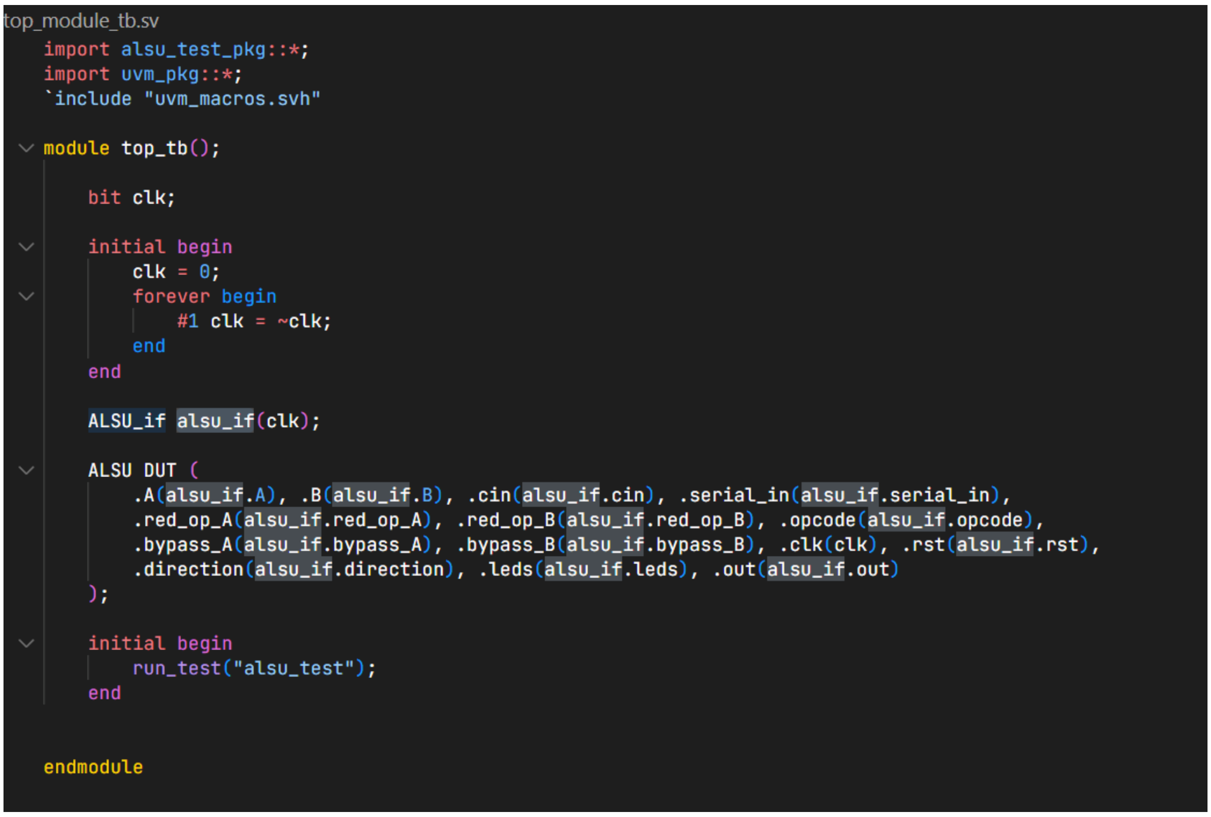
Digital Verification using SV and UVM
Assignment-5

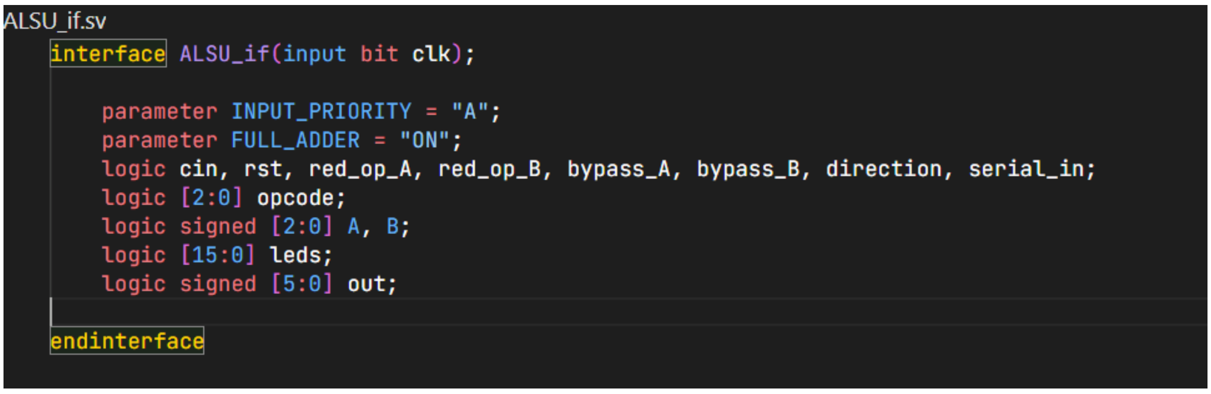
Name: Fares Khalaf Sultan

➢ **Part#1:**

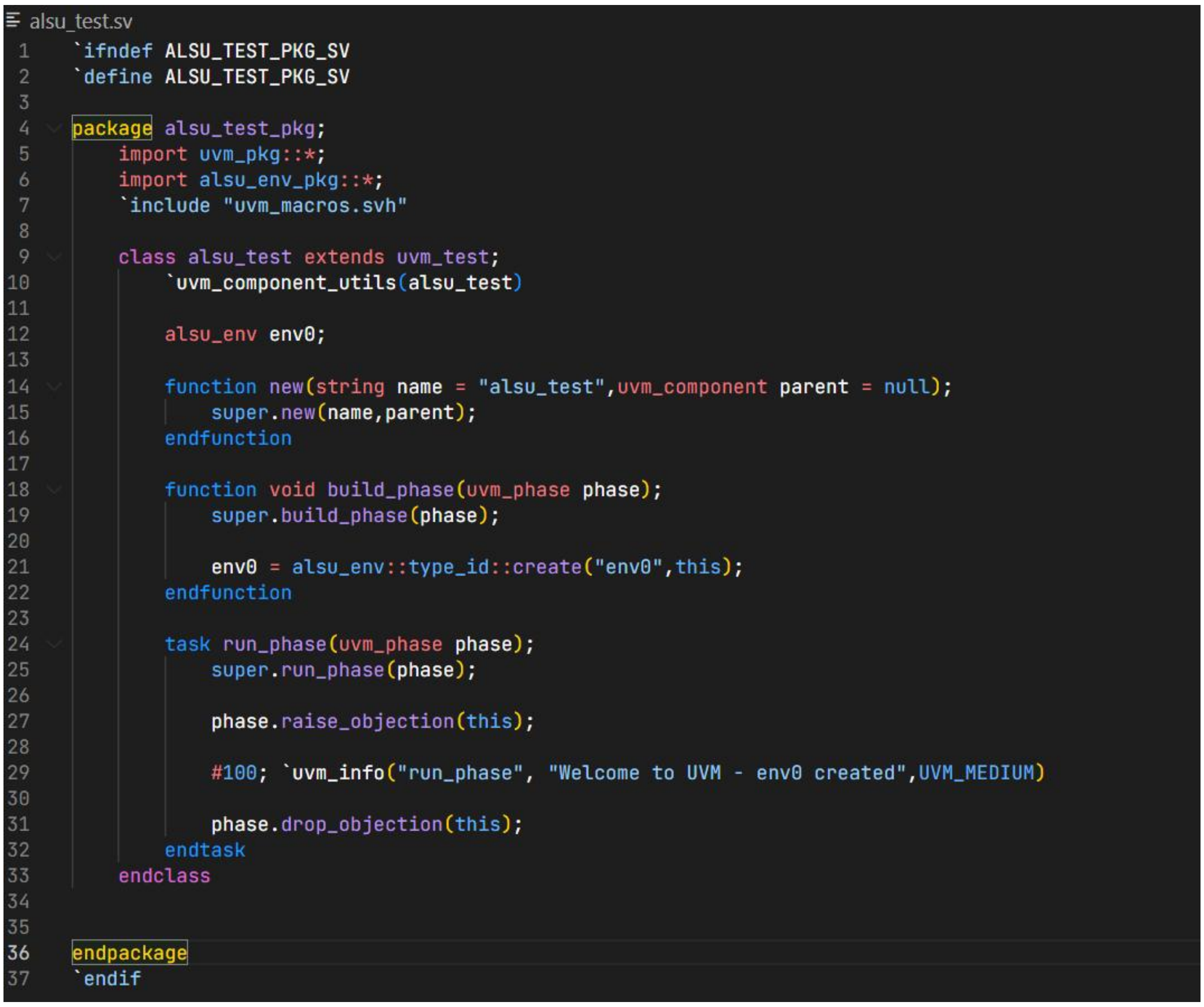
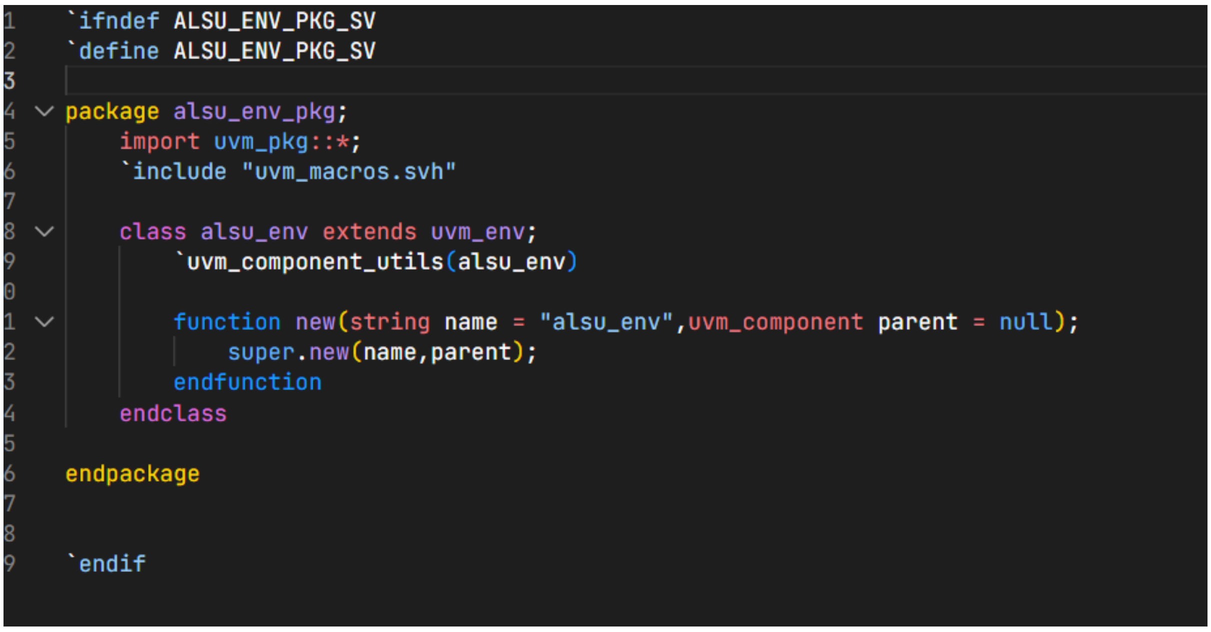
❖ **UVM Environment:**


