Journal of VLSI circuits and systems
RESEARCH ARTICLE
Journal of VLSI Circuits and Systems, ISSN: 2582-1458 Vol. 7, No. 2, 2025 (pp. 68-76)
WWW.VLSIJOURNAL.COM

Implementation of an Efficient RISC-V Processor Featuring a Novel Gshare Branch Prediction Technique

Tri-Duc Ta[1,2,3]*, Thanh-Phat Nguyen[1,3], Quoc-Thinh Tran[1,3]
[1]ASICLAB, University of Information Technology, Ho Chi Minh City, Vietnam.
[2]Faculty of Electronics and Telecommunication, The University of Science, Ho Chi Minh City, Vietnam.
[3]Vietnam National University, Ho Chi Minh City, Vietnam.

KEYWORDS:
RISC-V,
VLSI,
Branch prediction,
Gshare,
BTB,
PHT,
GHR

ARTICLE HISTORY:
Received: 18.11.2025
Revised: 10.12.2025
Accepted: 18.12.2025

DOI:
https://doi.org/10.31838/JCVS/07.02.08

Abstract

RISC-V, characterized by its straightforward and open-source instruction set design, is becoming a compelling platform for contemporary Internet of Things devices. Dynamic branch prediction, especially two-level methods and history/address hashing approaches such as Gshare, has demonstrated significant efficacy in alleviating control hazards in pipelined processors. This study introduces a five-stage (IF-ID-EX-MEM-WB) RISC-V core that incorporates a branch prediction unit (BPU), which merges a branch target buffer (BTB) and a pattern history table (PHT) featuring 256 entries with two-bit saturating counters. The PHT index is derived from XOR(GHR, PC[9:2]), while the BTB is refreshed utilizing the lower eight bits of the PC. The design was executed from RTL to GDSII with Cadence Genus, Conformal, and Innovus on GPDK045 (45 nm) technology, illustrating feasibility beyond research confined to RTL or FPGA, such as RVCoreP. RTL simulation verified the accurate execution of all 37 RV32I instructions and attained roughly 90.4% accuracy in branch prediction for branch-intensive tasks. Postlayout results indicate that the design attained the target frequency (exceeding 50 megahertz), with a recorded maximum frequency of roughly 75 MHz. The overall power consumption of the core is around 15.078 mW across an area of roughly 0.69 mm^2, resulting in a core density of nearly 70%. These data validate the feasibility of employing two-level branch prediction in lightweight RISC-V microcontrollers. In contrast to prior RISC-V cores that either rely on simple branch handling or evaluate Gshare only at the RTL/FPGA level, this work delivers a fully synthesizable RV32I microcontroller-class core that integrates a BTB+GHR+PHT Gshare predictor and is validated through a complete RTL-to-GDSII ASIC flow with postlayout power-performance-area analysis in a 45-nm standard-cell technology.

Authors' e-mail ID: ductt@uit.edu.vn, phatnt@uit.edu.vn, thinhtq@uit.edu.vn

Authors' ORCID IDs: 0000-0001-9458-012X, 0009-0003-4835-5599, 0009-0006-9743-4535

How to cite this article: Tri-Duc Ta, Thanh-Phat Nguyen, Quoc-Thinh Tran, Implementation of an Efficient RISC-V Processor Featuring a Novel Gshare Branch Prediction Technique, Journal of VLSI Circuits and System, Vol. 7, No. 2, 2025 (pp. 68-76).

Introduction

The open-source RISC-V instruction set architecture (ISA) has become a popular choice for academic and industrial research due to its simplicity, modularity, and scalability.[1,2] The proliferation of RISC-V cores in low-power embedded and Internet of Things (IoT) platforms[3,4] has made improving instruction-level parallelism while maintaining size and power consumption a major processor design challenge.

Pipeline performance is limited by conditional branch instruction control hazards, which reduce fetch efficiency and increase CPI. Dynamic branch prediction is a well-studied method that reduces control hazards by predicting a branch's outcome and target address before execution. Two-level adaptive branch predictors[5] and address/history hashing algorithms like Gshare[6] have outperformed static or one-bit prediction models since the early 1990s. The two-level model uses a Global

History Register (GHR) to store recent branch results and a pattern history table (PHT) with two-bit saturating counters to recall prediction patterns. To reduce aliasing among unrelated branches and reduce hardware costs, the Gshare version uses an XOR operation between the GHR and the program counter (PC) lower-order bits.

