International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 1, February 2021, pp. 265~271
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i1.pp265-271

FPGA configuration of an alloyed correlated branch predictor used with RISC processor for educational purposes

Hadeel SH. Mahmood
Department of Computer Engineering Techniques, Middle Technical University, Iraq

Article Info

Article history:

Received Feb 14, 2020
Revised Jul 10, 2020
Accepted Jul 28, 2020

Keywords:

Alloyed predictor
Correlated predictor
Dynamic branch prediction
FPGA
MIPS
Nested branches
RISC processor
VHDL

ABSTRACT

Instructions pipelining is one of the most outstanding techniques used in improving processor speed; nonetheless, these pipelined stages are constantly facing stalls that caused by nested conditional branches. During the execution of nested conditional branches, the behavior of the running branch depends on the history information of the previous ones; therefore, these branches have the greatest effect in reducing the prediction accuracy of a branch predictor among conditional branches. The purpose of this research is to reduce the stall cycles caused by correlated branches misprediction by introducing a hardware model of a branch predictor that combines both local and global prediction techniques. This predictor integrates the prediction characteristics of the alloyed predictor with those of the correlated predictor. the predictor design which implemented in VHDL (Very high-speed IC hardware description language) was inserted in previously designed MIPS (microprocessor without interlocked pipelined stages) processor and its prediction accuracy was confirmed by executing a program using the selection sort algorithm to sort 100 input numbers of different combinations ascendingly.

Corresponding Author:

Hadeel SH. Mahmood,
Department of Computer Engineering Techniques,
Middle Technical University,
Masafi Rd., Aldora, Baghdad, Iraq.
Email: hadeel.shakir@mtu.edu.iq

1. INTRODUCTION

The direction of a conditional branch depends on the outcome of its condition test which be either true or false. When it is true, the branch is taken and the instruction at the branch target address is fetched. Otherwise, the branch is untaken and the next instruction in ordinary sequence is fetched [1]. This repetitive pattern of a conditional branch has been applied in dynamic branch prediction to keep a local history for each conditional branch from which its future direction can be predicted [2].

In the case of nested conditional branches, the conditions may be related to each other, so the outcome of the running condition test will also depend on that of the previous conditions which in turn will cause the direction of current branch to be affected by the directions of preceding branches [3]. Consequently, each prediction is based on the histories of past branches rather than just its individual history. This global history besides the local history should be recorded by an effective branch prediction unit and exploited in revealing correlation among branches, or high-performance processors will continue to experience pipeline stalls due to correlated branches [4, 5]. For instance, Pentium Pro uses the result from the last two branches to select one of the four sets of branch history table (BHT) bits [6], while VIA Nano processor uses a prediction technique that amalgamates the local and global principles of prediction [7].

According to its parallelism which is also considered as one of the major factors that affect processor performance, field programmable gate array (FPGA) is recently adopted by many researchers for the branch prediction implementation [8-13]. Where in [8], an accurate and fast branch predictor that uses few resources on FPGA was implemented for general purpose, pipelined, single core soft processor, whereas [9] implemented a configurable VHDL model of a branch predictor unit which is composed of a branch direction predictor and a branch target buffer. A new TAGE branch predictor was proposed by [12], it can avoid being the critical path of the processor while keeping the same prediction accuracy as TAGE.

Later sections of this research are ordered as follows; section 2 expresses a theoretical review of how some dynamic branch predictors utilize the global history of previous branches in predicting current branch direction. Then the proposed VHDL design of the alloyed-correlated branch predictor and how it is implemented in FPGA is described at section 3. Section 4 reveals how the required results which confirmed the efficiency of the design are obtained. Finally, the conclusion is demonstrated in section 5.

2. THEORETICAL FEATURES OF CORRELATED PREDICTION

In order to reduce the branch misprediction penalty raised due to branch correlating, two levels of history should be used to guess the branch direction. The first level holds the patterns of former branches, while the second level catches the branch directions associated with the last occurrence of a particular pattern in the first level [14, 15]. Here is a summarized illustration of how the feature of two history levels is employed in designing some types of branch prediction schemes.