top_module_tb.sv

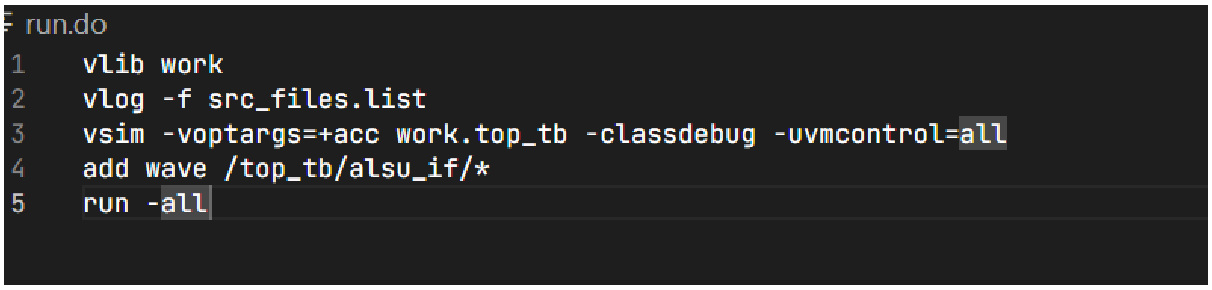
```
import alsu_test_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

module top_tb();

    bit clk;

    initial begin
        clk = 0;
        forever begin
            #1 clk = ~clk;
        end
    end

    ALSU_if alsu_if(clk);

    ALSU DUT (
        .A(alsu_if.A), .B(alsu_if.B), .cin(alsu_if.cin), .serial_in(alsu_if.serial_in),
        .red_op_A(alsu_if.red_op_A), .red_op_B(alsu_if.red_op_B), .opcode(alsu_if.opcode),
        .bypass_A(alsu_if.bypass_A), .bypass_B(alsu_if.bypass_B), .clk(clk), .rst(alsu_if.rst),
        .direction(alsu_if.direction), .leds(alsu_if.leds), .out(alsu_if.out)
    );

    initial begin
        run_test("alsu_test");
    end


endmodule
```


ALSU_if.sv

```
interface ALSU_if(input bit clk);

    parameter INPUT_PRIORITY = "A";
    parameter FULL_ADDER = "ON";
    logic cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
    logic [2:0] opcode;
    logic signed [2:0] A, B;
    logic [15:0] leds;
    logic signed [5:0] out;

endinterface
```

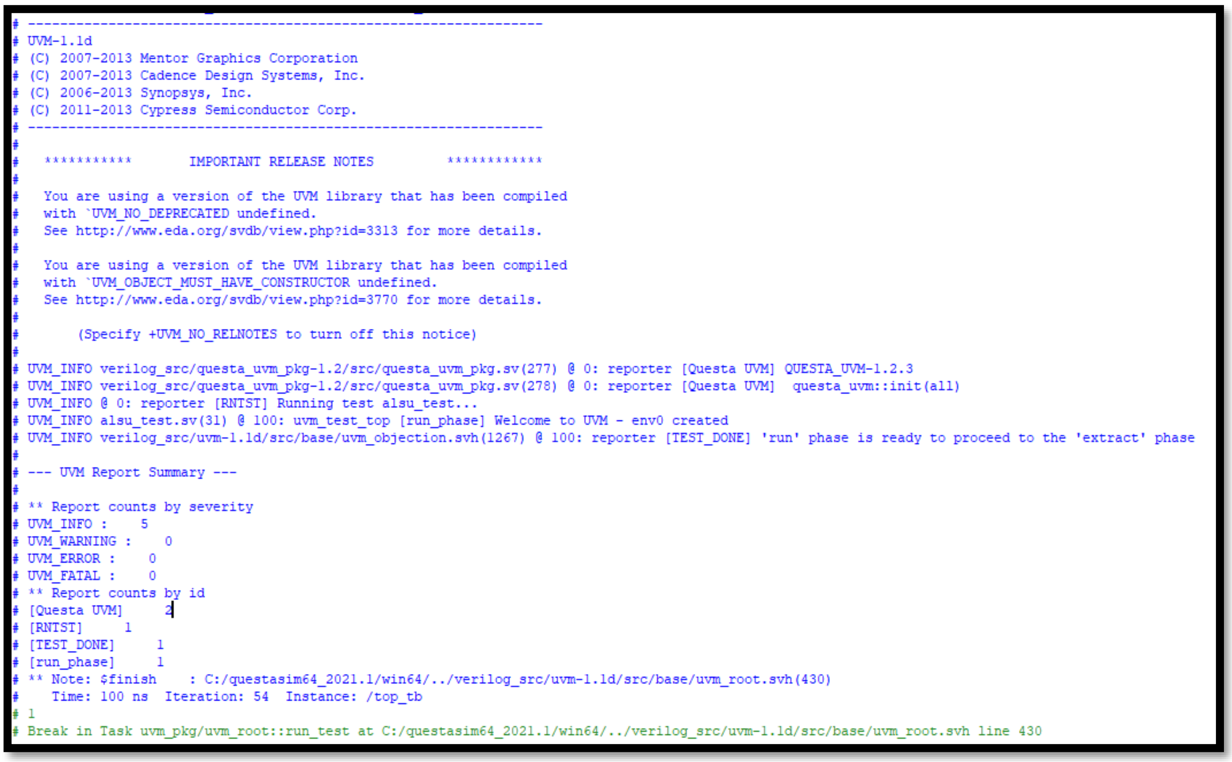

alsu_test.sv
```
`ifndef ALSU_TEST_PKG_SV
`define ALSU_TEST_PKG_SV

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

    class alsu_test extends uvm_test;
        `uvm_component_utils(alsu_test)

        alsu_env env0;

        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);

            env0 = alsu_env::type_id::create("env0",this);
        endfunction

        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            phase.raise_objection(this);

            #100; `uvm_info("run_phase", "Welcome to UVM - env0 created",UVM_MEDIUM)

            phase.drop_objection(this);
        endtask
    endclass


endpackage
`endif
```

```
`ifndef ALSU_ENV_PKG_SV
`define ALSU_ENV_PKG_SV

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

    class alsu_env extends uvm_env;
        `uvm_component_utils(alsu_env)

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

endpackage


`endif
```

❖ Do file:


run.do

```
vlib work
vlog -f src_files.list
vsim -voptargs=+acc work.top_tb -classdebug -uvmcontrol=all
add wave /top_tb/alsu_if/*
run -all
```

