IEICE Electronics Express, Vol.8, No.23, 1996–2001

Memory reduced MAP decoding for double-binary turbo decoder

Ji-Hoon Kim[a)]

Department of Electronics Engineering, Chungnam National University
99 Daehangno, Yuseong-gu, Daejeon 305–764, Korea
a) *jihoonkim@cnu.ac.kr*

Abstract: In this paper, we proposed the branch metric recovery scheme to reduce the memory requirement of MAP decoding for double-binary convolutional turbo code. The proposed technique exploits the fact that huge memory size is required to store the branch metrics in double-binary convolutional turbo decoder. In the proposed method, rather than storing the original branch metrics, the partial terms of the branch metric are stored and the branch metrics are recovered with simple additions. The implementation results based on the proposed technique are presented to show the extremely low overhead compared to the huge memory reduction.

Keywords: MAP decoding, max-log-MAP, double-binary turbo code
Classification: Wireless communication hardware

References

[1] C. Berrou, M. Jezequel, C. Douillard, and S. Kerouedan, "The advantages of nonbinary turbo codes," *Proc. IEEE Inf. Theory Workshop*, pp. 61–63, Sept. 2001.
[2] O. Muller, A. Baghdadi, and M. Jezequel, "ASIP-based multiprocessor SoC design for simple and double binary turbo decoding," *Proc. DATE 2006*, pp. 1330–1335.
[3] J. H. Kim and I. C. Park, "Bit-Level Extrinsic Information Exchange Method for Double-Binary Turbo Codes," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 56, no. 1, pp. 81–85, Jan. 2009.
[4] H. M. Choi, J. H. Kim, and I. C. Park, "Low-Power Hybrid Turbo Decoding Based on Reverse Calculation," *IEICE Trans. Fund. Electron. Comm. Comput. Sci.*, vol. E89-A, no. 3, pp. 782–789, March 2006.
[5] J. Kwak, S. M. Park, and K. Lee, "Reverse tracing of forward state metric in log-MAP and Max-log-MAP decoders with fixed point precision," *IEICE Trans. Commun.*, vol. E86-B, no. 1, pp. 451–455, Jan. 2003.
[6] J. He, Z. Wang, and H. Liu, "Memory-Reduced MAP Decoding for Double-Binary Convolutional Turbo Code," *Proc. ISCAS 2010*, pp. 469–472.

DOI: 10.1587/elex.8.1996
Received October 20, 2011
Accepted November 04, 2011
Published December 10, 2011

IEICE Electronics Express, Vol.8, No.23, 1996–2001

1 Introduction

The turbo code introduced in 1993 is one of the most powerful forward error correction channel codes, and provides near optimal performance approaching the Shannon limit. Recently, the double-binary turbo code has received a great attention and adopted in several mobile radio systems such as DVB-RCS and WiMAX (IEEE 802.16 standard), as it can offer many advantages over the single-binary turbo codes [1].

Although there are many implementations for classical single-binary turbo decoders, there has been little research dedicated to the hardware implementation of the double-binary turbo decoders [2, 3]. Compared to the single-binary turbo code, the double-binary turbo code requires much more memory in decoding. Especially, the size of memory required to perform Soft-Input Soft-Output (SISO) decoding increases hugely to process 2 bits at a time in double-binary turbo decoding.

For double-binary turbo decoding, this paper presents the way to reduce the branch metric memory requirement which leads to the low SISO decoding complexity. Also, for the effectiveness of the proposed technique, the implementation result of the SISO decoder based on the proposed technique is provided with comparison results to other techniques.

2 SISO decoding algorithm for double-binary turbo codes

A typical turbo decoder consists of two SISO decoders serially concatenated via an interleaver. Either the maximum *a posteriori* (MAP) algorithm or the soft-output Viterbi algorithm (SOVA) can be used for SISO decoding. As MAP-based turbo decoders provide much better performance than SOVA-based one, we focus on the MAP-based algorithm in this work. In practical implementation, Max-log-MAP algorithm is commonly employed due to the high complexity of the original MAP algorithm.

