An Efficient 4-D 8PSK TCM Decoder Architecture

Jinjin He, Zhongfeng Wang, *Senior Member, IEEE*, and Huping Liu, *Senior Member, IEEE*

***Abstract*—This paper presents an efficient architecture for a 4-D eight-phase-shift-keying trellis-coded-modulation (TCM) decoder. First, a low-complexity architecture for the transition metric unit is proposed based on substructure sharing. This scheme significantly reduces the required computation without degrading the performance. Then, a new hybrid T-algorithm for a Viterbi decoder is developed by applying a T-algorithm on both branch metrics (BMs) and path metrics (PMs). TCM encoders usually employ high-rate convolutional codes that yield many more transition paths per state than low-rate codes do. This makes it feasible to purge unnecessary additions by applying the T-algorithm on BMs. Applying the T-algorithm on BMs instead of PMs allows one to move the "search-for-the-optimal" operation out of the add–compare–select-unit (ACSU) loop. Hence, the clock speed will not be affected. In addition, by combining the T-algorithm on BMs and the T-algorithm on PMs, the hybrid T-algorithm can reduce the computations required with the conventional T-algorithm on PMs by as much as 50%.**

***Index Terms*—Add–compare–select unit (ACSU), T-algorithm, trellis-coded modulation (TCM), Viterbi decoder (VD), VLSI.**

I. Introduction

TRELLIS-CODED modulation (TCM) is a technique that combines error-correcting coding and modulation in digital communications [1]. The scheme gains noise immunity over uncoded transmission without expanding the signal bandwidth or increasing the transmission power. By partitioning signal set into groups, TCM uses signal mapping to increase the Euclidean distance, rather than the Hamming distance, between codes [2], [3]. In 1-D TCM systems, 2^{m+1} constellation points are usually used for transmitting m bits. TCM schemes using multidimensional constellations have been shown to further improve the error performance. In [4], a systematic approach to partition multidimensional signals is proposed. Although iterative error correction codes (ECCs) such as turbo codes [5] and low-density parity-check codes [6] are popular due to their near-Shannon-limit decoding performance, TCM has the advantage of low decoding latency over these iterative codes.

Reducing complexity while maintaining a high speed and a good performance is a challenging issue for multidimensional TCM decoders. In fact, only a few existing articles have discussed the implementation aspects of TCM decoders [7]–[10], among which only [10] is about 4-D eight-phase-shift-keying (8PSK) TCM. In [10], a general architecture for a 4-D 8PSK TCM decoder is presented. The auxiliary-trellis-based transition metric unit (TMU) and a conventional Viterbi decoder (VD) are employed in this design. In this paper, we propose a new low-complexity architecture for a 4-D 8PSK TCM decoder. The main contributions of the proposed scheme include an efficient architecture for the TMU to significantly reduce the computational complexity compared with existing schemes without degrading the performance and a hybrid T-algorithm and architecture for the VD. The hybrid T-algorithm applies the T-algorithm on both branch metrics (BMs) and path metrics (PMs), which results in significant reduction of the required computation compared with the conventional T-algorithm applied on PMs only.

The rest of this paper is organized as follows. Background information, including the 4-D 8PSK TCM system, TCM decoder, and design issues, will be introduced in Section II. In Section III, the low-complexity high-speed TMU is proposed. Section IV focuses on the architecture design of the VD with the hybrid T-algorithm. Section V provides FPGA implementation results, followed by conclusions in Section VI.

II. Background

A. Four-Dimensional TCM 8PSK System

The convolutional encoder and constellation mapper for the 4-D 8PSK TCM system designed in this paper can be found in [11]:

1) 64 trellis states;
2) rate-3/4 convolutional code;
3) rate of modulation: $\mathrm{Rm} = m/(m+1)$, where $m = 8$, 9, 10, or 11.

Construction of optimal TCM codes is beyond the scope of this paper, and the related details can be found in [4]. In our design, the decoders for the four different modulation rates are integrated into a reconfigurable 4-D 8PSK TCM decoder, in which common modules can be shared. Among the four modulation rates, the case of $\mathrm{Rm} = 11/12$ is the most complex one, which dominates the overall computational complexity. Therefore, for simplicity, we only discuss in detail the case of $\mathrm{Rm} = 11/12$; the design processes for the other three cases are similar, and we only provide some conclusions.

From the VLSI implementation perspective, a TCM encoder consists of four main parts: a differential encoder, a binary convolutional encoder, a multidimensional mapper, and a modulator (e.g., 8PSK), as shown in Fig. 1.

The differential encoder is shown in Fig. 2, where FA represents a full adder and D represents a D flip-flop. Unlike normal

Manuscript received July 18, 2008; revised November 26, 2008. First published June 05, 2009; current version published April 23, 2010. This work was supported in part by NASA under Grant NNX07AF84G and Grant NNX06AC23G.

J. He and H. Liu are with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331 USA (e-mail: hejin@eecs.orst.edu; zwang@eecs.orst.edu; hliu@eecs.orst.edu).

Z. Wang is with the Broadcom Corporation, Irvine, CA 92617 USA (e-mail: zfwang@broadcom.com).

Digital Object Identifier 10.1109/TVLSI.2009.2015325

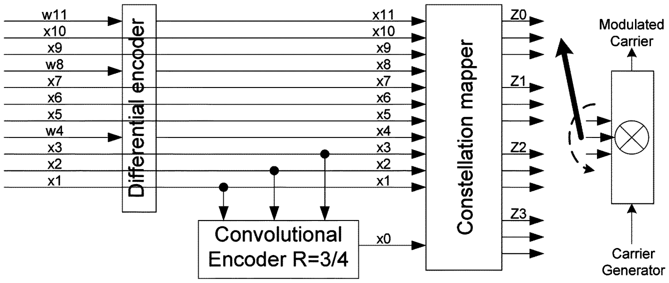


Fig. 1. Encoder of the 4-D 8PSK TCM for the case of $\mathrm{Rm} = 11/12$.

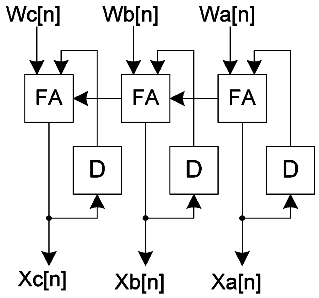


Fig. 2. Differential encoder.

differential encoding, the encoder here performs the mod-8 operation on the current input data and the data buffered in the delay elements.

