Reconfigurable Turbo Decoder With Parallel Architecture for 3GPP LTE System

Cheng-Chi Wong and Hsie-Chia Chang

***Abstract*—This brief presents a parallel architecture for the turbo decoder using the quadratic permutation polynomial interleaver. The supported block size ranges from 40 to 6144 with an increment of 8, and thus, it includes 188 sizes in the 3rd Generation Partnership Project Long Term Evolution standard. The proposed design can allow one, two, four, or eight soft-in/soft-out decoders to process each block with configurable iterations. To support all data transmissions in the parallel design, a multistage network with low complexity is also utilized. Moreover, a robust path metric initialization is given to improve the performance loss in small blocks and high parallelism. After fabrication in the 90-nm process, the 2.1-mm^2 chip can achieve 130 Mb/s with 219 mW for the size-6144 block and eight iterations.**

***Index Terms*—3rd Generation Partnership Project (3GPP) Long Term Evolution (LTE), quadratic permutation polynomial (QPP) interleaver, turbo decoder.**

I. Introduction

THE TURBO CODE can utilize an iterative decoding process to achieve the near Shannon limit performance [1]. A rate-1/3 turbo codeword is formed by the systematic data along with two parity checks, which are encoded from the information in the original and interleaved orders, respectively. The conventional turbo decoder consists of one soft-in/soft-out (SISO) decoder and two memories for the received codewords and the temporary decoding results. During the iterative decoding process, the SISO decoder calculates the log-likelihood ratio (LLR) for making decision and the extrinsic information for estimating the *a priori* probability. Such process alternates between two half-iterations: one is the component codeword from the original information, and the other is the component codeword from the permuted information.

Many standards adopt turbo codes as their forward error correction techniques [2]. The interleaver is essential to the impressive performance of the turbo code, but its pseudorandom property complicates the parallel processing of a single codeword. A specific mechanism is required to handle parallel data transmission with traditional interleavers [3]. To solve such a problem, the 3rd Generation Partnership Project (3GPP) Long Term Evolution (LTE) standard chooses the quadratic permutation polynomial (QPP) interleaver, whose contention-free property allows multiple SISO decoders to decode one codeword for higher throughput and lower latency [4], [5]. The QPP interleaver of a size-N block is expressed as

$$F(x) = f_1 x + f_2 x^2 \pmod{N} \tag{1}$$

where x stands for the original address, and $F(x)$ is the interleaved address. The determination of f_1 and f_2 is related to the block size [5].

In the 3GPP LTE standard, there are 188 different sizes ranging from 40 to 6144, and each size has its respective interleaver parameters f_1 and f_2. Since 8 is the common factor of all block sizes, any codeword can be processed concurrently by two, four, or eight SISO decoders. The design challenge for these parallel modes is how to transmit parallel data simultaneously through the interconnection between SISO decoders and memory modules. In this brief, a *multistage* network is exploited to resolve this issue. According to the interleaving parameters and parallelism, our design can determine the data directions and arrange the corresponding interconnection. The concept of such an approach is similar to the work in [6], but the parameter conditions and interconnecting mechanism are different. In addition, the proposed design takes the performance loss caused by high parallelism into consideration.

The remainder of this brief is organized as follows. Section II introduces the QPP interleaver and the design issues in a conventional parallel turbo decoder. Section III presents the *multistage* network to support necessary data transmission. Section IV discusses the performance compensation in the proposed design. Section V gives the implementation results, and Section VI concludes this brief.

II. Issues of Conventional Parallel Design

While using P SISO decoders, every received data block is divided into P size-M subblocks ($M = \lceil N/P \rceil$). For a proper address expression in the parallel architecture, we replace the x in (1) with $(sM + j)$, indicating the jth data in the sth subblock. After the substitution, the interleaving address is rewritten as the index q_j in the Q_sth subblock as

$$\begin{aligned} F(sM + j) &= f_1 sM + f_2 s^2 M^2 + 2 f_2 j s M + f_1 j + f_2 j^2 \\ &= Q_s M + q_j \pmod{N}. \end{aligned} \tag{2}$$

Note that $0 \le s,\ Q_s < P$ and $0 \le j,\ q_j < M$. The Q_s is determined by s and j, whereas q_j depends only on j [5].

