2021 IEEE 4th International Conference on Electronics Technology

Design of a High Performance Branch Predictor Based on Global History Considering Hardware Cost

2021 IEEE 4th International Conference on Electronics Technology (ICET) | 978-1-7281-7673-4/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICET51757.2021.9451111

Tao Jiang, Ning Wu, Fang Zhou, Liangkai Zhao, Fen Ge, and Jin Wen
College of Electronic and Information Engineering
Nanjing University of Aeronautics and Astronautics
Nanjing, China
jiangtao08@nuaa.edu.cn

Abstract—**Branch prediction is an indispensable technology in modern processor design and a key method to improve processor performance. When the CPU is executing programs, branch instructions account for 20% to 30% of all instructions, which affect the processor performance. Although dynamic branch predictor based on global history has a simple structure and occupies less memory, the existence of index aliasing problem influences branch prediction accuracy. In this paper, the traditional Gselect branch predictor was improved by the XOR operation between the index generated by Gselect branch predictor and the last 8 bits of instruction's PC address to ease the problem of index aliasing. Finally, the simulation was carried out by using SimpleScalar 3.0 simulator and benchmark programs. The experimental results show that when the length of GHR register is 12, the prediction rate of the newly improved branch predictor called XOR-Gselect is best and 1.93% higher than Gselect's on average.**

Keywords—branch predictor, global history, index aliasing, branch prediction rate

I. Introduction

Branch predictor is a momentous portion of modern processor to improve instruction execution efficiency and reduce power consumption by cutting down instructions' execution number on the erroneous path [1, 2, 3, 4]. Taking classical five-stage pipelined processor based on MIPS architecture as an example. If predicting that all branch instructions will not be executed in the instruction's fetching phase, all the instructions before the execution phase in the pipeline will be cleared from the pipeline and will be retrieved from the correct address again after finding an instruction that can be executed in instruction's execution phrase. This operation not only wastes the power consumption of the processor, but also reduces processor's instruction execution efficiency [5].

Branch prediction technology is divided into static prediction [6, 7] and dynamic prediction [8, 9]. Earliest static branch predictor's prediction rate is only about 50% [5], which is not acceptable. Dynamic branch predictor has experienced the development process of 1-bit branch predictor [2], 2-bit branch predictor [10, 11, 12], one-level branch predictor [5], two-level branch predictor [13,14], hybrid branch predictor [15, 16, 17] and neural network predictor [18, 19, 20].

Dynamic branch predictor based on global history register (GHR) uses the shifted branch history register to record the execution of the latest n instructions in the first level, and uses pattern history table (PHT) with four state values to predict the jump direction of branch instructions in the second level.

After the emergence of two-level branch predictor, the branch prediction accuracy is greatly improved. However, the existence of index aliasing affects branch prediction rate [21, 22]. In [23], structures of Gselect and Gshare are proposed, whose branch direction prediction rate is higher than traditional global branch predictors'. Because of the limit of GHR width, different branch instructions may have the same GHR parts, which corresponds to the same PHT saturation counter. In this way, the results of branch instructions will interfere with each other, thus reducing the prediction accuracy of branch predictor.

In the design of modern processor, high performance and low power consumption are two quite important factors needed to consider. Therefore, a branch predictor giving consideration to branch prediction rate and hardware cost both is the most needed and practical in processor design. Owing to the low prediction accuracy of static predictors, 1-bit predictors and one-level predictors, the high hardware cost of hybrid predictors and the shortcomings of neural branch predictors' high delay and long training time, the paper intends to do a research on dynamic branch predictors based on global history that have the virtues of very considerable branch prediction accuracy and low memory occupation.

As Gshare's method handles the problem of index aliasing better than Gselect's in usual [22, 23], this paper aims to improve the structure of traditional Gselect branch predictor to alleviate the effect of index aliasing on branch prediction accuracy. Finally, the simulation is carried out by using SimpleScalar 3.0 simulator and benchmark programs. The experimental results show that direction prediction rate of the improved branch predictor is 1.93% higher than that of the traditional Gselect branch predictor on average under the same GHR register length.

The rest of this paper is organized as follows: Section II introduces some disadvantages existing in classical dynamic branch predictors. In section III, improved branch predictor structure are proposed. Section IV introduces the experimental environment and experimental results. Finally, conclusions are given in Section V.

