Lab 4: First In, First Out (FIFO)

Shibo Dang

Digital Design and Computer Architecture: RISC-V Edition (Harris & Harris, Elsevier © 2021)

Objective

In this lab, you will try to use the SystemVerilog to write the FIFO.

1. FIFO

In computing and in systems theory, FIFO is an acronym for first in, first out (the first in is the first out), a method for organizing the manipulation of a data structure (often, specifically a data buffer) where the oldest (first) entry, or "head" of the queue, is processed first. [1]

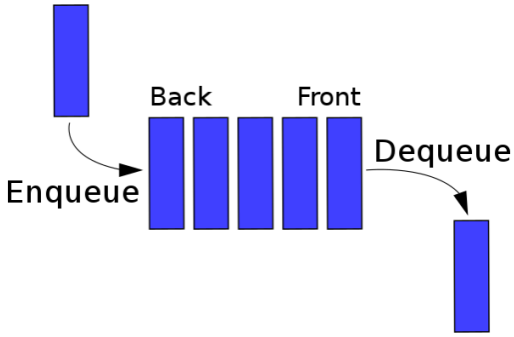

Figure 1: FIFO [1]

FIFO usually are divided into two categories: synchronize FIFO and asynchronous FIFO.

1) Synchronize FIFO

A Synchronous FIFO is a First-In-First-Out queue in which there is a single clock pulse for both data write and data read. In Synchronous FIFO the read and write operations are performed at the same rate. The number of rows is called depth or number of words of FIFO and number of bits in each row is called as width or word length of FIFO. This kind of FIFO is termed as Synchronous because the rate of read and write operations are same. [2]

What include into Synchronize FIFO:

- ONE clock
- Reset
- Write operation, enable, pointer
- Read operation, enable, pointer
- Empty, full flag
- (You need to recall the counter you write in the last lab. The counter is also an important part of this lab)

2) Asynchronous FIFO

An Asynchronous FIFO refers to a FIFO where the data values are written to the FIFO at a different rate and data values are read from the same FIFO at a different rate, both at the same time. The reason for calling it Asynchronous FIFO, is that the read and write clocks are not Synchronized. [2]

What include into Asynchronous FIFO:

- TWO clocks
- Reset
- Write operation, enable, pointer
- Read operation, enable, pointer
- Empty, full flag
- (You need to recall the counter you write in the last lab. The counter is also an important part of this lab)

What to Turn In

1. You should code the Synchronous FIFO and Asynchronous FIFO (both **16 x 8 FIFO**, 16 is the number of **rows**) by SystemVerilog with all the inputs and outputs, which are listed in the "what include into xxx FIFO" part. (Also, you can add the parts you need when you do the design.)
2. Write the testbench to test your design. For Asynchronous FIFO, you MUST set different clock frequencies for two different clocks.
3. Screenshot your waveform.
    (1) For both Synchronous FIFO and Asynchronous FIFO, you should include the following the situation in **ONE** waveform (This means one figure for synchronous, one for asynchronous.):
        ① Enqueue only, until full.
        ② Dequeue only, until empty.
        ③ Enqueue and Dequeue at the same time for several clock cycles. (More than **FIVE** clock cycles)
4. RTL Viewer schematics.

Additional Questions

1. For Asynchronous FIFO, to determine the full or empty, we should separate synchronous read and write pointers. In the case of crossing clock domains (under two clock frequencies), there is easy to occur meta-stable conditions. For this situation, you'd better transfer the binary pointer to the Gray code first, then synchronous. You can try to use Grey code in your Asynchronous FIFO design if you want. (Optional)

Reference

[1] Wikipedia contributors. "FIFO (computing and electronics)." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 13 Aug. 2022. Web. 24 Aug. 2022.
[2] Semiconductorclub "What is FIFO? | Synchronous FIFO | Asynchronous FIFO" Link: https://semiconductorclub.com/what-is-fifo-synchronous-fifo-asynchronous-fifo/#:~:text=An%20Asynchronous%20FIFO%20refers%20to,write%20clocks%20are%20not%20Synchronized.