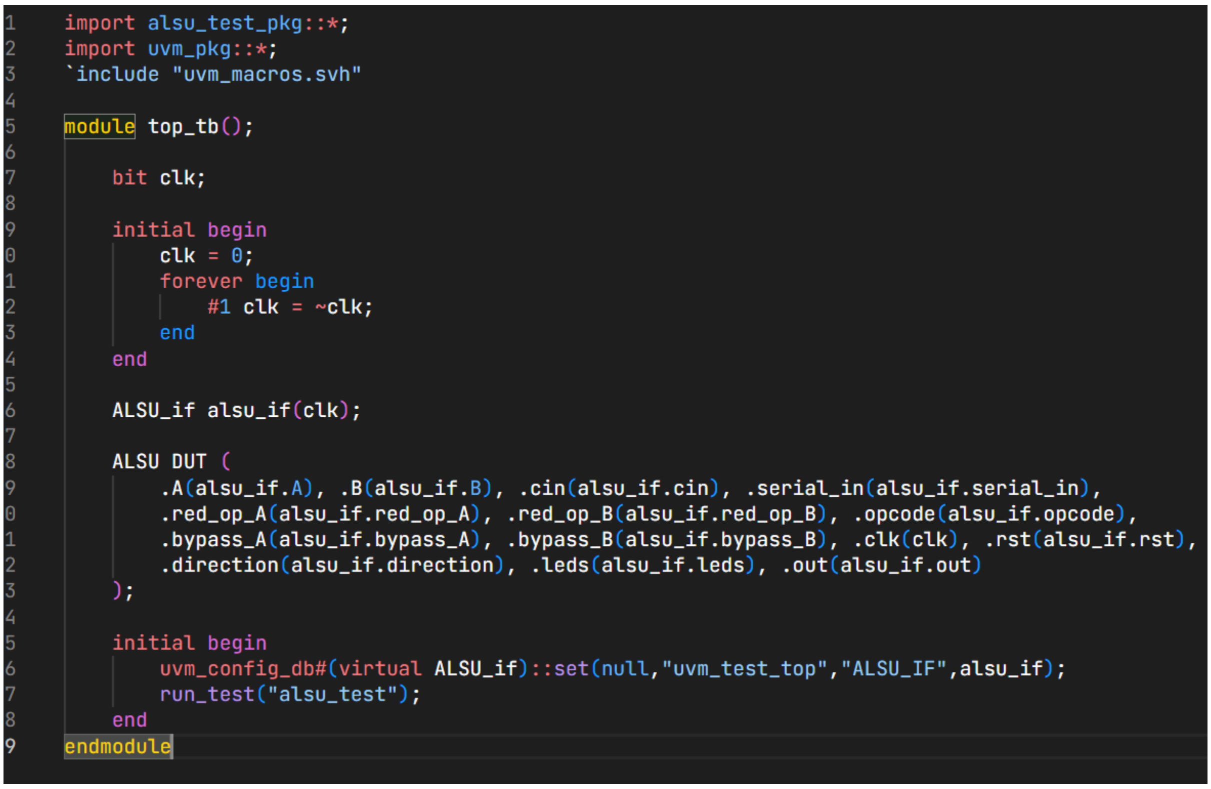
❖ Results:



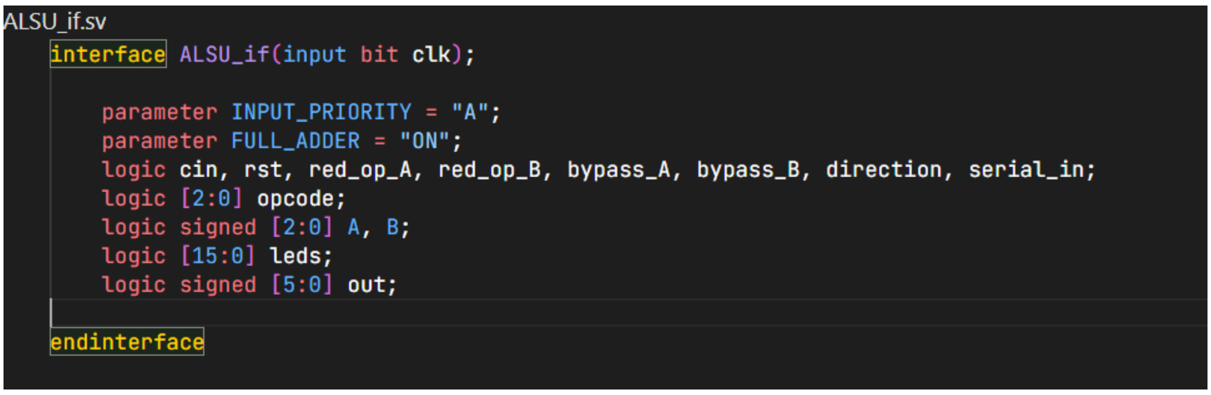
```
# ----------------------------------------------------------------
# UVM-1.1d
# (C) 2007-2013 Mentor Graphics Corporation
# (C) 2007-2013 Cadence Design Systems, Inc.
# (C) 2006-2013 Synopsys, Inc.
# (C) 2011-2013 Cypress Semiconductor Corp.
# ----------------------------------------------------------------
#
#   ***********       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(31) @ 100: uvm_test_top [run_phase] Welcome to UVM - env0 created
# 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    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 100 ns  Iteration: 54  Instance: /top_tb
# 1
# Break in Task uvm_pkg/uvm_root::run_test at C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430
```

➢ Part#2: Driving the interface:

❖ UVM Environment:

