Lab Report

Name: Dang Hung Thinh
ID: 22119137
Class: Monday morning class (1-5)

Note: every individual must submit a unique lab report form.

1. **Summarize (in your own words) the subject of this lab:**

   - This lab focuses on learning how to model and simulate simple combinational logic circuits using *SystemC* at the *behavioral level*. Specifically, we are asked to design a *4-to-1 multiplexer* and *a 2-to-4 decoder*, including creating *testbenches* and *simulation files* to verify their correct functionality.

2. **Describe the new concepts covered in this lab:**
   - Introduction to *SystemC behavioral modeling*
   - Designing *combinational logic* (MUX, Decoder) in *SystemC*
   - Writing and using *testbenches* to simulate and verify digital designs
   - Understanding the basic flow of SystemC simulation (e.g., `SC_MODULE`, `SC_METHOD`, sensitivity list)
3. **Describe how this lab built upon previous ones:**
   - The previous lab introduced us to the *SystemC development environment* and basic syntax. This lab builds upon that foundation by applying SystemC to design *functional digital components* and validating them through *simulation and testbenches,* which is a key step in digital system design.
4. **Describe the most difficult part of this lab for you:**
   - The most difficult part was writing the *testbench* to thoroughly verify the functionality of the multiplexer and decoder. It required careful planning to cover all input combinations and to ensure that timing and signal sensitivity were handled correctly in SystemC.

5. **Describe problems you faced and how you solved them:**
   - One problem was that the *multiplexer output didn't change* when expected. After debugging, I realized I forgot to register the process method with the correct *sensitivity list* using sensitive << sel << d0 << d1 << d2 << d3;. After fixing this, the output updated correctly. Also, getting used to SC_MODULE structure took some practice.

6. **Do you verify that the code included with this report is your's original work (yes/no)?**

7. **Submit your source code, testbench, and simulation output.**

   7.1. 4-to-1 multiplexer

   7.1.1. On Ubuntu

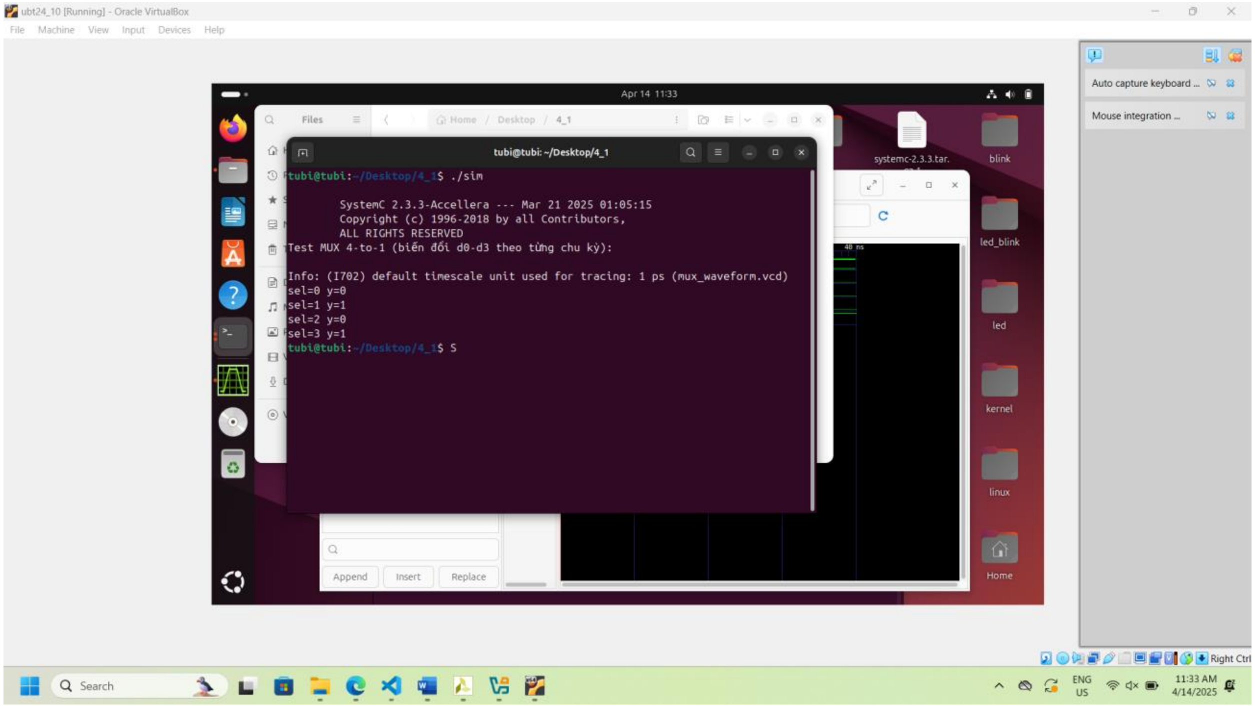
- SOURCE CODE:

```
#include <systemc.h>
SC_MODULE(Mux4to1) {
    sc_in<sc_uint<2>> sel;
    sc_in<bool> d0, d1, d2, d3;
    sc_out<bool> y;

    void do_mux() {
        switch (sel.read()) {
            case 0: y.write(d0.read()); break;
            case 1: y.write(d1.read()); break;
            case 2: y.write(d2.read()); break;
            case 3: y.write(d3.read()); break;
        }
    }

    SC_CTOR(Mux4to1) {
        SC_METHOD(do_mux);
        sensitive << sel << d0 << d1 << d2 << d3;
    }
};
```

