Implementation and Evaluation of Fock Matrix Calculation Program on the Cell Processor

Honda, Hiroaki
Research Institute for Information Technology, Kyushu University

Hayashi, Tetsuo
Faculty of Information Science and Electrical Engineering, Kyushu University

Inadomi, Yuichi
Research Institute for Information Technology, Kyushu University

Inoue, Koji
Research Institute for Information Technology, Kyushu University

他

Implementation and Evaluation of Fock Matrix Calculation Program on the Cell Processor

Hiroaki Honda[a)], Tetsuo Hayashi[b)], Yuichi Inadomi[a)], Koji Inoue[a)] and Kazuaki J. Murakami[a,b)]

[a)]*Research Institute for Information Technology, Kyushu University*
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan
[b)]*Faculty of Information Science and Electrical Engineering, Kyushu University*
6-10-1 Hakozaki, Higashi-ku, Fukuoka 812-8581, Japan

Abstract. Various processor architectures have been proposed until today, and the performance has improved remarkably. Recently, the Chip Multi-processors (CMPs), which has many processor cores onto a chip, are proposed for further performance improvement. The Cell processor is one of such CMP and shows high computational performance. Although this processor is designed for the multimedia, that high performance character can be utilized to molecular orbital calculation. In this study we implemented Fock matrix construction program on the Cell processor, and evaluated computational performance. As a result, there were two kinds of main stalls by the branch prediction and the data alignment, which are controlled by software mechanism for the simplification of the Cell processor hardware. It is possible to improve the performance about 30%, if the branch prediction hit ratio could be improved to 99%. For data alignment stall, a part of stalls, which is originated by data shuffle pipeline, could be decreased by preparing hardware data alignment mechanism.

Keywords: Fock matrix calculation, Chip Muti-processor, Cell processor, performance evaluation.
PACS: 02.70.-c, 07.05.Bx, 31.15.Ne

INTRODUCTION

Various processor architectures have been proposed until today, and the performance has improved remarkably. In recent years, several kinds of Chip Multi-processor (CMP) architectures, which contain many processor cores onto a chip, are proposed. It is possible to improve computational performance by two or more parallel executions without dependence by utilizing CMP architectures. The Cell processor was developed as one of such high performance CMPs for multimedia purpose.[1] In spite of the multimedia purpose, the high performance computational feature of the Cell processor could be useful for molecular orbital calculations. There is a great possibility that the CMPs of a high performance those are not the purpose of the scientific calculation will be developed in future.

Only several attempts have so far been made for the quantum chemistry calculations by using special computer hardware. The parallel computations of the Fock matrix have been performed by MOE special parallel computer boards.[2] This MOE computer system has developed into Embedded High Performance Computing (EHPC) computer system using compact-PCI boards.[3] To implement into this machine, the special purpose processor named ERIC was developed for two electron integral calculations. This ERIC has CMP architecture with one initial integration calculation processor core and four recurrence calculation cores. In another study, the qualitative assessment of various computer components was conducted for two electron integral calculations.[4] Recently, the Fock matrix calculation using GRAPE-DR of the SIMD type processor boards was attemped.[5] However, there is no example of the molecular orbital calculations and the discussions of the effectiveness of molecular orbital calculations by using a CMP processor, except the special purpose ERIC processor.

In this study, we implement the two electron Fock matrix calculation program by using the Cell processor as a CMP chip. Subsequently, from the result of performance evaluation, it is discussed what kinds of features of the Cell architecture influence the performance of Fock matrix calculation.

CELL PROCESSOR ARCHITECTURE

The Cell processor contains a Power Processor Element (PPE) core and 8 Synergistic Processor Element (SPE) cores.[1] The Cell architecture is shown in Figure 1. Each SPE contains 256KB scratch-pad Local Storage (LS) memory where its program and data is stored. These 9 processor cores are connected by Element Interconnect Bus (EIB) which organized as two clockwise and two counter clockwise data rings and has 25.6GB/s total bandwidth.

Main features concerning the Cell processor are two points: thread level parallelism by 9 processor cores; and low power consumption by thorough hardware simplification. According to this hardware simplification, it is possible to operate the processor by a high frequency of 4GHz. Therefore, the Cell processor has higher performance than general purpose Pentium 4 processor from the specification. However, following operations are controlled by software programmers:

- **branch prediction** determination whether a conditional branch in the instruction flow of a program is likely to be taken or not,
- **control of on-chip memory load/store** on-chip software control memory although on-chip cache memory of the hardware control is contained on recent high performance processors,
- **data alignment at load/store** data alignment on the memory when data are loaded from the memory to a register and stored from a register to the memory.

