Unified Convolutional/Turbo Decoder Design Using Tile-Based Timing Analysis of VA/MAP Kernel

Fan-Min Li, Cheng-Hung Lin, *Student Member, IEEE*, and An-Yeu (Andy) Wu, *Member, IEEE*

***Abstract*—To satisfy the advanced forward-error-correction (FEC) standards, in which the Convolutional code and Turbo code may co-exit, a prototype design of a unified Convolutional/Turbo decoder is proposed. In this paper, we systematically analyze the timing charts of both the Viterbi algorithm and the MAP algorithm. Then, three techniques, including Distribution, Pointer, and Parallel schemes, are introduced; they can be used as flexible tools in timing-chart analysis to either reduce memory size or to increase throughput rate. Furthermore, we propose a tile-based methodology to analyze the key features of timing charts, such as computing/memory units and hardware utilization. On the basis of the timing analysis, we developed a VA/MAP timing chart that has three modes (VA mode, MAP mode, and concurrent VA/MAP mode) by complementing the idle time of both VA and MAP decoding procedures. The new combined timing analysis helps us for constructing a unified component decoder with near 100% utilization rate of the processing element (PE) in both VA/MAP decoding functions.**

According to the triple-mode VA/MAP timing chart, we construct a triple-mode FEC kernel that can perform both Convolutional/Turbo decoding functions seamlessly for different communication systems. By integrating the FEC kernel with different size of memory, we can construct four types of FEC decoders for different application scenarios, such as 1) standalone Convolutional decoder (VA mode); 2) standalone Turbo decoder (MAP mode); 3) dual-mode Convolutional/Turbo decoder (VA mode and MAP mode); and 4) triple-mode Convolutional/Turbo decoder (VA mode, MAP mode, and concurrent VA/MAP mode). Finally, a prototyping FEC kernel processor that is compliant to 3GPP standard is verified in TSMC 0.18-μm CMOS process in the type of triple-mode FEC decoder.

***Index Terms*—Convolutional code, forward-error-correction code, log-MAP, maximum a-posteriori probability, reconfigurable FEC architecture, turbo code, Viterbi algorithm.**

I. Introduction

THE Viterbi Algorithm (VA) is a maximum-likelihood decoding method for Convolutional codes in digital communication systems [1], [2]. In recent years, Berrou *et al.* [3], [4] presented a new class of Forward-Error-Correction (FEC) code, Turbo code, which has near Shannon-limit superior decoding performance. Consequently, the 3G standards, such as cdma2000 [6] and WCDMA [7], have adopted it. A Turbo decoder consists of two soft-output component decoders and operates by iterative decoding. The Log-MAP algorithm, in general, has the approximate bit-error-rate (BER) performance as the MAP algorithm, but with lower complexity. Therefore, we adopted the Log-MAP algorithm (we use the name MAP for short for the Log-MAP algorithm in this paper) in the Turbo decoder design.

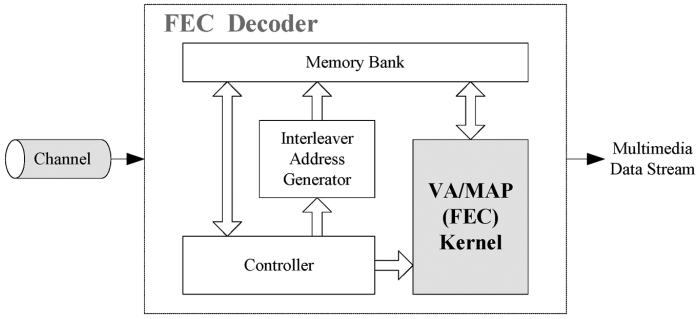


Fig. 1. Proposed unified FEC decoder.

For the 3GPP standards [5], the receiver may receive voice and data streams encoded by different FEC schemes. Our design can be targeted at infrastructure applications (mobile basestations); thus the decoder must process the data streams from several users simultaneously. Conventionally, the corresponding Convolutional decoder and Turbo decoder are built separately. For a cost-efficient design, the research on dual-mode Convolutional/Turbo decoder was published [15], [17]. However, there is still no systematic analysis (in terms of both hardware and timing efficiency) for the unified FEC decoder designs. In this paper, we propose a design methodology to integrate the two decoders, as shown in Fig. 1. The design can perform both FEC functions in terms of *timing association* (how to combine both functions in the time domain without conflict) and *hardware association* (how to perform both functions in the shared hardware with high utilization rate).

Most existing dual-mode FEC designs [15], [17] adopted timing charts to design the VA and MAP functions separately rather than in a joint fashion. In the literature, several timing-chart analysis of the VA [10] or MAP [11]–[14] is described. We first present three techniques (*Distribution*, *Pointer* and *Parallel* schemes) and propose a tile-based methodology to analyze the key features of timing charts, including computing units and hardware utilization. Then we propose the triple-mode VA/MAP timing chart by complementing the idle time of both VA and MAP decoding procedures. Finally, our proposed FEC kernel complied with 3GPP standard is verified in TSMC 0.18-μm CMOS process.

Manuscript received June 11, 2007; revised August 26, 2007. This work was supported by the National Science Council of Taiwan under Grant NSC 93-2215-E-002-011 and the Ph.D. fellowship program of MediaTeK Education Foundation. Chip fabrication was supported by Chip Implementation Center (CIC).

The authors are with the Graduate Institute of Electronics Engineering, Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. (e-mail: andywu@cc.ee.ntu.edu.tw).

Digital Object Identifier 10.1109/TVLSI.2008.2000514

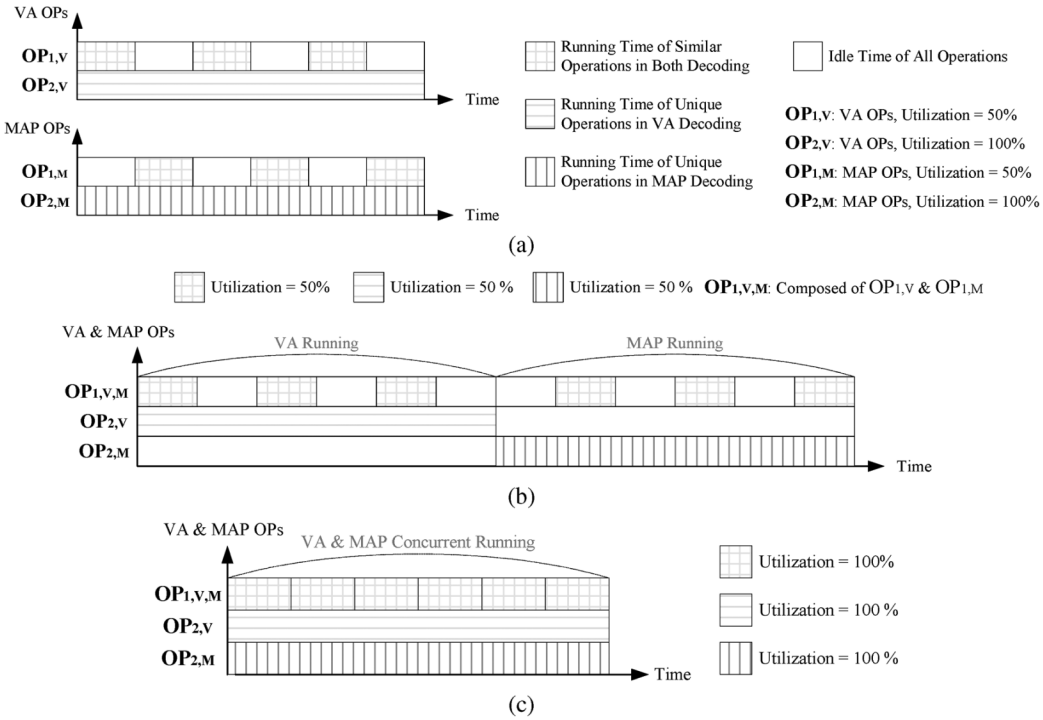


Fig. 2. Timing association. (a) Separated operations. (b) Exclusive operations. (c) Concurrent operations.

II. Timing Association and Hardware Association

A. Timing Association

Comparing the operations of VA decoding with the ones of MAP decoding, we can partition the operations of both into two parts: the same (or similar) operations ($\mathrm{OP}_{1,\mathrm{V}}$ and $\mathrm{OP}_{1,\mathrm{M}}$) and the unique operations ($\mathrm{OP}_{2,\mathrm{V}}$ and $\mathrm{OP}_{2,\mathrm{M}}$). Additionally, the operation OP_1 has the idle time because it requires the output values from the operation OP_2. Thus, the utilization rate of the operations $\mathrm{OP}_{1,\mathrm{V}}$ and $\mathrm{OP}_{1,\mathrm{M}}$ in respective timing charts in Fig. 2(a) is not 100%. Because the operation $\mathrm{OP}_{1,\mathrm{V}}$ is similar to the operation $\mathrm{OP}_{1,\mathrm{M}}$, we can reuse one operation $\mathrm{OP}_{1,\mathrm{V,M}}$. Consequently, both Fig. 2(b) and (c) can be considered as the combined decoding of VA and MAP with equal decoding time. Fig. 2(b) shows that one function is running when the other function is idle, and the utilization becomes about 50%. In recent research works, most existing dual-mode works [15], [17], adopted the combination in Fig. 2(b), although they shared one operation $\mathrm{OP}_{1,\mathrm{V,M}}$. On the other hand, in Fig. 2(c), the decoder can concurrently perform both decoding and achieve about 100% utilization rate by complementing the idle time of both operations. In this work, we propose the triple-mode VA/MAP timing chart that adopts the combination in Fig. 2(c). Thus, the decoder can increase the throughput rate, enhance the hardware utilization, and reduce the decoding latency.