Recently, many RISC-V implementations have tested lightweight and advanced branch predictors. Open-source cores like RVCoreP[2], Rocket Chip[7], and BOOM[8] integrate two-level or TAGE-based predictors, leading to significant performance enhancements over static architectures. Jin et al.[9] real-time pipeline optimization strategy based on branch prediction and data dependency analysis advances RISC-V branch prediction optimization research. McFarling's technical research introduced Gshare, a dynamic prediction method that uses an XOR operation between the GHR and the PC.[6] These systems often use or improve this approach. Side-channel vulnerabilities in the branch prediction unit (BPU) have piqued researchers' interest in branch predictor security.[10,11]

Despite these advances, most of the previously reported efforts only simulate or FPGA prototype, failing to demonstrate actual synthesis and postlayout (GDSII) viability, highlighting device physical limits. Few studies explain how to integrate a complete BPU—including BTB, GHR, and PHT—into a five-stage RISC-V pipeline. This study addresses the deficiency by designing an RTL-to-layout two-level Gshare-based branch predictor in a synthesizable RV32I core using the Cadence Genus, Conformal, and Innovus toolchain on GPDK045 (45 nm) technology. Functional correctness, improved prediction accuracy on branch-intensive tasks, and competitive postlayout performance show that resource-limited RISC-V microcontrollers can execute complex prediction logic.

This work makes the following contributions:

1. We design a five-stage, RV32I, microcontroller-class core that tightly integrates a two-level Gshare BPU comprising an eight-bit GHR, a 256-entry PHT with 2-bit saturating counters, and a 256-entry branch target buffer (BTB).
2. We present the concrete integration of this BTB+GHR+PHT Gshare predictor into the pipeline, detailing how the IF and EX stages cooperate with the hazard unit to resolve branches, update prediction structures, and handle mispredictions with a bounded penalty.
3. We implement the core using a complete ASIC RTL-to-GDSII flow in GPDK045 (45 nm) and report post-layout power–performance–area (PPA) metrics (frequency, power, area, and utilization), thereby providing silicon-level evidence of the cost and benefits of employing Gshare in lightweight RV32I designs.

Further study sections are organized as follows: Section II describes the theoretical basis for RISC-V pipeline design and traditional branch prediction methods, focusing on Gshare. The RV32I core's RTL design, integrated BPU, and ASIC physical architecture are covered in Section III. Results from functional simulation and postlayout PPA measurements are presented in Section IV. The final section summarizes the contributions.

Literature Review

UC Berkeley developed the versatile, open-source RISC-V ISA for both educational and commercial use.[1] Its modular design allows architects to tailor cores to specific performance, area, and power constraints, in contrast to proprietary ISAs such as ARM or x86. Several open-source cores have demonstrated the practical viability of RISC-V. The Rocket Chip generator[7] produces the in-order, five-stage Rocket core, which has become a common reference point for research prototypes and custom SoCs. BOOM[8] builds upon the same ecosystem by introducing a superscalar out-of-order pipeline and advanced branch prediction schemes such as TAGE to approach the performance of contemporary commercial processors. In Europe, the parallel ultra-low-power (PULP) platform developed by ETH Zurich and the University of Bologna targets energy-efficient IoT and embedded applications and has produced SoCs and the Ariane/CVA6 core,[12,13] demonstrating how RISC-V scales from microcontroller-class designs to 64-bit Linux-capable systems. Early commercial products such as the SiFive FE310 SoC,[4] which integrates the E31 RISC-V core, further validated RISC-V as a viable option for industrial manufacturing.

Across these design points, pipelined RISC-V cores share a common challenge: efficiently handling hazards during instruction execution. Control hazards caused by conditional branch instructions can significantly degrade performance by reducing instruction fetch efficiency and increasing the cycles per instruction (CPI). To mitigate this penalty, branch prediction has become a central technique for improving throughput in pipelined processors. Early approaches relied on static schemes (e.g., "backward taken, forward not taken"), which are simple but cannot exploit dynamic program behavior. Yeh and

Patt's two-level adaptive predictor[5] was a key breakthrough: it separates branch history, stored in a Branch History Table (BHT) or GHR, from pattern history, stored in a PHT of 2-bit saturating counters.

McFarling refined the two-level scheme with the Gshare predictor,[6] which computes the PHT index by XOR-ing the GHR with a subset of lower-order bits from the branch address. By combining address and history information, Gshare reduces destructive aliasing—where unrelated branches map to the same counter—while keeping hardware cost modest. This favorable trade-off has made Gshare a widely used baseline in both academic studies and industrial microarchitectures, especially for area- and power-constrained designs. Beyond classical two-level schemes, more sophisticated predictors have been proposed to capture long-range and complex correlations. The perceptron predictor of Jiménez and Lin[14] employs a simple neural network-like structure to learn linear correlations in branch behavior, and TAGE (TAgged GEometric history length), introduced by Seznec and Michaud,[15] organizes multiple tagged prediction tables with geometrically increasing history lengths, using a simpler predictor such as Gshare as a fallback. TAGE-based designs have consistently ranked among the top performers in championship branch prediction competitions and are used in high-performance processors such as BOOM.

