ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.14, No.18, 1–7

A multi-core-based heterogeneous parallel turbo decoder

Jianmin Zeng[1], Chubin Wu[1], Zhang Zhang[1a)], Xin Cheng[1], Guangjun Xie[1], Jun Han[2], Xiaoyang Zeng[2], and Zhiyi Yu[3]

[1] *School of Electronic Science & Applied Physics, Hefei University of Technology, 193 Tunxi Rd., Hefei, 230009, China*

[2] *State Key Laboratory of ASIC & System, Fudan University, 825 Zhangheng Rd., Shanghai, 201203, China*

[3] *School of Electronics and Information Technology, Sun Yat-sen University, 132 East Waihuan Rd., Guangzhou, 510006, China*

a) *zhangzhang@hfut.edu.cn*

Abstract: It has always been a challenging task to implement a turbo decoder because it's typically the most compute-intensive and time-consuming part in a wireless communication system. This becomes especially obvious when realizing a turbo decoder through CPUs or GPUs. In this paper, we present a heterogeneous and highly reconfigurable parallel turbo decoder for LTE by employing a multi-core processor platform. A modified sliding-window algorithm is proposed to fully exploit the parallelism of turbo decoder, and a SIMD hardware module is designed for the multi-core processor to accelerate the decoding process. Synthesized result in a 65-nm CMOS process shows that the whole system can run at a maximum clock frequency of 830 MHz, and a decoding throughput of 135 Mbps is achieved for a codeword block length of 6144 at 6 iterations. In addition, the speed-up rate compared to an unaccelerated implementation through the same multi-core platform is in the order of 800%.

Keywords: heterogeneous, parallel turbo decoder, multi-core, SDR

Classification: Integrated circuits

References

[1] H. Paul, *et al.*: "Implementation and analysis of forward error correction decoding for cloud-RAN systems," IEEE Int. Conf. Commun. Workshop (2015) 2708 (DOI: 10.1109/ICCW.2015.7247588).

[2] R. Li, *et al.*: "An efficient parallel SOVA-based turbo decoder for software defined radio on GPU," IEICE Trans. Fundamentals **E97-A** (2014) 1027 (DOI: 10.1587/transfun.E97.A.1027).

[3] M. Wu, *et al.*: "HSPA/LTE-A turbo decoder on GPU and multicore CPU," Asilomar Conf. Signals Syst. Comput. (2013) 824 (DOI: 10.1109/ACSSC.2013.6810402).

[4] A. Li, *et al.*: "Implementation of a fully-parallel turbo decoder on a general-purpose graphics processing unit," IEEE Access **4** (2016) 5624 (DOI: 10.1109/ACCESS.2016.2586309).

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017

[5] A. Cassagne, *et al.*: "Beyond Gbps turbo decoder on multi-core CPUs," Int. Symp. Turbo Codes Iterative Inf. Process. (2016) 136 (DOI: 10.1109/ISTC.2016.7593092).
[6] P. Ou, *et al.*: "A 65 nm 39 GOPS/W 24-core processor with 11 Tb/s/W packet-controlled circuit-switched double-layer network-on-chip and heterogeneous execution array," ISSCC Dig. Tech. Papers. (2013) 56 (DOI: 10.1109/ISSCC.2013.6487635).
[7] C.-C. Wong and H.-C. Chang: *Turbo Decoder Architecture for Beyond-4G Applications* (Springer, New York, 2014) 8-25.
[8] J. Huisken and J. Dielissen: "State vector reduction for initialization of sliding windows MAP," Int. Symp. Turbo Codes (2000) 387.
[9] Y. Sun and J. R. Cavallaro: "Efficient hardware implementation of a highly-parallel 3GPP LTE/LTE-advance turbo decoder," Integr. VLSI J. **44** (2011) 305 (DOI: 10.1016/j.vlsi.2010.07.001).
[10] C. Roth, *et al.*: "Efficient parallel turbo-decoding for high-throughput wireless systems," IEEE Trans. Circuits Syst. I, Reg. Papers **61** (2014) 1824 (DOI: 10.1109/TCSI.2013.2290831).

