Simulation Analyzes of System Characteristics and Parameters for Multithreading Processors

T. Marinov, M. Marinova

Key Words: *Computer systems; Multithreaded processors; Thread-level parallelism; Computer simulation analyses.*

Abstract. *The paper describes an attempt to evaluate the impact on overall processor performance of some of the important parameters in multithreading processing. The performance improvement is analyzed through simulation technique and the influence of number of threads, number of processors in multi-core environment and pipeline efficiency is presented in tables and figures. Some conclusions on the possibility to reach better performance are made, based on the presented results.*

1. Introduction

The achievement of higher performance of computers was always the main goal of computer specialists and constructors. The history of computer evolution is a sequence of steps to achieve more powerful computer systems. There are two main directions to realize these objectives - a technological and an architectural. Nowadays we can suggest that the possibilities of the technological direction are almost exhausted because of the limit of the element speed and their level of integration. That is why the main beliefs for further increasing of the system performance are connected with architectural changes that provide parallel processing at different functional levels. In this paper we evaluate through simulations the main characteristics having strong impact on the computer performance at thread levels.

2. Simulation

Simulation [1] is one of the most powerful analysis tools available to those responsible for the design and operation of complex processes or systems. In an increasingly competitive world, simulation has become a very powerful tool for the planning, design, and control of systems. No longer regarded as the approach of „last resort“, it is today viewed as an indispensable problem-solving methodology for engineers, designers, and managers. We will use the simulation as the process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behavior of the system and/or evaluating various strategies for the operation of the system. We consider the simulation to include both the construction of the model and the experimental use of the model for studying a problem.

The simulator [2] which we use for evaluating of thread-level parallelism in modern processors is CMP-Sim. It is a multi-core micro architectural simulation environment with a detailed cycle-accurate model for the key pipeline structures. CMP-Sim extends the Simple Scalar toolset with accurate models of the pipeline structures. The description is focused only on the changes made with respect to the Simple scalar. The goal of CMP-Sim is to provide a flexible simulation framework in which to conduct academic research related to modern computer architectures. The program can be run in parallel onto maximum 8 threads.

2.1. The Basic Modules of the Simulator

In the first stage Fetch, the bandwidth of the processor for fetching of instructions from threads is modelled. Every thread has owner PC (Program Counter) and thread id.

If branch instruction exists then next sequence of instructions is fetched from the other pointed location from branch

