A Trace-Level Value Predictor for Contrail Processors

Takenori Koushiro[1] Toshinori Sato[1,2] Itsujiro Arita[1]

[1] Department of Artificial Intelligence, Kyushu Institute of Technology
[2] Center for Microelectronic Systems, Kyushu Institute of Technology
680-4 Kawazu, Iizuka, 820-8502 Japan
{zin,tsato,arita}@mickey.ai.kyutech.ac.jp

Abstract

Contrail processors utilize multithreading for improving energy efficiency. In Contrail, an execution of an application is divided into two streams. One is called the speculation stream. It consists of the main part of the execution and is dispatched into the fast functional units. However, several regions of the execution are skipped by utilizing trace-level value prediction. The other stream is called the verification stream. It supports the speculation stream by verifying each data prediction, and is dispatched into the slow units. The key idea is that the trace-level value prediction translates each critical path into non-critical one and moves it from the speculation stream into the verification stream, and then the non-critical instructions are executed on the slow units. In this paper, we investigate a trace-level value predictor for Contrail processors.

Keywords: chip multi processors, simultaneous multithreading, energy efficiency, value prediction, trace construction

1 Introduction

The increasing popularity of portable and mobile computer platforms such as laptop PCs and smart cell phones is a driving force in the investigation of high-performance and power-efficient microprocessors. For example, modern embedded microprocessors support out-of-order execution. As the computing power of microprocessors for mobile devices increases, however, their power consumption also increases. In addition, while power is already a major design constraint in the area of mobile and embedded computer platforms, it has also become a limiting factor in general-purpose microprocessors.

The energy consumed in a microprocessor is the product of its active power and execution time. Thus, to reduce energy consumption, we should decrease either or both of them. The active power P_{active} and gate delay t_{pd} of a CMOS circuit are given by

$$P_{active} = fC_{load}V_{dd}^2 \quad (1)$$

$$t_{pd} \propto \frac{V_{dd}}{(V_{dd} - V_{th})^{\alpha}} \quad (2)$$

where f is clock frequency, C_{load} is load capacitance, V_{dd} is supply voltage, and V_{th} is the threshold voltage of the device. α is a factor depending upon the carrier velocity saturation and is about 1.3–1.5 in advanced MOSFETs [4]. Based on Eq.(1), it is easily found that power supply reduction is the most effective way to lower power consumption. However, Eq.(2) tells us that supply voltage reduction increases gate delay, resulting in a slower clock frequency. Thus, the computing performance of the microprocessor is diminished.

In order to mitigate the performance loss, we can exploit parallelism [2]. Two identical circuits are used in order to make each unit to work at half the original frequency while the original throughput is maintained. Since the speed requirement for the circuit becomes half, the supply voltage can be decreased. In this case, the amount of parallelism can be increased to further reduce the total power consumption. In this paper, we propose to utilize another kind of parallelism, which is thread level parallelism, for energy reduction with maintaining processor performance.

2 Contrail Processor Architecture

To reduce the energy consumption, we divide an execution of an application into two streams. One is called the *speculation stream* and consists of the main part of the execution. However, it exploits trace-level value prediction [8, 12], and thus several regions of the execution are skipped. In other words, the number of instructions in the speculation stream is smaller than that in the original execution, resulting in energy reduction. In contrast, the other stream is called the *verification stream* and supports the speculation