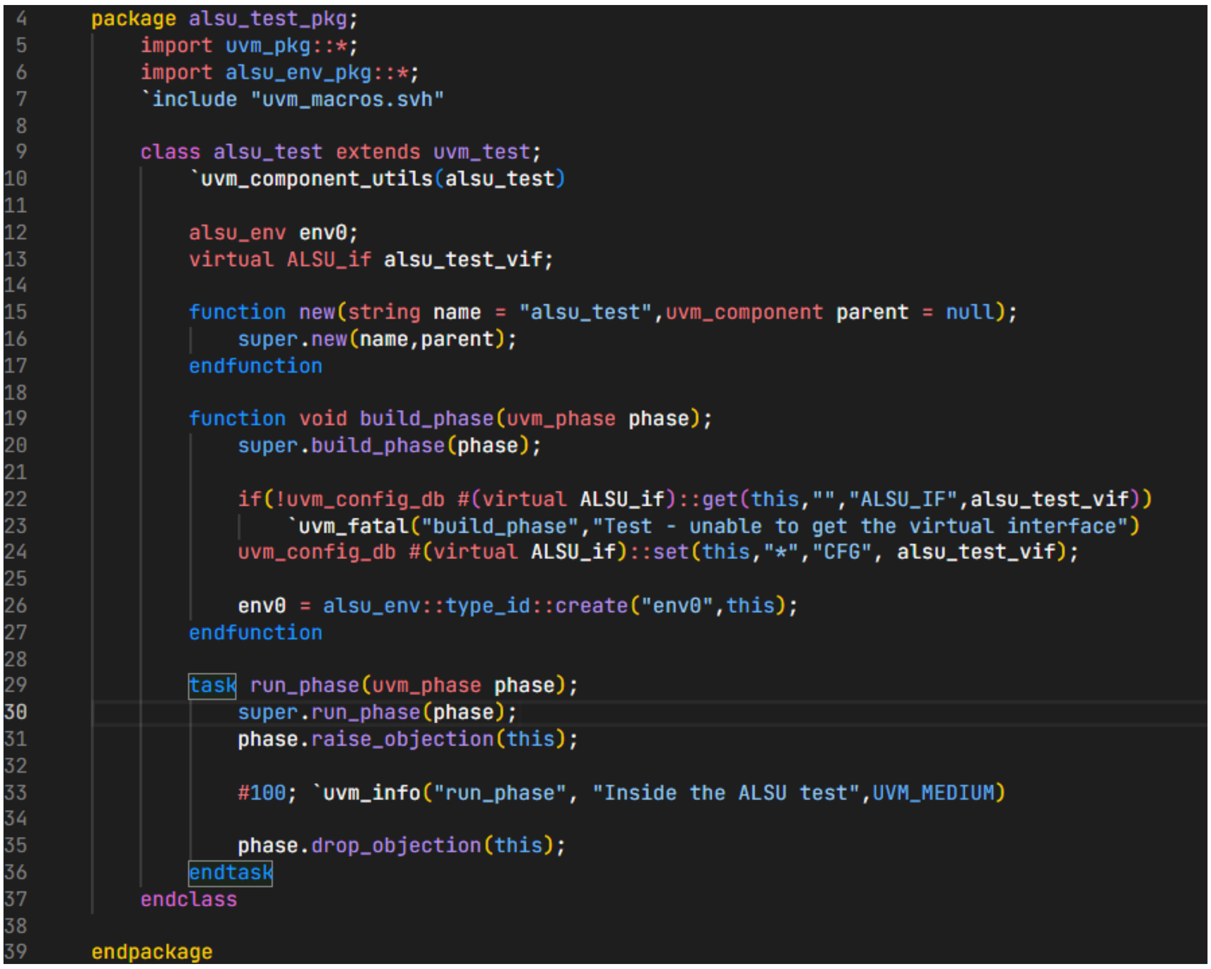


```
import alsu_test_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

module top_tb();

    bit clk;

    initial begin
        clk = 0;
        forever begin
            #1 clk = ~clk;
        end
    end

    ALSU_if alsu_if(clk);

    ALSU DUT (
        .A(alsu_if.A), .B(alsu_if.B), .cin(alsu_if.cin), .serial_in(alsu_if.serial_in),
        .red_op_A(alsu_if.red_op_A), .red_op_B(alsu_if.red_op_B), .opcode(alsu_if.opcode),
        .bypass_A(alsu_if.bypass_A), .bypass_B(alsu_if.bypass_B), .clk(clk), .rst(alsu_if.rst),
        .direction(alsu_if.direction), .leds(alsu_if.leds), .out(alsu_if.out)
    );

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

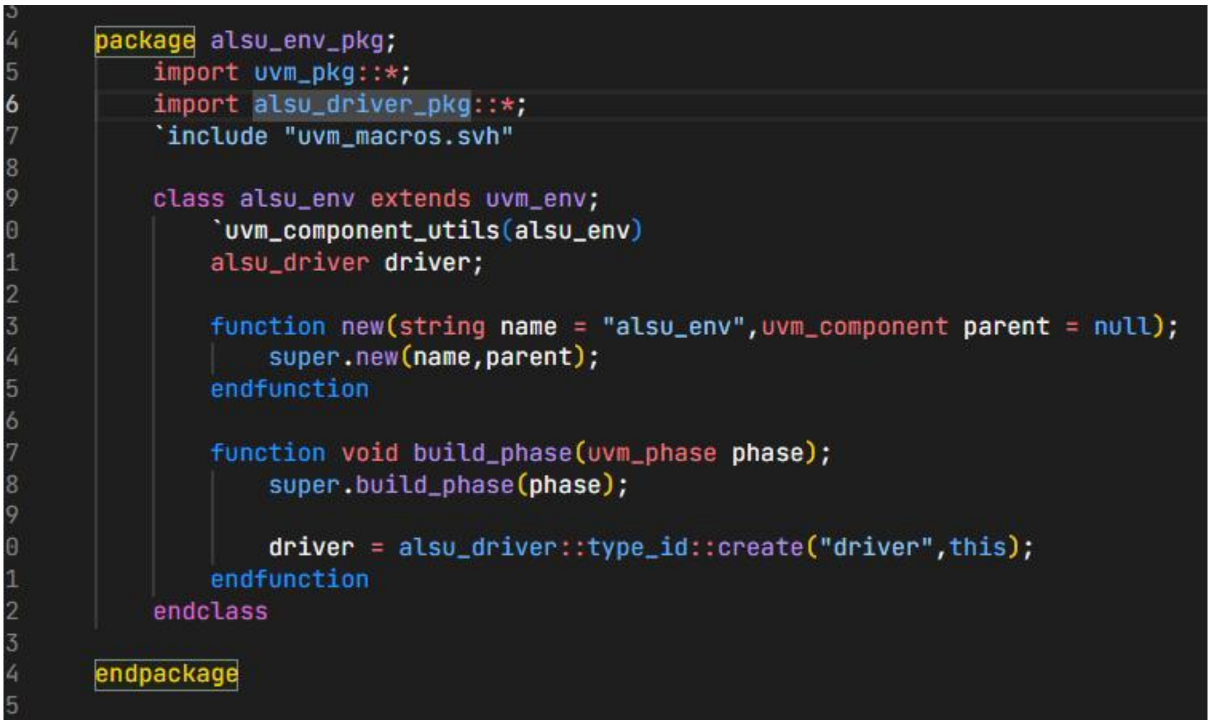


ALSU_if.sv

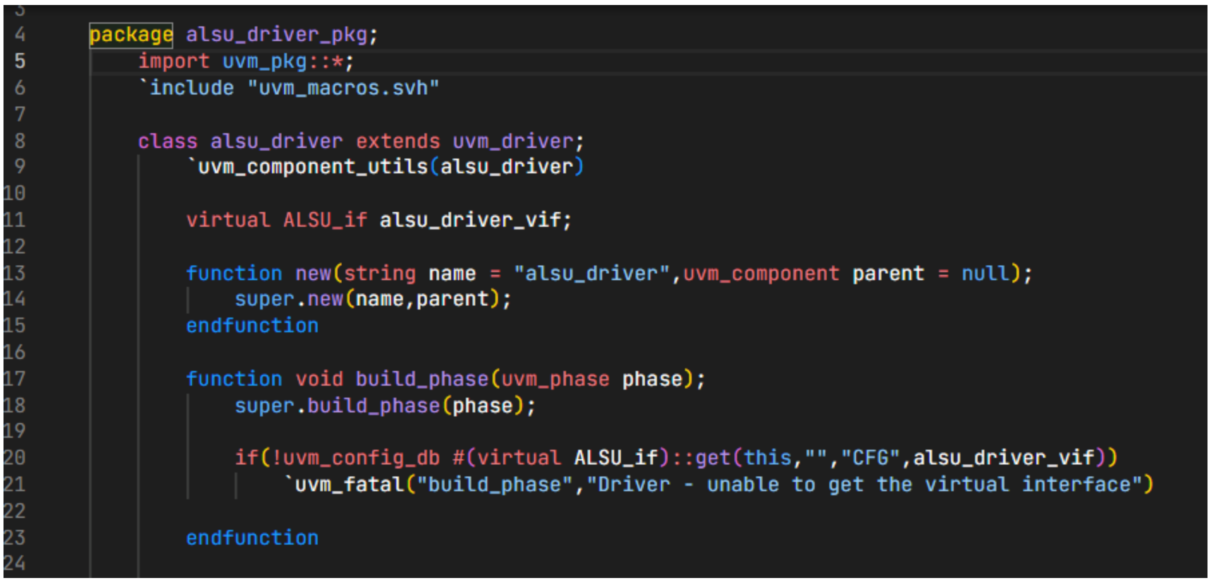
```
interface ALSU_if(input bit clk);

    parameter INPUT_PRIORITY = "A";
    parameter FULL_ADDER = "ON";
    logic cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
    logic [2:0] opcode;
    logic signed [2:0] A, B;
    logic [15:0] leds;
    logic signed [5:0] out;

endinterface
```



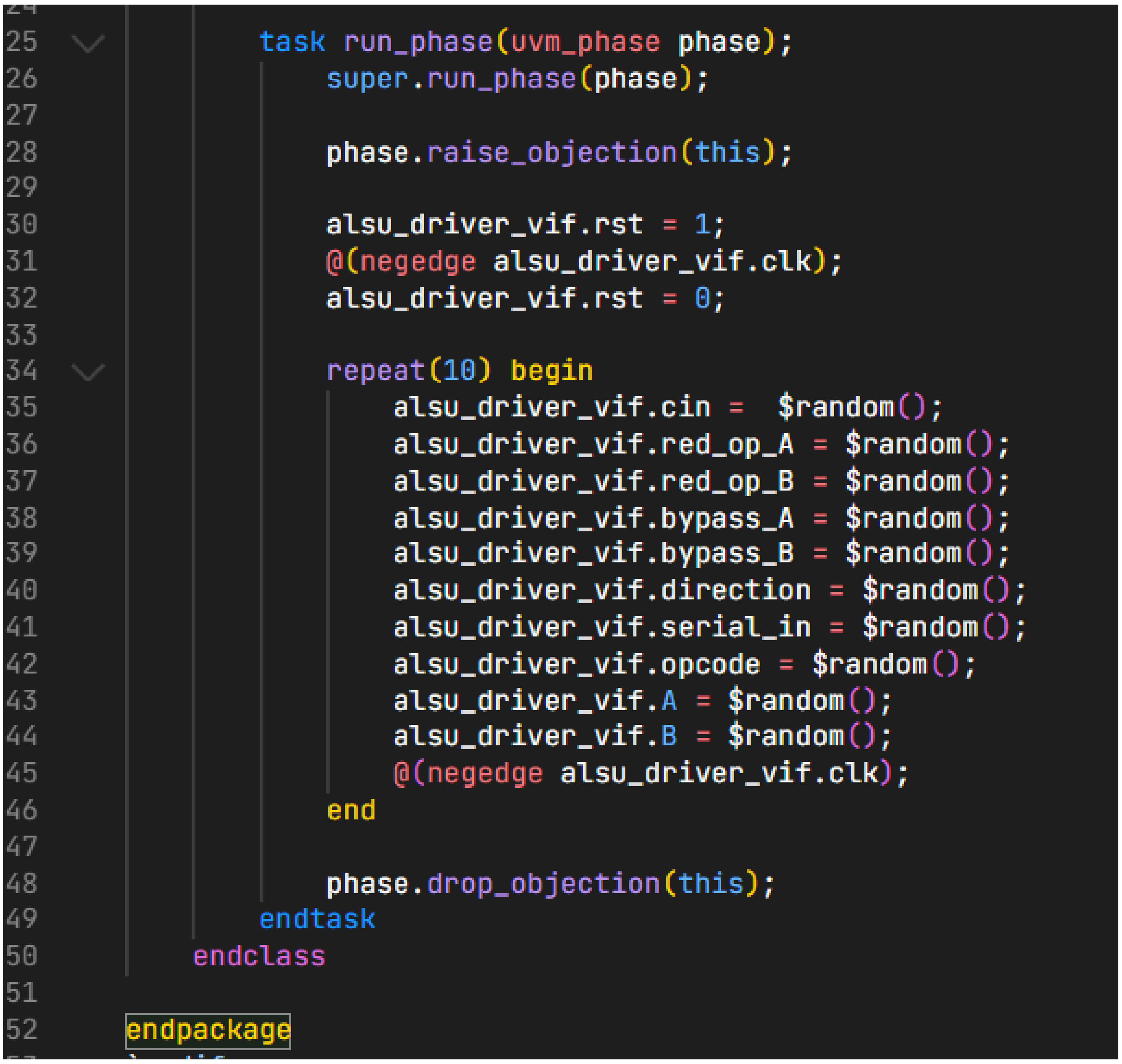
```systemverilog
package alsu_test_pkg;
    import uvm_pkg::*;
    import alsu_env_pkg::*;
    `include "uvm_macros.svh"