The RISC-V ecosystem has rapidly incorporated these branch prediction techniques. Enhanced five-stage soft processors such as RVCoreP[2] deploy two-level predictors on FPGA to demonstrate performance gains over static schemes in simple RV32I pipelines. High-performance out-of-order cores like BOOM[8] employ TAGE-like predictors to compete with commercial superscalar CPUs. Other work has explored how branch prediction interacts with pipeline timing; for example, Jin et al. propose a real-time optimization strategy that jointly considers branch prediction and data dependencies in RISC-V pipelines.[9] In parallel, the BPU has been examined from a security perspective: Kim et al. identify side-channel vulnerabilities associated with BPU structures,[10] and Kocher et al. analyze how different predictor organizations (e.g., Gshare versus TAGE) influence the susceptibility of RISC-V cores to Spectre-style attacks.[11]

In summary, existing RISC-V cores such as Rocket, BOOM, and Ariane/CVA6 primarily target high-performance or Linux-capable systems and often employ relatively complex branch predictors such as TAGE, while designs like RVCoreP evaluate two-level prediction only as an FPGA soft core. Classical Gshare-based studies, on the other hand, typically focus on prediction accuracy or security aspects in abstract or non-RISC-V contexts and do not provide detailed postlayout PPA results for a simple five-stage RV32I microcontroller-class core. Although prior work has thoroughly explored branch prediction accuracy[5,6,14,15] and microarchitectural security[10,11], most reported efforts rely on RTL simulation or FPGA prototyping and therefore do not fully reflect the physical constraints of VLSI implementation. As a result, empirical PPA data for integrating a conventional BTB+GHR+PHT Gshare predictor into a compact five-stage RV32I pipeline in a mainstream 45 nm process remain scarce, motivating the ASIC-level study presented in this work.

Proposed Architecture

Figure 1 illustrates that the system is constructed on an RV32I RISC-V core and features a traditional five-stage pipeline architecture (IF, ID, EX, MEM, WB). In this organization, conditional branches are resolved in the EX stage, and a misprediction flushes the IF and ID stages, resulting in a fixed two-cycle penalty from fetch to resolution. This pipeline architecture is derived from designs such as Rocket[7] and RVCoreP[2], which have demonstrated efficacy in performance and implementation simplicity on FPGAs and ASICs.

The execution flow of an instruction commences at the IF (Instruction Fetch) stage, which retrieves the 32-bit instruction from Instruction Memory (I-Mem) according to the address supplied by the PC. This stage incorporates a Multiplexer (MUX) that selects the pc_next address based on predictions from the BPU, rather than consistently use PC+4, so enabling the pipeline to retrieve instructions without awaiting the branch outcome from the EX stage. Subsequently, during the Instruction Decode (ID) phase, the instruction is interpreted to retrieve the opcode, funct3, funct7, and source register (rs1, rs2) fields. These data are transmitted to the Controller—a finite state machine that produces control signals (such as alu_ctrl and memWrite) for the following phases. Concurrently, the Register File retrieves the relevant operands for the execution stage.

The EX (Execute) stage is where arithmetic and logical operations are conducted through the ALU. This is also the location where conditional branch instructions are determined. The ALU recomputes the branch target address and generates a branch_taken signal in EX, which is compared against the speculative direction from the BPU and forwarded to the hazard unit