Each subblock will be stored in one individual memory. In the parallel architecture, the data will be transmitted from the sth memory to the Q_sth SISO decoder. In [7], a recursive approach for generating interleaving addresses $F(x)$ on-the-fly

Manuscript received August 1, 2009; revised November 3, 2009, January 8, 2010, and February 12, 2010; accepted March 24, 2010. Date of publication June 1, 2010; date of current version July 16, 2010. This work was supported by the National Science Council (NSC), Taiwan, under Contract NSC 98-2220-E-009-056. This paper was recommended by Associate Editor M. M. Mansour.

The authors are with the Department of Electronics Engineering and Institute of Electrics, National Chiao Tung University, Hsinchu 300, Taiwan (e-mail: ccweng.ee93g@nctu.edu.tw)

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2010.2048481

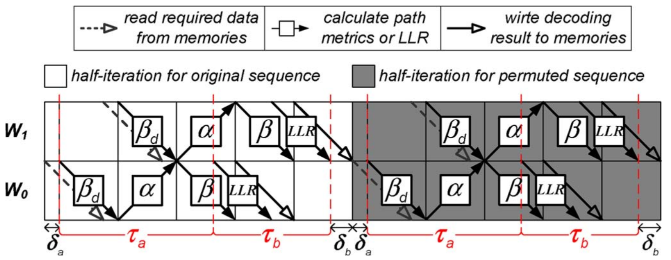

read required data from memories
calculate path metrics or LLR
wirte decoding result to memories
half-iteration for original sequence
half-iteration for permuted sequence
W1
W0
βd
α
β
LLR
δa
τa
τb
δb

Fig. 1. Processing schedule of one SISO decoder. β_d is the dummy backward path metric, β is the backward path metric, and α is the forward path metric.

is illustrated. The computation of j and q_j can be realized with a small address generator rather than a large look-up table. The fully connected network, which supports arbitrary interconnections, is a trivial solution for parallel data transmission. However, its area overhead grows rapidly as P increases, and the routing congestion would be another critical design issue.

The practical SISO decoder always adopts the sliding window method for less overhead [8]. There are three individual units for calculating the forward path metric α, the dummy backward path metric β_d, and the backward path metric β. Fig. 1 shows the process of two windows during two half-iterations [9]. β_d, α, β, and LLR are computed successively within the schedule of each window. Each half-iteration can be divided as follows: both δ_a and δ_b are pipeline delays and memory access time; τ_a is the time to get the necessary metrics for LLR in the first window W_0; and τ_b is the time to derive the LLRs and decisions of all windows. It takes τ_b out of the total execution time to generate outputs, and the ratio is viewed as the operating efficiency η in the following during throughput calculation:

$$\eta = \frac{\tau_b}{\delta_a + \delta_b + \tau_a + \tau_b}. \tag{3}$$

For simplicity, only the radix-2 structure is under discussion, and the window length is represented as L. From Fig. 1, δ_a and δ_b are both less than L cycles, τ_a ranges between $2L$ and $3L$ cycles, and τ_b is exactly $2L$ cycles. When the window number is K, only τ_b grows to $K \times L$ cycles, and the other terms remain unchanged. Here, we assume that the summation of δ_a, δ_b, and τ_a is approximated to $3L$ cycles, indicating $\eta = K/(K+3)$. It is obvious that a smaller K leads to a lower η.

The throughput of a parallel turbo decoder can be calculated from [3], [10]

$$\text{Throughput} = \frac{P \times \mathcal{F} \times \eta}{2 \times \mathcal{I}}. \tag{4}$$

$\mathcal{F}$ is the clock frequency, and $\mathcal{I}$ is the iteration number. With parallel processing, the shorter subblocks will make η decline, and the overall speedup $(P \times \eta)$ will be less than P.

