A Low-Power VLSI Architecture for Turbo Decoding

Seok-Jun Lee, Naresh R. Shanbhag, and Andrew C. Singer
Coordinated Science Laboratory, ECE Dept.
University of Illinois at Urbana-Champaign
1308 West Main Street, Urbana, IL 61801
[slee6,shanbhag,acsinger]@uiuc.edu

ABSTRACT

Presented in this paper is a low-power architecture for turbo decodings of parallel concatenated convolutional codes. The proposed architecture is derived via the concept of block-interleaved computation followed by folding, retiming and voltage scaling. Block-interleaved computation can be applied to any data processing unit that operates on data blocks and satisfies the following three properties: 1.) computation between blocks are independent, 2.) a block can be segmented into computationally independent sub-blocks, and 3.) computation within a sub-block is recursive. The application of block-interleaved computation, folding and retiming reduces the critical path delay in the add-compare-select (ACS) kernel of MAP decoders by 50% - 84% with an area overhead of 14% - 70%. Subsequent application of voltage scaling results in up to 65% savings in power for block-interleaving depth of 6. Experimental results obtained by transistor-level timing and power analysis tools demonstrate power savings of 20% - 44% for a block-interleaving depth of 2 in 0.25 μ m CMOS process.

Categories and Subject Descriptors

B.7.1 [Types and Design Styles]: VLSI; E.4 [Coding and Information Theory]: Error control codes

General Terms

Design

Keywords

Turbo decoding, low power architecture, MAP decoder

1. INTRODUCTION

Turbo codes [1] have shown remarkable receiver performance improvement over hard-decision decoding. As a result, turbo codes have been accepted as the coding technique in next generation wireless communication standards such

as Wideband CDMA (WCDMA) and the 3rd Generation Partnership Project (3GPP) for IMT-2000. Further, other turbo-based receiver techniques such as turbo equalizers [2] have been proposed in order to leverage the coding gains inherent in turbo processing.

Extensive research has already been done on low-power implementations of turbo code decoders and turbo equalizers [3]–[12]. These include low-power quantization of the log-likelihood ratio (LLR) [3]–[4], early termination [3]–[8], memory optimization [9]–[11], and optimization of computational complexity [12].

In this paper, we propose a low-power architecture for the turbo decoder of parallel concatenated convolutional codes (PCCC) where maximum *a posteriori* probability (MAP) decoder forms the main computational kernel of an iterative decoding process. Note that the above mentioned low-power techniques can be applied on top of the technique proposed in this paper for additional power savings. The proposed architecture is derived via the concept of *block-interleaved computation* followed by folding, retiming and voltage scaling. Block-interleaved computation can be applied to any data processing unit that operates on data blocks and satisfies the following three properties: 1.) computation between blocks are independent, 2.) a block can be segmented into computationally independent sub-blocks, and 3.) computation within a sub-block is recursive. The application of block-interleaved computation, folding and retiming reduces the critical path delay in the add-compare-select (ACS) kernel of the MAP decoder. Subsequent scaling of power supply results in significant power savings in turbo decoders.

The rest of this paper is organized as follows. Section 2 describes the low-power technique of block-interleaved computation. A review of the turbo decoding algorithm is provided in Section 3 followed by a description of the proposed low-power decoder architecture. Experimental results are discussed in Section 4, and Section 5 concludes the paper.

2. BLOCK-INTERLEAVED COMPUTATION

In this section, we describe how low-power operation can be achieved via block-interleaved computation followed by folding, retiming and power supply scaling. Consider the recursive architecture in Fig. 1. Note that the architecture in Fig. 1 cannot be easily pipelined or processed in parallel due to the presence of the feedback loop. However, if the data in a block of length N is processed independent of other blocks and the computations in a block can be reformulated such that a block is segmented into computationally independent sub-blocks, then one can parallelize the architecture

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISLPED'03, August 25–27, 2003, Seoul, Korea.

Copyright 2003 ACM 1-58113-682-X/03/0008 ...\$5.00.

Figure 1: Recursive computation.

Figure 2: Parallel block architecture for block-independent computation.

as shown in Fig. 2, where the parallelization factor M is 2 and a block X is divided into two sub-blocks, X_1 and X_2 .

