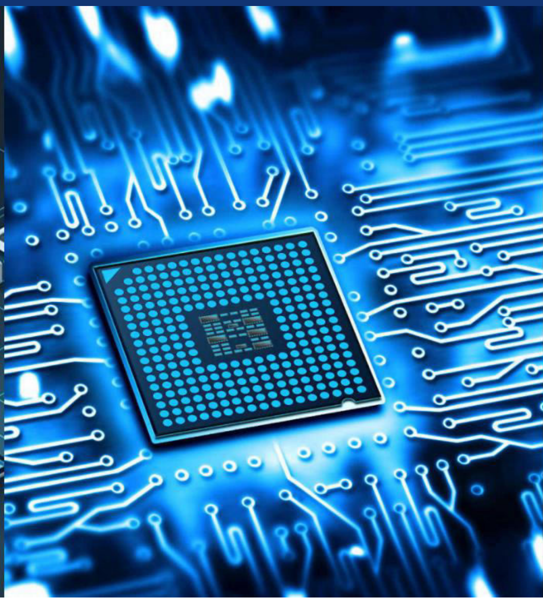
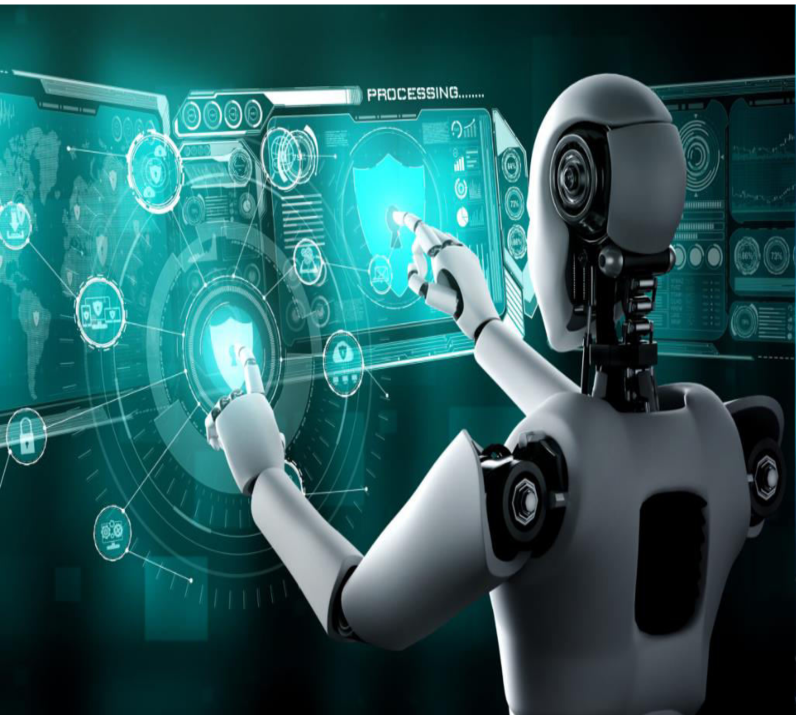
IJIRCCE

