted by several radio standards as the channel encoding scheme, such as the digital video broadcasting-return channel over satellite (DVB-RCS) [3], and the worldwide interoperability for microwave access [4]. Recently, in order to improve the error correction performance and the systematic throughput, DB-CTC was recommended by the IEEE 802.16m as the forward error correction code [5,6].

The typical decoder, using an iterative decoding algorithm for turbo-like codes, consists of 2 soft-in and soft-out (SISO) constituent decoders, where each one computes the extrinsic information using the outputs of the other one. However, due to the memory-intensive decoding architecture, frequent memory accesses are performed in the iteration procedures, and more than half of the entire power consumption is accounted for the accessing operation [7]. To design a power-efficient decoder for turbo-like codes, researchers have worked out different kinds of techniques to reduce the size of the state metrics cache (SMC). The authors in [8] reduced the bit width of the state metrics based on the saturation of the state metrics, which led to a decoding scheme with less storage for the SMC. Martina et al. compressed the decoder area by employing a nonuniform quantization technique [9], and at the cost of a slight performance loss, this method achieves 20%–50% reduction of the

*Correspondence: zmdjs@swu.edu.cn

state metrics memory area. As for the DB-CTC, Kim and Park studied the encoding of border metrics, where the energy consumed by the constituent decoder was reduced by approximately 26% [10]. Lin et al. introduced the traceback maximum a posteriori probability (MAP) decoding method to reduce the size of the SMC [11]. Additionally, researchers have developed low complexity [12–15] decoding algorithms and memory-reduced technologies [16,17] for DB-CTC. Although these schemes reduce power consumption effectively, the adopted algorithms are suboptimal; hence, performance loss is unavoidable in their hardware implementation.

Inspired by the memory-reduced technique in [11] and [18], in this paper, we propose a low-memory intensive decoding architecture to decrease the storage of the SMC. This research is based on our previous work of an efficient decoding algorithm for DB-CTC [19]. In the proposed decoding scheme, 6 of the 8 forward (or backward) state metrics are stored in the SMC at each time slot, while the 2 unstored state metrics can be recalculated by a compare-select–recalculate processing (CSRP) module with simple operations. As a small-sized SMC and less memory accesses are needed in the proposed decoding architecture, the overall power consumption of the DB-CTC decoder can be decreased. Moreover, the simulation results show that the new architecture gets near optimal performance compared to that of the classical MAP algorithm.

The remaining of this paper is organized as follows. Section 2 introduces an improved MAP algorithm, where exponential operations are performed outside of the constituent decoder. Section 3 proposes a low-memory intensive decoding scheme and shows how to recalculate the unstored state metrics by the CSRP module, where the structures of the CSRP module and the constituent decoder are also described in detail. Section 4 investigates the performance of the proposed architecture, such as the complexity of the recalculation, SMC organization, decoder timing diagram, and bit error rate (BER). Finally, section 5 gives the conclusion.

2. Improved MAP algorithm

The optimal decoding algorithm suitable for DB-CTC is the classical MAP algorithm [20]. By assuming that the code word is transmitted through an additive white Gaussian noise channel with noise variance σ^2, the calculation of branch metrics $\gamma_k^z(s',s)$ in the classical MAP algorithm is given by:

$$\gamma_k^z(s',s) = \exp\left(\frac{1}{2}L_c\left(r_k^{s1}x_k^{s1} + r_k^{s2}x_k^{s2} + r_k^{p1}x_k^{p1} + r_k^{p2}x_k^{p2}\right) + L_z^a(u_k)\right), \tag{1}$$

where k is the time slot, $(s',s) \in S = \{s_0, \cdots, s_7\}$ are the trellis states, z belongs to $\varphi = \{00, 01, 10, 11\}$, u_k is the information bits in pairs, $L_z^a(u_k)$ is the a priori log-likelihood ratio for $u_k = z$, $L_c = 2/\sigma^2$, $s1, s2$ are the systemic parts, $p1, p2$ are the parity parts, and X_k, R_k represent the transmitted and received code word, respectively.

Calculation of the branch metrics $\gamma_k^z(s',s)$ is a crucial point in the classical MAP algorithm. If $\gamma_k^z(s',s)$ is multiplied by a common factor ϕ_k, the decoding performance would not be degraded. If $\phi_k = \exp\left(-\frac{1}{2}L_c\left(r_k^{s1} + r_k^{s2} + r_k^{p1} + r_k^{p2}\right)\right)$, we get a new method for the calculation of the branch metrics $\bar{\gamma}_k^z(s',s)$, as below:

$$\begin{aligned} \bar{\gamma}_k^z(s',s) &= \gamma_k^z(s',s)\,\phi_k \\ &= \left(\bar{r}_k^{s1}\right)^{v_k^{s1}}\left(\bar{r}_k^{s2}\right)^{v_k^{s2}}\left(\bar{r}_k^{p1}\right)^{v_k^{p1}}\left(\bar{r}_k^{p2}\right)^{v_k^{p2}}\bar{L}_z^a(v_k), \end{aligned} \tag{2a}$$

$$\begin{cases} v_k^{\lambda} &= \left(x_k^{\lambda} + 1\right)/2 \\ \bar{r}_k^{\lambda} &= \exp\left(L_c r_k^{\lambda}\right) \end{cases} \qquad \lambda \in \{s1\,, s2\,, p1\,, p2\,\}\,, \tag{2b}$$

where v_k^{λ} belongs to $\{\,0\,,\,1\,\}$, $v_k = v_k^{s1} v_k^{s2}$ belongs to φ, and $\bar{L}_z^a(v_k) = P^a(v_k = z)/P^a(v_k = 00)$ is the a priori likelihood ratio for $v_k = z$.