2.1. Correlated branch predictor

This technique takes benefit of the correlation between conditional branches therefore it has improved branch prediction accuracy remarkably. Branch prediction process in correlated predictor passed through two levels of history as Figure 1 shows [16]. The first level which is an *m*-bit global history register (GHR) records the behavior of the last *m* branches. As for the second level, the pattern history table (PHT) holds the last direction taken by each branch when a certain pattern in the first level occurs [17-19]. The lower *n* bits of the branch address are used to select one of the 2^n entries of the PHT, where each entry consists of 2^m sets. one of these sets is chosen by the GHR. Each set holds a 2-bit saturating counter with saturation levels ranged from (00) as the minimum value to (11) as the maximum value, and the predicted path is specified by the MSB (Most Significant Bit) of the counter [1, 20].

Because this predictor only records correlation between *m*+1consecutive branches, the GHR may not catch any bit belonging to the same branch especially if there are many irrelevant branches in between. This may lead to branch misprediction which in turn will increase the branch penalty [21].

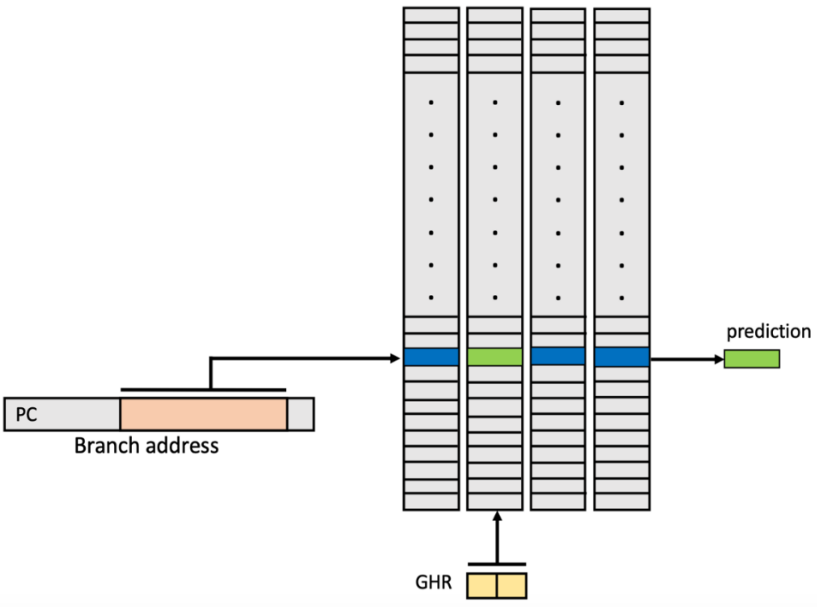

Figure 1. Design of the correlated branch predictor

2.2. Alloyed branch predictor

This dynamic scheme which also consists of two levels of history has been suggested to address the issue associated with the correlated predictor to some extent. Where a single *k*-bit GHR, a shift register, is used to record outcomes of the most recently executed *k* branches. This register which represents the first level is then concatenated with the lower *n* bits of the branch address to produce an *m*-bit index which is used to pick one of the 2^m counters of the second level performed by the pattern history table (PHT) as in Figure 2 [22-24]. The main drawback of the alloyed predictor lies in the number of history bits in the PHT index which might be insufficient to include all previous branches correlated with the branch being executed [25].

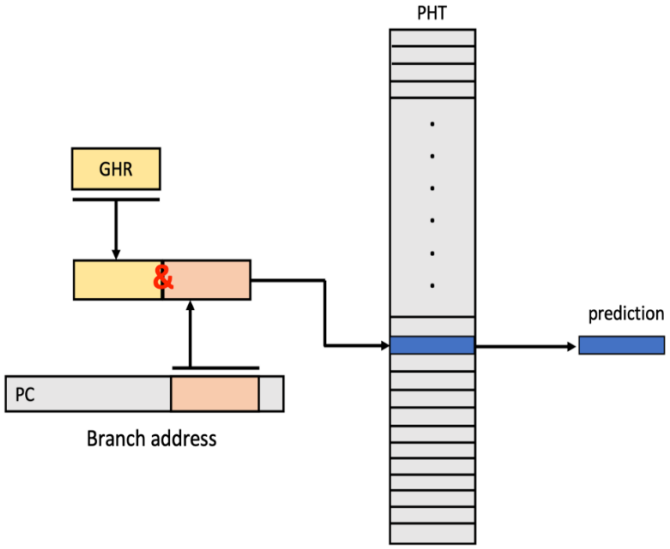

Figure 2. Design of the alloyed branch predictor

3. ALLOYED-CORRELATED BRANCH PREDICTOR IN VHDL

This research aims to increase the speedup of the processor by designing a dynamic branch predictor with higher accuracy and lower branch penalty. This predictor which is modelled in VHDL using Xilinx ISE (integrated synthesis environment) Design Suite 14.7 merges the prediction attitudes of both the alloyed predictor and the correlated predictor.