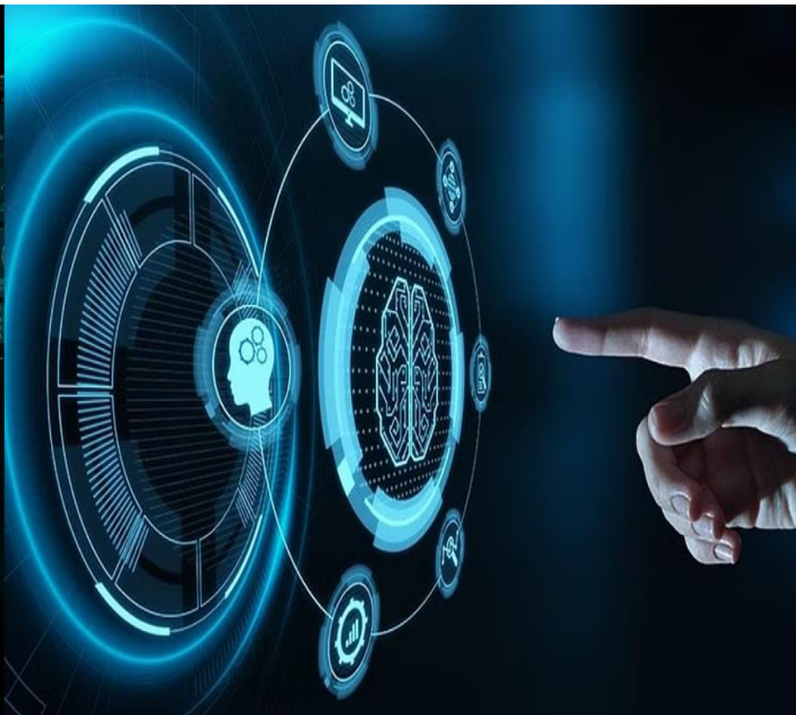
ISSN(O): 2320-9801
ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





Impact Factor: 8.771

Volume 14, Issue 3, March 2026

© 2026 IJIRCCE | Volume 14, Issue 3, March 2026 DOI: 10.15680/IJIRCCE.2026.1403120

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Design and Verification of Asynchronous FIFO

K.Lokesh[1], K.Renuka[2], K.Sirisha[3], V.N.Bhargava Krishna[4], Mrs.Karuna Gone[5]

U.G. Student, Department of ECE, SVIET Engineering College, Nandamuru, Pedana, Andhra Pradesh, India[1,2,3,4]

Assistant Professor, Department of ECE, SVIET Engineering College, Nandamuru, Pedana, Andhra Pradesh, India[5]

ABSTRACT: With the rapid development of integrated circuits, asynchronous First Input First Output (FIFO) is often used to solve the problem of data transmission across the clock domain. This paper mainly studies the key problem of asynchronous FIFO design - the generation of empty - full signal. To solve this problem, it is necessary to realize the synchronization of signal across the clock domain and convert binary code into gray code to reduce the probability of metastable state. The null and full signals generated by the asynchronous FIFO designed in this paper are false null and false full, but this does not affect the function of the asynchronous FIFO, and will only lose part of the performance. Through Modelsim simulation verification, the designed asynchronous FIFO can realize first-in, first-out of data and correctly generate empty and full signal, which meets the design requirements. The research of this paper is helpful for further application of asynchronous FIFO in data transmission across clock domains

KEYWORDS: Asynchronous FIFO; Metastable state; Empty - full signal; Cross clock domain.

I. INTRODUCTION

With the rapid development of integrated circuits, modern digital systems often adopt multi-clock domain design in order to improve the performance. Signals that cross the clock domain can generate metastases when they are transmitted. Asynchronous FIFO can effectively solve the metastable phenomenon of data transmission and storage across the clock domain [1]. To solve the above problems, this paper presents the research problem of how to generate empty full signal with asynchronous FIFO on the premise of reducing the probability of metastable state. This paper first introduces the four basic modules of asynchronous FIFO and the key problem of asynchronous FIFO design -- how to generate empty and full signal, and then carries on the Verilog design of the asynchronous FIFO, and finally carries on the simulation and result analysis of the designed asynchronous FIFO, and further analyzes the significance of empty and full signal. The research in this paper is helpful to the further development of asynchronous FIFO and makes a contribution to the application of asynchronous FIFO in signal transmission across the clock domain.

II. RELATED WORK

In earlier studies, FIFO (First-In First-Out) memory systems were mainly designed using synchronous techniques where both read and write operations were controlled by a single clock signal. While this approach was simple and effective for basic systems, it became inadequate for modern digital designs that operate across multiple clock domains. The major challenge in such systems is metastability, which occurs when signals pass between different clock domains and may lead to unpredictable behavior. To overcome this issue, researchers introduced asynchronous FIFO architectures that allow independent read and write operations using separate clocks. Several improvements were made in this area, including the use of dual-port RAM to enable simultaneous access and Gray code conversion to minimize errors caused by multiple bit transitions. Although these techniques improved reliability, earlier methods still faced issues such as increased hardware complexity, synchronization delays, and inefficient signal generation.