    class alsu_test extends uvm_test;
        `uvm_component_utils(alsu_test)

        alsu_env env0;
        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_IF",alsu_test_vif))
                `uvm_fatal("build_phase","Test - unable to get the virtual interface")
            uvm_config_db #(virtual ALSU_if)::set(this,"*","CFG", alsu_test_vif);

            env0 = alsu_env::type_id::create("env0",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
    endclass

endpackage
```



```systemverilog
package alsu_env_pkg;
    import uvm_pkg::*;
    import alsu_driver_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

endpackage
```

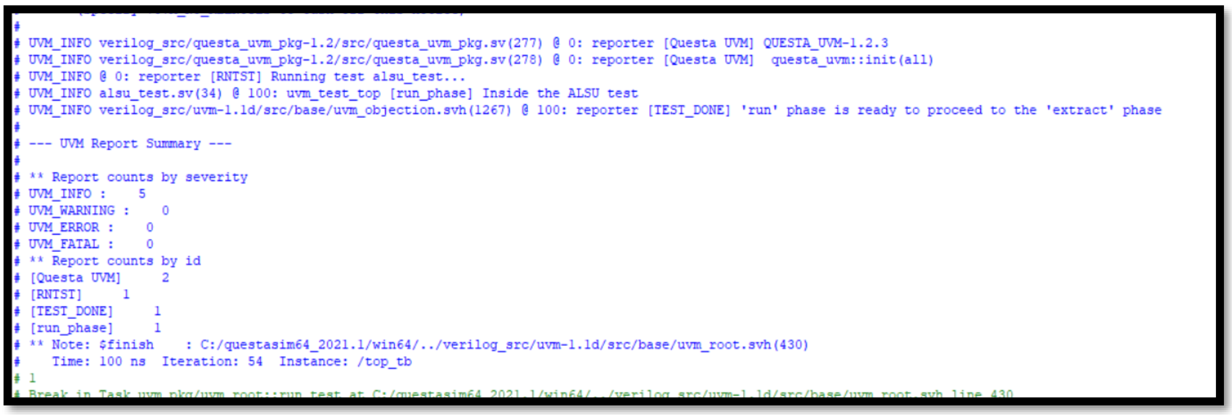


```systemverilog
package alsu_driver_pkg;
    import uvm_pkg::*;
    `include "uvm_macros.svh"

    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,"","CFG",alsu_driver_vif))
                `uvm_fatal("build_phase","Driver - unable to get the virtual interface")

        endfunction

```

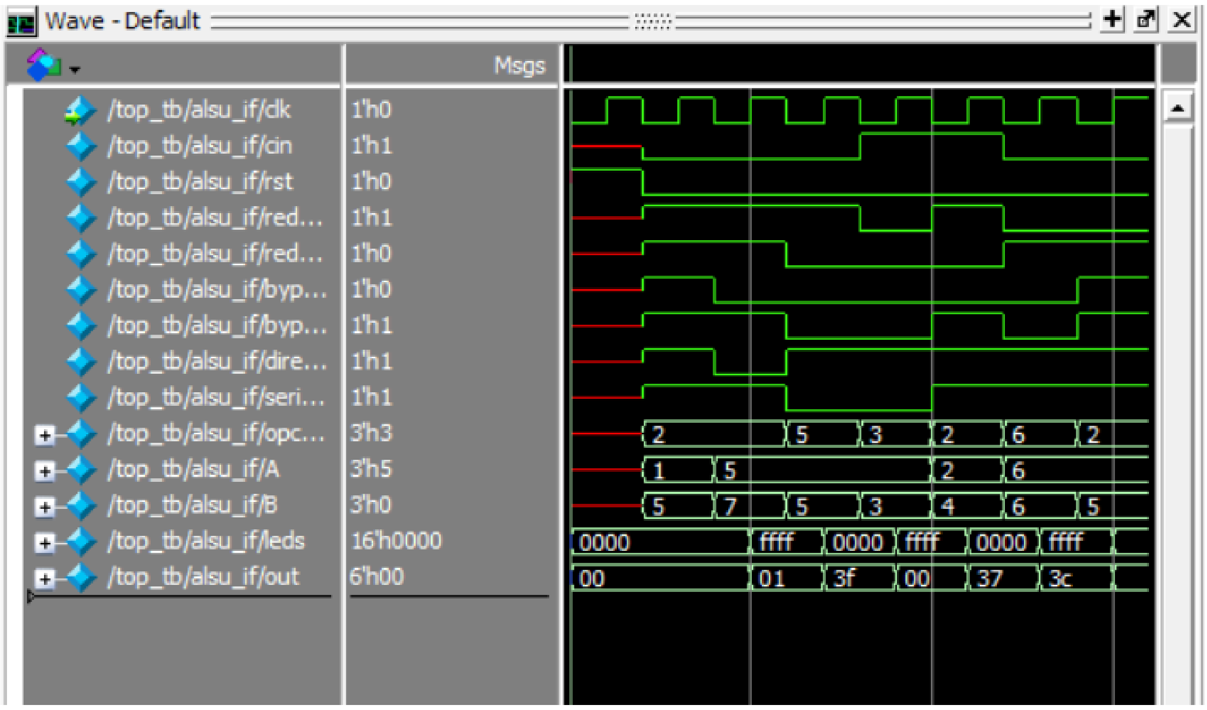


```systemverilog
        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            phase.raise_objection(this);

            alsu_driver_vif.rst = 1;
            @(negedge alsu_driver_vif.clk);
            alsu_driver_vif.rst = 0;

            repeat(10) begin
                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();
                alsu_driver_vif.opcode = $random();
                alsu_driver_vif.A = $random();
                alsu_driver_vif.B = $random();
                @(negedge alsu_driver_vif.clk);
            end

