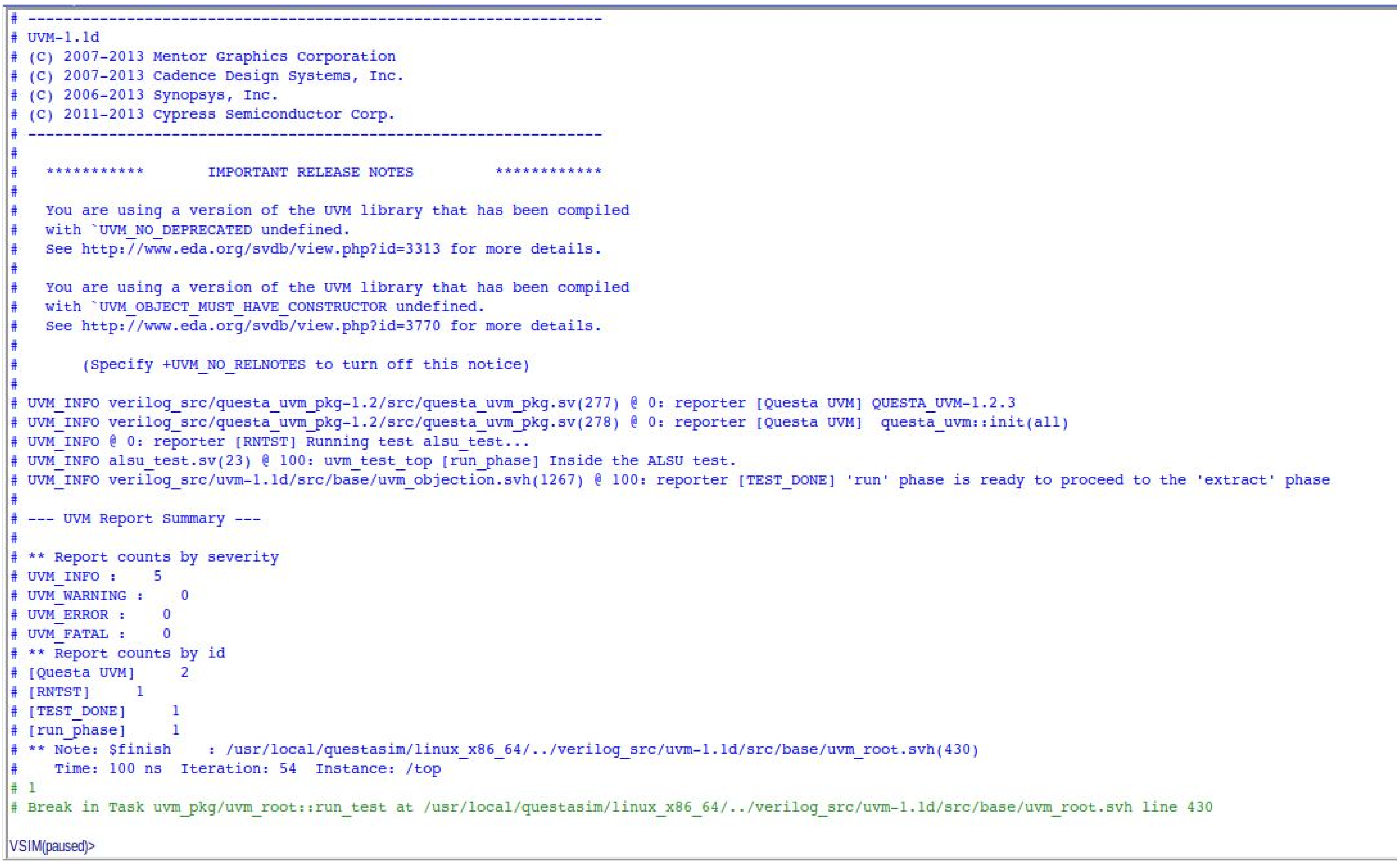
Assignment 5 UVM

Digital Design Verification

Contents

1 Part 1



```
# ----------------------------------------------------------------
# UVM-1.1d
# (C) 2007-2013 Mentor Graphics Corporation
# (C) 2007-2013 Cadence Design Systems, Inc.
# (C) 2006-2013 Synopsys, Inc.
# (C) 2011-2013 Cypress Semiconductor Corp.
# ----------------------------------------------------------------
#
#   ***********       IMPORTANT RELEASE NOTES         ************
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_NO_DEPRECATED undefined.
#   See http://www.eda.org/svdb/view.php?id=3313 for more details.
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_OBJECT_MUST_HAVE_CONSTRUCTOR undefined.
#   See http://www.eda.org/svdb/view.php?id=3770 for more details.
#
#       (Specify +UVM_NO_RELNOTES to turn off this notice)
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test alsu_test...
# UVM_INFO alsu_test.sv(23) @ 100: uvm_test_top [run_phase] Inside the ALSU test.
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 100: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :    5
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [run_phase]     1
# ** Note: $finish    : /usr/local/questasim/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 100 ns  Iteration: 54  Instance: /top
# 1
# Break in Task uvm_pkg/uvm_root::run_test at /usr/local/questasim/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430

VSIM(paused)>
```

Figure 1: Transcript

2 Part 2

2.1 top_module

```
import alsu_test_pkg::*;
import uvm_pkg::*;
‘include "uvm_macros.svh"
‘include "alsu_if.sv"

module top();
   bit clk;

   initial begin
     forever
        #1 clk = ~clk;
   end

   alsu_if alsuif (clk);

   ALSU #(.INPUT_PRIORITY("A"), .FULL_ADDER("ON")) DUT (
      .clk          (alsuif.DUT.clk),
      .cin          (alsuif.DUT.cin),
      .rst          (alsuif.DUT.rst),
      .red_op_A     (alsuif.DUT.red_op_A),
      .red_op_B     (alsuif.DUT.red_op_B),
      .bypass_A     (alsuif.DUT.bypass_A),
      .bypass_B     (alsuif.DUT.bypass_B),
      .direction    (alsuif.DUT.direction),
      .serial_in    (alsuif.DUT.serial_in),
      .opcode       (alsuif.DUT.opcode),
      .A            (alsuif.DUT.A),
      .B            (alsuif.DUT.B),