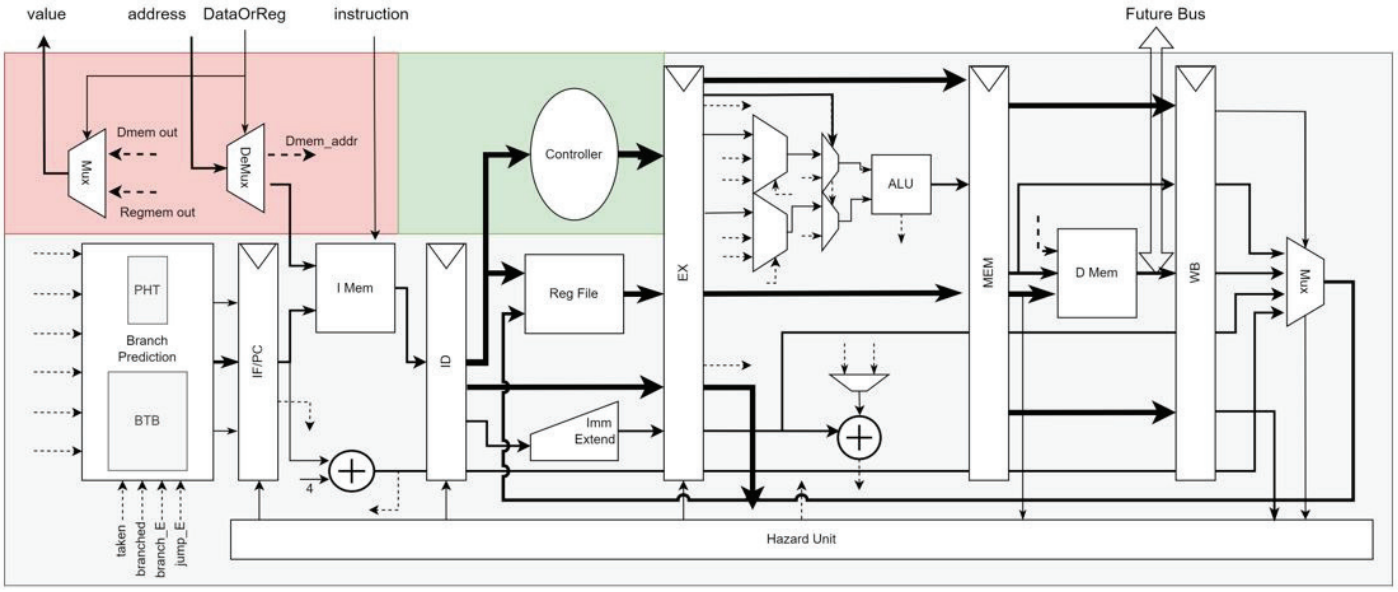

Fig. 1: Architecture of the RISC-V processor

to control pc_restore and pipeline flush. Subsequently, the MEM (Memory Access) stage engages with the Data Memory (D-Mem), executing a read operation for load instructions and a write operation for store instructions. The Write Back (WB) step is tasked with writing the result to the Register File by picking data from either the ALU or D-Memory via a multiplexer.

In addition to the primary data flow, a vital element is the hazard unit, responsible for overseeing and coordinating data flow across stages to guarantee proper pipeline functionality. It employs two fundamental ways to accomplish this. Data forwarding addresses Read-After-Write (RAW) problems by transmitting results directly from the EX or MEM stage to the ALU inputs during the EX stage. The second solution is pipeline stalling, employed to address load-use dangers when the outcome of a load command is not yet available. Upon identifying this conflict, the Hazard Unit introduces a "bubble" into the pipeline and suspends the IF–ID stages for one cycle to maintain data integrity.

A two-level Gshare (two-level adaptive branch predictor) BPU was developed and incorporated directly into the Instruction Fetch step of the pipeline to mitigate control hazards. The Gshare algorithm was selected for its ideal equilibrium between precision and hardware expenditure, markedly surpassing static predictors while being considerably less complex than sophisticated models such as TAGE.[15]

Figure 2 depicts the BPU architecture, which comprises three primary blocks. The fundamental element is the PHT, a predictive memory of 256 direct-mapped

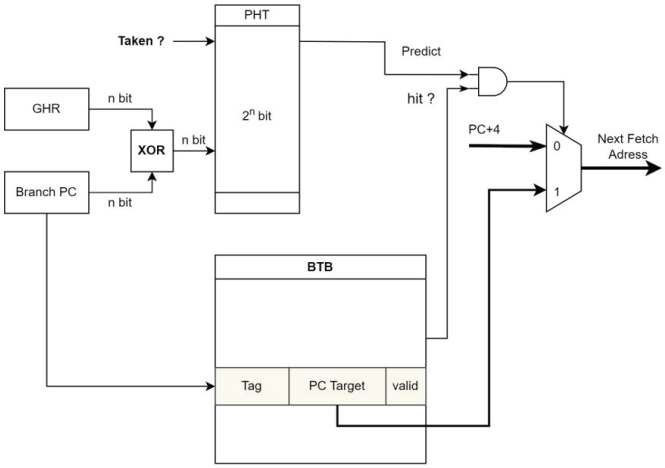

Fig. 2: Architecture of the BPU

entries, with each entry represented by a 2-bit saturating counter. The defining characteristic of Gshare is its method of accessing the PHT: the index is produced using an XOR operation between the eight least significant bits of the PC (PC[9:2]) and the value of the GHR. The GHR is an eight-bit register utilized to retain the Taken/Not-Taken state of the eight most recent branches. This integrated indexing method markedly diminishes aliasing, which arises when distinct branches correspond to the same counter. The BTB, a 256-entry direct-mapped cache, operates concurrently with the PHT, storing the target address (Target PC) of previously encountered branches, accompanied by a tag and a valid bit for branch identification. In this implementation, each BTB entry stores a 32-bit target address, an eight-bit tag derived from PC[9:2], and a one-bit valid flag, providing a compact organization suitable for microcontroller-class RV32I cores.

In each fetch cycle, the current PC (Branch PC) is utilized to simultaneously query both the PHT and the BTB. The PHT employs the Gshare index (PC[9:2] ⊕ GHR[7:0]) to access the two-bit counter, thus ascertaining the expected direction (Taken or Not-Taken). Concurrently, the BTB is accessed utilizing the PC address to ascertain whether it constitutes a hit or a miss. When the PHT counter shows a "Taken" state and the BTB indicates a "hit," the MUX in the IF stage will choose the target PC from the BTB as the Next Fetch Address. Otherwise, the pipeline continues to retrieve progressively using PC+4. This approach enables the pipeline to consistently prepare for the subsequent instruction retrieval, even prior to the determination of the actual branch conclusion in the EX stage. From a pipeline perspective, a branch instruction therefore spends one cycle in IF (speculative fetch on the predicted path), one cycle in ID (decoding and operand read), and is finally resolved in EX; if the prediction is correct, the subsequent instructions proceed without interruption, whereas an incorrect prediction triggers a flush and PC redirection.