Based on the new approach that $\bar{\gamma}_k^z(s', s)$ is computed, and assumes $\bar{\alpha}_k(s)$, $\bar{\beta}_k(s')$, $\bar{L}_z(v_k)$, and $\bar{L}_z^e(v_k)$ as the forward state metrics, the backward state metrics, the a posteriori likelihood ratio, and the extrinsic information, respectively, we have derived the following improved algorithm for the DB-CTC (please refer to [19] for details):

$$\bar{\alpha}_k(s) = \sum_{s' \in S, z \in \varphi} \bar{\gamma}_k^z(s', s)\, \bar{\alpha}_{k-1}(s'), \tag{3}$$

$$\bar{\beta}_k(s') = \sum_{s' \in S, z \in \varphi} \bar{\gamma}_{k+1}^z(s', s)\, \bar{\beta}_{k+1}(s), \tag{4}$$

$$\bar{L}_z(v_k) = \frac{\sum_{(s',s) v_k = z} \bar{\alpha}_{k-1}(s')\, \bar{\gamma}_k^z(s', s)\, \bar{\beta}_k(s)}{\sum_{(s',s) v_k = 00} \bar{\alpha}_{k-1}(s')\, \bar{\gamma}_k^z(s', s)\, \bar{\beta}_k(s)}, \tag{5}$$

$$\bar{L}_z^e(v_k) = \frac{\bar{L}_z(v_k)}{\left(\bar{r}_k^a\right)^{v_k^{s1}} \left(\bar{r}_k^b\right)^{v_k^{s2}} \bar{L}_z^a(v_k)} = \frac{\sum_{(s',s) v_k = z} \bar{\alpha}_{k-1}(s') \left(\bar{r}_k^{p1}\right)^{v_k^{p1}} \left(\bar{r}_k^{p2}\right)^{v_k^{p2}} \bar{\beta}_k(s)}{\sum_{(s',s) v_k = 00} \bar{\alpha}_{k-1}(s') \left(\bar{r}_k^{p1}\right)^{v_k^{p1}} \left(\bar{r}_k^{p2}\right)^{v_k^{p2}} \bar{\beta}_k(s)}. \tag{6}$$

In the above algorithm, if $\bar{r}_k^{\lambda}$ are calculated in an exponential preprocessing module, and then stored in the receive buffer, we find that only the multiply and addition operations exist in the iteration of the constituent decoder. Therefore, we get a generic decoding structure for the DB-CTC, as illustrated in Figure 1 (note that $\bar{r}_k^{p11}$ and $\bar{r}_k^{p22}$ are the interleaved parity parts).

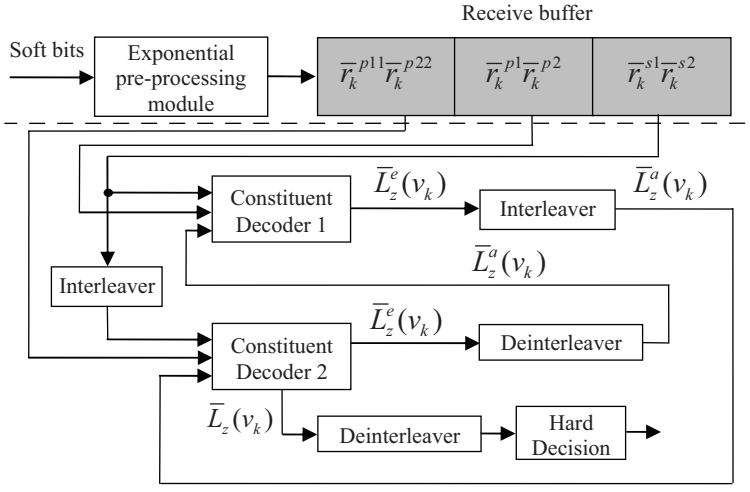

Figure 1. Decoder for DB-CTC based on the improved MAP algorithm.

3. Low-memory intensive decoding architecture

3.1. Parallel decoding structure and butterfly scheme

To increase the decoding speed, the parallel window (PW) decoding structure is proposed for the hardware implementation of the turbo-like decoder [21]. Figure 2 gives a generic decoding structure for DB-CTC using the PW technique. The received soft bits of a constituent decoder are divided into N nonoverlapping windows, and each of these windows works independently and simultaneously. Moreover, the values for the border metrics of each window come from the previous iteration of the 2 neighbor windows (considering the tail-biting characteristic of DB-CTC, the first and last windows are adjacent windows for each other). Additionally, it should be noted that for the first iteration, all of these border metrics are initialized by the same value.

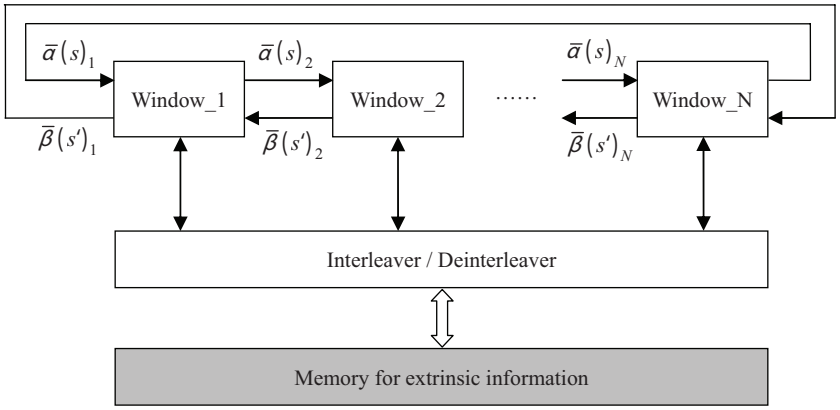

Figure 2. PW decoding structure for DB-CTC, where $\bar{\alpha}(s)_1, \bar{\beta}(s')_1, \cdots, \bar{\alpha}(s)_N, \bar{\beta}(s')_N$ are the border metrics.