      .leds         (alsuif.DUT.leds),
      .out          (alsuif.DUT.out)
   );

   initial begin
      uvm_config_db#(virtual alsu_if)::set(null, "uvm_test_top", "ALSU_VIF", alsuif);
      run_test("alsu_test");
   end
endmodule
```

2.2 alsu_test

```
package alsu_test_pkg;
import alsu_env_pkg::*;
import uvm_pkg::*;
‘include "uvm_macros.svh"

    class alsu_test extends uvm_test;
       ‘uvm_component_utils(alsu_test)
```

```

        alsu_env env;
        virtual alsu_if alsu_test_vif;

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

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

            if (!uvm_config_db#(virtual alsu_if)::get(this, "", "ALSU_VIF", alsu_test_vif))
            `uvm_fatal("NOVIF", "Virtual␣interface␣alsu_test_vif␣was␣not␣found␣in␣the␣configuration␣database");

        uvm_config_db#(virtual alsu_if)::set(this,"*","ALSU_VIF",alsu_test_vif);

        env = alsu_env::type_id::create("env",this);
        endfunction

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            phase.raise_objection(this);
            #100; `uvm_info("run_phase","Inside␣the␣ALSU␣test.",UVM_MEDIUM);
        phase.drop_objection(this);
        endtask : run_phase

    endclass : alsu_test

endpackage
```

2.3 alsu_if

```
interface alsu_if (
   input bit clk
);

  logic rst;
  logic cin;
  logic red_op_A;
  logic red_op_B;
  logic bypass_A;
  logic bypass_B;
  logic direction;
  logic serial_in;
  logic signed [2:0] A;
  logic signed [2:0] B;
  logic [2:0]        opcode;
  wire [15:0]        leds;
  wire signed [5:0]  out;

modport DUT (
    input clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
    output leds, out
);

modport TEST (
    output clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
    input leds, out
);

endinterface
```

2.4 alsu_env

```
package alsu_env_pkg;
import alsu_driver_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

    class alsu_env extends uvm_env;
        `uvm_component_utils(alsu_env)

        alsu_driver driver;

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

        function void build_phase(uvm_phase phase);
        super.build_phase(phase);
        driver = alsu_driver::type_id::create("driver",this);
        endfunction

    endclass : alsu_env

endpackage
```

