VOLUME 13 SPECIAL ISSUE 1, JUNE 2022 ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

DESIGNING AND VERIFICATION OF ASYNCHRONOUS FIFO

Piyush Kumar
ECE Department
GITS
Udaipur, India
pk5784652@gmail.com

Prof. Rahul Moud
ECE Department
GITS
Udaipur, India
rahul.moud@gits.ac.in

Abstract: Asynchronous FIFOs are often used to safely pass data from one clock domain to another asynchronous clock domain. Using a AFIFO to pass data from one clock domain to another clock domain requires multi -asynchronous clock design techniques. There are many ways to design a FIFO but still make it difficult to properly simulate and analyse the design. FIFO can be either synchronous or asynchronous. The basic difference between them is that the entire operation of synchronous FIFO is dependent on the single clock whereas asynchronous FIFO have separate clock for the write operation and read operation.
This paper discusses about Asynchronous FIFO design and verification using Verilog and analyse the outputs using simulation performed in Questasim.

Keywords: Asynchronous FIFO

I. INTRODUCTION

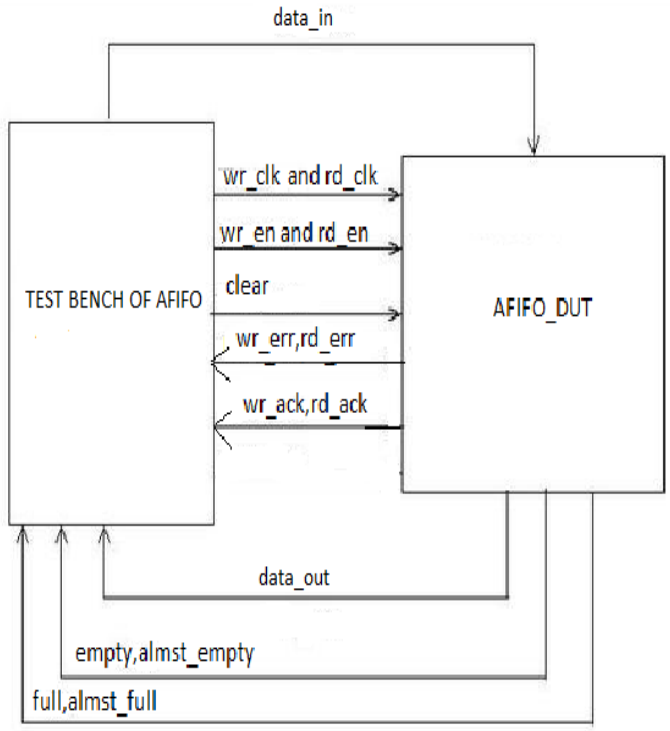
An asynchronous FIFO refers to a FIFO design where data values are written to a FIFO buffer from one clock domain and the data values are read from the same FIFO buffer from another clock domain, where the two clock domains are asynchronous to each other.

Asynchronous FIFOs are used to safely pass data from one clock domain to another clock domain. Advantage of Asynchronous FIFO is that the circuit can be reset with or without a clock present.

The functionality of FIFO is mostly depending on the control signals like rd_en and wr_en. They both are set to zero on reset.

The generation of FIFO pointers is of utmost importance because they are clocked at different clocks, so they need to be synchronized. These Synchronized pointers are then compared to generate full and empty conditions.

AFIFO is used in electronic circuits mainly for buffering and flow control.



II. ASYNCHRONOUS FIFO DESCRIPTION

Asynchronous FIFO has following ports which are shown in below figure -

The read and write ports can be operated on independent asynchronous clock domains.

All signals, either input or output, are synchronous to one of the two clocks. which performs an asynchronous reset of the entire FIFO.

Asynchronous initialization (**clear)** will force all FIFO flags to the active **(low)** state.

© 2020-2022, IJARCS All Rights Reserved

The control **(wr_en)** and data input **(data_in)** are sampled by the rising edge of **wr_clk** and should be synchronous to the **wr_clk**.

