Joshua Monson
ECEn 620 Assignment 5
10/9/2014

Overview

In this assignment I added a package to the memory model test bench that includes a transaction class (which replaces the transaction struct used in the last assignment). The class combines all of the data from the previously used transaction struct and variable printing and error checking included in the previous testbench. In addition we have also added a copy function and a constructor that generates random data and a unique ID for each transaction.

This report includes a printout of the System Verilog code for the updated test bench and transaction package. I have also include a waveform and transcript output files.

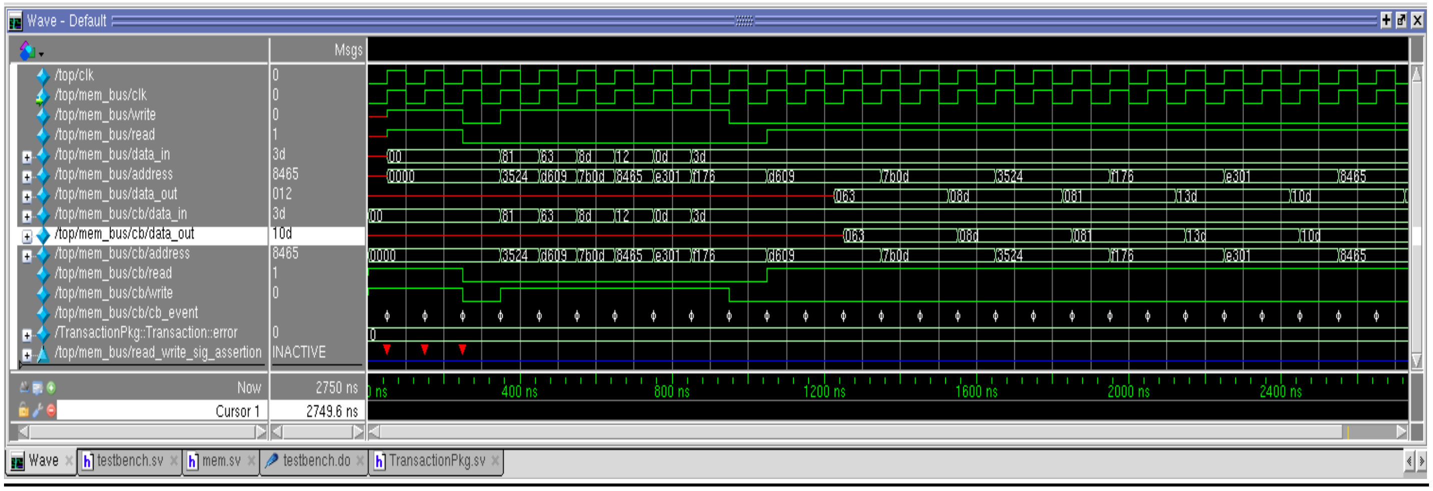
Test Bench Code

```
`default_nettype none
`timescale 1ns/100ps

import TransactionPkg::*;

program automatic test(mem_if.TEST mem_bus);


  Transaction Trans[];
  initial begin

    //Initialize Score Board
    Trans = new[6];

    //Generate Random Writes
    foreach(Trans[i]) begin
      Trans[i] = new();
    end

    mem_bus.cb.read <= 0;
    //Test Protocol Error Checker
    //This should produce three Errors
    mwrite(0, 0);
    mread(0);

    @mem_bus.cb;
    @mem_bus.cb
    @mem_bus.cb

    mem_bus.cb.write <= 0;
    mem_bus.cb.read  <= 0;

    //Perform Writes
    foreach(Trans[i]) begin
      write_memory(Trans[i]);
    end
    @(mem_bus.cb);
    mem_bus.cb.write <= 0;

    //Rearrange Array for Reads
    Trans.shuffle();

    //Perform Reads and Check Results
    foreach(Trans[i]) begin
     read_memory(Trans[i]);
     Trans[i].check();
    end
    mem_bus.cb.read <= 0;

    //Display the Actual Values Read
    $display("Print Read Values");
    foreach(Trans[i]) begin
     Trans[i].print_actual_read();
    end

    $display("Tests Performed: %d", Trans.size());
    Transaction::print_error_count();

  end

  final begin
    $display("Test Complete.");
  end

  ///////////////////////////
  //     Read Function     //
  ///////////////////////////
  //Note: does not return read value
  function void mread(logic [15:0] addr);
   mem_bus.cb.read <= 1;
   mem_bus.cb.address <= addr;
  endfunction

  ///////////////////////////
  //    Write Function     //
  ///////////////////////////
  function void mwrite(logic [15:0] addr, logic [7:0] data);
   mem_bus.cb.write <= 1;
   mem_bus.cb.data_in <= data;
   mem_bus.cb.address <= addr;
  endfunction


  task automatic write_memory(ref Transaction T);
    @(mem_bus.cb);
    mwrite(T.address, T.data_to_write);
  endtask

  task automatic read_memory(ref Transaction T);
    @(mem_bus.cb);
    mread(T.address);
    @(mem_bus.cb);
    @(mem_bus.cb);
    T.actual_read = mem_bus.cb.data_out;
  endtask

endprogram


//////////////////////
// Top-Level Module //
//////////////////////

module top;

  //Clock Generator
  bit clk = 0;
  always #50ns clk = ~clk;

  //Memory Interface
  mem_if mem_bus(clk);
  //Test Program
  test t1(mem_bus);
  //Memory Model
  my_mem mem(mem_bus);

endmodule
```