To further speed the iteration, butterfly scheduling [11,18] is adopted in this research. In the decoding window, let W be the window length, and $\bar{\alpha}_k(s)$, $\bar{\beta}_k(s')$, $k = [0, 1, \cdots, W]$ are computed from the opposite 2 ends of the window simultaneously. $\bar{\alpha}_0(s)$ and $\bar{\beta}_W(s')$ are the border metrics that come from the neighbor windows, while $\bar{\alpha}_W(s)$ and $\bar{\beta}_0(s')$ are the border metrics that should be output to the neighbor windows. When $0 \leq k \leq W/2 - 1$, $\bar{\alpha}_k(s)$ and $\bar{\beta}_{W-k}(s')$ are stored in the last-in and first-out (LIFO) SMC. When $W/2 \leq k \leq W - 1$, $\bar{\alpha}_k(s)$, $\bar{\beta}_{W-k}(s')$, and the stored state metrics in the SMC are used to compute extrinsic information $\bar{L}_z^e(v_{k+1})$ and $\bar{L}_z^e(v_{W-k})$ (or the a posteriori likelihood ratio $\bar{L}_z(v_{k+1})$ and $\bar{L}_z(v_{W-k})$).

Based on the introduced MAP algorithm in Section 2, and by adopting the PW decoding structure and the butterfly scheme, we propose a method to decrease the memory size of the SMC in the following sections.

3.2. Recalculation of the unstored forward state metrics

In Max-log-MAP and its derivatives, the Max operation is used to decrease the decoding complexity. Consequently, forward computation of the backward state metrics is considerably complicated [22]. In the forward recursion of the classical decoding architecture [15,23], 8 forward state metrics $\bar{\alpha}_k(s)$ are calculated and then stored in SMC_α at every time slot k. By adopting the introduced MAP algorithm in Section 2, our research shows that storing 6 of the 8 forward state metrics is enough, while at the time slot where the 2 unstored state

metrics are needed, they can be recomputed by a CSRP module with simple operations. For convenience of presentation, we write Eq. (3) in the form of a matrix as below:

$$\begin{bmatrix} \bar{\alpha}_k(s_0) \\ \bar{\alpha}_k(s_3) \\ \bar{\alpha}_k(s_4) \\ \bar{\alpha}_k(s_7) \end{bmatrix} = \begin{bmatrix} \bar{\gamma}_k^{00}(s_0,s_0) & \bar{\gamma}_k^{10}(s_1,s_0) & \bar{\gamma}_k^{01}(s_6,s_0) & \bar{\gamma}_k^{11}(s_7,s_0) \\ \bar{\gamma}_k^{11}(s_0,s_3) & \bar{\gamma}_k^{01}(s_1,s_3) & \bar{\gamma}_k^{10}(s_6,s_3) & \bar{\gamma}_k^{00}(s_7,s_3) \\ \bar{\gamma}_k^{10}(s_0,s_4) & \bar{\gamma}_k^{00}(s_1,s_4) & \bar{\gamma}_k^{11}(s_6,s_4) & \bar{\gamma}_k^{01}(s_7,s_4) \\ \bar{\gamma}_k^{01}(s_0,s_7) & \bar{\gamma}_k^{11}(s_1,s_7) & \bar{\gamma}_k^{00}(s_6,s_7) & \bar{\gamma}_k^{10}(s_7,s_7) \end{bmatrix} \begin{bmatrix} \bar{\alpha}_{k-1}(s_0) \\ \bar{\alpha}_{k-1}(s_1) \\ \bar{\alpha}_{k-1}(s_6) \\ \bar{\alpha}_{k-1}(s_7) \end{bmatrix}, \tag{7a}$$

$$\begin{bmatrix} \bar{\alpha}_k(s_1) \\ \bar{\alpha}_k(s_2) \\ \bar{\alpha}_k(s_5) \\ \bar{\alpha}_k(s_6) \end{bmatrix} = \begin{bmatrix} \bar{\gamma}_k^{00}(s_2,s_1) & \bar{\gamma}_k^{10}(s_3,s_1) & \bar{\gamma}_k^{01}(s_4,s_1) & \bar{\gamma}_k^{11}(s_5,s_1) \\ \bar{\gamma}_k^{11}(s_2,s_2) & \bar{\gamma}_k^{01}(s_3,s_2) & \bar{\gamma}_k^{10}(s_4,s_2) & \bar{\gamma}_k^{00}(s_5,s_2) \\ \bar{\gamma}_k^{10}(s_2,s_5) & \bar{\gamma}_k^{00}(s_3,s_5) & \bar{\gamma}_k^{11}(s_4,s_5) & \bar{\gamma}_k^{01}(s_5,s_5) \\ \bar{\gamma}_k^{01}(s_2,s_6) & \bar{\gamma}_k^{11}(s_3,s_6) & \bar{\gamma}_k^{00}(s_4,s_6) & \bar{\gamma}_k^{10}(s_5,s_6) \end{bmatrix} \begin{bmatrix} \bar{\alpha}_{k-1}(s_2) \\ \bar{\alpha}_{k-1}(s_3) \\ \bar{\alpha}_{k-1}(s_4) \\ \bar{\alpha}_{k-1}(s_5) \end{bmatrix}. \tag{7b}$$

In the forward direction, the forward state metrics, except for $\bar{\alpha}_k(s_4), \bar{\alpha}_k(s_6), k \in \{0, 1, \cdots, W/2-1\}$, are stored in SMC_α. Considering that butterfly scheduling is adopted in our proposed decoding architecture, when the kth decoding slot comes, it is just the time slot to calculate the backward state metrics $\bar{\beta}_k(s')$. Therefore, $\bar{\gamma}^z_{k+1}(s',s)$ is used not only for the recursion of the backward state metrics $\bar{\beta}_k(s')$, but also for recalculation of the 2 unstored forward state metrics.