If we now fold [13] the parallel architecture in Fig. 2 by a factor of $M = 2$, we obtain the folded block-interleaved architecture shown in Fig. 3. Note that the folded block-interleaved architecture is inherently pipelined. Therefore, an application of retiming [13] (see Fig. 4) results in reduction of the critical path delay by a factor of two over that of the original architecture in Fig. 1. It is clear that voltage scaling can now be employed to reduce the power dissipation without impacting the throughput.

In summary, block-interleaved computation requires that processing be done in a sub-block-based manner within a block where computations within a sub-block are recursive and computations between sub-blocks are independent. In such a case, processing M sub-blocks simultaneously results in a folded block-interleaved architecture which is inherently pipelined. Subsequent retiming and voltage scaling can then be used to reduce power while maintaining throughput.

3. LOW-POWER TURBO DECODER ARCHITECTURE

In this section, we first review the turbo decoding algorithm and its VLSI architecture. Next, we apply the technique of block-interleaved computation as described in section 2 to derive a low-power decoder architecture.

3.1 PCCC and its turbo decoding

The turbo coding technique considered in this paper is made up of two recursive systematic convolutional (RSC) encoders connected in parallel as shown in Fig. 5. The bit sequences going from one encoder to another are permuted by an interleaver. The turbo decoder of PCCC contains two

Figure 3: Folded block-interleaved architecture.

Figure 4: Retimed folded block-interleaved architecture.

Figure 5: Block diagram of PCCC and its turbo decoding, where I and I^{-1} denote block interleaving and de-interleaving, respectively.

SISO MAP decoders which are associated with two RSC encoders as depicted in Fig. 5. The decoding of the observed sequences is performed iteratively via exchanging with each other decoder the improved soft output information. Although two SISO MAP decoders process independent observed sequences, turbo decoders can be implemented via the serial architecture as shown in Fig. 6, where one SISO MAP decoder is shared for a pair of decoders in a time-multiplexed way.

Numerous algorithms [14] exist that can be employed for SISO MAP decoders. Among them, the log-domain MAP algorithm (log-MAP) is popular as it formulates the computations as sums rather than multiplies [11]. If $\mathbf{c}$ is a block of N coded bits and $\mathbf{y}$ is a block of N channel output bits, then the goal of the log-MAP algorithm is to estimate the *a posteriori* probability LLR for the k th coded bit, which is defined as

$$\Lambda(c_k) \triangleq \ln \frac{P(c_k = +1|\mathbf{y})}{P(c_k = -1|\mathbf{y})} = \ln \frac{\sum_{\mathbf{c}: c_k = +1} p(\mathbf{c}, \mathbf{y})}{\sum_{\mathbf{c}: c_k = -1} p(\mathbf{c}, \mathbf{y})}. \quad (1)$$

The forward metric ($\alpha_k(s)$), the backward metric ($\beta_k(s)$), and branch metric ($\gamma_k(s', s)$) are defined as [11]:

$$\gamma_k(s', s) = \ln p(y_k|s, s') + \ln p(s|s') \quad (2)$$

$$\alpha_k(s) = \max_{s'} [\alpha_{k-1}(s') + \gamma_k(s', s)] \quad (3)$$

$$\beta_{k-1}(s') = \max_s [\beta_k(s) + \gamma_k(s', s)], \quad (4)$$

where s and s' are trellis states and the $\max^*$ operation is implemented as

$$\max^*(x, y) \approx \max(x, y) + \ln(1 + e^{-|x-y|}). \quad (5)$$

Figure 6: Serial turbo decoding block diagram.

Figure 11: Folded block-interleaved ACS architecture.

Table 1: Complexity comparison between conventional and block-interleaved architectures after synthesis for $M = 2$.

	Relative Area		Critical path delay	
	Logic	Memory	Pipelined ACS	ACS
K=3	1.14	1.10	2.89 nsec	4.91 nsec
K=4	1.20	1.16	4.14 nsec	5.95 nsec
K=5	1.26	1.25	5.29 nsec	9.40 nsec

$(M-1)(N/M)$ in consecutive clock cycles. Compared with the conventional implementation in Fig. 9, the folded block-interleaved architecture in Fig. 11 has M delays in the feedback loop of the ACS recursion. Thus, retiming can be employed to pipeline the critical path by M levels. The price we pay is an increase in the memory requirements and the pipelining registers.