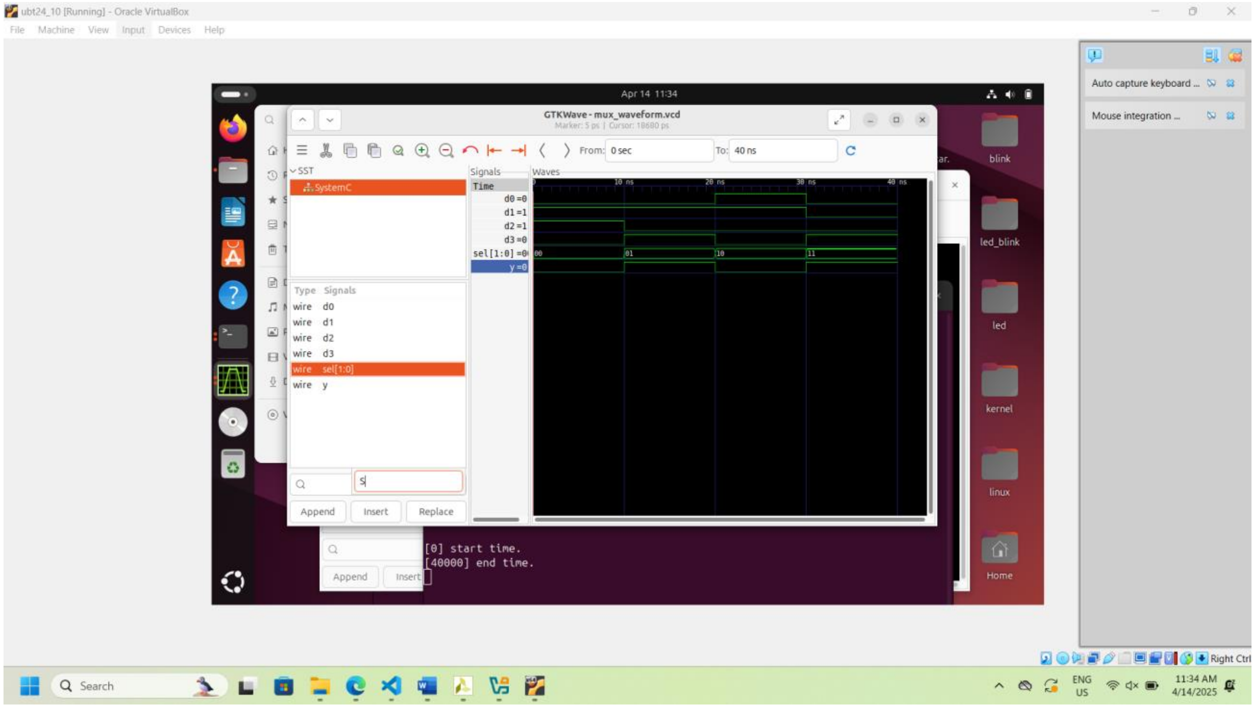
- TEST BENCH:

```
#include <systemc.h>
#include "mux4to1.cpp"

int sc_main(int argc, char* argv[]) {
    sc_signal<sc_uint<2>> sel;
    sc_signal<bool> d0, d1, d2, d3;
    sc_signal<bool> y;

    Mux4to1 mux("MUX");
    mux.sel(sel);
    mux.d0(d0);
    mux.d1(d1);
    mux.d2(d2);
    mux.d3(d3);
    mux.y(y);

    // Tạo waveform
    sc_trace_file *wf = sc_create_vcd_trace_file("mux_waveform");
    sc_trace(wf, sel, "sel");
    sc_trace(wf, d0, "d0");
    sc_trace(wf, d1, "d1");
    sc_trace(wf, d2, "d2");
    sc_trace(wf, d3, "d3");
    sc_trace(wf, y, "y");

    cout << "Test MUX 4-to-1 (biến đổi d0-d3 theo từng chu
kỳ):\n";

    // Chu kỳ 1: sel = 0, d0 = 1
    sel = 0;
    d0 = 0; d1 = 1; d2 = 1; d3 = 0;
    sc_start(10, SC_NS);
    cout << "sel=" << sel.read() << " y=" << y.read() << endl;

    // Chu kỳ 2: sel = 1, d1 = 1
    sel = 1;
    d0 = 0; d1 = 1; d2 = 0; d3 = 1;
    sc_start(10, SC_NS);
    cout << "sel=" << sel.read() << " y=" << y.read() << endl;

    // Chu kỳ 3: sel = 2, d2 = 1
    sel = 2;
    d0 = 1; d1 = 1; d2 = 1; d3 = 0;
    sc_start(10, SC_NS);
    cout << "sel=" << sel.read() << " y=" << y.read() << endl;

    // Chu kỳ 4: sel = 3, d3 = 1
    sel = 3;
    d0 = 0; d1 = 0; d2 = 0; d3 = 1;
    sc_start(10, SC_NS);
    cout << "sel=" << sel.read() << " y=" << y.read() << endl;

    sc_close_vcd_trace_file(wf);
    return 0;
}
```