Path metric initialization is another essential issue in the parallel architecture because of the shorten trellis structure. The traditional sliding window method in [8] executes the *dummy calculation* on adjacent windows for initial path metrics. As shown in Fig. 1, it sets all beginning states of β_d equally probable and then executes over several trellis stages to derive a reliable initial β. In addition to the initial β of every window, the parallel architecture would also take some latency to calculate the initial α of every subblock. The work in [11] utilizes the boundary α and β of every window from the previous iteration to initialize α and β in the current iteration. Extra storage elements for the previous value of each subblock are required, and the overhead is considerable if the block has numerous windows [12]. Consequently, the dummy calculation is suitable for large blocks due to consistent cost; the method using previous metrics works better in small blocks for its shorter latency.

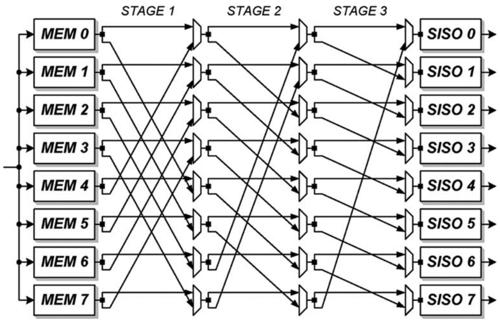


Fig. 2. Multistage network for parallel architecture.

III. Multistage Interconnection Network

From the characteristics of the QPP interleaver, a multistage network based on the barrel shifter is developed for parallel data transmission [13]. Fig. 2 shows the network structure with $P = 8$. For $i = 0 \sim 2$, the proposed network can shift data 2^i locations in stage $(3 - i)$ and accomplish the transmission by using appropriate selection signals upon these two-to-one multiplexers. The behavior of linking each memory module to its corresponding SISO decoder can be regarded as shifting each subblock by a certain offset. The following theorem demonstrates the relationship among the offset of all P subblocks, and it would show that the network can support the 3GPP LTE standard. We will use the notation $a|b$ if a divides b and use $a \nmid b$ otherwise.

Theorem 1: For $2^{i+1}|P$ and any (N, f_1, f_2) in the 3GPP LTE standard, the offset of the sth subblock, which is from s to Q_s, is congruent to the offset of the $(s+2^i)$th subblock modulo 2^{i+1}.

Proof:

1) From (2), the interleaved index q_j can be expressed as

$$\begin{aligned} q_j &= f_1 j + f_2 j^2 \pmod{M} \\ &= f_1 j + f_2 j^2 - \kappa M \end{aligned} \tag{5}$$

where κ is independent of s.

2) We can find $Q_s M$ by substituting (5) for the q_j in (2). Then, Q_s can be derived by the following steps:

$$\begin{aligned} Q_s M &= f_1 s M + f_2 s^2 M^2 + 2 f_2 j s M + \kappa M \pmod{N} \\ Q_s &= f_1 s + f_2 s^2 M + 2 f_2 j s + \kappa \pmod{P}. \end{aligned}$$

3) Let Δs be the difference between s and Q_s, i.e.,

$$\begin{aligned} \Delta s &= Q_s - s \pmod{P} \\ &= (f_1 - 1)s + f_2 s^2 M + 2 f_2 j s + \kappa \pmod{P}. \end{aligned}$$

4) Similarly, $\Delta(s + 2^i)$ is further calculated as

$$\Delta(s+2^i) = \Delta s + (f_1 - 1)2^i + f_2(2^{2i} + 2^{i+1}s)M + 2 f_2 j 2^i.$$

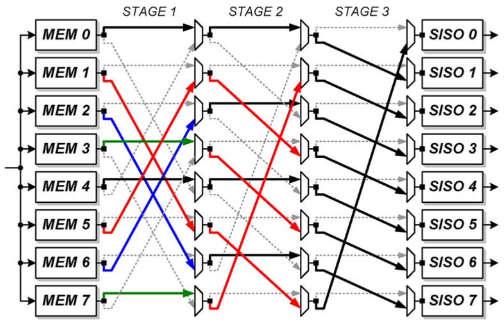


Fig. 3. Data from these eight memories $\{0 \sim 7\}$ are sent to SISO decoders $\{1, 0, 7, 6, 5, 4, 3, 2\}$ via the proposed network.