To calculate $\bar{\alpha}_k(s_6),\ k = [W/2-1\ ,\ \cdots\ ,\ 0]$, we can derive 4 equations from Eq. (7a):

$$\begin{cases} \bar{\gamma}_{k+1}^{01}(s_6,s_0)\,\bar{\alpha}_k\ (s_6) = \bar{\alpha}_{k+1}(s_0) - \bar{\gamma}_{k+1}^{00}(s_0,s_0)\,\bar{\alpha}_k(s_0) \\ \qquad\qquad -\bar{\gamma}_{k+1}^{10}(s_1,s_0)\,\bar{\alpha}_k(s_1) - \bar{\gamma}_{k+1}^{11}(s_7,s_0)\,\bar{\alpha}_k(s_7) \\ \bar{\gamma}_{k+1}^{10}(s_6,s_3)\,\bar{\alpha}_k(s_6) = \bar{\alpha}_{k+1}(s_3) - \bar{\gamma}_{k+1}^{11}(s_0,s_3)\,\bar{\alpha}_k(s_0) \\ \qquad\qquad -\bar{\gamma}_{k+1}^{01}(s_1,s_3)\,\bar{\alpha}_k(s_1) - \bar{\gamma}_{k+1}^{00}(s_7,s_3)\,\bar{\alpha}_k(s_7) \\ \bar{\gamma}_{k+1}^{11}(s_6,s_4)\,\bar{\alpha}_k(s_6) = \bar{\alpha}_{k+1}(s_4) - \bar{\gamma}_{k+1}^{10}(s_0,s_4)\,\bar{\alpha}_k(s_0) \\ \qquad\qquad -\bar{\gamma}_{k+1}^{00}(s_1,s_4)\,\bar{\alpha}_k(s_1) - \bar{\gamma}_{k+1}^{01}(s_7,s_4)\,\bar{\alpha}_k(s_7) \\ \bar{\gamma}_{k+1}^{00}(s_6,s_7)\,\bar{\alpha}_k(s_6) = \bar{\alpha}_{k+1}(s_7) - \bar{\gamma}_{k+1}^{01}(s_0,s_7)\,\bar{\alpha}_k(s_0) \\ \qquad\qquad -\bar{\gamma}_{k+1}^{11}(s_1,s_7)\,\bar{\alpha}_k(s_1) - \bar{\gamma}_{k+1}^{10}(s_7,s_7)\,\bar{\alpha}_k(s_7) \end{cases}. \tag{8}$$

Because of the recursive process, $\bar{\alpha}_{k+1}(s), s = [s_0, s_3, s_4, s_7]$ are obtained by the feedback, the branch metrics $\bar{\gamma}^z_{k+1}(s',s)$ are computed by the branch metrics unit (BMU), and $\bar{\alpha}_k(s_0)$, $\bar{\alpha}_k(s_1)$, and $\bar{\alpha}_k(s_7)$ are obtained by accessing SMC_α. Therefore, $\bar{\alpha}_k(s_6)$ is the only unknown metric in Eq. (8). In theory, each equation in Eq. (9) is qualified to recalculate $\bar{\alpha}_k(s_6)$, but our numerical simulation shows that only the equation with the maximum value of $\bar{\alpha}_k(s_6)$'s coefficients is valid. Otherwise, a minor error of the recalculation would degrade the accuracy in the next recursion. Consequently, a Max operation with 4 operands is employed to decide which equation is the suitable one. For example, if we get a result by:

$$Max\left[\bar{\gamma}^{01}_{k+1}(s_6,s_0), \bar{\gamma}^{10}_{k+1}(s_6,s_3), \bar{\gamma}^{11}_{k+1}(s_6,s_4), \bar{\gamma}^{00}_{k+1}(s_6,s_7)\right] = \bar{\gamma}^{01}_{k+1}(s_6,s_0), \tag{9}$$

the first equation in Eq. (8) is selected to calculate $\bar{\alpha}_k(s_6)$. As the recalculation keeps running, $\bar{\alpha}_k(s_6), k = [W/2-1, \cdots, 0]$ are computed in the backward direction.

Meanwhile, we can recalculate $\bar{\alpha}_k(s_4), k = [W/2-1, \cdots, 0]$ by deriving 4 equations from Eq. (7b) as follows:

$$\begin{cases} \bar{\gamma}^{01}_{k+1}(s_4,s_1)\,\bar{\alpha}_k(s_4) = \bar{\alpha}_{k+1}(s_1) - \bar{\gamma}^{00}_{k+1}(s_2,s_1)\,\bar{\alpha}_k(s_2) \\ \qquad\qquad -\bar{\gamma}^{10}_{k+1}(s_3,s_1)\,\bar{\alpha}_k(s_3) - \bar{\gamma}^{11}_{k+1}(s_5,s_1)\,\bar{\alpha}_k(s_5) \\ \bar{\gamma}^{10}_{k+1}(s_4,s_2)\,\bar{\alpha}_k(s_4) = \bar{\alpha}_{k+1}(s_2) - \bar{\gamma}^{11}_{k+1}(s_2,s_2)\,\bar{\alpha}_k(s_2) \\ \qquad\qquad -\bar{\gamma}^{01}_{k+1}(s_3,s_2)\,\bar{\alpha}_k(s_3) - \bar{\gamma}^{00}_{k+1}(s_5,s_2)\,\bar{\alpha}_k(s_5) \\ \bar{\gamma}^{11}_{k+1}(s_4,s_5)\,\bar{\alpha}_k(s_4) = \bar{\alpha}_{k+1}(s_5) - \bar{\gamma}^{10}_{k+1}(s_2,s_5)\,\bar{\alpha}_k(s_2) \\ \qquad\qquad -\bar{\gamma}^{00}_{k+1}(s_3,s_5)\,\bar{\alpha}_k(s_3) - \bar{\gamma}^{01}_{k+1}(s_5,s_5)\,\bar{\alpha}_k(s_5) \\ \bar{\gamma}^{00}_{k+1}(s_4,s_6)\,\bar{\alpha}_k(s_4) = \bar{\alpha}_{k+1}(s_7) - \bar{\gamma}^{01}_{k+1}(s_2,s_7)\,\bar{\alpha}_k(s_2) \\ \qquad\qquad -\bar{\gamma}^{11}_{k+1}(s_3,s_6)\,\bar{\alpha}_k(s_3) - \bar{\gamma}^{10}_{k+1}(s_5,s_6)\,\bar{\alpha}_k(s_5) \end{cases}. \tag{10}$$

Similarly, to select the suitable equation in Eq. (10) for the recalculation, we calculate the maximum value among the coefficients of $\bar{\alpha}_k(s_4)$. For instance, if we get a result by:

$$Max\left[\bar{\gamma}^{01}_{k+1}(s_4,s_1), \bar{\gamma}^{10}_{k+1}(s_4,s_2), \bar{\gamma}^{11}_{k+1}(s_4,s_5), \bar{\gamma}^{00}_{k+1}(s_4,s_6)\right] = \bar{\gamma}^{11}_{k+1}(s_4,s_5), \tag{11}$$

the third equation in Eq. (11) is used to recalculate $\bar{\alpha}_k(s_4)$. As the recalculation keeps running, $\bar{\alpha}_k(s_4), k = [W/2-1, \cdots, 0]$ are computed in the backward direction.