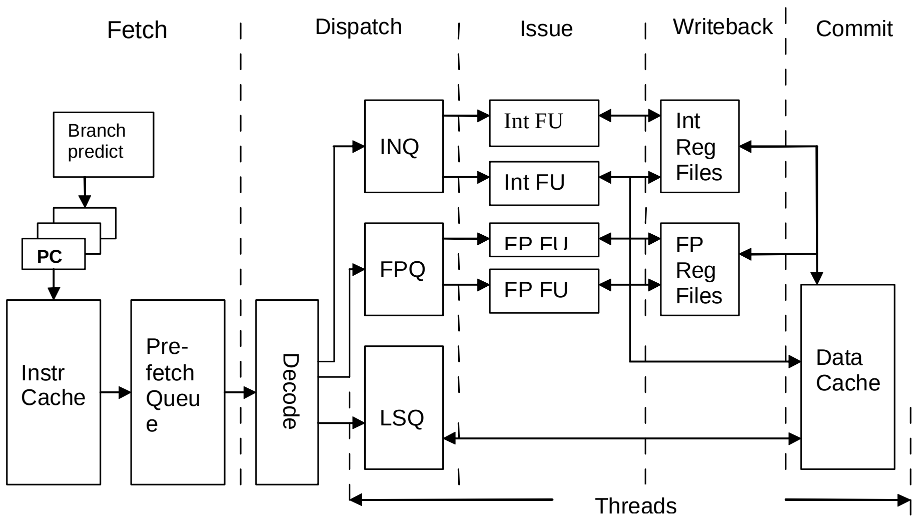

Figure 1. The base architecture of multithreading processor

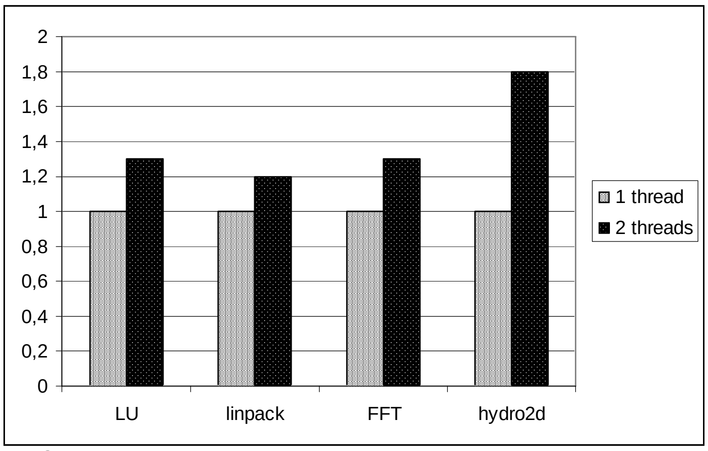

Figure 2. Influence of number of threads over program execution (2 threads)

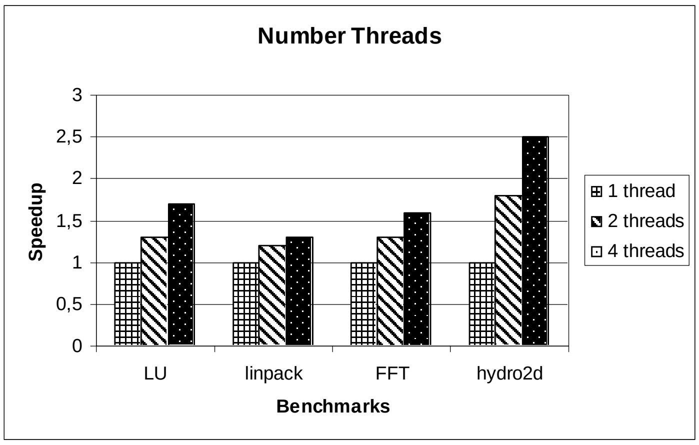

Figure 3. Influence of number of threads over program execution (4 threads)

predictor. After fetching from instruction cache these instructions are issued into prefetch queue. In the second stage Dispatch, the instructions are decoded and analyses for data dependencies between instructions in each thread are performed. After resolving data dependencies instructions are feed back into integer queue, floating-point queue and load-store queue.

When fetched, instructions from the integer queue are feed into integer pipelines for execution, and those from the floating-point queue - into floating-point pipelines respectively. In this stage thread id's are transferred. The previous operation is released when functional pipelines are accessed, and then the oldest operation with his ready operands is committed. The results, generated from functional units (in Writeback stage and Commit stage), are stored in register files (Int Reg Files and FP Reg Files). In the last stage, Commit Stage, the instructions are stored in program order, and the Reorder Buffer is „scanning" for executed instruction into different threads.

3. Simulation Results

Four benchmarks are used for simulations [3] - LU, linpak, FFT, hydro2d. As input parameters we use the number of threads and the number of processors.

