數位系統技術

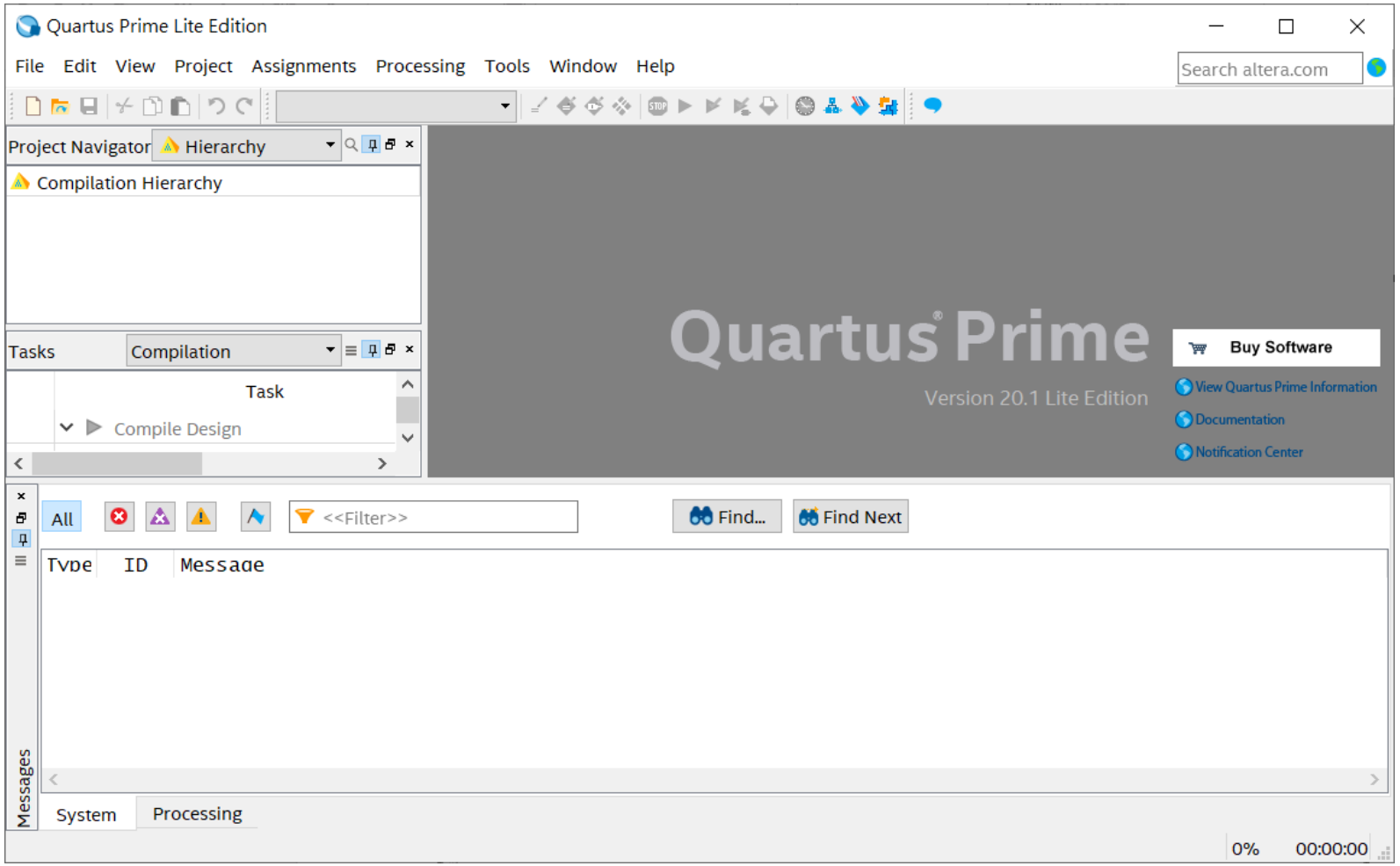
Lab 02
Full Adder 設計

Ren-Der Chen (陳仁德)
Department of Computer Science and
Information Engineering
National Changhua University of Education
E-mail: rdchen@cc.ncue.edu.tw
Spring, 2024