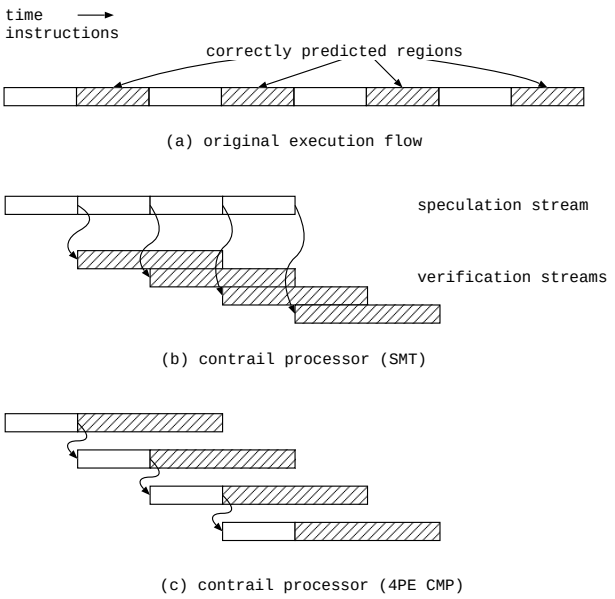

Figure 1: Execution on a Contrail processor

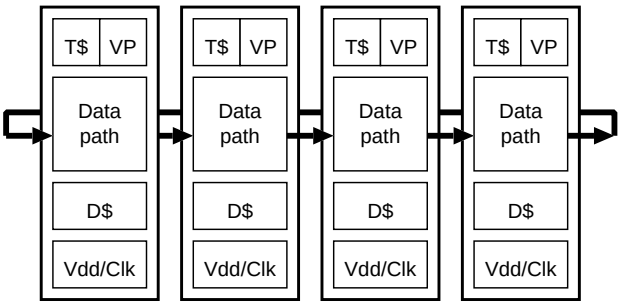

Figure 2: CMP-based Contrail processor

stream by verifying each data prediction. The key idea is that the trace-level value prediction translates each critical path into a non-critical one and we move it from the speculation stream into the verification stream. Hence, the verification stream can execute slowly if the data prediction accuracy is considerably high. We can reduce the clock frequency of the datapath for the verification stream. Furthermore, the supply voltage is also reduced. From these considerations, its energy consumption is significantly reduced. This technique is called Contrail architecture [13,14], because the speculation stream runs ahead just like a jet plane, and the verification stream is left behind by the speculation stream and fades away in the manner of a contrail.

Each stream executes as a thread on a simultaneous multi-threading (SMT) processor [5,20], whose execution core consists of dual speed pipelines, or a chip multiprocessor (CMP) [7,19], each processing element (PE) of which independently works at the variable clock frequency and supply voltage [6, 18]. On the SMT processor, the speculation stream is dispatched into a high-speed pipeline and the verification stream is dispatched into a low-speed and low-supply-voltage pipeline. On the other hand, in the case of the CMP, the speculation stream is executed in a high-speed PE and the verification stream is executed in a low-speed and low-supply-voltage PE. In the ideal case, that means there are no mispredictions; the speculation stream finishes silently and waits for the verification process. In the case in which a misprediction occurs, the execution of the speculation stream is squashed at the point where the misprediction is detected and processor state is recovered by the verification stream.

We explain how the program is logically executed on a Contrail processor, using Figure 1. We assume that half of the original execution of an application is ideally predicted and is distributed uniformly as explained in Figure 1(a). This is a reasonable assumption, since it has been reported that 59% of dynamic traces can be reused with the help of the value prediction [8]. Under this assumption, the execution is divided into speculation and verification streams in a Contrail processor with three contexts (1 and 2 for the speculation and verification streams, respectively) as depicted in Figure 1(b). The predicted regions are skipped in the speculation stream and execute in the verification streams while enlarging their execution time.

In the case of the CMP-based Contrail processor, the speculation and verification streams are executed distributively in several PEs, as shown in Figure 1(c). When an easily predictable region is detected, the head thread forks a speculation stream on the next PE and it turns into a verification stream. Thus, the CMP-based Contrail processor can be build as a ring-connected CMP such as the MultiScalar [15] processor or the SuperThreaded [17] processor, as shown in Figure 2. Each PE has its dedicated voltage/frequency controller. It also has a trace cache [10] and a trace-level value predictor. One of the differences from these processors is that the Contrail processor does not require any mechanism to detect memory dependence violations. Since the Contrail processor strongly relies on trace-level value prediction, any memory dependence violations cannot occur. Instead, it suffers from data value mispredictions. This simplifies hardware

complexity. This is because value mispredictions can be detected locally in an PE, while detecting memory dependence violations requires a complex mechanism such as ARB [15] or Versioning Cache [3].