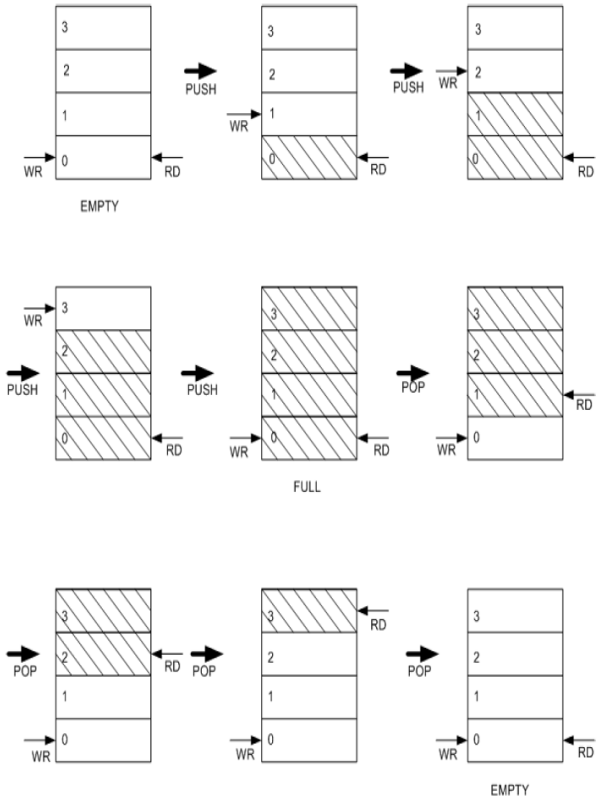
For the read side the read control **(rd_en)** should be synchronous to the **rd_clk** and the output data (**data_out)** is valid at rising edge of rd_**clk**.

The **wr_ack** and **wr_err** signals indicate acknowledgment or rejection of requested write operations. Similarly, **rd_ack** and **rd_err** signals indicate the acknowledgment or rejection of read operations.

All these handshake signals are synchronous to their respective clock domains and indicate the acknowledgment or rejection of requests during the prior rising clock edge.

All status outputs are synchronous to their respective clock domains and sampled only by logic operating on a synchronous clock.

FIFO Full and Empty description:

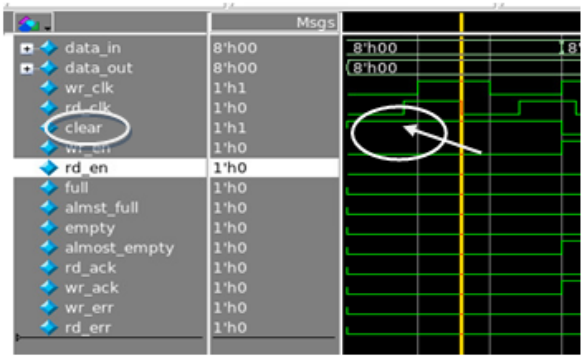


The **almst_empty** flag is active when the FIFO has one data word to empty. The **almst_full** flag is active when the FIFO has only one available memory location to full.

III. SIMULATION AND BEHAVIOR OF AFIFO SIGNALS

A. Reset (clear)

We can see through below waveform when **clear** signal is high, all status and handshake signal of are set to Zero (0).