TABLE I
EQUIVALENT GATE COUNT OF TWO NETWORKS

Parallelism	Fully-connected network	The proposed network
$P = 8$	0.57k (100%)	0.38k (66.7%)
$P = 16$	2.43k (100%)	1.07k (44.0%)
$P = 32$	9.63k (100%)	2.58k (26.8%)

Synthesis result with 90 nm technology: parallel 6-bit data transmission and path delay ≤ 1 ns.

5) Because $(f_1 - 1)$ and f_2 are even numbers in the standard, and 2^{i+1} is a factor of P, we have

$$\Delta s \equiv \Delta(s + 2^i) \pmod{2^{i+1}}. \tag{6}$$

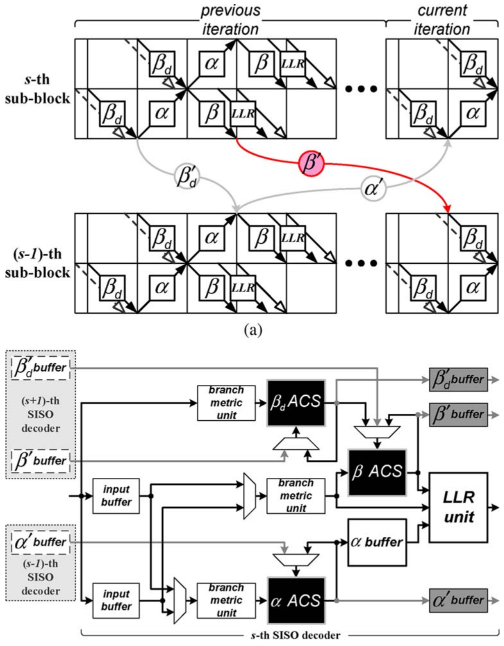
■

The $\Delta s (\bmod P)$ is the shift amount after passing these $\log_2 P$ stages, and its binary expression determines whether the data of the sth subblock are rotated in every stage or not. The congruence in (6) indicates that the last $(i+1)$ bits of Δs and $\Delta(s + 2^i)$ are equivalent so the two subblocks can share the same selection signal in stage $(\log_2 P - i)$. Fig. 3 illustrates an example of parallel data transmission with $N = 64$, $f_1 = 7$, $f_2 = 16$, $P = 8$, and $j = 2$. The Δs's of eight subblocks are $\{1, 7, 5, 3, 1, 7, 5, 3\}$, and these values satisfy (6). The multiplexers with common input sources are controlled by the same 1-bit signal, so there are 4, 2, and 1 controlling bits for the three stages, respectively.

In fact, the above theorem holds as P is a power of 2, $P|N$, $2\nmid f_1$, and $2|f_2$. Our multistage network leads to a lower complexity in the parallel turbo decoder with QPP interleaver. Table I shows the overhead of two interconnecting networks as P is 8, 16, and 32. Both mechanisms have a short path delay so that the data can be transmitted immediately. The proposed network can get a significant area saving, particularly in higher parallelism. In addition, less routing effort can be achieved for all necessary interconnections as well.

IV. PERFORMANCE ANALYSIS IN PARALLEL DESIGN

Before discussing the performance issue, some related characteristics of our design are introduced. The Max-Log-MAP algorithm is exploited [14], and only the rate-1/3 code is considered. The window length is fixed at 16 for less area overhead and tolerable performance loss at around 10^{-5} bit error rate. As M is not divisible by 16, each subblock has $\lfloor M/16 \rfloor$ length-16 windows along with one smaller window. The quantized data include 6-bit received codewords, 9-bit metrics, 10-bit LLR, and 6-bit extrinsic information. A 0.75 scaling factor is applied for extrinsic information [15]. Our design can execute all block sizes for at most eight iterations, and we use fewer iterations for smaller blocks due to the similar performance as compared with further iterations.