The potential effect of the Contrail processor architecture on energy-efficiency is estimated as follows: We determine the clock frequency and supply voltage for the verification pipeline at half those for the speculation pipeline. Energy consumption is calculated as follows: For the speculation stream, energy consumption becomes half that of the original execution since the number of instructions is reduced by half. In contrast, for the verification streams, the sum of every execution time remains unchanged since the execution time of each instruction increases doubly while the total number of instructions is reduced by half. Its energy consumption is decreased by the reduction of the clock frequency and the supply voltage. Based on Eq.(1), it is reduced to $\frac{1}{8}$. Thus, the total energy savings is 37.5%. It is true that the effectiveness of Contrail processors (energy savings) depends on the value prediction accuracy and the size of each predicted region. However, we have confirmed that the potential effect of Contrail processors on energy savings is substantial.

One of the differences between this architecture and previously proposed pre-computing architectures [11,16] is that the Contrail processor architecture does not rely on redundant execution. In the ideal case, the number of executed instructions is unchanged. Another difference is that its target is the improvement of energy efficiency instead of the improvement of performance.

3 Trace-Level Value Prediction

As described above, the Contrail processors strongly rely on the trace-level prediction. In order to minimize the overhead in energy consumption due to value prediction, any low-cost trace-level value predictor is required. In this section, we will propose a low-cost trace-level predictor, which we call the decoupled trace-level value predictor.

Figure 3 shows a conventional trace-level value predictor [12], which holds up to four unique values for every register value. The first table, VHT, consists of `Tag`, `LRU Info`, `next PC`, `Register Identifiers`, `Register Values`, and `Value History Pattern` fields. The `Tag` field is used for distinguishing individual among traces. The `LRU Info` field keeps track of the order in which the four data values appear. The four values are identified using binary encoding, {`00, 01, 10, 11`}. The `Register Identifiers` field tells which registers are included in the associated trace. Each `Register Values` holds `State`, `Stride`, and four `Data Values` fields. The `State` field decides whether the prediction should be initiated. The `Stride` field holds the difference between the last two values generated by the instruction. The `Value History Pattern` field saves the history of the last p outcomes for an instruction. The history is maintained as a $2p$-bit pattern consisting of the p binary encoding (2-bit) codes explained above. Every $2p$-bit pattern kept in the `Value History Pattern` field is used for indexing the second table, the Pattern History Table (PHT). An entry in the PHT has four saturating up-down counters, each of which is associated with a corresponding value held in the VHT `Data Values` field. The counter values are used for selecting one of the four values as the predicted value. When an actual value is obtained, the counter corresponding to the correct outcome is incremented and the others are decremented.

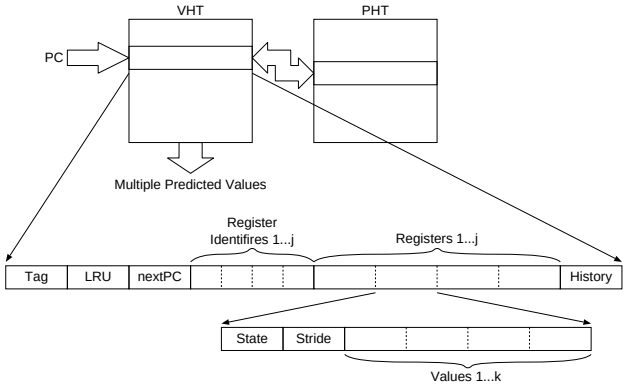

Figure 3: Conventional trace-level value predictor

As can be easily observed, the hardware cost of the conventional trace-level predictor increases significantly as the number of registers included in a trace is increased. In order to solve the hardware cost explosion, we propose to decouple the trace information from the value prediction information as shown in Figure 4. We call this predictor the decoupled trace-level value predictor. It consists of a table keeping trace information, which we call the trace table (TT), and an instruction-level value predictor. The TT consists of `Tag`, 2-bit saturating up-down counter (`2bC`), `next PC`, `Register Identifiers`, and `PCs` fields. The `Tag` field is used for distinguishing individual among traces. The `2bC` field decides whether the prediction should be initiated. The `Register Identifiers` field tells which registers are included in the associated trace,