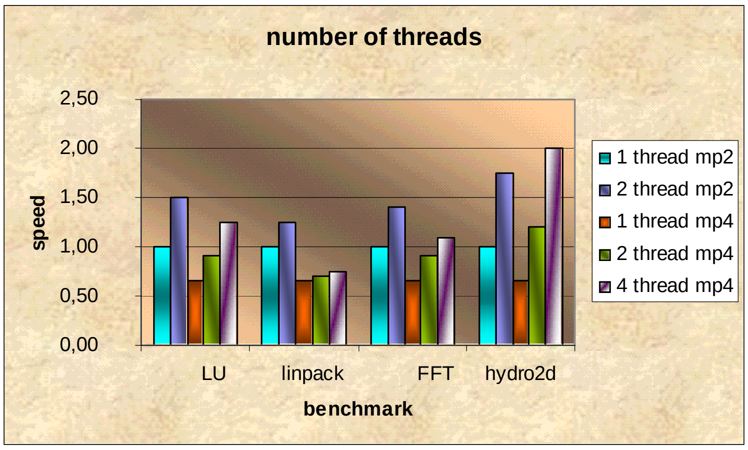

Figure 4. Processor performance in multi-core environment

Table 1. Inefficiency in 2-core architecture

	Integer Issue	Load/Store Issue	FP Issue	Integer IQ	FP IQ	Integer reg	FP reg
LU	37,00	33,00	7,00	15,00	9,00	8,50	1,00
linpack	31,00	20,00	0,00	7,50	0,00	12,00	0,00
FFT	38,00	15,00	8,00	22,00	7,00	3,00	1,50
hydro2d	22,00	17,00	13,00	3,00	23,50	1,00	14,00
Average	32,00	21,25	7,00	11,88	9,88	6,12	4,13

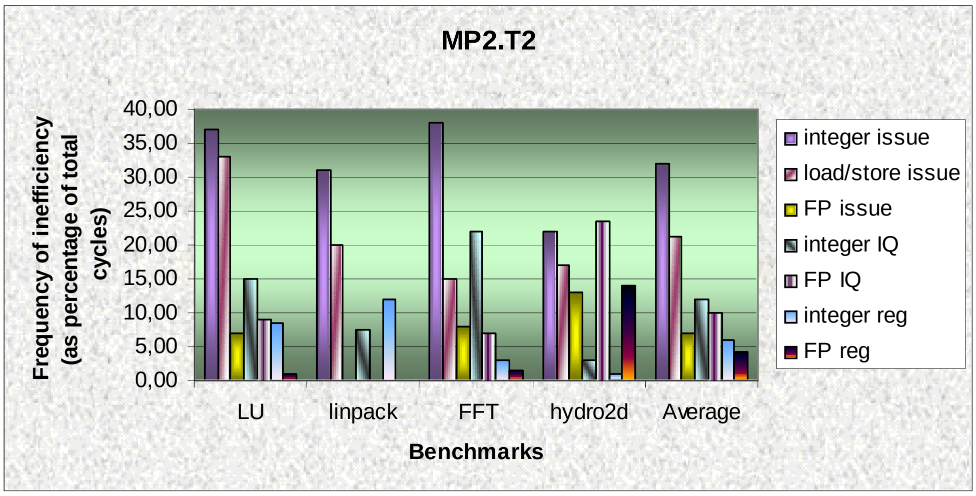

Figure 5. Frequency of pipeline inefficiency with 2-core (MP2.T2)

Table 2. Inefficiency in 4-core architecture

	Integer Issue	Load/Store Issue	FP Issue	Integer IQ	FP IQ	Integer reg	FP reg
LU	31,50	58,50	16,50	2,50	0,00	13,50	5,00
Linpack	24,50	34,50	1,00	0,50	0,00	0,00	0,00
FFT	22,00	26,00	15,50	5,00	0,00	8,50	2,00
hydro2d	17,00	42,50	30,00	1,00	4,50	4,00	21,50
Average	23,75	40,38	15,75	2,25	1,13	6,50	7,13