            phase.drop_objection(this);
        endtask
    endclass

endpackage
```

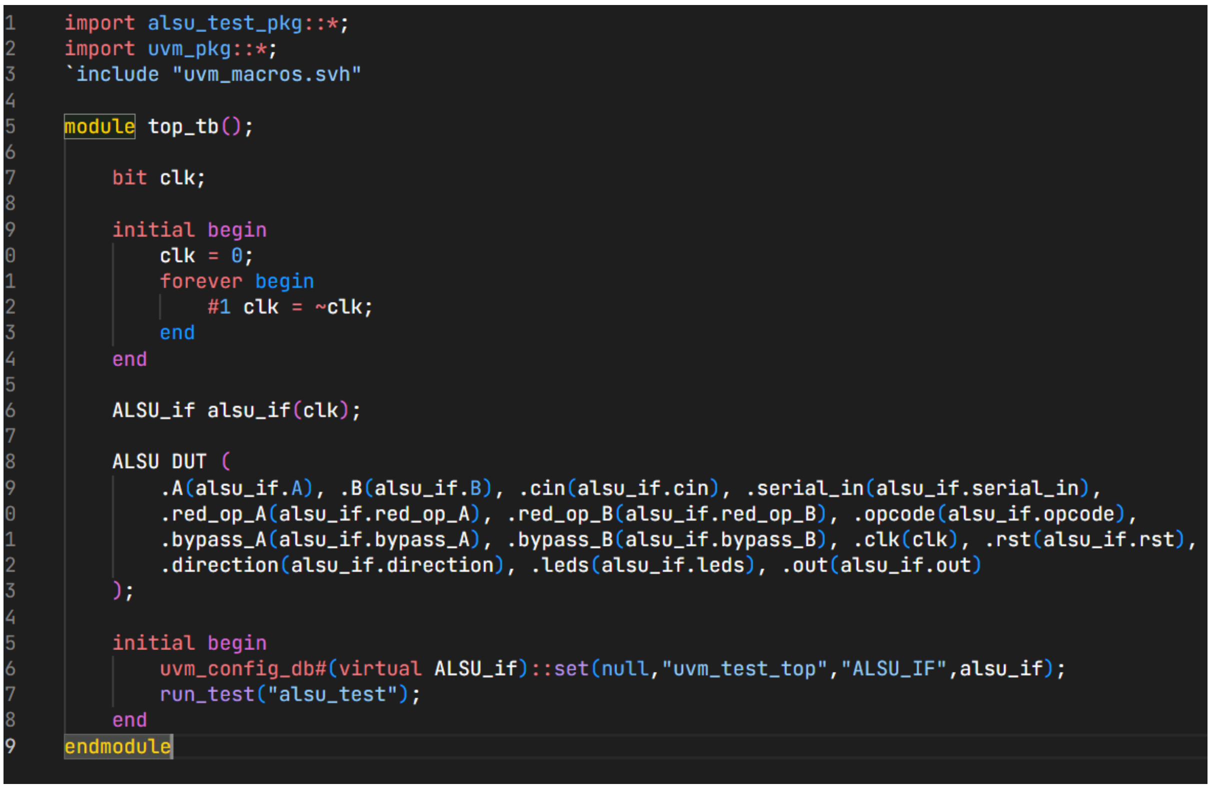
❖ Results:



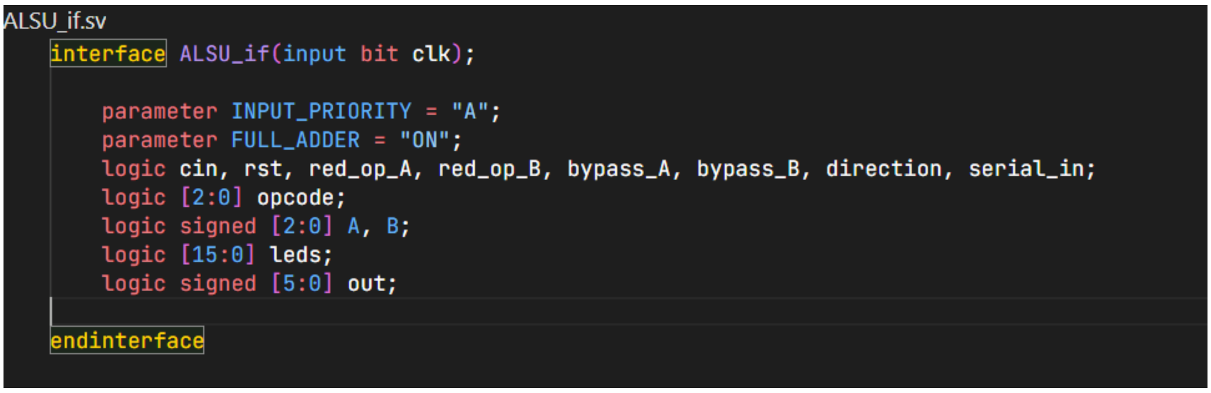
```
#
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(277) @ 0: reporter [Questa UVM] QUESTA_UVM-1.2.3
# UVM_INFO verilog_src/questa_uvm_pkg-1.2/src/questa_uvm_pkg.sv(278) @ 0: reporter [Questa UVM]  questa_uvm::init(all)
# UVM_INFO @ 0: reporter [RNTST] Running test alsu_test...
# UVM_INFO alsu_test.sv(34) @ 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    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 100 ns  Iteration: 54  Instance: /top_tb
# 1
# Break in Task uvm_pkg/uvm_root::run_test at C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430
```


Wave - Default
Msgs
/top_tb/alsu_if/clk 1'h0
/top_tb/alsu_if/cin 1'h1
/top_tb/alsu_if/rst 1'h0
/top_tb/alsu_if/red... 1'h1
/top_tb/alsu_if/red... 1'h0
/top_tb/alsu_if/byp... 1'h0
/top_tb/alsu_if/byp... 1'h1
/top_tb/alsu_if/dire... 1'h1
/top_tb/alsu_if/seri... 1'h1
/top_tb/alsu_if/opc... 3'h3
/top_tb/alsu_if/A 3'h5
/top_tb/alsu_if/B 3'h0
/top_tb/alsu_if/leds 16'h0000
/top_tb/alsu_if/out 6'h00
2 5 3 2 6 2
1 5 2 6
5 7 5 3 4 6 5
0000 ffff 0000 ffff 0000 ffff
00 01 3f 00 37 3c

➢ Part#3: config object:

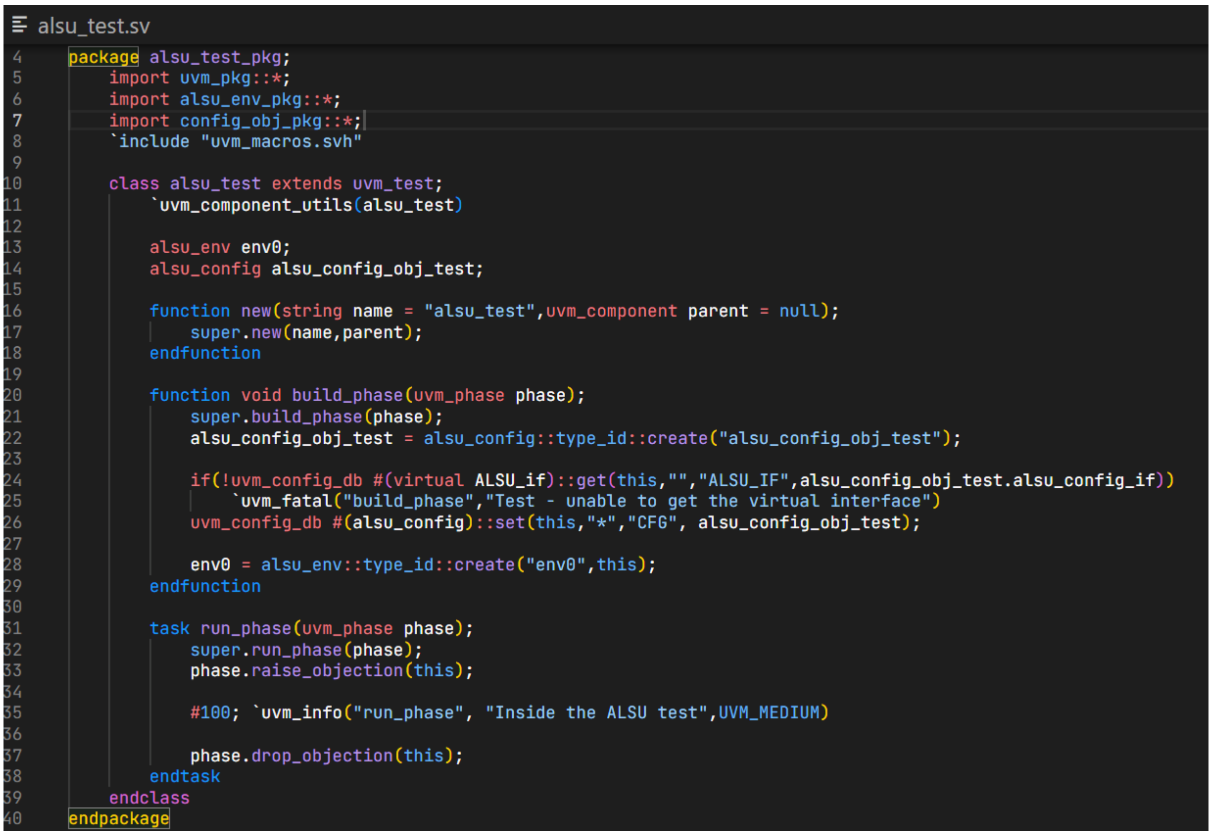


```
import alsu_test_pkg::*;
import uvm_pkg::*;
`include "uvm_macros.svh"

module top_tb();

    bit clk;

    initial begin
        clk = 0;
        forever begin
            #1 clk = ~clk;
        end
    end

    ALSU_if alsu_if(clk);

    ALSU DUT (
        .A(alsu_if.A), .B(alsu_if.B), .cin(alsu_if.cin), .serial_in(alsu_if.serial_in),
        .red_op_A(alsu_if.red_op_A), .red_op_B(alsu_if.red_op_B), .opcode(alsu_if.opcode),
        .bypass_A(alsu_if.bypass_A), .bypass_B(alsu_if.bypass_B), .clk(clk), .rst(alsu_if.rst),
        .direction(alsu_if.direction), .leds(alsu_if.leds), .out(alsu_if.out)
    );

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



```
ALSU_if.sv
interface ALSU_if(input bit clk);

    parameter INPUT_PRIORITY = "A";
    parameter FULL_ADDER = "ON";
    logic cin, rst, red_op_A, red_op_B, bypass_A, bypass_B, direction, serial_in;
    logic [2:0] opcode;
    logic signed [2:0] A, B;
    logic [15:0] leds;
    logic signed [5:0] out;

endinterface
```