2.5 alsu_driver

```
package alsu_driver_pkg;

import uvm_pkg::*;
`include "uvm_macros.svh"

      class alsu_driver extends uvm_driver;
      `uvm_component_utils(alsu_driver)

      virtual alsu_if alsu_driver_vif;

      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#(virtual alsu_if)::get(this, "", "ALSU_VIF", alsu_driver_vif)) begin
          ‘uvm_fatal("NOVIF", "Virtual␣interface␣alsu_driver_vif␣was␣not␣found␣in␣the␣configuration␣database")
        end
      endfunction

      task run_phase(uvm_phase phase);
        phase.raise_objection(this);


        alsu_driver_vif.rst = 1;      // Assert reset
        repeat (2) @(posedge alsu_driver_vif.clk); // Wait for a couple of clock cycles
        alsu_driver_vif.rst = 0;      // Deassert reset


        repeat (100) begin
          alsu_driver_vif.A          = $random;
          alsu_driver_vif.B          = $random;
          alsu_driver_vif.opcode     = $random;
          alsu_driver_vif.cin        = $random;
          alsu_driver_vif.red_op_A   = $random;
          alsu_driver_vif.red_op_B   = $random;
          alsu_driver_vif.bypass_A   = $random;
          alsu_driver_vif.bypass_B   = $random;
          alsu_driver_vif.direction = $random;
          alsu_driver_vif.serial_in = $random;

          // Wait for the next positive clock edge
          @(posedge alsu_driver_vif.clk);
        end

        phase.drop_objection(this);
          endtask

    endclass
endpackage
```