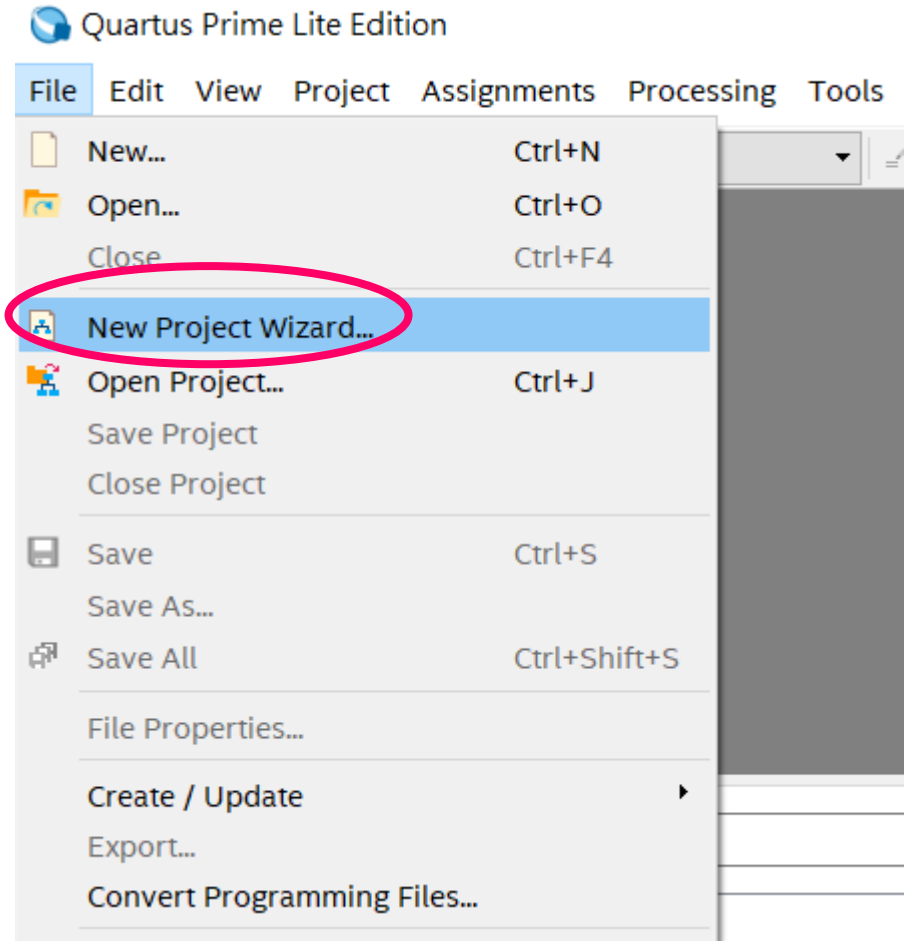
實驗目的

- 熟悉 Quartus 軟體之操作
- 學習如何以schematic方式設計half adder及full adder
- 學習如何對所設計之電路進行functional及timing simulation
- 學習如何將所設計之電路燒錄至FPGA晶片中，並利用實驗板週邊進行電路功能驗證