The ratio of the memory requirements of the conventional turbo decoding architecture and that of the proposed block-interleaved turbo decoding architecture is given by

$$\eta = \frac{B_{IN} + B_{DL} + B_b}{B_{IN} + (B_{DL} + B_b)M}, \quad (7)$$

where B_{IN} is the size of the buffer required to hold the inputs and LLR values, and B_{DL} and B_b are the buffer size of the delay-line and backward metric RAMs in Fig. 8, respectively. For $N = 1024$, $L = 16$, $M = 2$, and $K = 3$, η is 1.1 as summarized in Table 1 and linearly increases as M becomes large. The increase in the logic complexity is also linearly proportional to M and the logic complexity ratio of two architectures is determined via synthesis as shown in Table 1, where encoder polynomials, $[5, 7]_s$, $[13, 15]_s$, and $[23, 35]_s$, are employed for $K = 3, 4$, and 5 . We note that the hardware complexity is increased by 14% - 25% and the delay reduced by 30% - 43% for $M = 2$.

In order to reduce power, we scale the supply voltage of the block-interleaved architecture such that the block processing time is equal to that of the conventional architecture. The conventional architecture requires $N + 2L$ cycles to process a block (see Fig. 7). The proposed architecture requires

$$M \times \left(\frac{N}{M} + 2L \right) = N + 2ML, \quad (8)$$

cycles for processing one block (see Fig. 10). Equating the block processing times of the two architectures we get

$$\begin{aligned} \tau_{cri,p} \times (N + 2ML) &= \tau_{cri,s} \times (N + 2L) \\ \tau_{cri,p} &= \tau_{cri,s} \frac{(N + 2L)}{(N + 2ML)}, \end{aligned} \quad (9)$$

where, $\tau_{cri,p}$ and $\tau_{cri,s}$ are the critical path delays of block-interleaved and conventional architectures, respectively. Thus, we can reduce the supply voltage such that $\tau_{cri,p}$ is equal to $\tau_{cri,s} \times \frac{N+2L}{N+2ML}$.

The propagation delay of CMOS circuits is given by

$$\tau = \frac{C_L V_{dd}}{\beta(V_{dd} - V_t)^\alpha}, \quad (10)$$

where C_L is the load capacitance, α is the velocity saturation index, β is the gate transconductance, and V_t is the device threshold voltage [13]. Assuming that $\alpha = 2$ in (10), and substituting (10) into (9), we get

$$\frac{\frac{C_L V_p}{M}}{\beta(V_p - V_t)^2} = \frac{C_L V_s}{\beta(V_s - V_t)^2} \cdot \frac{(N + 2L)}{(N + 2ML)}, \quad (11)$$

where V_s and V_p are the supply voltages of conventional and proposed block-interleaved architectures, respectively. By solving (11), the ratio of supply voltages, defined as $V_{sc} = \frac{V_p}{V_s}$, can be derived as

$$V_{sc} = \frac{(2c_2 + c_1(1 - c_2)^2) + \sqrt{(2c_2 + c_1(1 - c_2)^2)^2 - 4c_2^2}}{2}, \quad (12)$$

where $c_1 = \frac{N+2ML}{M(N+2L)}$ and $c_2 = \frac{V_t}{V_s}$. If $\alpha = 1$, then

$$V_{sc} = \frac{c_2}{1 - c_1(1 - c_2)}. \quad (13)$$

Note that V_{sc} decreases more rapidly for the case when $\alpha = 1$ than for $\alpha = 2$. Therefore, more power savings are expected with technology scaling.