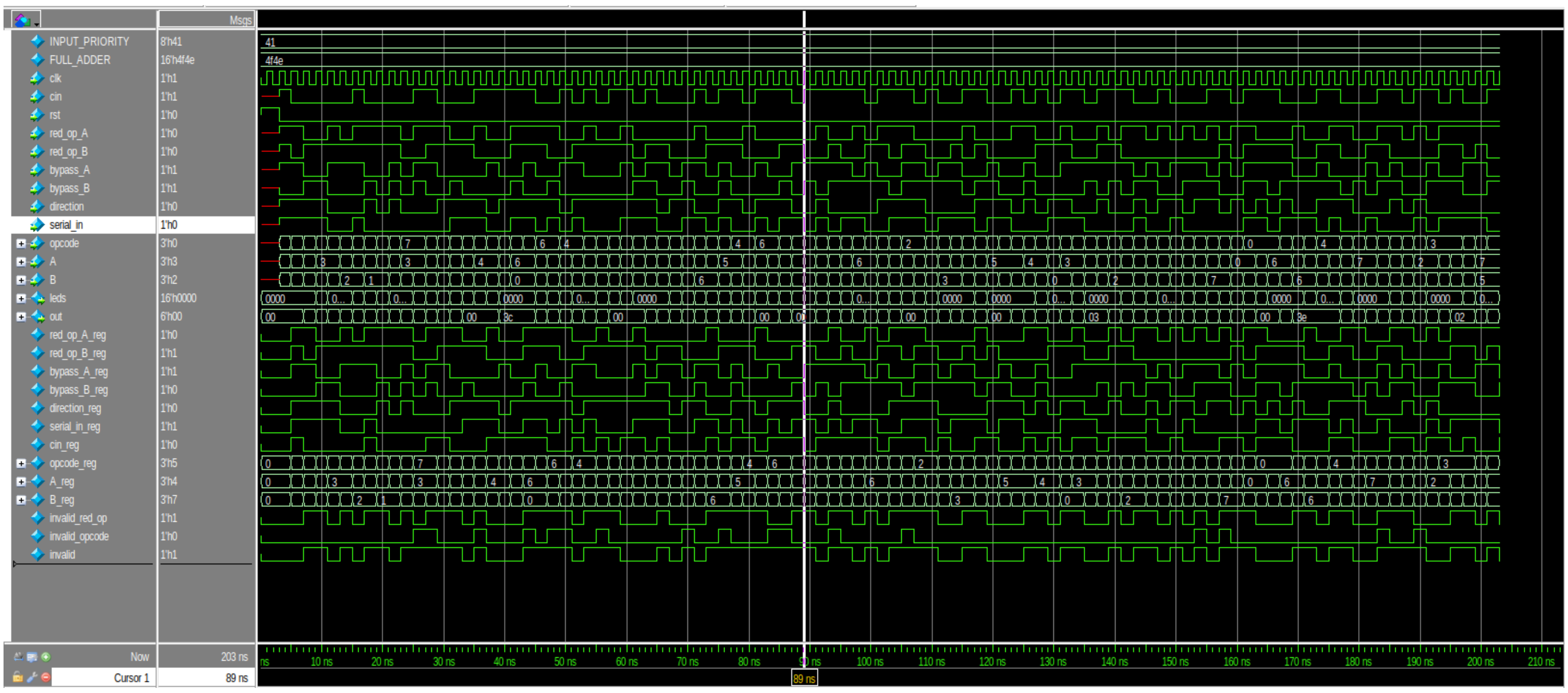
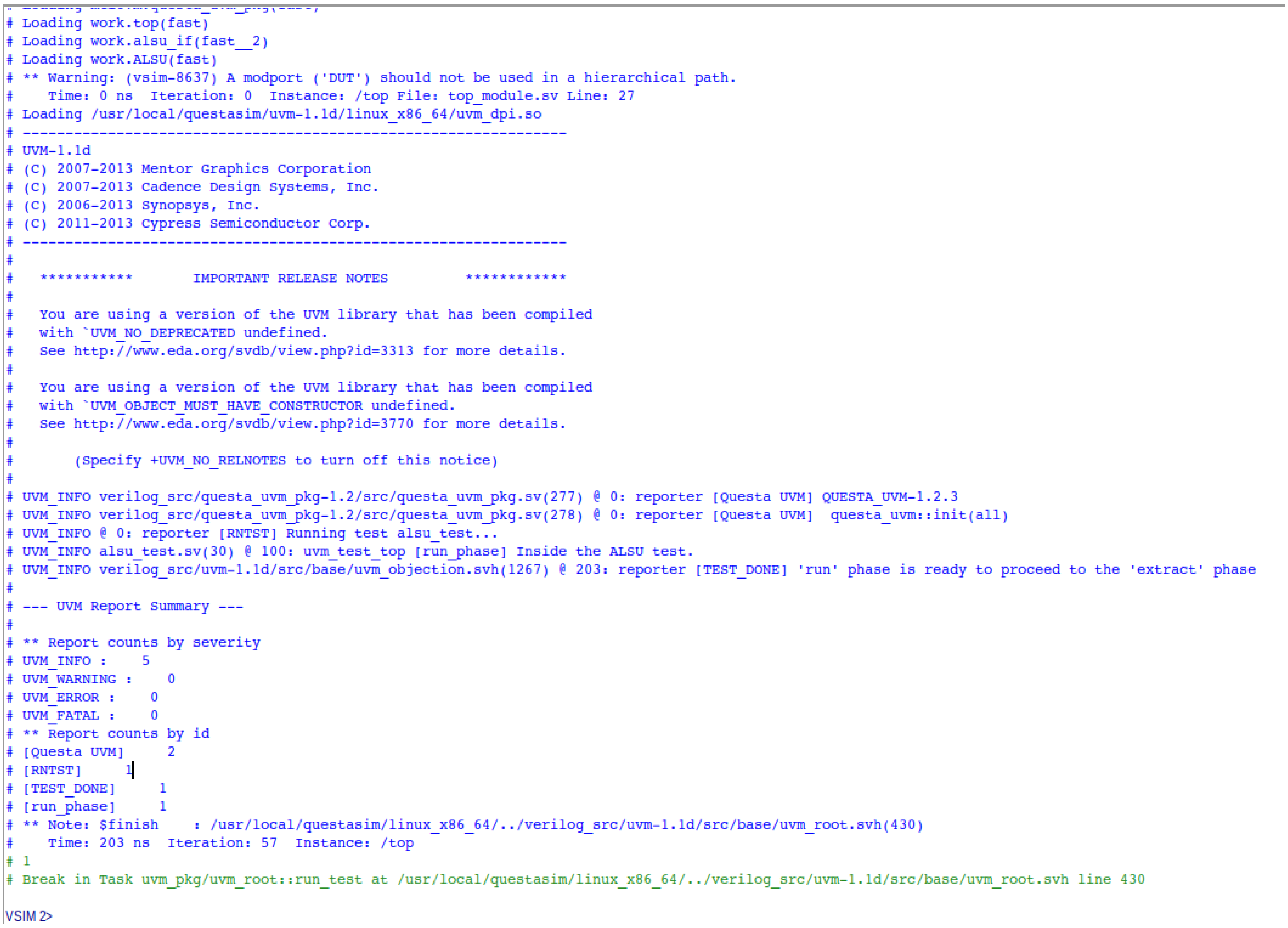