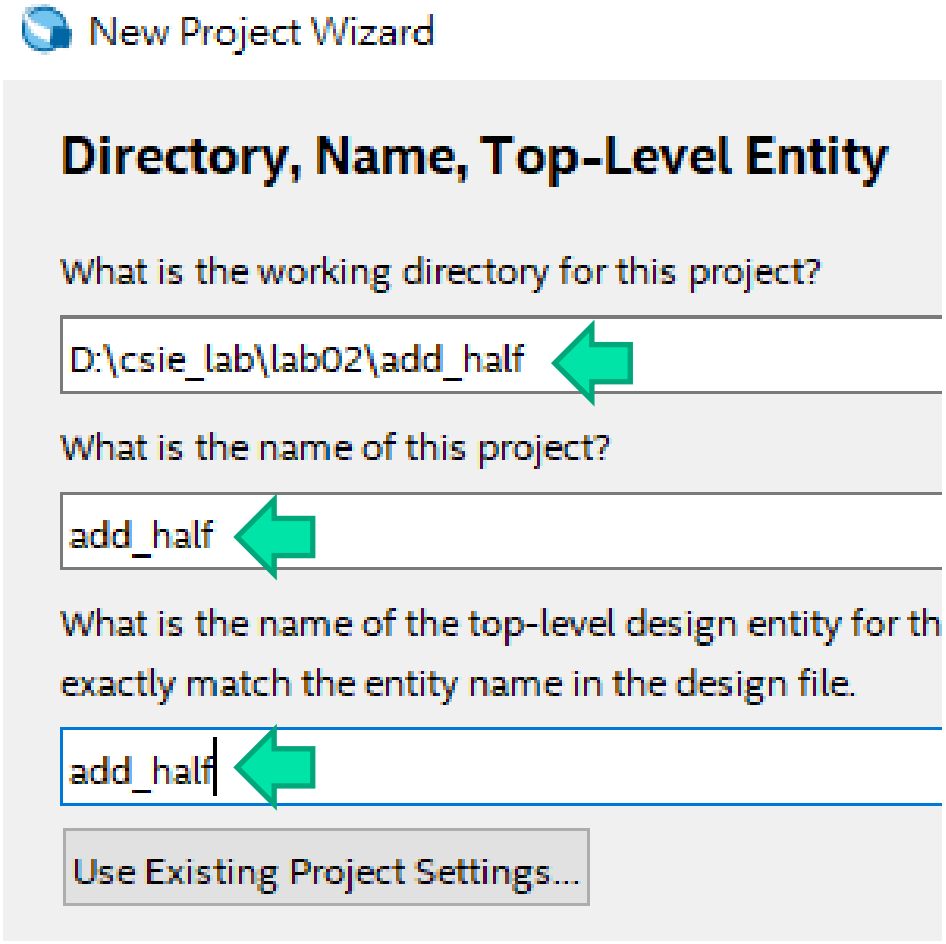
Get Started


Quartus Prime Lite Edition
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
Project Navigator Hierarchy
Compilation Hierarchy
Tasks Compilation
Task
Compile Design
Quartus Prime
Version 20.1 Lite Edition
Buy Software
View Quartus Prime Information
Documentation
Notification Center
All
<<Filter>>
Find...
Find Next
Type ID Message
Messages
System
Processing
0%
00:00:00

