Asynchronous FIFO

System Description

An Asynchronous FIFO (First-In, First-Out) buffer is a type of memory structure that temporarily stores data and is used for communication between two digital systems that operate at different clock frequencies (i.e., they are asynchronous with respect to each other). The FIFO's primary purpose is to allow data transfer between these systems without the need for synchronization between their clocks, which helps prevent data corruption and timing issues. First-In First-Out (FIFO) means that the first data item written into the buffer is the first one to be read out. This ordering helps ensure that data flows consistently and predictably, even when the systems are not synchronized.

The Asynchronous FIFO operates based on two separate clock domains:

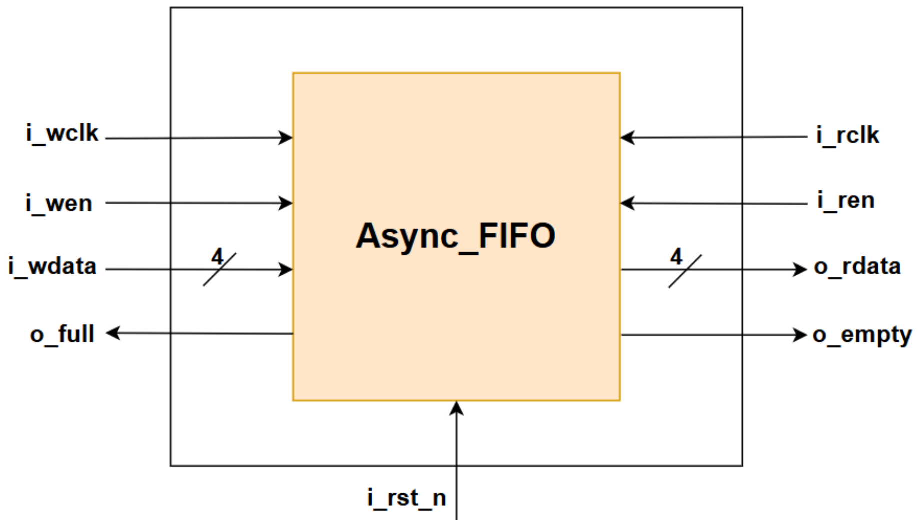
- **Write Clock (WCLK)**: Controls the write operations.
- **Read Clock (RCLK)**: Controls the read operations.

The FIFO keeps track of the data's position with **write and read pointers**. Data is written sequentially at the write pointer location which Increments with each write operation to indicate the next free location in the buffer and read sequentially from the read pointer location which increments with each read operation to indicate the next data to be read. Both pointers move independently based on their respective clock signals.

To manage the data flow and maintain FIFO properties, the following status flags are used:

- **Empty**: Indicates that there is no data available to read. The FIFO is empty when the read pointer reaches the same position as the write pointer.
- **Full**: Indicates that the FIFO has no space available to write new data. The FIFO is full when the write pointer reaches the same position as the read pointer after completing one full cycle through the buffer.

Since the FIFO is asynchronous, specialized techniques are used to safely transfer control signals (such as the status flags) between the read and write clock domains. **You have to choose a technique in your design.**



In this design, the FIFO depth is set to 8 locations, with each location holding 4-bit wide data. Thus, both the written and read data will be 4 bits wide. The read and write pointers are managed internally and do not appear in the interface.

Signals

Table 1

Signal	Width	Direction	Description
i_rst_n	1	Input	Negative edge asynchronous system reset in both domains.
i_wclk	1	Input	Positive edge clock of the write domain.
i_wen	1	Input	Write enable at the write domain: (1: write operation , 0: no write operation)
i_wdata	4	Input	Data written in the FIFO at the write domain in case of write operation.
o_full	1	Output	Flag Indicates that the FIFO is full (there is no space available to write new data).
i_rclk	1	Input	Positive edge clock of the read domain.
i_ren	1	Input	read enable at write domain: (1: read operation , 0: no read operation)
i_rdata	4	Output	Read data from the FIFO at the read domain in case of read operation.
o_empty	1	Output	Flag Indicates that the FIFO is empty (there is no data available to read).

Deliverables

You have to submit a zip file that contains:

- A PDF file that contains:
    - System Block Diagram.
    - The simulation waveform that shows the module is working correctly.
- The Vivado project folder contains:
    - RTL code of top module and any sub-blocks.
    - Testbench code of the system.