Figure 2: simulation waveform



```
# Loading mtiUvm.questa_uvm_pkg(fast)
# Loading work.top(fast)
# Loading work.alsu_if(fast__2)
# Loading work.ALSU(fast)
# ** Warning: (vsim-8637) A modport ('DUT') should not be used in a hierarchical path.
#    Time: 0 ns  Iteration: 0  Instance: /top File: top_module.sv Line: 27
# Loading /usr/local/questasim/uvm-1.1d/linux_x86_64/uvm_dpi.so
# ----------------------------------------------------------------
# UVM-1.1d
# (C) 2007-2013 Mentor Graphics Corporation
# (C) 2007-2013 Cadence Design Systems, Inc.
# (C) 2006-2013 Synopsys, Inc.
# (C) 2011-2013 Cypress Semiconductor Corp.
# ----------------------------------------------------------------
#
#   ***********       IMPORTANT RELEASE NOTES         ************
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_NO_DEPRECATED undefined.
#   See http://www.eda.org/svdb/view.php?id=3313 for more details.
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_OBJECT_MUST_HAVE_CONSTRUCTOR undefined.
#   See http://www.eda.org/svdb/view.php?id=3770 for more details.
#
#       (Specify +UVM_NO_RELNOTES to turn off this notice)
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test alsu_test...
# UVM_INFO alsu_test.sv(30) @ 100: uvm_test_top [run_phase] Inside the ALSU test.
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 203: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :    5
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [run_phase]     1
# ** Note: $finish    : /usr/local/questasim/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 203 ns  Iteration: 57  Instance: /top
# 1
# Break in Task uvm_pkg/uvm_root::run_test at /usr/local/questasim/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430

VSIM 2>
```

Figure 3: Transcript : all test cases passed

3 Part 3

3.1 top_module

```
import alsu_test_pkg::*;
import uvm_pkg::*;
‘include "uvm_macros.svh"
‘include "alsu_if.sv"

module top();
   bit clk;

   initial begin
     forever
        #1 clk = ~clk;
   end

   alsu_if alsuif (clk);

   ALSU #(.INPUT_PRIORITY("A"), .FULL_ADDER("ON")) DUT (
      .clk          (alsuif.DUT.clk),
      .cin          (alsuif.DUT.cin),
      .rst          (alsuif.DUT.rst),
      .red_op_A     (alsuif.DUT.red_op_A),
      .red_op_B     (alsuif.DUT.red_op_B),
      .bypass_A     (alsuif.DUT.bypass_A),
      .bypass_B     (alsuif.DUT.bypass_B),
      .direction    (alsuif.DUT.direction),
      .serial_in    (alsuif.DUT.serial_in),
      .opcode       (alsuif.DUT.opcode),
      .A            (alsuif.DUT.A),
      .B            (alsuif.DUT.B),
      .leds         (alsuif.DUT.leds),
      .out          (alsuif.DUT.out)
   );

   initial begin
      uvm_config_db#(virtual alsu_if)::set(null, "uvm_test_top", "ALSU_VIF", alsuif);
      run_test("alsu_test");
   end
endmodule
```

3.2 alsu_test

```
package alsu_test_pkg;
import alsu_env_pkg::*;
import alsu_config_obj_pkg::*;
import uvm_pkg::*;
‘include "uvm_macros.svh"

    class alsu_test extends uvm_test;
       ‘uvm_component_utils(alsu_test)

        alsu_env env;
        alsu_config_obj alsu_config_obj_test;

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

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

        alsu_config_obj_test = alsu_config_obj::type_id::create("alsu_config_obj_test");

                if (!uvm_config_db#(virtual alsu_if)::get(this, "", "ALSU_VIF", alsu_config_obj_test.alsu_config_vif))
               ‘uvm_fatal("NOVIF", "Virtual␣interface␣alsu_test_vif␣was␣not␣found␣in␣the␣configuration␣database");

        uvm_config_db#(alsu_config_obj)::set(this,"*","CFG",alsu_config_obj_test);

        env = alsu_env::type_id::create("env",this);
        endfunction

        task run_phase(uvm_phase phase);
            super.run_phase(phase);
            phase.raise_objection(this);
            #100; ‘uvm_info("run_phase","Inside␣the␣ALSU␣test.",UVM_MEDIUM);
        phase.drop_objection(this);
        endtask : run_phase

    endclass : alsu_test

endpackage
```

3.3 alsu_if

```
interface alsu_if (
   input bit clk
);

  logic rst;
  logic cin;
  logic red_op_A;
  logic red_op_B;
  logic bypass_A;
  logic bypass_B;
  logic direction;
  logic serial_in;
  logic signed [2:0] A;
  logic signed [2:0] B;
  logic [2:0]        opcode;
  wire [15:0]        leds;
  wire signed [5:0]  out;

modport DUT (
    input clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
    output leds, out
);

modport TEST (
    output clk, cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in, opcode, A, B,
    input leds, out
);

endinterface
```