II. Shortages of Several Classical Dynamic Branch Predictors

A. Branch Predictor Based on 1-bit Saturation Counter

Structure of 1-bit branch predictor is shown in Fig. 1. As can be seen from Fig. 1, if the final execution result of branch instruction is jump, that is, the value in 1-bit saturation counter is 1, then the execution result of branch instruction is also predicted as jump. If the prediction is correct, the value of the saturation counter does not change. Otherwise, the value in the saturation counter needs to be updated.

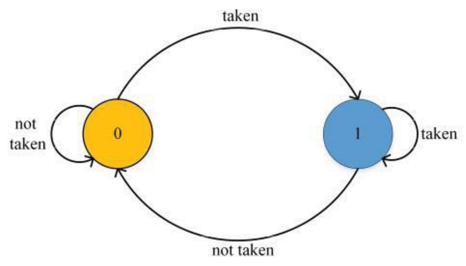

Fig. 1. 1-bit branch prediction

Compared with static branch predictor, 1-bit branch predictor has better prediction accuracy in many cases. However, if the jump direction of branch instruction changes every time, the accuracy of branch prediction based on branch instruction's last n execution results will be 0 [5]. The situation of branch instructions with frequent direction changes is shown in Fig. 2. In actual, the prediction method based on 1-bit saturation counter is never adopted in modern processor design.

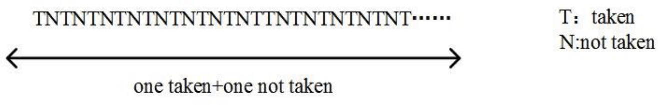

Fig. 2. The situation of branch instructions with frequent direction changes

B. Branch Predictor Based on 2-bit Saturation Counter

2-bit branch predictior is to predict the direction of this time according to the results of the first two execution of a branch instruction. This method is represented by a state machine with four states. When the state machine code is 11, the counter is in saturated state, which means that the branch instruction will be predicted to jump this time. When the state machine code is 10, the counter is in unsaturated state, which means that the branch instruction will be predicted as jump this time. The other two states are similar. The structure of 2-bit branch predictor is shown in Fig. 3.

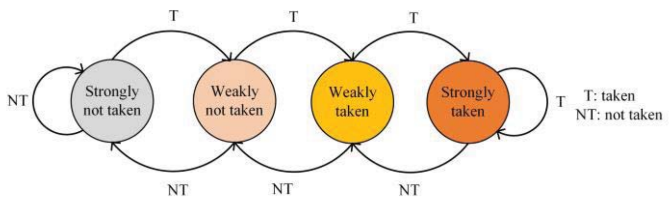

Fig. 3. A branch prediction method based on 2-bit saturation counter

When the direction of branch instruction is always in one direction, the state machine will be in saturation state. Under this circumstance, the prediction accuracy is high, and the prediction result will be changed only when the prediction fails twice in succession. But when the direction of branch instruction always changes, the state machine can not be in saturated state, which means the relatively low prediction accuracy of branch predictor. Therefore, prediction results got by 2-bit branch prediction method are still not very reliable.

C. One-Level Branch Predictor

Considering the actual memory capacity, one-level branch predictor uses part of PC value to address PHT. This method is shown in Fig. 4.

There are two problems in the one-level branch predictor. The first problem is index aliasing, and the other is that one-level branch predictor's prediction rate is very low in some cases. When the execution of branch instruction is shown in Figure 2 and the state of state machine is weakly not taken, the accuracy of branch prediction is 50% [5], which is also unacceptable.

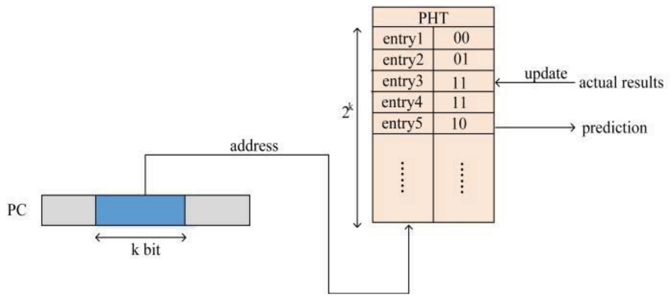

Fig. 4. One-level branch predictor

D. Two-level branch predictor

The Bimode branch predictor that is one of two-level predictors has high branch prediction rate and will be used in the following section to compare with newly proposed predictor. In case, it is necessary to introduce the structure of Bimode. The structure of Bimode branch predictor is shown in Fig. 5.