The asynchronous FIFO system is designed to enable safe and efficient data transfer between two different clock domains. Unlike conventional memory systems, FIFO does not require external addressing because it automatically manages data using internal pointers. The architecture mainly consists of four essential modules: dual-port RAM for data storage, write control module, read control module, and clock synchronization module. The dual-port RAM allows both read and write operations to occur simultaneously, improving system performance. The write control module manages the writing of data based on the write clock and ensures that data is not written when the memory is full. Similarly, the read control module manages data retrieval based on the read clock and prevents reading when the memory is empty. The clock synchronization module plays a crucial role in transferring pointer information safely between clock domains, thereby avoiding errors.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

III. METHODOLOGY

The methodology of the proposed design involves implementing the asynchronous FIFO using Verilog HDL with a focus on reliability and efficiency. The system uses two pointers, namely the write pointer and the read pointer, to keep track of memory locations for data storage and retrieval. These pointers are initially in binary form but are converted into Gray code to reduce the chances of metastability, since Gray code allows only one bit to change at a time. To ensure proper communication between different clock domains, the pointer values are synchronized using a two-stage flip-flop delay mechanism. The empty and full conditions are determined by comparing the synchronized pointer values. If both pointers are equal, the FIFO is considered empty, and if the write pointer reaches a specific condition relative to the read pointer, the FIFO is considered full. Although the system may occasionally generate false empty or false full signals, these are acceptable as they provide a safety margin and prevent incorrect operations.

To track the position of data within the FIFO, the system uses two pointers: the write pointer and the read pointer. These pointers are initially in binary form but are converted into Gray code to reduce metastability issues, as only one bit changes at a time in Gray code. Since the read and write operations occur in different clock domains, the pointer values must be synchronized before comparison. This is achieved using a two-stage flip-flop synchronization technique, which delays the pointer signals and ensures safe transfer between clock domains without causing errors.

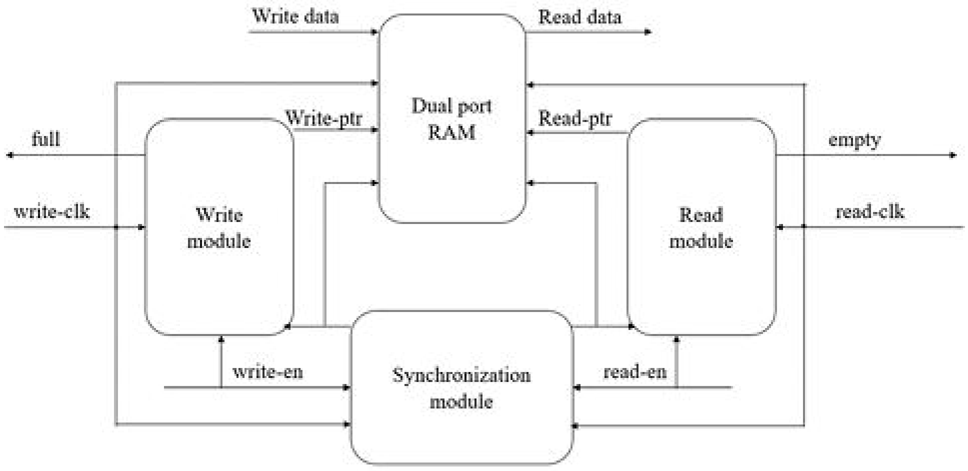


Fig. 1 Asynchronous FIFO basic module

The Fig1 represents the basic architecture of an asynchronous FIFO system, which is used to transfer data between two different clock domains. At the center of the diagram is the **dual-port RAM**, which acts as the storage element. It allows data to be written and read simultaneously through separate ports. On the left side, the **write module** operates using the write clock (write-clk) and is responsible for accepting input data, generating the write pointer (write-ptr), and controlling write operations based on the write enable signal (write-en). It also checks whether the FIFO is full and prevents further writing when the full condition is reached.