3.4 alsu_env

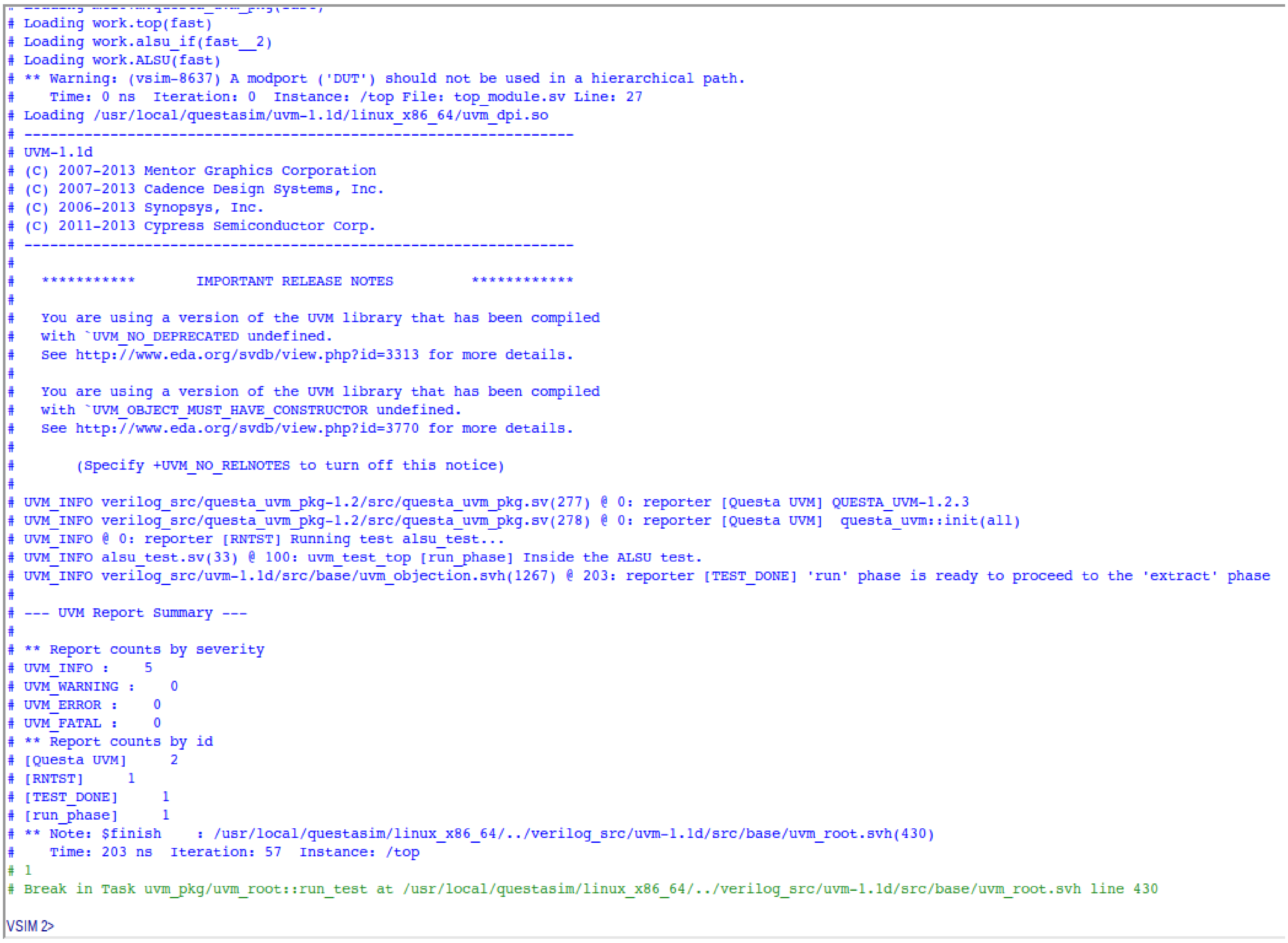
```
package alsu_env_pkg;
import alsu_driver_pkg::*;
import uvm_pkg::*;
‘include "uvm_macros.svh"

    class alsu_env extends uvm_env;
        ‘uvm_component_utils(alsu_env)

        alsu_driver driver;

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

        function void build_phase(uvm_phase phase);
        super.build_phase(phase);
        driver = alsu_driver::type_id::create("driver",this);
        endfunction

    endclass : alsu_env

endpackage
```

