A Near-Future Prediction Method for Low Power Consumption on a Many-Core Processor

Takeshi Kodaka, Akira Takeda, Shunsuke Sasaki, Akira Yokosawa, Toshiki Kizu, Takahiro Tokuyoshi,
Hui Xu, Toru Sano, Hiroyuki Usui, Jun Tanabe, Takashi Miyamori and Nobu Matsumoto
Center for Semiconductor Research and Development, Toshiba Corporation, Kawasaki, Japan

***Abstract*—We developed a method that predicts the required number of cores for executing threads in the near future on a many-core processor. It is designed for low power consumption without performance degradation. The evaluation result confirmed the proposed method is effective on a 32-cores processor.**

I. Introduction

Recently, high performance and low power consumption are important requirements for the embedded SoCs. Many-core processors are proposed as one of the approaches to fulfill them [1]. These processors achieve high performance using parallel processing and reduce power consumption by controlling the power state of cores. For low power consumption, it is necessary to accurately predict the required number of cores for executing an application program.

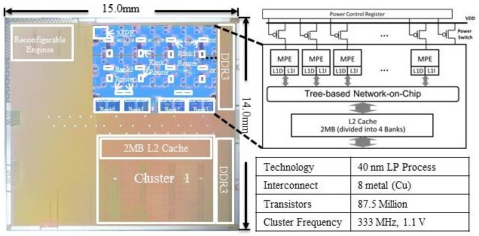
This paper proposes a method that predicts the required number of cores to execute threads by using control information of the scheduler for low power consumption without performance degradation, and reports our evaluation of the method on a many-core processor.

II. Target Hardware Architecture

A chip micrograph and a block diagram of a target many-core processor [1] are shown in Fig. 1. The many-core processor consists of 32 cores. The core in the many-core processor is based on a 3-way VLIW processor called the Media Processing Engine (MPE). Each MPE has an L1 instruction cache and an L1 data cache. In addition to the L1 caches, there is an L2 cache shared among all MPEs and connected through tree-based NoC. This architecture is designed so that the performance improves in proportion to the number of MPEs. Therefore, by changing the number of MPEs, this architecture can handle various applications. For low power consumption, this architecture has a per-core basis power gating. Every MPE has its own power switch.

III. Proposed Near-Future Prediction Method

We designed a method that predicts the required number of MPEs for executing threads in the near future. The method is based on a parallel processing scheme [2] which achieves the scalable performance on a many-core processor. The method's goal is the provision of near-future MPE usage information for low power consumption without performance degradation.



Technology	40 nm LP Process
Interconnect	8 metal (Cu)
Transistors	87.5 Million
Cluster Frequency	333 MHz, 1.1 V

Fig. 1. Chip micrograph and block diagram of the many-core processor

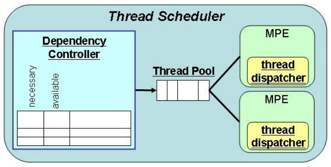


Fig. 2. Block diagram of the base thread scheduler

Fig. 2 shows the structure of the base thread scheduler. To develop the prediction method, we focused on the control data in the dependency controller. The dependency controller has counters to hold the number of necessary data (*necessary count*) and the number of available data (*available count*) for the thread. At first, the number of waiting data (*waiting count*) of each thread in the dependency controller is calculated by Equation 1.

$$waiting\,count = necessary\,count - available\,count \quad (1)$$

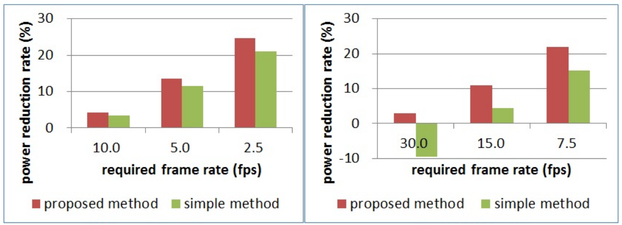
Next, the number of threads whose waiting count is 1 (*wait1 thread count*) and the number of threads ready to execute in the thread pool (*ready thread count*) are counted. Then, the required number of MPEs for executing threads (*predicted required mpe count*) is predicted by Equation 2.

$$predicted\,required\,mpe\,count = \\ ready\,thread\,count + (wait1\,thread\,count - \alpha) \quad (2)$$

In Equation 2, if $(wait1\,thread\,count - \alpha)$ is less than zero, take zero for $(wait1\,thread\,count - \alpha)$, and α is a constant value that depends on the data send-receive relationship among threads to prevent overprediction.