1 Introduction

Turbo codes has been adopted by many digital communication standards such as LTE/LTE-A due to its excellent error correction performance. Turbo decoders are usually implemented through Application Specific Integrated Circuit (ASIC) for reaching high-throughput and low-latency. However, the non-programmability of ASIC makes it inflexible to adapt to the rapidly developing protocols. This will become severe in the future wireless communications when most functions of a radio access network will be virtualized on centralized cloud platforms [1].

To design turbo decoders that are of good flexibility, many researchers tend to adopt programmable devices like GPUs or CPUs. In [2, 3, 4], GPUs are used to realize turbo decoding algorithm for their computing power and programmability. By fully utilizing the enormous parallelism of GPUs, many codeword blocks can be decoded simultaneously. The work in [5] implements a turbo decoder by employing high-end CPUs. However, their implementations may suffer from some problems, either with too low throughput or too high latency.

Our compromise solution between ASICs and GPUs (or CPUs) is by using a multi-core processor platform, a general purpose processor oriented to communication and multi-media. Firstly, we proposed a modified decoding algorithm adaptive to the multi-core processor in order to increase the degree of parallelism. Then a SIMD hardware accelerator module is designed for the processor to accelerate computing. In the end, the modified algorithm is mapped on the multi-core processor, through both pure software and hardware accelerators. The turbo decoder is divided into several sub decoders by using as many cores and can decodes a maximum of four codeword blocks by utilizing SIMD instructions of the processor. Consequently, the parallelism of turbo decoder is fully exploited, in all aspects of trellis stage level, SISO decoder level and turbo decoder level.

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017

2 Architecture of multi-core processor

The multi-core processor [6] we adopt to implement our turbo decoder is shown in Fig. 1. The processor is divided into 6 clusters, forming a 4×6 2D mesh topology. All of the clusters comprise 4 SIMD MIPS-like cores and several bank of memories. The ISA of these MIPS-like cores are compatible with that of MIPS32 4KE series. Besides, each cluster on the four corners in Fig. 1 also contains an execution array, which consists of many dedicated hardware accelerators, such as FFT unit and accelerators for turbo decoding. A hybrid architecture of packet-controlled circuit-switched network is used to globally connect all the cores, which makes the NOC scalable and flexible.

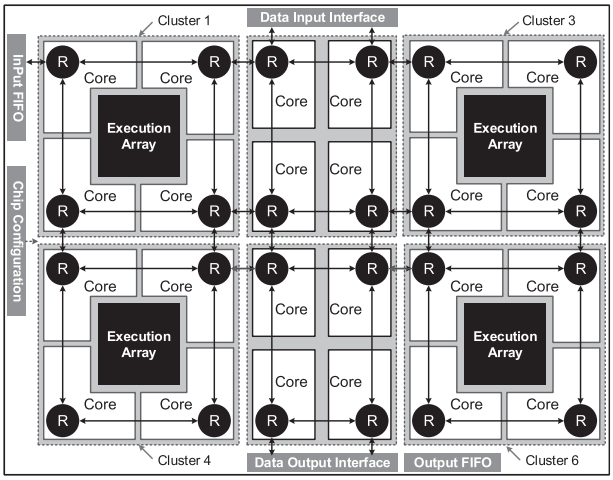

Fig. 1. Architecture of multi-core processor

3 Principles of turbo decoding

In this section, we will briefly review the principles of turbo decoding and the Max-Log-MAP algorithm for 3GPP LTE. Basically, a turbo decoder consists of two component SISO decoders and several interleavers. Each SISO decoder computes the *a posteriori* information L_e by using the channel received information L_s and L_p, and the *a priori* information L_a. The *a posteriori* information then will be used as the *a priori* information of the other SISO decoder after a permutation. Then decoding process goes on iteratively. The branch metrics can be derived as Eq. (1) if a BPSK modulation is applied [7].