On the right side, the **read module** works with the read clock (read-clk) and manages the reading of data from the RAM. It generates the read pointer (read-ptr), controls read operations using the read enable signal (read-en), and ensures that data is only read when the FIFO is not empty. The output data is taken from the RAM through this module. The **empty** signal indicates that there is no data available to read.

At the bottom, the **synchronization module** plays a crucial role in handling communication between the two different clock domains. It synchronizes the read and write pointers by passing them through delay elements (usually flip-flops), ensuring safe comparison and preventing metastability issues. The synchronized pointers are then used to generate the **full** and **empty** signals accurately. Overall, all these modules work together to maintain proper data flow in a First-In First-Out manner while ensuring reliable operation across different clock domains.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

IV. EXPERIMENTAL RESULT

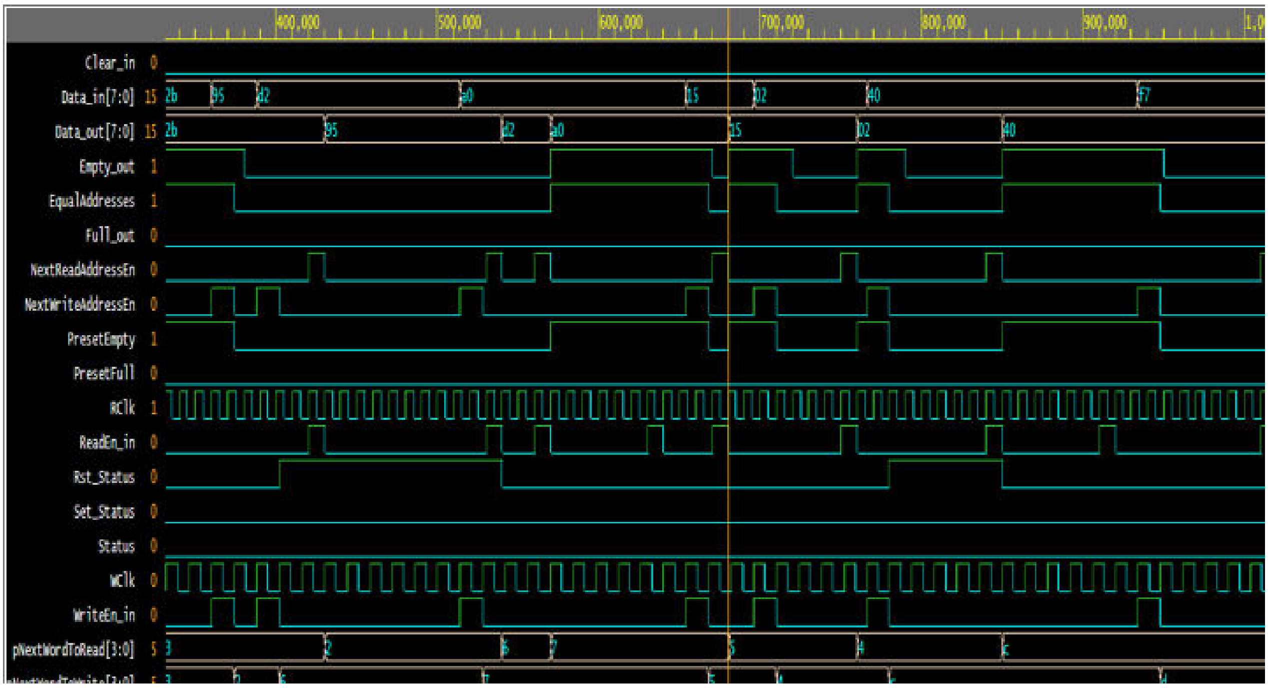


FIG2 OUTPUT RESULTH

The Fig2 given simulation result waveform illustrates the behavior of the asynchronous FIFO during write operations. The **write clock (wclk)** is continuously toggling, indicating the active write domain. The **write enable signal (write_en)** controls when data is written into the FIFO. Whenever the write enable is high, data is successfully written into the memory, which can be observed through the incrementing values of the write address pointer (**wr_ptr** or related address signals). The waveform shows that the pointer values increase sequentially, confirming that data is being stored in consecutive memory locations.