The accuracy of a prediction is validated solely after the branch instruction is determined in the Execute (EX) step. The actual outcome (Taken or Not-Taken) is relayed to the BPU to update all three components. The GHR register is shifted left by one bit, with the new actual outcome entered as the least significant bit. The two-bit counter at the relevant Gshare index in the PHT is modified (incremented if Taken, decremented if Not-Taken) in accordance with the saturating principle illustrated in Figure 3. If the branch was indeed "Taken," the BTB is updated with the (PC, Target PC) pair, and its valid bit is assigned a value of 1. All three structures (GHR, PHT, BTB) are updated in the same cycle when the branch reaches EX, so that subsequent branches immediately see the updated global history and target information. Upon a misprediction (when the IF prediction diverges from the EX result), a flush signal is triggered to eliminate the erroneously fetched instructions and revert the PC to the accurate location (pc_restore). This method results in a penalty of roughly two cycles; however, it substantially decreases the overall frequency of pipeline stalls when branches are accurately predicted.

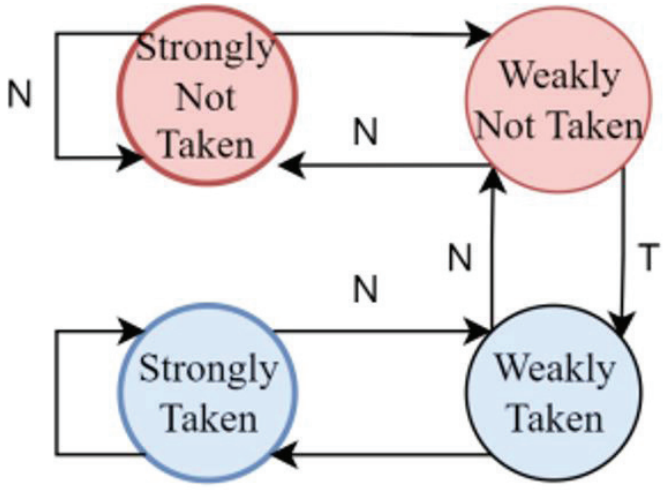

Fig. 3: State diagram of a 2-bit saturating counter

Results

The design underwent thorough verification in three stages: RTL functional verification, branch prediction performance assessment, and physical (ASIC) implementation to provide Power–Performance–Area (PPA) metrics. The findings indicate that the RISC-V RV32I core, when combined with the Gshare two-level branch predictor, attains functional stability, elevated prediction accuracy, and commendable resource efficiency. For clarity, this section first summarizes the functional verification, then explains the branch prediction benchmarking method, and finally describes the postlayout PPA extraction procedure used to obtain the reported numbers.

The RTL code was simulated and validated via Cadence Xcelium and Vivado Simulator. All 37 instructions in the base RV32I instruction set (comprising R, I, S, B, U, and J formats) were evaluated using specific test cases to verify the functional accuracy of each component. The simulation waveform results, depicted in Figure 4, demonstrate that the pipeline accurately handles R, I, and S-type instruction categories. Memory (Load/Store) and Register File access operations were executed precisely, demonstrating the accuracy of the Datapath and memory access control logic.

Simultaneously, the hazard unit was validated using data conflict scenarios, encompassing RAW and load-use risks. The simulation results demonstrated that the data forwarding, and pipeline stalling methods functioned correctly, ensuring the pipeline operated seamlessly without any deadlocks. A comprehensive test program, comprising four nested for-loops, was completed successfully, demonstrating that the entire pipeline, including the BPU, functions correctly under actual operating conditions. For the branch prediction evaluation, this benchmark was augmented with two hardware counters: a "TotalBranches" counter, which is incremented whenever the controller decodes a conditional branch instruction in the ID stage, and a "Mispredictions" counter, which is incremented whenever the BPU asserts the flush signal in the EX stage