$$\gamma(S_k, S_{k+1}) = \frac{1}{2} x_k^{(0)} (L_a(u_k) + L_s(u_k)) + \frac{1}{2} x_k^{(1)} L_p(u_k) \tag{1}$$

Where u_k and $x_k^{(j)} (j \in \{0,1\}, x_k^{(j)} \in \pm 1)$ denotes the *k-th* original information bit and modulated bits, respectively. And the forward and backward state metrics can be recursively computed as follows:

$$\alpha(S_k) = \max_{S_{k-1}} [\gamma(S_{k-1}, S_k), \alpha(S_{k-1})] \tag{2}$$

$$\beta(S_k) = \max_{S_{k+1}} [\gamma(S_k, S_{k+1}), \beta(S_{k+1})] \tag{3}$$

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017

The log-likelihood ratio for the information sequence and the *a posteriori* information will be as Eq. (4) and Eq. (5), where ξ is a scaling factor.

$$LLR(u_k) = \max_{\boldsymbol{u}:u_k=1}[\alpha(S_k) + \gamma(S_k, S_{k+1}) + \beta(S_{k+1})] - \max_{\boldsymbol{u}:u_k=0}[\alpha(S_k) + \gamma(S_k, S_{k+1}) + \beta(S_{k+1})] \quad (4)$$

$$L_e(u_k) = \xi \times (LLR(u_k) - L_a(u_k) - L_s(u_k)) \quad (5)$$

4 Multi-core-based heterogeneous parallel turbo decoder

4.1 Overall architecture

The overall architecture of proposed turbo decoder is presented in Fig. 2. The decoder is split into several heterogeneous sub SISO decoders, and globally managed by a unit called Hub. The Hub unit, made up by two or more MIPS-like cores, is realized by pure software and mainly in charge of data demultiplexing, interleaving, and distributing/gathering data to/from the sub SISO decoders. The data exchange and state synchronization between the Hub and sub SISO decoders is through the global packet-circuit network. This kind of architecture is highly scalable and can be efficiently deployed to the multi-core platform owing to the NOC's flexibility. For instance, up to 8 sub SISO decoders are employed to decode the long codeword blocks for a low latency realization in our experiment.

A sub SISO decoder comprises two cores and two hardware accelerators. Each accelerator is invoked by a core, making up a minimum decoding unit. Data exchange between cores and accelerators is also through a packet-circuit network, which is independent from the global one. Each codeword block is divided into several to dozens of windows. The two decoding units of a sub SISO decoder work simultaneously to decode one window, one processes forwardly and the other backwardly. Note that both the cores and accelerators support SIMD operations. This allows a maximum of four codeword blocks to be decoded simultaneously, exploiting the parallelism in the turbo decoder level.

4.2 Optimized decoding algorithm

A sliding window scheme called Next Iteration Initialization (NII) [8] has been widely employed in many turbo decoder implementations [9, 10] because it requires no dummy backward metrics and guard windows compared to the conventional algorithm. We further split a NII sliding window into two parts, each of whom is decoded by a decoding unit. As previously depicted, the two decoding units of a sub SISO decoder traverse a window from the opposite direction at the same time. Now suppose decoding unit *I* in the Fig. 2 traverses forwardly at the beginning, and unit *II* backwardly, computing α and β, respectively. After half an window's traversal, they exchange their metrics data, and traverse the same half-window inversely, starting from where they were. In this duration, β and α will be computed by unit *I* and unit *II*, respectively. Besides, the *a posteriori* information L_e will also be computed by both of the cores.