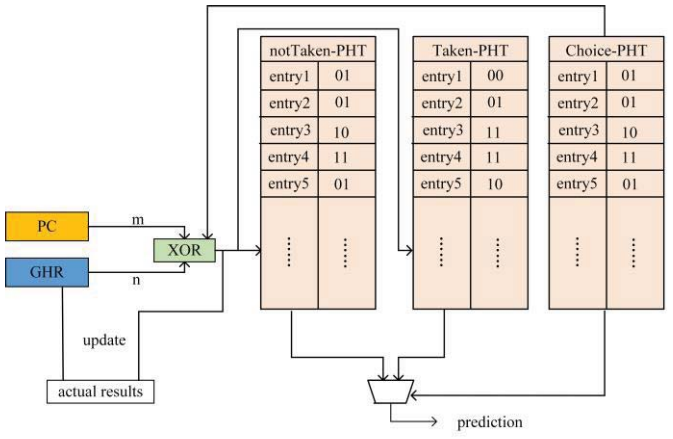

Fig. 5. The structure of Bimode branch predictor

The Bimode branch predictor has three PHT tables, including the taken PHT, the not-taken PHT and the choice PHT, which are composed of 2-bit saturation counters. When predicting that the branch direction is jump, if the result obtained from the history table is 00 or 01, the results of not-taken PHT will be adopted. Otherwise, results of taken PHT will be used. The Bimode branch predictor's GHR values and PHT table values are updated in a similar way to Gshare and Gselect, but only GHR and PHT table which is selected in the choice PHT are updated.

Though Bimode branch predictor has high branch direction prediction rate, hardware consumption has to be considered due to its three PHT tables.

E. *Hybird and Neural Network Predictor*

Hybird branch predictor is of great prediction accuracy, but hardware cost is a considerable problem because it combines several prediction methods. Besides, neural network predictor has the malpractices of high delay and long training time [19].

III. New Scheme Proposal

The Gselect branch predictor uses the high bits of PC value and the low bits of GHR as the index of PHT, but does not use the low bits of PC value fully. Although using fewer global history bits and more PC bits can make Gselect's branch prediction precision more higher [23], the role of PC values can not be brought into full play. As it is hard to accurately discriminate the lost parts of instructions, the decrease of Gselect branch prediction rate is caused eventually [21].

In order to make full use of PC values of branch instructions, XOR operation was used between the index generated by Gselect branch predictor and low 8 bits of PC value to get the optimized index in this paper. For the structural characteristics, we call the branch predictor of the new scheme as XOR-Gselect branch predictor. The structure of XOR-Gselect branch predictor is shown in Fig. 6.

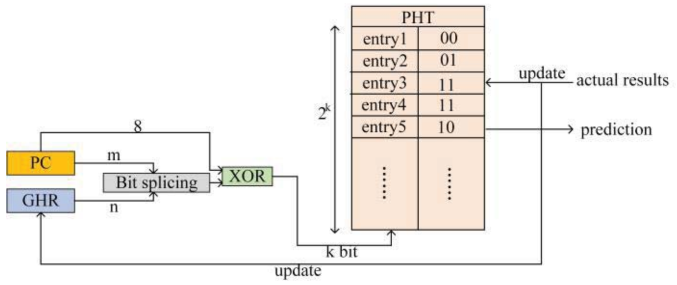

Fig. 6. The structure of XOR-Gselect branch predictor

Table I shows an example of the index aliasing of Gselect branch predictor. It is assumed that width of PC and GHR is 16. Gselect splices the high 8 bits of PC value with low 8 bits of GHR to get the new index. XOR-Gselect uses the XOR operation between PC's low 8 bits and index generated by Gselect.

TABLE I. Index Generation of Gselect and XOR-Gselect Branch Predictor

Index Name	Value	Gselect	XOR-Gselect
Case1: PC	1010001111001111	1010001110101010	1010001101100101
Case1: GHR	0011110010101010		
Case2: PC	1010001100000001	1010001110101010	1010001110101011
Case2: GHR	1010111110101010		

As can be seen from Table I, for two cases without same PC value and GHR value, index aliasing will only occur when using Gselect branch predictor. In this case, we can not prove that the newly proposed branch predictor has better property. Therefore, it is necessary to use the SimpleScalar 3.0 [3, 24, 25, 26] to execute hundreds of thousands of instructions to further judge the performance of XOR-Gselect branch predictor.

IV. Experimental Environment and Results

A. *Experimental Environment*

To evaluate the performance of the proposed branch predictor, a simulation platform is established based on SimpleScalar 3.0 simulator. SimpleScalar 3.0 is a superscalar 5-level pipelined RISC architecture simulator. It simulates pipelining, cache, disorder execution and branch prediction in processor architecture. Besides, SimpleScalar 3.0 is a well-known simulator and is usually used for program performance analysis, detailed microarchitecture modeling and hardware and software collaboration verification.