The proposed design of the alloyed-correlated branch predictor shown in Figure 3 is placed within a predesigned 32-bit MIPS processor that is taught as a main subject in the microprocessor architecture course, and its predictive action depends on two levels of history:

- The first level is a 7-bit GHR that records the actual direction of the most recent seven branches.
- The second level is a PHT with 1024 entries. Each entry has 4 sets, and each set contains a 2-bit saturating counter that is initialized to a default value of "11".

During the fetch stage, the seltF (9:0) address is created, as in the alloyed predictor, by concatenating the GHR upper five bits (GHRF (6:2)) with the lower five bits (pcF (6:2)) of the Program Counter (PC) which holds the branch address. Subsequently, the seltF is used to select one of the 1024 entries of the PHT. Here the proposed predictor acts as a correlated predictor and uses the two lower bits GHRF (1:0) of the GHR to choose only one set from the four sets of the indexed entry according to Table 1. The upper bit (predictF (1)) of the selected set content (predict (1:0)) provides the predicted direction as taken or not taken whenever it is '1' or '0' respectively.

Later in the decode stage as shown in Figure 3, whenever the branch is really taken (pcsrcD=1), the corresponding saturating counter is incremented until the maximum value (11) is reached, otherwise it is decremented until the minimum value (00) is reached. Every saturating counter is modelled as a mealy finite state machine (FSM) in which the predfsmop (1:0) output depends on the pcsrcD input besides the current state (predictD (1:0)) as in Figure 4. The output (predfsmop (1:0)) and the next state are generated at the next state logic which is a combinational logic, whereas the current state is held by the current state register which is modeled as a D flip-flop.