Additionally, the timing relationship between the clock and enable signals demonstrates that data is written only at the correct clock edges, ensuring proper synchronization. The absence of irregularities or unexpected changes in the waveform indicates stable operation. The controlled increment of pointer values also shows that overflow conditions are avoided, meaning the FIFO does not allow writing when it is full. Overall, the result confirms that the write operation of the asynchronous FIFO is functioning correctly, with proper control, synchronization, and sequential data storage.

V. CONCLUSION

In this paper, the basic module of asynchronous FIFO is designed and implemented by Verilog, and the null and full signal is generated under the condition of reducing the probability of metastable state. It can be seen from the simulation diagram that the designed asynchronous FIFO meets the requirement that the data written first is read out first. When the data in the asynchronous FIFO reaches or approaches the maximum depth of double-port RAM, full signal is generated; when there is no data in the asynchronous FIFO or near the empty signal, the expected effect is achieved. A further explanation is given for the empty - full signal: the empty - full signal generated by asynchronous FIFO is sometimes not the true vacuum full. This design uses a small part of performance loss to exchange for the safety margin, which is in line with the design goal

The simulation results clearly demonstrate that the designed asynchronous FIFO operates correctly and efficiently across different clock domains. The system successfully maintains the First-In First-Out (FIFO) principle, ensuring that

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

data written first is read first without any loss or corruption. The independent operation of read and write clocks confirms the ability of the design to handle clock domain crossing effectively. Additionally, the proper generation of **empty** and **full** signals prevents overflow and underflow conditions, ensuring safe data transfer. The use of synchronization techniques and Gray code conversion helps in reducing metastability issues and improves reliability. Overall, the design achieves a good balance between performance and safety, making it suitable for real-time digital systems and communication applications.

REFERENCES

[1] Wang GC, Ma YP, Lu HT, Chen XQ, Li P. Clock Domains Cross FIFO Interface Design of Multichannel Continuous DDR2 Read and Write. Applied Mechanics and Materials. 2014;556 562:1622-1626.
[2] Yadlapati A, Kishore Kakarla H. Design and Verification of Asynchronous FIFO with Novel Architecture Using Verilog HDL. Journal of Engineering and Applied Sciences. 2019;14(1):159-163.
[3] Van De Goor AJ, Zorian Y. Effective march algorithms for testing single-order addressed memories. Journal of Electronic Testing. 1994;5(4):337-345.
[4] Kumar A, Shankar S, Sharma N. Verification of Asynchronous FIFO using System Verilog. International Journal of Computer Applications. 2014;86(11):16-20.
[5] Liu BQ, Liu MZ, Yang G, Mao XB, Li HL. Research and Design of Asynchronous FIFO Based on FPGA. Applied Mechanics and Materials. 2014;644-650:3440-3444.
[6] Konstantinou D, Psarras A, Nicopoulos C, Dimitrakopoulos G. The Mesochronous Dual-Clock FIFO Buffer. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems. 2020;28(1):302-306.
[7] Akhare* M, Narkhede N. Design and Verification of Generic FIFO using Layered Test bench and Assertion Technique. International Journal of Engineering and Advanced Technology. 2019;8(6):5254 5260.
[8] Kim HK, Wang LT, Wu YL, Jone WB. Testing of Synchronizers in Asynchronous FIFO. Journal of Electronic Testing. 2013;29(1):49-72
[9] Cui ZJ, Hu LL. Design of SDH Positive/Zero/Negative Justification Circuits Based on FPGA. Advanced Materials Research. 2013; 748:874-878.
[10] Abdel- hafeez S, Gordon- Ross A. Reconfigurable FIFO memory circuit for synchronous and asynchronous communication. International Journal of Circuit Theory and Applications. 2021;49(4):938 952.

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING