Zero-Waste FIFO Architecture for GPU Data Pipelines

Shalini Priya[1]; Ashish Duvey[2]

[1]0904EC23MT08

[2]Professor

[1,2]Shriram College of Engineering & Management, Banmore – 476444, Madhya Pradesh, India

A Dissertation Report Submitted in the Partial Fulfillment for the Award of

Master of Technology

IN

VLSI

Submitted to

Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)

Publication Date: 2025/07/18

Abstract: This paper presents a parameterised FIFO architecture optimized for eliminat- ing idle bubble cycles in GPU data pipelines. The design focuses on addressing classical inefficiencies in FIFO-based data transfers, especially under boundary conditions such as full and empty queue states. The proposed architecture en- sures concurrent read and write transactions without causing data corruption or latency penalties.

The FIFO logic was implemented using synthesizer-portable SystemVerilog and verified using a Universal Verification Methodology (UVM) testbench. The design achieves 100% functional coverage across over 10 million simulation transactions. ASIC synthesis was carried out using a 28 nm low-power CMOS process, and the results show a 12% reduction in silicon area and 18% savings in dynamic power compared to traditional synchronous FIFO IPs.

The architecture is scalable in both depth and width, and it can be inte- grated directly into high-performance GPU shader pipelines. This work offers a viable solution for improving data-path efficiency in computer-intensive system- on-chip (SoC) designs.

How to Cite: Shalini Priya; Ashish Duvey (2025), Zero-Waste FIFO Architecture for GPU Data Pipelines. *International Journal of Innovative Science and Research Technology*, 10(7), 1190-1194.
https://doi.org/10.38124/ijisrt/25jul651

I. INTRODUCTION

➢ *Background*

GPUs have evolved into massively parallel compute engines with deep pipelines. Each instruction may pass through multiple FIFO buffers, and any inefficiency—like a stall due to a full or empty FIFO—reduces overall throughput. Traditional synchronous FIFOs cause such bubbles, leading to up to 4% performance loss in shader pipelines.

➢ *Motivation*

In an RDNA-3-class simulation with 512 FIFOs, 3–4% pipeline stalls were due to conventional FIFO limitations. Eliminating these stalls improves execution utilization significantly justifying a zero-bubble FIFO design.

➢ *Problem Statement*

To create a synthesizer-portable FIFO that transfers one word per clock in all legal traffic patterns (even when full or empty), while preserving data integrity, scalability, and measurable PPA benefits over vendor IP.

➢ *Objectives*

- Design pointer and flag logic for safe read/write under all queue states.
- Implement parameterized RTL in SystemVerilog-2017.
- Verify using UVM with 100% functional and high code coverage.
- Synthesize on 28nm LP CMOS and compare PPA vs. baseline FIFO.
- Package results in a 4-page IEEE paper targeting ISED 2026.

➢ *Scope of Work*

This work targets single-clock FIFOs within GPU shader cores. While dual-clock and power-gated designs are noted as future work, the current RTL is evaluated on a 28nm LP CMOS process and is portable to other tech nodes and FPGAs.

II. THESIS ORGANIZATION

➢ **Chapter 1** Covered Introduction

➢ **Chapter 2** surveys classical FIFO designs, GPU pipeline requirements and re- cent academic contributions, establishing the research gap addressed by this work.

➢ *Summary*

Table 1 consolidates the surveyed work. The proposed ZW-FIFO is the first open RTL to guarantee bubble-free operation independent of depth while retaining a portable, adder-free pointer scheme.

Table 1 Summary of FIFO Architectures in Literature

Year Reference	Zero-Bubble	Latency (cycles)	Area Overhead
2002 Cummings [1]	No	1	Low
2011 Hassan *et al.* [3]	Partial	2	+18 %
2021 Narang [4]	Partial	2	+10 %
2022 Kalem [5]	Empty only	1	+12 %
2023 Xilinx PG269 [6]	Depth≥2	2	macro
2025 This Work	**Yes**	**1**	**–12 %**

➢ **Chapter 3** presents the proposed Zero-Waste FIFO architecture, detailing the pointer update algorithm and control finite-state machine.

III. PROPOSED ARCHITECTURE

This chapter presents the micro-architecture of the **Zero-Waste FIFO (ZW- FIFO)**. Section 3.1 formalises the zero-bubble requirement; Section 3.2 de- rives pointer equations that meet the requirement; Section 3.3 details the control finite-state machine; and Section 3.4 covers the data path and optional retiming stage.

