Questa SystemVerilog Testbench

LAB 1: Getting Started with SV Testbench

Goal Write a simple testbench for a 2-port arbiter
Get familiar with:
- interfaces,
- clocking blocks,
- program blocks,
- driving,
- sampling

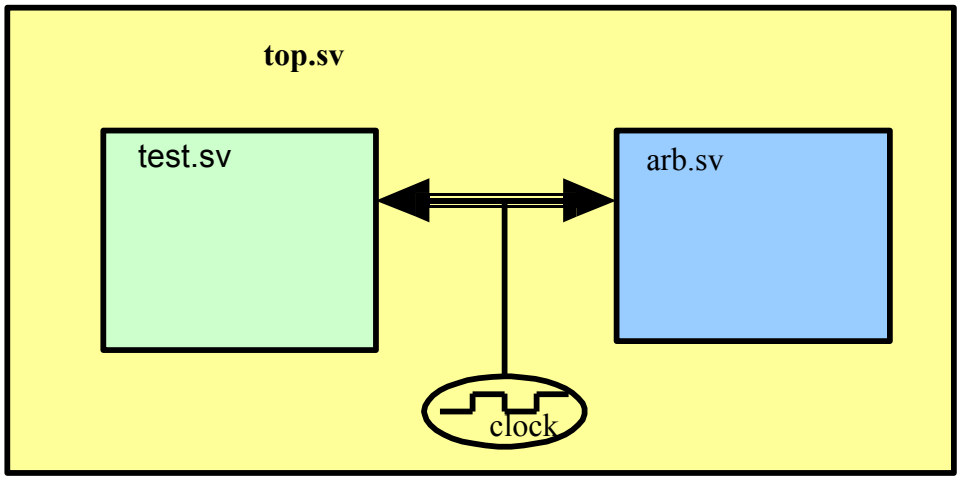
Location From the course Website, download the file lab1.tar.gz
gunzip lab1.tar.gz
tar -xvf lab1.tar

Design Arbiter

Relevant Files in lab1 directory:

`hdl/arb.sv`	Arbiter DUT
`hdl/arb_if.sv`	Arbiter interface
`hdl/top.sv`	Top level program
`tests/test.sv`	Testbench

Testbench Environment



Steps to hook up a DUT to a Testbench

1. Create DUT interface with modports and clocking blocks
2. Create testbench program
3. Create top module
4. Compile and run

1) *Complete the interface definition* including the clocking block **`hdl/arb_if.sv`**
   ➔ Signal descriptions of the arbiter are defined in **`arb.sv`**
   ➔ Use **`logic`** type for interface signals.
2) *Complete the task* ***`reset_test`*** *in the file* ***`tests/test.sv`***
   Assert the reset signal (active high), make sure you are not making any requests for two cycles, and then check that the grant is all zeroes one cycle later.
3) *Complete the task* **`request_grant_test`** in **`tests/test.sv`**
   You should check that arbiter works correct for a single request on both channels, and for multiple requests. The arbiter implements a round-robin arbitration algorithm.
   Use in-line assertions as shown in **`test.sv`**

** Search for "**`LAB`**" in the lab files to see where to add your code.