LUCID VLSI

Learning Unlimited. Coding InDelible.

9

SystemVerilog LAB9

After completing this lab, you should be able to:

Implement clocking blocks in interface.

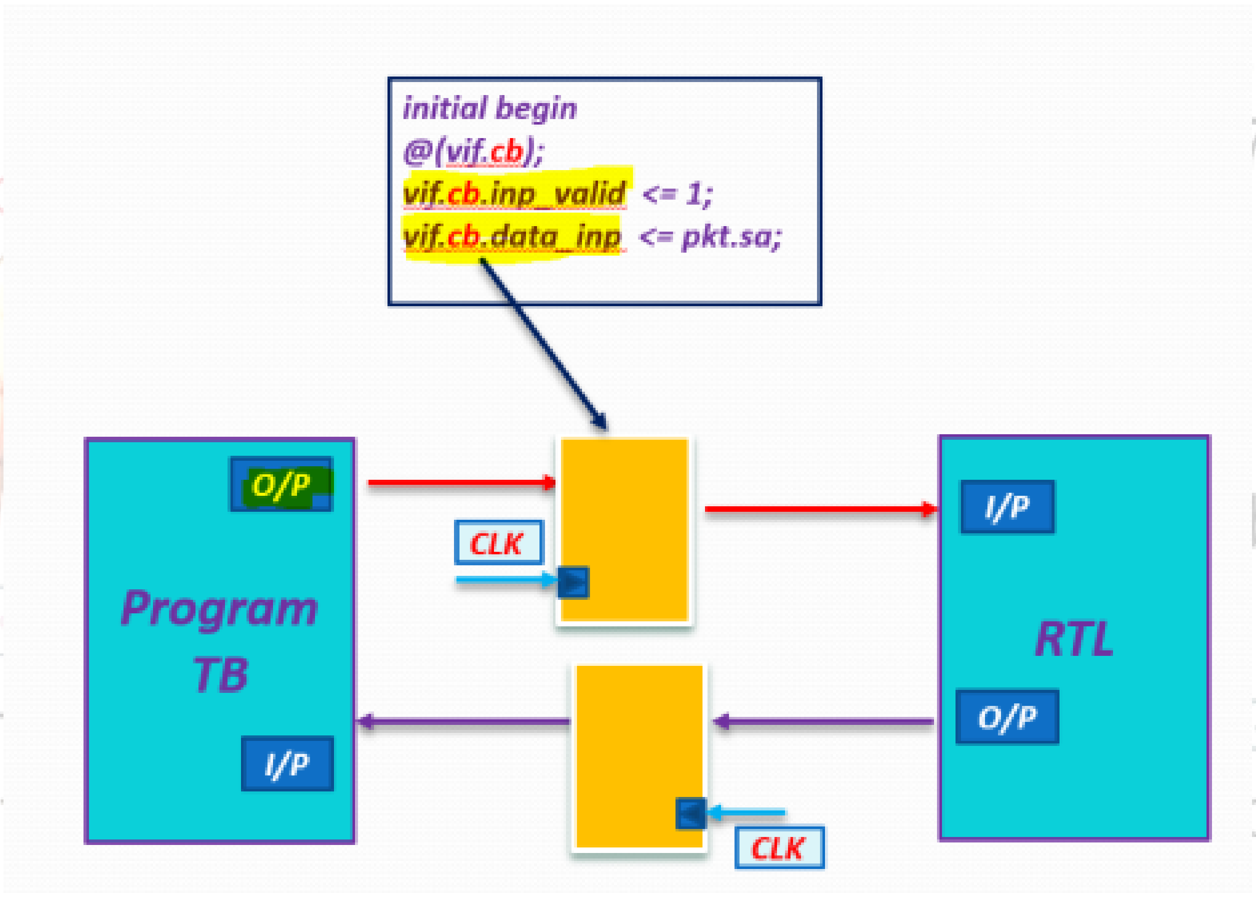
Understand how to import clocking block to modport.

Understand how to access signals of clocking block through interface port in TB.

Understand how to drive stimulus using clocking block drive operations.

Understand how to sample dut output signals using clocking block in TB.

Verify DUT behavior with the help of self-checking mechanism in program block(testbench).



```
initial begin
@(vif.cb);
vif.cb.inp_valid <= 1;
vif.cb.data_inp <= pkt.sa;
```

LAB9: Copy lab8 directory as lab9

Step 1: Define clock port in interface port list with input direction.

1) Open file top.sv and define clk as input in interface port list.
   Add this code in Section 7 in top.sv
   Ex:

```
//Section 7: Define interface with clk as input
interface router_if(input clk);
```

Step 2: Define clocking block in interface with required signals.

1) Define clocking block with required signals and with appropriate directions w.r.t TB. Add this code Section 10 of top.sv file.
   1) Specify direction output if you want to drive signal from TB.
   2) Specify direction input if you want to sample signal from TB.

Ex: //Section 10 :Define the clocking block

```
clocking cb @(posedge clk);
  output dut_inp;    //Direction are w.r.t TB
  output inp_valid; // drive signal from TB
  input dut_outp;
  input outp_valid; // sample signal from TB
  input busy;
  input error;
endclocking
```

Step 3: Import clocking block into modport.

1) Import clocking block into modport.
   Add this code in Section9 in top.sv

Old modport: //Section 9:Define modport for TB
```
modport tb_mod_port (output reset,dut_inp,inp_valid,
                     input outp_valid,dut_outp,busy,error);
```

New modport: import clocking block
```
modport tb_mod_port (clocking cb , output reset);
```

Step 4: Access all dut (ports) signals using vif port.

1) Open testbench.sv and replace all dut signals access with vif as vif.cb.
   We will have to drive/sample signals which are part of clocking block (in interface).

Ex:

Old code: Signals are part of interface Port vif.
```
vif.reset  <= 1;
vif.inp_valid <=1;
vif.dut_inp <=1;
wait(vif.busy==0);
```

New Code: Signals are part of clocking block in interface.
```
vif.reset  <= 1; //reset is not part of clocking block
vif.cb.inp_valid <=1;
vif.cb.dut_inp <=1;
wait(vif.cb.busy==0);
```

Step 5: Replace @(posedge clk) with @(vif.cb) in testbench.sv

1) Open testbench.sv and replace all @(posedge clk) with @(vif.cb).
   Ex:
   Old code: Waiting on clock cycles
   ```
   @(posedge clk);
   repeat(5) @(posedge clk);
   ```
   New Code: Waiting on clocking block
   ```
   @(vif.cb);
   repeat(5) @(vif.cb);
   ```

Step 6: Pass clk port to interface instance (port list of interface instance).

1) Specify the clk port in interface instance port list in top.sv file.
   Add this code in Section 8 of top.sv
   ```
   //Section 8: Instantiate interface
   router_if   router_if_inst (clk);
   ```

Step 7: Run the simulation and validate the output of DUT with the results printed by self-checking mechanism.

1. Run the simulation and check the test Passed or Failed and debug the if there are any failures.