"Design and Implementation of Burst FIFO"

Sathwika Katkam
Dept. of ECE
Student of CBIT(A)
Hyderabad, India

Chandrasekhar E
Dept. of ECE
CBIT(A)
Hyderabad, India

Chandra Sekhar P
Dept. of ECE
CBIT(A)
Hyderabad, India

Neeraja B
Dept. of ECE
CBIT(A)
Hyderabad, India

Abstract*—In high-performance digital systems, efficient data buffering is critical to maintaining throughput and minimizing latency. Standard First-In First-Out (FIFO) memory buffers typically support single-word data transfers per clock cycle, which can limit performance in applications demanding high-speed data movement. This paper presents the design and implementation of a Burst FIFO architecture capable of transferring multiple data words in a single burst operation, significantly increasing data throughput. Unlike conventional FIFOs, the proposed Burst FIFO supports programmable burst sizes, allowing flexible and efficient handling of varying data loads. It also incorporates features such as clock domain crossing support and robust mechanisms to prevent overflow and underflow. This makes the architecture highly suitable for bandwidth-intensive applications such as image processing pipelines, high-speed serial interfaces, and memory subsystem integration. By enabling burst-mode transactions, the system reduces control overhead and improves overall efficiency. Simulation results confirm the correctness and performance of the design, showing that it meets the requirements of modern digital designs where traditional FIFOs fall short. The Burst FIFO offers a compelling solution for systems where transferring large blocks of data in fewer cycles is essential to achieving optimal performance.

***Keywords*—Throughput,Bandwidth-intensive,Image processing,Serial interfacing,Overflow and Underflow,Subsystem integration.**

I. INTRODUCTION:

Efficient and reliable data buffering is essential in modern digital systems, particularly when components operate at different speeds or across varying clock domains. The First-In First-Out (FIFO) memory buffer is widely used to temporarily store data and manage these timing differences. However, in high-speed applications involving large volumes of data, traditional single-word FIFO operations can become a performance bottleneck. To overcome this limitation, a Burst FIFO is introduced, which extends the conventional FIFO by supporting burst mode operations—allowing multiple data words to be written or read in a single, continuous sequence. This approach enhances data throughput, minimizes control overhead, and reduces latency. Burst FIFOs are especially effective in applications such as video processing, digital signal processing, and memory interface bridging, where data is commonly handled in blocks. This work focuses on the design of a Burst FIFO that supports programmable burst sizes, ensures data integrity across clock domains, and incorporates efficient control logic. The design is validated through simulation and analysis, demonstrating its suitability for high-speed, real-time digital systems.

II. LITERATURE SURVEY:

Cummings [1] proposed Simulation and Synthesis Techniques for Asynchronous FIFO Design and the summary of the paper includes advanced simulation and synthesis methods for designing asynchronous FIFO circuits. The focus is on addressing challenges such as timing mismatches and data integrity in systems where data timing is not synchronized. The paper demonstrates how specialized tools and strategies can optimize asynchronous FIFO performance and reliability. These techniques improve circuit efficiency, minimize the need for complex clocking mechanisms, and help reduce power consumption, making them suitable for real-time and high-performance applications operating across different timing domains.Chang et al [2] proposed A Robust Ultra-Low Power Asynchronous FIFO Memory