In the double-binary turbo codes, the three log-likelihood ratio (LLR) outputs of the k-th symbol (2 bits) are expressed as follows.

$$\begin{aligned}\Lambda_k^{(z)} \cong & \max_{(s_k \to s_{k+1}, z)} \left[\tilde{\alpha}_k(s_k) + \tilde{\gamma}_{k+1}(s_k \to s_{k+1}) + \tilde{\beta}_{k+1}(s_{k+1})\right] \\ & - \max_{(s_k \to s_{k+1}, 00)} \left[\tilde{\alpha}_k(s_k) + \tilde{\gamma}_{k+1}(s_k \to s_{k+1}) + \tilde{\beta}_{k+1}(s_{k+1})\right]\end{aligned} \tag{1}$$

where z belongs to $\phi = \{01, 10, 11\}$, s_k is the state of an encoder at time k, and $\tilde{\alpha}$, $\tilde{\beta}$ and $\tilde{\gamma}$ are the forward, backward, and branch metrics, respectively. The metrics are calculated as expressed in Eq. (2)–(4), where A is the set of states at time $k-1$ connected to state s_k, and B is the set of states at time $k+1$ connected to state s_k.

$$\tilde{\alpha}_k(s_k) \cong \max_{s_{k-1} \in A} \left[\tilde{\alpha}_{k-1}(s_{k-1}) + \tilde{\gamma}_k(s_{k-1} \to s_k)\right] \tag{2}$$

$$\tilde{\beta}_k(s_k) \cong \max_{s_{k+1} \in B} \left[\tilde{\beta}_{k+1}(s_{k+1}) + \tilde{\gamma}_{k+1}(s_k \to s_{k+1})\right] \tag{3}$$

$$\begin{aligned}\tilde{\gamma}_k(s_k \to s_{k+1}) &= \ln\left[P(\mathbf{y}_k|\mathbf{x}_k) \cdot P(u_k = z)\right] \\ &= x_k^{s_1} y_k^{s_1} + x_k^{s_2} y_k^{s_2} + x_k^{p_1} y_k^{p_1} + x_k^{p_2} y_k^{p_2} + L_{e,IN}^{(z)}\end{aligned} \tag{4}$$

DOI: 10.1587/elex.8.1996
Received October 20, 2011
Accepted November 04, 2011
Published December 10, 2011

where z belongs to $\varphi = \{00, 01, 10, 11\}$, u_k is the input symbol consisting of two bits, $P(u_k)$ is *a priori* probability of u_k, and $\mathbf{x}_k$ and $\mathbf{y}_k$ are transmitted and received codewords associated with u_k, respectively. The superscripts p and s denote the parity bits and systematic bits, respectively. In Eq. (4), $L_{e,IN}^{(z)}$ is the extrinsic information received from the other SISO decoder and the code is assumed to be transmitted through an AWGN channel with a noise variance σ^2. Since the Max-log-MAP decoding algorithm is independent of the signal-to-noise ratio (SNR), $L_c = 2/\sigma^2$ is usually set to a constant value, although it can be obtained from channel estimation.

After the turbo decoder has completed a fixed number of iterations or met some other convergence criteria, a final decision on the bits must be made. This is accomplished by computing the LLR of each bit in the couple (A_k, B_k) according to

$$\begin{aligned} \Lambda(A_k) &= \max\left(\Lambda_k^{10}, \Lambda_k^{11}\right) - \max\left(\Lambda_k^{00}, \Lambda_k^{01}\right) \\ \Lambda(B_k) &= \max\left(\Lambda_k^{01}, \Lambda_k^{11}\right) - \max\left(\Lambda_k^{00}, \Lambda_k^{10}\right) \end{aligned} \tag{5}$$

where $\Lambda_k^{00} = 0$. The hard bit decisions can be found by comparing each of these likelihood ratios to a threshold.

3 Memory requirement of SISO decoder with sliding window

The sliding window technique is effective in reducing the memory size required to store metric values. A large frame is split into a number of small windows and the MAP decoding is applied to each window independently [4]. Fig. 1 shows the conventional sliding window diagram where forward metrics are calculated prior to backward metrics. In the sliding window technique, however, the initial values of the backward metrics at the border of each window are required. To obtain the reliable initial values of each window, the dummy calculation is performed for the backward metrics as shown in Fig. 1. If the window size is sufficiently long, the initial values obtained by the dummy calculation do not degrade performance. In this work, the window size is set to 32.

As explained, several memory blocks to hold forward metrics and branch metrics are required. Table I indicates the memory required in the conventional double-binary SISO decoder. In this work, the received input LLRs are represented in 4 bits, branch metrics and state metrics are represented in 9 bits and 10 bits, respectively. Since two processes, forward metric calculation and LLR calculation, require the forward metrics as shown in Fig. 1, the forward metric memory consists of two memory banks. In the hardware implementation, the forward metrics and backward metrics are normalized by subtracting the value of state 0, from other metrics at the same trellis stage in order to avoid overflow in state metrics. Accordingly, we can eliminate the need to store the metric value of state 0. As shown in Fig. 2, since the SISO decoder takes two systematic bits and two parity bits as inputs, the number of possible branch metrics is 16 in the double-binary turbo decoder while the number of possible branch metrics is 4 in the classical single-binary