B. Hardware Association

Comparing the architecture of Convolutional decoder with the one of Turbo decoder, we can partition their *Logic* (L) and *Memory* (M) into two parts: the similar architecture ($\mathrm{L}_{1,\mathrm{C}}$ is similar to $\mathrm{L}_{1,\mathrm{T}}$, and $\mathrm{M}_{1,\mathrm{C}}$ is similar to $\mathrm{M}_{1,\mathrm{T}}$) and the unique architecture ($\mathrm{L}_{2,\mathrm{C}}$ is unique to $\mathrm{L}_{2,\mathrm{T}}$, and $\mathrm{M}_{2,\mathrm{C}}$ is unique to $\mathrm{M}_{2,\mathrm{T}}$) in Fig. 3(a). Because the logic $\mathrm{L}_{1,\mathrm{C}}$ is similar to $\mathrm{L}_{1,\mathrm{T}}$, the decoder can share one logic $\mathrm{L}_{1,\mathrm{C,T}}$. By the timing association stated in Section II-A, the utilization of $\mathrm{L}_{1,\mathrm{C}}$, $\mathrm{L}_{1,\mathrm{T}}$, and $\mathrm{L}_{1,\mathrm{C,T}}$ is the same as $\mathrm{OP}_{1,\mathrm{V}}$, $\mathrm{OP}_{1,\mathrm{M}}$, and $\mathrm{OP}_{1,\mathrm{V,M}}$, respectively. Thus, the decoder can perform *exclusive* (or alternate) *decoding* or *concurrent decoding* like the timing association in Fig. 2(b), (c). Then circuit-sharing techniques are applied to merge the main functions in logic circuits ($\mathrm{L}_{1,\mathrm{C,T}}$, $\mathrm{L}_{2,\mathrm{C}}$, and $\mathrm{L}_{2,\mathrm{T}}$ in the FEC kernel). Consequently, a FEC kernel, which has three modes (*VA* mode, *MAP* mode, and *concurrent VA/MAP* mode), can be developed. If the designer composes the FEC kernel with the memory unit $(\mathrm{M}_{1,\mathrm{C}} + \mathrm{M}_{2,\mathrm{C}})$ that the Convolutional decoder needs, it becomes a standalone *Convolutional decoder*. If the designer composes it with the memory unit $(\mathrm{M}_{1,\mathrm{T}} + \mathrm{M}_{2,\mathrm{T}})$ of a Turbo decoder, it can perform as a standalone *Turbo decoder*. Then, if the designer composes it with a common memory $(\mathrm{M}_{1,\mathrm{C,T}} + \mathrm{M}_{2,\mathrm{C}} + \mathrm{M}_{2,\mathrm{T}})$ that Convolutional decoder and Turbo decoder can perform exclusively, it becomes a *Dual-mode Convolutional/Turbo decoder*. Note that the Dual-mode decoder shares one common memory $\mathrm{M}_{1,\mathrm{C,T}}$ instead of two memories $\mathrm{M}_{1,\mathrm{C}}$ and $\mathrm{M}_{1,\mathrm{T}}$, and thus it only can execute one mode at the same time, which is called *exclusive decoding*. Last, if the designer composes the FEC kernel with the memory $(\mathrm{M}_{1,\mathrm{C}} + \mathrm{M}_{2,\mathrm{C}} + \mathrm{M}_{1,\mathrm{T}} + \mathrm{M}_{2,\mathrm{T}})$ that Convolutional decoder and Turbo decoder can perform concurrently, it becomes a *Triple-mode Convolutional/Turbo decoder*. Because the Triple-mode decoder can perform three modes (especially concurrent VA/MAP mode), it can do *concurrent* (not separated) *decoding*. Therefore, the FEC kernel, which has

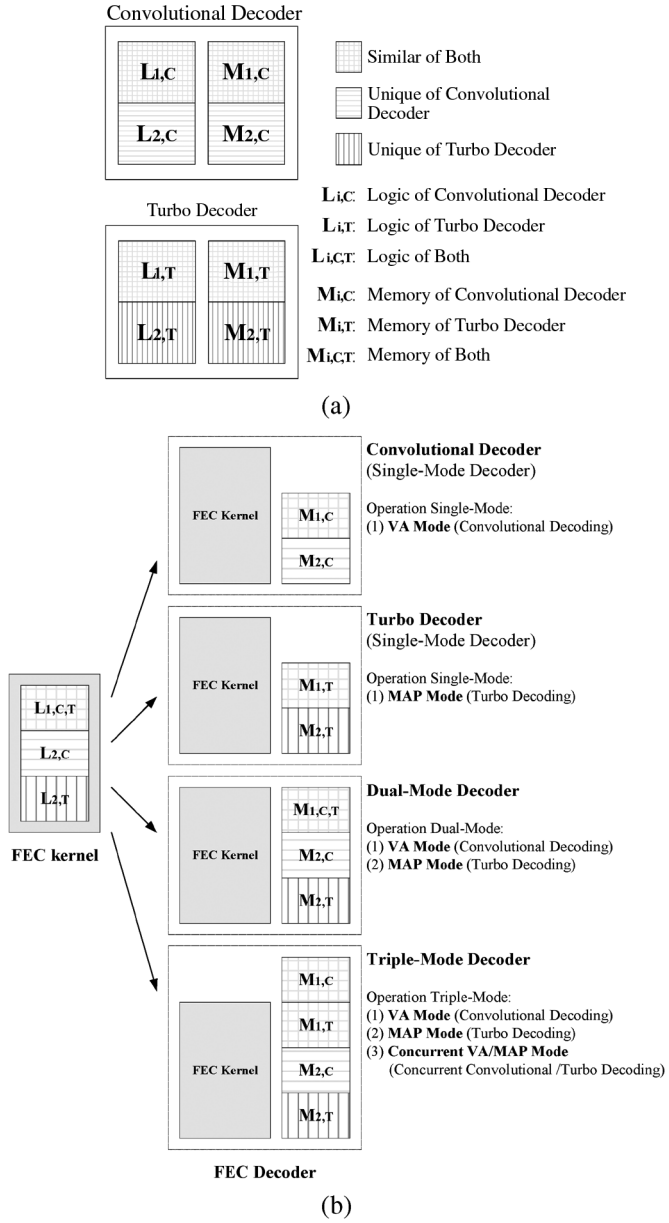


Fig. 3. Hardware association. (a) Individual. (b) Sharing.

three modes itself, can be run at four applications for different scenarios as shown in Fig. 3(b).

III. Timing Analysis of Viterbi Decoding

A. Timing Charts of Viterbi Decoding

The Viterbi algorithm proposed in 1967 [1] is a maximum-likelihood decoding method for Convolutional codes. For the global-decoding process, the decoder needs large memory to store the decision bits. The sliding-widow method is adopted in Fig. 4(a) [11]. The long received frame is partitioned into several sub blocks (sliding windows) whose length L is about five times the constraint length. For the timing chart denoted Type B in Fig. 4(a), the horizontal axis is the decoding time, and the vertical axis is the received symbol sequence. The decoder performs full parallel recursion; that is, all states complete one transition in one clock cycle. In Area I, the Path-Metric (PM)

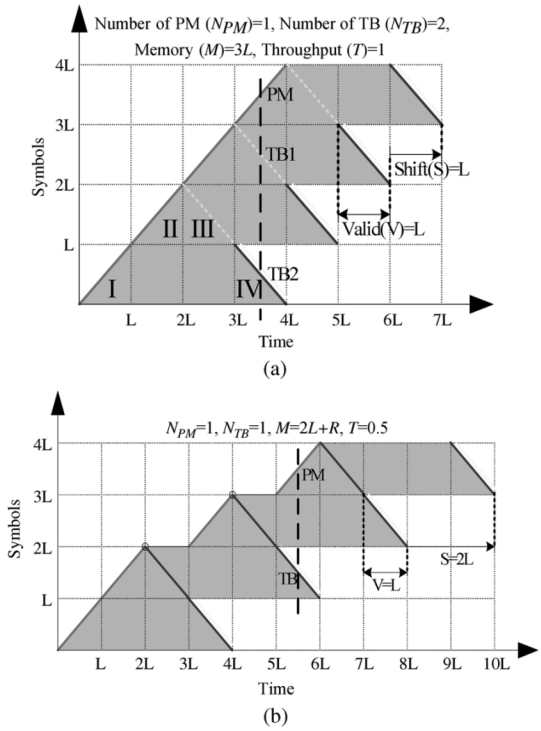


Fig. 4. VA timing chart. (a) Type B. (b) Type A.

operation begins with equal probability from the head of the first window and stores the decision bits. When the PM operation reaches the tail of the second window in Area II, the decoder can choose any state, perform the invalid Trace-Back (TB) operation in Area III, and then find the converged state in the head of the second window. Last, the valid TB operation performs from the tail of the first window and obtains the decoded information in Area IV. The sliding-window method just reuses three windows of memory, and the decoder yields almost the same performance as global decoding. For simply describing the characteristics of timing charts, we define parameters as follows.

L	The sliding window *Length*.
V	The length of the *Valid*-decoding (*Valid TB*) region.
S	The length of the *Shift* from the present decoding operation to the next decoding operation.
T	The number of decoded bits per clock cycle (called *Throughput*).
M	The *Memory* size.
N_{TB}	The *Number* of *TB* units.
N_{PM}	The *Number* of forward recursive *PM* units.
D_F	The degree of the *Forward Distribution* technique to a complete operation.
P_F	The degree of the *Forward Pointer* technique to a complete operation.

The *Throughput* (T) can be calculated by *Valid*-decoding region (V) and *Shift* length (S) in the following:

$$T = V/S. \tag{1}$$

In Fig. 4(a), the valid decoding (valid TB) region is L $(V = L)$, and the shift from the present decoding to the next decoding is L $(S = L)$. The shaded region means how much time and how

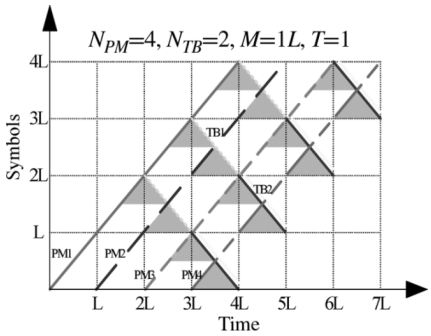


Fig. 5. VA timing chart: forward distribution technique, Type C ($D_F = 4$).

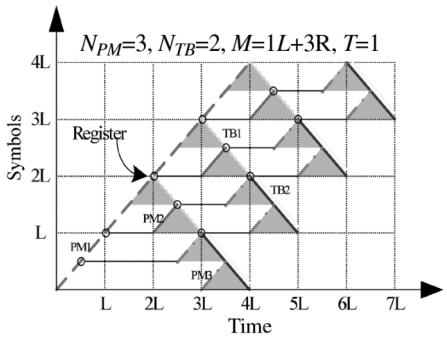


Fig. 6. VA timing chart: forward pointer technique, Type D ($P_F = 4$).

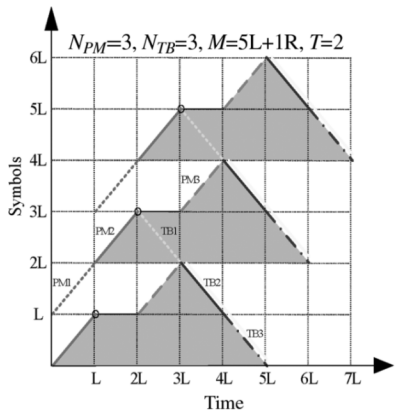


Fig. 7. VA timing chart: parallel technique, Type E ($V = 2L$ and $S = L$).

many decision bits must be kept. Observing the vertical cross line, we find that the decoder needs three windows of memory ($M = 3L$), one forward PM recursive unit ($N_{PM} = 1$), and two TB units ($N_{TB} = 2$). Then the throughput is one ($T = L/L = 1$). If the designers change the parameters V and S, they can generate other timing charts, such as Type A ($V = L$ and $S = 2L$) in Fig. 4(b). Compared with Type B, Type A needs less memory size ($M = 2L+R$) and fewer computing units ($N_{PM} = 1$ and $N_{TB} = 1$), but it loses throughput ($T = L/2L = 0.5$). Note that the stored data (M) in the memory includes two-window decision bits ($2L$) and one-stage accumulated path metrics (R). We can find that there is a trade-off between memory, computing units, and throughput.