- **SIMULATION OUTPUT:**



```
tubi@tubi:~/Desktop/4_1$ ./sim

        SystemC 2.3.3-Accellera --- Mar 21 2025 01:05:15
        Copyright (c) 1996-2018 by all Contributors,
        ALL RIGHTS RESERVED
Test MUX 4-to-1 (biến đổi d0-d3 theo từng chu kỳ):

Info: (I702) default timescale unit used for tracing: 1 ps (mux_waveform.vcd)
sel=0 y=0
sel=1 y=1
sel=2 y=0
sel=3 y=1
tubi@tubi:~/Desktop/4_1$ S
```



```
[0] start time.
[40000] end time.
```

7.1.2. On HLS vivado

- **Source code:**

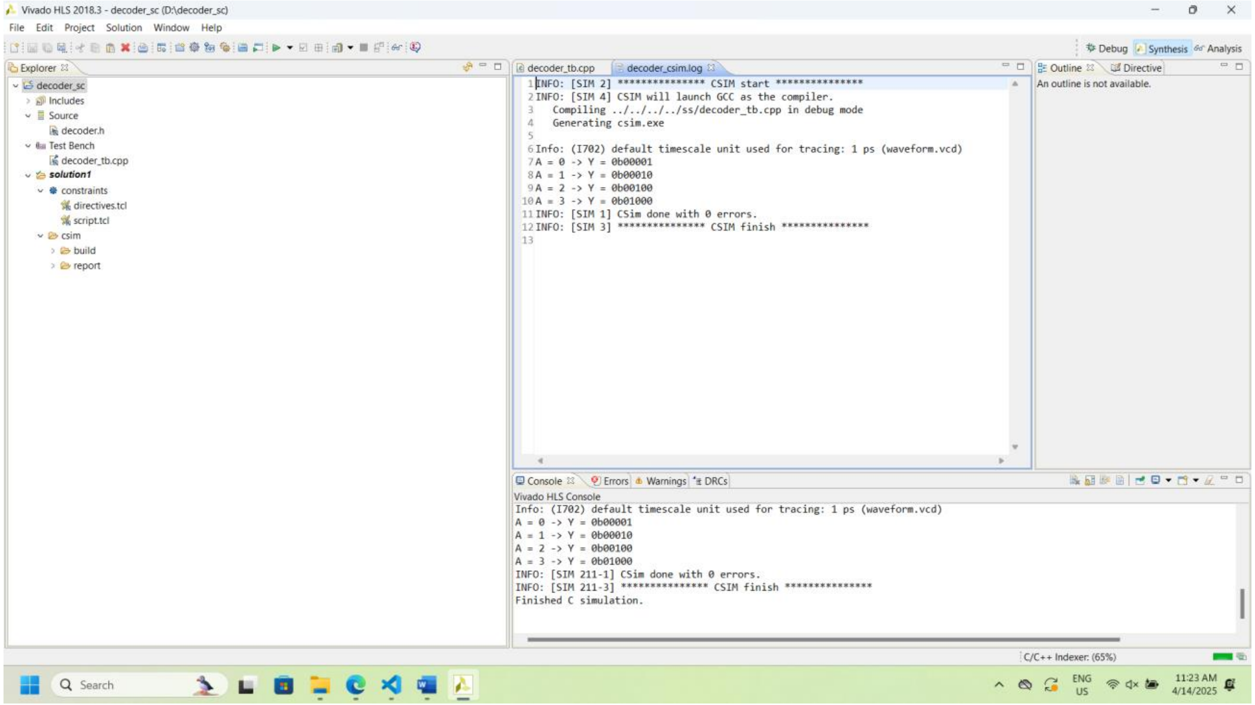
```
#include "mux_func.h"
void mux4to1(
    ap_uint<2> sel,
    ap_uint<1> d0,
    ap_uint<1> d1,
    ap_uint<1> d2,
    ap_uint<1> d3,
    ap_uint<1> &y
) {
    switch (sel) {
        case 0: y = d0; break;
        case 1: y = d1; break;
        case 2: y = d2; break;
        case 3: y = d3; break;
        default: y = 0; break;
    }
}
```

- **Library:**

```
#ifndef MUX_FUNC_H
#define MUX_FUNC_H

#include <ap_int.h>

// Hàm top-level cho Vivado HLS
void mux4to1(
    ap_uint<2> sel,
    ap_uint<1> d0,
    ap_uint<1> d1,
    ap_uint<1> d2,
    ap_uint<1> d3,
    ap_uint<1> &y
);

#endif
```

- **Test bench:**

```
#include <iostream>
#include "mux_func.h"

int main() {
    ap_uint<2> sel;
    ap_uint<1> d0 = 1, d1 = 0, d2 = 1, d3 = 0;
    ap_uint<1> y;

    std::cout << "=== MUX 4:1 TEST ===" << std::endl;

    for (int i = 0; i < 4; ++i) {
        sel = i;
        mux4to1(sel, d0, d1, d2, d3, y);
        std::cout << "sel = " << sel << " -> y = " << y << std::endl;
    }

    return 0;
}
```



```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
   Compiling ../../../../ss/decoder_tb.cpp in debug mode
   Generating csim.exe

Info: (I702) default timescale unit used for tracing: 1 ps (waveform.vcd)
A = 0 -> Y = 0b00001
A = 1 -> Y = 0b00010
A = 2 -> Y = 0b00100
A = 3 -> Y = 0b01000
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