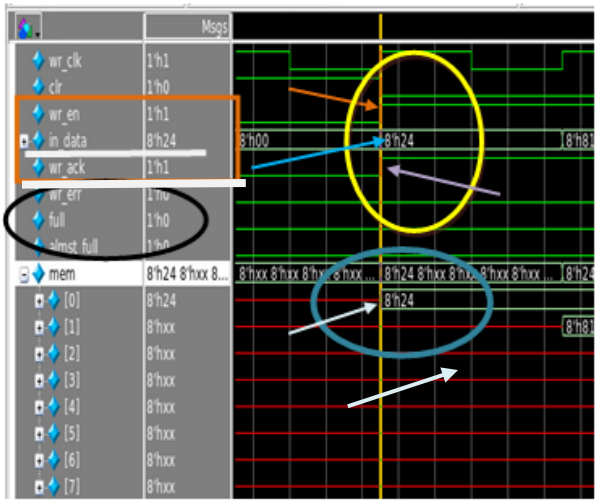


B. Writing Data in the Memory

Wr_en and Wr_ack

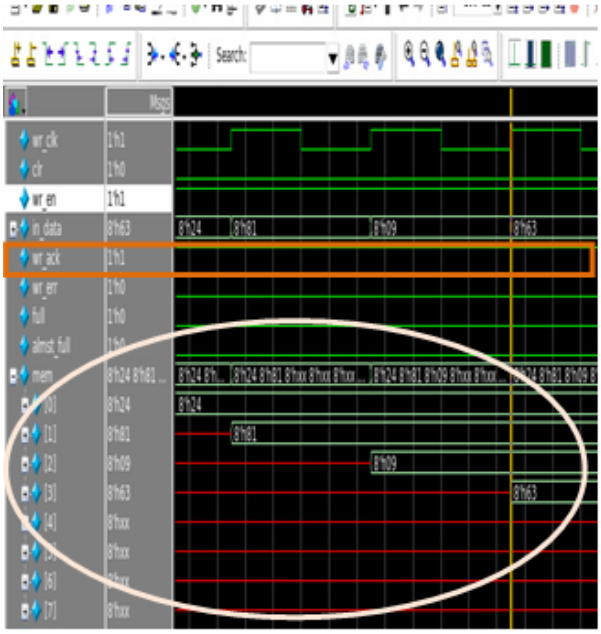
In this FIFO We used a memory which have 8- locations and each location is 8-bit wide.

[7:0] mem [0:7]

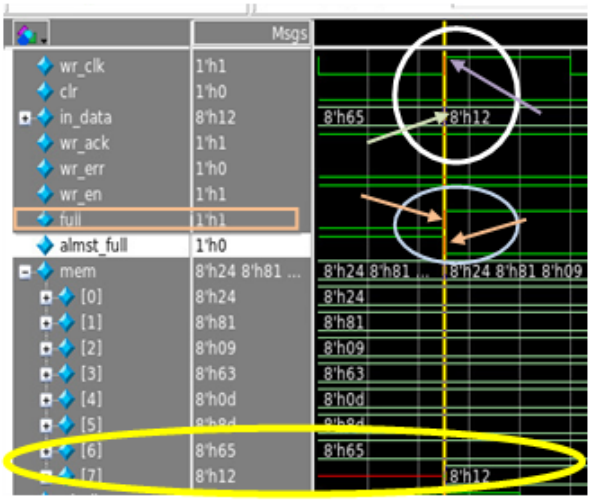


In the above waveform when **wr_en** is high then the **data_in** is store in the **memory** at posedge of **wr_clk** and the **wr_ack** signal is high, which indicates successful write operation.

© 2020-2022, IJARCS All Rights Reserved



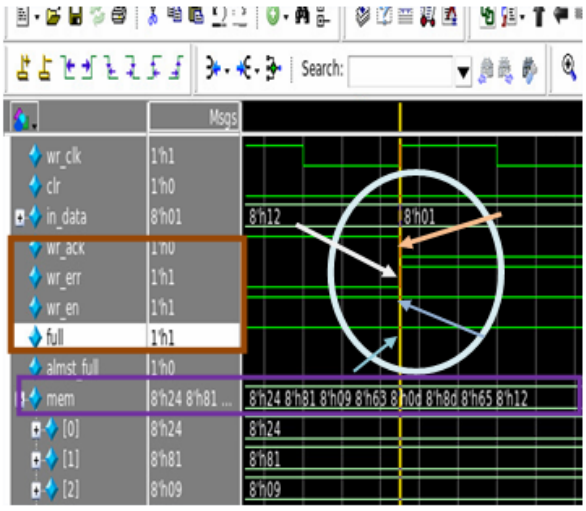
Full and Almst_full



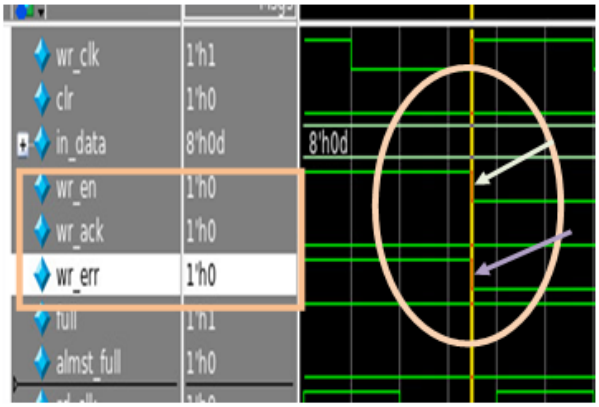
In the Above waveform, we can see when there are only one available location in memory the **almst_full** flag is active high and when that last location is fill with data_in than the **full** flag is active high.