Moreover, we present three techniques: *Distribution*, *Pointer*, and *Parallel*. 1) The distribution technique can *reduce the memory by interleaving extra operations (recursive units) and distributing the original single block (window) data*. 2) The pointer technique can *reduce the memory by interleaving extra operations (recursive units) into the original single block (window) data and use registers as pointers to provide the initial probability*. 3) Last, the parallel technique can *increase throughput rate by using more operations (recursive units) at the same time*.

1) Distribution Technique: The technique can reduce memory by interleaving extra operations and distributing the original single block data. Observing the Type B in Fig. 4(a), the designer can use four PM recursive units, and thus four small triangular areas ($D_F = 4$) of Type C ($M = L$ and $N_{PM} = 4$) in Fig. 5 replace one big triangular area of Type B ($M = 3L$ and $N_{PM} = 1$). The parameter D_F denotes the degree of the forward distribution technique to a complete operation. Consequently, the memory size is reduced to approximately $1/D_F$, but the PM recursive units increase to approximately D_F.

2) Pointer Technique: For the Type C in Fig. 5, although many PM recursive units are used, only parts of decision bits near the shaded triangular areas are useful. If the decoder can just run the PM operation near the triangular areas, the hardware or power can be reduced. To achieve it, the decoder must provide the initial probability for the PM operation. For the Type D in Fig. 6, the decoder uses one PM recursive unit to pre-compute the initial probability, stores it in the register, and provides for the other PM recursive units near triangular areas. The parameter P_F denotes the degree of the pointer technique to a complete operation. Consequently, the memory is reduced to approximately $1/P_F$, and the recursive units are approximately the same; however, the registers increase to approximately P_F.

3) Parallel Technique: For the Type B in Fig. 4(a), the valid decoding region is L ($V = L$) and the shift length is L ($S = L$). Then the throughput of Type B is one ($T = V/S = 1$). To increase the throughput, the designer can increase parameter V or reduce parameter S. Let $V = 2L$ and $S = L$; then the throughput of Type E in Fig. 7 becomes two. Note that the PM operations may need additional ones to obtain the converged initial probability. Thus, the throughput increases, but the memory size and the recursive units also increase.

In consideration of the hardware cost and throughput, the design can use distribution, pointer, and parallel techniques jointly.

B. Tile-Based Methodology for Viterbi Timing Composition

For the above discussion, we can find that the timing chart is composed of several identical operations, called *operating tiles*. For example, Fig. 8(a) illustrates a normal-operating tile, including invalid PM, valid PM, invalid TB, and valid TB operations. For systematic analysis, we define parameters as follows.

I_{pm}	The length of the *Invalid PM* operation.
V_{PM}	The length of the *Valid PM* operation.
I_{TB}	The length of the *Invalid TB* operation.
V_{TB}	The length of the *Valid TB* operation.
CV	The *Characteristic Vector* of a complete tile.

By the sliding window method, the I_{PM} and I_{TB} must be assigned to equal or longer than one L. Thus, the invalid PM operation can obtain the converged PM values for the following

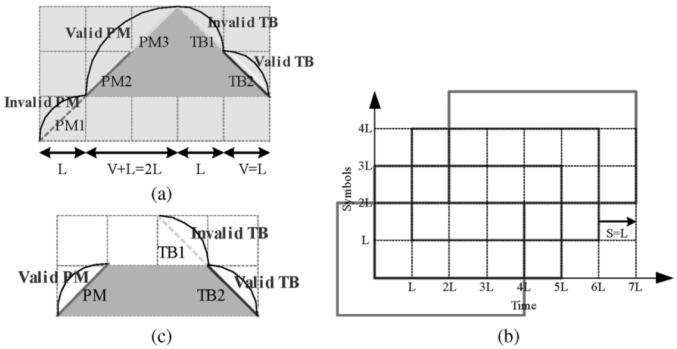


Fig. 8. Tile-based methodology for VA. (a) Normal operating tile. (b) Composed timing chart ($S = L$). (c) Modified operating tile.

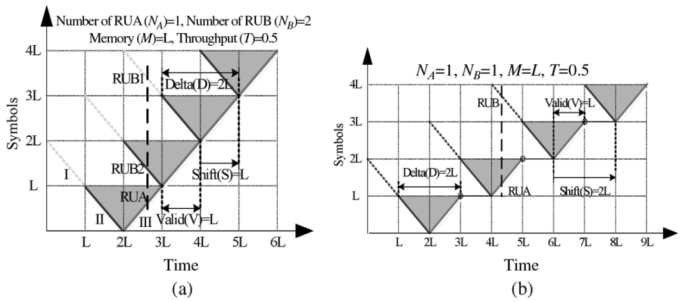


Fig. 9. MAP timing chart. (a) Type B. (b) Type A.

valid PM operation, and the invalid TB operation can obtain the converged state for the following valid TB operation. Here, we assign I_{PM} and I_{TB} to one L ($I_{PM} = L$ and $I_{TB} = L$). Note that the valid decoding region is defined as the length of the valid TB operation ($V = V_{TB}$). For the normal tile in Fig. 8(a), the V_{PM} is decided by I_{TB} and V_{TB} ($V_{PM} = I_{TB} + V_{TB}$). Then the *Characteristic Vector* (CV) specifies the key features of the tile as follows:

$$CV \equiv [I_{PM}, V_{PM}, I_{TB}, V_{TB}] = [L, L+V, L, V]. \quad (5)$$

Let the valid decoding region be L ($V = L$), and then $CV = [L, 2L, L, L]$. Next, we use the tile to compose a complete timing chart in Fig. 8(b) by assigning the shift length to L ($S = L$). Then the number of PM recursive units (N_{PM}) and TB units (N_{TB}) can be given by

$$\begin{aligned} N_{PM} &= Ceil[\text{Total Length of } PM/S] \\ N_{TB} &= Ceil[\text{Total Length of } TB/S] \end{aligned} \quad (6)$$

where the function $Ceil[X]$ rounds X to the nearest integers towards infinity. Thus, $N_{PM} = Ceil[(I_{PM} + V_{PM})/S] = Ceil[(L + 2L)/L] = 3$, $N_{TB} = Ceil[(I_{TB} + V_{TB})/S] = Ceil[(L + L)/L] = 2$, and $T = V/S = L/L = 1$. However, because the present tile may overlap the next and last tiles, a practical timing chart needs removing extra operations. Therefore, the modified operating tile in Fig. 8(c) replaces the normal one. The characteristic vector of the modified tile is $CV = [0, L, L, L]$; thus, the number of the computing units are obtained by $N_{PM} = Ceil[L/L] = 1$ and $N_{TB} = Ceil[2L/L] = 2$. Consequently, Type B in Fig. 4(a) illustrates the practical timing chart.

IV. Timing Analysis of MAP Decoding

A. Timing Charts of MAP Decoding

In 1974, Bahl *et al.* [8] proposed a new method, MAP algorithm, to decode the Convolutional codes.

For the timing chart denoted Type B in Fig. 9(a) [14], the invalid Beta operation (backward *Recursive Beta Unit*, *RUB*) begins with equal probability from the tail of the second window in Area I. When the Beta operation reaches the head of the second window, the converged beta values are obtained. Then, the valid Beta operation generates the valid values in Area II and stores them in the memory. In Area III, the Alpha operation (forward *Recursive Alpha Unit*, *RUA*) starts from the head of the first window. At the same time, combing the alpha values with the valid beta values fetched from the memory results the *Log-Likelihood Ratio* (LLR) values. The sliding-window method just reuses one window of memory. Except the common parameters of both VA and MAP algorithms defined in Section III-A, we also define others:

V	The length of the *Valid* decoding (LLR output) region.
D	The *Displacement* from the beginning of the Beta operation to the ending of the Alpha operation for the same received data.
N_A	The *Number* of *RUA*.
N_B	The *Number* of *RUB*.
D_F	The degree of the *Forward Distribution* technique to a completed operation.
D_B	The degree of the *Backward Distribution* technique to a completed operation.
P_F	The degree of the *Forward Pointer* technique to a completed operation.
P_B	The degree of the *Backward Pointer* technique to a completed operation.

In Fig. 9(a), the valid decoding region is L ($V = L$), the shift length is L ($S = L$), and the displacement is $2L$ ($D = 2L$). The shaded area means how much time and how many valid beta values must be kept. Observing the vertical cross line, we can find that the decoder needs one window of memory ($M = L$), one RUA ($N_A = 1$), and two RUBs ($N_B = 2$). The throughput is one ($T = V/S = 1$). For the same hardware as Type B ($D = 2L$), the stored values can be composed of half alpha values and half beta values ($D = L$), or they can be all alpha values ($D = 0$). For another example in Fig. 9(b), Type A ($V = L$ and $S = 2L$) just uses one RUA ($N_A = 1$) and one RUB ($N_B = 1$); however, the throughput becomes half ($T = V/S = 0.5$).

In the following, we present the three techniques stated in Section III-A for the MAP decoding. Note that the distribution and pointer techniques include forward and backward ways.

1) Distribution Technique: To reduce the memory, the designer can use the *backward distribution* technique by interleaving additional recursive units and distributing the single block data. Observing the Type B in Fig. 9(a), we use extra RUBs, and thus four small triangular areas ($D_B = 4$) of Type C ($M = L/4$, $N_A = 1$, and $N_B = 5$) in Fig. 10(a) can replace the one big triangular area of Type B ($M = L$, $N_A = 1$, and $N_B = 2$). The parameter D_B denotes the degree of the backward distribution technique to a complete decoding operation. Consequently, the memory can be reduced to approximately $1/D_B$, but the number of RUBs will increase to approximately

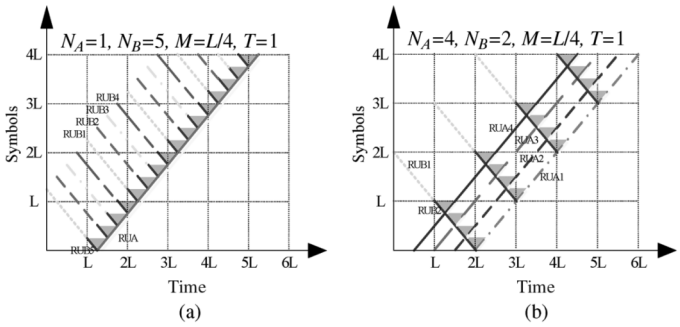


Fig. 10. MAP timing chart. (a) Backward distribution technique, Type C ($D_B = 4$). (b) Forward distribution technique, Type D ($D_F = 4$).