This paper uses SimpleScalar 3.0 sim-bpred simulator and benchmark programs for simulation and uses Gselect, Bimode [27, 28] and XOR-Gselect predictors to be research objects. Besides, we need to compile toolchain based on RISC-V well that is used to compile benchmark programs, including anagram, test-fmath, test-long, test-lswlr, test-math and test-printf [29].

B. *Experimental Results*

1) *The impact of GHR length on branch direction prediction rate*

With the increasing of GHR's length, more history information of branch instructions can be recorded, which is more conducive to improve the prediction rate of branch predictor. In order to achieve the highest branch prediction rate, we need to study the factors that affect the branch prediction rate. These factors are mainly divided into the length of GHR register and the size of PHT table for two-level branch predictors. As the XOR-Gselect branch predictor is improved from the structure of Gselect branch predictor, and as mentioned above, the GHR register length of Gselect branch predictor is exponentially related to the size of PHT table. we only need to consider the impact of GHR length on XOR-Gselect predictor.

The performance comparison of branch predictors with different test sets under different GHR register length is shown in Fig. 7. As can be seen from Figure 7, when the length of GHR register increases to 12, the branch direction prediction rate of six test sets reaches saturation. For this reason, GHR length is set as 12.

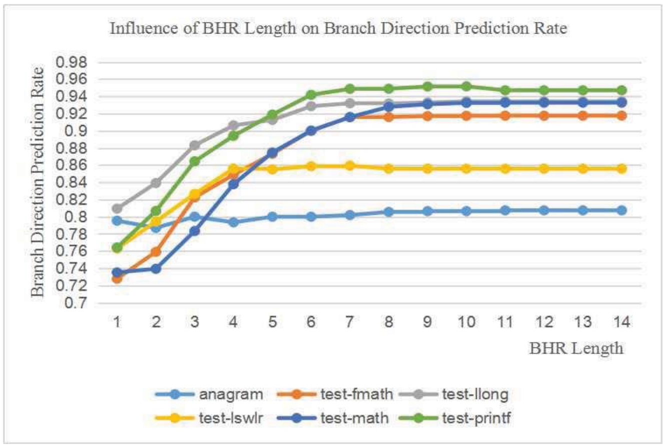

Fig. 7. The effect of GHR register's length on branch prediction

2) The effect of number of PC value's low bits on branch direction prediction rate

To prove that the low bits number of PC value used in this paper has the greatest impact on the prediction rate of branch predictor, the experiment controls the size of PHT table is 4096 and the length of GHR is 12, and gradually increases the number of PC value's low bits used to generate the index to PHT. The prediction results can be seen in Fig. 8.

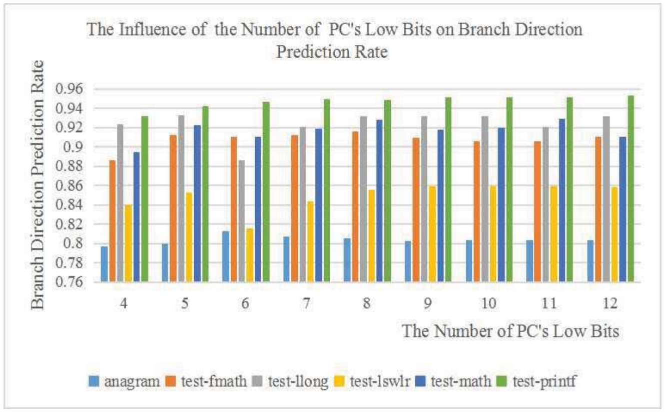

Fig. 8. The effect of number of PC's low bits length on branch prediction

After calculation, the average branch prediction rate of XOR-Gselect is obtained by using different number of PC's low bits based on six test sets, which is shown in Fig. 9.

