DESIGN SPECIFICATION DOCUMENT: ASYNCHRONOUS FIFO

Pooja Satpute **Nandini Maddela**
Rafath Achugatla **Divya Sri Ayluri**

1. Features:

The Asynchronous FIFO memory module serves essential functions in digital systems:

● FIFO stores data. This is useful when the system writing data and the system reading data are in different clock domains.

● Due to transfer of data between different clock domains, there is a possibility of data corruption or the data could be over-written. FIFO ensures there is synchronization to avoid these problems.

● To ensure correct read pointer and write pointer values are used for FIFO full and FIFO empty conditions, pointers are converted to gray code.

● Conditions like overflow and underflow are taken care of using implementation that stops accepting data when the FIFO is full and pushing data out when the FIFO is empty.

2. FIFO implementation:

Image Source: Sunburst Design, links provided below.

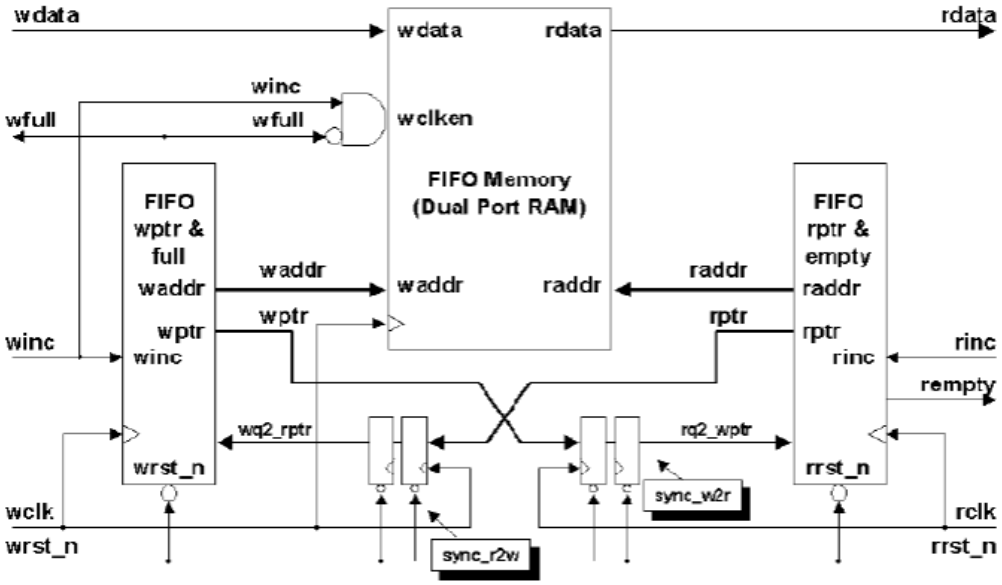

Figure 5 - FIFO1 partitioning with synchronized pointer comparison

3. Functional description:

FIFO is a first in first out data structure that is used to store data. The depth of the FIFO needs to be calculated depending upon the sender and receiver domain frequencies. It is usually employed in cases where there is a difference in two clock domains sending and receiving data. If the depth of the FIFO is not calculated properly the data written by the sender domain can be lost. Even after carefully calculating the FIFO depth, there could be cases where sender domain is trying to write to an already full FIFO or receiver is trying to read from an empty FIFO. To mitigate this condition, full and empty flags are asserted to let the system know that the FIFO is full or empty and then invalidate these requests. This is done by implementing pointers for write and read operations. These pointers are then synchronized in respective domains so that appropriate full and empty flags can be asserted or de-asserted. These pointers could result in having metastable values while sending from one domain to another domain, hence they are gray coded.

4. FIFO DEPTH CALCULATION

- Given: Sender clock frequency(fA) = 500MHz
- Write idle cycles = 2
- Write burst size = 1024
- Receiver clock frequency(fB) = 225MHz
- Read Idle cycles = 1

fA > fB with idle cycles in both write and read.

a. The no. of idle cycles between two successive writes is two clock cycle. It means that, after writing one data, module A is waiting for two clock cycles, to initiate the next write. So, it can be understood that for every three clock cycles, one data is written.
b. The no. of idle cycles between two successive reads is one clock cycle. It means that, after reading one data, module B is waiting for one clock cycle, to initiate the next read. So, it can be understood that for every two clock cycles, one data is read.
c. Time required to write one data item = 3 * 1/500MHz = 6ns
d. Time required to write all the data in the burst = 1024 * 6ns = 6144ns
e. Time required to read one data item = 2* 1/225MHz = 8.88ns
f. So, for every 8.88 nSec, the module B is going to read one data in the burst. So, in a period of 6144 nSec, 1024 no. of data items can be written.
g. The number of data items can be read in a period of 6144nsec,
(6144ns / 8.88ns) = 691
h. The remaining number of bytes to be stored in the FIFO = 1024 – 691=333
i. So, the FIFO which has to be in this scenario must be capable of storing 333 data items.

So, the minimum depth of the FIFO should be 333.

Reference:

Cummings, Clifford E. "Simulation and synthesis techniques for asynchronous FIFO design." *SNUG 2002 (Synopsys Users Group Conference, San Jose, CA, 2002) User Papers*. Vol. 281. 2002.

Depth calculation:

https://hardwaregeeksblog.wordpress.com/wp-content/uploads/2016/12/fifodepthcalculationmadeeasy2.pdf