Verification of Synchronous FIFO

Harsh Patel[1] - 20bec084@nirmauni.ac.in, Pratik makawana[2] – 20bec090@nirmauni.ac.in
Electronics and Communication, Institiute of Technology, Nirma University Ahmedabad, India

***Abstract*—A FIFO register is a type of register having sequential circuits that is used for pipelining in sequential instruction execution. In terms of data structure, FIFO is comparable to a queue, but it is a hardware device that functions in a cyclical fashion. It has a variety of uses, including data storage, pipelining,and buffering. A Verilog testbench is created to check the condition of the system in all feasible scenarios.**

***Index Terms*—Working of FIFO,Layered Testbench,Flowchart,simulation**

I. Introduction

For memories read and write of any information and data utilising certain control logic, FIFO stands for first in firstout employing queue approach. The control circuitry and clock domain are absolutely essential to FIFO's operation. Thetransition of each clock is frequently utilised to manage the flow of data from source to destination. FIFOs are classified asSynchronous or Asynchronous based on their clock domain. There are other ways for constructing and synthesising FIFOs,but this paper focuses solely on the memory that is used to store data in the clock domain, which can be either sync or async, or single or several clock cycles.

Because of their high operating speed, synchronous FIFOs are the best solution for high-performance systems. Syn- chronous FIFOs have a slew of other benefits that boost system performance while reducing complexity. Status flags include synchronous flags, half-full, programmable almost-empty, andalmost-full flags, among others.

II. Types of FIFO

1. Synchronous FIFO
2. Asynchronous FIFO

III. Working of FIFO

A FIFO is a register with a set of bits that allows data tobe stored sequentially in and out. The DUT in this example is an 8-bit FIFO with a total of 16 memory elements, which means it may store up to 16 different 8-bit data and output itas needed based on its entry in the FIFO. To ensure that the FIFO's function is uninterrupted, there are several registers and inputs. The primary programme will be structured around four parts.This programme operates as a circular queue, which means that when the last memory element is filled and the first few items have already been read, FIFO will write new data from the first memory element. "writepointer" is the first module. There are two pointers in the memory of 16 registers that will store the value of the index in the FIFO memory fromwhich the last read and write operations are done. A pointerto the most recent write operation is saved in the register wptr, and its location will aid in the execution of subsequent operations.Though a 4-bit pointer can address 16 indices, wptris a 5-bit pointer with the MSB bit used to check for overflow and underflow issues when the FIFO is in circular format.The first condition to check is whether or not the FIFO is full.Ifit isn't full, it will look to see if the write register 'we' is setto 1, indicating that the user is ready to write new data to the memory.

If both conditions are met, wptr will be set to +1; otherwise,it will remain unchanged. A flag register in the status module determines whether the FIFO is full or not."Readpointer" comes in second. A pointer, similar to wptr, is used to record the value of index, which points to the FIFO memory index where the last read operation was done. It's abbreviated as rptr.Even if the read command is not sent by the input bit're' each time, output is seen regardless of whether rptr is modified or not. A status flag is checked and input register're' is checked to modify the value of rptr; if both conditions are met, the value of rptr is changed to +1; otherwise, the value of rptr is not changed. "memoryarray" is the third module. The values of the 16 separate 8-bit registers must be stored in an array by this module. If the write signal is given as input, it will check for the condition and assign the value to the memory element. It checks for no conditions before reading the memory and outputs from the memory element. The last four bits of wptr and rptr are designed to point to the address where changeis supposed to occur. "statuescheck" is the last module. This module will determine whether or not the weather stack is full or empty. Because FIFO is circular, once the wptr has reachedthe maximum index, all four LSB bits will be set to 0 and the fifth bit will be set to 1.

As a result, there's a first bit comparator that checks whether wptr has started the loop again or not. This 5th bit is XORed; if it is 1, both loops are distinct; otherwise, both loops are the same. The difference between the two values of the last four bits of wptr and rpte is stored in a register. If the difference is 0, it means that they are both in the same place. If the comparator value is 0, the wptr and rptr are in the same loop and the condition is under flow; however, if the comparator value is 1, the wptr and rptr are in a different loop and the FIFO is overflowing.When the reset bit is set, the values of wptr, rptr, and all status flags are set to 0, and the memory array is cleared.

