JKIICE

Short Paper

한국정보통신학회논문지 Vol. 23, No. 12: 1734~1737, Dec. 2019

슈퍼스칼라 프로세서를 위한 고성능 하이브리드 동적 분기 예측

Hybrid Dynamic Branch Prediction to Reduce Destructive Aliasing

Jongsu Park[*]

[*]Staff Engineer, System LSI Division, Samsung Electronics, Hwaseong, 18448 Korea

ABSTRACT

This paper presents a prediction structure with a Hybrid Dynamic Branch Prediction (HDBP) scheme which decreases the number of stalls. In the application, a branch history register is dynamically adjusted to produce more unique index values of pattern history table (PHT). The number of stalls is also reduced by using the modified gshare predictor with a long history register folding scheme. The aliasing rate decreased to 44.1% and the miss prediction rate decreased to 19.06% on average compared with the gshare branch predictor, one of the most popular two-level branch predictors. Moreover, Compared with the gshare, an average improvement of 1.28% instructions per cycle (IPC) was achieved. Thus, with regard to the accuracy of branch prediction, the HDBP is remarkably useful in boosting the overall performance of the superscalar processor.

Keywords : Branch, Dynamic Prediction, Branch History, Prediction Accuracy

Ⅰ. Introduction

The most recent studies on high performance superscalar processor architecture focused on implementing deeper pipelining, instruction prefetching, multiple issuing, and dynamic scheduling to improve performance [1,2]. However, we could not obtain the satisfactory results with these innovations because the misprediction penalty becomes higher with the usage of deeper pipelining and the wider issue technique [3]. Better branch prediction mechanisms are vital to prevent the flushed cycles caused by misprediction. Therefore, more accurate branch predictors are required to achieve high performance. Many studies have been proposed on two-bit saturating counter and a two-level branch predictor consisting of one level of PHT whose entries are also filled with two-bit saturating counter information [4]. McFarling also proposed a variation of the global-history two-level predictor called gshare. The gshare algorithm attempts to make better use of the index bits by hashing the branch history register (BHR) and the program counter (PC) to select an entry from the PHT. The combination of the BHR and PC tends to contain more information because of the non-uniform distribution of PC values and branch histories [5]. Recently, tracking data dependency and its application to branch prediction was proposed in response to the importance of selecting the optimal history length [6]. In addition, the Dynamic per Branch History Length Adjustment (DpBHLA) branch predictor adjusts BHR length both dynamically and logically [7]. Whenever a branch instruction is executed, this branch predictor saves the information in the data dependent history table, which includes register writing instructions, such as load or computation instructions. The record is saved in the branch register dependency table (BRDT), whose entry stores the key branch information and is equal to the number of physical registers.

The proposed HDBP scheme adjusts dynamically the most efficient history length per branch instruction, which records data dependencies among the branches,

Received 25 November 2019, **Revised** 29 November 2019, **Accepted** 29 November 2019

Open Access http://doi.org/10.6109/jkiice.2019.23.12.1734 print ISSN: 2234-4772 online ISSN: 2288-4165

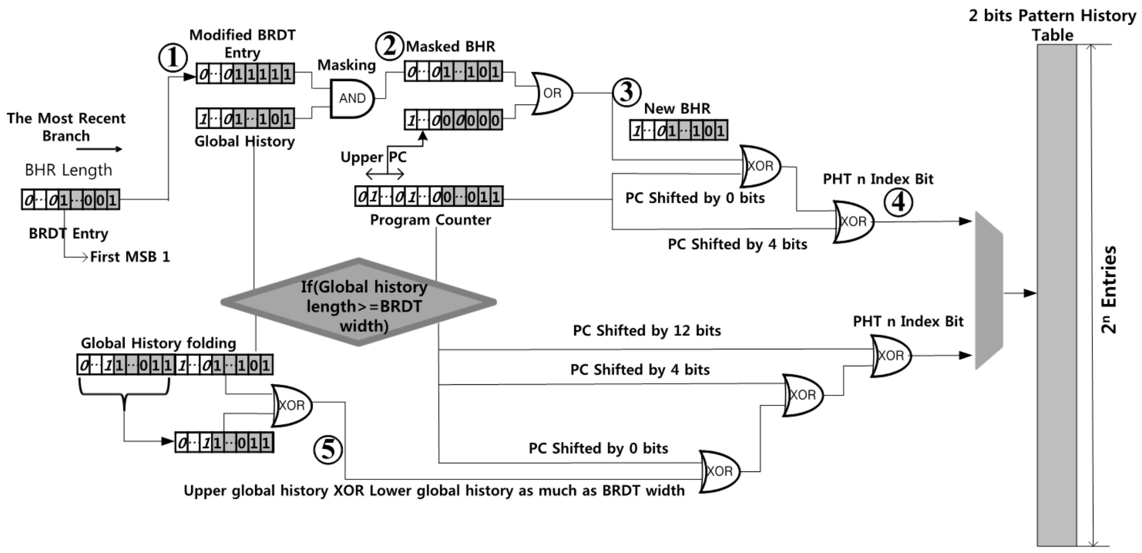

Fig. 1 Proposed Architecture

decides the optimal length of the global history, and utilizes a diversely shifted program counter to increase the index rate of the PHT entries. Furthermore, if the global history register (GHR) length is longer than the BRDT width, GHR is folded in half with XOR logic to make a unique pattern. Thus, a dynamic history length adjustment predictor, as well as modified gshare predictor with a long history length, is provided.

