International Journal of Scientific Research and Engineering Development-– Volume 5 Issue 5, Sep- Oct 2022

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Asynchronous FIFO Design for Bi-mode MAC protocol

Harini E*, Dr. Vasundhara Patel K.S**

*(Department of Electronics and Communication, BMS College of Engineering, Bangalore
Email: harinianjani96@gmail.com)

** (Department of Electronics and Communication, BMS College of Engineering, Bangalore
Email:vasu.ece@bmsce.ac.in)

Abstract:

When any protocol is implemented FIFO design is the mandatory module, which provides delay compensation and synchronization between two clock domain signals. Media access control (MAC) is the sublayer of Data link layer in the OSI model of computer networking. Each layer in the model has defined set of protocols to handle particular operation in the data transfer. MAC is responsible detecting and fixing the error during the data transfer, which intern has many protocols to handle the tasks. The major module in MAC protocol is FIFO which collects data from the upper layer, encapsulates it and transmit the data to physical layer. This paper presents the design of asynchronous FIFO module for the MAC protocol that operates at bi-mode, i.e., two data link speeds based on application requirement.

Keywords **—Asynchronous FIFO, MAC, OSI.**

I. INTRODUCTION

The medium access control (MAC, also known as media access control) sublayer is the layer that controls the hardware responsible for interfacing with the wired, optical, or wireless transmission medium. The MAC and logical link control (LLC) sublayers make up the data link layer. The LLC manages flow and multiplexes data for the logical connection, whereas the MAC manages flow and multiplexes data for the transmission channel.When sending data to another network device, the MAC sublayer encapsulates higher-level frames into frames appropriate for transmission medium. When the appropriate channel access mechanism allows it, it adds a frame check sequence to catch transmission faults and transmits the data to the physical layer. For topologies with a collision domain, controlling when data is sent and when to wait is necessary in order to avoid collisions. In the event that a jam signal is detected, the MAC is also in charge of compensating for collisions by initiating retransmission. Upon receiving data from the physical layer, the MAC block checks the sender's frame check sequences, ascertains data integrity, and eliminates the sender's preamble and padding before sending the data up to the upper levels. Out of all the modules in the MAC, FIFO is the major part of the design, which is present at the entry of transmission and reception modules.

First in, first out (FIFO) is used in computer science and systems theory to describe a way of managing the modification of a data structure, frequently a data buffer, where the oldest (first) entry, or "head" of the queue, is processed first. FIFO is of two types Synchronous and Asynchronous. A synchronous FIFO is a First-In-First-Out queue in which a single clock pulse is used for both data writing and reading. In a synchronous FIFO, read and write operations happen simultaneously. The width or word length of a FIFO is the number of bits in each row, whereas the depth or word count of a FIFO is the number of rows.

Asynchronous FIFOs are FIFOs where data values are simultaneously read from the FIFO at one rate and written to it at another rate. It is referred to as an asynchronous FIFO since the read and write clocks are not synchronised.The essential need for an asynchronous FIFO comes when working with systems that run at different data speeds. Due to the different speeds of data flow, we will need asynchronous FIFO to synchronise the data flow between the systems. The fundamental purpose of an Asynchronous FIFO is to move data from one clock domain to another.

II. PROPOSED DESIGN

When working with systems that operate at various data rates, the fundamental requirement for an asynchronous FIFO arises. We will require asynchronous FIFO to synchronise the data flow between the systems due to the various rates of data flow. An Asynchronous FIFO's primary function is to transfer data from one clock domain to another. The proposed design is shown in Figure 1.