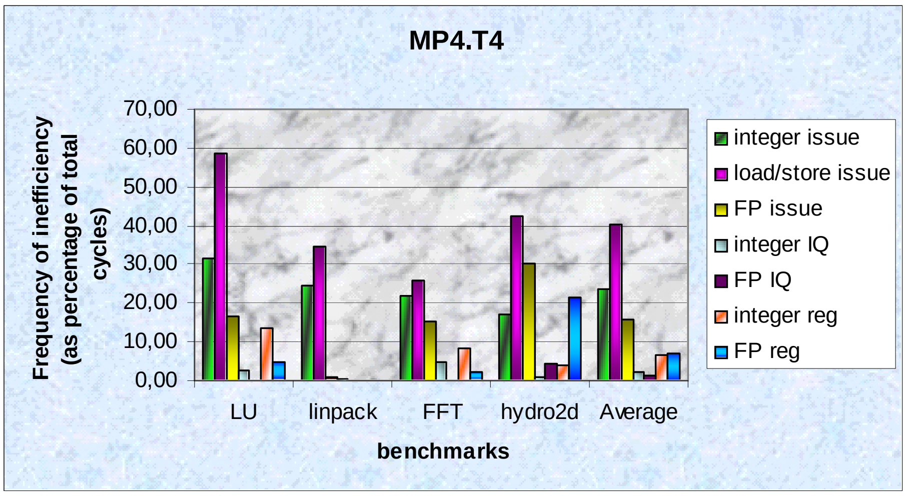

Figure 6. Frequency of pipeline inefficiency with 4-core (MP4.T4)

3.1. Influence of Number of Threads over Processor Performance

The *figure 2* shows that when using two threads the performance improvement of the processor is about one and a half time.

The average improvement is less than two times and depends from the application program. The best results are achieved with benchmark hydro2d. Better improvement will be possible if more sophisticated techniques for data and control dependencies resolving are applied - register renaming, shelving and 3-way branch prediction.

3.2. Influence of Number of Processors for the Overall Performance

The influence of number of processors is analyzed using two processors (MP2) and four processors (MP4) with one-thread and two-thread environment. The upper limit in overall processor performance improvement is about 3 times, depending on the concrete application. These figures are quite promising for the introduction of the multithreading technique as base architecture in multi-core environment.

3.3. Analyses of Pipeline Efficiency

In this paragraph we analyze in details the pipeline performance in multi-core environment (instruction issue, functional units, and instruction queues, renaming registers). The graphics present the frequency of pipeline inefficiency with 2-core (MP2.T2) and 4-core architecture (MP4.T4). In the *tables* for every one of the seven parameters the percentage shows how long the resource was unused during the overall execution.

The pipeline efficiency is a very important characteristic of the multi-core architecture, having very strong impact on the overall performance. Special care should be taken by the designers in order to assure high percentage of pipeline usage for large number of application programs.

4. Conclusions

Based on the simulation results we can assure, that the involving of multithreading in multi-core processors increases their performance in some extent, but the limit is lower than the number of threads, depending strongly on the concrete applications. In some cases it is impossible to perform the instructions speculatively without special techniques for data and flow dependencies resolving. Also, special care should be taken by the designers in order to assure high percentage of pipeline usage for large number of application programs.

In any case we can assume, that the multithreaded processing has the potential to double the processor performance with upper limit in overall processor performance improvement about 3 times for 4-core, depending from the concrete application. These figures are quite promising for the introduction of the multithreading technique as base architecture in multi-core environment.

References

1. Conte, T., C. Gimarc. Brick. Fast Simulation of Computer Architectures. Kluwer Academic Publishers, 1995.
2. Baldawa, S. CMP-SIM: A Flexible CMP Architectural Simulation Environment. UMI, 2008.
3. Stallings, W. Computer Organization and Architecture. Prentice Hal, 2010.

Manuscript received on 7.03.2011

Todor Marinov *is Ph. D. student at IITC-BAS. His intersts include: multicore processor architectures, simulation tools, parallel programming with CUDA C and OpenCL.*

Contacts:
e-mail: todormarinov@yahoo.com.

Maria Marinova *received the Ph.D. degree from Technical University of Sofia - branch Plovdiv in 2006 in Computer Science. Currently, she is an Assistant at Technical University of Sofia - branch Plovdiv, Department Computer Systems and Technologies. Her research interests include: multithreading programming, processor architectures and high performance computers.*

Contacts:
e-mail: m_marinova@tu-plovdiv.bg