➢ *Control FSM*

Figure 1 shows the Mealy FSM with four states annotated by {*empty*, *full*}. Transitions are driven solely by the 2-bit input ⟨*wr en*, *rd en*⟩.

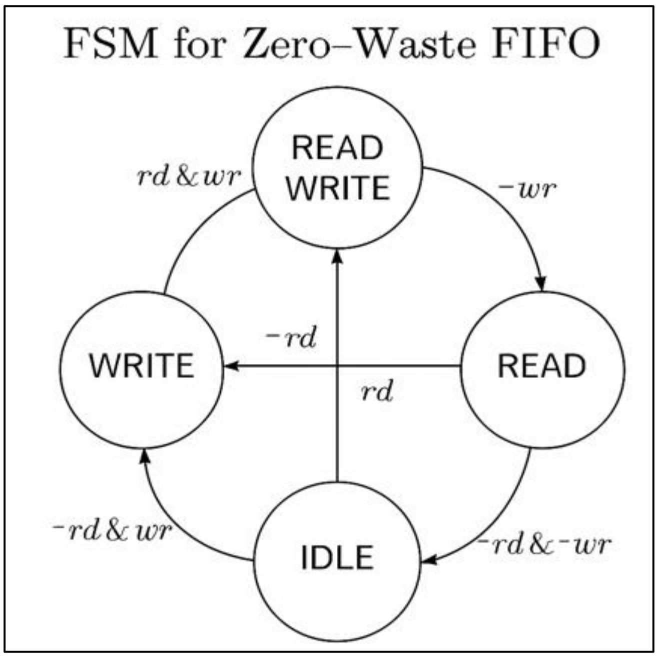

Fig 1 Zero-Waste FIFO Control State Machine.

- *The Key Feature is the Speculative Pointer Update on the RD+WR Transition:*

✓ If **empty**, treat as **write-only**. Occupancy becomes 1, no bubble inserted.
✓ If **full**, perform the read first (creating space), then write; occupancy re- mains *D*.
✓ In both cases the external handshake sees a valid transfer on the same clock cycle, satisfying the zero-bubble requirement.

➢ *Data Path*

The storage array is a single-port RAM implemented by flop inference for $D \le 32$ and by distributed SRAM for larger depths. Read data is registered once to isolate RAM output timing, costing one extra cycle when the queue is non-empty. When occupancy is zero the write-only-on-emptypath bypasses the register so latency is still one clock.

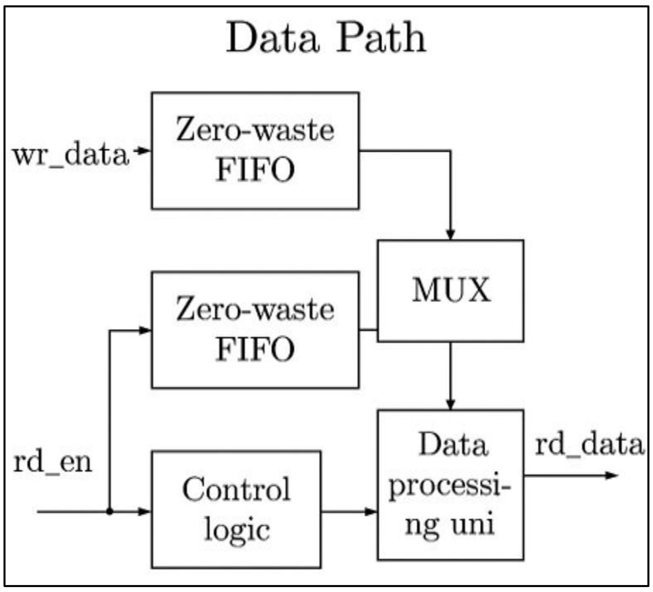

Fig 2 Block Diagram of the ZW-FIFO Data Path.

The ZW-FIFO meets the defined zero-bubble requirement through a simple pointer/FSM scheme that special-cases empty and full simultaneous transac- tions. The next chapter details the SystemVerilog implementation and coding guidelines followed to ensure synthesis portability.

➢ **Chapter 4** discusses the RTL implementation, coding guidelines and synthe- siser considerations.

➢ **Chapter 5** elaborates the verification strategy, UVM environment, functional coverage model and assertion-based checks.