The details of the convolutional encoder will be discussed in Section IV. The mapping equation is expressed in (1), where

$$Z_i, i = 0, \ldots, 3$$

corresponds to four sets of 8PSK symbols and

$$x^{(k)}, i = 0, \ldots, 11$$

represents 12 binary input bits [10]

$$\begin{pmatrix} Z_0 \\ Z_1 \\ Z_2 \\ Z_3 \end{pmatrix} = \left\{ \left(4x^{(11)} + 2x^{(8)} + x^{(4)}\right) \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} + 4 \begin{pmatrix} 0 \\ x^{(10)} \\ x^{(9)} \\ x^{(10)} + x^{(9)} + x^{(7)} \end{pmatrix} + 2 \begin{pmatrix} 0 \\ x^{(6)} \\ x^{(5)} \\ x^{(6)} + x^{(5)} + x^{(3)} \end{pmatrix} + \begin{pmatrix} 0 \\ x^{(2)} \\ x^{(1)} \\ x^{(2)} + x^{(1)} + x^{(0)} \end{pmatrix} \right\} (\mathrm{mod}8). \quad (1)$$

Correspondingly, the TCM decoder includes a TMU; a soft-decision VD, which provides optimal decoding performance for convolutional codes; a demapper; and a differential decoder (DFD), as shown in Fig. 3. The delay chain is used to buffer the paths associated with each BM generated by the TMU. The "path" refers to the 12 bits of information from the four dimensions of 8PSK signals. Since there are only 16 BMs [10] in this case, using a 4-b index is more efficient than using the 12-b path information inside the VD. When the VD outputs the decoded index, the corresponding path will be sent to the demapping unit (DMU) and DFD for further processing. Implementation of the DMU and DFD is straightforward; the most challenging and hardware-consuming parts are the VD and TMU.

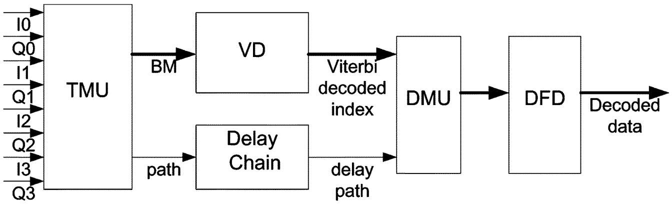


Fig. 3. Four-dimensional 8PSK TCM decoder diagram.

B. TMU

The main difference between the trellis diagrams of a TCM system and a normal convolutional coded system is that, in the former case, there are parallel paths in state transitions, which extend from one state and end at the same state. As shown in Fig. 1, for any encoded sequence "$x3x2x1x0$," there are 256 different 4-D 8PSK signal groups (combinations of $x11 - x4$). Therefore, before performing Viterbi decoding, a TMU is used to find the optimal paths among each group of the parallel paths as the BMs. Theoretically, the BM for a soft-decision VD in multidimensional TCM is represented as the sum of the Euclidian distance from each dimension of the signal sets [4]. In a multidimensional TCM decoder, the TMU could be very complex. For example, as shown in Fig. 1, each BM has 256 candidates. The structure of the auxiliary trellis has been presented in [10] to remove the redundant operations in the TMU by sharing intermediate results among all the BM candidates. However, the computational complexity is still considerably high. The details of the auxiliary-trellis-based structure will be discussed in Section III.

C. VD

A typical functional diagram of the VD is shown in Fig. 4. First, the BMs from the TMU are fed into the add–compare–select unit (ACSU) that recursively computes PMs and outputs decision bits for each possible state transition. Then, the decision bits generated by the ACSU are stored in and retrieved from the survivor-path memory unit (SMU) in order to decode the source bits along the final survivor path. The PMs of the current iteration are stored in the PM unit (PMU) and read out for use in the next iteration.

For VD implementation, research efforts have focused on ACSU and SMU. ACSU implementation is critical because the feedback loop makes it the bottleneck for high-speed applications. Additionally, as the constraint length K increases, the number of states and computation in ACSU increase exponentially. Several algorithms such as the M-algorithm [12] and the T-algorithm [13], [14] have been proposed to reduce the computation in ACSU. In the M-algorithm, this is achieved by

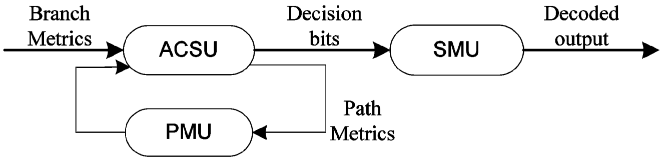


Fig. 4. Functional diagram of a VD.

keeping M optimal states and purging the rest. In the T-algorithm, a threshold T is set, and the difference between each PM and the optimal one is calculated. The algorithm compares the differences with T; only the states with a difference that is less than T survive and are used for the calculation in the next cycle. Both methods have the potential for significant power savings. In practical implementation, however, the T-algorithm is more popular than the M-algorithm for two reasons: 1) It can adaptively adjust the number of purged states according to run-time channel conditions, and 2) it has lower computational complexity since the T-algorithm only needs to find out the optimal PM (maximum or minimum value, depending on different implementations), while the M-algorithm involves a sorting process for $2^{(k-1)}$ values. However, the T-algorithm still requires the search of the optimal PM in the ACS loop. The extra comparison operation will affect the clock speed of the entire design. Although it is possible to implement a fully parallel structure for the comparison operation, it will cause significant hardware overhead. Recently, several novel VLSI architectures are proposed to remove or reduce the loop latency with reasonable hardware overhead. For example, in [19], instead of finding the exact optimal PM, an estimated value is obtained. Then, a compensation scheme outside the loop is used to adjust the estimated optimal PM.

There are two basic schemes for SMU implementation: the register-exchange (RE) scheme and the trace-back (TB) scheme. Because of the maximum-likelihood (ML) path searching in SMU, the VD usually causes a decoding latency proportional to the constraint length K. The RE scheme is well suited for low-latency high-speed applications, while the TB scheme consumes less power. Although there exist extensive works on reducing the latency of the TB scheme [15]–[18], none of them could achieve the latency of the RE scheme, even at the expense of a huge power consumption overhead. Another advantage of using the RE scheme is that the T-algorithm on PM can be efficiently exploited in RE design. Thus, the RE scheme is adopted in our design.

The algorithms discussed previously are general solutions for VDs. TCM schemes usually employ high-rate codes (e.g., the rate-3/4 codes in our case), and the unique properties of high-rate codes can be explored to construct more efficient architectures for the corresponding VD.

III. Low-Complexity TMU Design

There are two common ways for the design of TMU: 1) ROM-based approach and 2) arithmetic (online) computation-based solution. In order to reduce hardware, as well as to pipeline the computation unit for high clock speeds, the second approach is adopted in our design. Exploiting the auxiliary trellis [10], we design two efficient methods to reduce the computational complexity.