Transaction Package Code

```
package TransactionPkg;

  class Transaction;
    static int error = 0;
    static int current_id = 0;
    int id;
    shortint address;
    byte data_to_write;
    bit [8:0] expected_read;
    bit [8:0] actual_read;

    function new();
      //Assign new id
      this.id = current_id++;
      //Assign Random Address
      this.address = $random;
      //Assign Random Data
      this.data_to_write = $random;
      this.expected_read = {^this.data_to_write, this.data_to_write};
    endfunction

    function void print_actual_read();
      $display("Transaction %d: @ time %d actual read from address %04X is
%02X", this.id, $time, this.address, this.actual_read);
    endfunction

    function void check();
      if( this.expected_read != this.actual_read) begin
    $display("Transaction %d Check Error @ time %d: Read value %02X from
address %04X, expected %02X", this.id, $time, this.actual_read, this.address,
this.expected_read);
    error++;
   end
  endfunction

  function Transaction copy();

   Transaction ret = new();
   ret.address = this.address;
   ret.data_to_write = this.data_to_write;
   ret.expected_read = this.expected_read;
   ret.actual_read = this.actual_read;
   return ret;

  endfunction

  static function void print_error_count();
   $display("Total Transaction Errors @ time %d: %d", $time, error);
  endfunction

 endclass

endpackage
```

Waveform



Transcripts Showing 3 Assertion Errors and Zero Read Errors

# vsim -novopt top
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.top
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.testbench_sv_unit
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.TransactionPkg

```
# Loading sv_std.std
# Loading work.TransactionPkg
# Loading work.testbench_sv_unit
# Loading work.top
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.mem_if
# Loading work.mem_if
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.test
# Loading work.test
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.my_mem
# Loading work.my_mem
```

# ** Error: Assertion error.

```
#    Time: 50 ns Started: 50 ns  Scope: top.mem_bus.read_write_sig_assertion File:
/net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/sverilog/mem.sv Line: 23
```

# ** Error: Assertion error.

```
#    Time: 150 ns Started: 150 ns  Scope: top.mem_bus.read_write_sig_assertion File:
/net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/sverilog/mem.sv Line: 23
```

# ** Error: Assertion error.

```
#    Time: 250 ns Started: 250 ns  Scope: top.mem_bus.read_write_sig_assertion File:
/net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/sverilog/mem.sv Line: 23

# Print Read Values

# Transaction          1: @ time             27500 actual read from address d609 is 63
# Transaction          2: @ time             27500 actual read from address 7b0d is 8d
# Transaction          0: @ time             27500 actual read from address 3524 is 81
# Transaction          5: @ time             27500 actual read from address f176 is 13d
# Transaction          4: @ time             27500 actual read from address e301 is 10d
# Transaction          3: @ time             27500 actual read from address 8465 is 12

# Tests Performed:          6
# Total Transaction Errors @ time             27500:          0
```

Transcripts Showing Read Errors

```
# vsim -novopt top
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.top
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.testbench_sv_unit
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.TransactionPkg
# Loading sv_std.std
# Loading work.TransactionPkg
# Loading work.testbench_sv_unit
# Loading work.top
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.mem_if
# Loading work.mem_if
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.test
# Loading work.test
# Refreshing /net/fpga1/users/joshuas2/ECEn620/Assignments/Assignment5/work.my_mem
# Loading work.my_mem
# Transaction          1 Check Error @ time               9500: Read value 00 from address d609, expected 63
# Transaction          2 Check Error @ time              12500: Read value 00 from address 7b0d, expected 8d
# Transaction          0 Check Error @ time              15500: Read value 00 from address 3524, expected 81
# Transaction          5 Check Error @ time              18500: Read value 00 from address f176, expected
13d
# Transaction          4 Check Error @ time              21500: Read value 00 from address e301, expected
10d
# Transaction          3 Check Error @ time              24500: Read value 00 from address 8465, expected 12

# Print Read Values

# Transaction          1: @ time              24500 actual read from address d609 is 00
# Transaction          2: @ time              24500 actual read from address 7b0d is 00
# Transaction          0: @ time              24500 actual read from address 3524 is 00
# Transaction          5: @ time              24500 actual read from address f176 is 00
# Transaction          4: @ time              24500 actual read from address e301 is 00
# Transaction          3: @ time              24500 actual read from address 8465 is 00

# Tests Performed:          6
# Total Transaction Errors @ time              24500:          6
```