```
Info: (I702) default timescale unit used for tracing: 1 ps (waveform.vcd)
A = 0 -> Y = 0b00001
A = 1 -> Y = 0b00010
A = 2 -> Y = 0b00100
A = 3 -> Y = 0b01000
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] *************** CSIM finish ***************
Finished C simulation.
```

7.2. 2-to-4 decoder

7.2.1. On Ubuntu

- SOURCE CODE:

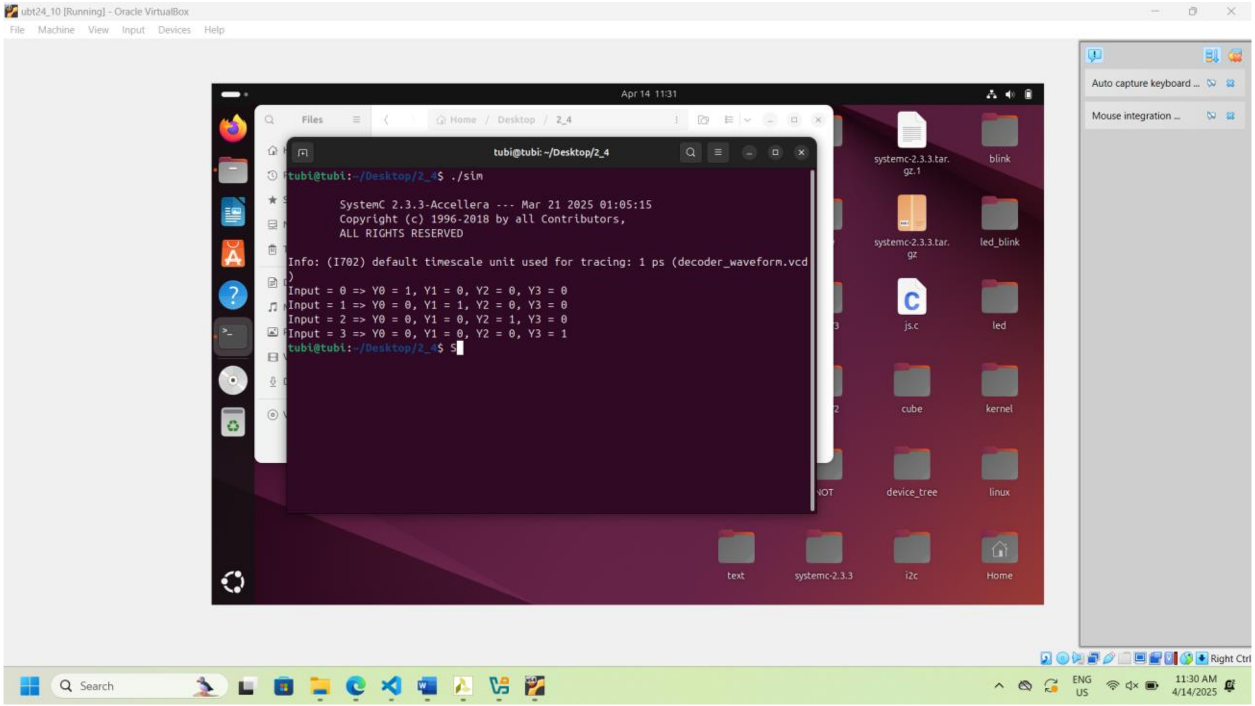
```
#include <systemc.h>
#include "decoder2to4.cpp"

int sc_main(int argc, char* argv[]) {
    sc_signal<sc_uint<2>> in;
    sc_signal<bool> y0, y1, y2, y3;

    Decoder2to4 decoder("DECODER");
    decoder.in(in);
    decoder.y0(y0);
    decoder.y1(y1);
    decoder.y2(y2);
    decoder.y3(y3);

    for (int i = 0; i < 4; i++) {
        in = i;
        sc_start(1, SC_NS);
        cout << "Input = " << in.read() << " => "
            << "Y0 = " << y0.read() << ", "
```

