IAES Journal of Electrical and Electronics Engineering
Vol. 10, No. 2, December 2023, pp. 31~35
ISSN: 2338-7432, DOI: 10.11591/jeee.v10i2.pp31-35

Configuration of an alloyed-correlated branch predictor for RISC processors: a review of history integration, FPGA implementation, and performance evaluation

Andri Febriyanto[1], Watra Arsadiando[2]
[1]Faculty of Industrial Technology, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
[2]Embedded Systems and Power Electronics Research Group, Yogyakarta, Indonesia

Article Info

Article history:

Received May 28, 2021
Revised Feb 5, 2022
Accepted Apr 6, 2022

Keywords:

Alloyed-correlated branch predictor
Branch prediction
FPGA
Global history
Local history
Pipeline architecture
RISC processor

ABSTRACT

Branch prediction is an important mechanism for improving instruction throughput in pipelined reduced instruction set computer (RISC) processors. Conditional branches interrupt sequential instruction fetching, and incorrect predictions can introduce pipeline flushing, execution delays, and additional processor overhead. Conventional local and global predictors address different aspects of branch behavior, but neither consistently captures all dependencies in programs containing nested or correlated branches. Alloyed-correlated branch prediction combines features of alloyed and correlated prediction to exploit multiple forms of branch-history information within a shared prediction structure. This review examines its theoretical foundations, hardware organization, and implementation in RISC processors, with particular attention to the FPGA-based design reported by Mahmood in 2021. The discussion covers global history registers (GHRs), pattern history tables (PHT), saturating counters, prediction indexing, update mechanisms, and integration into a five-stage microprocessor without interlocked pipelined stages (MIPS) pipeline. Reported performance results are considered alongside implementation constraints and experimental limitations. The review identifies important challenges involving history selection, table aliasing, hardware resources, prediction latency, and benchmark diversity. It concludes by proposing improvements in adaptive history management, systematic FPGA evaluation, and reproducible processor-level benchmarking.

This is an open access article under the CC BY-SA license.

1. INTRODUCTION

Branch instructions are essential to program execution because they support conditional decisions, loops, function calls, and changes in control flow. In a pipelined processor, however, the next instruction address may remain uncertain until the branch condition and target address have been resolved. Waiting for this information can interrupt instruction fetching and reduce the utilization of pipeline stages. Branch prediction addresses this problem by estimating the direction of a branch before its actual outcome becomes available [1], [2].

Dynamic branch predictors use information collected during program execution to improve subsequent predictions. Local prediction focuses on the previous behavior of an individual branch, whereas global prediction considers the outcomes of recently executed branches. Correlated prediction exploits relationships between different branches, particularly when the outcome of one conditional instruction depends on earlier decisions. These approaches provide useful information, but their effectiveness depends on whether the selected history representation captures the behavior of the branch being predicted [3]–[6].

Alloyed prediction was introduced to make local and global history information available within a common prediction structure. Rather than maintaining separate predictor components and selecting between them, the approach combines different history information when accessing prediction-table entries. This

organization can reduce wrong-history mispredictions and make more efficient use of limited hardware resources [7]. Mahmood subsequently presented an alloyed-correlated branch predictor implemented in very high-speed integrated circuit hardware description language (VHDL) and integrated into a five-stage microprocessor without interlocked pipelined stages (MIPS) processor, providing a practical design that combines alloyed and correlated prediction characteristics [8].

This review examines the configuration and operating principles of that implementation in the context of established branch-prediction research. It considers the predictor's history organization, table structure, indexing mechanism, counter updates, and FPGA-oriented design considerations. It also evaluates the reported performance evidence and identifies limitations that should be addressed before the architecture is adopted in more demanding RISC processor implementations. The objective is to establish a technically traceable assessment from the branch-prediction problem to hardware configuration and future design improvements.

2. THEORETICAL FOUNDATIONS OF BRANCH PREDICTION

2.1. Local, global, and correlated prediction

Local branch prediction maintains information about the recent outcomes of an individual branch instruction. This approach is useful when a branch follows a repetitive pattern, such as a loop termination condition or a recurring program decision. A local history register can represent the sequence of recent outcomes and use that sequence to select a prediction-table entry. However, local prediction may perform poorly when the behavior of a branch depends strongly on preceding branches rather than on its own history [3], [4].