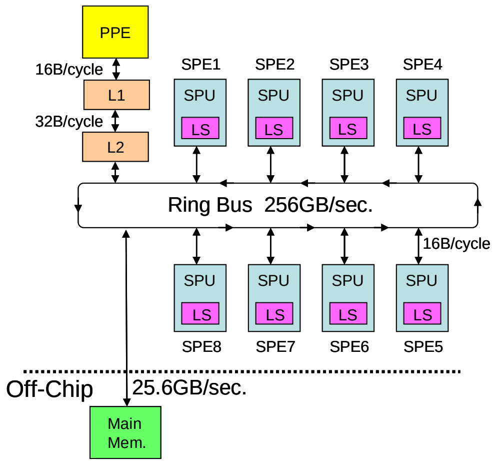

FIGURE 1. Cell processor architecture block diagram

IMPLIMENTATION OF FOCK MATRIX CALCULATION

To implement the Fock matrix calculation program, we employed integral driven algorithm and the general recurrence formula for contracted Gaussian functions by Obara for two electron integral calcutations.[6] By using these schemes, integral driven algorithm is described by shell 4-fold loop structure. The loop body is consisted of Initial Integral Calculation (IIC), Recurrence Calculation (RC), and Partial Fock accumulation calculation (FP). Each successive IIC, RC, and PF does not depend on other shell 4-fold loop elements. Therefore, there are a lot of parallelisms between each loop body calculation.

To accomplish high performance calculation by the Cell processor, it is indispensable to decrease the amount of inter-SPE data transfer and to use load balancing mechanism among SPEs. In this study, each SPE has the role of successive IIC and RC computation and the PPE has PF calculations and task distribution to SPEs. Moreover, for inter PPE and SPEs synchronous communications is accomplished by the implementation of SPEs control data flags saved on the PPE memory, which is possible to be referred and rewrited from each SPE by direct memory access mechanism.

PERFORMACE EVALUATION

Models of Performance Evaluations and Experimental Environments

It is possible to emulate various virtual environments by setting the Cell processor simulator named mambo.[7] In this study, we performed performance evaluations by applying following various Cell environment models:

- **BASE** conventional Cell
- **PBP(Perfect Branch Prediction)** BASE model with 100% hit ratios of all branch predictions
- **EDP(Extended Double Precision)** BASE model with equal performances of double precision and single precision calculations
- **EDP + PBP** combination model of EDP and PBP
- **PentiumD** Pentium D processor of operation frequency 3.2GHz.

For software object code generation of the Cell processor, ppu-gcc and spu-gcc, which are distributed as the Cell processor software development tools, were used with compiler option -O2 –g. For object code generation of PentiumD processor, gcc version 3.2 with compiler option –O2 –g was employed, and PAPI library was linked, which acquires processor hardware counter for performance measurement. Fock matrix calculations were performed for C_4 molecule with DZV basis set.

Results and Discussion

Figure 2 shows the experimental result. Each data consists of the following processing clock cycles:

- **instruction issue** total number of clock cycles that requires to execute whole instructions without stall cycles
- **branch miss stall** number of stall cycles that originates in misses of branch predictions
- **dependency stall** number of stall cycles that originates in data dependencies
- **other stalls** number of stall cycles from other reasons.

Here, stall means a processor idle state when a processor can not process next procedure.

First, we focus our attention to the comparison between BASE and PentiumD models. The result of BASE model shows that the percentage of original instruction issues is only 1/3, and the remaining is occupied by 1/3 branch prediction miss stall and 1/3 data dependency stall. As a result, BASE model consumes 4 times longer execution times than PentiumD model. This shows the contradiction to the high performance of the Cell processor. Main reason of this comes from insufficient loop unrolling code optimizations with large scale registers of the Cell processor.

Next, we turn to discuss about the effect of software branch prediction on the performance. By comparison of BASE and PBP models, branch prediction hit ratio is 60%. If this hit ratio could be improved to 99% as PentiumD processor, it would be possible to improve the performance about 30% for BASE model.