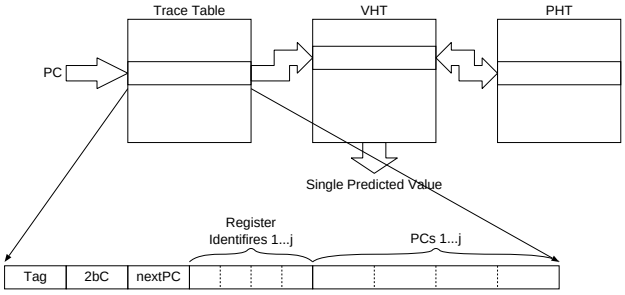

Figure 4: Decoupled trace-level value predictor

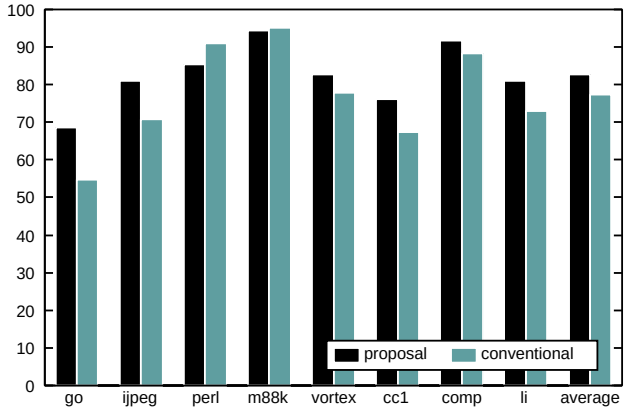

Figure 5: %Prediction accuracies

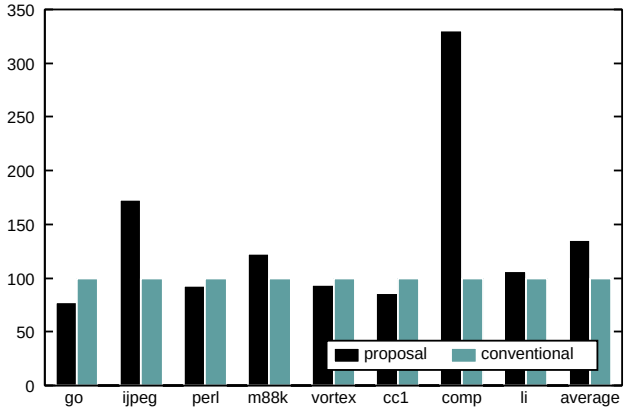

Figure 6: %The number of correctly predicted values

and each instruction address in the `PCs` field is associated with the register identified in the `Register Identifiers` field. Based on the trace information provided by the TT, the instruction-level value predictor is referred multiple times. That is, multiple register values are predicted sequentially. This reduces the hardware cost of the value predictor considerably, while it requires several clock cycles to predict the whole register values in a trace.

4 Results

In this section, we compare the conventional and our proposed decoupled trace-level predictors on prediction accuracy and hardware cost. The evaluation of processor performance is remained for the future study.

We implemented a functional simulator using the SimpleScalar tool set (ver.2.0) [1]. The SimpleScalar/PISA instruction set architecture (ISA) is based on MIPS ISA. In this paper, we use Sathe et al.'s trace-level value predictor [12] as a representative of conventional predictors. It has a 4096-entry VHT. Similarly, our decoupled trace-level predictor as 4096-entry VHT, while the VHT is the instruction-level value predictor. We vary the number of the TT entry between 128 and 4096. We keep four register values per trace. The SPEC95 CINT benchmark suite is used for this study. We use the object files provided by University of Wisconsin Madison [1]. The candidate instructions predicted by the value predictors are register-writing ones, and do not include branch and store instructions.