```
             << "Y1 = " << y1.read() << ", "
             << "Y2 = " << y2.read() << ", "
             << "Y3 = " << y3.read() << endl;
    }

    return 0;
}
```

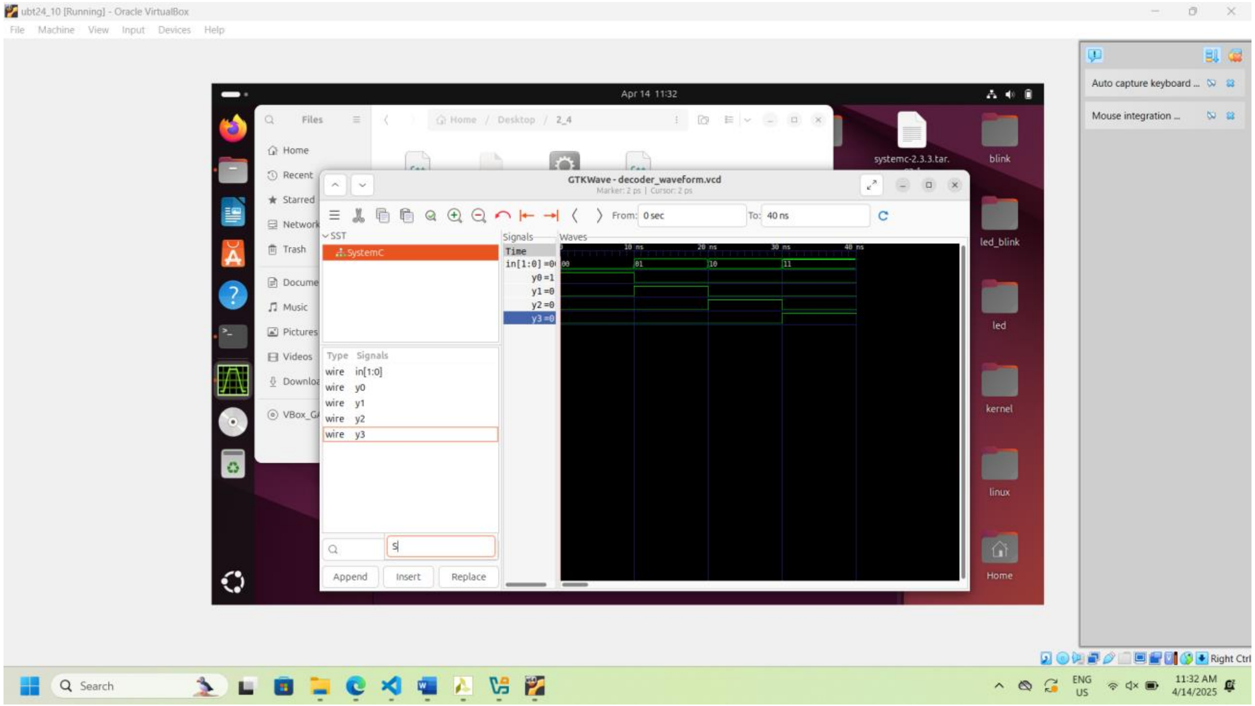
- TEST BENCH:

```
#include <systemc.h>
#include "decoder2to4.cpp"

int sc_main(int argc, char* argv[]) {
    sc_signal<sc_uint<2>> in;
    sc_signal<bool> y0, y1, y2, y3;

    Decoder2to4 decoder("DECODER");
    decoder.in(in);
    decoder.y0(y0);
    decoder.y1(y1);
    decoder.y2(y2);
    decoder.y3(y3);

    // Tạo file waveform
    sc_trace_file *wf =
sc_create_vcd_trace_file("decoder_waveform");
    sc_trace(wf, in, "in");
    sc_trace(wf, y0, "y0");
    sc_trace(wf, y1, "y1");
    sc_trace(wf, y2, "y2");
    sc_trace(wf, y3, "y3");