Ⅱ. Proposed Architecture

A recent study suggested a dynamic branch predictor that combined PC and branch history in a dynamic way because the best history length depends on program [7-8]. This section presents the HDBP which takes advantage of both the dynamic branch predictor and the static predictor. Fig. 1 shows a block diagram of the HDBP. The HDBP records a basic block execution that includes register writing instructions at the BRDT. Therefore, when a register has data dependency with previously executed basic blocks, the data dependency is easily recognized by the indexing BRDT. Because a branch path is judged by a register in the instruction, the known register's data dependency allows more accurate branch prediction. Based on the contents of the BRDT, the proposed architecture examines how many determined the number of branch instructions existing between the current branch instruction and previous branches that have data dependency with a register in the current executing branch instruction. After examination, the proposed architecture chooses BHR bits which are necessary for branch prediction. First, each BRDT entry records the correlated information that the latest computation writes into the related physical register. “1” means that the present executing branch instruction in the global history is affected by the physical register. “0” means there is no affection. The proposed architecture sets all to “1” from the first most significant bit (MSB) 1 to the least significant bit (LSB) to obtain the related global history length. Second, ANDing-logic is used between the modified BRDT entry and global history to obtain strongly correlated bits in the global history register. Third, the new BHR and the upper PC is required to avoid the aliasing problem. Using the appropriate upper PC value instead of resetting to 0 in masked BHR decreases the aliasing occurrence rate and accesses different PHT entries per branch. Fourth, the proposed architecture accesses the PHT entry after doing exclusive-OR arithmetic between the new BHR and PC shifted by 0 and 4 bits to make a unique pattern. Fifth, the gshare predictor was modified, which is capable of including information about a lengthy branch history to achieve a high prediction rate with a variety of shifted PC values. The proposed architecture uses the global history folding technique under specific conditions because the history length of the conventional gshare preditor is as limited as the PHT entries are. For a specific branch condition, some of the correlated branches may have appeared at a large distance in the dynamic instruction stream. It sometimes includes important information, which is apt to miss a function containing many branches due to short BHR. Therefore, a long global history register was folded in half in order not to lose a information about long branch history. As a result, the proposed architecture could use only the same size as the PHT index bits used in gshare predictor. To predict branch outcomes more accurately with correlations of branch instructions, the proposed architecture used both a folding of the global history register in half and the exclusive-OR arithmetic with 0, 4, and 12 shifted program counter values. Consequently, exclusive-OR logics with modified global history and a variety of shifted PC values produced patterns that were more accurate to index the PHT in order to decrease the aliasing rate.

Ⅲ. Measurement Results

To measure the proposed HDBP, SimpleScalar with SPEC2000 benchmarks were used. Aliasing rate, performance improvement, and IPC were compared among the HDBP, DpBHLA, and gshare. The aliasing rate is the total number of aliasing occurrences over the PHT entry size from 1,024 to 8,192. The Performance is the branch direction-prediction rate. In other words, it connotes the accuracy of the branch prediction.

Fig. 2 depicts the aliasing rate with gshare and DpBHLA, and Fig. 3 shows the improvement rate of aliasing occurrences when varying PHT size. Aliasing occurred least in the HDBP, regardless of the size of the PHT entries. That is, on average, the aliasing rate of the proposed structure decreased to 44.1% with gshare, and to 34.9% with DpBHLA. Fig. 4 shows the performance improvement of the HDBP over gshare, with each benchmark according to each PHT size. Because the HDBP had the smallest number of aliasing occurrences, regardless of the number of PHT entries, it predicted the branch outcome more accurately than did other branch predictors that are used with most programs. On average, when the number of the PHT entries were 1K, 2K, 4K and 8K, performance improvement increased by 20.62%, 18.80%, 20.46% and 16.37%, respectively. In other words, irrespective of the size of the PHT entries, the performance improvement increased to 19.06%, on average. Fig. 5 shows the IPC, which is the standard for a processor’s performance with each PHT entry size. On average, the HDBP could achieve 1.390 IPC compared to 1.370 with gshare and 1.385 with DpBHLA.