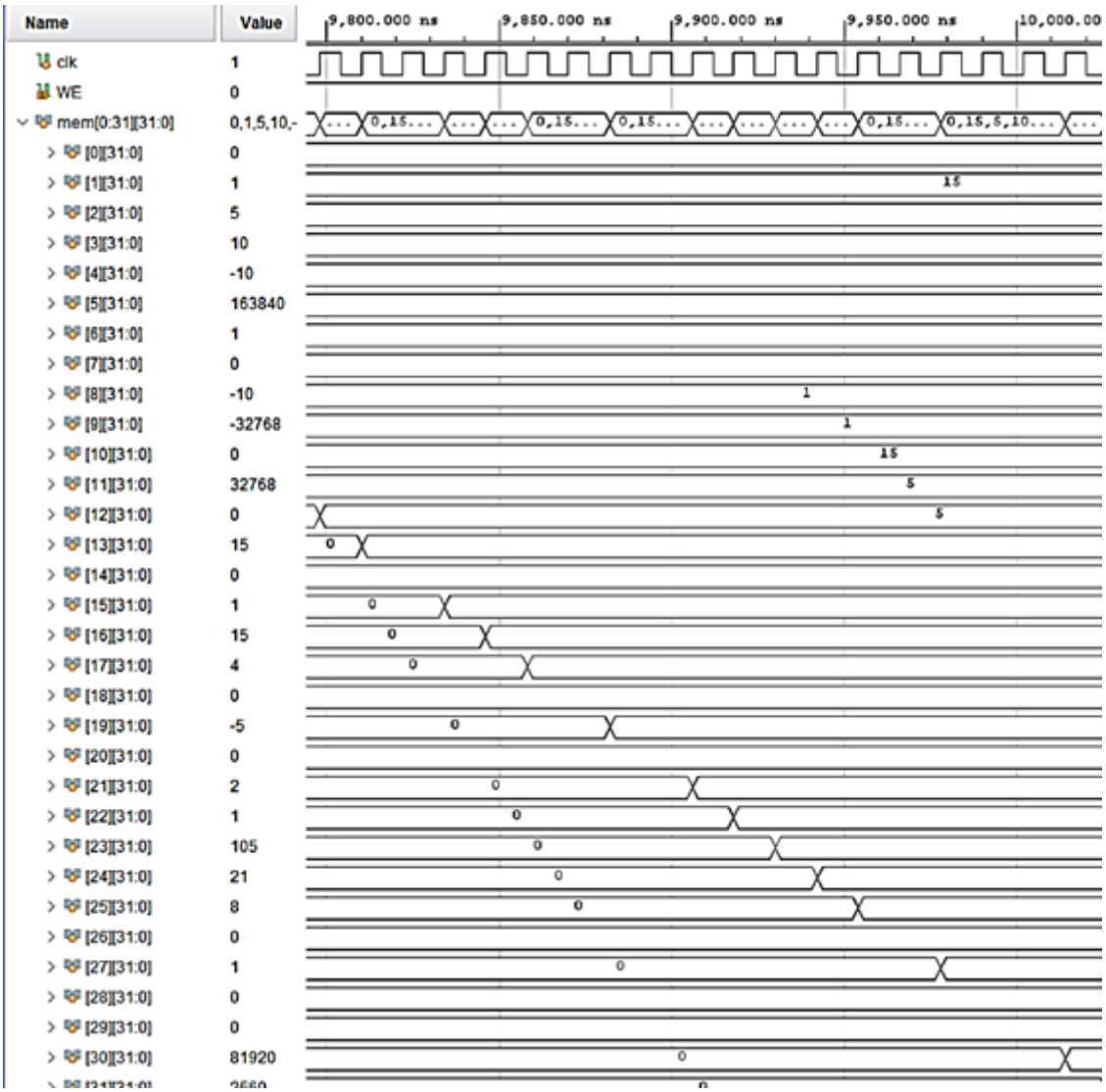

Fig. 4: Waveform of simulation results for R, I, and S instruction groups

to recover from an incorrect prediction. The resulting counts of TotalBranches and Mispredictions, together with the derived accuracy, are summarized in Table 1.

The efficacy of the eight-bit Gshare BPU was assessed by the simulation of a test program comprising several conditional branches. The findings are encapsulated in Table 1. The core executed 23,321 conditional branches with 2,234 mispredictions, corresponding to an accuracy of approximately 90.4%. The Gshare predictor thus provides a substantial improvement over simple static schemes such as Always Taken or Backward Taken Forward Not-Taken, which typically achieve only 60–70% accuracy, while still incurring modest hardware overhead. Although it is less accurate than more advanced predictors such as TAGE[15] or the perceptron predictor,[14] Gshare remains a suitable choice for lightweight RV32I pipelines where area and power are critical constraints.

Table 1: Branch prediction performance of the BPU Gshare

Parameter	Value
Total Branches	23,321
Mispredictions	2,234
Accuracy	≈ 90.4%
Misprediction Penalty	2 cycles

The accuracy rate is determined using the formula:

$$\text{Accuracy} = \left(1 \frac{\text{Mispredictions}}{\text{Total Branches}}\right) \times 100\%$$

where

- *Mispredictions* refer to the quantity of erroneous predictions.
- *TotalBranches* refers to the aggregate count of branch instructions.

The design was completely executed utilizing the RTL-to-GDSII process with Cadence tools on GPDK045 (45 nm) technology. The procedure commenced with logic synthesis utilizing Genus to transform the RTL code into an efficient gate-level netlist. The functional equivalence of this netlist was subsequently validated against the original RTL with Conformal LEC. Subsequently, the Innovus tool executed the Place and Route (PnR) procedure.

Figure 5 illustrates the design outcome following the placement (cell arrangement) and Clock Tree Synthesis phases, prior to the completion of the final routing (wiring) stage. All verification procedures, including logic equivalence check (LEC), design rule check (DRC), and layout versus schematic (LVS), were successfully completed, confirming manufacturability. Table 2 presents the postlayout findings, indicating a maximum frequency of 75 MHz, total power consumption of 15.078 mW, a cell area of roughly 0.69 mm^2, and a design density of around 69.9%. The successful DRC and LVS verifications affirm that the design adheres to physical specifications and is suitable for actual manufacturing.