    // Test lần lượt các giá trị đầu vào
    for (int i = 0; i < 4; i++) {
        in = i;
        sc_start(10, SC_NS);  // Đợi 10ns cho mỗi giá trị
        cout << "Input = " << in.read() << " => "
             << "Y0 = " << y0.read() << ", "
             << "Y1 = " << y1.read() << ", "
             << "Y2 = " << y2.read() << ", "
             << "Y3 = " << y3.read() << endl;
    }

    // Đóng waveform
    sc_close_vcd_trace_file(wf);
    return 0;
}
```

- **SIMULATION OUTPUT:**



```
tubi@tubi:~/Desktop/2_4$ ./sim

        SystemC 2.3.3-Accellera --- Mar 21 2025 01:05:15
        Copyright (c) 1996-2018 by all Contributors,
        ALL RIGHTS RESERVED

Info: (I702) default timescale unit used for tracing: 1 ps (decoder_waveform.vcd
)
Input = 0 => Y0 = 1, Y1 = 0, Y2 = 0, Y3 = 0
Input = 1 => Y0 = 0, Y1 = 1, Y2 = 0, Y3 = 0
Input = 2 => Y0 = 0, Y1 = 0, Y2 = 1, Y3 = 0
Input = 3 => Y0 = 0, Y1 = 0, Y2 = 0, Y3 = 1
tubi@tubi:~/Desktop/2_4$ S
```



7.2.2. On HLS vivado

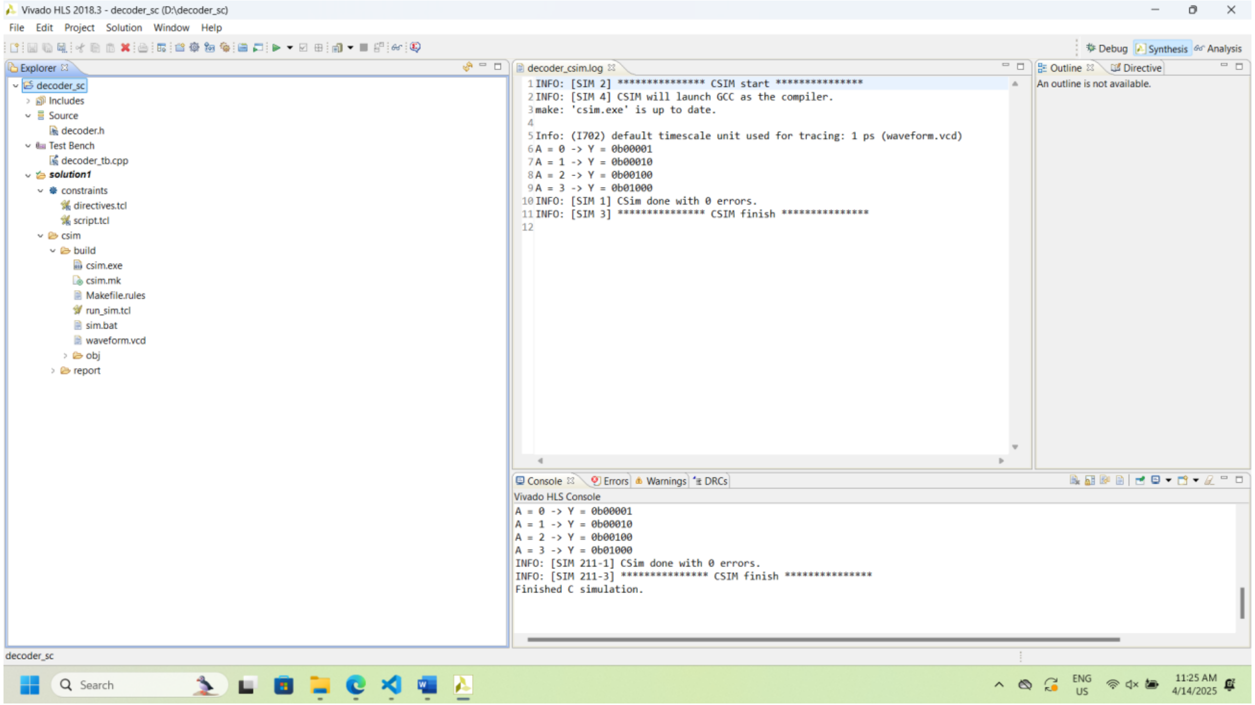
```
D:.
|  .cproject
|  .project
|  .vivado_hls_log_all.xml
|  vivado_hls.app
|
+---.apc
|  |  autopilot.apfmapping
|  |
|  +---.src
|  \---.tb
+---.settings
|      decoder_sc.Debug.launch
|      decoder_sc.Release.launch
|       language.settings.xml
|
\---solution1
    |   directives.tcl
    |   script.tcl
    |   solution1.aps
    |   solution1.directive
    |   solution1.log
    |
    +---.autopilot
    |   |   .autopilot_exit
    |   |
    |   \---db
    |           .message_csim.xml
    |           .message_syn.xml
    |           autopilot.flow.log
    |           dsp_style
    |
    +---.tcls
    \---csim
        |   .lst_opt.tcl
        |
        +---build
        |   |   csim.exe
        |   |   csim.mk
        |   |   Makefile.rules
        | |  run_sim.tcl