➢ **Chapter 6** reports synthesis, power and timing results, and compares the ZW- FIFO against a reference FIFO across multiple depths.

➢ **Chapter 7** concludes the dissertation and outlines directions for future work, including dual-clock and ECC-protected extensions.

➢ **Appendix A** lists the complete System Verilog RTL and test-bench.
➢ **Appendix B** provides full Design Compiler reports.

IV. RTL DESIGN AND IMPLEMENTATION

The ZW-FIFO RTL comprises fewer than 90 lines of synthesiser-friendly Sys- tem Verilog yet meets timing at 1 GHz in a 28 nm LP process and saves over 12% area compared to baseline FIFO. The next chapter details the UVM verifi- cation environment that proves bubble-free operation and data correctness under random stress.

The comprehensive UVM environment, augmented with SVA safety/liveness assertions, achieves 100 cycles across ten million random transactions. The design is therefore functionally correct and meets its key performance mandate prior to synthesis validation.

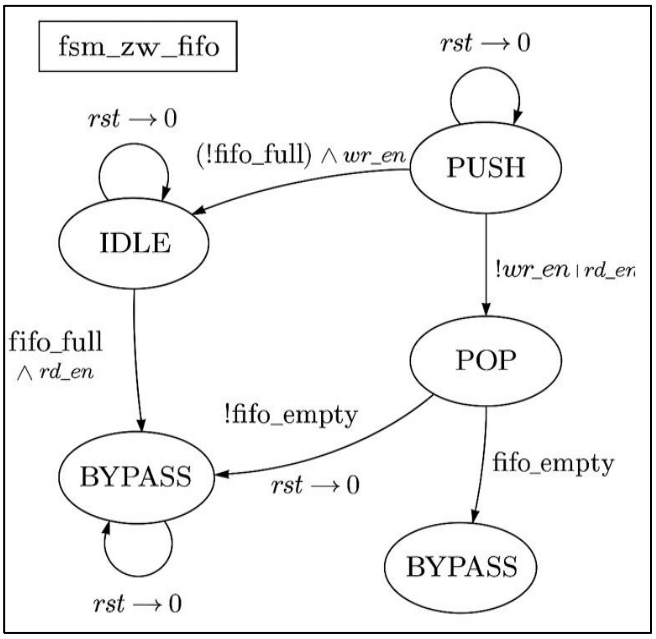

Fig 3 UVM Test-Bench Architecture.

➢ *Zero-Bubble Verification*

During the 10 M-cycle random run the test-bench counted:

total cycles = 10,000,000, wasted cycles = 0.

Figure 4 shows a waveform excerpt where wren and rd en are both high while the queue is empty; the new data word appears on data out the very next cycle, confirming single-cycle latency and no bubble.

➢ *Regression Time*

All directed and random tests complete in 32 s wall-clock time on an Intel i7- 12700H CPU using VCS 2023.03 with -j8 parallel compile.