The proposed prediction method uses the number of threads that will become ready to execute in the near future. Consequently, an MPE can be prepared to execute a thread before it becomes ready to run. Therefore, the performance degradation is unlikely to occur.

978-3-9815370-0-0/DATE13/©2013 EDAA



(a) Super Resolution

(b) H.264 decoding

Fig. 3. Power reduction rate

Moreover, the method does not use information related to the number of MPEs on a processor. This means that prediction overhead does not increase even when the number of MPEs in a processor increases. Therefore, the proposed method is suitable for a many-core processor.

IV. Evaluation of the Proposed Method on the Many-Core Processor

In this section, we describe an implementation of the proposed prediction method to the many-core processor introduced in section II and show an evaluation result of the proposed method.

A. *Implementation of the Proposed Method to the Many-Core Processor*

The many-core processor has a per-core power gating that enables control of the supply of power to each MPE. To control the supply of power by power gating, it is necessary to save or restore contexts of the software running on an MPE and the MPE registers.

The proposed prediction method is applied to a thread scheduler. The proposed method is implemented to perform prediction when an MPE fetches a thread. If there is a difference between the number of MPEs that currently used for executing threads and predicted required mpe count, the thread scheduler conducts power operation so that both of them are equal.

B. *Evaluation*

In this evaluation, the super resolution from 1920x1080 to 3840x2180 and the H.264 1080p decoding are used. We decided α of the proposed method based on some trial runs. To compare with the proposed method, a simple prediction method, which monitors idleness of an MPE and turns off the MPE if the length of the idleness exceeds the threshold, is also evaluated.

First, we show the influence of the proposed method on the performance of the applications. We evaluated the performance of the target applications and calculated speed-down ratios against the performance without using the proposed method. The performance degradation of the proposed method is 1.97% for super resolution and 0.04% for H.264 decoding, while that of the simple method is 2.39% and 17.37%, respectively. These results indicate the negative impact of the proposed method on performance is slight.

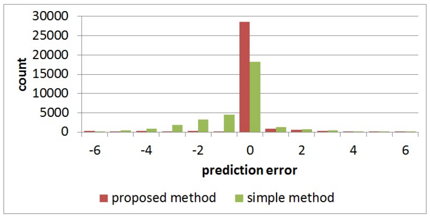


Fig. 4. Prediction error (H.264 decoding 7.5fps)

Next, Fig. 3 shows the power consumption of the proposed method. The result for super resolution is in (a) and that for H.264 decoding is in (b). The x-axis indicates the required performance of the application as a processing frame rate. The y-axis indicates the power reduction ratio against the power consumption without using the proposed method. As shown in Fig. 3, the power reduction ratio becomes higher as required performance decreases. In the case of H.264 decoding at 30 fps, the proposed method achieved power reduction of 3.0% despite the negative reduction (-9.4%) for the simple method. This result confirms the proposed method reduces power consumption efficiently.

Finally, we show prediction error of the proposed method. Fig. 4 is a result for H.264 decoding at 7.5 fps. The x-axis indicates the prediction error that is the difference between the number of ready threads and the number of MPEs that currently use for executing threads controlled by the prediction. The y-axis indicates count of each error. The ratio of zero prediction error, which means no error, is 86.0% for the proposed method, while it is 54.5% for the simple method. This result means the proposed method has better prediction accuracy. Besides, many non-zero prediction errors of the proposed method are positive values. This result indicates that, in accordance with the prediction, MPEs are prepared to run before actual use. It confirms the proposed method operates as we designed.

V. Conclusions

In this paper, we proposed a prediction method for low power consumption without performance degradation. The proposed method predicts the required number of MPEs for executing threads in the near future by using information of waiting data count of threads. The evaluation results of the proposed method on a many-core processor show the proposed method reduce power consumption efficiently without performance degradation. In the next phase of this work, we intend to develop a more effective low power management method such as one combining DVFS with power gating.

References

[1] H. Xu, J. Tanabe, H. Usui, S. Hosoda, T. Sano, K. Yamamoto, et al., “A Low Power Many-Core SoC with Two 32-Core Clusters Connected by Tree Based NoC for Multimedia Applications”, SYMPOSIUM ON VLSI CIRCUITS, pages 150–151, 2012.

[2] T. Kodaka, S. Sasaki, T. Tokuyoshi, R. Ohyama, N. Nonogaki, K. Kitayama, et al., “Design and Implementation of Scalable, Transparent Threads for Multi-Core Media Processor”, in Proc. of the Conf. on DATE, pages 1035-1039, 2009.