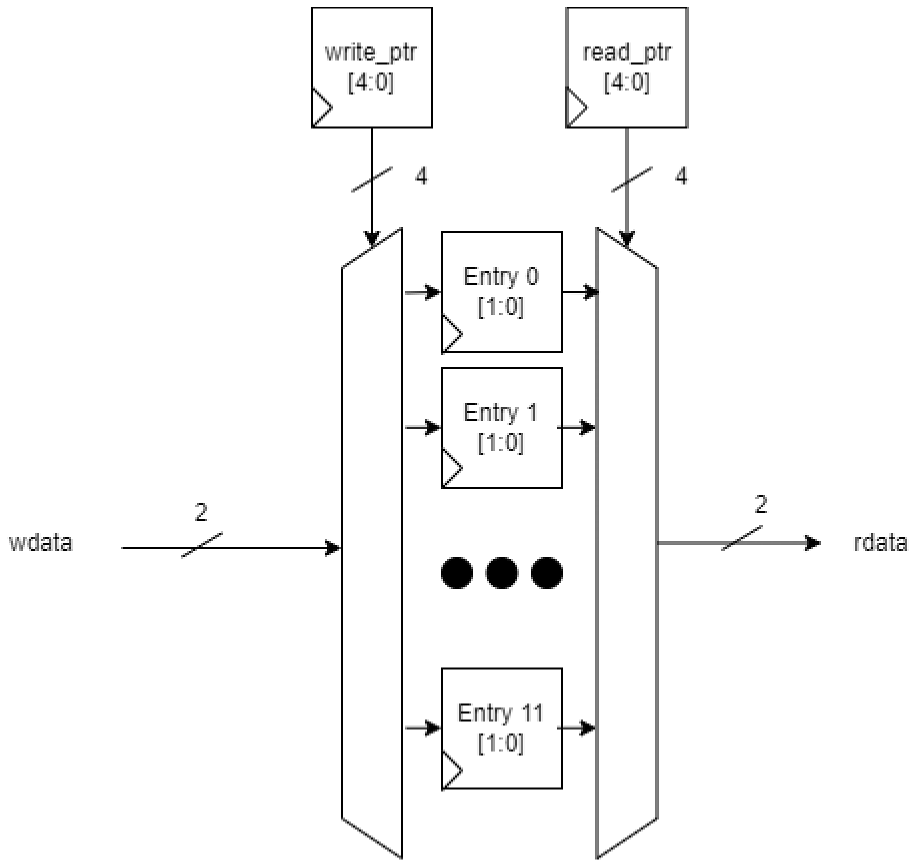
The purpose of this lab is to teach you how to use Verilator for verification of an RTL design. Verilator allows you to create testbenches in C++, which has much more flexibility than Verilog. For example, you can create a golden model in C++ and compare its outputs with your design.

For this lab, you will be verifying a FIFO. A handful of bugs have been added to it, and you must identify and fix these bugs. Below is a block diagram of the FIFO. When the FIFO is full, it should not accept any more writes. It is assumed that the reader will ignore rdata when empty is asserted.



Note: This block diagram is incomplete. Certain details such as how the read and write pointers are incremented and reset, control and status signals, and what constants correspond to, are not included. Part of this lab is figuring out these details through the Verilog, but in a real design you should include these details in your documentation.

A basic framework for a testbench has been provided. Some include statements necessary for compilation of the testbench have been removed, so you should start by reading the slides included in this lab and then adding the required include statements.

The testbench is formatted to use case statements. This simplifies finding the absolute time of an event, unlike a delay statement based testbench. The variable expected_queue can be used as a golden model, but you must think carefully about where and when to update this queue. The behavior of the FIFO is simple enough that you could update the queue using a handful of variables and some logic, but you could wrap all of this into a class instance instead.

When you find a bug in the FIFO, you should document it such that someone who is not familiar with the design and has only seen the basic diagram can understand. Snapshots of waves or examples are encouraged. You should also document how you fixed the bug. Additionally, your tests should be designed to maximize coverage (keep an eye out for a coverage lab later).

Lab objectives:

- Fix the C++ testbench
- Find, document, and fix all bugs in the FIFO