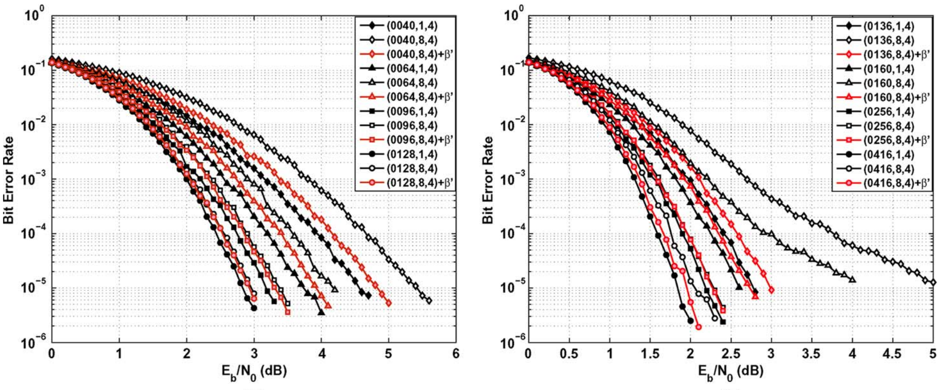
(a)

(b)

Fig. 4. (a) Processing schedule of parallel subblocks. (b) Architecture of the sth SISO decoder in parallel design.

Our design combines the dummy calculation with the previous path metric to support various block sizes with high parallelism. Fig. 4(a) shows the processing schedule of two adjacent subblocks during two successive iterations, and some special initializations are imposed on the parallel architecture. The $\boldsymbol{\beta}'_d$ operation indicates that each SISO decoder will pass the boundary β_d to its backward SISO decoder. Therefore, the first windowed β_d of the sth subblock can update the last windowed β of the $(s-1)$th subblock in the same iteration. Similarly, the $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'$ operations refer to the transmission of path metrics between two iterations. The $\boldsymbol{\alpha}'$ operation can avoid the latency for dummy calculation in every half-iteration. In each subblock, the initial β_d at the last window will be the previous β from the neighboring SISO decoder, whereas the initial β_d's at the other windows are zero. The $\boldsymbol{\beta}'$ operation is used in conjunction with the dummy β_d computation so that it can get a more robust β initialization from a very short trellis. Fig. 4(b) demonstrates the corresponding SISO decoder, where the add–compare–select (ACS) units are used to compute the path metric. Extra buffers and multiplexers for $\boldsymbol{\beta}'_d$, $\boldsymbol{\beta}'$, and $\boldsymbol{\alpha}'$ are added to the conventional architecture in [9]. Compared with the SISO decoder in [12], the previous β's are fed into the β_d-ACS rather than β-ACS.

Fig. 5 presents the fixed-point simulation results of small blocks with $P = 1$ and $P = 8$. The modes with $P = 8$ apply both $\boldsymbol{\alpha}'$ and $\boldsymbol{\beta}'_d$ operations. When parallel processing makes the whole subblock or the last window of each subblock too small, the shortened trellis structure lowers the reliability of path metrics. In these cases, the $\boldsymbol{\beta}'$ operation is used to compensate



(a)

(b)

Fig. 5. Performance of small-size blocks with $P = 1$ and $P = 8$ in the additive white Gaussian noise (AWGN) channel and BPSK modulation; the legend format is (block size, parallelism, iteration), and those legends with β' stand for the use of previous β. (a) Block size: 40, 64, 96, and 128. (b) Block size: 136, 160, 256, and 416.

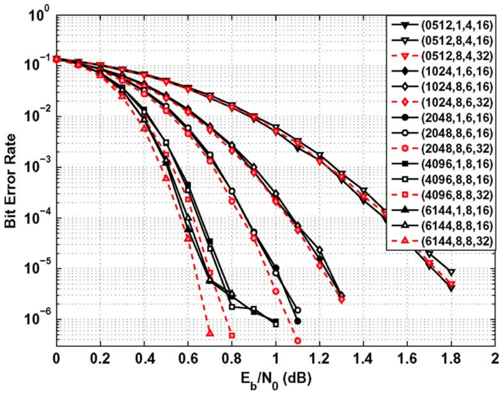


Fig. 6. Performance of large-size blocks (512, 1024, 2048, 4096, and 6144) with $P = 1$ and $P = 8$ in the AWGN channel and BPSK modulation; the legend format is (block size, parallelism, iteration, window length).