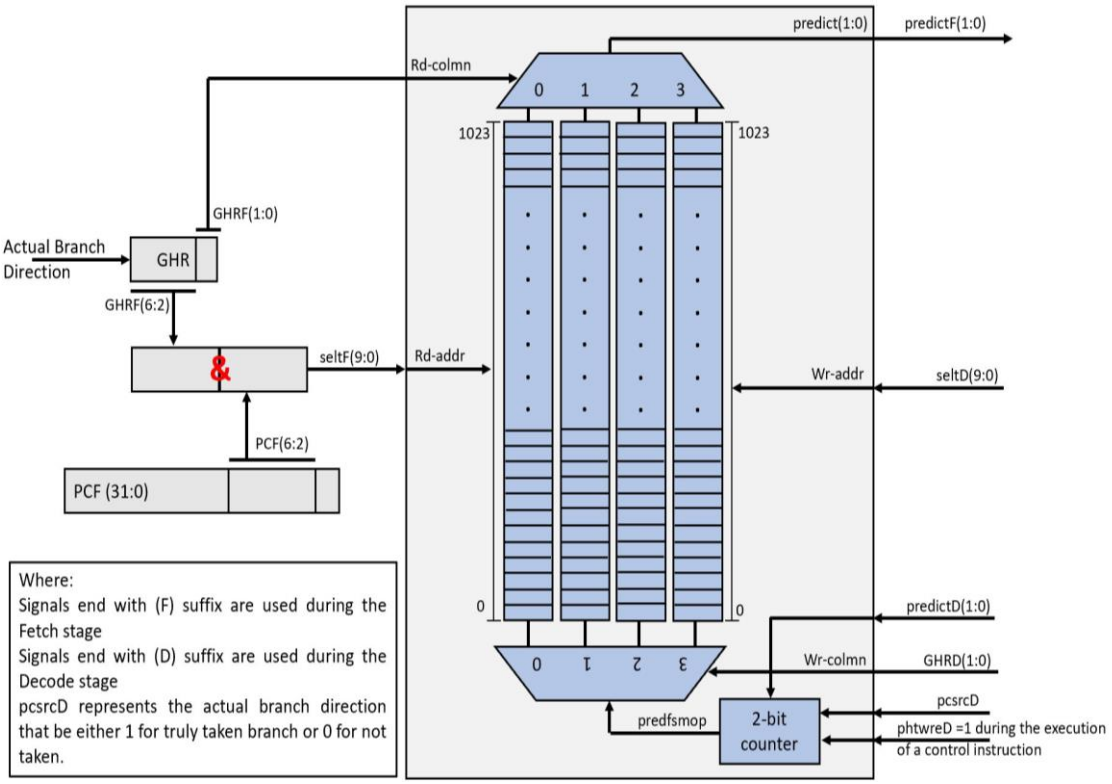

Figure 3. Design of the alloyed-correlated branch predictor

Table 1. Set selection

GHRF(1:0)	Selected set
00	Set0
01	Set1
10	Set2
11	Set3

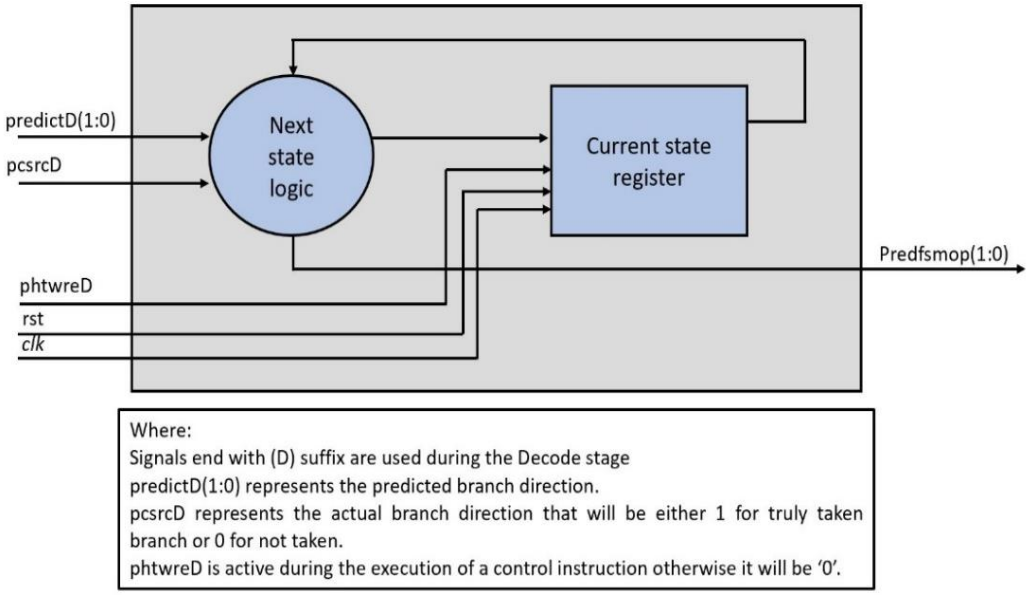

Figure 4. The block diagram of 2-bit saturating counter

After completing the VHDL implementation of the alloyed-correlated branch predictor module, a schematic representation of its components has been created using the RTL and Technology Schematic viewer of Xilinx ISE Design Suite 14.7. As illustrated in Figure 5, this schematic represents the branch predictor design by generic symbols to each of the PHT, the GHR and the 2-bit saturating counter.