3.3. Recalculation of the unstored backward state metrics

In the process of backward recursion, the 8 backward state metrics $\bar{\beta}_k(s')$ are calculated at every time slot k, and 6 of them are stored in SMC_β. Rewriting Eq. (4) in the form of a matrix, we get:

$$\begin{bmatrix} \bar{\beta}_k(s_0) \\ \bar{\beta}_k(s_1) \\ \bar{\beta}_k(s_6) \\ \bar{\beta}_k(s_7) \end{bmatrix} = \begin{bmatrix} \bar{\gamma}^{00}_{k+1}(s_0,s_0) & \bar{\gamma}^{11}_{k+1}(s_0,s_3) & \bar{\gamma}^{10}_{k+1}(s_0,s_4) & \bar{\gamma}^{01}_{k+1}(s_0,s_7) \\ \bar{\gamma}^{10}_{k+1}(s_1,s_0) & \bar{\gamma}^{01}_{k+1}(s_1,s_3) & \bar{\gamma}^{00}_{k+1}(s_1,s_4) & \bar{\gamma}^{11}_{k+1}(s_1,s_7) \\ \bar{\gamma}^{01}_{k+1}(s_6,s_0) & \bar{\gamma}^{10}_{k+1}(s_6,s_3) & \bar{\gamma}^{11}_{k+1}(s_6,s_4) & \bar{\gamma}^{00}_{k+1}(s_6,s_7) \\ \bar{\gamma}^{11}_{k+1}(s_7,s_0) & \bar{\gamma}^{00}_{k+1}(s_7,s_3) & \bar{\gamma}^{01}_{k+1}(s_7,s_4) & \bar{\gamma}^{10}_{k+1}(s_7,s_7) \end{bmatrix} \begin{bmatrix} \bar{\beta}_{k+1}(s_0) \\ \bar{\beta}_{k+1}(s_3) \\ \bar{\beta}_{k+1}(s_4) \\ \bar{\beta}_{k+1}(s_7) \end{bmatrix}, \tag{12a}$$

$$\begin{bmatrix} \bar{\beta}_k(s_2) \\ \bar{\beta}_k(s_3) \\ \bar{\beta}_k(s_4) \\ \bar{\beta}_k(s_5) \end{bmatrix} = \begin{bmatrix} \bar{\gamma}^{00}_{k+1}(s_2,s_1) & \bar{\gamma}^{11}_{k+1}(s_2,s_2) & \bar{\gamma}^{10}_{k+1}(s_2,s_5) & \bar{\gamma}^{01}_{k+1}(s_2,s_6) \\ \bar{\gamma}^{10}_{k+1}(s_3,s_1) & \bar{\gamma}^{01}_{k+1}(s_3,s_2) & \bar{\gamma}^{00}_{k+1}(s_3,s_5) & \bar{\gamma}^{11}_{k+1}(s_3,s_6) \\ \bar{\gamma}^{01}_{k+1}(s_4,s_1) & \bar{\gamma}^{10}_{k+1}(s_4,s_2) & \bar{\gamma}^{11}_{k+1}(s_4,s_5) & \bar{\gamma}^{00}_{k+1}(s_4,s_6) \\ \bar{\gamma}^{11}_{k+1}(s_5,s_1) & \bar{\gamma}^{00}_{k+1}(s_5,s_2) & \bar{\gamma}^{01}_{k+1}(s_5,s_5) & \bar{\gamma}^{10}_{k+1}(s_5,s_6) \end{bmatrix} \begin{bmatrix} \bar{\beta}_{k+1}(s_1) \\ \bar{\beta}_{k+1}(s_2) \\ \bar{\beta}_{k+1}(s_5) \\ \bar{\beta}_{k+1}(s_6) \end{bmatrix}. \tag{12b}$$

In the backward direction, the backward state metrics, except for $\bar{\beta}_k(s_4), \bar{\beta}_k(s_6), k \in \{W, \cdots, W/2+1\}$, are stored in SMC_β. By the same method described in Section 3.2, the unstored backward state metrics can also be recursively recalculated at their corresponding time slot.

3.4. Proposed decoding architecture with the CSRP module

Based on the aforementioned discussion, recalculation of the unstored state metrics includes 3 steps, which is called the CSRP operation in this paper, as shown in Figure 3.

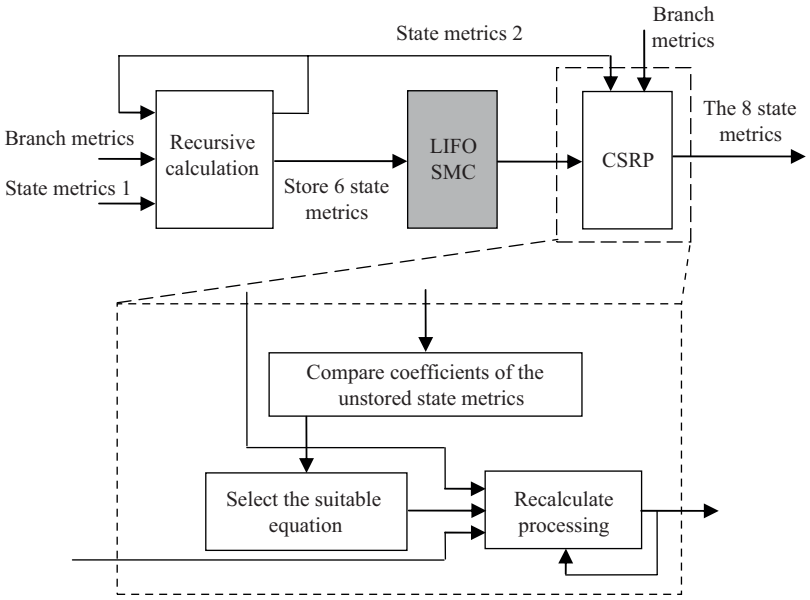

Figure 3. Recalculation of the unstored state metrics by the CSRP module.