the initial β and improve the performance degradation significantly. As shown in Fig. 5(a), the loss of size-40 block at 10^{-5} error rate is reduced from 1.0 to 0.3 dB, whereas the loss of size-64 block is reduced from 0.5 to 0.2 dB. In Fig. 5(b), the size-136, size-160, and size-416 blocks with β' achieve superior performance improvement in $P = 8$. On the other hand, both initialization schemes can achieve similar performance for the modes with $16P|N$, such as $N = 128$ and $N = 256$. Fig. 6 demonstrates the performance of large blocks. From the results whose window lengths are 16, the performance degrades slightly after using multiple SISO decoders. The loss in size-512 block is about 0.1 dB, and the losses in the others can be negligible. However, the error floor regions of size-4096 and size-6144 blocks appear before 10^{-6} error rate. Extending the window can enhance the performance of these blocks, but it would also introduce more area overhead.

The selection of the processing schedule and window length is influenced by performance and hardware cost. When L is 16, one SISO decoder without β' needs 34.6 k gates count. The utilization of β' would increase the equivalent gate count to 36.9 k. It costs low overhead to guarantee the error correction capability of small blocks in the parallel architecture. If the window length L is extended to 32, then the SISO decoder requires additional 10 k gates count to store more temporary data. Since this growth is substantial, we have to make a tradeoff between area and performance.

TABLE II
THROUGHPUT OF SELECTED MODES WITH 275-MHz FREQUENCY

Size (Iteration)	Throughput (Mb/s) / Operating Efficiency η (%)			
	$P = 1$	$P = 2$	$P = 4$	$P = 8$
40 (4)	**16** / 46.5	**21** / 30.3	**31** / 22.7	**47** / 17.2
128 (4)	**25** / 73.6	**40** / 58.2	**56** / 41.0	**71** / 25.8
256 (4)	**29** / 84.8	**51** / 73.6	**80** / 58.2	**113** / 41.0
512 (4)	**32** / 91.8	**58** / 84.8	**101** / 73.6	**160** / 58.2
1024 (6)	**21** / 95.7	**42** / 91.8	**78** / 84.8	**135** / 73.6
2048 (6)	**22** / 97.8	**44** / 95.7	**84** / 91.8	**155** / 84.8
4096 (8)	**16** / 98.9	**34** / 97.8	**66** / 95.7	**126** / 91.8
6144 (8)	**17** / 99.3	**34** / 98.5	**67** / 97.1	**130** / 94.4

V. IMPLEMENTATION RESULTS

The proposed design consists of eight SISO decoders and eight separate memory modules. The block sizes and the interleaving parameters must satisfy $40 \leq N \leq 6144$, $8|N$, $2 \nmid f_1$, and $2|f_2$ due to the available memory and network constraints, and it can support the 188 block sizes in the 3GPP LTE standard. In addition to the configurable iteration number $\mathcal{I}$, the parallelism can be 1, 2, 4, or 8 at each block size. After determining N, f_1, f_2, $\mathcal{I}$, and P, this decoder would initialize the address generator and network controller within 16 cycles; then, it starts decoding the received blocks. Our design is fabricated with 90-nm process and operated successfully at 275 MHz from the measurement results. Table II lists the operating efficiency η and the throughput derived from (4) in various modes. For small blocks, there is a noticeable decline in η, leading to less throughput improvement than large blocks in higher parallelism. When P is 8, the blocks with $N \geq 256$ can achieve 100 Mb/s. To our knowledge, the maximal throughput of a 3GPP LTE turbo decoder chip is 186 Mb/s for size-6144 blocks [16]. By setting $\mathcal{I}$ to 6, the proposed decoder can approximate such target at the expense of 0.1-dB performance loss at 10^{-5} error rate.