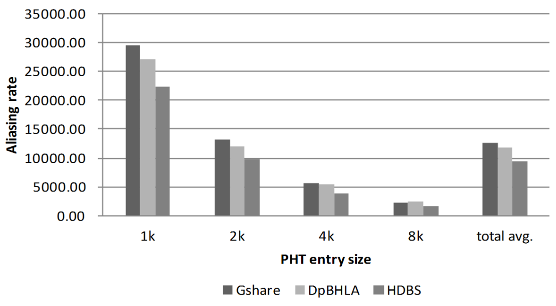

Fig. 2 Aliasing rate with each PHT entry size

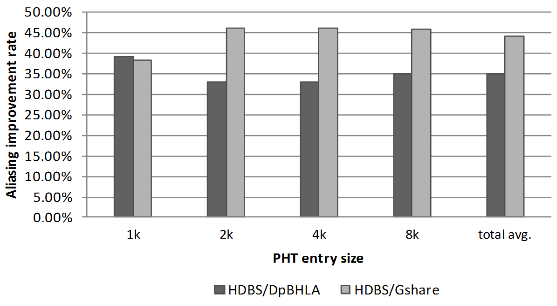

Fig. 3 Aliasing improvement with each PHT entry size

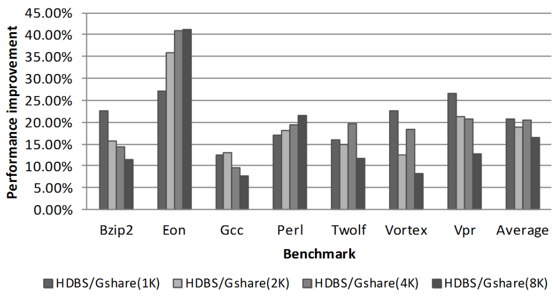

Fig. 4 Performance improvement with benchmarks

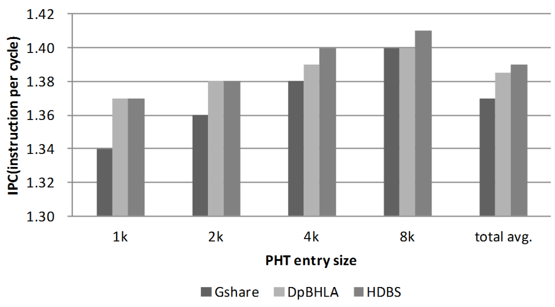

Fig. 5 IPC with each PHT entry size

Ⅳ. Conclusion

As recent microprocessors request high performance using deeper pipelining and issuing multiple instructions per cycle, accurate branch predictors have become an import part of modern processor architecture. The proposed HDBP is a completely dynamic per-branch method with a static prediction scheme. To make more identified patterns and prevent destructive aliasing, dynamically new BHR was made, a variety of stage PC values were accumulated, and a folded global history register was used in the predictor.

The simulation results showed that when the PHT entries were 1K, 2K, 4K, and 8K, the aliasing rate decreased by 38.3%, 46.0%, 46.1%, and 45.8%, respectively, compared with gshare. It also decreased by 34.9% with DpBHLA. In addition, when the proposed HDBP is used, the aliasing occurrence rate is the best in 4K PHT entries. Because of the sharp reduction in aliasing occurrences, branch performance improved by 20.46% in 4K entries. IPC also increased by 1.28% on average, regardless of the number of PHT entries. In conclusion, the HDBP is an efficient branch predictor because less aliasing occurs than in DpBHLA and gshare. Furthermore, the HDBP greatly decreases branch prediction miss rate and increases the IPC.

REFERENCES

[1] A. Mondelli, “Revisiting wide superscalar microarchitecture,” Ph. D. dissertation, University of Rennes 1, Rennes, France, 2017.

[2] Y. Hou, H. He, X. Yang, D. Guo, X. Wang, J. Fu, and K. Qiu. (2016, October). FuMicro: A fused microarchitecture design integrating in-order superscalar and VLIW. *VLSI Design* [Internet] Available: http://dx.doi.org/10.1155/2016/8787919.

[3] M. Alipour, T. E. Carlson, D. Black-Schaffer, and S. Kaxiras, “Maximizing limited resources: a limit-based study and taxonomy of out-of-order commit,” *Journal of Signal Processing Systems*, vol. 91, pp. 379-397, Apr. 2019.

[4] T.-Y. Yeh, and Y. N. Patt, “Alternative implementations of two-level adaptive branch prediction,” in *Proceedings of the 19th International Symposium on Computer Architecture*, Queensland: Australia, pp. 124-134, 1992.

[5] S. McFarling, “Combining branch predictors,” Western Research Laboratory, Palo Alto: CA, Technical Report TN-36, 1993.

[6] R. Thomas, M. Franklin, C. Wilkerson, and J. Stark, “Improving branch prediction by dynamic dataflow-based identification of correlated branches from a large global history,” in *Proceedings of the 30th International Symposium on Computer Architecture*, San Diego: CA, pp. 314-323, 2003.

[7] J. W. Kwak, and C. S. Jhon, “Dynamic Per-Branch History Length Adjustment to Improve Branch Prediction Accuracy,” *Microprocessors and Microsystems*, vol. 31, pp. 63-76, Feb. 2007.

[8] S. Mittal, A survey of techniques for dynamic branch prediction. (2018, September). *Concurrency and Computation* [Internet]. Available: https://doi.org/10.1002/ cpe.4666.