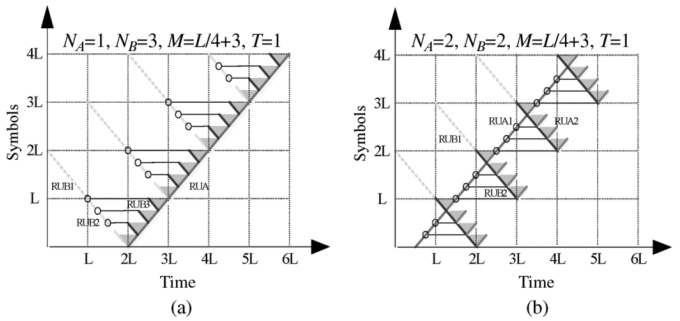


Fig. 11. MAP timing chart. (a) Backward pointer technique, Type E ($P_B = 4$). (b) Forward pointer technique, Type F ($P_F = 4$).

D_B. Moreover, we also can use the *forward distribution* technique to reduce memory, such as the Type D ($M = L/4$, $N_A = 4$, $N_B = 2$) in Fig. 10(b). Compared with the backward distribution technique, the forward way performs by additional RUAs. The parameter D_F indicates the degree of the forward distribution technique to a complete decoding operation. Then the memory can be reduced to approximately $1/D_F$, but the number of RUAs will increase to approximately D_F. Last, combining backward and forward distribution techniques can reduce the memory to about $1/(D_B * D_F)$. The total number of RUBs and RUAs will increase to approximately $(D_B * D_F)$.

2) Pointer Technique: In Fig. 10, only part values near the shaded triangular areas are useful. Thus, the pointer technique stated in Section III-A can be used, and it also includes forward and backward ways. Type E ($P_B = 4$) in Fig. 11(a) [13] illustrates the *backward pointer* technique. The parameter P_B indicates the degree of the backward pointer technique to a complete decoding operation. The memory can be reduced to approximately $1/P_B$, and the hardware can be approximately the same; however, the registers will increase to approximately P_B. Furthermore, Type F ($P_F = 4$) in Fig. 11(b) illustrates the *forward pointer* technique. The parameter P_F indicates the degree of the forward pointer technique to a complete decoding operation. The memory can be reduced to approximately $1/P_F$, and the hardware can be approximately the same; however, the registers will increase to approximately P_F. Last, combining backward and forward pointer techniques can reduce the memory to about $1/(P_B * P_F)$. The total number of RUBs and RUAs can be approximately the same, but the registers will increase to approximately $(P_B * P_F)$.

3) Parallel Technique: To improve the throughput, the designer also can use the parallel technique stated in Section III-A

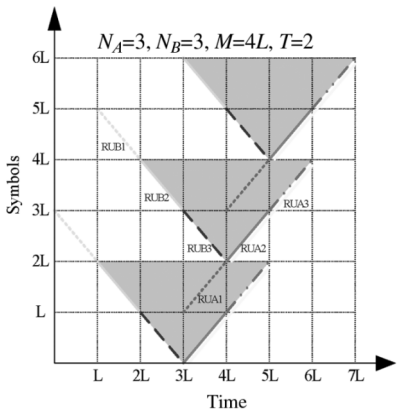


Fig. 12. MAP timing chart. Parallel technique, Type G ($V = 2L$ and $S = L$).

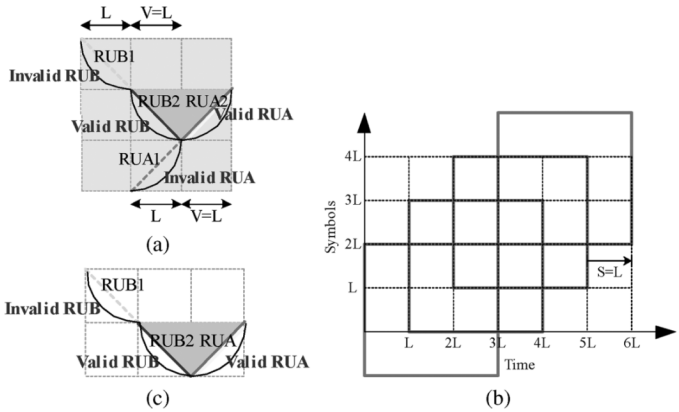


Fig. 13. Tile-based MAP timing chart. (a) Normal-operating tile. (b) Composed timing chart ($S = L$). (c) Modify-operating tile.

for the MAP decoding. For example, let the valid decoding region be $2L$ ($V = 2L$), and then the throughput of Type G ($V = 2L$ and $S = L$) in Fig. 12 [15] becomes two ($T = V/S = 2$).

Last, we only list the MAP part of all timing charts in Table I. We can observe the relationship between the parameters, techniques, cost, and performance.

B. Tile-Based Methodology for MAP Timing Composition

The timing chart of the MAP decoding also can be composed of several operating tiles. Fig. 13(a) illustrates a normal operating tile, including invalid RUA, valid RUA, invalid RUB, and valid RUB operations. For the systematic analysis, we define some parameters as follows.

- I_A The length of *Invalid* forward-recursive-*Alpha* operation.
- V_A The length of *Valid* forward-recursive-*Alpha* operation.
- I_B The length of *Invalid* backward-recursive-*Beta* operation.
- V_B The length of *Valid* backward-recursive-*Beta* operation.
- CV The *Characteristic Vector* of a completed normal operation.

By the sliding-window method, the I_A and I_B must be assigned to equal or longer than one L [12]. Thus, the invalid RUA operation can obtain the converged alpha values for the following valid RUA operation, and the invalid RUB operation can obtain the converged beta values for the following valid

TABLE I
SUMMARY OF MAP TIMING-CHART FAMILY

Type	Techniques					Parameters			Cost & Performance				Similar work	Note
	Distribution		Pointer		Parallel	V	S	D	N_A	N_B	M	T		
	Forward	Backward	Forward	Backward										
A	-	-	-	-	-	L	$2L$	$2L$	1	1	L	0.5	-	Fig. 9(b)
B	-	-	-	-	-	L	L	$2L$	1	2	L	1	[14]	Fig. 9(a)
C	-	$\surd$ (D_B=4)	-	-	-	L	L	$L/2$	1	5	$L/4$	1	-	Fig. 10(a)
D	$\surd$ (D_F=4)	-	-	-	-	L	L	$L/2$	4	2	$L/4$	1	[12]	Fig. 10(b)
E	-	-	-	$\surd$ (P_B=4)	-	L	L	$L/2$	1	3	$L/4$+3R	1	[13]	Fig. 11(a)
F	-	-	$\surd$ (P_F=4)	-	-	L	L	$L/2$	2	2	$L/4$+3R	1	-	Fig. 11(b)
G	-	-	-	-	$\surd$	$2L$	L	$4L$	3	3	$4L$	2	[15]	Fig. 12

RUB operation. Here, we assign I_A and I_B to one L ($I_A = L$ and $I_B = L$). Note that the valid decoding region is defined as the length of the valid RUA operation ($V \equiv V_A$). For the normal tile in Fig. 8(a), the V_B is based on V_A ($V_B = V_A$). Thus, the *Characteristic Vector* (CV) specifies the key features of the tile in the following:

$$CV = [I_A, V_A, I_B, V_B] = [L, V, L, V]. \tag{4}$$

Let the valid decoding region be L ($V = L$), and then $CV = [L, L, L, L]$. Next, we use the tile to compose a complete timing chart by assigning the shift to L ($S = L$) in Fig. 13(b). The number of RUA (N_A) and RUB (N_B) can be given by

$$\begin{aligned} N_A &= Ceil[\text{Total Length of } RUA/S] \\ N_B &= Ceil[\text{Total Length of } RUB/S]. \end{aligned} \tag{5}$$

Thus, $N_A = Ceil[(I_A + V_A)/S] = Ceil[(L + L)/L] = 2$, $N_B = Ceil[(I_B + V_B)/S] = Ceil[(L + L)/L] = 2$, and $T = V/S = L/L = 1$. Because the present tile may overlap the next and last tiles, the modified operating tile in Fig. 13(c) replaces the normal operating tile. The characteristic vector of the modified tile is $CV = [0, L, L, L]$; thus, the number of the computing units are obtained by $N_A = Ceil[L/L] = 1$ and $N_B = Ceil[(L+L)/L] = 2$. Consequently, Type B in Fig. 9(a) illustrates the practical timing chart.

V. TIMING ANALYSIS OF TRIPLE-MODE VA/MAP DECODING

A. Utilization Analysis

The hardware utilization is an important factor in evaluating a design. To minimize the idle time of computing units can improve the hardware utilization. Now we use the basic operating tiles to compute the utilization as follows:

$$\begin{aligned} \text{Utilization of } Y &\equiv \frac{\text{Running Time}}{\text{Running Time} + \text{Idle Time}} \\ &= \frac{\text{Total Length of } Y/S}{Ceil[\text{Total Length of } Y/S]} \end{aligned} \tag{6}$$

where the Y can be replaced by RUA, RUB, PM, and TB. For the example in Fig. 13(c), $S = L$ and $CV = [I_A, V_A, I_B, V_B] = [0, L, L, L]$. Then the *utilization of RUA* $= [(0 + L)/L]/Ceil[(0 + L)/L] = 100\%$, and the *utilization of RUB* is also 100%.

B. Tile-Based Analysis on the Utilization of Complementary VA/MAP

Now we try to find a suitable timing chart composed of two different functions. For the Type A of the VA decoding in Fig. 4(b), $S = 2L$ and $CV = [I_{PM}, V_{PM}, I_{TB}, V_{TB}] = [0, L, L, L]$. Then the *utilization of* $PM = [(0 + L)/2L]/Ceil[(0 + L)/2L] = 50\%$, and the *utilization of* $TB = [(L + L)/L]/Ceil[(L + L)/L] = 100\%$. Moreover, for the Type A (with assigning D to zero) of the MAP decoding in Fig. 9(b), $S = 2L$ and $CV = [I_A, V_A, I_B, V_B] = [0, L, L, L]$. Then the *utilization of RUA* $= [(0 + L)/2L]/Ceil[(0 + L)/2L] = 50\%$, and the *utilization of RUB* $= [(L + L)/L]/Ceil[(L + L)/L] = 100\%$. Observing the tiles of the VA and MAP decoding, the utilization of both operations is not all 100%. Because of using the same forward recursion, the PM operation of the VA decoding is similar to the Alpha operation of the MAP decoding. Thus, we have the idea of combing both PM and RUA operations by sharing one *Forward Recursive Unit* (RUF) to achieve near 100% utilization of the VA/MAP decoding. Fig. 14(a) illustrates the complementary tile of the VA/MAP decoding. Using the basic complementary tile, we compose a complementary timing chart in Fig. 14(b). Thus, the VA/MAP decoder can perform in three modes: 1) *VA* mode; 2) *MAP* mode; or even 3) *concurrent VA/MAP* mode.