with Self-Adaptive Power Control and the summary of the paper includes the implementation of a self-adaptive power control mechanism to enhance the energy efficiency of asynchronous FIFO memory systems. The design dynamically adjusts power consumption based on system workload, ensuring low energy usage without sacrificing performance. The FIFO memory architecture emphasizes robustness and is optimized for low-power applications such as portable devices. Experimental results show significant power reduction compared to conventional FIFO designs, demonstrating that integrating adaptive power control can effectively improve both energy savings and overall system performance.Tjia [3] proposed Comparison of Efficiency Between FIFO and LIFO Methods for Food and Beverage Business Sector, and the summary of the paper includes an evaluation of the effectiveness of First-In-First-Out (FIFO) and Last-In-First-Out (LIFO) inventory methods within the food and beverage industry. The study reveals that FIFO is more efficient for managing perishable goods, ensuring the utilization of older stock first to reduce waste and maintain freshness. FIFO also aligns with health regulations and enhances product quality. Conversely, the LIFO method, which prioritizes the use of newer stock, is less effective in preventing spoilage and maintaining quality in the food and beverage sector. While LIFO may offer financial advantages in specific contexts, such as cost fluctuation scenarios, it is generally unsuitable for perishable goods due to the risk of stock obsolescence. The paper concludes that FIFO is the optimal inventory management method for businesses prioritizing quality, compliance, and operational efficiency. LIFO, on the other hand, may be employed strategically for financial benefits but requires careful consideration of its limitations regarding perishability.Yadlapati and Kakarla [4] proposed Design and Verification of Asynchronous FIFO with Novel Architecture Using Verilog HDL, and the summary of the paper includes the development of an asynchronous FIFO memory structure aimed at enhancing data communication between different clock domains. The novel architecture introduces an internal signal mechanism to determine full and empty states, eliminating the need for separate status bits, thereby simplifying control logic and improving reliability. Implemented using Verilog HDL, the design underwent simulation-based verification, demonstrating reduced latency and increased throughput. The results indicate that this architecture effectively addresses synchronization challenges, making it suitable for high-speed, clock-independent communication in embedded and communication systems.Zhang [5] proposed Optimizing Data Transfer Across Asynchronous Clock Domains: A Comprehensive Approach to Asynchronous FIFO Circuit Design, and the summary of the paper includes the development of an asynchronous FIFO circuit aimed at enhancing data transfer between asynchronous clock domains. The design employs Gray code for pointer synchronization and robust control structures to address challenges like metastability and timing mismatches. Simulation results demonstrate reduced latency and improved data integrity, indicating the design's effectiveness in facilitating reliable communication across varying clock frequencies.

III. METHODOLOGY USED:

Burst means writing or reading into multiple locations at a single point of time. As per the requirement the burst_write number and burst_read number is decided.

A burst FIFO (First-In-First-Out) is a type of memory buffer used in digital systems to handle bursts of data efficiently. It allows multiple data words to be written or read in a single operation, rather than one at a time. This is particularly useful in applications where data transfer occurs in bursts, such as in communication systems or high-speed data processing.

Implementing a burst FIFO involves careful design to efficiently handle data bursts while maintaining the FIFO's fundamental characteristic: first-in, first-out behavior. Here's a high-level breakdown:

1. Memory Array

- A memory array is the core of the FIFO, often implemented using RAM.
- It stores the incoming data in sequential locations, ensuring the order of data remains intact.

2. Write and Read Pointers

- Write Pointer: Keeps track of where the next data item should be written.
- Read Pointer: Tracks the location of the next data item to be read.
- These pointers wrap around when they reach the end of the memory array, ensuring continuous operation.

3. Burst Control Logic

- This logic enables burst operations. For instance, when a burst write happens, the control logic ensures multiple words are written sequentially in one cycle (or a few cycles).
- Similarly, during a burst read, multiple words are read out in sequence.

4. Flags and Status Signals

- Status signals like empty, full, and almost full help manage the FIFO's operation:
  - Full: Indicates the FIFO cannot accept more data.

- Empty: Indicates no data is available to read.
- Almost Full: Alerts the system to slow down or stop writing data.

5. Clocking

- Burst FIFOs often operate in systems with different clock domains (e.g., data produced at a higher clock rate than consumed). In such cases, dual-clock FIFOs are used to synchronize data transfer between these domains.

6. Pipeline Registers (Optional)

- Pipeline registers may be added to ensure high-speed operation, reducing delays during burst transfers.

7. Burst Size Configuration

- The implementation often includes registers or control signals to configure the burst size (e.g., 4, 8, or 16 words).

BLOCK DIAGRAM WITH INPUTS AND OUTPUTS

If the write pulse is HIGH for one active clock edge, and the burst_write is 4, then 4 consecutive locations of the FIFO are written and in the same way burst_read keeps reading data.In Burst, the write/read pointer address is updated according to the burst_write/burst_read size.

FIFO full: When unused FIFO locations are less than burst_write size, then full is HIGH.