For simplicity, the optimized parallel algorithm for decoding unit *I* is depicted as Algorithm 1. Note that step 3 and 4, 5 and 6, 12 and 13 are executed

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017

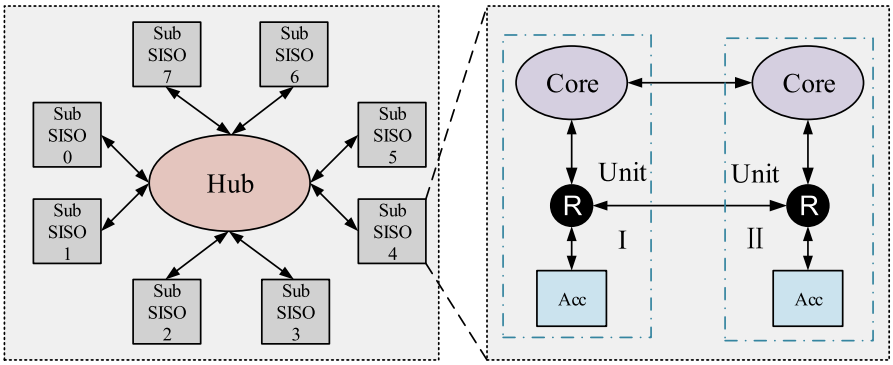

Fig. 2. Overall architecture of the heterogeneous parallel turbo decoder

simultaneously, respectively. The algorithm for the other decoding unit is pretty much the same, and is depicted as Algorithm 2.

Algorithm 1 Optimized decoding algorithm for decoding unit *I*

1: $\alpha[0] \leftarrow alphaInit()$;
2: **for** $k = 1$; $k \leq \frac{w}{2}$; $k = k + 1$ **do**
3: $\gamma[k-1] \leftarrow gammaCalc(L_{sys}[k-1], L_p[k-1], L_e[k-1])$;
4: $gammaStoreToMem(\gamma[k-1])$;
5: $\alpha[k] \leftarrow alphaCalc(\gamma[k-1], alpha[k-1])$;
6: $alphaStoreToMem(\alpha[k]$;
7: **end for**
8: $\beta[\frac{w}{2}] \leftarrow betaInit()$;
9: **for** $k = \frac{w}{2} - 1$; $k \geq 0$; $k = k - 1$; **do**
10: $\gamma[k] \leftarrow gammaLoadFromMem()$;
11: $\alpha[k] \leftarrow alphaLadFromMem()$;
12: $L_e[k] \leftarrow LeCalc(\gamma[k], \alpha[k], \beta[k+1])$;
13: $\beta[k] \leftarrow betaCalc(\gamma[k], \beta[k+1]$;
14: **end for**

4.3 Hardware architecture of accelerator

Fig. 3 shows the hardware architecture of the proposed accelerator module. The accelerator mainly consists of Branch Metric unit (BMU), State Metric unit (SMU), LLR unit (LLRU) and LIFOs. The BMU computes branch metrics and stores them to the LIFO when a decoding unit traverses a half-window for the first time. At the second traversal, the metrics will be loaded from the LIFO. The SMU computes different state metrics at different stages. Take the decoding unit *I* in Fig. 2 for example, the SMU computes forward metrics and stores them to the LIFO at the beginning traversal, and computes backward metrics at the second traversal. The LLRU picks up data from LIFOs and the SMU buffer, and then computes the *a posteriori* information immediately after state metrics are computed at the second traversal.

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017

Algorithm 2 Optimized decoding algorithm for decoding unit *II*

1: $\beta[w] \leftarrow betaInit()$;
2: **for** $k = w - 1$; $k \geq fracw2$; $k = k - 1$; **do**
3: $\gamma[k] \leftarrow gammaCalc(L_{sys}[k], L_p[k], L_e[k])$;
4: $gammaStore(\gamma[k])$;
5: $\beta[k] \leftarrow betaCalc(\gamma[k], \beta[k+1])$;
6: $betaStore(\beta[k])$;
7: **end for**
8: $\alpha[\frac{w}{2}] \leftarrow alphaInit()$;
9: **for** $k = \frac{w}{2}$; $k \leq w - 1$; $k = k + 1$; **do**
10: $\gamma[k] \leftarrow gammaLoad()$;
11: $\beta[k+1] \leftarrow betaLoad()$;
12: $L_e[k] \leftarrow LeCalc(\gamma[k], \alpha[k], \beta[k+1])$;
13: $\alpha[k+1] \leftarrow alphaCalc(\gamma[k], \alpha[k-1])$;
14: **end for**