3.5 alsu_driver

```
package alsu_driver_pkg;

import uvm_pkg::*;
import alsu_config_obj_pkg::*;
`include "uvm_macros.svh"

      class alsu_driver extends uvm_driver;
      `uvm_component_utils(alsu_driver)

      virtual alsu_if alsu_driver_vif;
      alsu_config_obj alsu_config_obj_driver;

      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_obj)::get(this, "", "CFG", alsu_config_obj_driver)) begin
          `uvm_fatal("NOVIF", "Virtual␣interface␣alsu_driver_vif␣was␣not␣found␣in␣the␣configuration␣database")
        end
      endfunction

      function void connect_phase(uvm_phase phase);
        super.connect_phase(phase);
        alsu_driver_vif = alsu_config_obj_driver.alsu_config_vif;
      endfunction


      task run_phase(uvm_phase phase);
        phase.raise_objection(this);


        alsu_driver_vif.rst = 1;      // Assert reset
        repeat (2) @(posedge alsu_driver_vif.clk); // Wait for a couple of clock cycles
        alsu_driver_vif.rst = 0;      // Deassert reset


        repeat (100)begin
          alsu_driver_vif.A         = $random;
          alsu_driver_vif.B         = $random;
          alsu_driver_vif.opcode    = $random;
          alsu_driver_vif.cin       = $random;
          alsu_driver_vif.red_op_A  = $random;
          alsu_driver_vif.red_op_B  = $random;
          alsu_driver_vif.bypass_A  = $random;
          alsu_driver_vif.bypass_B  = $random;
          alsu_driver_vif.direction = $random;
          alsu_driver_vif.serial_in = $random;

          // Wait for the next positive clock edge
          @(posedge alsu_driver_vif.clk);
        end

        phase.drop_objection(this);
          endtask

    endclass
endpackage
```