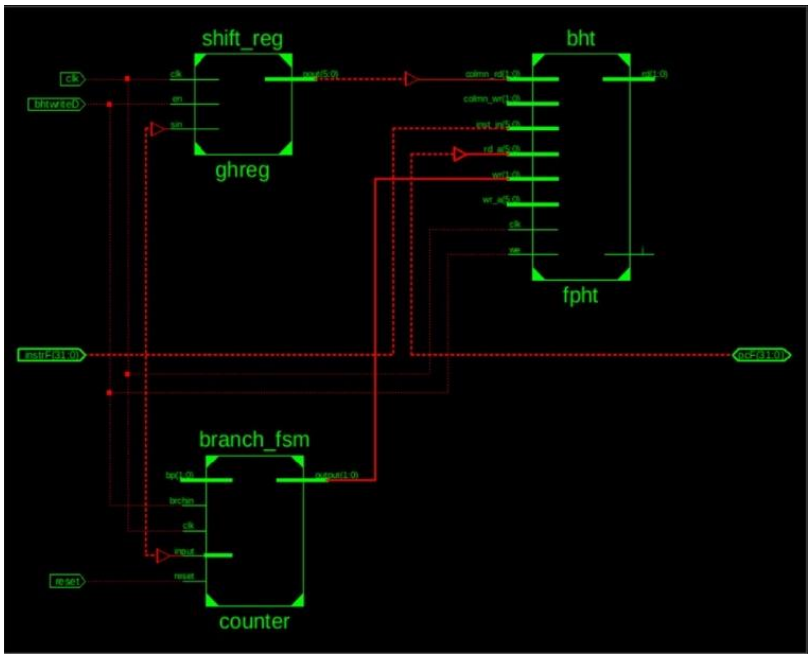

Figure 5. Schematic module of the alloyed-correlated branch predictor

4. RESULTS AND DISCUSSIONS

The alloyed-correlated branch predictor gathers the design aspects of both alloyed and correlated predictors, and to prove effectiveness of the designed predictor, its performance was compared with that of each of alloyed and correlated predictors in terms of program execution time, predictor performance, clock per instruction (CPI) and processor speedup. In this section, the selection sort algorithm that consists of conditional branches as 20% out of its all instructions was used to sort 100 integer numbers of three different distributions: sorted, uniform and normal. Since this algorithm is in-place comparison algorithm, using different types of input data would make the correlation between branch conditions uneven, this made it possible to demonstrate the impact of the proposed design. It can be clearly seen from Table 2 that the alloyed-correlated branch predictor achieved the best results by having the least execution time across all of the three groups of input numbers which in turn means it took the least number of clock cycles to execute the sorting program.

Table 2. Comparison in terms of program execution time and number of clock cycles between three branch prediction designs

Branch prediction design		Program Execution time	No. of clock cycles	Clock period
Sorted[a] numbers	Alloyed	5055 ns	505.5	10 ns
	Correlated	5085 ns	508.5	10 ns
	Alloyed-Correlated	4425 ns	442.5	10 ns
Uniform[b] numbers	Alloyed	5325 ns	532.5	10 ns
	Correlated	5375 ns	537.5	10 ns
	Alloyed-Correlated	5235 ns	523.5	10 ns
Normal[c] numbers	Alloyed	5355 ns	535.5	10 ns
	Correlated	5365 ns	536.5	10 ns
	Alloyed-Correlated	5235 ns	523.5	10 ns

[a] Sorted number: numbers that are already arranged ascendingly.
[b] Uniform numbers: numbers that are symmetrically distributed.
[c] Normal numbers: numbers that are naturally ordered.

The reciprocal relation between the predictor performance and its CPI is presented in Figure 6, that is, the higher the performance, the less the CPI and vice versa. Where the highest performance for the alloyed-correlated branch predictor (225.989) was recorded in part (a) of Figure 6 during the arrangement of the sorted numbers group, which in part (b) corresponding to the lowest CPI value (1.516) for arranging the same input group. The effect of the three prediction schemes on MIPS processor speedup are revealed in Figure 7. It can be deduced that processor speedup rates of the first two predictors (alloyed and correlated) were almost close for all combinations of input data, then these values increased noticeably when using the alloyed-correlated predictor.