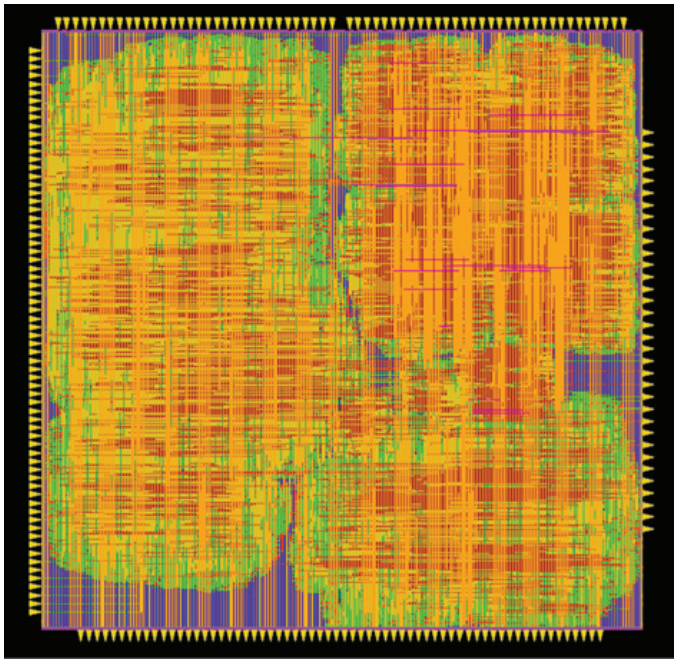

Fig. 5: Layout image of the core after placement and CTS steps

A postlayout simulation was conducted using Cadence Xcelium, employing an RC-extracted gate-level netlist to validate the design's accuracy considering realistic delays. The waveform findings presented in Figure 6 demonstrate that the output signals (DataOrReg, Value, Address, Check_address) function reliably and align with the RTL simulation, confirming that the pipeline upholds both functional and timing accuracy postphysical synthesis.

Table 3 provides a comparative analysis of the design's competitiveness against other prominent RISC-V initiatives.

In contrast to Ckristian et al., that design employed the older TSMC 130 nm technology, attaining a higher frequency of 160 MHz while consuming greater power at 26.72 mW.[12] The reduced area (0.12 mm^2) results from the utilization of SRAM macro blocks, in contrast to the existing design, which employs register-based memory for I-Mem and D-Mem—an appropriate selection for the dimensions of a microcontroller core. In contrast to Zaruba et al., the Ariane (CVA6) core is designed for the high-performance 64-bit segment, manufactured using 22 nm FD-SOI technology,[13] achieves a frequency of 1.7 GHz, and supports Linux. The present study, despite its reduced frequency, concentrates on a lightweight 32-bit microcontroller architecture, emphasizing energy economy and synthesizability. Miyazaki et al. employed

Table 2: PPA parameters of the RV32I+Gshare core (postlayout, 45 nm)

Parameter	Value
Technology	GPDK 45 nm
Max Frequency (Fmax)	75 MHz
Total Power (@75MHz)	15.078 mW
Cell Area	~0.69 mm^2
Design Density	69.90%

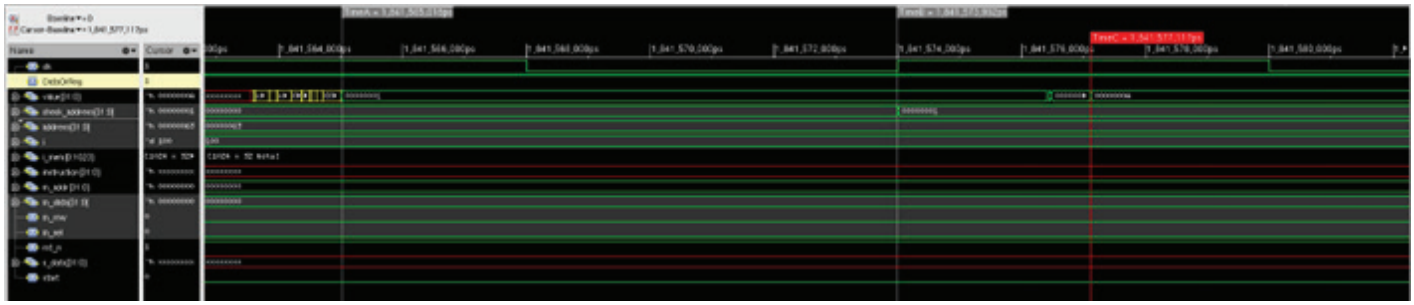

Fig. 6: Postlayout simulation waveform of the RC-extracted netlist

Table 3: Results comparison with other RISC-V designs

Parameter	This Work	[12]	[13]	[2]
ISA	RV32I (Gshare)	RV32IM	RV64G (Linux-ready)	RV32I (Two-level)
Technology	GPDK 45 nm	TSMC 130 nm	22 nm FD-SOI	Xilinx Artix-7
Frequency	75 MHz	160 MHz	1.7 GHz	150 MHz
Power	15.08 mW	26.72 mW	N/A	N/A
Area	~0.69 mm^2	0.12 mm^2	1.51 mm^2	N/A

the RVCoreP core to develop a five-stage pipeline and a two-level predictor;[2] nevertheless, their research was limited to an FPGA soft-core implementation, without physical PPA data. This research advances to the practical ASIC level, offering precise quantitative data for a Gshare-integrated RISC-V pipeline, thus addressing the deficiency of experimental data present in current studies.