FIFO empty: When used locations are less than the burst_read size

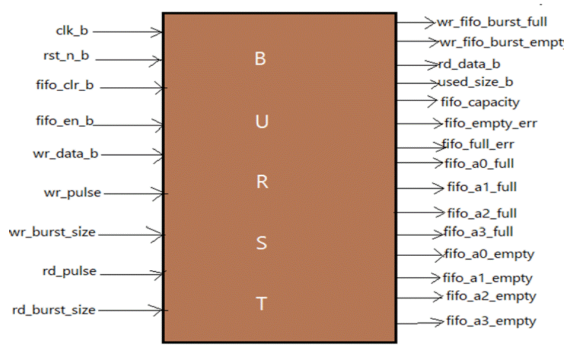

Fig. 1.Block diagram of burst fifo

Used size: It increases when FIFO is written, the used size increases by burst_write size. And decreases by burst_read size when FIFO is read.

IV. CONCLUSION:

The Burst FIFO design provides an effective solution for high-throughput, low-latency data buffering in modern digital systems. Its ability to handle burst-mode operations enhances performance over traditional FIFO designs, especially in block-based data applications. The system's support for programmable burst sizes, clock domain crossing, and robust control logic ensures versatility and reliability. Simulation results confirm the design's functionality and stability. Overall, the Burst FIFO significantly improves data transfer efficiency in high-speed systems such as image processing and communication interfaces.

V. RESULTS:

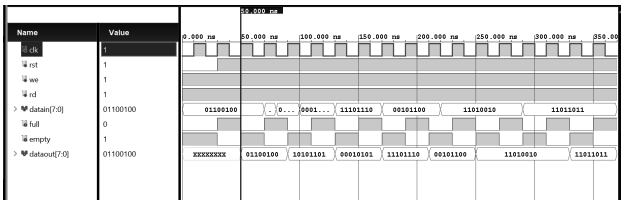

Fig. 2.Simulation result of fifo

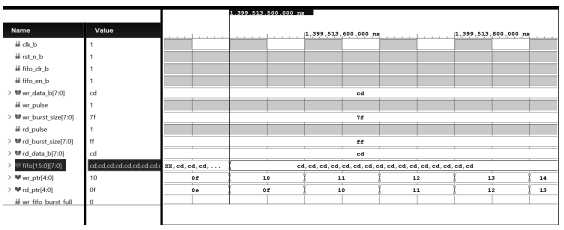

Fig. 3.Simulation result of burst fifo

Table. 1.Comparison table of fifo and burst fifo

Sl no:	Features	FIFO	Burst FIFO
1	Data Access	One word per clock cycle	Multiple words per clock cycle
2	Throughout	Moderate	Higher due to burst transfer
3	Area Utilization	Lower	Higher
4	Power Consumpti	Lower	Higher due to the burst

	on		loads
5	Memory Utilization	Not fully utilized	Efficiently utilized with burst capability
6	Use Case	Suitable for continuous, small data flow	Ideal for high speed,block data transfer

VI. REFERENCES:

[1] C. E. Cummings, "Simulation and synthesis techniques for asynchronous FIFO design," *Proc. Synopsys Users Group (SNUG) Conf.*, San Jose, CA, USA, pp. 89–102, 2002.

[2] M.-T. Chang, P.-T. Huang, and W. Hwang, "A robust ultra-low power asynchronous FIFO memory with self-adaptive power control," *Proc. IEEE Int. SoC Conf.*, USA, pp. 1–6, Sep. 2008.

[3] F. A. Tjia, "Comparison of efficiency between FIFO and LIFO methods for food and beverage business sector," *Proc. Faculty of Economic and Business Conf.*, Atma Jaya Makassar Univ., Makassar, Indonesia, pp. 1–10, 2023.

[4] A. Yadlapati and H. K. Kakarla, "Design and verification of asynchronous FIFO with novel architecture using Verilog HDL," *J. Eng. Appl. Sci.*, vol. 14, no. 1, pp. 159–163, 2019.

[5] X. Zhang, "Optimizing data transfer across asynchronous clock domains: A comprehensive approach to asynchronous FIFO circuit design," *Proc. Highlights in Science, Engineering and Technology*, ESAC 2024, Wenzhou, China, vol. 87, pp. 31-36, 2024.