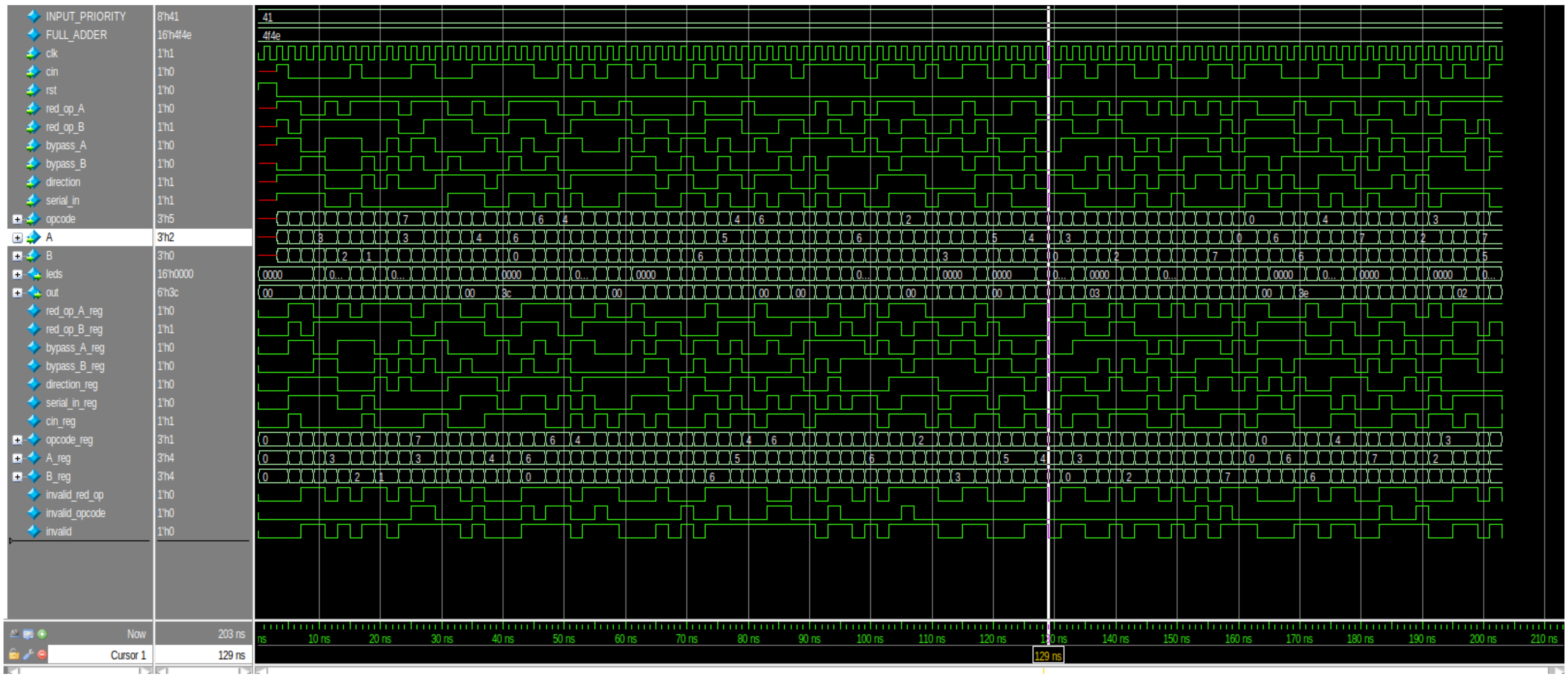
3.6 alsu_config_obj

```
package alsu_config_obj_pkg;

  import uvm_pkg::*;
  `include "uvm_macros.svh"

  class alsu_config_obj extends uvm_object;
    `uvm_object_utils(alsu_config_obj)

    virtual alsu_if alsu_config_vif;

    function new (string name = "alsu_config_obj");
      super.new(name);
    endfunction

  endclass

endpackage
```



```
# Loading work.top(fast)
# Loading work.alsu_if(fast__2)
# Loading work.ALSU(fast)
# ** Warning: (vsim-8637) A modport ('DUT') should not be used in a hierarchical path.
#    Time: 0 ns  Iteration: 0  Instance: /top File: top_module.sv Line: 27
# Loading /usr/local/questasim/uvm-1.1d/linux_x86_64/uvm_dpi.so
# ----------------------------------------------------------------
# UVM-1.1d
# (C) 2007-2013 Mentor Graphics Corporation
# (C) 2007-2013 Cadence Design Systems, Inc.
# (C) 2006-2013 Synopsys, Inc.
# (C) 2011-2013 Cypress Semiconductor Corp.
# ----------------------------------------------------------------
#
#   ***********       IMPORTANT RELEASE NOTES         ************
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_NO_DEPRECATED undefined.
#   See http://www.eda.org/svdb/view.php?id=3313 for more details.
#
#   You are using a version of the UVM library that has been compiled
#   with `UVM_OBJECT_MUST_HAVE_CONSTRUCTOR undefined.
#   See http://www.eda.org/svdb/view.php?id=3770 for more details.
#
#       (Specify +UVM_NO_RELNOTES to turn off this notice)
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test alsu_test...
# UVM_INFO alsu_test.sv(33) @ 100: uvm_test_top [run_phase] Inside the ALSU test.
# UVM_INFO verilog_src/uvm-1.1d/src/base/uvm_objection.svh(1267) @ 203: reporter [TEST_DONE] 'run' phase is ready to proceed to the 'extract' phase
#
# --- UVM Report Summary ---
#
# ** Report counts by severity
# UVM_INFO :    5
# UVM_WARNING :    0
# UVM_ERROR :    0
# UVM_FATAL :    0
# ** Report counts by id
# [Questa UVM]     2
# [RNTST]     1
# [TEST_DONE]     1
# [run_phase]     1
# ** Note: $finish    : /usr/local/questasim/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 203 ns  Iteration: 57  Instance: /top
# 1
# Break in Task uvm_pkg/uvm_root::run_test at /usr/local/questasim/linux_x86_64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430

VSIM 2>
```

Figure 4: Transcript



INPUT_PRIORITY	8'h41
FULL_ADDER	16'h4f4e
clk	1'h1
cin	1'h0
rst	1'h0
red_op_A	1'h0
red_op_B	1'h1
bypass_A	1'h0
bypass_B	1'h0
direction	1'h1
serial_in	1'h1
opcode	3'h5
A	3'h2
B	3'h0
leds	16'h0000
out	6'h3c
red_op_A_reg	1'h0
red_op_B_reg	1'h1
bypass_A_reg	1'h0
bypass_B_reg	1'h0
direction_reg	1'h0
serial_in_reg	1'h0
cin_reg	1'h1
opcode_reg	3'h1
A_reg	3'h4
B_reg	3'h4
invalid_red_op	1'h0
invalid_opcode	1'h0
invalid	1'h0
Now	203 ns
Cursor 1	129 ns

Figure 5: simulation waveform

note : in top module i put interface as hierarchical for simplicity to not change design code only