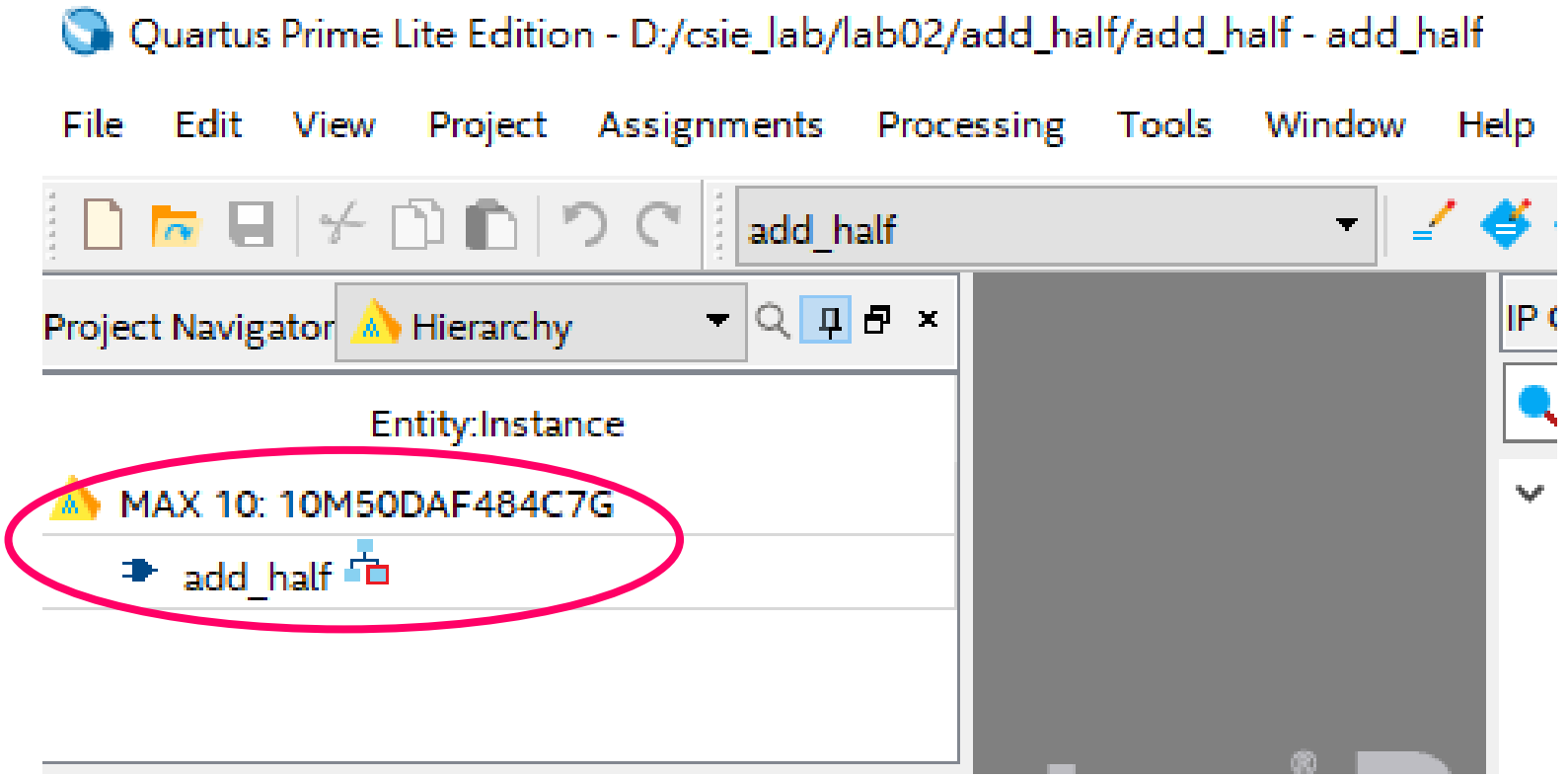
Create a New Project



Create a New Project - add_half



Select a Device



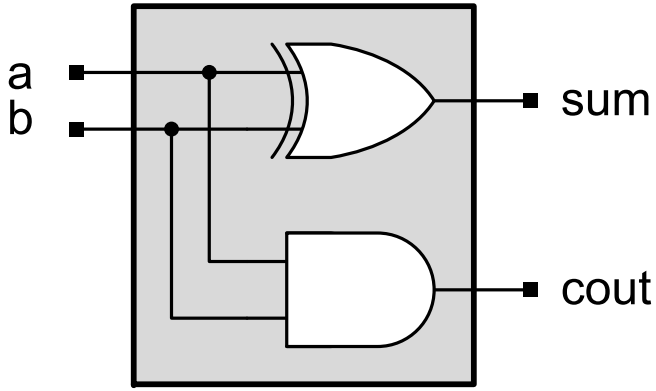