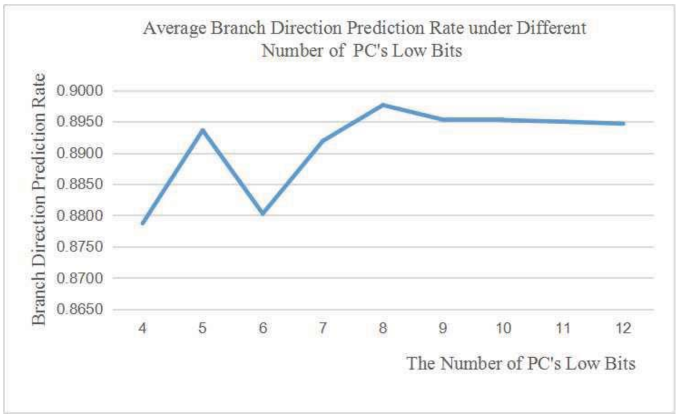

Fig. 9. The effect of GHR register's length on branch prediction

3) Comparasion of three predictors' performance

According to the above experimental results, GHR register length is selected as 12. The branch direction rate of Bimode, Gselect and XOR-Gselect is tested and analyzed. Based on six compiled test sets, the branch direction prediction rate of three branch predictors is shown in Fig. 10.

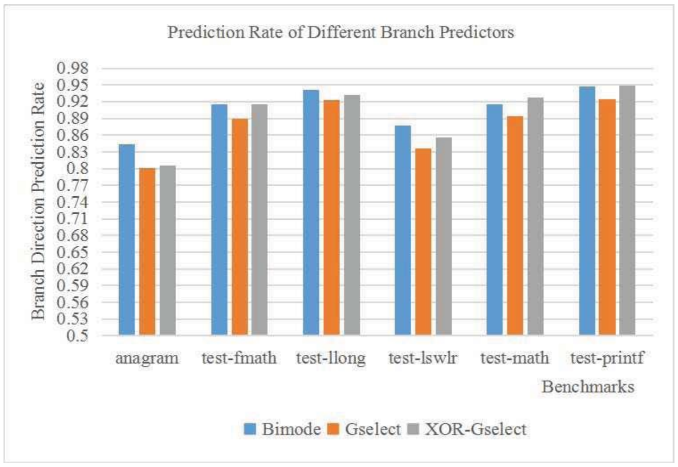

Fig. 10. Branch direction prediction rate of different branch predictors

It can be seen from the Figure 10 that the prediction rate of XOR-Gselect branch predictor is higher than that of traditional Gselect branch predictor. When using test-printf test set and GHR length is 12, XOR-Gselect's prediction rate is highest and can be up to 94.89%. In the same condition, Gselect's prediction rate is only 92.53%.

After calculation, the prediction accuracy of XOR-Gselect branch predictor is 1.93% higher than that of Gselect branch predictor on average based on six test sets. However, the average prediction rate of these two branch predictors is still lower than that of Bimode branch predictor. Gselect's branch prediction rate is 2.89% lower than Bimode's and XOR-Gselect's is 0.96% lower than that of Bimode.

When focusing on branch predictors' direction prediction accuracy, hardware cost is also needed to consider. In [30], hardware cost estimation equation of branch predictors is given. Hardware cost of three branch predictors base on global history can be calculated by Equation (1).

$$(g+1)\times p\times 8 \qquad (1)$$

In the formula, g represents the length of GHR and p represents the number of entries in the PHT. When the GHR's width of three branch predictors is same, according to (1), the hardware cost of Bimode branch predictors is three times than that of the other two because Bimode predictor uses three PHT tables.

V. Conclusion

Branch predictor is one of the important ways to improve the performance of processor. In this paper, XOR-Gselect branch predictor further alleviates the influence of index aliasing. Based on six benchmark test sets, experimental results show that the branch prediction rate of XOR-Gselect is 1.93% higher than that of traditional Gselect branch predictor on average when GHR length is 12. At the same time, XOR-Gselect's prediction rate is 94.89% and reaches the highest when using one benchmark called test-printf. Although the prediction accuracy of XOR-Gselect branch predictor is still not higher than that of Bimode branch predictor, the hardware cost of Bimode branch predictor is three times than that of XOR-Gselect and Gselect branch predictors due to its three pattern history tables. When considering the hardware cost and prediction accuracy comprehensively, XOR-Gselect branch predictor is a good choice.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (No.61774086), and the Fundamental Research Funds for Central Universities (No. NP2019102).

REFERENCES

[1] S. Mittal, "A survey of techniques for dynamic branch prediction," Concurr. Comput.-Pract. Exp, vol. 31, January 2019.

[2] J. E. Smith: "A study of branch prediction strategies," IEEE Int. Symp. Comput. Archit, 1998.

[3] Choi, et al., "NTB branch predictor: dynamic branch predictor for high-performance embedded processors," J. Supercomput, vol. 72, pp. 1679-1693, May 2016.