Wr_err

In the below waveform we can see, we are requesting a write operation while the **full** flag is active(high) but this will not cause any change in the current state of the FIFO, and the **wr_err** handshake signal will indicate the rejection of these invalid requests.

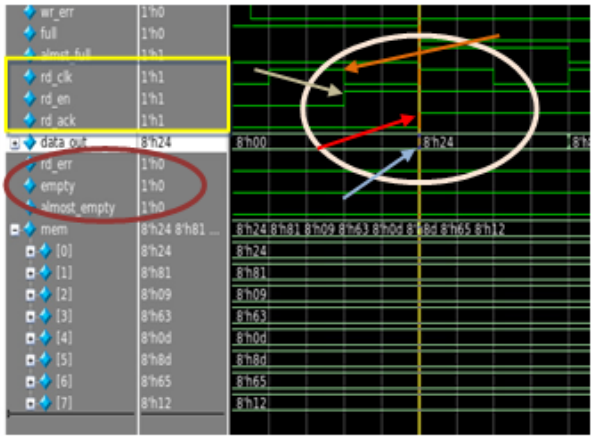


In the below figure, when we disable the **wr_en** signal then the **wr_error** flag will become zero (0).



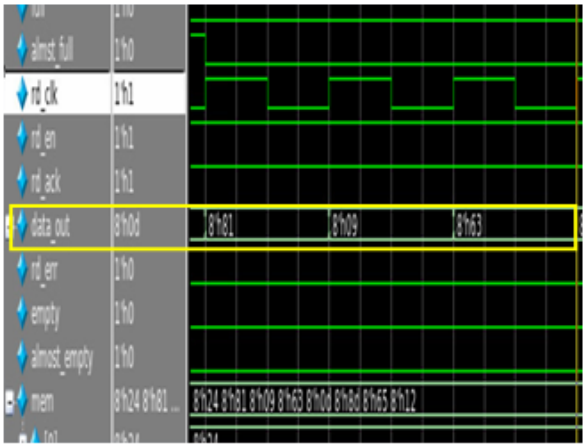
C. Read Data from the Memory

Rd_en and Rd_ack



In the above waveform **rd_en** signal enable at negedge of **rd_clk** but it will read data at posedge of **rd_clk** from the **memory** and the **rd_ack** signal will be active high, which indicates successful read operation.