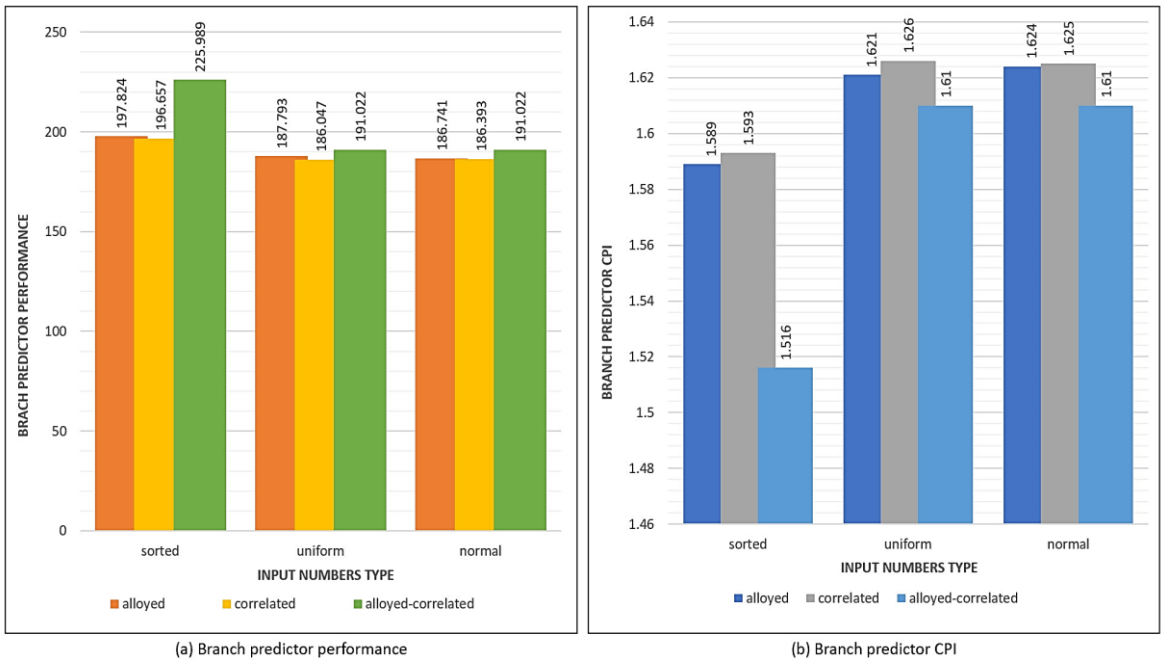

Figure 6. Branch predictor performance and clock per instruction (CPI) required by each scheme

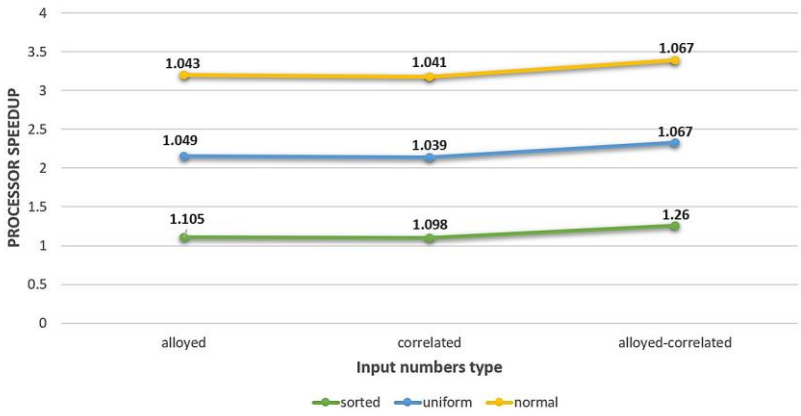

Figure 7. Comparison between three branch prediction schemes in term of processor speedup

5. CONCLUSION

In this research, a hardware model for a proposed dynamic branch predictor design has been configured in VHDL and synthesized using Xilinx ISE design suite 14.7 tool. This design decreases the penalty of branch misprediction caused by correlated branches by amalgamating the prediction mechanisms of two predictors (the alloyed predictor and the correlated predictor). After completing

the design, it has been included inside the fetch stage of a 5 stages pipelined MIPS processor then the selection sort algorithm was executed in order to arranging input numbers of different combinations ascendingly. The obtained results showed the effectiveness of the alloyed-correlated predictor design. At the end, it should be mentioned that this research is intended to be used in a computer architecture course where students will be able to design, implement, and debug a dynamic branch predictor and study its efficacy in minimizing the effect of nested conditional branches on prediction accuracy.

REFERENCES

[1] J. Hennessy and D. Patterson, "Computer Architecture: A Quantitative Approach," *5th ed. CA, USA: Morgan Kaufmann Publishers Inc.*, 2011.

[2] K. Thangarajan, et al., "Survey of branch prediction schemes for pipelined processors," *Proceedings of the Thirty-Fourth Southeastern Symposium on System Theory*, pp. 324-328, 2002.

[3] S. McFarling, "Combining Branch Predictors," *Digital Equipment Company -Western Research Laboratory*, vol. 49, 1993.