Figure 5 shows value prediction accuracies of the conventional trace-level predictor (right bar) that has a 4096-entry VHT and the decoupled trace-level predictor (left bar) that has a 4096-entry TT and a 4096-entry VHT. For six of eight programs, the decoupled trace-level predictor achieves higher prediction accuracy. On average, the prediction accuracy is improved by 5.4 point.

Figure 6 shows the difference between the numbers of correctly predicted values of the two trace-level predictors. That of the proposed one (left bar) is shown as the percentage normalized by that of the conventional predictor (right bar). As can be easily observed, except for `ijpeg` and `compress`, the two predictors achieve comparable performance. In contrast, in the cases of the two programs, the decoupled predictor increases the number of correctly predicted values significantly.

Figure 7 shows the effect of the number of entries in the TT on value prediction accuracy. We vary the number of the TT entry between 128 and 4096. For each group of four bars, the bars from left to right indicate the normalized prediction accuracies in the cases where the TT has 128, 512, 1024, and 4096 entries, respectively. We can find that the 1024-entry

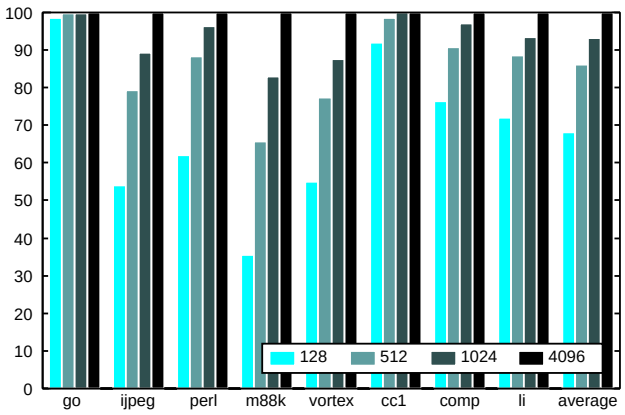

Figure 7: %Change of prediction accuracies

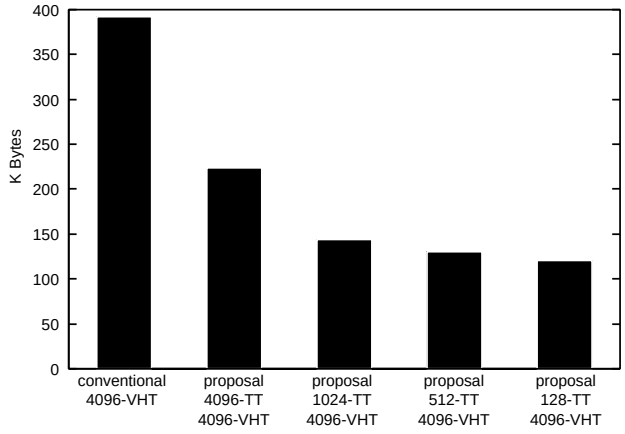

Figure 8: Hardware cost (KB)

TT achieves sufficient accuracy, compared with the 4096-entry TT.

Figure 8 shows the hardware cost of the predictors in bytes. The first bar (see from left to right) indicates the hardware cost of the conventional trace-level predictor that has a 4096-entry VHT. The rest bars indicate the cost of the decoupled predictors that have a 4096-VHT with a 4096-, a 1024-, a 512-, and a 128-entry TTs, respectively. As we have already seen, the decoupled predictor with a 1024-entry TT achieves sufficient prediction performance. Thus, we should compare the first and the third bars. As can be easily observed, the hardware cost of the decoupled predictor is three times smaller than that of the conventional predictor.

5 Summary

Currently, it is expected that multi-threading and dual-power functional units are key techniques for energy reduction [9]. We have proposed such an energy-efficient processor architecture based on value prediction and multi-threading techniques. These techniques exploit thread level parallelism, resulting in mitigating performance loss caused by the supply voltage reduction. The proposed architecture has a potential of approximately 40% energy savings with maintaining processor performance.

In this paper, we investigate trace-level value predictors for the Contrail processors. In order to minimize the overhead in energy consumption due to value prediction, any low-cost trace-level value predictor is required. We proposed the decoupled trace-level value predictor. From the detailed simulation results, we find that the decoupled trace-level predictor can attain the comparable prediction performance of the conventional predictor with less than one third of the hardware cost.