Design Example: Half Adder

a\b	0	1
0	0	1
1	1	0

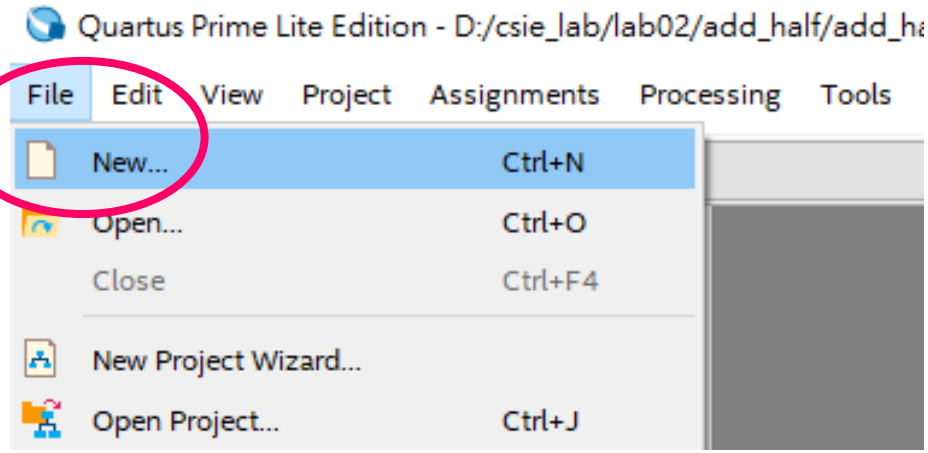
sum = a ⊕ b

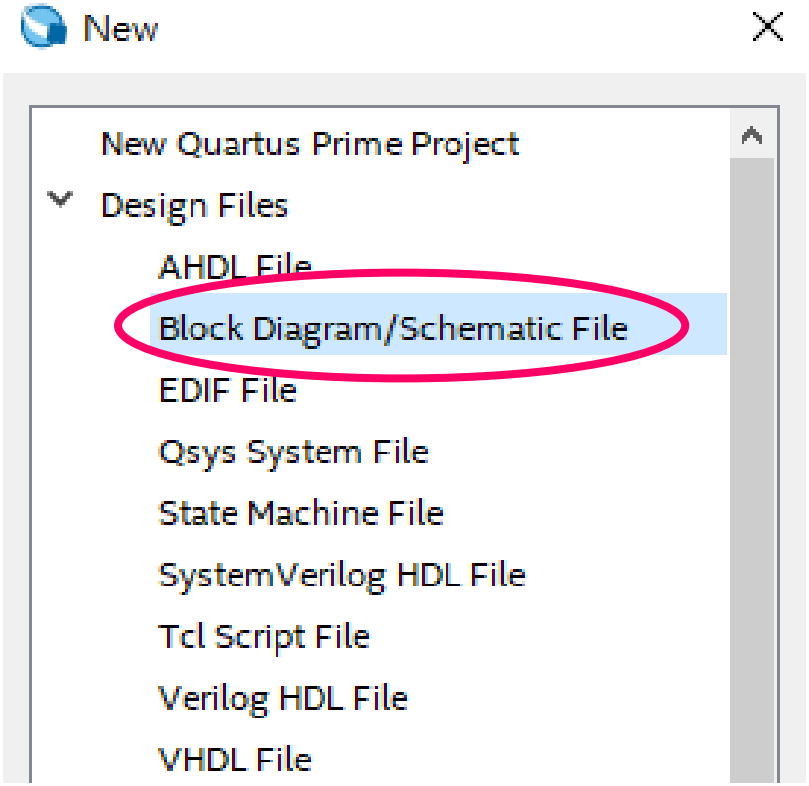
a\b	0	1