C. Complementary Timing Charts of VA/MAP

For the complementary timing chart in Fig. 14(b), the utilization of RUF becomes just 50% and the one of RUB becomes 0% when operating in VA mode. If the decoder always performs in fixed VA mode (it does not perform in MAP mode), we can use the extra utilization to reduce the memory. In Fig. 14(b), the VA decoding can adopt the distribution technique ($D_F = 2$) by using the extra 50% utilization of RUF as shown in Fig. 15(a). Again, we can translate the RUB into an additional RUF by exchanging the input and output ports of trellis wires. Thus, the VA decoding can adopt the forward pointer technique ($P_F = 2$ and $P_F = 4$) by using the RUB as shown in Fig. 15(b) and (c). Note that, for the pointer technique, the designer must tradeoff between the reduced memory of decision bits and the increased registers of path metrics, based on the window length and the word length. On the other hand, if the decoder always performs in fixed MAP mode, the utilization of RUF also becomes just 50%, and the MAP decoding can adopt the pointer technique ($P_F = 4$) by using the extra 50% as shown in Fig. 15(d).

D. Realization of Triple-Mode Timing Chart

In Fig. 14, the timing chart is based on the same constraint length of the VA decoding and the MAP decoding. However, in advanced communication systems, the constraint length of both decoding is usually different. In cdma2000 [6] or WCDMA [7],

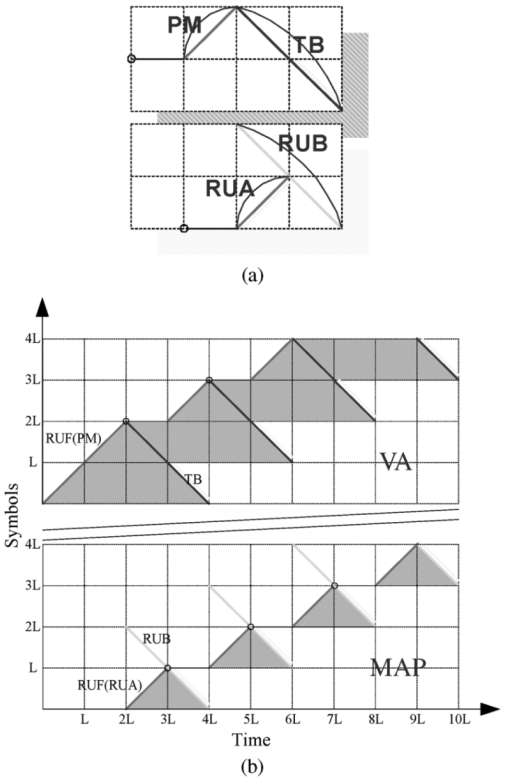


Fig. 14. (a) Complementary tile of the VA/MAP decoding. (b) Composed triple-mode VA/MAP timing chart.

the constraint length of Convolutional codes is 9 (256 states per trellis stage), and the one of Turbo codes is 4 (8 states per trellis stage).

By evaluating the cost and utilization, we use eight *Add-Compare-Select* (ACS) units to compose a RU. Then, the RU performs full parallel recursion for MAP decoding and partial parallel recursion for VA decoding. Thus, the MAP decoder can perform one stage transition of 8 states each clock cycle; however, the VA decoder must spend 32 clock cycles performing one stage transition of 256 states. Fig. 16 illustrates the practical realization of the triple-mode timing chart. Let the sliding window length of the MAP decoding be 32 $(L = 32)$. In Fig. 16, the block length in the horizontal axis is L (32 clock cycles). Then in 32 clock cycles, the MAP decoding completes one sliding window (32 stages $*$ 8 states per stage), and the VA decoding completes one trellis stage (one stage $*$ 256 states per stage). Therefore, the step size in the MAP-part vertical axis is L (32 stages), and the step size in the VA-part vertical axis is one (one stage). Moreover, let the sliding window length of VA be 45 $(L' = 45)$. Then the VA decoding completes one sliding window in 1440 clock cycles (L' stages $*$ 32 clock cycles per stage).

Finally, the decoder in concurrent VA/MAP mode performs both parts of the timing chart in Fig. 16, and the decoder in MAP mode performs the MAP part. In particular, when the decoder performs in VA mode, the RUF operation can run continuously as shown in Fig. 17(a). Fig. 17(b) illustrates the magnification of Fig. 17(a). The TB operation performs the full parallel trace backing, and thus it spends 90 clock cycles $(2L' = 90)$ for invalid and valid trace backing. Last, the throughput of the MAP decoding in MAP mode is the same as the one in concurrent VA/MAP mode, but the throughput of the VA decoding in VA mode become twice the one in concurrent VA/MAP mode.

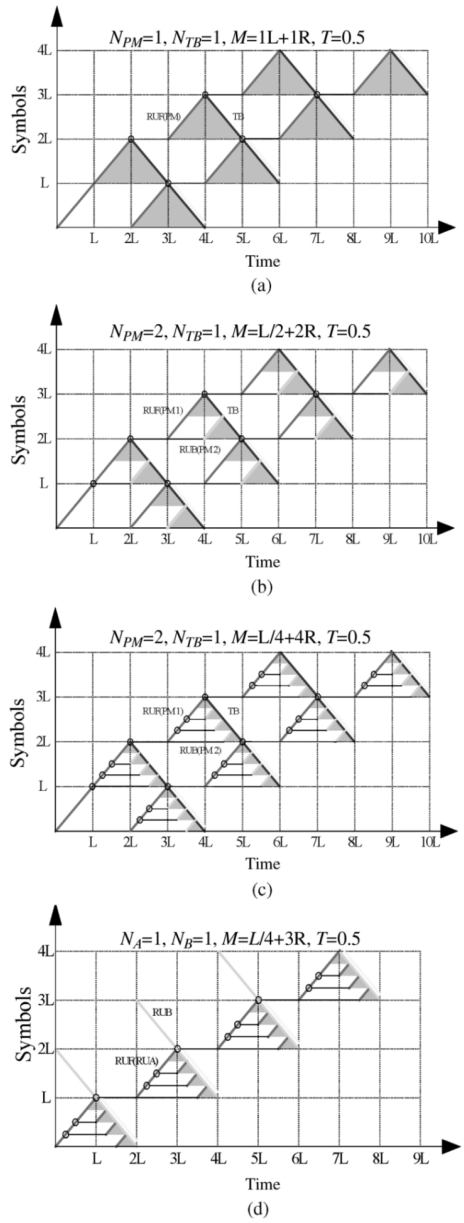


Fig. 15. Timing chart of fixed mode decoding. (a) VA mode-1 $(D_F = 2)$. (b) VA mode-2 ($D_F = 2$ and $P_F = 2$). (c) VA mode-3 ($D_F = 2$ and $P_F = 4$). (d) MAP mode-1 $(P_F = 4)$.

E. Analysis of Hardware Utilization in the Triple-Mode Timing Chart

The utilization of the triple-mode timing chart stated in Fig. 14 is listed in Table II(a). By (6), the utilization of a computing unit is the running time divided by the summation of the running time and the idle time. The analysis includes three key computing units: RUF, RUB, and TB unit. In the VA decoding, the RUF spends 50% utilization $(L/2L = 0.5)$ performing the PM operation, and the TB unit spends 100% utilization $(2L/2L = 1)$. In the MAP decoding, the RUF spends 50% utilization $(L/2L = 0.5)$ performing the Alpha operation, and the RUB spends 100% utilization $(2L/2L = 1)$. In the concurrent VA/MAP decoding, the utilization of the

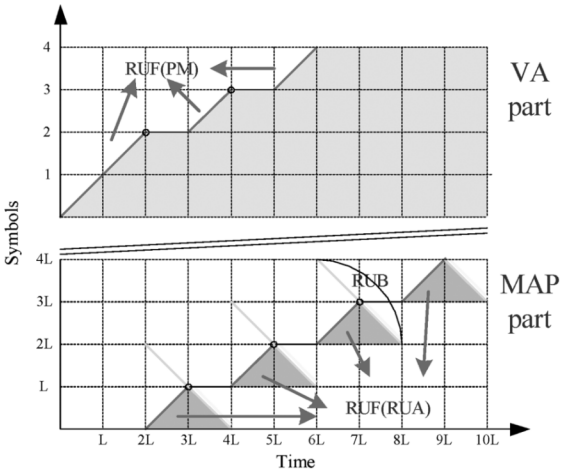


Fig. 16. Triple-mode timing chart with $K = 9$ in Convolutional codes and $K = 4$ in Turbo codes.

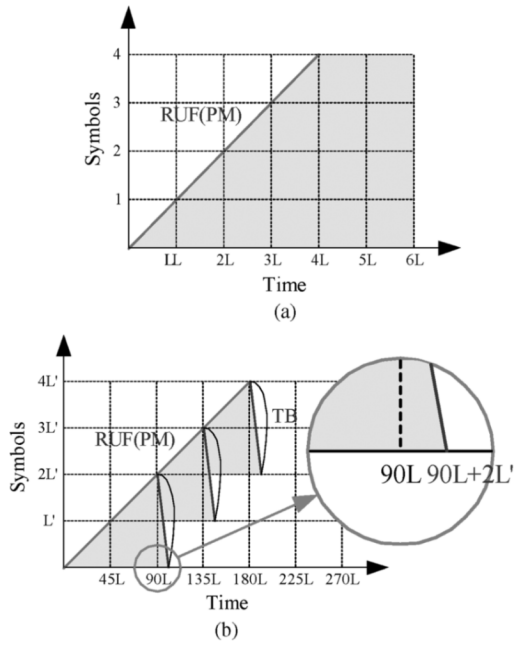


Fig. 17. Timing chart of VA mode decoding. (a) Zoom-in vertical axis. (b) Zoom-out horizontal axis.

TB unit and RUB is 100%. Moreover, RUF also spends 100% utilization performing both PM and Alpha operations.

On the other hand, the utilization of Fig. 16 is listed in Table II(b). In the VA decoding, the RUF performs (partial parallel) the PM operation and spends 100% utilization. In Fig. 17(b), because the VA decoding time is extended $45L$ times, the TB unit spends 6.25% utilization $(2L'/45L = (2*45)/(45*32) = 90/1440 = 0.0625)$. In the MAP decoding, the RUF spends 50% utilization performing the Alpha operation, and the RUB spends 100% utilization. In the concurrent VA/MAP decoding, the VA decoding is slotted into the MAP decoding. The TB unit spends 3.125% utilization $(2L'/90L = (2*45)/(90*32) = 90/2880 = 0.03125)$, and the RUF also spends 100% utilization performing both PM and Alpha operations.