        | |  sim.bat

        | |  waveform.vcd

        | |

        | \---obj

        |        .dir

        |        decoder_tb.d

        |        decoder_tb.o

        |

        \---report
```

- **Library & source code:**

```
#ifndef DECODER_H
#define DECODER_H

#include <systemc.h>

SC_MODULE(decoder) {
    sc_in<sc_uint<2>> A;
    sc_out<sc_uint<4>> Y;

    void decode_process() {
        switch (A.read()) {
            case 0: Y.write(0b0001); break;
            case 1: Y.write(0b0010); break;
            case 2: Y.write(0b0100); break;
            case 3: Y.write(0b1000); break;
            default: Y.write(0); break;
        }
    }

    SC_CTOR(decoder) {
        SC_METHOD(decode_process);
        sensitive << A;
    }
};

#endif // DECODER_H
```

- **Test bench:**

```
#include <iostream>
#include "systemc.h"
#include "decoder.h"

int sc_main(int argc, char* argv[]) {
    sc_signal<sc_uint<2>> A_sig;
    sc_signal<sc_uint<4>> Y_sig;

    decoder uut("decoder");
    uut.A(A_sig);
    uut.Y(Y_sig);

    // Ghi VCD
    sc_trace_file *wf = sc_create_vcd_trace_file("waveform");
    sc_trace(wf, A_sig, "A");
    sc_trace(wf, Y_sig, "Y");

    for (int i = 0; i < 4; ++i) {
        A_sig.write(i);
        sc_start(1, SC_NS);
        std::cout << "A = " << A_sig.read() << " -> Y = " <<
Y_sig.read().to_string(SC_BIN) << std::endl;
    }

    sc_close_vcd_trace_file(wf);
    return 0;
}
```



```
INFO: [SIM 2] *************** CSIM start ***************
INFO: [SIM 4] CSIM will launch GCC as the compiler.
make: 'csim.exe' is up to date.

Info: (I702) default timescale unit used for tracing: 1 ps (waveform.vcd)
A = 0 -> Y = 0b00001
A = 1 -> Y = 0b00010
A = 2 -> Y = 0b00100
A = 3 -> Y = 0b01000
INFO: [SIM 1] CSim done with 0 errors.
INFO: [SIM 3] *************** CSIM finish ***************
```

```
A = 0 -> Y = 0b00001
A = 1 -> Y = 0b00010
A = 2 -> Y = 0b00100
A = 3 -> Y = 0b01000
INFO: [SIM 211-1] CSim done with 0 errors.
INFO: [SIM 211-3] *************** CSIM finish ***************
Finished C simulation.
```