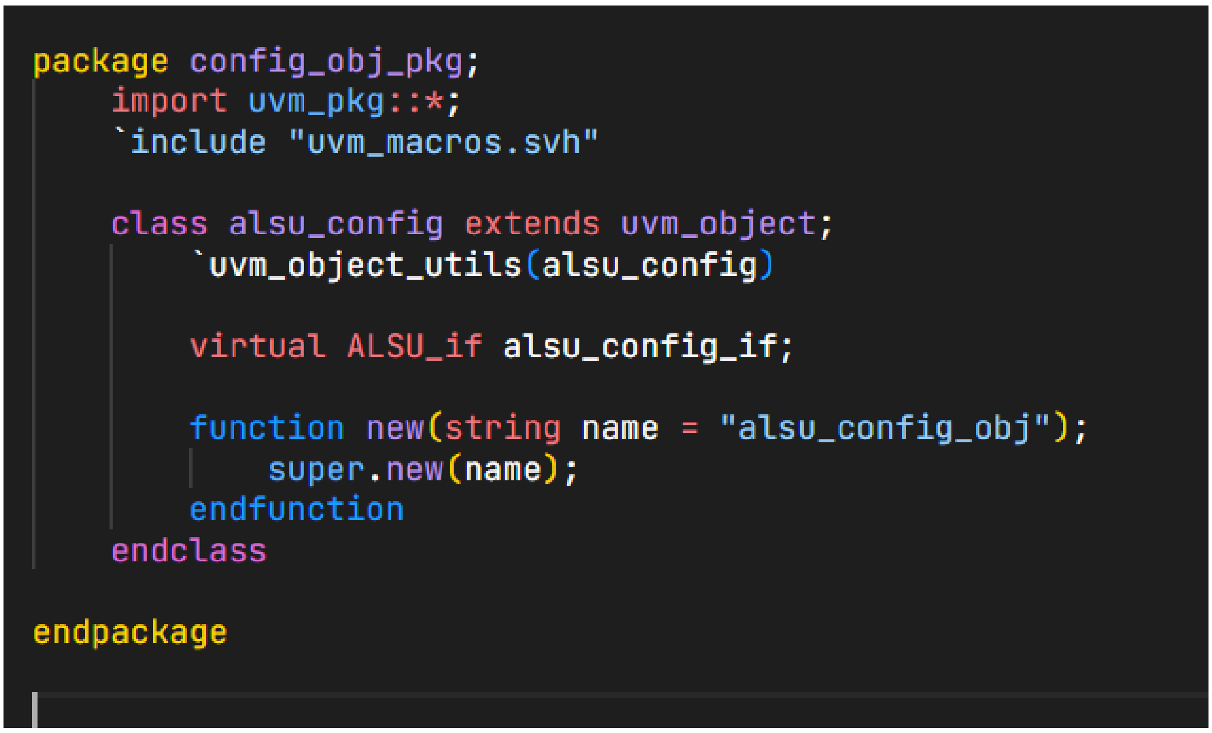
- **Test:**



alsu_test.sv

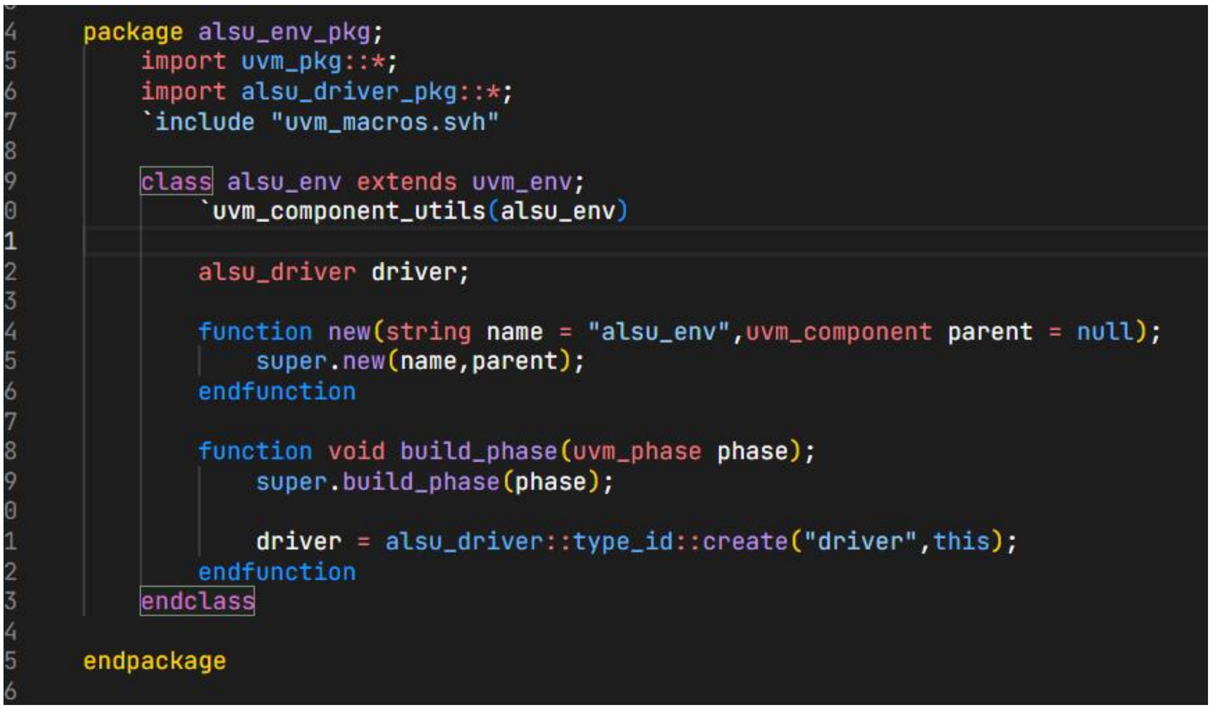
```
package alsu_test_pkg;
    import uvm_pkg::*;
    import alsu_env_pkg::*;
    import config_obj_pkg::*;
    `include "uvm_macros.svh"

    class alsu_test extends uvm_test;
        `uvm_component_utils(alsu_test)

        alsu_env env0;
        alsu_config 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::type_id::create("alsu_config_obj_test");

            if(!uvm_config_db #(virtual ALSU_if)::get(this,"","ALSU_IF",alsu_config_obj_test.alsu_config_if))
                `uvm_fatal("build_phase","Test - unable to get the virtual interface")
            uvm_config_db #(alsu_config)::set(this,"*","CFG", alsu_config_obj_test);

            env0 = alsu_env::type_id::create("env0",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
    endclass
endpackage
```

- **Config object:**



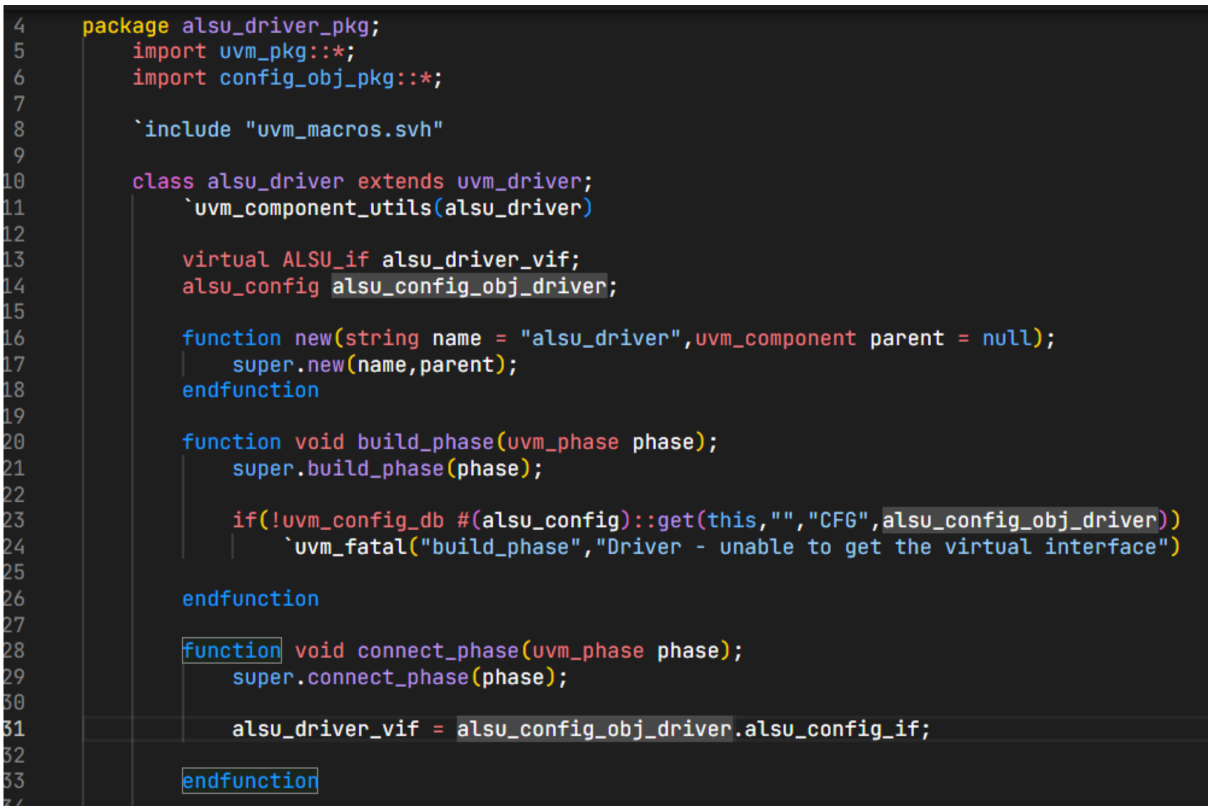
```
package config_obj_pkg;
    import uvm_pkg::*;
    `include "uvm_macros.svh"

    class alsu_config extends uvm_object;
        `uvm_object_utils(alsu_config)

        virtual ALSU_if alsu_config_if;

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

endpackage
```