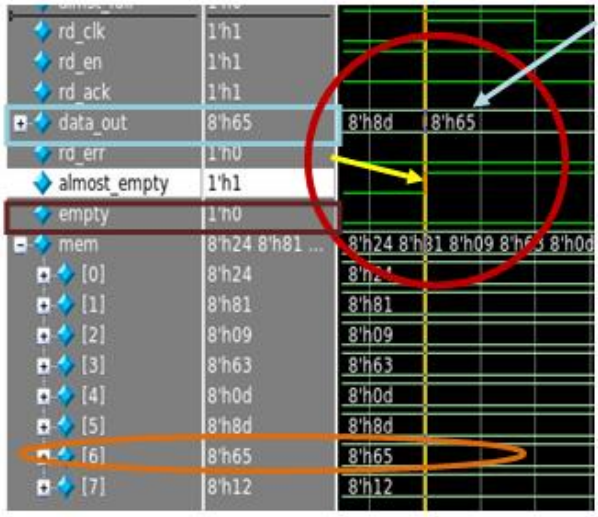
© 2020-2022, IJARCS All Rights Reserved

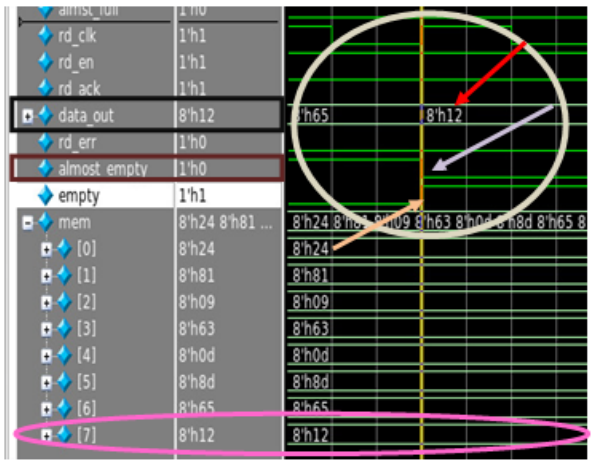


Empty and Almst_empty

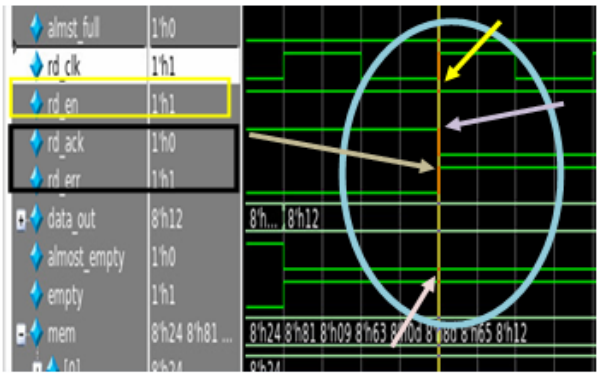
Below two waveforms shows the **empty** and **almst_empty** flags,

almst_empty flag is high when the FIFO has one data word. **empty** flag is high only when last data word is read out.



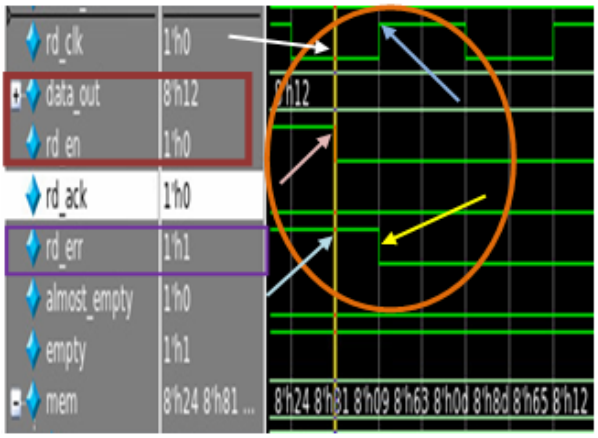


Rd_err



In the above waveform we can see, we are requesting a read operation while the **empty** flag is active, but this will not cause any change in the current state of the FIFO, and the **rd_err** handshake signal will indicate the rejection of these invalid requests.

In the below waveform, when we disable the **rd_en** signal then the **rd_error** flag will become zero (0) at posedge of **rd_clk**.



© 2020-2022, IJARCS All Rights Reserved

IV. CONCLUSION

In this paper, we have accomplished design and verification of the asynchronous FIFO using Verilog language. The said Design Under Test was successfully simulated and analyzed for various operations like read and write along with status flags and handshake signals.

V. ACKNOWLEDGEMENTS

Thanks to Mr. Rahul moud sir who have contributed towards development of the Asynchronous FIFO.

VI. REFERENCE

[1] P Rajshekhar Rao, Manju Nanda., "Implementation and Verification of Asynchronous FIFO Under Boundary Condition", International Journal of Engineering Research & Technology (IJERT) 2278-0181.

[2] Mohini Akhare, Nitin Narkhede., "Design and Verification of Generic FIFO using Layered Test bench and Assertion Technique", International Journal of Engineering and Advanced Technology (IJEAT). Issue-6, August 2019.

[3] Lincy DF, S.Thenappan., "ASYNCHRONOUS FIFO DESIGN USING VERILOG", International Research Journal of Engineering and Technology (IRJET), Volume: 07 Issue: 09, Sep 2020.

[4] Dadhania Prashant C., "Designing Asynchronous FIFO", International journal of information, knowledge and research in electronics and communication engineering, Volume 02, issue 02, pp-561-563

© 2020-2022, IJARCS All Rights Reserved