The power consumption of the block-interleaved architecture can be expressed as

$$\begin{aligned} P_p &= C_p \cdot V_p^2 \cdot \frac{1}{\tau_{cri,p}} = A_{sc} C_s \cdot V_{sc}^2 V_s^2 \cdot \frac{N + 2ML}{N + 2L} \frac{1}{\tau_{cri,s}} \\ &= A_{sc} V_{sc}^2 \frac{N + 2ML}{N + 2L} P_s, \end{aligned} \quad (14)$$

where A_{sc} is the area increase factor due to increased memory requirement and P_s is the power consumed by the conventional architecture. Then, the power savings can be computed as

$$P_{sav} = \frac{P_s - P_p}{P_s} \times 100 = (1 - A_{sc} V_{sc}^2 \frac{N + 2ML}{N + 2L}) \times 100. \quad (15)$$

Under the assumptions $\alpha = 1$ and $\alpha = 2$ in (10), the critical path delay ratio, area overhead ratio, and power savings for $K = 3$, $N = 1024$, and $L = 16$, are depicted in Fig. 12. It is observed that at a certain point, there is no further power savings due to the area overhead. Further, as the number of states ($= 2^{K-1}$) is increased, the power savings decrease because the area overhead increases rapidly for large values of K (see Fig. 13).

4. EXPERIMENTAL RESULTS

In this section, the design flow followed in this work is briefly described and experimental results obtained via transistor level timing and power analysis tools are provided for $M = 2$.

Figure 12: Delay ratio ($\frac{\tau_{cri,p}}{\tau_{cri,s}}$), area increase factor ($A_{sc}-1$), and power savings (P_{sav}) versus the number of sub-blocks.

Figure 13: Power savings (P_{sav}) versus different K values

Word-lengths of internal variables of a MAP decoder when applied to a turbo code decoder were determined via computer simulations. The word lengths were chosen such that the loss in coding gain due to quantization effects was less than 0.05 dB at a bit error rate of $= 10^{-4}$.

Several sliding window log-MAP decoders with different constraint lengths and ACS architectures were implemented in VHDL and their functionality was verified. Then, the designs were synthesized in *Synopsys Design Compiler* using TSMC 0.25 μm technology standard cell library. The synthesized design is placed and routed using *Cadence Silicon Ensemble*. *PathMill* was used to determine the critical path from post layout simulations. The supply voltage of the block-interleaved architecture was reduced until the delay of the critical paths in the block-interleaved and conventional architecture were found to satisfy $\tau_{cri,p} = \tau_{cri,s} \frac{(N+2L)}{(N+2ML)}$. *NanoSim* was used to measure the power consumption of

Table 2: Critical path delay(nsec) given $V_s = V_p = 2.5V$ for constraint length K .

Architecture	K=3	K=4	K=5
Conventional	13.994	17.767	20.015
Block-Interleaved	7.126	9.793	12.174

Table 3: Value of V_p at which $\tau_{cri,p} = \tau_{cri,s} \frac{(N+2L)}{(N+2ML)}$.

	K=3	K=4	K=5
V_p	1.7V	1.9V	1.9V

Table 4: Turbo decoding architecture evaluation.

	Conventional design			
	V_{dd}	Throughput	Power	Frequency
K=5	2.5V	4.706Mbps	705mW	50MHz
K=4	2.5V	5.302Mbps	708mW	56.338MHz
K=3	2.5V	6.723Mbps	814mW	71.429MHz
	Proposed low-power design			
	V_{dd}	Throughput	Power	Frequency
K=5	1.9V	4.741Mbps	563mW	53.33MHz
K=4	1.9V	5.229Mbps	496mW	58.824MHz
K=3	1.7V	6.734Mbps	455mW	75.758MHz

MAP decoders and parameterized dual/single port SRAM cells [15] were employed to estimate the memory power consumption of input buffer, LLR buffer, and interleaver.

Measured results on the critical path delay are summarized in Table 2 for $V_s = V_p = 2.5V$. Figure 14 depicts critical path delays of the block-interleaved architecture over different values of V_p , where the dashed lines denote the required critical path delay, $\tau_{cri,p}$, such that $\tau_{cri,p} = \tau_{cri,s} \frac{(N+2L)}{(N+2ML)}$. For $K = 3$, $N = 1024$ and $L = 16$ are employed and for $K = 4$ and $K = 5$, $N = 1024$ and $L = 32$ are chosen because a longer warm-up period is required for larger values of K .