DOI: 10.1587/elex.8.1996
Received October 20, 2011
Accepted November 04, 2011
Published December 10, 2011

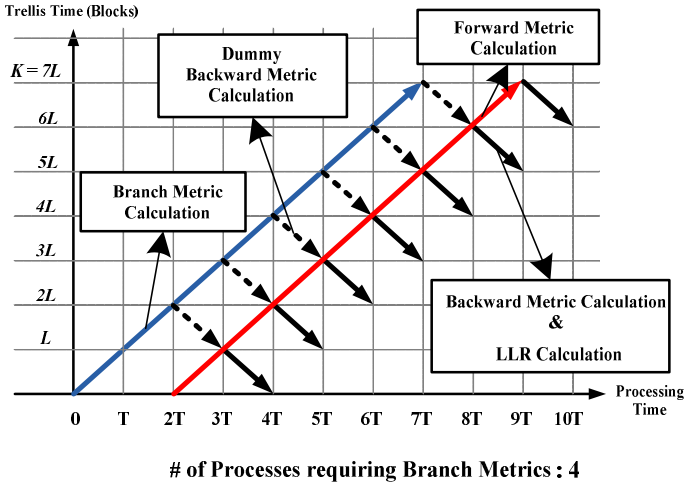

Fig. 1. Sliding window diagram with dummy metric calculation

turbo decoders for W-CDMA and CDMA2000. Also, as illustrated in Fig. 1, four processes require the accesses to the branch metric memory on their own window processing, simultaneously. Therefore, in hardware implementation, the branch metric memory consists of four memory banks [3, 4]. In double-binary turbo decoder implementation, since the branch metric memory increases four times compared to the classical single-binary turbo decoder, the total memory size required in the SISO decoder increases hugely as denoted in Table I.

4 Proposed branch metric memory reduction techniques

There are several approaches to reduce the forward metric memory in classical single-binary turbo decoder [4, 5]. However, mainly due to the branch metric memory, increased memory requirement in double-binary turbo decoder implementation leads to huge area occupation and high power consumption, which are not appropriate for mobile applications. To lower the complexity of the SISO decoder, we propose three techniques to reduce the branch metric memory requirement as described in Fig. 2. Basically, the proposed techniques does not store whole 16 branch metric values, but stores only partial values required to recover the branch metrics *on demand*. From Eq. (4), we can obtain the following relation and used in Proposed II.

$$\tilde{\gamma}_k = (y_k^{s_1} + y_k^{s_2}) + x_k^{p_1} y_k^{p_1} + x_k^{p_2} y_k^{p_2} + (L_{e,IN}^{(z)} + L_i^{(z)}) \tag{6}$$

where $L_i^{(z)}$ is the intrinsic information defined as follows.

$$L_i^{01} = -2y^{s_2}, \qquad L_i^{10} = -2y^{s_1}, \qquad L_i^{11} = -2y^{s_1} - 2y^{s_2} \tag{7}$$

Therefore, by storing only essential sub-metrics as shown in Fig. 2, we can significantly reduce the memory size required to store branch metrics at the expense of the simple calculation expressed in Eq. (6). We can reduce the branch memory size further by keeping the bit-level extrinsic information and

DOI: 10.1587/elex.8.1996
Received October 20, 2011
Accepted November 04, 2011
Published December 10, 2011

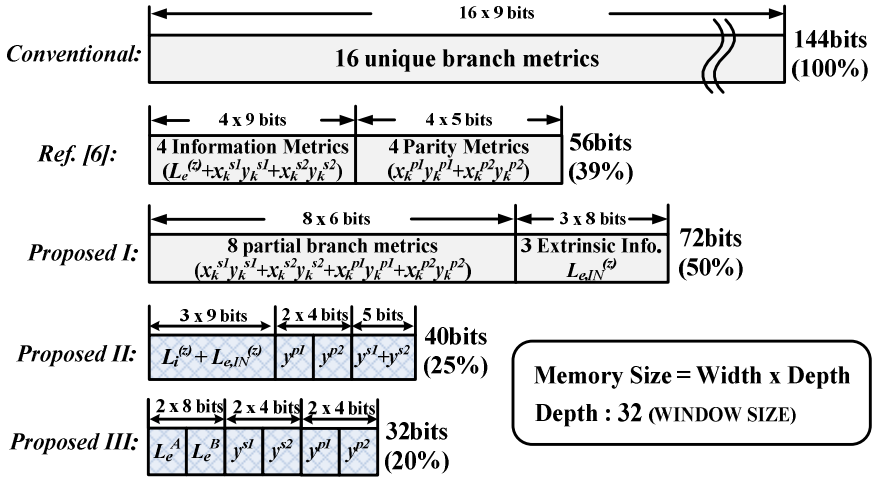

Fig. 2. Branch metric memory width comparison

the symbol-level extrinsic information values are recovered by the bit-level extrinsic information values later as indicated in Fig. 2 as *Proposed III*.