A. Simplified Computation of Euclidian Metrics

Given a received 8PSK symbol (I_r, Q_r) and a target constellation point A: (I_s, Q_s) $(s = 0, 1, 2, \ldots, 7$ for 8PSK signals), the Euclidian distance is computed as

$$\begin{aligned} d_s &= (I_r - I_s)^2 + (Q_r - Q_s)^2 \\ &= \left(I_r^2 + I_s^2 + Q_r^2 + Q_s^2\right) - 2I_rI_s - 2Q_rQ_r. \end{aligned} \tag{2}$$

Note that, for all d_s, $(I_r^2 + I_s^2 + Q_r^2 + Q_s^2)$ remains the same. Finding the minimum of $(d_0, d_1, d_2, \ldots, d_7)$ (the closest constellation point to the received signal) is equivalent to finding the maximum of $(I_rI_0 + Q_rQ_0, I_rI_1 + Q_rQ_1, I_rI_2 + Q_rQ_2, \ldots, I_rI_7 + Q_rQ_7)$. Thus, we only need to consider the following distance:

$$d'_s = I_rI_s + Q_rQ_s. \tag{3}$$

Furthermore, because all the possible constellation points are equally spaced on a circle with the phase of $s\pi/4$, $s = 0, 1, 2 \ldots, 7$, it has been found in [10] that I_s and Q_s satisfy the following conditions:

$$\begin{aligned} I_s &= -I_{(s+4)\mathrm{mod}8} \\ Q_s &= -Q_{(s+4)\mathrm{mod}8}. \end{aligned}$$

Hence, only four sets of (3) need to be calculated. The Euclidean metrics $C_i (i = 0, 1, 2, 3)$ are computed as

$$\begin{aligned} C_0 &= |d'_0| = |I_r| \\ C_1 &= |d'_1| = |(I_r + Q_r) \times 0.707| \\ C_2 &= |d'_2| = |Q_r| \\ C_3 &= |d'_3| = |(Q_r - I_r) \times 0.707|. \end{aligned} \tag{4}$$

It is clear from (4) that only two multiplications are required.

B. Sharing of BM Computation

In the 4-D 8PSK TCM decoder, 16 BMs are selected from 4096 (i.e., 8^4) candidates every cycle. From Fig. 1, the 16 BMs are related to the 16 combinations of $x3$, $x2$, $x1$, and $x0$. Let BM 0000 denote the branch metrics related to $(x3, x2, x1, x0) = (0, 0, 0, 0)$, let BM 0001 denote the BM related to $(x3, x2, x1, x0) = (0, 0, 0, 1)$, and so on. Each candidate is the sum of the four Euclidian metrics of the received signal set (Z_0, Z_1, Z_2, Z_3). By applying the aforementioned simplified Euclidian metrics to BMs, the number of candidates is reduced from 4096 to 256 (i.e., 4096/16) since the number of Euclidian metrics from each signal set is reduced by half. Now, each BM is selected among only 16, rather than 256, candidates. A straightforward implementation requires three serial addition stages to compute the candidates and several comparison steps to find the BMs. Since the results at each addition stage can be shared for further computation, using the auxiliary trellis [10] can eliminate redundant additions.

Fig. 5 is an example that illustrates the process of computing BM 0000. From the left to the right, C_i $(i = 0, 1, 2, 3)$ in each step denotes the Euclidean metrics of the received symbol from

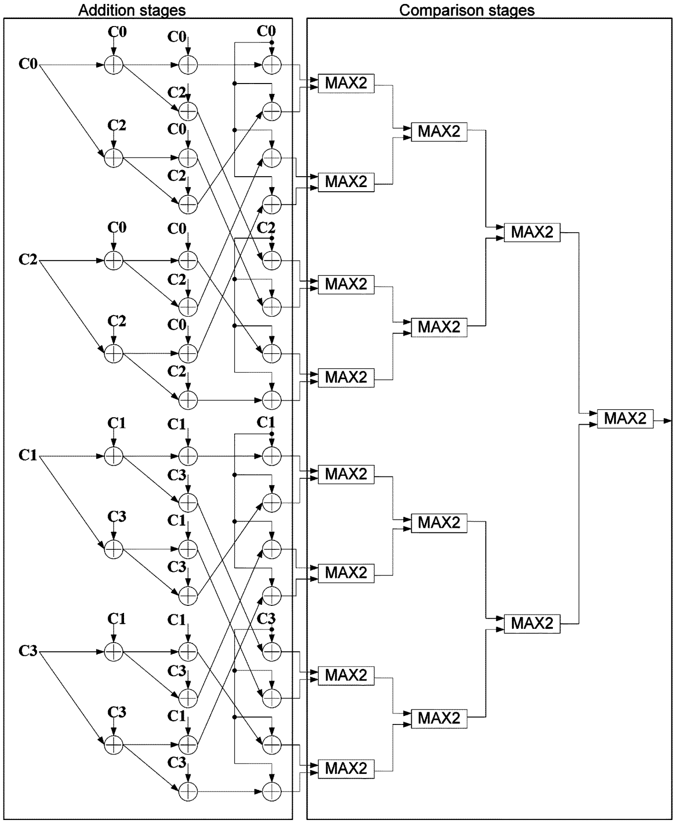


Fig. 5. Example of auxiliary trellis to simplify the BM calculation.

TABLE I
GROUPING OF THE CANDIDATES FOR BM 0000

	Z_0	Z_1	Z_2	Z_3		Z_0	Z_1	Z_2	Z_3
G1	C_0	C_0	C_0	C_0	G2	C_1	C_1	C_1	C_1
	C_0	C_2	C_2			C_1	C_3	C_3	
	C_2	C_2	C_0			C_3	C_3	C_1	
	C_2	C_0	C_2			C_3	C_1	C_3	
G3	C_0	C_0	C_2	C_2	G4	C_1	C_1	C_3	C_3
	C_0	C_2	C_0			C_1	C_3	C_1	
	C_2	C_2	C_2			C_3	C_3	C_3	
	C_2	C_0	C_0			C_3	C_1	C_1	

Z_0, Z_1, Z_2, and Z_3, respectively. The same rule applies to all equations and figures in the rest of this paper. The 16 candidates for BM 0000 are $C_0+C_0+C_0+C_0$, $C_1+C_1+C_1+C_1$, $C_0+C_0+C_2+C_2$, $C_1+C_1+C_3+C_3$, $C_0+C_2+C_0+C_2$, $C_1+C_3+C_1+C_3$, $C_0+C_2+C_2+C_0$, $C_1+C_3+C_3+C_1$, $C_2+C_2+C_2+C_2$, $C_3+C_3+C_3+C_3$, $C_2+C_2+C_0+C_0$, $C_3+C_3+C_1+C_1$, $C_2+C_0+C_2+C_0$, $C_3+C_1+C_3+C_1$, $C_2+C_0+C_0+C_2$, $C_3+C_1+C_1+C_3$.

By using the auxiliary trellis, the total number of additions is reduced from 768 (i.e., 3 × 16 × 16) with the straightforward implementation to 336 (i.e., $4^2+4^3+4^4$). However, the number of comparisons remains the same. In this paper, we propose a new approach that further reduces the number of additions by half. In addition, about two-thirds of the comparisons are removed.

The key idea of the improvement is as follows. Let us again take BM 0000 as an example. In the auxiliary-trellis approach, as shown in Fig. 5, to compute each candidate BM, three serial addition stages are performed to compute 16 candidates in parallel. Then, one survival BM is chosen from the 16 candidates. A careful examination of the added Euclidian distances reveals that, in the last addition stage, there are only four different values to be added onto the 16 candidates, as indicated in Table I.