1) Compare coefficients of the unstored state metrics.

2) Select the appropriate equation among the 4 candidate equations.

3) Recalculate the unstored state metrics in the recalculate processing module recursively.

It should be noted that state metrics 1 is α_{in} or β_{in} (they are the border metrics that are delivered from the neighboring windows of the previous iteration), while state metrics 2 is $\bar{\alpha}_{W/2}(s)$ or $\bar{\beta}_{W/2}(s')$. Take constituent decoder 1 as an example, where the overall decoding architecture with the CSRP module for DB-CTC is presented in Figure 4, and the decoding procedure can be decomposed into 2 main steps:

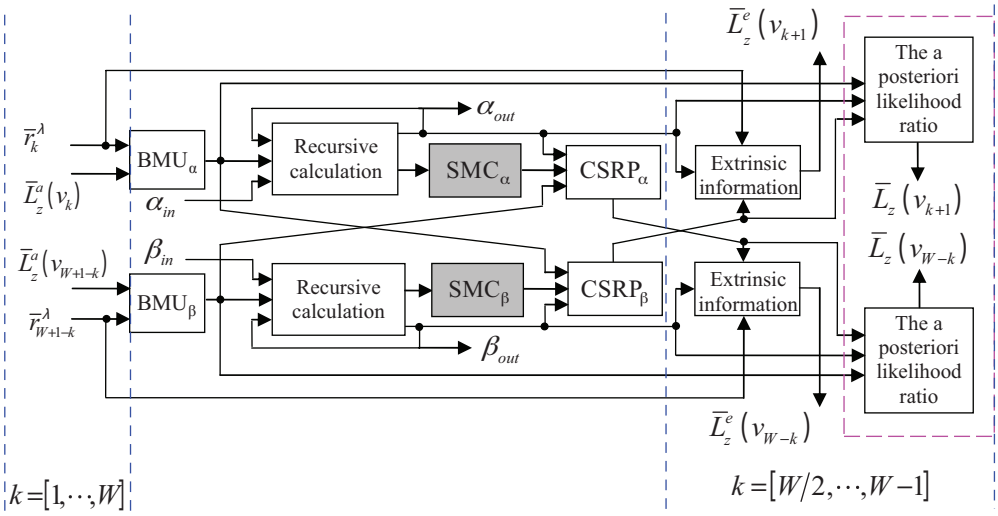

Figure 4. Proposed decoding architecture with the CSRP module for DB-CTC.

1) When $1 \leq k \leq W/2-1$:

1.1) Using the preprocessed soft bits $\bar{r}_k^{\lambda}$ (or $\bar{r}_{W+1-k}^{\lambda}$) and the a priori likelihood ratio $\bar{L}_z^a(v_k)$ (or $\bar{L}_z^a(v_{W+1-k})$), the branch metrics $\bar{\gamma}_k^z(s',s)$ (or $\bar{\gamma}_{W+1-k}^z(s',s)$) are computed by the BMU_α (or BMU_β).

1.2) The values of $\bar{\alpha}_0(s)$ (or $\bar{\beta}_W(s')$) are initialized as α_{in} (or β_{in}). Subsequently, $\bar{\alpha}_k(s)$ (or $\bar{\beta}_{W-k}(s')$) are computed by the recursive calculation module, and then $\bar{\alpha}_0(s), \bar{\alpha}_k(s), s=[s_0,s_1,s_2,s_3,s_5,s_7]$ are stored in SMC_α (or $\bar{\beta}_W(s'), \bar{\beta}_{W-k}(s'), s'=[s_0,s_1,s_2,s_3,s_5,s_7]$ are stored in SMC_β).

2) When $W/2 \leq k \leq W$:

2.1) By the same operations in Step 1.1, branch metrics $\bar{\gamma}_k^z(s',s)$ (or $\bar{\gamma}_{W+1-k}^z(s',s)$) are computed recursively.

2.2) When $k = W/2$, $\bar{\alpha}_{W/2}(s)$ (or $\bar{\beta}_{W/2}(s')$) are computed by the recursive calculation module, instead of be stored in SMC_α (or SMC_β), they are used by CSRP_α (or CSRP_β) to initiate state metrics 2.

2.3) When $W/2 \leq k \leq W-1$, $\bar{\alpha}_k(s)$ are computed in the forward direction, and the corresponding 6 backward state metrics in SMC_β are read out to recalculate $\bar{\beta}_{k+1}(s_4)$ and $\bar{\beta}_{k+1}(s_6)$ by CSRP_β. Subsequently, $\bar{L}_z^e(v_{k+1})$ are computed with $\bar{\alpha}_k(s)$, $\bar{\beta}_{k+1}(s')$, and $\bar{r}_{k+1}^{\lambda}$ by the extrinsic information module (in the last iteration, the a posteriori likelihood ratio $\bar{L}_z(v_{k+1})$ is computed). Simultaneously, $\bar{\beta}_{W-k}(s')$ are computed in the backward direction, and the corresponding 6 forward state metrics in SMC_α are read out to recalculate $\bar{\alpha}_{W-k-1}(s_4)$ and $\bar{\alpha}_{W-k-1}(s_6)$ by CSRP_α. Subsequently, $\bar{L}_z^e(v_{W-k})$ are computed with $\bar{\alpha}_{W-k-1}(s)$, $\bar{\beta}_{W-k}(s')$, and $\bar{r}_{W-k}^{\lambda}$ by the extrinsic information module (in the last iteration, the a posteriori likelihood ratio $\bar{L}_z(v_{W-k})$ is computed).