For the recent dual-mode works [15], [17], they perform the VA decoding and the MAP decoding separately. When one of VA or MAP decoding is running, the other is taking

TABLE II
UTILIZATION RATE OF KEY COMPUTING COMPONENTS: (a) TRIPLE-MODE TIMING CHART FOR THE SAME CONSTRAINT LENGTH (b) TRIPLE-MODE TIMING CHART FOR THE CONVOLUTIONAL CODES $K = 9$ AND TURBO CODES $K = 4$. (c) DUAL-MODE TIMING CHART FOR THE CONVOLUTIONAL CODES $K = 9$ AND TURBO CODE $K = 4$

Mode	RUF (PM)	TB	RUF (RUA)	RUB
VA	50%	100%	0%	0%
MAP	0%	0%	50%	100%
VA/MAP (Concurrent)	50%	100%	50%	100%

(a)

Mode	RUF (PM)	TB	RUF (RUA)	RUB
VA	100%	6.25%	0%	0%
MAP	0%	0%	50%	100%
VA/MAP (Concurrent)	50%	3.125%	50%	100%

(b)

Mode	RUF (PM)	TB	RUF (RUA)	RUB
VA	100%	6.25%	0%	0%
MAP	0%	0%	50%	100%
VA/MAP (Exclusive)	50%	3.125%	25%	50%

(c)

rest. Table II(c) illustrates that the VA decoding and the MAP decoding are exclusive, such as the designs in [15]. Assume that the execution time of both decoding is the same. Then, the utilization of the computing units in either the VA decoding or the MAP decoding become just half.

VI. VLSI ARCHITECTURE OF TRIPLE-MODE FEC DECODER

To satisfy advanced multiple specifications of the communication systems, it is necessary to design a unified FEC decoder as shown in Fig. 1, where we can just change control signals or memory banks for different systems. Therefore, according to the triple-mode VA/MAP timing chart in Fig. 16, we propose a VA/MAP (FEC) kernel, which can be reconfigured for different decoding methods or parameters. Due to the different frame lengths of Convolutional codes and Turbo codes in standards, which affect the memory size, we decided to separate the standard-dependent memory banks and elaborate the computing kernel. The designer just needs to combine the FEC kernel with extra logic or memory to construct the complete FEC decoder for various applications.

A. Architecture of Triple-Mode FEC Decoder

Fig. 18(a) illustrates the proposed architecture of the FEC decoder, including the FEC kernel, extra standard-dependent memory, and extra logic. Fig. 18(b) illustrates the data path of the FEC kernel, including computing modules, wires, and memory blocks. The grid areas are the storage. According to our proposed triple-mode VA/MAP timing chart, the FEC kernel includes one *RUF (PM/RUA)* module, one *RUB* module, and one *TB* module. Moreover, the distance between the received symbol and each branch symbol is computed immediately for the RU, so the transition probability- computing units (*BM/Gamma0* module and *Gamma1* module) are built. Based on the trellis decoding, the design needs the *EETR0* module and *EETR1* module to reconfigure trellis routers for different generator polynomials. The *LLR* module receives transition

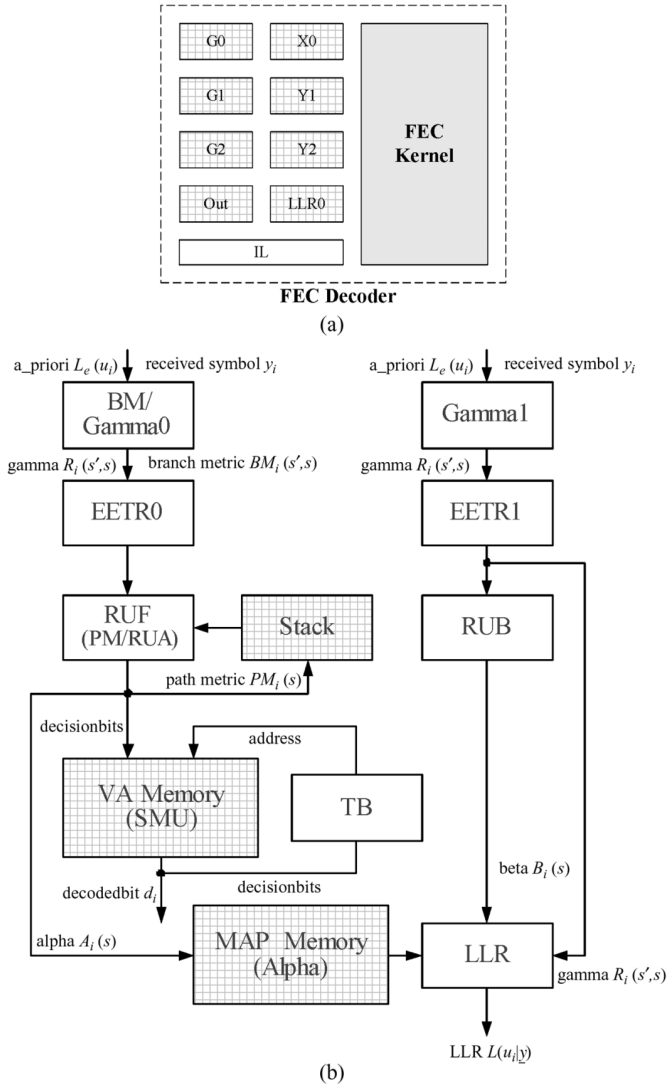


Fig. 18. Block diagram. (a) FEC decoder. (b) FEC kernel.

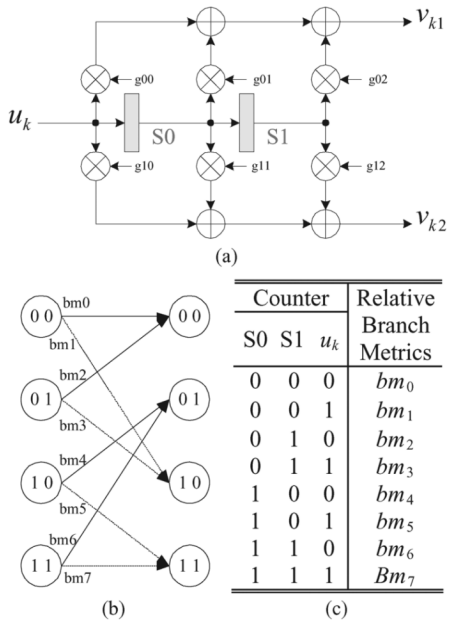


Fig. 19. $K = 3$ Nonsystematic Convolutional codes. (a) NSC encoder. (b) Trellis. (c) Truth table.

probability gamma values, forward recursive alpha values, and backward recursive beta values to obtain LLR values.

1) VA Mode (Convolutional Decoding): According to the timing chart of VA mode in Fig. 17, for a 256-state trellis, 512 branch metrics $BM_i(s', s)$ are generated from one received symbol y_i. The *BM/Gamma0* module computes the branch metrics $BM_i(s', s)$ and transmits them into the *RUF* modules to obtain path metrics $PM_i(s)$ via the *EETR0* module. The *RUF* module includes 8 ACS units and obtains path metrics $PM_i(s)$ of 256 transition states in 32 clock cycles. In the procedure of recursion, we need the *Stack* block to store $2 * 256$ path metrics composed of old and updated values. Moreover, two sliding window lengths $(2 * 45)$ of decision bits are stored in the *Survivor Memory Unit (SMU)* block. Then, the *TB* module can decode information by trace backing. Furthermore, a complete Convolutional decoder needs extra memory to store received frame ($G0$, $G1$, and $G2$ blocks) and decoded bits (*Out* block). For 3GPP standards, the maximum frame length of Convolutional codes is 504 (we can use the specified memory size, 512).

2) MAP Mode (Turbo Decoding): According to the MAP part of the triple-mode timing chart in Fig. 16, the FEC kernel uses two RU modules. The *RUF* module obtains forward recursive alpha values $A_i(s)$, and the *RUB* module obtains backward recursive beta values $B_i(s)$. The transition probability-computing units (*BM/Gamma0* module and *Gamma1* module) immediately obtain the transition probability gamma values $R_i(s', s)$ used by RUs. For an 8-state trellis, 16 gamma values $R_i(s', s)$ are generated from one received symbol y_i. The gamma values $R_i(s', s)$ need not to be stored but transmitted to the *RUF* module via the *EETR0* module. We store one window of $32 * 8$ alpha values $A_i(s)$ in the MAP memory (*Alpha* block). Moreover, because the *RUF* module must wait the *RUB* module in the timing chart, we must store 8 alpha values in the *Stack* block to be the initial values of the next Alpha operation. Then the *RUB* module performs the Beta operation and obtains beta values $B_i(s)$. At the same time, the LLR values $L_i(u_i|y)$ are generated in the *LLR* module by the beta values $B_i(s)$, gamma values $R_i(s', s)$, and alpha values $A_i(s)$ fetched from the *Alpha* block.

Last, a complete Turbo decoder needs extra memory to store received frame ($X0$, $Y1$, and $Y2$ blocks) and the output of LLR values ($LLR0$ block). For 3GPP standards, the maximum frame length of Turbo codes is 5114 (we can use the specified memory size, 5120). Moreover, the Turbo decoder also needs extra logic to perform interleaver (*IL* module). Thus, after several iterations, the decoded bits can be obtained by the hard decision of the LLR values $L_i(u_i|y)$.

3) Concurrent VA/MAP Mode (Concurrent Convolutional/Turbo Decoding): According to the timing chart in Fig. 16, the sharing blocks of both decoding include the *BM/Gamma0* module, *EETR0* module, *RUF* module and *Stack* block. Most of the function blocks in the MAP part perform the same procedures as the decoding in MAP mode. However, the function blocks in the VA part just use the half utilization to perform the VA decoding. Furthermore, the sharing blocks work alternately

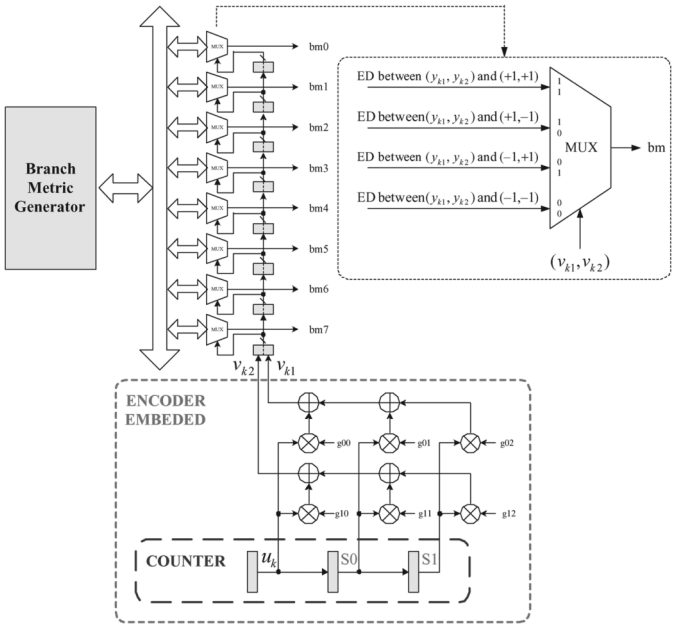


Fig. 20. EETR in VA mode.