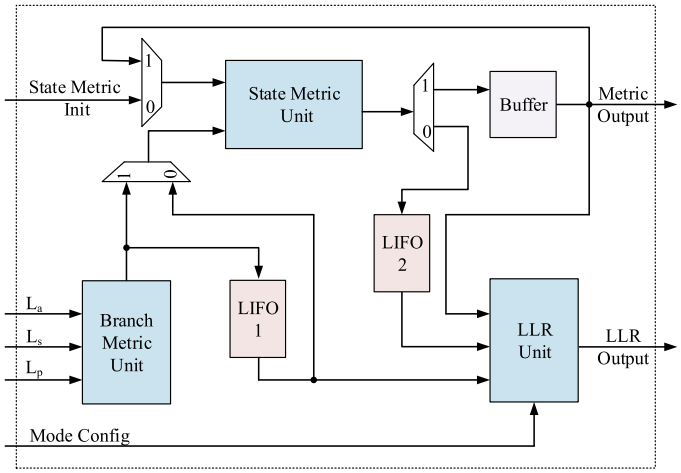

Fig. 3. Hardware architecture of accelerator

5 Result and comparison

A turbo decoder with 8 sub SISO decoders is implemented on the multi-core processor platform by employing 20 cores and 16 accelerators in total, and the processor is programmed by both C and assembly language, which mainly manages the accelerators invocation and SIMD operations. The whole system has been synthesized and simulated in a TSMC 65-nm CMOS LP technology. The decoding throughput in bits per second can be computed as:

$$Thr = \frac{K_{bl} \times N_{bl}}{T_{total}} \tag{6}$$

where K_{bl} is the length of a codeword block, N_{bl} is the number of codeword blocks that can be decoded at one time, and T_{total} represents the total running time of decoding process.

The proposed design is compared with some state-of-the-art turbo decoder implementations in Table I. It's obviously that the others suffer from problems of

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017

too low throughput or too high latency, or both. Take work in [5] for example, although an excellent decoding throughput has been achieved, the decoding latency is too long to be tolerated. We also can see that our design provides a trade-off between latency and throughput. In addition, the heterogeneous turbo decoder achieves an approximate speed-up rate of 800% compared to the unaccelerated implementation through the same multi-core platform.

Table I. Performance comparison with other works. The block length is 6144 for all the works.

Work	Platform	Algorithm	Iters	Latency (μs)	Thr. (Mbps)
[2]	GTX 580	BR-SOVA	5	192	127.8
[3]	GTX 680	EML-MAP	6	2657	37.0
[4]	GTX 680	FPTD	36	403	18.7
[5]	2xE5-2680v3	ELM-MAP	6	3293	716.4
Ours	Multi-core NOC Platform	Modified-MAP	6	**182**	**135.0**

6 Conclusion

This paper proposes a compromise method for designing a parallel and reconfigurable turbo decoder by employing a multi-core processor platform. A modified sliding windows algorithm is proposed to fully exploit the parallelism of turbo decoder and a hardware module is also designed for the multi-core processor to accelerate the decoding process. The results show that decoder can reach a maximum throughput of 135 Mbps at 6 iterations, which is about 8 times that of an unaccelerated implementation by using the same multi-core platform. And it also has advantages over the state-of-the-art implemented through GPUs or CPUs.

Acknowledgments

This work is supported by National Natural Science Foundation of China (61404043, 61674049, 61401137), Anhui science and technology special project (16030901007), the key laboratory of Infrared Imaging Material and Detectors, Shanghai Institute of Technical Physics, CAS (IIMDKFJJ-13-06, IIMDKFJJ-14-03). The authors would like to thank State Key Laboratory of ASIC & System of Fudan University for technical support.

DOI: 10.1587/elex.14.20170768
Received July 24, 2017
Accepted August 17, 2017
Publicized September 4, 2017
Copyedited September 25, 2017