2.4) When $k = W$, $\bar{\alpha}_W(s)$ (or $\bar{\beta}_0(s')$) are the forward (or backward) state metrics that are calculated in Step 2.3, being initiated as the border metrics α_{out} (or β_{out}), they are delivered to the neighboring windows and will be used in the next iteration.

4. Performance analysis and BER simulation

In this section, we analyze the proposed decoding architecture in terms of the recalculated complexity, decoder timing diagram, and memory organization. In addition, to show the near optimal decoding performance of the new scheme, the result of the BER simulation is also presented.

4.1. Complexity of the recalculation

Based on the aforementioned discussion, recalculation of the unstored state metrics is performed in the CSRP module. To convenience the hardware implementation, 1 Max operation with 4 operands is decomposed into 3 Max operations with 2 operands as below [17]:

$$Max(a,b,c,d) = Max(Max(a,b), Max(c,d)). \tag{13}$$

Considering the structure of the equations in Eq. (8), once a suitable equation is decided, the unstored state metrics can be recalculated by 4 multiply and 3 addition operations. As seen from the description in Section 3.4, when $W/2 \leq k \leq W-1$, there are 4 state metrics that should be recalculated by the CSRP_α and CSRP_β modules. As a result, we summarize the overall complexity of the recalculation in Table 1.

Table 1. Complexity of the recalculation.

Operations / State metrics	Addition	Multiply	*Max* (2 operands)
Forward state metrics	4 × 3 × W/2	4 × 4 × W/2	4 × 3 × W/2
Backward state metrics	4 × 3 × W/2	4 × 4 × W/2	4 × 3 × W/2
Total	4 × 3 × W	4 × 4 × W	4 × 3 × W

4.2. Decoder timing diagram

In Figure 4, the forward state metrics $\bar{\alpha}_k(s)$ are recursively computed from the beginning of the window to the end of the window, while the backward state metrics $\bar{\beta}_k(s')$ are recursively computed in the opposite direction. Therefore, computation of the extrinsic information is separated into 2 parts, and then they are parallel computed from the middle of the window to the beginning and the end of the window. Figure 5 describes the timing flow of the proposed decoding architecture, and shows that the proposed architecture is a bidirectional high-speed decoding scheme.

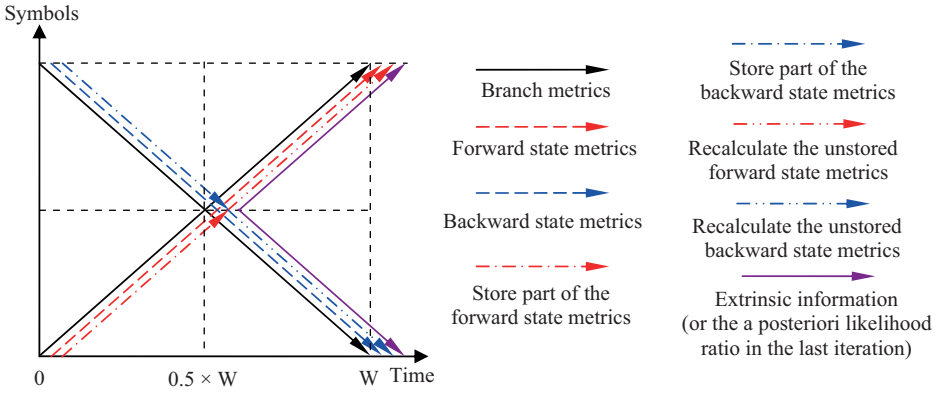

Figure 5. Timing diagram of the proposed decoding architecture.

4.3. Memory organization

Since the proposed decoding architecture uses butterfly decoding scheduling, the branch metrics are computed from the opposite 2 directions by BMU_α and BMU_β simultaneously, and no memory is used for the branch metrics. More importantly, by employing the CSRP module in this architecture, only 6 forward (or backward) state metrics are needed to be stored at the corresponding time slot. Therefore, the memory storage of the SMC is decreased by 25% compared with the classic decoding architecture in [15] and [23], and is also smaller than the decoding architectures in [10], [11], and [17]. Given the quantization for each of the state metrics $J = 10$, memory organizations of the 4 decoding architectures are summarized in Table 2.

5. Performance of the BER

To verify the effectiveness of our proposed decoding architecture, the enhanced Max-log-MAP (EML-MAP), the classical MAP and the proposed decoding architecture with the CSRP module are selected for comparison. In the simulation, the interleaver parameters of the code rate-1/3 DB-CTC are in accordance with the requirements of 802.16m [5]. The number of the iterations performed by the decoder is 8, and the signal-to-noise ratio ranges from 0.1 dB to 1.5 dB. The bit frame length and window size equals 800 and 20, respectively. The results in

Figure 6 show the BER performance of the proposed decoding architecture is about 0.05 dB inferior to that of the classical MAP algorithm, although our scheme performs low-memory intensive decoding.

Table 2. Summary of the memory organization.

Memory / Architecture	SMC_α	SMC_β	Sign bits	Branch metrics	Total
The classical architecture in [15] and [23]	8 × J × W	0	0	8 × J × W	16 × J × W
Architecture in [10] and [17]	7 × J × W	0	0	8 × J × W	15 × J × W
Architecture in [11]	6 × J × W/2	6 × J × W/2	4 × W	0	(6 × J + 4) × W
The proposed architecture	6 × J × W/2	6 × J × W/2	0	0	6 × J × W

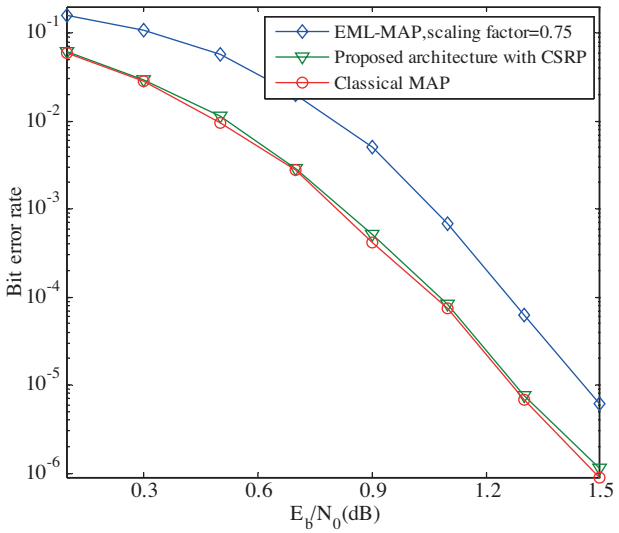

Figure 6. BER of the proposed decoding architecture.