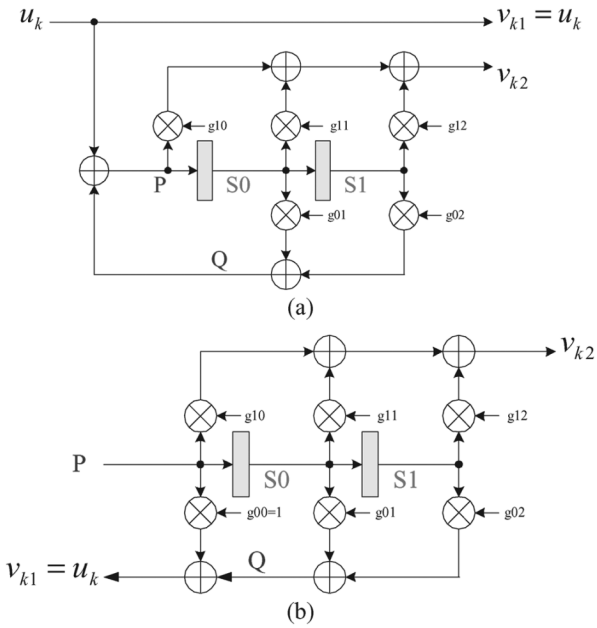


Fig. 21. $\mathrm{K} = 3$ systematic turbo encoder. (a) RSC. (b) Equivalent NSC.

in the VA decoding or the MAP decoding, but the others work continuously in the same decoding.

B. DSP Module Designs of the Triple-Mode FEC Kernel

1) Encoder Embedded Trellis Router (EETR): For different channels, the information bits may be encoded with relative generator polynomials; therefore, the decoder needs a suitable trellis network. For different generator polynomials, the connection of branch wires is the same, but the branch symbols are changed. Thus, we propose a method of using an encoder with the input of generator polynomials to generate the branch symbols in the initial time. Then the branch symbols can be used in computing the branch metrics or gamma values in the running time. We divide the EETR into two modes to illustrate in the following.

a) VA mode: For a $K = 3$ *Non Systematic Convolutional* (NSC) encoder in Fig. 19(a), the information bit u_k and present state $(S0, S1)$ generate the output symbol (v_{k1}, v_{k2}). Fig. 19(b), (c) illustrates the trellis and the truth table, in which the information $(u_k, S0, S1)$ decide the location bm_k of each branch symbol (v_{k1}, v_{k2}). Because the branch symbols just include four situations: $(+1, +1)$, $(+1, -1)$, $(-1, +1)$, and $(-1, -1)$. Thus, the Euclidean distance of four situations can be pre-computed in the *Branch-Metric-Generator* module, and then be transmitted to relative locations by a multiplexer. In Fig. 20, the encoder-embedded counter $(u_k, S0, S1)$ generates the control signals (v_{k1}, v_{k2}) according to the sequence of branch locations bm_k. In addition, the control singles illustrate that the branch symbol on the branch location bm_k is (v_{k1}, v_{k2}). Using the control signals, the multiplexer can select the relative branch metrics from the *Branch-Metric-Generator* module and then transmit to the RU module. Note that the 1-bit multiplier is an and-operation, and the 1-bit adder is an exclusive-or-operation.

b) MAP mode: For a $\mathrm{K} = 3$ Turbo encoder, it is composed of two RSC encoders, one of which is shown in Fig. 21(a). As well as the VA mode, the information bit u_k and present state $(S0, S1)$ decide the output symbol (v_{k1}, v_{k2}). Observe the P node in Fig. 21(a):

$$\begin{aligned} u_k \oplus Q = P &\Rightarrow u_k = P \oplus Q \\ &\Rightarrow u_k = (P \cdot g00) \oplus Q, \text{ where } \quad g00 = 1 \\ &\Rightarrow v_{k1} = (P \cdot g00) \oplus (S0 \cdot g01) \oplus (S1 \cdot g02), \\ &\qquad \text{where } g00 = 1 \\ &\Rightarrow v_{k1} = \text{function}(P, S0, S1), \text{ where } g00 = 1 \end{aligned} \tag{7}$$

where $\oplus$ denotes the 1-bit adder, $Q = (S0 \cdot g01) \oplus (S1 \cdot g02)$, and $v_{k1} = u_k$ since the v_{k1} is the systematic bit. Let the first coefficient of the generator polynomials be one $(g00 = 1)$, and then using the equivalent NSC architecture in Fig. 21(b) with the counter $(P, S0, S1)$ can sequentially create control signal (v_{k1}, v_{k2}) according to the sequence of branch locations bm_k. The control signal (v_{k1}, v_{k2}) also equals the value obtained by the original RSC architecture in Fig. 21(a). Using the control signals, the multiplexer can select the relative gamma value $R_i(s', s)$ from the *Gamma-Generator* module and then transmit to the RU module.

2) BM/Gamma Module: The branch metrics is similar to the gamma values except that the gamma values include *a priori* probability. Therefore, the *Branch-Metric-Generator* module and the *Gamma- Generator* module can be merged into the *BM/Gamma* module. The *BM/Gamma* module computes the *Euclidean Distance* (ED) between the received symbol y_i and each branch symbol $b(s', s)$ of the trellis. The conditioned probability value $\Pr(y_i|x_i)$ can be approximated in the following [2]:

$$\Pr(y_i|x_i) \approx e^{-ED_i}, \quad \text{where } ED_i \equiv \sum_{j=0}^{n-1} (y_{i,j} - b_j)^2 \tag{8}$$

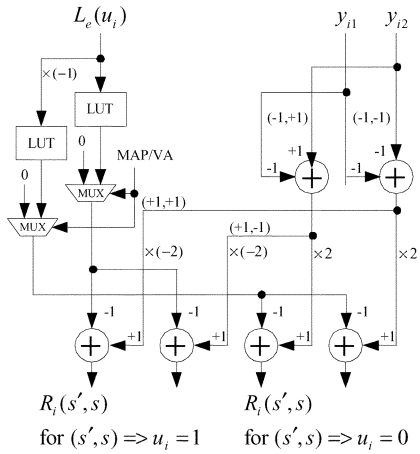


Fig. 22. BM/Gamma architecture for $n = 2$.

and $b_j \in \{1, -1\}$ is the bit of the n-length symbol $b(s', s)$. Then, the gamma values of all branches can be simplified as follows:

$$\begin{aligned}\max_{\text{all-branch}} &\{\log \Pr(u_i) - ED_i\} \\ &= \max_{\text{all-branch}} \left\{ \log \Pr(u_i) - \sum_{j=0}^{n-1} (y_{i,j} - b_j)^2 \right\} \\ &= \max_{\text{all-branch}} \left\{ \log \Pr(u_i) + 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j \right\} \end{aligned} \quad (9)$$

where the *a priori* probability $\Pr(u_i)$ can be obtained by the extrinsic information $L_e(u_i)$ of the previous iteration.

$$\Pr(u_i = 0) = \frac{1}{1 + e^{L_e(u_i)}} \text{ and } \Pr(u_i = 1) = \frac{e^{L_e(u_i)}}{1 + e^{L_e(u_i)}}. \quad (10)$$

Again, (9) can be simplified as follows:

$$\begin{aligned} R_i('s,s) &\approx \log \Pr(u_i) + 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j \\ &= \begin{cases} 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j - \log \left[1 + e^{-L_e(u_i)}\right], \\ \quad \text{as } ('s,s) \Rightarrow u_i = 1 \\ 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j - \log \left[1 + e^{L_e(u_i)}\right], \\ \quad \text{as } ('s,s) \Rightarrow u_i = 0 \end{cases} \\ &= \begin{cases} 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j - LUT\left[L_e(u_i)\right], \\ \quad \text{as } ('s,s) \Rightarrow u_i = 1 \\ 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j - LUT\left[-L_e(u_i)\right], \\ \quad \text{as } ('s,s) \Rightarrow u_i = 0. \end{cases} \end{aligned} \quad (11)$$

Because the code rate of the Turbo code in cdma2000 is 1/2, there are four situations $(+1, +1)$, $(+1, -1)$, $(-1, +1)$, and $(-1, -1)$ to compute the gamma values.

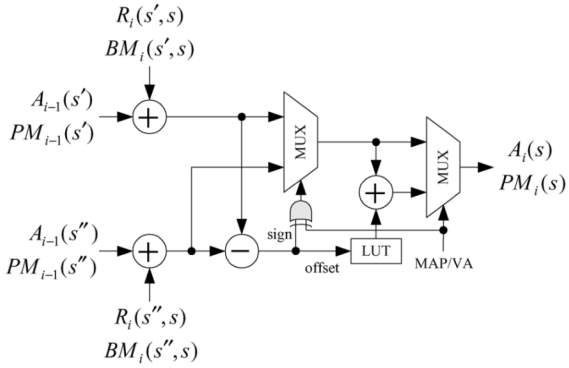


Fig. 23. ACSOS architecture.

In the VA mode, the branch metrics $BM_i(s', s)$, which are similar to gamma values except the priori probability $\Pr(u_i)$, can be calculated by

$$BM_i(s', s) = 2 \sum_{j=0}^{n-1} y_{i,j} \cdot b_j. \quad (12)$$

For Convolutional codes of the code rate 1/2, the branch metrics have four saturations $(+1, +1)$, $(+1, -1)$, $(-1, +1)$, and $(-1, -1)$. Moreover, the outputs from $(+1, +1)$ and $(+1, -1)$ paths equal the sign inversion of the outputs from $(-1, -1)$ and $(-1, +1)$ paths. Last, the architecture combining the Gamma operation and the BM operation is shown in Fig. 22.

3) RUF Module: In the VA decoding, the decoder searches the path of minimum accumulated distance. However, the decoder in the MAP decoding searches the path of maximum accumulated probability. The comparator is based on the subtraction and using the sign bit to decide the accumulated path. Thus, we can use the exclusive-or logic to switch the function of selecting the minimum or maximum path. Therefore, either in the MAP decoding or in the VA decoding, they can use the same *RUF* module. Moreover, by the Log-MAP algorithm [10], the approximation needs a correction function (LUT) to compensate (offset) in the MAP decoding. Thus, the decoder also can perform the PM operation or the Alpha operation by using a switch to change the circuit function. Consequently, the *RUF* module comprises eight *ACS-Offset-Select* (ACSOS) units that obtain alpha values $A_i(s)$ or path metrics $PM_i(s)$ as shown in Fig. 23.