Fig. 7 shows the chip microphoto, where the 2.10-mm^2 core area includes 0.634-mm^2 memory. The total gate count is 602 k with 81.02% core utilization. Fig. 8 illustrates the measured power consumption of various sizes in four parallel modes. The power of size-4096 block is 111, 128, 162, and 213 mW, and the power of size-6144 is 111, 128, 164, and 219 mW for different parallel modes (i.e., 1, 2, 4, and 8, respectively). As P increases from 1 to 8, both size-4096 and size-6144 blocks have around 7.6 times speedup while costing double power. The power growth is mainly caused by the increasing switching activity of more utilized SISO decoders,

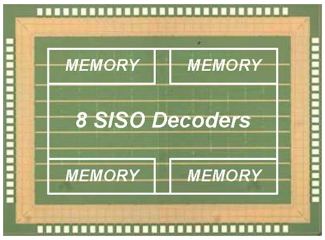


Fig. 7. Power consumption from measurement with 1.0 V and 275 MHz.

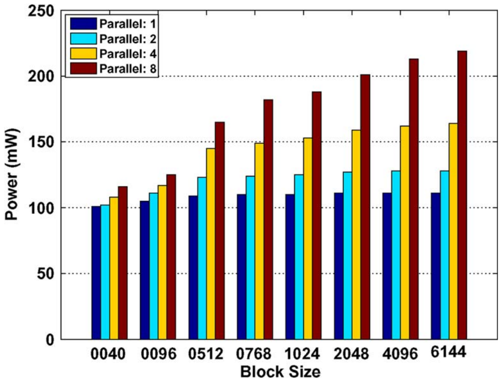


Fig. 8. Power consumption from measurement with 1.0 V and 275 MHz.

TABLE III
CHIP SUMMARY AND COMPARISON OF DIFFERENT PARALLEL DESIGNS

	Proposed	[6]	[16]	[17]	[18]
Max. Block Size	6144	128k	6144	480	6144
Max. Parallelism	8	16	8	10	32
Max. Iteration	8	$\mathcal{I}$	8	4	6
Technology	90 nm	90 nm	130 nm	130 nm	65 nm
Supply Voltage	1.0 V	N/A	1.2 V	1.2 V	N/A
Area (mm^2)	2.10	11.1	10.7 (5.12†)	6.66 (3.20†)	N/A
Gates Count	602k	N/A	800k	635k	N/A
Frequency (MHz)	275	N/A	250	100	200
Throughput (Mb/s)	130	131	186	115	711
Power (mW)	219	762	N/A	197	N/A
Energy Efficiency (nJ/(bit·iteration))	0.21	$\frac{5.80}{\mathcal{I}}$	0.61 (0.20‡)	0.43 (0.14‡)	N/A

† The normalized factor for area is 0.48 $(= (90\ \text{nm}/130\ \text{nm})^2)$.
‡ The normalized factor for energy efficiency is 0.33 $(= (1.0\ \text{V}/1.2\ \text{V})^2 \times (90\ \text{nm}/130\ \text{nm})^2)$ [17].

and there is a certain common power dissipation in all the modes. In addition, the η, affected by block size deeply, is also an important factor to switching activity. When P is fixed, more power is required in larger blocks, and the increment is in proportion to the change in η.

The throughput is 130 Mb/s while using eight SISO decoders to process the size-6144 block for eight iterations. The power consumption is 219 mW with 1-V supply in this mode, and the energy efficiency is 0.21 nJ/(bit $\cdot$ iteration). Table III lists the chip summary and the comparison with simulation results in [6], [17], and [18] and measurement results in [16]. All these works are parallel turbo decoders with contention-free interleavers, and the last three designs utilize the radix-4 structure. Except unavailable iteration $\mathcal{I}$ in [6], the results of each design are derived with its largest block size, iteration, and parallelism. The technology scaling of area and energy efficiency is given for reference.

VI. CONCLUSION

This brief has discussed the parallel architecture of a turbo decoder using QPP interleaver. A multistage network is introduced to handle the necessary parallel data transmission and relieve routing congestion. Moreover, the dummy calculation and the previous boundary path metrics are exploited together for more robust initialization and better performance. Based on these approaches, a turbo decoder chip with the reconfigurable property of block size, interleaving parameters, parallelism, and iteration is fabricated to support the 3GPP LTE system. In conclusion, the proposed method and architecture facilitate the parallel turbo decoder implementation to achieve both higher throughput and flexibility.