6. Conclusion

Based on an improved MAP algorithm for DB-CTC, we have proposed a low-memory intensive decoding architecture for the design of a low-power consumed decoder. When compared with the conventional decoding architecture, our scheme leads to a 25% reduction in the memory size for the LIFO SMC, and the unstored state metrics are effectively recalculated by a CSRP module. At the price of the computational complexity performed by the CSRP module, the number of memory accesses is also reduced by 25%. Since the PW decoding technique and the butterfly decoding scheduling are adopted, the proposed decoding architecture also achieves a high decoding speed. The simulation and comparison show that the new architecture has almost the same BER performance as that of the classical MAP algorithm. Therefore, the proposed decoding architecture can be applied in the very-large–scale integration implementation of a low-power and high speed DB-CTC decoder.

Acknowledgments

This work is supported by the National Important Special Project of China: Research & Evaluation of 802.16m (No. 2009ZX03003-010). The authors would like to thank the anonymous reviewer and the editor for their insightful comments and suggestions on this paper.

References

[1] C. Berrou, M. Jezequel, "Non-binary convolutional codes for turbo coding", Electronics Letters, Vol. 35, pp. 39–40, 1999.

[2] C. Berrou, M. Jezequel, C. Douillard, S. Kerouedan, "The advantage of non-binary turbo codes," Information Theory Workshop, pp. 61–63, 2001.

[3] Digital Video Broadcasting (DVB), [Online], Available: http://www. dvb.org/.

[4] Worldwide Interoperability for Microwave Access (WiMAX), [Online], Available: http://www.wimaxforum.org/home/.

[5] IEEE P802.16m/D5-2010, IEEE Standard for Local and Metropolitan Area Networks, Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems, New York, 2010.

[6] IEEE Standard 802.16^{TM}-2009, IEEE Standard for Local and Metropolitan Area Networks, Part 16: Air Interface for Broadband Wireless Access Systems, New York, 2009.

[7] C. Schurgers, F. Catthoor, M. Engels, "Memory optimization of MAP turbo decoder algorithms", IEEE Transactions on Very Large Scale Integration Systems, Vol. 9, pp. 305–312, 2001.

[8] H. Liu, J.P. Diguet, C. Jego, M. Jezequel, E. Boutillon, "Energy efficient turbo decoder with reduced state metric quantization", IEEE Workshop on Signal Processing and Systems, pp. 237–242, 2007.

[9] M. Martina, G. Masera, "State metric compression techniques for turbo decoder architectures", IEEE Transactions on Circuits and Systems, Vol. 58, pp. 1119–1128, 2011.

[10] J.H. Kim, I.C. Park, "Double-binary circular turbo decoding based on border metric encoding", IEEE Transactions on Circuits and Systems-II: Express Briefs, Vol. 55, pp. 79–83, 2008.

[11] C.H. Lin, C.Y. Chen, A.Y. Wu, "Low-power memory-reduced traceback MAP decoding for double-binary convolutional turbo decoder", IEEE Transactions on Circuits and Systems-I: Regular Papers, Vol. 56, pp. 1005–1016, 2009.

[12] J. Vogt, A. Finger, "Improving the max-log-MAP turbo decoder", Electronics Letters, Vol. 36, pp. 1937–1938, 2000.

[13] S. Papaharalabos, P. Sweeney, B.G. Evans, "Constant log-MAP decoding algorithm for duo-binary turbo codes", Electronics Letters, Vol. 42, pp. 709–710, 2006.

[14] S. Papaharalabos, P.T. Mathiopoulos, G. Masera, M. Martina, "On optimal and near-optimal turbo decoding using generalized max* operator", IEEE Communications Letters, Vol. 13, pp. 522–524, 2009.

[15] J.J. He, Z.F. Wang, H.P. Liu, "Memory-reduced MAP decoding for double-binary convolutional turbo code", Proceedings of the IEEE International Symposium on Circuits and Systems, pp. 469–472, 2010.

[16] C.H. Lin, C.Y. Chen, A.Y. Wu, "Area-efficient scalable MAP processor design for high-throughput multistandard convolutional turbo decoding", IEEE Transactions on Very Large Scale Integration Systems, Vol. 19, pp. 305–318, 2011.

[17] J.H. Kim, I.C. Park, "Bit-level extrinsic information exchange method for double-binary turbo codes", IEEE Transactions on Circuits and Systems-II: Express Briefs, Vol. 56, pp. 81–85, 2009.

[18] E. Boutillon, C. Douillard, G. Montorsi, "Iterative decoding of concatenated convolutional codes: Implementation issues", Proceedings of the IEEE, Vol. 95, pp. 1201–1227, 2007.

[19] M. Zhan, L. Zhou, C. Wang, "A new algorithm without exponential and logarithm computations in iterative decoding for double binary convolutional turbo code", IEEE 7th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–4, 2011.

[20] L.R. Bahl, J. Cocke, F. Jelinek, J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", IEEE Transactions on Information Theory, Vol. 20, pp. 284–287, 1974.

[21] A. Abbasfar, K. Yao, "An efficient and practical architecture for high speed turbo decoders", Proceedings of IEEE Vehicular Technology Conference, pp. 337–341, 2003.

[22] D.S. Lee, I.C. Park, "Low-power log-MAP decoding based on reduced metric memory access", IEEE Transactions Circuits and Systems-I: Regular Papers, Vol. 53, pp. 1244–1253, 2006.

[23] M. Martina, M. Nicola, G. Masera, "A flexible UMTS-WiMax turbo decoder architecture", IEEE Transactions on Circuits and Systems-II: Express Briefs, Vol. 55, pp. 369–373, 2008.