Global branch prediction records the outcomes of recently executed branches in a shared global history register (GHR). The recorded sequence provides contextual information that can reveal relationships between different conditional branches. Correlated prediction uses this information to estimate the outcome of a current branch based on previous branch patterns. Yeh and Patt [3], [4] established the two-level adaptive prediction framework, while Pan *et al*. [5] demonstrated how branch correlation can improve prediction accuracy.

A limitation of global-history prediction is that unrelated branches can introduce noise into the history information. A branch may depend on an earlier branch whose outcome is no longer represented in a short history register. Conversely, a long history may introduce additional information that increases table aliasing and hardware requirements. These limitations motivate predictor structures that combine different history representations while controlling storage and indexing complexity [6], [9]–[12].

2.2. Alloyed and alloyed-correlated prediction

Alloyed prediction combines local and global history information within a common prediction-table indexing structure. This organization is intended to address wrong-history mispredictions, which occur when the predictor uses a type of history that is unsuitable for the current branch. Skadron *et al.* [7] showed that wrong-history errors can represent an important source of prediction failures and proposed alloyed prediction as an alternative to conventional hybrid selection. Subsequent work on branch-history organization and local predictors further demonstrates the importance of choosing appropriate history information [9], [10].

The alloyed-correlated design considered here extends the basic concept by combining an alloyed-style index with additional correlated-history selection. In the reported implementation, a portion of the GHR is combined with the branch address to select a prediction-table entry. Other history bits select one of several counter groups within that entry. This arrangement allows the predictor to use both address-related and branch-correlation information without requiring entirely separate local and global predictor modules [8].

3. HARDWARE CONFIGURATION OF THE ALLOYED-CORRELATED PREDICTOR

3.1. Main architectural components

The principal components of the reported architecture are the GHR, pattern history table (PHT), two-bit saturating counters, and associated selection logic. The GHR records the outcomes of recently executed branches, while the PHT stores prediction states associated with particular history and address combinations. The counter-selection mechanism determines which prediction state is used during instruction fetching. These components are integrated into the processor's fetch and decode operations to provide dynamic branch prediction [8].

The reported design uses a seven-bit GHR and a PHT containing 1,024 entries. Each PHT entry contains four sets, with each set holding a two-bit saturating counter. The predictor therefore contains 4,096 two-bit counters, corresponding to 8,192 bits of counter storage, excluding the GHR and additional control logic. The design initializes the counters to the binary state 11, representing a strong taken prediction [8].

3.2. History indexing and counter selection

The predictor uses two distinct stages of selection to identify the appropriate prediction state. First, the upper five bits of the seven-bit GHR are concatenated with five lower bits of the branch program counter to form a ten-bit index. This index selects one of the 1,024 PHT entries. Second, the lower two bits of the GHR select one of the four counter sets within that entry [8].

The selected two-bit counter provides the branch-direction prediction through its most significant bit. A value indicating the taken state causes the processor to fetch from the predicted branch target, while the alternative state supports sequential instruction fetching. Once the actual branch outcome becomes available, the corresponding counter is incremented or decremented according to the outcome. The counter remains within its two-bit saturation limits, preventing a single outcome from immediately reversing a strongly established prediction [3], [8].

Table 1 summarizes the reported hardware configuration and its functional roles. The table distinguishes the history register, PHT organization, address indexing, counter selection, and update behavior. These parameters are important because changing the history length or table organization affects prediction accuracy, storage requirements, and access timing. The values in the table describe Mahmood's published implementation rather than universal settings for all alloyed-correlated predictors.