C. Implementation and Performance Comparisons

For the 3GPP standards, the constraint lengths of Convolutional codes and Turbo codes are 9 and 4. For an efficient VLSI design, the decoder reuses the smaller RU composed of eight ACSOS units. After fixed-point analysis, we adopted word length 6 bits for the received symbol and 8 bits for the path metrics, alpha, and beta values with unbiased recursion. The sliding window length of the Viterbi decoding L' is 45 and the one of the Turbo decoding L is 32. Finally, our proposed FEC kernel for 3GPP standards is verified in TSMC 0.18 μm CMOS process, and Fig. 24 shows the layout view and chip summary. The post-layout simulations show that the throughput rate can achieve: 1) 3.12 Mb/s for Convolutional decoding in VA mode; 2) 4.17 Mb/s@6 iterations for Turbo decoding in MAP mode; and 3) 1.56 Mb/s for Convolutional decoding and 4.17 Mb/s@6

TABLE III
COMPARISON OF TRIPLE-MODE FEC DECODER: (a) COMPARISONS. (b) APPLICATION. (c) AREA PERCENTAGE

		University of Kaiserslautern [18]	Bell Labs Research, Lucent Tech. [16]	Our Proposed **Triple-Mode Decoder**
Technique Process		0.18 μm CMOS	0.18 μm CMOS	0.18 μm CMOS
Timing Association		Exclusive	Exclusive	Concurrent
Hardware Association		Sharing	Sharing	Sharing
Operation Mode		Dual	Dual	Triple
Number of RU (8ACSUs)		1	2	2
Logic Gate Count		46K	85K	65K
Memory Size		176K bits	239K bits	161.443K bits (Except IL) (Internal + External)
Clock Frequency		70MHz	128.8MHz	100MHz
Throughput	(1) Convolutonal	384Kbps	< 1.54Mbps	3.12Mbps
	(2) Turbo	12.2Kbps@6 it	4.1Mbps (TC@6 it)	4.17Mbps@6 it
	(3) Both (Assume the same decoding time)	192Kbps (CC) 6.1Kbps (TC@6 it) (Exclusive decoding)	<0.77Mbps (CC) 2.05Mbps (TC@6it) (Exclusive decoding)	1.56Mbps (CC) 4.17Mbps (TC@6 it) (Concurrent decoding)

(a)

		Convolutional Decoder (Single-mode Decoder)	Turbo Decoder (Single-mode Decoder)	Dual-mode Decoder	**Triple-Mode Decoder**
External Logic		-	Interleaver	Interleaver	Interleaver
External Memory		9.728 Kb	122.88 Kb	122.88 Kb	132.608 Kb
Throughput	(1) Convolutonal	3.12Mbps	-	3.12Mbps	3.12Mbps
	(2) Turbo	-	4.17Mbps@6 it	4.17Mbps@6 it	4.17Mbps@6 it
	(3) Both (Assume the same decoding time)	-	-	1.56Mbps (CC) 2.08Mbps (TC@6 it)	1.56Mbps (CC) 4.17Mbps (TC@6 it)

(b)

Triple-Mode Decoder	Ctrl	RUF	RUB	TB	LLR	BM/Gamma0	Gamma1	EETR0	EETR1	Area (%)
(1) Convolutional Decoding	√	√	-	√	-	√	-	√	-	63.15%
(2) Turbo Decoding	√	√	√	-	√	√	√	√	√	92.9%
(3) Concurrent Convolutional /Turbo Decoding	√	√	√	√	√	√	√	√	√	100.0%
Area (%)	24.9%	22.1%	22.0%	7.1%	6.0%	5.0%	4.8%	4.05%	4.05%	

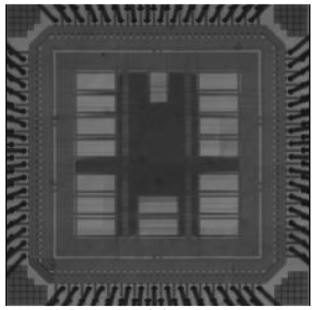
(c)

iterations for Turbo decoding in concurrent VA/MAP mode (assume that the decoding time of Convolutional codes and Turbo codes are the same to compute the throughput rate). Note that the throughput of the MAP decoding is twelve times the one of the turbo decoding since the turbo decoder runs six iterations of two MAP decoding for each frame.

The comparison table is listed in Table III(a), in which our proposed design considers both the timing association and the hardware association. Then, the FEC kernel can be combined external memory or the interleaver for different applications. For 3GPP standards, the extra memory size (except the interleaver/de-interleaver), extra logic, and decoding throughput for four applications, are estimated and listed in Table III(b). It shows that the triple-mode decoder has more about 8% memory than the dual-mode decoder, but the throughput rate of Turbo decoding is double when operating in both decoding. Note that the maximal frame length of Convolutional codes (504) is about 10% the one of Turbo codes (5114). Last, the logic area percentage of the FEC kernel is listed in Table III(c). Compared with the isolated VA part, the FEC kernel has about 58% overhead. Compared with the isolated MAP part, the triple-mode FEC kernel has about 8% overhead. If the FEC kernel is combined with the external memory, its overhead percentage can reduce because the area of the memory in a complete decoder is large. That is the reason we separate the memory banks that depend on standards, and elaborate the computing kernel.

VII. CONCLUSION

In this paper, we have investigated the timing association and hardware association for combining two different algorithms, such as Viterbi and MAP algorithms. We have discussed several timing charts and have presented three techniques, including distribution, pointer and parallel techniques, which can reduce memory or increase throughput in either VA decoding or MAP decoding. Moreover, we have used the tile-based methodology to analyze the key features of timing charts, including computing units and hardware utilization. Consequently, we have proposed the triple-mode VA/MAP timing chart, which satisfies the timing association goal by complementing the idle time in both decoding. Moreover, we have proposed a triple-mode VA/MAP (FEC) kernel that satisfies the hardware association goal by sharing the similar components. Thus, the FEC kernel can be integrated into a FEC decoder with extra memory or logic for four applications: 1) *Convolutional* decoder, 2) *Turbo* decoder, 3) *Dual-mode* decoder, and 4) *Triple-mode* decoder. Our



(a)

Technique		TSMC 0.18 um
Logic Gate		65K
Supply Voltage		1.8V
Max. Frequency		100MHz
Power		320mW@100MHz
Die Size		2.86 x 2.86 mm^2
Through-put	Exclusive VA mode	3.12Mbps
	Exclusive MAP mode	50Mbps
	Concurrent VA/MAP mode	1.56Mbps(VA part) 50Mbps(MAP part)

(b)

Fig. 24. (a) Die photo. (b) Chip summary.

design also considers both hardware and timing association, and has been verified for 3GPP specification.

REFERENCES

[1] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. IT-13, pp. 260–269, Apr. 1967.

[2] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, pp. 268–278, Mar. 1973.

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

[4] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding. Turbo-codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261–1271, Oct. 1996.

[5] *Mobile Radio Communications: Second and Third Generation Cellular and WATM Systems*, R. Steele and L. Hanzo, Eds., 2nd ed. New York: Wiley, 1999.

[6] cdma2000. Third Generation Partnership Project. [Online]. Available: http://www.3gpp.org/

[7] WCDMA. Third Generation Partnership Project 2. [Online]. Available: http://www.3gpp2.org/

[8] L. Bahl, J. Cocke, F. Jelinck, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.

[9] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. Int. Conf. Commun. (ICC'95)*, Jun. 1995, pp. 1009–1013.

[10] K. K. Parhi and T. Nishitani, *Digital Signal Processing for Multimedia Systems*. New York: Marcel Dekker, 1999.

[11] M. M. Mansour and N. R. Shanbhag, "VLSI architecture for SISO-APP decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 4, pp. 627–650, Aug. 2003.

[12] E. Boutillon, W. J. Gross, and P. G. Gulak, "VLSI architectures for the MAP algorithm," *IEEE Trans. Commun.*, vol. 51, no. 2, pp. 175–185, Feb. 2003.

[13] C. Schurgers, F. Catthoor, and M. Engels, "Memory optimization of MAP turbo decoder algorithm," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 2, pp. 305–312, Apr. 2001.

[14] Z. Wang, Z. Chi, and K. K. Parhi, "Area-efficient high-speed decoding schemes for turbo decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 6, pp. 902–912, Dec. 2002.

[15] M. Bickerstaff *et al.*, "A unified turbo/Viterbi channel decoder for 3GPP mobile wireless in 0.18 μm CMOS," in *IEEE ISSCC Dig. Tech. Papers*, 2002, vol. 1, pp. 124–451.

[16] M. C. Shin and I. C. Park, "A programmable turbo decoder for multiple 3G wireless standards," in *IEEE ISSCC Dig. Tech. Papers*, 2003, pp. 154–155.

[17] G. Kreiselmaier, T. Vogt, N. Wehn, and F. Berens, "Combined turbo and convolutional decoder architecture for UMTS wireless applications," in *Proc. 15th Symp. Integr. Circuits Syst. Design*, 2002, pp. 337–342.

Fan-Min Li received the B.S. degree in electrical engineering from National Chung Hsing University, Taichung, Taiwan, R.O.C., in 2000, the M.S. degree in electrical engineering from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 2002, and the Ph.D. degree in electronics engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 2007.

His research interests include the VLSI architecture design, communication systems, and speech signal processing.

Cheng-Hung Lin received the B.S. degree in electronic engineering from Fu Jen Catholic University, Taipei, Taiwan, R.O.C., in 2002, and the M.S. degree in electrical engineering from National Central University, Taoyuan, Taiwan, R.O.C., in 2004. He is currently working toward the Ph.D. degree in electronic engineering at National Taiwan University, Taipei, Taiwan, R.O.C.

His research interests include the design of very large scale integration architectures and circuits for digital signal processing and communication systems. He is currently working on the hardware design for coding systems.

An-Yeu (Andy) Wu (S'91–M'96) received the B.S. degree from National Taiwan University, Taipei, Taiwan, R.O.C., in 1987, and the M.S. and Ph.D. degrees from the University of Maryland, College Park, in 1992 and 1995, respectively, all in electrical engineering.

From August 1995 to July 1996, he was a Member of Technical Staff (MTS) at AT&T Bell Laboratories, Murray Hill, NJ, working on high-speed transmission IC designs. From 1996 to July 2000, he was with the Electrical Engineering Department of National Central University, Taiwan. In August 2000, he joined the faculty of the Department of Electrical Engineering and the Graduate Institute of Electronics Engineering, National Taiwan University, where he is currently a Professor. His research interests include low-power/high-performance VLSI architectures for DSP and communication applications, adaptive/multirate signal processing, reconfigurable broadband access systems and architectures, and SoC platform for software/hardware co-design.

Dr. Wu served as an Associate Editor for *EURASIP Journal of Applied Signal Processing* from 2001 to 2004, and acted as the leading Guest Editor for a special issue on Signal Processing for Broadband Access Systems: Techniques and Implementations of the same journal (published in December 2003). He also served as the Associate Editor of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS from 2003 to 2005. He is now an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I: REGULAR PAPERS. He has served on the technical program committees of many major IEEE International Conferences, including ICIP, SiPS, AP-ASIC, ISCAS, ISPACS, ICME, SOC, and A-SSCC. He received the A-class Research Award from National Science Council four times between 1997 to 2000. He received the Macronix International Corporation (MXIC) Young Chair Professor Award in 2003. In 2004, he received the Distinguished Young Engineer Award from The Chinese Institute of Electrical Engineering, Taiwan. In 2005, he received two research awards, the Dr. Wu Ta-you Award (young Scholar Award) and the President Fu Si-nien Award, from National Science Council and National Taiwan University, respectively, for his research works in VLSI system designs.