In double-binary turbo decoding, the bit-level extrinsic information can be expressed as follows [4].

$$\begin{aligned} L_{be}^{A} &= \ln\left(\frac{p(u_k=10)+p(u_k=11)}{p(u_k=00)+p(u_k=01)}\right) \\ &= \ln\left(\frac{\exp[L_e^{10}]+\exp[L_e^{11}]}{1+\exp[L_e^{01}]}\right) \simeq \max\left(L_e^{10}, L_e^{11}\right) - \max\left(0, L_e^{01}\right) \\ L_{be}^{B} &= \ln\left(\frac{p(u_k=01)+p(u_k=11)}{p(u_k=00)+p(u_k=10)}\right) \\ &= \ln\left(\frac{\exp[L_e^{01}]+\exp[L_e^{11}]}{1+\exp[L_e^{10}]}\right) \simeq \max\left(L_e^{01}, L_e^{11}\right) - \max\left(0, L_e^{10}\right) \end{aligned} \tag{8}$$

The symbol-level extrinsic information required to recover the branch metrics can be obtained by simple bit-to-symbol conversion introduced in [3].

5 Implementation results

The only overhead of the proposed technique is a few additions for branch metric recovery. With the quantization presented in Section 3, the turbo decoder based on the proposed technique was described in Verilog-HDL and synthesized with a 0.13μm 1-poly 6-metal standard CMOS process. To mitigate the long computation time of double-binary SISO decoding, retiming and migrating the common operator techniques are applied. Accordingly, the turbo decoder can operate at the high operating frequency of 200 MHz, which leads to higher peak throughput due to the reduced critical path delay. Also, for the given throughput specification, the supply voltage can be lowered leading lower operating frequency of the decoder and reduced power consumption. Table I indicates the single-port SRAM size required in a SISO decoder and also provides the hardware overhead for branch metric recovery. In the proposed techniques, the total memory size required in a SISO decoder can be reduced by 40.2%, 58.1%, and 62.6% at the expense of 3.1%, 4.8%, and

DOI: 10.1587/elex.8.1996
Received October 20, 2011
Accepted November 04, 2011
Published December 10, 2011

Table I. Single-port SRAM size and logic overhead required for a SISO decoder

	Forward Metric Memory	Branch Metric Memory	Total Memory	Recovery Circuit Gate Count
*Conventional**	2 banks, 32 * (10 * 7) bits / bank	4 banks, 32 * 144 bits / bank	22912 bits (100%)	N.A.
	4480 bits	18432 bits		
Ref. [6]	2 banks, 32 * (10 * 7) bits / bank	4 banks, 32 * 56 bits / bank	11648 bits (50.8%)	2.42k gates
	4480 bits	7168 bits		
Proposed I	2 banks, 32 * (10 * 7) bits / bank	4 banks, 32 * 72 bits / bank	13696 bits (59.8%)	1.96k gates
	4480 bits	9216 bits		
Proposed II	2 banks, 32 * (10 * 7) bits / bank	4 banks, 32 * 40 bits / bank	9600 bits (41.9%)	3.07k gates
	4480 bits	5120 bits		
Proposed III	2 banks, 32 * (10 * 7) bits / bank	4 banks, 32 * 32 bits / bank	8576 bits (37.4%)	5.72k gates
	4480 bits	4096 bits		

* The gate count of the conventional SISO decoder : 64.1k gates.

8.9% logic overhead. Compared to the branch memory reduction technique in [6] where the branch metrics are expressed as the sum of the information metric and the parity metric, three proposed techniques provide additional trade-off between the degrees of memory reduction versus recovery circuit complexity. Since the critical path is laid on the calculation of forward and backward state metrics, the recovery logic for branch metrics does not affect the critical path delay. Also, even if the other sliding window is adopted, the proposed technique is still effective because the proposed technique is not dependent on the sliding window type [4].

6 Conclusion

This paper presents the technique to minimize the branch metric memory, which takes significant silicon area in nonbinary turbo decoder implementation because of the increased number of branch metrics. By exploit the facts that the branch metrics can be decomposed and recovered on demand, the total memory size required in a double-binary SISO decoder can be reduced by up to 62.6% at the negligible logic overhead.

Acknowledgments

This study was financially supported by research fund of Chungnam National University in 2011.

DOI: 10.1587/elex.8.1996
Received October 20, 2011
Accepted November 04, 2011
Published December 10, 2011