This property can be exploited to reduce the number of additions in the last stage by rearranging the calculation flow in Fig. 5. In the proposed approach, the 16 candidates are divided at the end of the second addition stage into four groups, which is represented by Gk $(k=0,1,2,3)$ in Table I. The four candidates in each group are supposed to add the same Euclidian metric of Z_3 later. Before the last addition operation, one survival candidate is chosen from each group. Then, in the last addition stage, only four additions, instead of 16, are required. Based on the two-step comparison method, an algorithmic strength reduction scheme is introduced, which significantly reduces the needed comparisons. As shown in Fig. 6, the computation procedure is rearranged: Two serial comparison operations are inserted between the second and third addition operations. Furthermore, the survival candidates after comparison step 1 can be shared by four BM computations (i.e., BM 0000, BM 0001, BM 1000, and BM 1001 share the same survival candidates, as listed in Tables I–IV).

The other 12 BMs have the same property and can be categorized into three clusters; in each cluster, four BMs share the survivor candidates after comparison step 1. The total number of additions of the proposed architecture is 4^2+4^3 from the first two addition stages plus 4 × 16 from the third addition stage. Meanwhile, the comparison operations are also reduced. The computational complexities of the three methods are summarized in Table V.

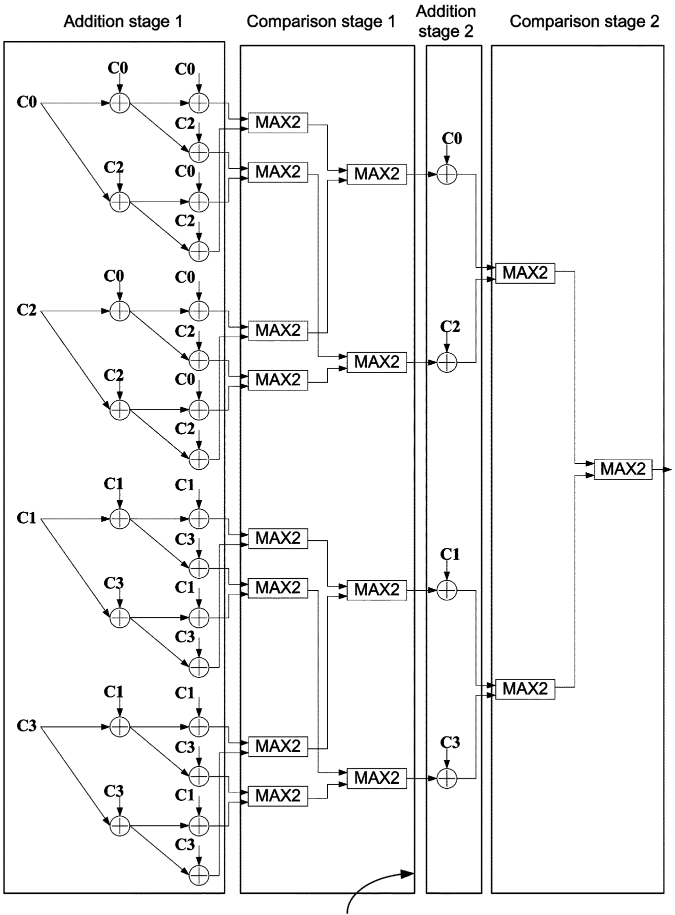
The same structure can be extended to the cases of $\mathrm{Rm}=10/11$ and $\mathrm{Rm}=9/10$, in which the same method could be employed. For the case of $\mathrm{Rm}=8/9$, since there are only two candidates for each BM, the structure based on the conventional auxiliary trellis [10] is as efficient as the proposed approach. A detailed comparison of computational complexity is provided in Table VI.

IV. HYBRID T-ALGORITHM FOR VD

A. T-Algorithm on BMs

In the 4-D 8PSK TCM system, signals with different modulation rates use the same convolutional code, as shown in Fig. 7.

Unlike low-rate codes (e.g., 1/2 and 1/3) that have two incoming paths for each state, high-rate convolutional codes generate many more paths in state transitions (eight in the aforementioned case due to the 3 input bits). When applying Viterbi decoding on these codes, more BMs are involved (16 in the aforementioned case because of the 4 output bits). Motivated by the conventional T-algorithm, we found that eliminating redundant additions is an effective way to reduce the computational complexity. In our case, since the number of BMs is so large, BMs also play a very important role in determining the overall complexity. Since PMs and BMs are evenly distributed in ACSU calculations, purging a BM is equivalent to purging four PMs in terms of equivalent number of additions being eliminated. Therefore, the T-algorithm could be applied on BMs instead of



The 4 survival candidates form this stage can be shared by BM computations for X=0001, 1000, and 1001

Fig. 6. Computation of BM 0000 using the proposed two-step comparison method.

TABLE II
GROUPING OF THE CANDIDATES FOR BM 0001

	Z_0	Z_1	Z_2	Z_3		Z_0	Z_1	Z_2	Z_3
G1	C_0	C_0	C_0	C_1	G2	C_1	C_1	C_1	C_2
	C_0	C_2	C_2			C_1	C_3	C_3	
	C_2	C_2	C_0			C_3	C_3	C_1	
	C_2	C_0	C_2			C_3	C_1	C_3	
G3	C_0	C_0	C_2	C_3	G4	C_1	C_1	C_3	C_0
	C_0	C_2	C_0			C_1	C_3	C_1	
	C_2	C_2	C_2			C_3	C_3	C_3	
	C_2	C_0	C_0			C_3	C_1	C_1	

TABLE III
GROUPING OF THE CANDIDATES FOR BM 1000

	Z_0	Z_1	Z_2	Z_3		Z_0	Z_1	Z_2	Z_3
G1	C_0	C_0	C_0	C_2	G2	C_1	C_1	C_1	C_3
	C_0	C_2	C_2			C_1	C_3	C_3	
	C_2	C_2	C_0			C_3	C_3	C_1	
	C_2	C_0	C_2			C_3	C_1	C_3	
G3	C_0	C_0	C_2	C_0	G4	C_1	C_1	C_3	C_1
	C_0	C_2	C_0			C_1	C_3	C_1	
	C_2	C_2	C_2			C_3	C_3	C_3	
	C_2	C_0	C_0			C_3	C_1	C_1	

TABLE IV
GROUPING OF THE CANDIDATES FOR BM 1001

	Z_0	Z_1	Z_2	Z_3		Z_0	Z_1	Z_2	Z_3
G1	C_0	C_0	C_0	C_3	G2	C_1	C_1	C_1	C_0
	C_0	C_2	C_2			C_1	C_3	C_3	
	C_2	C_2	C_0			C_3	C_3	C_1	
	C_2	C_0	C_2			C_3	C_1	C_3	
G3	C_0	C_0	C_2	C_1	G4	C_1	C_1	C_3	C_2
	C_0	C_2	C_0			C_1	C_3	C_1	
	C_2	C_2	C_2			C_3	C_3	C_3	
	C_2	C_0	C_0			C_3	C_1	C_1	

TABLE V
COMPUTATIONAL COMPLEXITY COMPARISON FOR THE $\mathrm{RM} = 11/12$ CASE

Methods	Additions	Comparisons
Straightforward	768	240
Conventional auxiliary	336	240
Proposed 2-step comparison	144	96

TABLE VI
COMPUTATIONAL COMPLEXITY COMPARISON FOR ALL CASES

Rm		8/9	9/10	10/11	11/12
Straightforward implement	Additions	96	192	384	768
	Comparisons	16	48	112	240
Conventional auxiliary trellis	Additions	56	112	208	336
	Comparisons	16	48	112	240
Proposed 2-Step comparison	Additions	56	80	112	144
	Comparisons	16	32	64	96

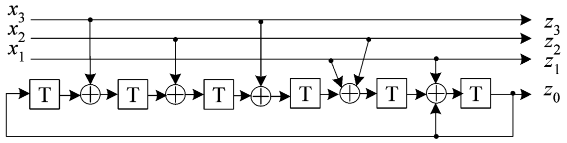


Fig. 7. Rate-3/4 convolutional encoder.