Table 1. Reported configuration of the alloyed-correlated branch predictor

Component	Reported configuration	Functional role
Processor	32-bit MIPS, five-stage pipeline	Executes instructions and integrates prediction
Description language	VHDL	Hardware modeling and implementation
Development environment	Xilinx ISE Design Suite 14.7	Design and schematic development
Global History Register	7 bits	Stores recent branch outcomes
PHT capacity	1,024 entries	Stores branch prediction states
Counter organization	Four sets per entry	Supports correlated-history selection
Counter size	2 bits per set	Represents prediction confidence
PHT index	Five GHR bits concatenated with five PC bits	Selects a table entry
Set selection	Two remaining GHR bits	Selects one of four counters
Initial counter state	11	Strongly taken
Counter update	Increment or decrement after branch resolution	Learns from actual outcomes

The configuration in Table 1 illustrates how the predictor combines two levels of history selection within a relatively compact structure. The ten-bit index provides 1,024 possible PHT locations, while the remaining GHR bits provide four alternative prediction states at each location. This organization creates a total of four counter choices for each indexed address-history combination. The resulting architecture offers a practical example of integrating alloyed and correlated prediction without implementing a full tournament predictor [7], [8].

3.3. Integration into the RISC pipeline

The reported predictor is integrated into a pre-existing 32-bit MIPS processor with five pipeline stages. Its prediction mechanism operates during instruction fetching, allowing the processor to make an early decision about the next instruction address. The actual branch outcome becomes available later in the pipeline, where the prediction state can be updated. This separation between prediction and learning is fundamental to dynamic branch-prediction hardware [8], [13].

The VHDL implementation uses finite-state-machine logic and flip-flops to represent the two-bit saturating counters. The counter's next state depends on its current state and the actual branch outcome supplied during the decode stage. The design was developed using Xilinx ISE design suite 14.7, and its schematic representation identifies the GHR, PHT, and counter structures. This implementation provides a useful educational model for understanding the relationship between branch history and processor control flow [8].

4. FPGA IMPLEMENTATION AND PERFORMANCE EVALUATION

The reported experiment evaluated the alloyed-correlated predictor using a selection-sort program that arranged 100 integer values. Three input distributions were considered: already sorted, uniformly distributed, and normally distributed data. The study compared the proposed architecture with separate alloyed and correlated predictors. Program execution time and clock-cycle counts were used to assess the resulting processor behavior [8].

The results reported by Mahmood show that the alloyed-correlated predictor achieved the shortest execution time for all three input groups. For sorted inputs, the reported execution time was 4,425 ns, compared with 5,055 ns for the alloyed predictor and 5,085 ns for the correlated predictor. For uniformly

distributed inputs, the corresponding values were 5,235 ns, 5,325 ns, and 5,375 ns. For normally distributed inputs, the proposed design required 5,235 ns, compared with 5,355 ns and 5,365 ns for the two reference predictors [8].

Table 2 presents these reported results for direct comparison. The values indicate that the proposed configuration improved execution time in the tested selection-sort workload. However, the experiment does not establish that the same improvement would occur across other algorithms, processor pipelines, or benchmark suites. The results should therefore be interpreted as evidence from a specific implementation and test environment rather than as a universal performance guarantee.

Table 2. Reported execution-time comparison of branch-prediction configurations

Input distribution	Alloyed predictor (ns)	Correlated predictor (ns)	Alloyed-correlated predictor (ns)
Sorted	5,055	5,085	4,425
Uniform	5,325	5,375	5,235
Normal	5,355	5,365	5,235

The results in Table 2 suggest that combining the two prediction approaches can reduce execution time under the reported conditions. The largest difference occurs for sorted inputs, where the alloyed-correlated design performs better than both individual predictors. The improvements are smaller for uniform and normal inputs, indicating that the benefit depends on the branch patterns generated by the workload. A broader evaluation is required to determine how the architecture behaves with different branch densities, nested loops, and more complex control-flow structures [8], [14].

5. DESIGN CONSIDERATIONS AND LIMITATIONS

The first design consideration is the trade-off between history length and hardware complexity. A longer GHR can represent more branch relationships, but it may also increase the number of possible history patterns and create additional aliasing in a finite PHT. A shorter history reduces storage and indexing complexity but may omit important correlations. The selected seven-bit history should therefore be considered an implementation choice that requires workload-specific validation [7], [9], [10], [12].