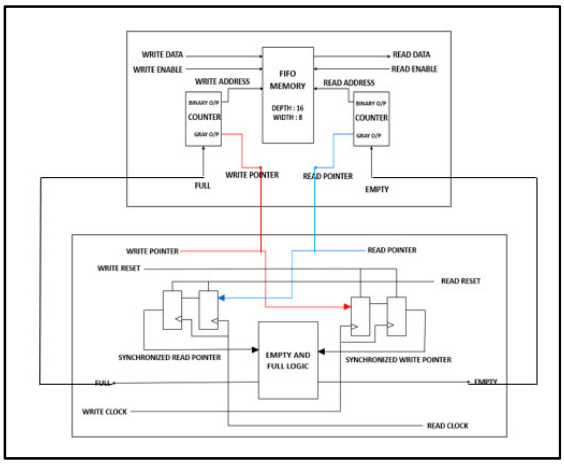

Figure 1: Proposed Asynchronous FIFO design

One must comprehend the operation of FIFO pointers in order to comprehend FIFO design. Both pointers are reset to zero upon reset, which also happens to be the location of the current word, because the write pointer always leads to the next FIFO word that needs to be written. In a FIFO-write operation, the write pointer is incremented to refer to the following location that has to be written after the memory address that it points to has been written to.Similar to that, the read pointer constantly draws focus to the FIFO word that is now being read. Once again, upon reset, the FIFO is empty, both points are reset to zero, and the read pointer is pointing to erroneous data (due to the empty flag being asserted and the FIFO being empty). The read pointer, which is still addressing the first FIFO memory word's contents, immediately drives the first valid word onto the FIFO data output port so that the receiver logic can read it. The write pointer increases as soon as the first data word is written to the FIFO, the empty flag is cleared, and the first FIFO memory word is addressed.The receiver logic does not need to require two clock periods to read the data word because the read pointer is always pointing to the following FIFO word to be read. The receiver would clock twice to output the data word from the FIFO and once again to read the data word into the receiver if the receiver required to first increase the read pointer before reading a FIFO data word. That would be absurdly ineffective.

When the read and write pointers are equal, the FIFO is empty. This circumstance develops when the write pointer catches up to the read pointer after reading the last word from the FIFO, or when both points are reset to zero after a reset operation.The write pointer has wrapped around and caught up to the read pointer when the pointers are equal once more, and the FIFO is full. The write pointer advances the unused MSB while setting the remaining bits back to zero as it moves past the last FIFO address. The read pointer is applied similarly. If the MSBs of the two points are different, the write pointer has looped more often than the read pointer. If the MSBs of the two points are equivalent, it means that both pointers have wrapped exactly once.

III. RESULTS AND SIMULATION

All Memory is used to store the incoming data. Read and write logic to read from and write to the

FIFO memory address. The logic implemented for writing and reading the data to and from the FIFO. Two clocks for reading and writing data is maintained. Two separate signals are maintained to enable read and write signal. The flag is set if the FIFO is full to avoid overflow of FIFO that ensures no data is missed during the transaction. The simulation output of the asynchronous logic is shown below.

Figure 2: Simulation Output of Asynchronous FIFO Design

IV. CONCLUSIONS

Most of the designs have FIFO module defined for delay compensation, synchronization and data storage. Between synchronous and asynchronous FIFO, asynchronous FIFO is preferred in the design as it can handle different clock domains signals interact with each other, as most of the current designs implemented are of multiple clock domains. Hence the design of stable asynchronous FIFO is of outmost important, which can be instantiated any number of times, implemented in majority of the designs, with error sensing mechanism and an interrupt to raise FIFO overflow. The design contains the modules mostly implemented for MAC protocols.

ACKNOWLEDGMENT

The Author wishes to acknowledge guidance and support from Dr Vasundhara Patel K.S and Department of Electronics and Communication for their help in writing this journal. The author would also like to thank the BMS College of Engineering for encouraging their students in research-oriented studies.

REFERENCES

[1] N. Ahmed, D. De, F. A. Barbhuiya and M. I. Hussain, "*MAC Protocols for IEEE 802.11ah-Based Internet of Things: A Survey*," in *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 916-938, 15 Jan.15, 2022, doi: 10.1109/JIOT.2021.3104388.

[2] M. C. Utami, D. R. Sabarkhah, E. Fetrina and M. Q. Huda, "*The Use of FIFO Method For Analysing and Designing the Inventory Information System*," *2018 6th International Conference on Cyber and IT Service Management (CITSM)*, 2018, pp. 1-4, doi: 10.1109/CITSM.2018.8674266.

[3] Xin Wang, T. Ahonen and J. Nurmi, "*A synthesizable RTL design of asynchronous FIFO*," *2004 International Symposium on System-on-Chip, 2004. Proceedings.*, 2004, pp. 123-128, doi: 10.1109/ISSOC.2004.1411164.

[4] H. Ashour, "*Design, simulation and realization of a parametrizable, configurable and modular asynchronous FIFO*," *2015 Science and Information Conference (SAI)*, 2015, pp. 1391-1395, doi: 10.1109/SAI.2015.7237325.

[5] D. Wyland, "New features in synchronous FIFOs," *Proceedings of WESCON '93*, 1993, pp. 580-585, doi: 10.1109/WESCON.1993.488598.