PMs. The process is the same as in the conventional T-algorithm applied on PMs: first, finding the optimal BM—the maximum value of all 16 BMs in our case—and setting up a threshold, and then comparing the difference between each BM and the maximum value. If the difference is larger than the threshold, the corresponding BM is purged, which means that, in ACSU, the additions involving the purged BMs are not performed.

The benefit of applying the T-algorithm on BMs is obvious. In our case, finding the maximum value of 16 BMs is more efficient than searching for the maximum value of 64 PMs, not only because the number of BMs is less than the number of PMs but also due to the fact that each BM is the sum of the Euclidian metrics from four signal sets. The maximum value of BMs is equivalent to the sum of the maximum Euclidian metrics from each signal set. Finding four maximum values and then adding them is much more convenient than finding the maximum value of 16 arbitrary data. Additionally, the processes of finding the maximum value of BMs and calculating 16 BMs could take place in parallel since the maximum BM is derived directly from the Euclidian metrics. This property can be explored to eliminate the latency caused by the process of calculating the maximum value in a normal case that needs to obtain the values of all BMs first.

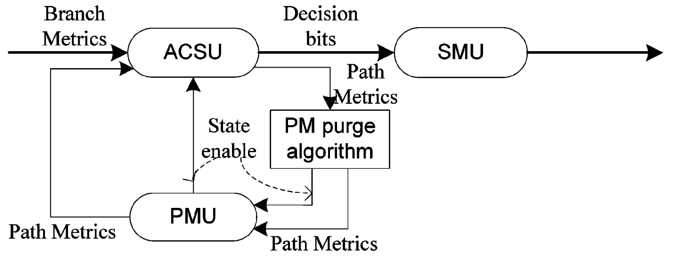


Fig. 8. Functional diagram of a VD with the T-algorithm on PM.

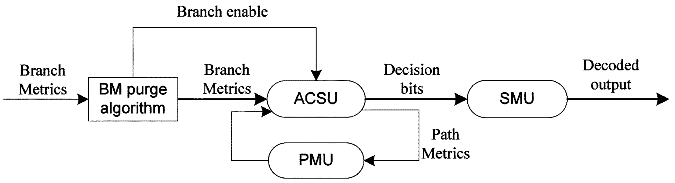


Fig. 9. Functional diagram of a VD with the T-algorithm on BM.

A more important advantage of the proposed method is the improvement of clock speed. The computation of the BMs is a feedforward process that can be theoretically pipelined to achieve a clock speed as high as needed, and there is no additional process or calculation in the ACS loop. By comparing Figs. 4, 8, and 9, it is easily seen that the VD with the T-algorithm applied on BMs can maintain almost the same clock speed as the regular VD in Fig. 4.

We have pointed out in Section II that the conventional T-algorithm applied on PM could perform well for the RE scheme. When a state is purged in ACSU, the corresponding registers in SMU are not updated; thus, the power consumption is reduced. Generally, the scheme can be implemented by clock gating. For the T-algorithm on BMs, when the BMs involved for calculating a new PM are all purged and the PM cannot get a valid value, the corresponding state can be treated as "purged." Again, the registers associated with the "purged" state will not be updated in SMU.

It is necessary to examine the bit-error-rate (BER) performance and computational complexity of the conventional VD with the T-algorithm applied on PMs and the proposed VD with the T-algorithm applied on BMs. Figs. 10 and 11 show the BER performance of these two methods for the case of $\mathrm{Rm} = 11/12$. When the threshold drops to below 0.3, a dramatic performance loss for both cases is observed; at a threshold of 0.3, the BER performance of the proposed scheme is slightly better than that of the conventional scheme with the T-algorithm applied on PMs.

The corresponding average computational complexity for the case with a threshold of 0.3 is shown in Fig. 12. The average computational complexity refers to the number of additions in each cycle. For a regular VD, the computational complexity is 64 (number of states) $\times$ 8 (number of input BMs per state) $= 512$. From the simulation result shown in Fig. 12, we observe that, as the signal-to-noise ratio (SNR) changes, the number of purged computation changes accordingly in the case of the T-algorithm on PMs, whereas in the case of the T-algorithm on BMs, the number of purged computation remains almost the same, regardless of the channel condition. It may be noted that the computational complexity of the proposed T-algorithm applied on BMs looks more like that of the M-algorithm rather than the conventional T-algorithm. Therefore, when the SNR is low, the computational complexity of the proposed scheme is lower than that of the conventional scheme. In the high-SNR region, however, the conventional scheme has a lower complexity than the proposed one.

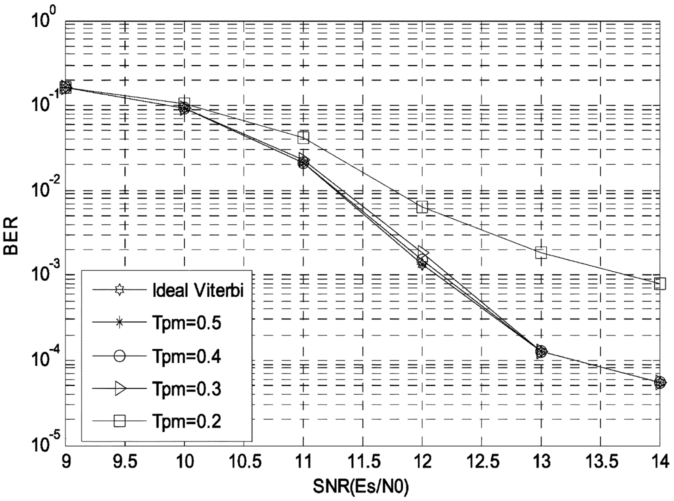


Fig. 10. BER performance of the T-algorithm on PMs.

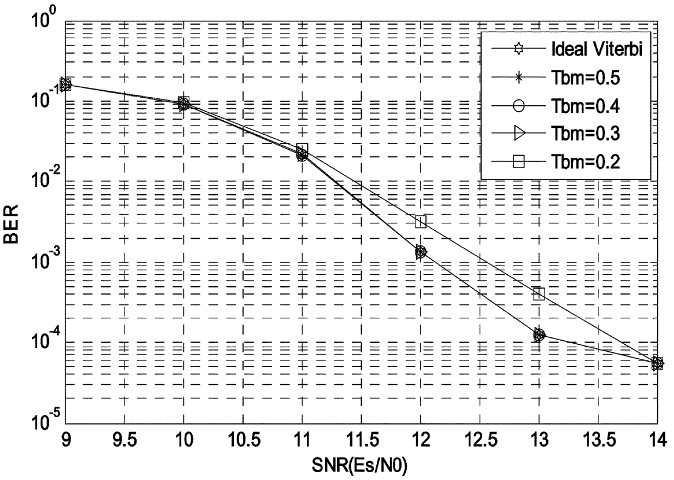


Fig. 11. BER performance of the T-algorithm on BMs.