The difference between BASE and EDP models shows the differences between the double precision floating point arithmetic and the single precision one. Each stall clock cycle of EDP model decreases to half. This is because ratios of double precision to single precision arithmetic latencies are 13/6 for both addition and multiplication operations.

Finally, we compare PBP+EDP model with PentiumD one. Total latency of PBP+EDP model is longer than PentiumD model, and a half of this latency came from dependency stalls. The data dependency stalls occur frequently in the place of SHUF and LS processor pipelines. SHUF and LS indicate data alignment and load/store pipeline stage, respectively. It was reported that in IIC sections based on general recurrence formula over contracted Gaussian function, enormous number of load/store instructions are executed for intermediate integrals.[8] Furthermore, from the result of assembly code static inspections, enormous numbers of Load/Store and data alignment shuffle instructions are generated to describe the Load/Stores of intermediate integrals. Therefore, SHUF and LS pipeline stalls are originated from the Load/Stores of intermediate integrals in IIC. This SHUF stall came from software control of data alignment.

It is difficult to drastically decrease the number of Load/Stores while using the same Fock matrix calculation algorithms, however we assume that the processor has a feature of hardware data alignment mechanism, we can say that great part of SHUF stalls would be decreased.

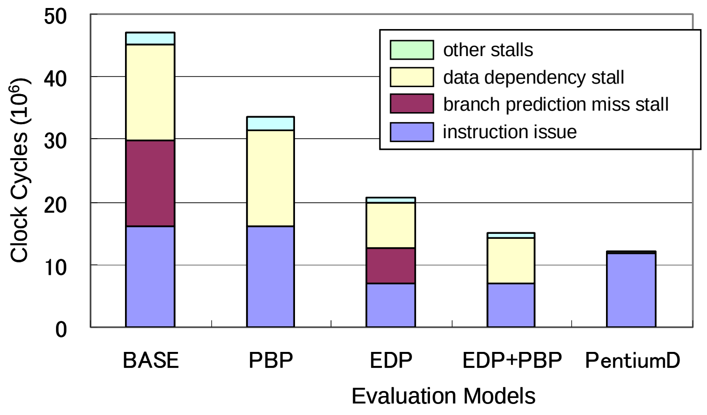

FIGURE 2. Number of total clock cycles of each evaluation model and classified process items

CONCLUSIONS

In this study we implemented Fock matrix calculation program to the Cell processor, which has Chip Multi-processor architecture, and evaluated the performance. The Cell processor has a high performance feature by simplifying hardware. It was shown that the influence of this simplification appeared in the execution time as a stall, and there were two kinds of main stalls by the branch prediction and the data alignment, which are controlled by software mechanism. It is possible to improve the performance about 30%, if the branch prediction hit ratio could be improved to 99%. For data alignment stall, a part of stalls, which is originated by data shuffle pipeline, could be decreased by preparing hardware data alignment mechanism.

ACKNOWLEDGMENTS

This work has been supported in part by the Grant-in-Aid for Young Scientists (A) No.17680005 of the Ministry of Education, Science, Sports and Culture (MEXT). We are grateful for their supports.

REFERENCES

1. "Cell Broadband Engine", http://cell.scei.co.jp/index_e.html.
2. K. Hashimoto *et al.*, *Proceedings of the 1999 ACM/IEEE conference on Supercomputing*.
3. K. Nakamura *et al.*, *Proceedings of HPC-Asia 2002*, December 2002.
   K. Nakamura *et al.*, *Proceedings of the workshop on unique chips and systems(UCAS)*, 87-94, 2005.
4. T. Ramdas, G. Egan, and D. Abramson and K Baldridge, *Theor Chem. Acc.*, **118**, 1432-2234 (2007).
5. Y. Matsubara, K. Yasuda, J. Makino, and S. Ebisuzaki, *Book of Abstructs, 3E07 Bunshikouzou Sougou Touronnkai, Shizuoka, Japan,* September, 2006.
6. H. Honda, T. Yamaki, and S. Obara, **117**, 1457-1469 (2002).
7. "Cell Broadband Engine Software Development Kit", http://www.ibm.com/developerworks/power/cell.
8. Y. Inadomi, S. Obara and U. Nagashima, *J.Comp.Chem.Jpn*, **4**, 175-178 (2005).