The second consideration concerns predictor latency and pipeline timing. A branch predictor must deliver its result early enough to support instruction fetching without becoming a critical timing path. Concatenation, table access, and counter selection must therefore be implemented with careful attention to logic depth and memory organization. FPGA-based predictor studies demonstrate that a theoretically accurate design may still face practical limitations involving available memory blocks, logic resources, and operating frequency [15]–[17].

The third limitation concerns the reported experimental methodology. The use of a single sorting algorithm and 100 input values provides a useful demonstration but does not offer broad evidence of general-purpose processor performance. The study also does not provide a comprehensive evaluation across standard benchmark suites, multiple FPGA devices, power consumption, or detailed resource utilization. Future experiments should report prediction accuracy, mispredictions per thousand instructions, cycles per instruction, execution time, logic utilization, memory usage, and maximum operating frequency [8], [13]–[18].

6. RESEARCH OPPORTUNITIES AND FUTURE DIRECTIONS

Future research should investigate adaptive history management to improve the predictor's ability to capture different branch-correlation patterns. A configurable GHR length could allow designers to evaluate alternative history sizes under a fixed hardware budget. Further work could also investigate dynamic selection of the counter sets or alternative indexing functions to reduce conflict aliasing. These modifications should be evaluated against the original configuration to determine whether additional complexity produces measurable benefits [7], [9], [10], [12], [19].

Another important direction is systematic FPGA validation using representative RISC workloads. The predictor should be tested with established benchmark programs and a range of branch-intensive applications rather than a single sorting workload. Hardware evaluation should also include synthesis reports, resource utilization, clock frequency, power consumption, and timing analysis. This would provide a clearer understanding of the trade-off between prediction accuracy and implementation cost [15]-[17].

Finally, future implementations should consider compatibility with more advanced RISC processor architectures. The MIPS-based educational implementation provides a useful starting point, but modern RISC-V and other embedded processor designs may have different pipeline depths, instruction-fetch widths, and branch-resolution stages. The predictor's history-update mechanism must also account for speculative

execution and recovery after misprediction. These considerations are essential for translating the proposed configuration into a practical processor front end [13], [14], [20].

7. CONCLUSION

The alloyed-correlated branch predictor combines alloyed and correlated history information to improve conditional branch prediction in pipelined RISC processors. The reviewed configuration uses a seven-bit GHR, a 1,024-entry PHT, four two-bit counter sets per entry, and a two-stage history-selection mechanism. Its implementation in VHDL and integration into a five-stage MIPS processor demonstrate how the architecture can be translated into a hardware-oriented design. The reported selection-sort experiment shows lower execution times than the separate alloyed and correlated predictors for the three tested input distributions. Nevertheless, the available evidence is limited by the narrow benchmark scope and the absence of comprehensive FPGA resource, power, and timing measurements. The architecture should therefore be viewed as a promising educational and experimental configuration rather than a fully established solution for all RISC processors. Further research should prioritize adaptive history organization, aliasing reduction, standard benchmark evaluation, and physical FPGA validation. These improvements would provide stronger evidence of its suitability for embedded and general-purpose processor applications.

REFERENCES

[1] J. E. Smith, "A study of branch prediction strategies," in *Proceedings - International Symposium on Computer Architecture*, 1981, pp. 135–148, doi: 10.1145/285930.285980.

[2] S. Mittal, "A survey of techniques for dynamic branch prediction," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 1, 2019, doi: 10.1002/cpe.4666.

[3] T. Y. Yeh and Y. N. Patt, "Two-level adaptive training branch prediction," *Proceedings of the Annual International Symposium on Microarchitecture, MICRO*, pp. 51–61, 1991, doi: 10.1145/123465.123475.

[4] T. Y. Yeh and Y. N. Patt, "Alternative implementations of two-level adaptive branch prediction," *Proceedings of the Ninth Annual International Symposium on Computer Architecture*, pp. 124–134, 1993, doi: 10.1145/139669.139709.

[5] S.-T. Pan, K. So, and J. T. Rahmeh, "Improving the accuracy of dynamic branch prediction using branch correlation," pp. 76–84, 1992, doi: 10.1145/143365.143490.

[6] C. Young, N. Gloy, and M. D. Smith, "Comparative analysis of schemes for correlated branch prediction," in *Conference Proceedings - Annual International Symposium on Computer Architecture, ISCA*, 1995, pp. 276–286, doi: 10.1145/225830.224438.

[7] K. Skadron, M. Martonosi, and D. W. Clark, "Taxonomy of branch mispredictions, and alloyed prediction as a robust solution to wrong-history mispredictions," *Parallel Architectures and Compilation Techniques - Conference Proceedings, PACT*, pp. 199–206, 2000, doi: 10.1109/pact.2000.888344.

[8] H. S. H. Mahmood, "FPGA configuration of an alloyed correlated branch predictor used with RISC processor for educational purposes," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 1, pp. 265–271, 2021, doi: 10.11591/ijece.v11i1.pp265-271.

[9] S. Reches and S. Weiss, "Implementation and analysis of path history in dynamic branch prediction schemes," *IEEE Transactions on Computers*, vol. 47, no. 8, pp. 907–912, 1998, doi: 10.1109/12.707596.

[10] J. W. Kwak, J. H. Kim, and C. S. Jhon, "The impact of branch direction history combined with global branch history in branch prediction," *IEICE Transactions on Information and Systems*, vol. E88-D, no. 7, pp. 1754–1758, 2005, doi: 10.1093/ietisy/e88-d.7.1754.

[11] I. C. K. Chen, J. T. Coffey, and T. N. Mudge, "Analysis of branch prediction via data compression," *SIGPLAN Notices (ACM Special Interest Group on Programming Languages)*, vol. 31, no. 9, pp. 128–137, 1996, doi: 10.1145/248209.237171.

[12] P. Michaud, A. Seznec, and R. Uhlig, "Trading conflict and capacity aliasing in conditional branch predictors," in *Proceedings of the 24th annual international symposium on Computer architecture*, May 1997, pp. 292–303, doi: 10.1145/264107.264211.

[13] H. P. Baranda, J. C. Penha, and R. Ferreira, "Implementation of a branch predictor in the 5-stage MIPS pipeline," *International Journal of Computer Architecture Education*, vol. 6, no. 1, pp. 18–26, Dec. 2017, doi: 10.5753/ijcae.2017.4862.

[14] N. Soundararajan *et al.*, "Towards the adoption of local branch predictors in modern out-of-order superscalar processors," *Proceedings of the Annual International Symposium on Microarchitecture, MICRO*, pp. 519–530, 2019, doi: 10.1145/3352460.3358315.

[15] D. Wu, K. Aasaraai, and A. Moshovos, "Low-cost, high-performance branch predictors for soft processors," *2013 23rd International Conference on Field Programmable Logic and Applications, FPL 2013 - Proceedings*, 2013, doi: 10.1109/FPL.2013.6645536.

[16] D. Wu and A. Moshovos, "Advanced branch predictors for soft processors," *2014 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2014*, 2014, doi: 10.1109/ReConFig.2014.7032495.

[17] E. Sedano, D. Chaver, and J. Resano, "Implementation of a hardware branch-predictor evaluation platform based on FPGAs," *2009 Ph.D. Research in Microelectronics and Electronics, PRIME 2009*, pp. 44–47, 2009, doi: 10.1109/RME.2009.5201346.

[18] C. A. Rathi, G. Rajakumar, T. A. Kumar, and T. S. A. Samuel, "Design and development of an efficient branch predictor for an in-order RISC-V processor," *Journal of Nano- and Electronic Physics*, vol. 12, no. 5, pp. 1–4, 2020, doi: 10.21272/jnep.12(5).05021.

[19] P. Y. Chang, E. Hao, T. Y. Yeh, and Y. Patt, "Branch classification: a new mechanism for improving branch predictor performance," *International Journal of Parallel Programming*, vol. 24, no. 2, pp. 133–158, 1996, doi: 10.1007/BF03356745.

[20] A. Seznec, S. Felix, V. Krishnan, and Y. Sazeides, "Design tradeoffs for the Alpha EV8 conditional branch predictor," *Conference Proceedings - Annual International Symposium on Computer Architecture, ISCA*, pp. 295–306, 2002, doi: 10.1109/ISCA.2002.1003587.