Although the proposed T-algorithm on BMs cannot save as many computations as the conventional T-algorithm on PMs can in the high-SNR region, it still has the advantages of lower complexity, higher throughput, and better BER performance.

B. Hybrid T-Algorithm

The proposed T-algorithm on BMs not only provides an alternative to the conventional T-algorithm on PMs but also gives insights leading to techniques that can further reduce computations.

From Figs. 8 and 9, we observe that implementing the T-algorithm on BMs does not require any change to the architecture of the conventional T-algorithm on PMs. This feature makes it possible to combine the two schemes together and purge more computations in ACSU. The functional diagram of the resulting

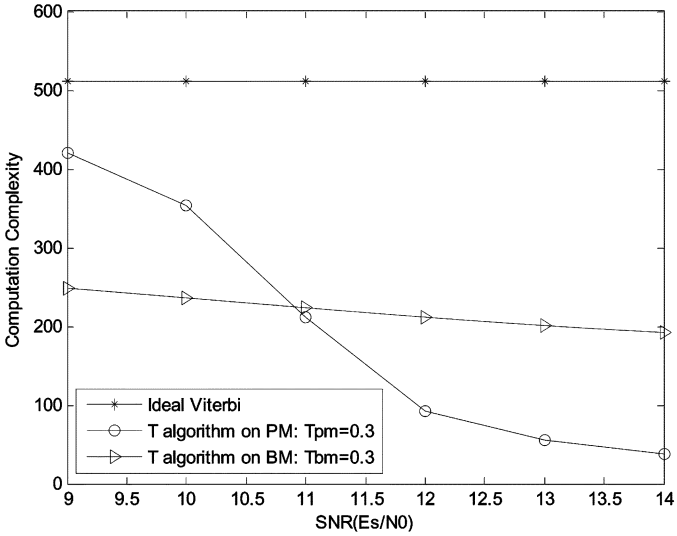


Fig. 12. Computational complexities of the T-algorithm on PMs and BMs.

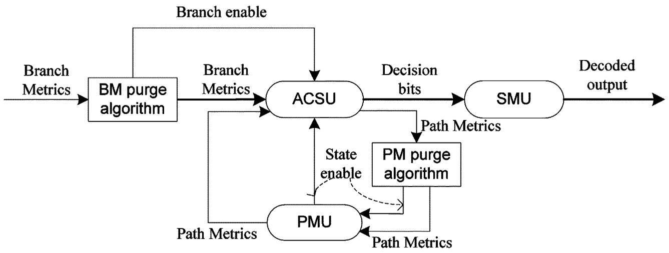


Fig. 13. Functional diagram of a VD with the hybrid T-algorithm.

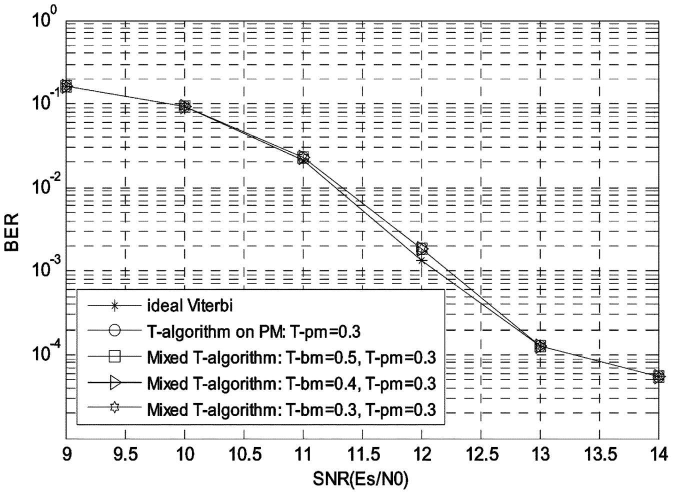


Fig. 14. BER performance of the hybrid T-algorithm.

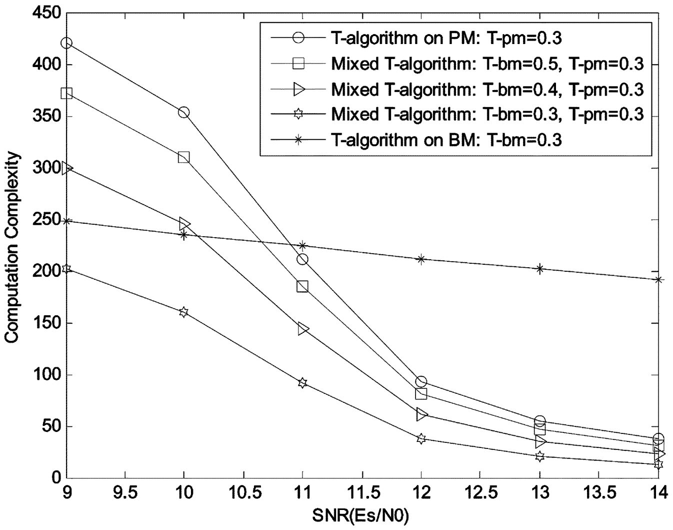


Fig. 15. Computational complexity of the hybrid T-algorithm.

scheme is shown in Fig. 13, which is called the "hybrid T-algorithm" in this paper. The details of the hybrid T-algorithm are described as follows. The PM at time slot n is calculated from the sum of PMs at time slot $n-1$ and BMs at time slot n as

$$\begin{aligned} PM_n^j = \max\Big(&PM_{n-1}^{j1} + BM_n^{j1}; PM_{n-1}^{j2} + BM_n^{j2};\\ &PM_{n-1}^{j3} + BM_n^{j3}; PM_{n-1}^{j4} + BM_n^{j4};\\ &PM_{n-1}^{j5} + BM_n^{j5}; PM_{n-1}^{j6} + BM_n^{j6};\\ &PM_{n-1}^{j7} + BM_n^{j7}; PM_{n-1}^{j8} + BM_n^{j8}\Big) \end{aligned}$$

where the subscript denotes the time slot and the superscript j denotes the label of any arbitrary state. Each PM is selected from eight candidates. The conventional T-algorithm on PMs checks the purging flags for states. For example, if PM^{j1} is purged at time slot $n-1$, then ACSU will not perform $PM_{n-1}^{j1} + BM_n^{j1}$ at time slot n. The T-algorithm on BMs applies the same rule on BMs. The hybrid T-algorithm checks the purging flags for both states and BMs. The addition operation is performed only if both the PM and BM involved in the operation are retained.