IV. DIAGRAM.

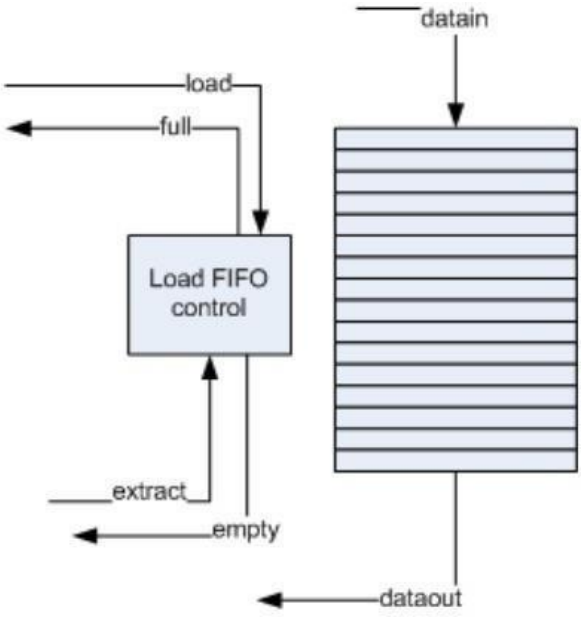

Fig. 1

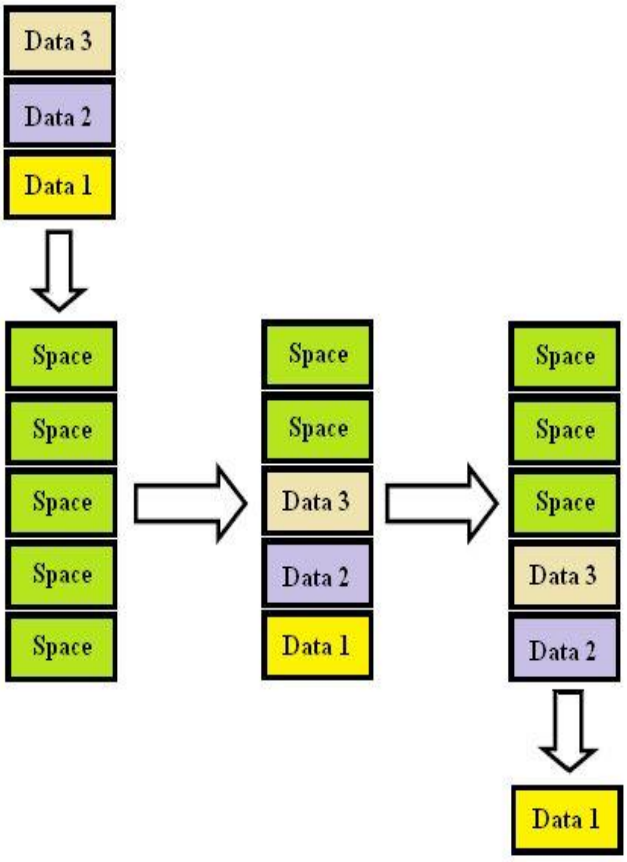

Fig. 2

V. Two types of verification in FIFO

Verification of synchronous FIFO designs is important to ensure that the circuit operates as expected, meets its performance requirements, and does not cause any errors or data loss during the data transfer process. The verification process involves various steps, including functional verification, timing verification, and verification of corner cases.

1) Functional Verification
2) Timing Verification

1) Functional Verification

Functional verification involves testing the FIFO for its basic functionality, such as data input, data output, and data storage. This is done by applying test vectors to the input ports and verifying the output results against the expected results.

2) Timing Verification

Timing verification involves testing the FIFO for its timing requirements, such as setup time, hold time, and clock frequency. This is done by applying timing constraints to the input ports and verifying that the FIFO meets the required timing specifications.