ACKNOWLEDGMENT

The authors would like to thank UMC, NCTU Si2 Lab, NCTU-MTK Research Center, and CIC for their assistance.

REFERENCES

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

[2] K. Gracie and M.-H. Hamon, "Turbo and turbo-like codes: Principles and applications in telecommunications," *Proc. IEEE*, vol. 95, no. 6, pp. 1228–1254, Jun. 2007.

[3] M. J. Thul, F. Gilbert, T. Vogt, G. Kreiselmaier, and N. Wehn, "A scalable system architecture for high-throughput turbo-decoders," *J. VLSI Signal Process.*, vol. 39, no. 1/2, pp. 63–77, Jan. 2005.

[4] Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access; Multiplexing and Channel Coding (Release 8), 3GPP Std. TS 36.212, Dec. 2008.

[5] O. Y. Takeshita, "On maximum contention-free interleavers and permutation polynomials over integer rings," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 1249–1253, Mar. 2006.

[6] I. Ahmed and C. Vithanage, "Dynamic reconfiguration approach for high speed turbo decoding using circular rings," in *Proc. 19th ACM Great Lakes Symp. VLSI*, May 2009, pp. 475–480.

[7] R. Asghar, D. Wu, J. Eilert, and D. Liu, "Memory conflict analysis and implementation of a re-configurable interleaver architecture supporting unified parallel turbo decoding," *J. Signal Process. Syst.*, vol. 60, no. 1, pp. 15–29, Jul. 2009.

[8] S. A. Barbulescu, "Iterative decoding of turbo codes and other concatenated codes," Ph.D. dissertation, Univ. South Australia, Adelaide, Australia, 1996.

[9] G. Masera, M. Mazza, G. Piccinini, F. Viglione, and M. Zamboni, "Architectural strategies for low-power VLSI turbo decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 10, no. 3, pp. 279–285, Jun. 2002.

[10] R. Dobkin, M. Peleg, and R. Ginosar, "Parallel VLSI architecture for MAP turbo decoder," in *Proc. IEEE Int. Symp. Pers., Indoor, Mobile Radio Commun.*, Sep. 2002, pp. 15–18.

[11] S. Yoon and Y. Bar-Ness, "A parallel MAP algorithm for low latency turbo decoding," *IEEE Commun. Lett.*, vol. 6, no. 7, pp. 288–290, Jul. 2002.

[12] Z. He, P. Fortier, and S. Roy, "Highly-parallel decoding architecture for convolutional turbo codes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 10, pp. 1147–1151, Oct. 2006.

[13] C.-C. Wong, Y.-Y. Lee, and H.-C. Chang, "A 188-size 2.1 mm^2 reconfigurable turbo decoder chip with parallel architecture for 3GPP LTE system," in *Proc. Symp. VLSI Circuits*, Jun. 2009, pp. 288–289.

[14] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal decoding algorithm," in *Proc. IEEE Int. Conf. Commun.*, Jun. 1995, pp. 1009–1013.

[15] J. Vogt and A. Finger, "Improving the max-log-MAP turbo decoder," *Electron. Lett.*, vol. 36, no. 23, pp. 1937–1939, Nov. 2000.

[16] J.-H. Kim and I.-C. Park, "A unified parallel radix-4 turbo decoder for mobile WiMAX and 3GPP-LTE," in *Proc. IEEE Custom Integr. Circuits Conf.*, Sep. 2009, pp. 487–490.

[17] C.-H. Lin, C.-Y. Chen, A.-Y. Wu, and T.-H. Tsai, "Low-power memory-reduced traceback MAP decoding for double-binary convolutional turbo decoder," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 5, pp. 1005–1016, May 2009.

[18] Y. Sun, Y. Zhu, M. Goel, and J. R. Cavallaro, "Configurable and scalable high throughput turbo decoder architecture for multiple 4G wireless standard," in *Proc. IEEE Int. Conf. Appl.-Specific Syst., Archit. Process.*, Jul. 2008, pp. 209–214.