FIFO X67970_en

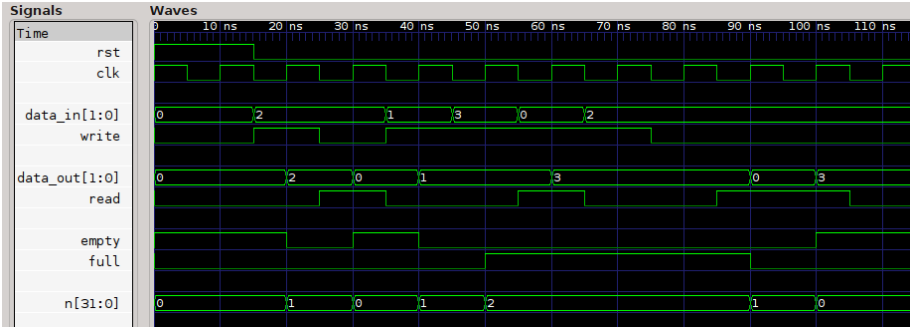
Design a First-In First-Out (FIFO) buffer, also called a queue.

The FIFO receives 2-bit data items and can store up to two items. Every read and write operation must be performed in one cycle. Simultaneous read and write operations are also possible.

The write operation stores the input data (*data_in*) into the FIFO when *write*=1. The storage takes place at the rising edge of the clock. The read operation at the rising edge of the clock when *read*=1.

The signals *empty* and *full* indicate when the FIFO is empty or full, respectively. A read operation when *empty* is invalid. A write operation is invalid when the FIFO is full and no read operation occurs simultaneously. On an invalid write, the input data is ignored.

The following waveform illustrates the behavior of the FIFO. The last row reports the number of elements stored in the FIFO at each cycle.



Specification

module *fifo*(*clk*, *rst*, *write*, *data_in*, *read*, *data_out*, *full*, *empty*);
    input *clk*;
    input *rst*;
    input *write*;
    input [1:0] *data_in*;
    input *read*;
    output [1:0] *data_out*;
    output *full*;
    output *empty*;

Input

- *clk*, *rst*: Clock and synchronous reset signals.
- *write*: If active during the rising edge, signals that the data currently on *data_in* is to enter the queue. If the queue is full, it does nothing (except if *read* is also active on the same cycle).

- *data_in* is the data to be added to the queue.
- *read*: If active during the rising edge, signals that we want to remove the item currently at the head of the queue. If the queue is empty, it does nothing.

Output

- *data_out* is the current item at the head of the queue. If the queue is empty (*empty*=1) or *read*=0, the value of *data_out* is a don't care.
- *full* , *empty* area each a one bit signal that show the current state of the FIFO.

Problem information

Author: Javier de San Pedro Martín

Generation: 2026-07-14T06:45:07.464Z

© *Jutge.org*, 2006–2026.
https://jutge.org