Advanced Branch Predictors for Soft Processors

Di Wu, Jorge Albericio and Andreas Moshovos
Electrical and Computer Engineering Department
University of Toronto
peterwudi.wu@utoronto.ca, jorge@eecg.toronto.edu, moshovos@eecg.toronto.edu

***Abstract*—The abstract goes here.**

I. Introduction

Field Programmable Gate Arrays (FPGAs) are increasingly popular to be used in embedded systems. Such designs often employ one or more embedded microprocessors, and there is a trend to migrate these microprocessors to the FPGA platform. Although these soft processors cannot match the performance of a hard processor, soft processors have the advantage that the designers can implement the exact number of processors to efficiently fit the application requirements.

Current commercial soft processors such as Altera's Nios II [1] and Xilinx's Microblaze [2] are in-order pipelines with five to six pipeline stages. These processors are often used for less computation-intensive applications, for instance, system control tasks.

The architecture of current commercial soft processors are based on simple single-issue pipelines with few variations, such as the Xilinx Microblaze [2] with three-stage pipeline and the Altera Nios II [3] with up to five pipeline stages. Despite their ease of use relative to FPGA hardware design, they are predominantly used for system control tasks because of the large disparity between the performance possible on a soft processor versus FPGA hardware. To support more compute-intensive tasks on soft processors, they must be able to scale up performance by using increased FPGA resources.While this problem has been thoroughly studied in traditional hard processors [4], an FPGA substrate leads to different tradeoffs and conclusions. In addition, traditional processor architecture research favoured features that benefit a large application domain, while in a soft processor we can appreciate features which benefit only a few applications since each soft processor can be configured to exactly match the application it is executing. These key differences motivate new research into scaling the performance of existing soft processors while considering the configurability and internal architecture of FPGAs.

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

II. Perceptron Predictor

Section **??** introduced that perceptron predictor maintains vectors of weights in a table. It produces a prediction through the following steps: (1) a vector of weight (i.e., a perceptron) is loaded from the table. (2) multiply the weights with their corresponding global history (1 for taken and -1 for not-taken). (3) sum up all the products, predict taken if the sum is positive, and not-taken otherwise. Each of these steps poses difficulties to map to the FPGA platform. The rest of this section addresses these problems.

A. Perceptron Table Organization

Each weight in a perceptron is typically 8-bit wide, and perceptron predictors usually use at least 12-bit global history [3]. The depth of the table, on the other hand, tends to be relatively shallower (e.g. 64 entries for 1KB hardware budget). This requires a very wide but shallow memory, which does not map well to BRAMs on FPGAs. For example, the widest configuration that a M9K BRAM on Altera Stratix IV chip is 36-bit wide times 1k entries. If we implement the 1KB perceptron from [3], which uses 96-bit wide perceptrons 12-bit global history

As Fig. **??** shows, 64-entry

B. Multiplication

C. Adder Tree

III. TAgged GEometric history length Branch Predictor (TAGE)

IV. Conclusion

The conclusion goes here.

References

[1] Altera Corp., "Nios II Processor Reference Handbook v9.0," 2009.

[2] *MicroBlaze Processor Reference Guide*, Xilinx Inc., July 2012.

[3] D. A. Jimenez and C. Lin, "Dynamic Branch Prediction with Perceptrons," in *Intl' Symposium on High-Performance Computer Architecture*, January 2001.