Portland State
UNIVERSITY

ECE593: CROSS REFERENCE DOCUMENT

TEAM MEMBERS:

1. Mangala Bhat (PSU ID: 969559278)
1. Piyush Purwat(PSU ID: 941496418)
2. Venkata Sai Pranav Twarakavi (PSU ID: 951485914)

Portland State
UNIVERSITY

1. **Top.sv:**

   Description: Top module containing instantiation of DUV ,interface and test.Additionally, coverage module binding is done to this module using bind module.

2. **Transaction.sv:**

   Description: This class contains all necessary primary ports to generate stimulus, additionally three dynamic arrays to generate required sequences beforehand and pulse jtag ports by iterating through the array are defined.

3. **Generator.sv:**

   Generator is class that supplies data to driver. It consists of a constructor and a task gen_run which randomization an instance of the transaction class.

   Functions used:

   new: Inputs to this function are mailbox and event handles , this function acts as a constructor.

   Tasks used:

   gen_run:
   -Input to this task is an integer
   -The randomize function is executed the same number of times as given by the value of the integer

-It also triggers an event called entrans once the required stimulus is driven

4. Driver.sv:

Driver is a class which is responsible for driving pin level data to design under verification , it consists of a constructor and a task drv_run , this task drives input stimulus obtained from the generator through the interface .

Functions used:

new: Inputs to this function are interface handles , this function acts as a constructor.

Tasks used:

Reset: Inputs to this task are interface handles,it waits for reset assertion and prints whether DUV was subjected to reset.

drv_transmit: The most significant task of the driver , this task is responsible for driving the stimulus at every clock edge to the interface . It drives the TMS pins to program an instruction in the register and drives required TDI depending on the decoded instruction, one by one from the randomized scan chain.This task forks off 2 thread for its purpose.

drv_run: This is a top level task in the driver , it forks 2 threads , 1 thread waits for reset while the other thread executes the drv_transmit task .This block of code is present in a fork ..join_any loop , therefore the execution come out if reset is asserted and the running process is disabled by disable fork construct.

-

5. **Jtag_intf.sv**:

The interface contains all ports of the DUV and the main function is to facilitate driving data to the DUV since classes are dynamic in nature. A virtual instance of this interface is used in different classes to drive or observe values in the interface.

6. Monitor.sv:

This class is responsible for taking output data of the DUV from the interface and send them to scoreboard using a mailbox, it collects data from the DUV using the task mon_run.

Functions used:

new: Inputs to this function are mailbox handles , this function acts as a constructor.

Tasks used:

RecieveFromInf: inputs to this task constitute a virtual interface handle and a transaction class handle.This task is responsible for collecting output stimulus from the interface.

Mon_run: This task takes two intergers as inputs ,these integers correspond to the size of dynamic arrays generated in accordance with the scan chain length in the transaction class , this task is used to put data into the mailbox at appropriate times , this data is made available to the scoreboard.

7. Scoreboard.sv:

This class is responsible for taking comparing the data obtained from the monitor and reference model and throw errors if any mismatch is present in output patterns.

Functions used:

new: Inputs to this function are mailbox handles , this function acts as a constructor

Tasks used:

Scb_run: This task has two integer inputs which correspond o the scan chain length targeted and it uses these two values in a for loop to get values from mailbox and compare them to check if the DUV outputs and reference model outputs depict the same functionality.

8. Reference_model.sv:

This module is a reference intended to be used for checking against the design, it is modelled according to specification using always blocks.

9. Environment.sv:

This class is used as a container for all the other classes and are driven in the order required
To construct and provide orderly stimulus to the design.

Functions used:

new: Inputs to this function are mailbox handles , this function acts as a constructor

build: This function creates objects for all class handles and provides required arguments for mailboxes.

Tasks used:

Scb_run : This task has two inputs for instruction register and data register widths which will be subsequently passed to monitor and scoreboard.The task executes core functionality from all the classes contained by it in order (generator,driver,monitor and scoreboard).Additionally it calls endoftest task.

endoftest: This task contains three wait statements (event from monitor,loop control variables from driver and scoreboard) to maintain synchronization.

10.Coverage.sv:

This class contains cover groups to extract coverage for the following scenarios,

Pins : This covergroup consists of coverpoint covering all primary ports and whether

All the jtag pins are toggling

Reset1: This covergroup checks whether reset is asserted when there are a sequence of 5 1's is observed at TMS.

TDI_TDO_cov: This covergroup checks whether there is toggle in bith TDI and TDO

Pins.

fsm_cov: This covergroup checks if an instance of all the states that the tap controller is capable of traversing for different instructions and it is crossed with tms value

int_reg_cov: This covergroup checks if all internal registers are hit in the simulation.

parallel_case: This covergroup check whether different instructions have been latched in the internal decode register.

11.**Test.sv**:

This file contains all sets of constraint random test cases implemented in the project..