[4] D. Jimenez, "Reconsidering complex branch predictors," *Proceedings of the 9th International Symposium on High-Performance Computer Architecture*, pp. 43-52, 2003.

[5] A. Seznec, et al., "Design tradeoffs for Alpha EV8 conditional branch predictor," *ACM SIGARCH Computer Architecture News*, pp. 295-306, 2002.

[6] S. Gochman, et al., "The Intel Pentium M Processor: Microarchitecture and Performance," *Intel Technology Journal*, vol. 7, no. 2, pp. 21-59, 2003.

[7] J. W. Kwak and C. S Jhon, "High-performance embedded branch predictor by combining branch direction history and global branch history," *Proceedings IET Computers & Digital Techniques*, vol. 2, no. 2, pp. 142-154, 2008.

[8] Di Wu, K. Aasaraai, and A. Moshovos, "Low-cost, high-performance branch predictors for soft processors," *23rd International Conference on Field Programmable Logic and Applications (FPL)*, 2013.

[9] F. Prasad, et al., "Evaluating Branch Predictor Configurations for a MIPS-like Pipeline," *Swedish System-on-Chip Conference*, 2014.

[10] V. Saljooghi, et al., "Configurable RTL model for level-1 caches," *NORCHIP*, pp. 1-4, 2012.

[11] E. Sedano, et al., "Implementation of a hardware branch-predictor evaluation based on FPGA," *Ph.D. Research in Microelectronics and Electronics*, pp. 44-47, 2009.

[12] K. Matsui, et al., "An Efficient Implementation of a TAGE Branch Predictor For Soft Processor on FPGA," *IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*, pp. 108-115, 2019.

[13] P. Yiannacouras, et al., "Exploration and Customization of FPGA-Based Soft Processors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26 , no. 2, pp. 266-277, 2007.

[14] M. Gaudesi, et al., "On the in-field test of Branch Prediction Units using the correlated predictor mechanism," *17th International Symposium on Design and Diagnostics of Electronic Circuits & Systems*, pp. 286-289, 2014.

[15] T. Mudge, et al., "Correlation and Aliasing in Dynamic Branch Predictors," *23rd Annual International Symposium on Computer Architecture (ISCA'96)*, pp. 22-22, 1996.

[16] R. Thomas, et al., "Improving branch prediction by dynamic data flow-based identification of correlated branches from a large global history," *Proceedings of the 30th Annual International Symposium on Computer Architecture*, pp. 314-323, 2003.

[17] S. Pan, et al., "Improving the accuracy of dynamic branch prediction using branch correlation," *ASPLOS V: Proceedings of the fifth international conference on Architectural support for programming languages and operating systems*, pp. 76-84, 1992.

[18] F. Gao and S. Sair, "Exploiting Intra-function Correlation with Global History Stack," *International Workshop on Embedded Computer Systems*, pp. 172-181, 2005.

[19] A. Omondi, "The Microarchitecture of Pipelined and Superscalar Computers," *1st ed., Netherlands: Kluwer Academic publishers*, 1999.

[20] S. Sechrest, C.-C. Lee, and T. N. Mudge, "Correlation and aliasing in dynamic branch predictors," *Proceedings of the 23rd International Symposium on Computer Architecture*, vol. 24, no. 2, pp. 22-32, 1996.

[21] N. Panwar, M. Kaur and G. Singh, "Performance Analysis of Branch Prediction Unit for Pipelined Processors," *International Journal of Computer Applications*, vol. 128, no. 16, pp. 975-8887, 2015.

[22] Z. Lu, et al., "Alloyed branch history: Combining global and local branch history for robust performance," *International Journal of Parallel Programming*, vol. 31, no. 2, pp. 137-177, 2003.

[23] H. Arora, S. Kotecha and R. Samyal, "Dynamic Branch Prediction Modeler for RISC Architecture," *Proceedings International Conference Machine Intelligence and Research Advancement (ICMIRA)*, pp. 397-401, 2013.

[24] Di Wu and A. Moshovos, "Advanced branch predictors for soft processors," *International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, pp. 1-6, 2014.

[25] T. Zhang, et al., "Static Techniques to Improve Power Efficiency of Branch Predictors," *High Performance Computing-HIPC2004*, pp. 274-285, 2004.