VI. Result

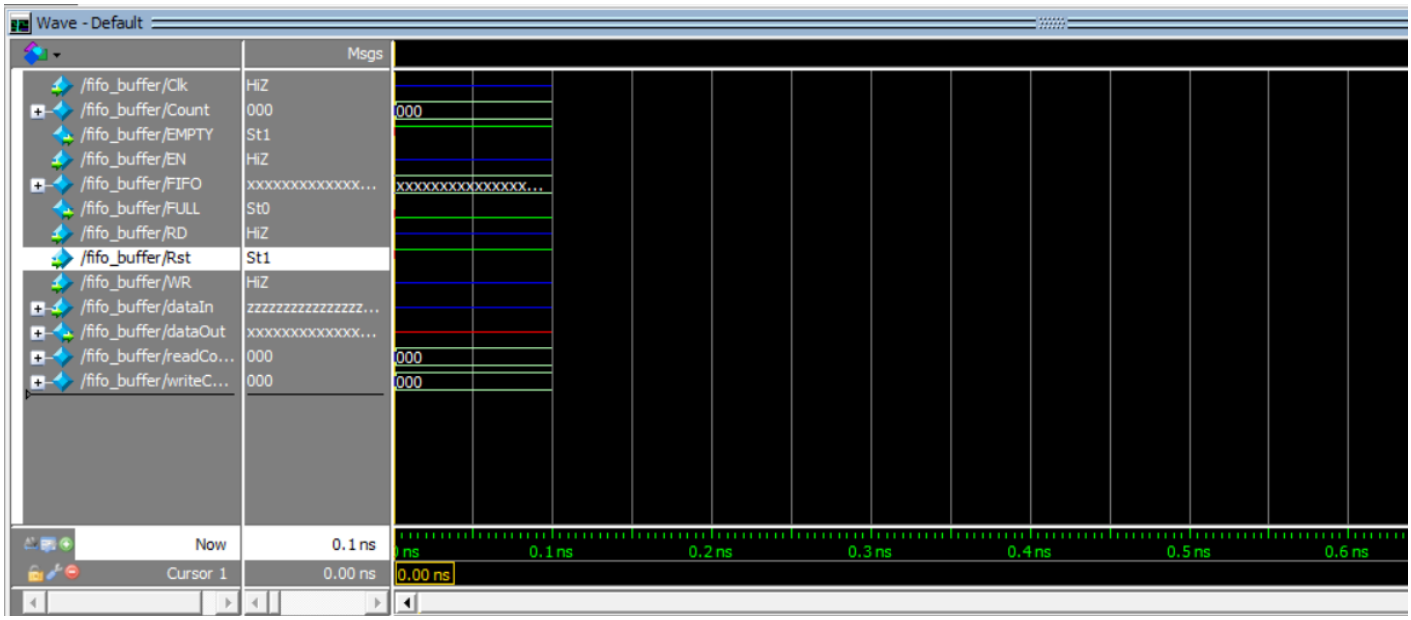

Fig. 3 the reset waveform

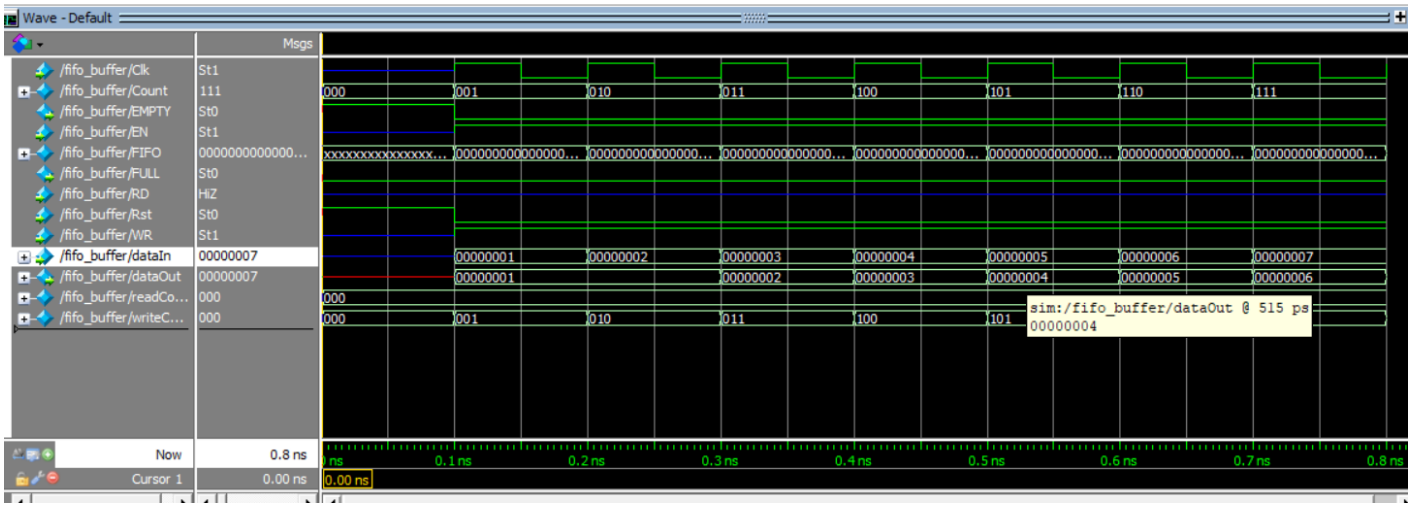

Fig.4 write condition waveform

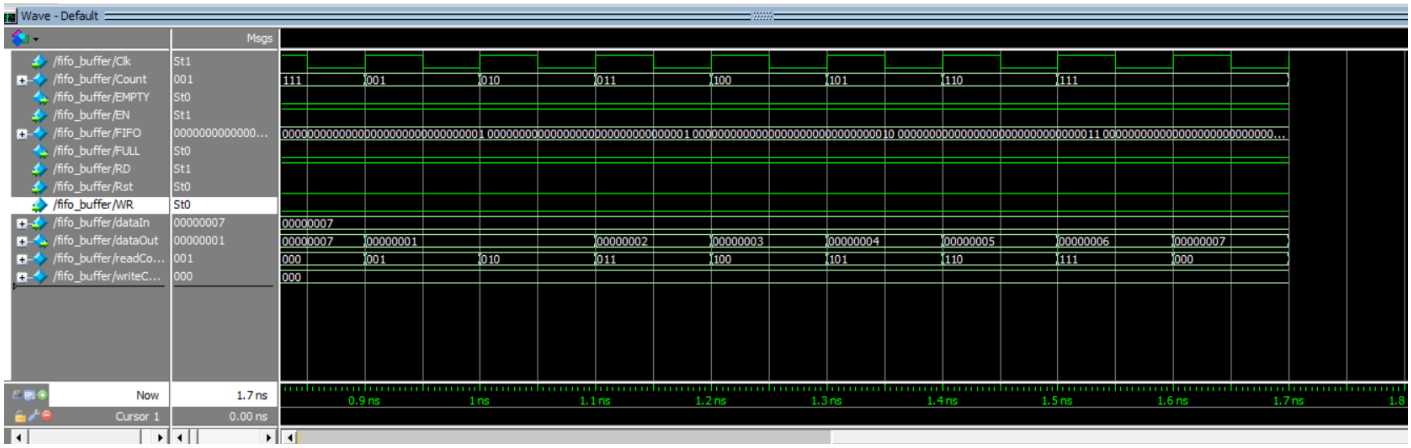

Fig. 5 read condition waveform

VII. Application

In electrical circuits, FIFOs are commonly used for buffering and flow control between hardware and software. In its hardware form, a FIFO consists mostly of a series of read and write pointers, as well as storage and control circuitry. Any suitable storage device, such as static random access memory (SRAM), flip-flops, or latches, can be employed. For FIFOsof any size, a dual-port SRAM is commonly utilised, with one port dedicated to writing and the other to reading.

VIII. Advantages

The obvious benefit of FIFO is that it is the most extensively utilised technique of inventory valuation worldwide. It's also the most precise means of aligning predicted cost flow with actual cost flow, giving firms a more accurate picture of inventory expenses. It also mitigates the effects of inflation by assuming that the cost of purchasing newer inventory will be higher than the cost of purchasing older inventory. Finally, inventory obsolescence is reduced.

IX. Conclusion

FiFo is a data structure that is similar to Queue. The workings of FIFO are detailed in this publication. This paper will help you understand how Testbench works and how to create a test bench for a FIFO structure. It will also show you how to use FIFo.

Refrences:

[1] Clifford E. Cummings, "Synthesis and Scripting Tech-niques for Designing Multi-Asynchronous Clock Designs," SNUG 2001 (Synopsys Users Group Conference, San Jose, CA, 2001) User Papers, March 2001, Section MC1, 3rd paper.

[2] Dinesh Tyagi, former CAE Manager for Synopsys
DesignWare product, personal communication.

[3] Clifford E. Cummings and Don Mills, "Synchronous Resets? Asynchronous Resets? I am so confused! How willI everknow which to use?," SNUG 2002 (Synopsys Users Group Conference, San Jose, CA, 2002) User Papers, March 2002,Section TB2, 1st paper.

[4] Clifford E. Cummings and Peter Alfke, "Simulation and Synthesis Techniques for Asynchronous FIFO Design with Asynchronous Pointer Comparisons," SNUG 2002 (Syn- opsys Users Group Conference, San Jose, CA, 2002) User Papers,March 2002, Section TB2, 3rd paper.