Conclusion

This study has finalized the design, simulation, and ASIC implementation of a five-stage RISC-V processor, incorporating a two-level Gshare branch prediction algorithm. The findings indicate that the branch prediction algorithm functions accurately, can be physically implemented, and operates efficiently in resource-constrained settings.

The RV32I core was comprehensively built and validated, effectively replicating all 37 instructions in the foundational RV32I set, while incorporating a complete BPU that includes a BTB, GHR, and PHT with 2-bit saturating counters. The design attained over 90% branch prediction accuracy, a postlayout operating frequency (Fmax) of approximately 75 MHz, total power consumption of about 15.078 mW, a core size of approximately 0.69 mm^2, and a cell density of approximately 69.9%, indicating a proficient equilibrium between performance and hardware resources.

This work's focal point is the execution of the comprehensive ASIC design process from RTL to GDSII utilizing the Cadence Genus, Conformal, and Innovus toolchain on GPDK045 (45 nm) technology. The design successfully passed LEC, DRC, and LVS checks, confirming the accuracy of the RTL description and the physical layout. This initiative represents a substantial advancement beyond projects limited to simulation or FPGA, confirming the Gshare two-level branch prediction algorithm's complete feasibility for implementation in actual RISC-V microcontrollers. This study has shown that incorporating a two-level Gshare algorithm into an RISC-V pipeline is totally viable, both functionally and physically, paving the way for the development of compact, energy-efficient, and more intelligent RISC-V microprocessors in the future.

Acknowledgement

This research was supported by the VNUHCM University of Information Technology's Scientific Research Support Fund.

References

1. Waterman, A., Lee, Y., Asanović, K., & Patterson, D. "The RISCV Instruction Set Manual. Volume 1: User-level ISA, version 2.0," UC Berkeley Dept of EE and CS, Tech. Rep., 2014.
2. Miyazaki, H., et al. (2020). RVCoreP: an optimized RISC-V soft processor of five-stage pipelining. IEICE Transactions on Information and Systems, 103(12), 2494–2503.
3. Schiavone, P. D., et al. (2021). Arnold: an eFPGA-augmented RISC-V SoC for flexible and low-power IoT end nodes. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 29(4), 677–690.
4. SiFive, Inc. SiFive E31 Core Complex Manual v1p2, 2024 https://static.dev.sifive.com/E31-RISCVCoreIP.pdf (accessed on December 10, 2024).
5. Yeh, T.-Y., & Patt, Y. N. (1991). Two-level adaptive training branch prediction. In: Proceedings of the 24th annual international symposium on Microarchitecture.
6. McFarling, S. *Combining branch predictors*. Vol. 49. Technical Report TN-36, Digital Western Research Laboratory, 1993.
7. Asanovic, K., et al. (2016). The rocket chip generator. EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2016-17 4.
8. Celio, C., et al. (2017). BOOMv2: an open-source out-of-order RISC-V core. In: First Workshop on Computer Architecture Research with RISC-V (CARRV).
9. Jin, Z., Di, H., Hu, T., & Wang, P. (2025). Real-Time Optimization of RISC-V Processors Based on Branch Prediction and Division Data Dependency. *Applied Sciences*, 15(2), 632. https://doi.org/10.3390/app15020632
10. Kim, J., Jang, H., & Shin, Y. (2025). A survey of side-channel attacks on branch prediction units. ACM Computing Surveys, 57(11), 1–36.
11. Kocher, P., et al. (2020). Spectre attacks: exploiting speculative execution. Communications of the ACM, 63(7), 93–101.

12. Duran, C., et al. (2017). A system-on-chip platform for the internet of things featuring a 32-bit RISC-V based microcontroller. In: 2017 IEEE 8th Latin American Symposium on Circuits & Systems (LASCAS). IEEE.
13. Zaruba, F., & Benini, L. (2019). The cost of application-class processing: energy and performance analysis of a Linux-ready 1.7-GHz 64-bit RISC-V core in 22-nm FDSOI technology. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 27(11), 2629–2640.
14. Jiménez, D. A., & Lin, C. (2001). Dynamic branch prediction with perceptrons. In: Proceedings HPCA Seventh International Symposium on High-Performance Computer Architecture. IEEE.
15. Seznec, A., & Michaud, P. (2006). A case for (partially) tagged geometric history length branch prediction. The Journal of Instruction-Level Parallelism, 8, 23.