Acknowledgments

This work is supported in part by Grant-in-Aid for Scientific Research (No.13558030) from the Japan Society for the Promotion of Science.

References

[1] D.Burger, T.M.Austin, "The SimpleScalar Tool Set, Version 2.0," ACM SIGARCH Computer Architecture News, vol.25, no.3, June 1997.

[2] A.P.Chandrakasan and R.W.Brodersen, "Minimizing power consumption in digital CMOS circuits," Proceedings of IEEE, vol.83, no.4, April 1995.

[3] S.Gopal, T.N.Vijaykumar, J.E.Smith, G.S.Sohi, "Speculative Versioning Cache", 4th International Symposium on High Performance Computer Architecture, February 1998.

[4] T.Hiramoto, M.Takamiya, "Low Power and Low Voltage MOSFETs with Variable Threshold Voltage Controlled by Back-Bias", IEICE Transactions on Electronics, vol.E83-C, no.2, February 2000.

[5] Intel Corporation, "Introduction to Hyper-Threading technology", White paper, September 2001.

[6] Intel Corporation, "Intel$^{(R)}$ XScale^{TM} Technology," `http://developer.intel.com/design/intelxscale/`, 2002.

[7] K. Olukotun, B.A. Nayfeh, L. Hammond, K. Wilson, K. Chang, "The Case for a Single-Chip multiprocessor," 7th International Conference on Architectural Support for Programming Languages and Operating Systems, October 1996.

[8] M. L. Pilla, A. T. da Costa, F. M. G. Franca, P. O. A. Navaux, "Predicting Trace Inputs with Dynamic Trace Memoization: Determining Speedup Upper Bounds", 10th International Conference on Parallel Architectures and Compilation Techniques, WiP session, September 2001.

[9] J.Rattner, "Electronics in the Internet age", 10th Int. Conf. on Parallel Architectures and Compilation Techniques, Keynote Address, September 2001.

[10] E.Rotenberg, S.Bennet, J.Smith, "Trace Cache: a Low Latency Approach to High Bandwidth Instruction Fetching," 29th International Symposium on Microarchitecture, December 1996.

[11] A.Roth, G.Sohi, "Speculative Data-Driven Multithreading", 7th International Symposium on High-Performance Computer Architecture, January 2001.

[12] R.Sathe, K.Wang, M.Franklin, "Techniques for Performing Highly Accurate Data Value Prediction", Microprocessors and Microsystems, vol.22, no.6, November 1998.

[13] T.Sato, I.Arita, "Contrail Processors for Converting High-Performance into Energy-Efficiency," 10th International Conference on Parallel Architectures and Compilation Techniques, WiP session, September 2001.

[14] T.Sato, T.Koushiro, A.Chiyonobu, I.Arita, "Power and Performance Fitting in Nanometer Design," 5th International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems, January 2002.

[15] G.S.Sohi, S.E.Breach, T.N.Vijaykumar, "Multiscalar Processors," 22nd International Symposium on Computer Architecture, June 1995.

[16] K. Sundaramoorthy, Z. Purser, E. Rotenberg, "Slipstream Processors: Improving Both Performance and Fault Tolerance" 9th International Conference on Architectural Support for Programming Languages and Operating Systems, November 2000.

[17] J-Y.Tsai, P-C.Yew, "The Superthreaded Architecture: Thread Pipelining for Run-time Data Dependence Checking and Control Speculation", 5th International Conference on Parallel Architecture and Compilation Techniques, October 1996.

[18] Transmeta Corporation, "Crusoe^{TM} Processor Model TM5800," Product Brief, May 2001.

[19] M.Tremblay, "An Architecture for the New Millenium," Hot Chips 11, August 1999.

[20] D.M.Tullsen, S.J.Eggers, H.M.Levy, "Simultaneous Multithreading: Maximizing On-Chip Parallelism", 22nd International Symposium on Computer Architecture, June 1995.