Let us again focus on the example of $\mathrm{Rm} = 11/12$. The BER performances of the hybrid T-algorithm with different parameters are shown in Fig. 14. Since the hybrid T-algorithm is developed based on the conventional T-algorithm on PMs, we keep the threshold of 0.3 for PMs. The threshold for purging BMs is selected to be equal to or greater than 0.3, according to the simulation result in Fig. 11. It is observed from Fig. 14 that when T-bm (the threshold of BMs) is equal to or greater than 0.4, the BER performance of the hybrid T-algorithm on VD is identical to that of the conventional T-algorithm on PM; thus, the three curves nearly overlap in Fig. 14. When T-bm is equal to 0.3, a slight difference of less than 0.02 dB is observed in the region with $\mathrm{SNR} < 11$ dB.

Fig. 15 shows the computational complexity of the proposed hybrid T-algorithm. For comparison, the complexity of the T-algorithm on PMs and that of the T-algorithm on BMs are also shown. Compared with the conventional T-algorithm on PMs, the computational complexity is reduced by more than 50% when the threshold is set to 0.3.

These simulation results indicate that, with the hybrid T-algorithm, more calculations can be eliminated, while its BER performance is almost the same as that of the conventional T-algorithm on PMs. However, a simple straightforward combination of the conventional T-algorithm on PMs with the proposed T-algorithm on BMs will lose the advantage of speed of the latter. To retain the advantage of speed, we need a novel architecture. Here, we take advantage of the SPEC-T algorithm proposed in [19] to reduce the length of critical path. The key

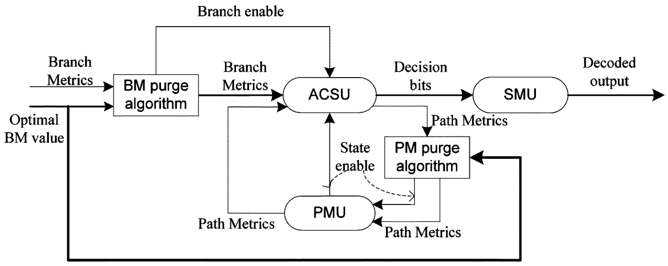


Fig. 16. Hybrid T-algorithm VD with a SPEC-T algorithm.

idea of the SPEC-T algorithm is to use an estimated optimal PM value derived from the optimal BM value, instead of searching for the optimal PM in each cycle. Since the T-algorithm on BMs will generate an optimal BM anyway, very little additional logic is needed. The functional diagram of the resultant scheme is shown in Fig. 16. The optimal BM value is forwarded to the PM purging unit to calculate the estimated optimal PM value. For most of the time, the VD uses the estimated optimal PM value to perform the T-algorithm on PM. In the meantime, some compensation schemes are needed to adjust the estimated optimal value regularly. A straightforward compensation scheme is to find the real optimal PM in a number of cycles and compute the estimation error for the next adjustment. The clock speed will not be affected by this searching process since it can be pipelined into several clock cycles outside the ACSU loop.

It is necessary to mention that when the number of BMs is not very large (e.g., for a rate-1/2 code), purging BMs may cause considerable performance loss. Thus, the proposed T-algorithm on BMs and the hybrid T-algorithm are only suitable for dealing with high-rate codes or codes that generate many BMs (e.g., codes of rate 1/3 or lower).

V. FPGA Implementation

The design is first modeled in Matlab for performance simulation and finite-word-length analysis. Simulation results show that quantizing the input signal using 7 b is a good choice because, compared with the unquantized system, a 6-b quantization will cause a performance loss of 0.8 dB, while a 7-b quantization causes only a 0.2-dB loss. Then, the 4-D 8PSK TCM decoder with 7-b-quantized input is implemented with Verilog HDL, simulated in ModelSim, and synthesized using Xilinx Virtex-4 FPGA (XC4VLX160). TMU and ACSU are synthesized individually first to verify the hardware and clock speed improvements. Then, the entire design is synthesized.

A. TMU

The TMUs with both the conventional auxiliary-trellis structure and the proposed two-step comparison architecture are implemented. Both TMUs are designed to have the capability to deal with four different modulation rates. In addition, three pipeline stages are used for both cases to increase the clock speed. Synthesis results are shown in Table VII, where the utilization of the device on FPGA is shown in percentage.

We have concluded in Section III that the proposed architecture could reduce the computational complexity by 67%. Due to the similar number of registers used for pipelining the circuits for both cases, Table VII only shows half of the hardware reduction.

TABLE VII
Synthesis Results for TMU

Xc4vlx160-12ff1148 Virtex 4, speed scale 12		
	Number of 4-input LUT	Number of Slice Flip-flop
Auxiliary trellis [10]	14686(10%)	3939(2%)
2-step comparison	7912(5%)	1280(0%)

TABLE VIII
Synthesis Results for ACSU

Xc4vlx160-12ff1148 Virtex 4, speed scale 12			
	Number of 4-input LUT	Number of Slice Flip-flop	Max clock speed (MHz)
Full trellis	18522(13%)	576(0%)	103.8
T-algorithm on PMs	23326(17%)	576(0%)	42.5
T-algorithm on BMs	18626(13%)	576(0%)	102.06
SPEC-*T* algorithm	21716(16%)	1842(1%)	90.8
Hybrid *T*-algorithm	22342(16%)	1842(1%)	88.2

B. ACSU With and Without the T-Algorihtm

In the VD design, we have proposed new schemes for ACSU in this paper. Note that the proposed new schemes can be combined with any type of SMU. Thus, when they are verified by FPGA, only the ACSUs (including the registers for PMs) are compared. Synthesis results are shown in Table VIII.

In Table VIII, "Full trellis" refers to the conventional Viterbi algorithm without any state-reducing scheme, "T-algorithm on PMs" is the conventional T-algorithm in [13], "T-algorithm on BMs" is the first proposed scheme, "SPEC-T algorithm" is the scheme proposed in [19] (except that we extend all the paths from each unpurged state; while in [19], only one path extends from each unpurged state, which will cause performance loss), and "Hybrid T-algorithm" employs the SPEC-T algorithm for purging PMs. Compared with the "Full trellis" case, the "T-algorithm on BMs" case requires a slightly more hardware overhead, while the "T-algorithm on PMs" case needs more resources. Also, applying the T-algorithm on BMs does not affect much the clock speed.

C. Four-Dimensional 8PSK TCM Decoder

The 4-D 8PSK TCM decoder employs the proposed two-step comparison architecture for the TMU and the hybrid T-algorithm for the VD. The decoder is reconfigurable and can decode signals with four different modulation rates. Table IX summarizes the synthesis results.

Since the device utilization for the entire 4-D 8PSK TCM decoder is very high, the delay due to wire routing increases. However, it is still much faster than the conventional T-algorithm on PMs. The only work similar to our design that we could find from the literature is [10]. In [10], the decoder to decode

TABLE IX
SYNTHESIS RESULTS FOR THE TCM DECODER

Xc4vlx160-12ff1148 Virtex 4, speed scale 12			
Device Type	4-input LUT	Slice Flip-flop	Max clock speed (MHz)
Number of Devices	110804	14743	78.9
Utilization	80%	10%	--

8/9 modulation rate is implemented on FPGA with an auxiliary-trellis structure for the TMU and a full trellis VD. Note that, in the implementation in [10], the highest data rate reported is 460 Mb/s. In our design, the data rate for the modulation rate of 8/9 is up to $78.9 \times 8 = 631.2\ \mathrm{Mb/s}$, which is much faster than the work reported in [10].