Utilizing reduced supply voltages listed in Table 3 and the post-layout spice netlists, the power consumption of turbo decoding with the conventional and block-interleaved architectures was measured. For measurement purposes, randomly generated input patterns were used. The results are summarized in Table 4, where power savings of 20% - 44% are achieved. As predicted in Fig. 13, the measurement results show that the power savings decrease as the constraint length, K , increases. The throughput in Table 4 is determined by the critical path delay assuming that 5 iterations are carried out.

5. CONCLUDING REMARKS

In this paper, we proposed a new low-power design methodology which exploits block-interleaved pipelining and voltage scaling. The ACS kernel is reformulated via the proposed design methodology thereby achieving power-savings of turbo decoders. Our analysis shows that the power saving is maximized at the interleaving depth, $M = 6$, and decreased after that point because the area overhead become dominant. Experimental results obtained by transistor-level timing and power analysis tools demonstrate power savings of 20% - 44% for a block-interleaving depth of $M = 2$ in 0.25 μm CMOS process. Further, the proposed approach can be applied on top of other low-power turbo decoding design schemes such as early termination.

Figure 14: V_p versus critical path delay measured via PathMill.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant CCR 99-79381 and ITR 00-85929. Authors would like to thank Mr. Brian Lam for his help in physical design and estimating critical path delay and power.

6. REFERENCES

- [1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. of IEEE Int. Conf. on Comm., Geneva*, May 1993, pp. 1064-1070.
- [2] C. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Transactions on Telecommunications*, vol. 6, pp. 507-511, September-October 1995.
- [3] Z. Wang, H. Suzuki, and K. K. Parhi, "VLSI implementation issues of turbo decoder design for wireless applications," in *Proc. of IEEE Signal Processing Systems(SiPS): Design and Implementation*, October 1999, pp. 503-512.
- [4] D. Garrett, B. Xu, and C. Nicol, "Energy efficient turbo decoding for 3G mobile," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'01)*, Huntington Beach, CA, 2001, pp. 328-333.
- [5] H. Suzuki, Z. Wang, and K. K. Parhi, "A K=3, 2Mbps low-power turbo decoder for 3rd generation W-CDMA systems," in *Proc. of IEEE 2000 Custom Integrated Circuits Conf.*, 2000, pp. 39-42.
- [6] O. Leung, C. Yue, C. Tsui, and R. Cheng, "Reducing power consumption of turbo code decoder using adaptive iteration with variable supply voltage," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'99)*, San Diego, CA, 1999, pp. 36-41.
- [7] P. H. Wu and S. M. Pisuk, "Implementation of a low complexity, low power, integer-based turbo decoder," in *Proc. Global Telecommunication Conf.*, vol. 2, 2001 pp. 946-951.
- [8] S. Lee, N. Shanbhag, and A. Singer, "Low-power turbo equalizer architecture," in *Proc. of IEEE Signal Processing Systems(SiPS): Design and Implementation*, October 2002, pp. 33-38.
- [9] D. Garrett and M. Stan, "Low power architecture of the soft-output Viterbi algorithm," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'98)*, Monterey, CA, 1998, pp. 262-267.
- [10] C. Schurgers, F. Catthoor, and M. Engels, "Energy efficient data transfer and storage organization for a MAP turbo decoder module," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'99)*, San Diego, CA, 1999, pp. 76-81.
- [11] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE Journal on Selected areas in comm.*, vol. 16, no. 2, pp. 260-264, Feb. 1998.
- [12] M. M. Mansour and N. R. Shanbhag, "Low-power VLSI decoder architecture for LDPC codes," in *Proc. of IEEE Int. Symp. Low Power Electronics Design (ISLPED'02)*, Monterey, CA, 2002, pp. 284-289.
- [13] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*, John Wiley & Sons, Inc., 1999.
- [14] J. Vogt, K. Koora, A. Finger, and G. Gettweis, "Comparison of different turbo decoder realization for IMT-2000," in *Proc. of Globecom'99*, 1999, 2704-2708.
- [15] M. M. Mansour, M. M. Mansour, and A. Mehrotra, "Parameterized macrocells with accurate delay models for core-based design," in *ISQED 2003*, San Jose, CA, March 2003.