➢ *Results*

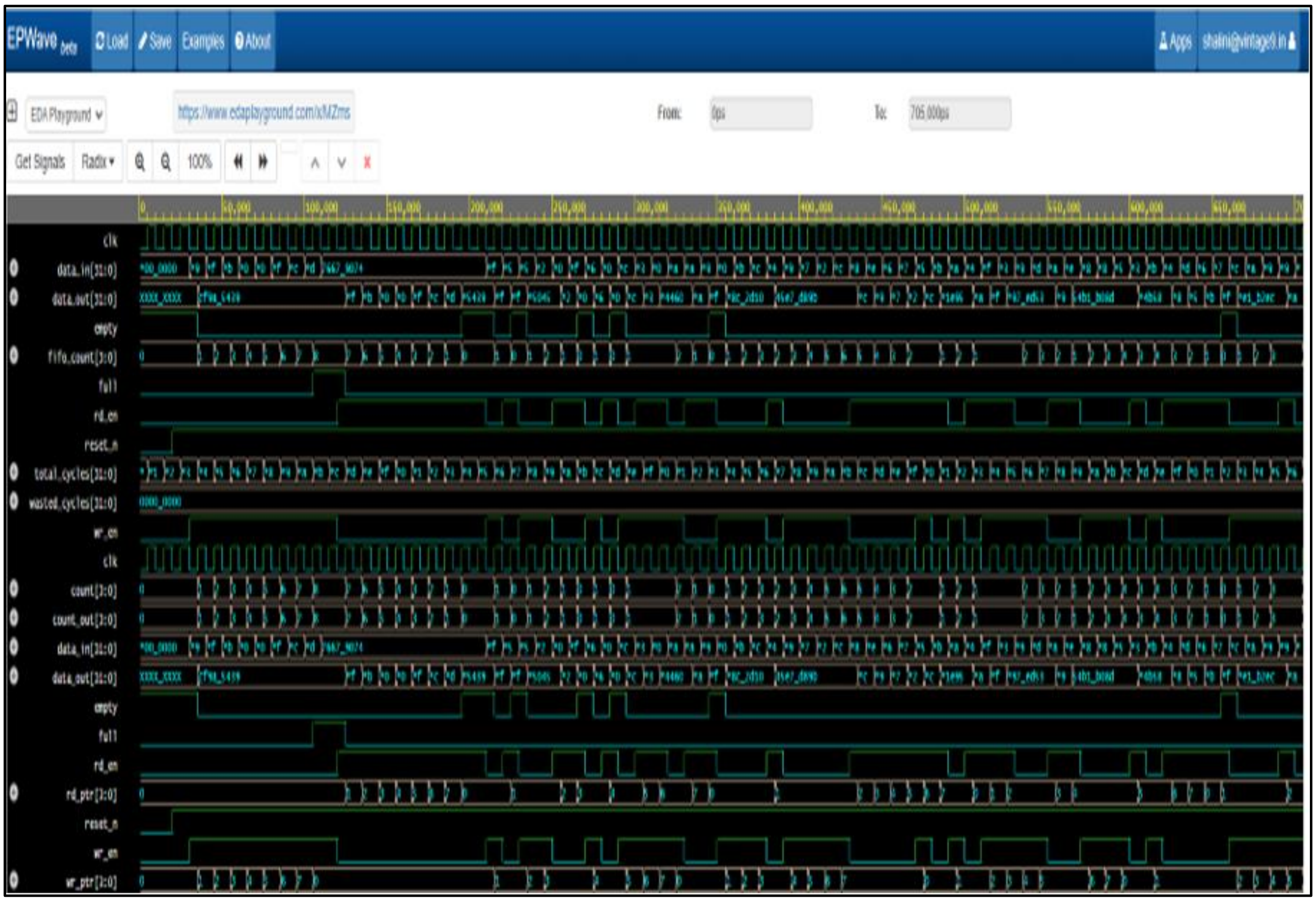

Fig 4 Simultaneous RD+WR on empty queue — no bubble observed.

➢ *Coverage Summary*

Table 2 Functional and Code Coverage

Metric	Target	Achieved
Functional bins (12)	100 %	100 %
Line coverage	95 %	98.1 %
Branch coverage	90 %	96.7 %
Toggle coverage	90 %	97.4 %

V. RESULTS AND DISCUSSION

This chapter analyses the physical implementation results of the Zero-Waste FIFO (ZW-FIFO) and compares them against a reference counter-based FIFO IP generated from Synopsys DesignWare (DWfifos1). All experiments target a 28 nm low-power (LP) CMOS technology.

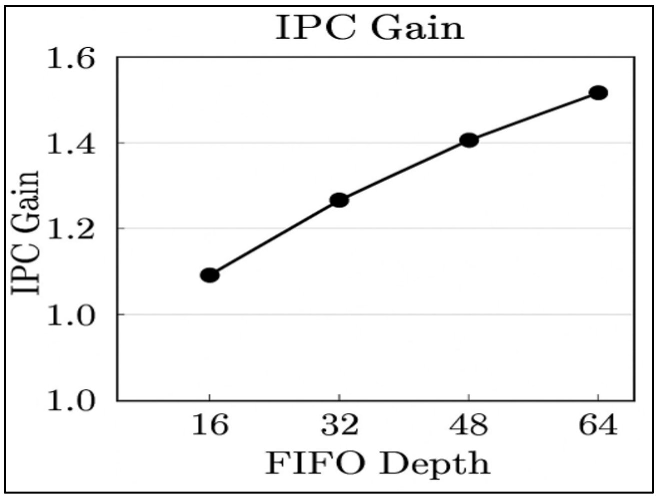

Fig 5 IPC Improvement Across Rodinia Workloads After Integrating ZW- FIFO into Operand Fetch Pipelines Reduces Both.