- **Env:**

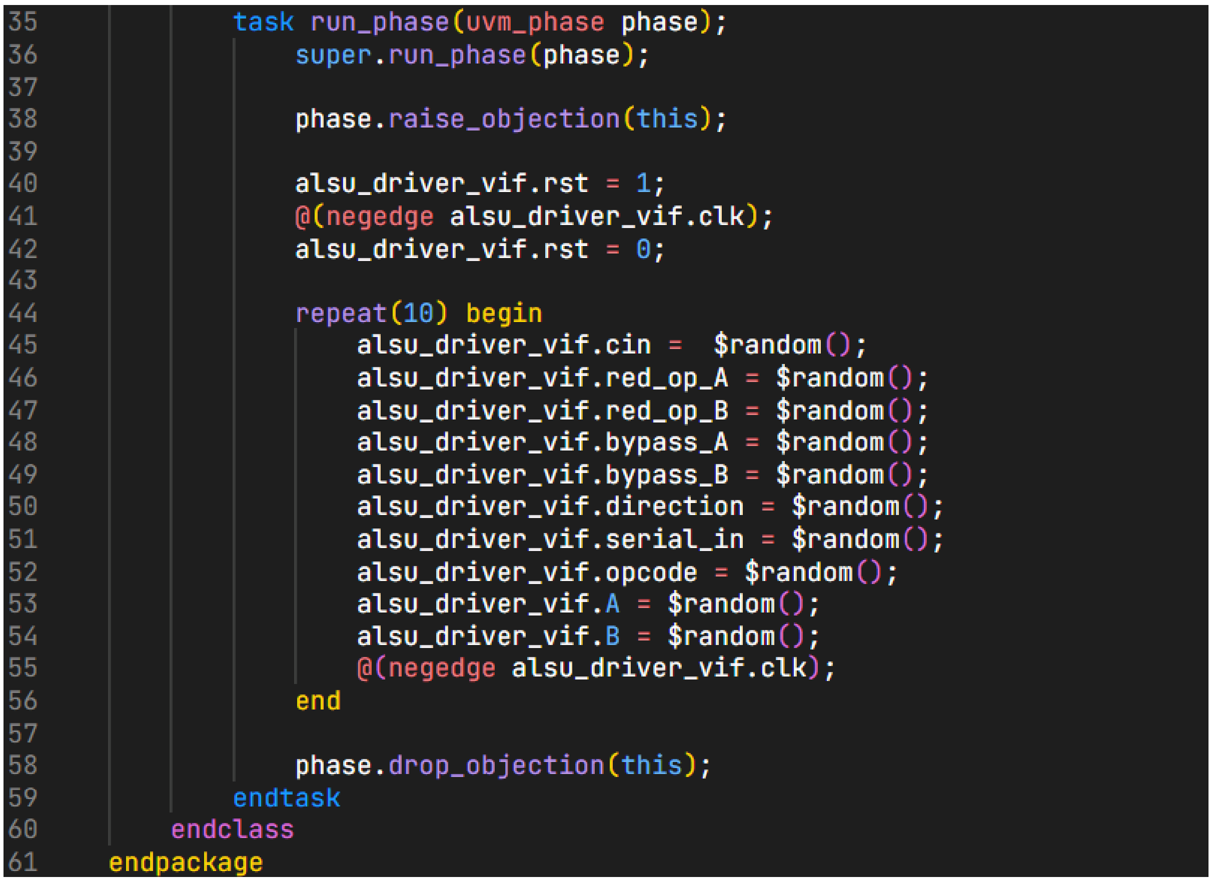


```
package alsu_env_pkg;
    import uvm_pkg::*;
    import alsu_driver_pkg::*;
    `include "uvm_macros.svh"

    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

endpackage
```

- **Driver:**

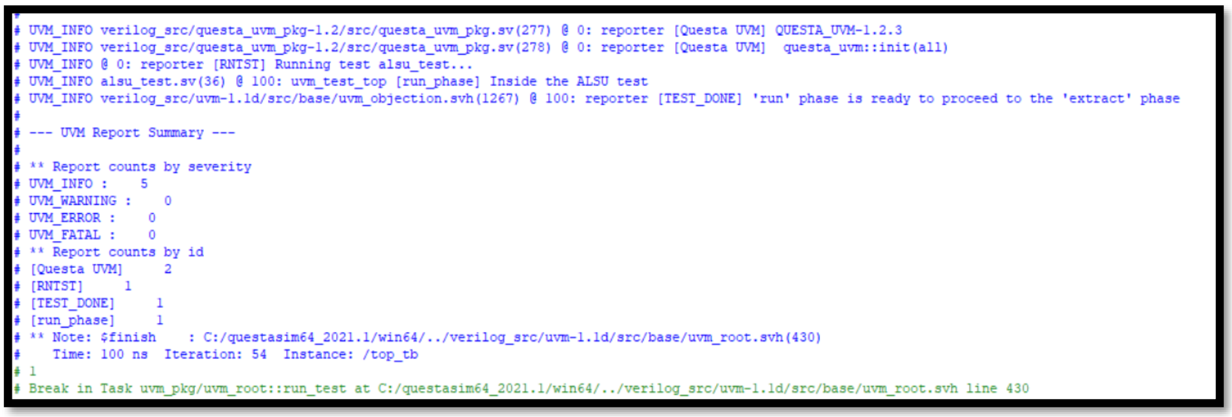


```
package alsu_driver_pkg;
    import uvm_pkg::*;
    import config_obj_pkg::*;

    `include "uvm_macros.svh"

    class alsu_driver extends uvm_driver;
        `uvm_component_utils(alsu_driver)

        virtual ALSU_if alsu_driver_vif;
        alsu_config 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)::get(this,"","CFG",alsu_config_obj_driver))
                `uvm_fatal("build_phase","Driver - unable to get the virtual interface")

        endfunction

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

            alsu_driver_vif = alsu_config_obj_driver.alsu_config_if;

        endfunction
```



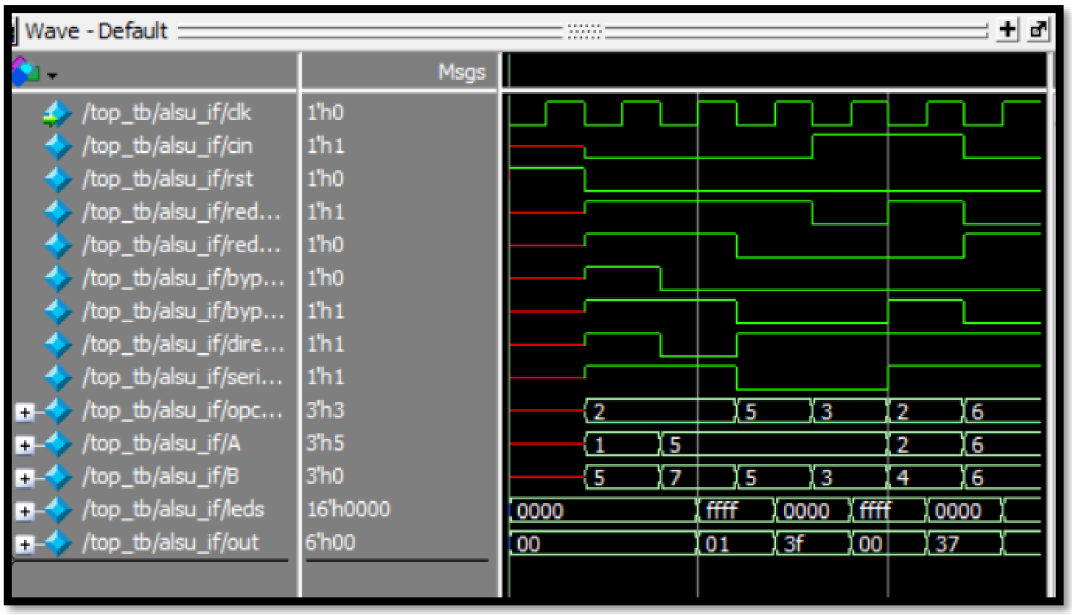
```
        task run_phase(uvm_phase phase);
            super.run_phase(phase);

            phase.raise_objection(this);

            alsu_driver_vif.rst = 1;
            @(negedge alsu_driver_vif.clk);
            alsu_driver_vif.rst = 0;

            repeat(10) begin
                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();
                alsu_driver_vif.opcode = $random();
                alsu_driver_vif.A = $random();
                alsu_driver_vif.B = $random();
                @(negedge alsu_driver_vif.clk);
            end

            phase.drop_objection(this);
        endtask
    endclass
endpackage
```

❖ Result:



```
#
# 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(36) @ 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    : C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh(430)
#    Time: 100 ns  Iteration: 54  Instance: /top_tb
# 1
# Break in Task uvm_pkg/uvm_root::run_test at C:/questasim64_2021.1/win64/../verilog_src/uvm-1.1d/src/base/uvm_root.svh line 430
```

Waveform:


Wave - Default
Msgs
/top_tb/alsu_if/clk 1'h0
/top_tb/alsu_if/cin 1'h1
/top_tb/alsu_if/rst 1'h0
/top_tb/alsu_if/red... 1'h1
/top_tb/alsu_if/red... 1'h0
/top_tb/alsu_if/byp... 1'h0
/top_tb/alsu_if/byp... 1'h1
/top_tb/alsu_if/dire... 1'h1
/top_tb/alsu_if/seri... 1'h1
/top_tb/alsu_if/opc... 3'h3
/top_tb/alsu_if/A 3'h5
/top_tb/alsu_if/B 3'h0
/top_tb/alsu_if/leds 16'h0000
/top_tb/alsu_if/out 6'h00
2 5 3 2 6
1 5 2 6
5 7 5 3 4 6
0000 ffff 0000 ffff 0000
00 01 3f 00 37