[4] Evers, et al., "Understanding branches and designing branch predictors for high-performance microprocessors," Proc. IEEE, vol. 89, pp. 1610-1620, November 2001.

[5] Y. B. Yao, Design of Superscalar Processor, Tsinghua University Press, China, 2014.

[6] D. R. Kaeli and P. G. Emma, "Branch history table prediction of moving target branches due to subroutine returns," IEEE Int. Symp. Comput. Archit, 1991, pp. 34-42.

[7] Fisher, et al., "Predicting conditional branch directions from previous runs of a program," ACM Sigpl. Notices, vol. 27, pp. 85-95, September 1992.

[8] H. Patil and J. Emer, "Combining static and dynamic branch prediction to reduce destructive aliasing," IEEE Int. Symp. High-Perform. Comput. Archit, 2000, pp.251-262.

[9] P. Wang and F. Q. Si, "Dynamic Prediction of the Thermal Nonlinear Process Based on Deep Hybrid Neural Network," E3S Web Conf, vol.
162, pp. 01007, 2020.

[10] R. Nair, "Optimal 2-bit branch predictors," IEEE Trans. Comput, vol. 44 pp. 698-702, May 1995.

[11] J. K. F. LEE and A. J. Smith, "Branch Prediction Strategies and Branch Target Buffer Design," Computer, vol. 17, pp.6-22, 1984.

[12] Elkhouly, et al., "2-Bit Branch Predictor Modeling Using Markov Model," Proc. Comput. Science, vol. 62, pp.467-469, 2015.

[13] H. K. Kim, et al., "A high-performance branch predictor design considering memory capacity limitations," Int. Conf. Circuits, Syst. Simul, 2017, pp. 49-53.

[14] T. Y. Yeh, "Two-level adaptive training branch prediction," Proc. 24th Ann. Symp. Worksh. Micr, 1991, pp. 51-61.

[15] P. Trivedi and S. Shah, "Reduced-hardware Hybrid Branch Predictor Design, Simulation & Analysis," 7th Intern. Conf. Smart Computing & Comm, 2019, pp. 6.

[16] M. Das, et al., "Shared Pattern History Tables in Multicomponent Branch Predictors With a Dealiasing Cache," IEEE Embed. Syst. Lett, vol. 12, pp. 95-98, September 2020.

[17] R. E. Kessler, "The Alpha 21264 microprocessor," IEEE Micro, vol. 19, pp.24-36, 1999.

[18] Hida, Itaru , et al., "An energy-efficient dynamic branch predictor with a two-clock-cycle naive Bayes classifier for pipelined RISC microprocessors," IEICE Nonlin. Theory Applic. Ice, vol. 8, 2017.

[19] L. Zhang, et al., "A Dynamic Branch Predictor Based on Parallel Structure of SRNN," IEEE Access, vol. 8, pp. 86230-86237, 2020.

[20] Y. H. Mao, et al., "Exploring Convolution Neural Network for Branch Prediction." IEEE Access, vol. 8, pp. 152008-152016, 2020.

[21] H. K. Kim, et al., "A high-performance branch predictor design considering memory capacity limitations," Int. Conf. Circuits, Syst. Sim, pp. 49-53, 2017.

[22] J. W. Park, et al., "A Branch Predictor Design to Improve Prediction Rate by Reducing Index Aliasing in Application Processors," 12th Int. Conf. Inform. Comm. Techn. Syst, 2019, pp. 193-196.

[23] S. McFarling, "Combining branch predictors," Digital Western Research Laboratory, Tech. Rep, 1993.

[24] Burger, Doug, and T. M. Austin, "The SimpleScalar tool set, version 2.0," Acm Sig. Comp. Arch. News, vol. 25, pp. 13-25, June 1997.

[25] Austin T, et al., "SimpleScalar: an infrastructure for computer system modeling," Computer, vol. 35, pp.59, February 2002.

[26] https://www.simplescalar.com.

[27] K. Kise, et al., "The bimode++ branch predictor," Inn. Arch. Future Gen. High-Performance Proc. Syst, 2005.

[28] C. C. Lee, et al., "The bi-mode branch predictor," Proc. 30th Ann. Int. Symp. Micr, 1997.

[29] B. Das, et al., "Impact of Inaccurate Design of Branch Predictors on Processors' Power Consumption," IEEE Ninth International Conference on Dependable IEEE, 2011, pp. 335-342.

[30] M. S. M. Haque, et al., "Enhancing Branch Predictors using Genetic Algorithm." The 8th Int. Conf. Mod. Sim. Appl. Optim, 2019, pp. 5.