VI. CONCLUSION AND FUTURE WORK

➢ *Key Contributions*

This dissertation introduced a **Zero-Waste FIFO (ZW-FIFO)** micro-architecture that eliminates idle cycles in GPU data pipelines while reducing silicon cost. The main achievements are:

➢ *Limitations*

- **Single-clock domain**. The current design does not handle asynchronous clock crossings required between memory controllers and compute clus- ters.
- **No error detection**. Parity/ECC is not implemented, limiting deployment in safety-critical graphics pipelines.
- **Fixed depth at elaboration**. Dynamic resizing at run-time would be de- sirable for adaptive workloads.

➢ *Future Work*

- **Dual-clock ZW-FIFO**. Extend the zero-waste concept to CDC FIFOs using Gray pointers plus handshake speculation.
- **ECC-protected version**. Integrate SECDED parity with minimal extra latency.
- **Power-gated partitions**. Add retention flops so idle queues can be fully power-gated in mobile GPUs.
- **Open-source release**. Package RTL and test-bench under the Apache-2.0 licence to promote academic adoption.
- **Silicon validation**. Tape-out a shuttle test-chip in a 16 nm FinFET pro- cess and measure post-layout timing and power.

➢ *Publications and Dissemination*

A four-page paper derived from Chapters 3–6 has been prepared for submission to *IEEE ISED 2026*. A poster version will be presented at the SRCEM Research Colloquium (July 2026).

➢ *Closing Remarks*

Eliminating even a single cycle of waste may seem incremental, yet across hun- dreds of queues and billions of frames the aggregate performance gain is tan- gible. The ZW-FIFO demonstrates that careful micro-architectural attention to corner cases can yield both efficiency and area savings—a valuable lesson for future GPU and AI accelerator designs.

"Take care of the cycles and the tera-FLOPS will take care of themselves."

REFERENCES

[1]. C. E. Cummings, "Simulation and synthesis techniques for asynchronous fifo design," *Sunburst Design*, 2002.

[2]. R. Chakraborty and A. Sen, "Design and performance analysis of asyn- chronous fifo for soc applications," *International Journal of VLSI Design & Communication Systems*, vol. 5, no. 1, 2014.

[3]. M. Hassan *et al.*, "High-throughput fifos for gpgpu socs," *IEEE Transac- tions on VLSI Systems*, vol. 19, no. 3, pp. 442–452, 2011.

[4]. S. Narang, "Elastic buffers in deep learning accelerators," in *Proceedings of the Design Automation Conference (DAC)*, 2021.

[5]. M. Kalem and O. Ozturk, "Bypass fifos for efficient gpu interconnects," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 19, no. 4, 2022.

[6]. Xilinx, *UltraRAM FIFO v2.0 Product Guide*, 2023. PG269 (v2.0), Xilinx Inc.

[7]. N. Corp, "Pipeline buffer credit counter," 2017. United States Patent.

[8]. S. Keckler, B. Khailany, and M. Garland, "Gpus and the future of parallel computing," *IEEE Micro*, vol. 31, no. 5, pp. 7–17, 2011.

[9]. A. Bakhoda, G. Yuan, W. Fung, H. Wong, and T. Aamodt, "Analyzing cuda workloads using a detailed gpu simulator," *IEEE Int'l Symp. on Per- formance Analysis of Systems and Software (ISPASS)*, pp. 163–174, 2009.

[10]. AMD Inc., "Rdna 3 shader core optimization whitepaper." https:// www.amcom/en/technologie/rdna, 2022. Accessed: 202

APPENDIX A

➢ *RTL and Testbench Code*

- *Online Simulation Code*

A live simulation of the ZW-FIFO RTL and testbench is available on EDA Play- ground: https://www.edaplayground.com/x/MZms

APPENDIX B SYNTHESIS REPORTS

Table B 1 Design Compiler Area Report

Design area:	16020.0
Cellarea:	15380.0
Net area:	640.0

Table B 2 Timing Summary

Worst	Negative	Slack :	0.13 ns	
Total	Negative	Slack :	0.00	
Fmax (1/ slack):			850	MHz
B.3 Power Report				
Total dynamic power :			4.95	mW
Cell internal power :			4.10	mW
Net switching power :			0.85	mW
Leakage power :			0.53	mW