Assignment 5

TOP :

ALSU_top.sv

```
1   import uvm_pkg::*;
2   `include "uvm_macros.svh"
3   import ALSU_test_pkg::*;
4
5   module ALSU_top ();
6       bit clk;
7
8       initial begin
9           forever #1 clk = ~clk;
10      end
11
12      ALSU_if alsuif (clk);
13      ALSU DUT (alsuif);
14
15      initial begin
16          uvm_config_db#(virtual ALSU_if)::set(null,"uvm_test_top","ALSU_IF",alsuif);
17          run_test ("ALSU_test");
18      end
19
20  endmodule
```

CONFIG:

ALSU_config.sv

```
1   package ALSU_config_pkg;
2   import uvm_pkg::*;
3   `include "uvm_macros.svh"
4
5       class ALSU_config extends uvm_object;
6           `uvm_object_utils(ALSU_config)
7
8           virtual ALSU_if ALSU_vif;
9
10          function new (string name = "ALSU_config");
11              super.new(name);
12
13          endfunction
14
15      endclass
16  endpackage
```

TEST :

ALSU_test.sv

```
1  package ALSU_test_pkg;
2  import ALSU_env_pkg::*;
3  import ALSU_config_pkg::*;
4  import uvm_pkg::*;
5  `include "uvm_macros.svh"
6
7  class ALSU_test extends uvm_test;
8    `uvm_component_utils(ALSU_test)
9
10   ALSU_env env;
11   ALSU_config ALSU_cfg;
12   virtual ALSU_if ALSU_vif;
13
14   function new (string name = "shift_reg_env" , uvm_component parent = null);
15     super.new (name,parent);
16   endfunction
17
18   function void build_phase (uvm_phase phase);
19     super.build_phase (phase);
20     env = ALSU_env::type_id::create("env",this);
21     ALSU_cfg = ALSU_config::type_id::create("ALSU_cfg");
22
23     if (! uvm_config_db #(virtual ALSU_if)::get(this,"","ALSU_IF",ALSU_cfg.ALSU_vif))
24       `uvm_fatal("build_phase"," unable to get the virtual interface ");
25
26       uvm_config_db#(ALSU_config)::set(this,"*","CFG",ALSU_cfg);
27
28   endfunction
29
30   task run_phase (uvm_phase phase);
31     super.run_phase(phase);
32     phase.raise_objection(this);
33     #100; `uvm_info("run_phase","welcome to the uvm",UVM_MEDIUM)
34     phase.drop_objection(this);
35
36   endtask //run_phase
37
38 endclass
39 endpackage
```

ENV :

ALSU_env.sv

```
1   package ALSU_env_pkg;
2   import ALSU_driver_pkg::*;
3   import uvm_pkg::*;
4   `include "uvm_macros.svh"
5
6   class ALSU_env extends uvm_env;
7     `uvm_component_utils (ALSU_env);
8
9     ALSU_driver driver;
10
11    function new (string name = "shift_reg_env" , uvm_component parent = null);
12      super.new (name,parent);
13    endfunction
14
15    function void build_phase(uvm_phase phase);
16      super.build_phase(phase);
17      driver = ALSU_driver::type_id::create("driver",this);
18    endfunction
19
20  endclass
21
22  endpackage
```

INTERFACE :

ALSU_if.sv

```
1   interface ALSU_if (input clk);
2   logic cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
3   logic [2:0] opcode;
4   logic signed [2:0] A, B;
5   logic [15:0] leds;
6   logic signed [5:0] out;
7
8   modport DUT (
9   input clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
10  output leds,out);
11
12  modport TEST (
13  output cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
14  input clk,leds,out);
15
16  endinterface //ALSU_if
```

DRIVER :

ALSU_driver.sv

```
package ALSU_driver_pkg;
import ALSU_config_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

    class ALSU_driver extends uvm_driver;
        `uvm_component_utils(ALSU_driver)

  virtual ALSU_if ALSU_vif;
  ALSU_config ALSU_cfg;

  function new (string name = "ALSU_driver" , uvm_component parent = null);
    super.new (name,parent);
  endfunction

  function void build_phase (uvm_phase phase);
    super.build_phase(phase);

    if (!uvm_config_db#(ALSU_config)::get(this,"","CFG",ALSU_cfg))
        `uvm_fatal ("build_phase","unable to get configuration object");
  endfunction

  function void connect_phase(uvm_phase phase);
    super.connect_phase(phase);
    ALSU_vif = ALSU_cfg.ALSU_vif;  /////
  endfunction

  task run_phase (uvm_phase phase);
    super.run_phase(phase);
      ALSU_vif.rst = 0;
      ALSU_vif.cin = 0;
      ALSU_vif.red_op_B = 0;
      ALSU_vif.red_op_A = 0;
      ALSU_vif.bypass_B = 0;
      ALSU_vif.bypass_A = 0;
      ALSU_vif.direction = 0;
      ALSU_vif.serial_in = 0;
      ALSU_vif.opcode = 0;
      ALSU_vif.A = 0;
      ALSU_vif.B = 0;
      @(negedge ALSU_vif.clk);
      ALSU_vif.rst = 1;

      forever begin
          @(negedge ALSU_vif.clk);
          ALSU_vif.A = $random;
          ALSU_vif.B = $random;
      end
```

DO :

ALSU_do.do

```
1  vlib work
2  vlog -f ALSU_files.list
3  vsim -voptargs=+acc work.ALSU_top -classdebug -uvmcontrol=all
4  add wave /ALSU_top/alsuif/*
5  run -all
```

DISPLAY :