VI. CONCLUSION

We have proposed an efficient architecture for 4-D 8PSK TCM decoders. The efforts were focused on two most complex parts in the decoder: TMU and VD. For the TMU, the proposed architecture reduced the computational complexity to about one-third of an existing low-complexity architecture presented in [10], while the BER performance is not compromised. For the VD, we first proposed a new structure for the T-algorithm applied on BMs. Compared with the conventional T-algorithm applied on PMs, the new scheme has the advantages of lower complexity and higher throughput. Based on the proposed T-algorithm on BMs and the conventional T-algorithm on PMs, we then designed a hybrid T-algorithm scheme. Simulation results show that the hybrid T-algorithm can further reduce more than half of the computations compared with the conventional T-algorithm while still maintaining the same BER performance. Finally, we proposed a new architecture for the hybrid T-algorithm VD based on the SPEC-T algorithm that can retain the speed advantage of the conventional scheme. Implementation results in FPGA validated the excellent performance of the proposed algorithms.

REFERENCES

[1] E. Bigilri, D. Divsalar, P. J. McLane, and M. K. Simon, *Introduction to Trellis-Coded Modulation With Applications*. New York: Macmillan, 1991.

[2] S. G. Wilson, *Digital Modulation and Coding*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[3] M. Y. Rhee, *Error-Correcting Coding Theory*. New York: McGraw-Hill, 1989.

[4] S. S. Pietrobon and R. H. Deng, "Trellis-coded multidimensional phase modulation," *IEEE Trans. Inf. Theory*, vol. 36, no. 1, pp. 63–89, Jan. 1990.

[5] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo codes," in *Proc. IEEE ICC*, 1993, pp. 1064–1070.

[6] D. MacKay and R. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 33, no. 6, pp. 457–458, Mar. 1997.

[7] H.-L. Lou, P. Tong, and J. M. Cioffi, "A programmable codec design for trellis coded modulation," in *Proc. IEEE GLOBECOM*, Nov. 1997, pp. 944–947.

[8] J.-Y. Wang and M.-C. Lin, "A trellis coded modulation scheme with a convolutional processor," in *Proc. IEEE ISIT*, Jun. 1997, p. 389.

[9] A. Dinh and X. Hu, "A hardware-efficient technique to implement a trellis code modulation decoder," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 6, pp. 745–750, Jun. 2005.

[10] D. Servant, "Design and analysis of a multidimensional trellis coded demodulator," M.S. thesis, Signals, Sens. Syst., KTH, Stockholm, Sweden, 2004.

[11] *Bandwidth-Efficient Modulations*, CCSDS 401(3.3.6) Green Book, Apr. 2003.

[12] C. F. Lin and J. B. Anderson, "M-algorithm decoding of channel convolutional codes," in *Proc. Conf. Inform. Sci. Syst.*, Princeton, NJ, Mar. 1986, pp. 362–366.

[13] S. J. Simmons, "Breadth-first trellis decoding with adaptive effort," *IEEE Trans. Commun.*, vol. 38, no. 1, pp. 3–12, Jan. 1990.

[14] F. Chan and D. Haccoun, "Adaptive Viterbi decoding of convolutional codes over memoryless channels," *IEEE Trans. Commun.*, vol. 45, no. 11, pp. 1389–1400, Nov. 1997.

[15] G. Feygin and P. G. Gulak, "Architectural tradeoffs for survivor sequence memory management in Viterbi decoders," *IEEE Trans. Commun.*, vol. 41, no. 3, pp. 425–429, Mar. 1993.

[16] Y. Gang, A. T. Erdogan, and T. Arslan, "An efficient pre-traceback architecture for the Viterbi decoder targeting wireless communication applications," *IEEE Trans. Circuits Syst. I, Reg Papers*, vol. 53, no. 9, pp. 1918–1927, Sep. 2006.

[17] R. C. S. Morling and N. Haron, "Novel low-latency low-power Viterbi decoder traceback memory architecture," in *Proc. IEEE MELECON*, Dubrovnik, Croatia, May 2004, pp. 179–183.

[18] C.-C. Lin, Y.-H. Shih, H.-C. Chang, and C.-Y. Lee, "Design of a power-reduction Viterbi decoder for WLAN applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 6, pp. 1148–1156, Jun. 2005.

[19] F. Sun and T. Zhang, "Parallel high-throughput limited search trellis decoder VLSI design," *IEEE Trans. Very large Scale Integr. (VLSI) Syst.*, vol. 13, no. 9, pp. 1013–1202, Sep. 2005.

Jinjin He (S'08) was born in Nanjing, China, in 1981. She received the B.S. and M.S. degrees in electrical engineering from Fudan University, Shanghai, China, in 2003 and 2006, respectively, and is currently working toward the Ph.D. degree at the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis.

Her research interests include high-speed/low-power VLSI design for digital signal processing and digital communications.

Zhongfeng Wang (M'00–SM'05) received the B.E. and M.S. degrees from the Department of Automation, Tsinghua University, Beijing, China, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, in 2000.

From 1990 to 1995, he was with Beijing Huahai New Technology Development Company, Beijing, as a Principal Engineer and a Technical Manager. He joined Morphics Technology Inc. (currently part of Infineon Technologies) in 2000 as a Member of the Technical Staff. In 2002, he moved to the National Semiconductor Company as a Senior Staff Design Engineer. In 2003, he became an Assistant Professor with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis. Since June 2007, he has been a Senior Principle Scientist with the Broadcom Corporation, Irvine, CA. He has published over 80 technical papers and has filed several U.S. patent applications. His current research interests include low-power/high-speed VLSI design, specifically VLSI design for digital signal processing and digital communications.

Dr. Wang was an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: REGULAR PAPERS from 2003 to 2005 and is an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS and the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS. He has also served as a Technical Program Committee Member for numerous IEEE and ACM conferences. He is currently a member of the Technical Committee of VLSI Systems and Applications and the Technical Committee of Circuits and Systems for Communications in the IEEE CAS Society. He was the recipient of the Best Student Paper Award at the 1999 IEEE Workshop on Signal Processing Systems and a corecipient of the IEEE CAS Society VLSI Transactions Best Paper Award in 2007.

Huaping Liu (S'95–M'97–SM'08) received the B.S. and M.S. degrees in electrical engineering from Nanjing University of Posts and Telecommunications, Nanjing, China, in 1987 and 1990, respectively, and the Ph.D. degree in electrical engineering from the New Jersey Institute of Technology, Newark, in 1997.

From July 1997 to August 2001, he was with Lucent Technologies, Murray Hill, NJ. He joined the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, in September 2001, where he is currently an Associate Professor. His research interests include ultrawideband systems, multiple-input–multiple-output orthogonal frequency-division multiplexing schemes, channel coding, and modulation and detection techniques for multiuser time-varying fading environments.