0	0	0
1	0	1

cout = a · b

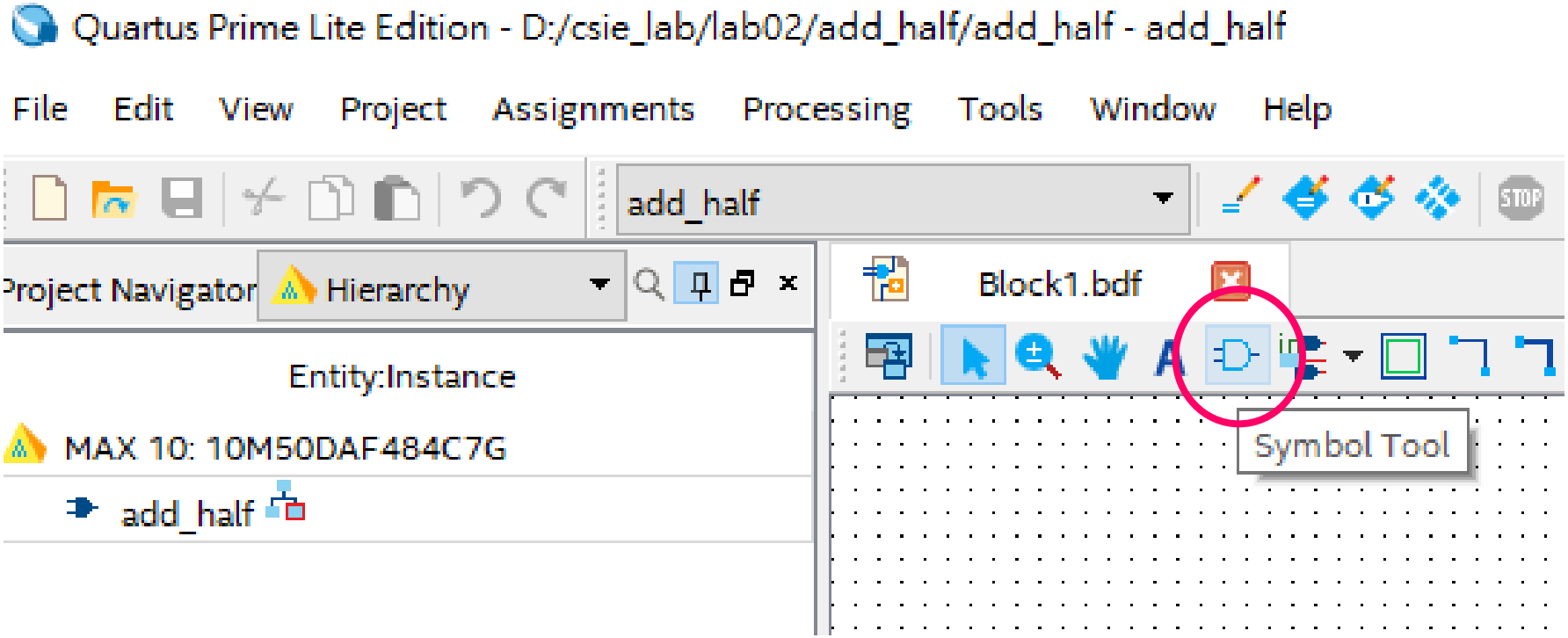


Use Block Editor for Schematic Design

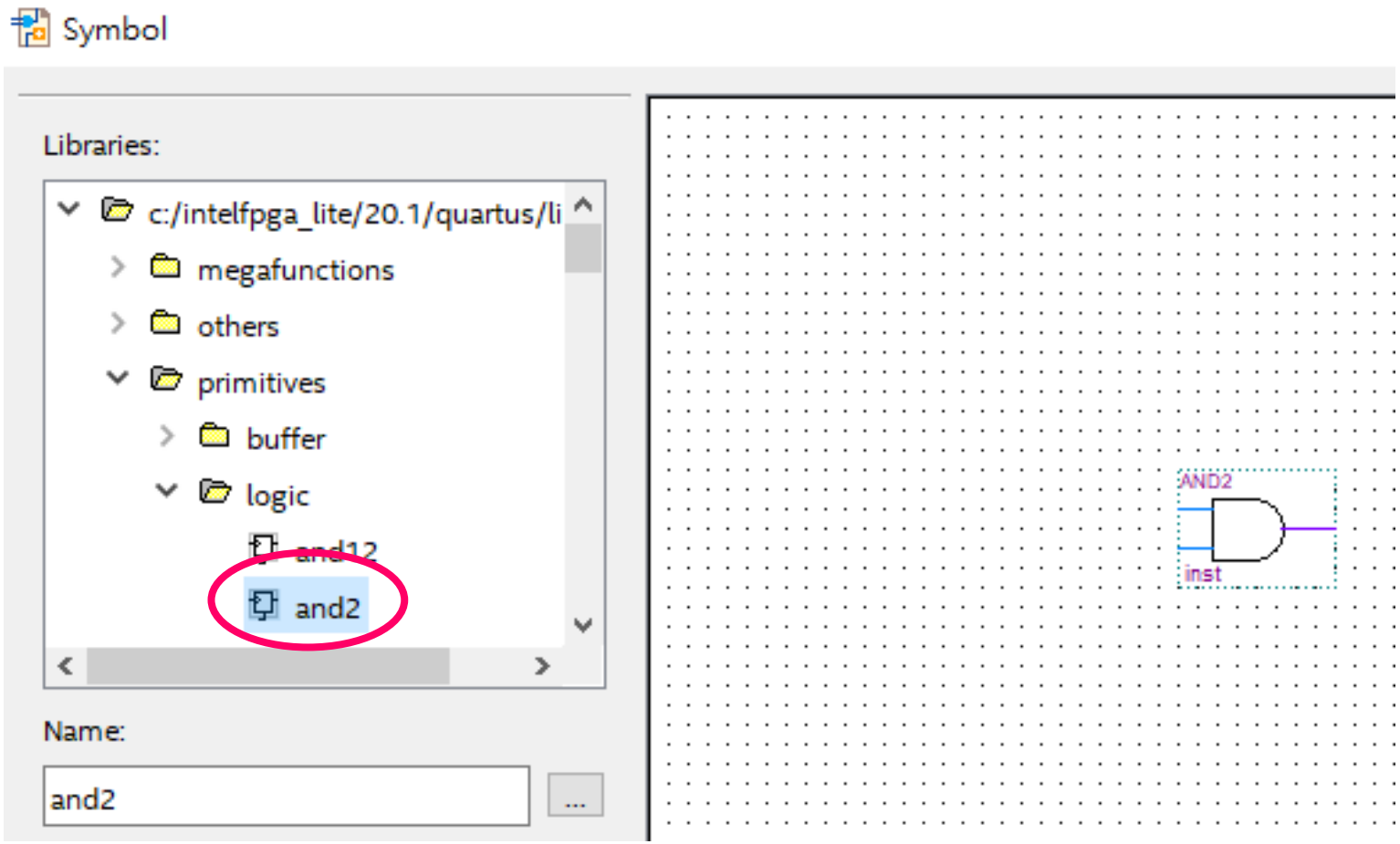




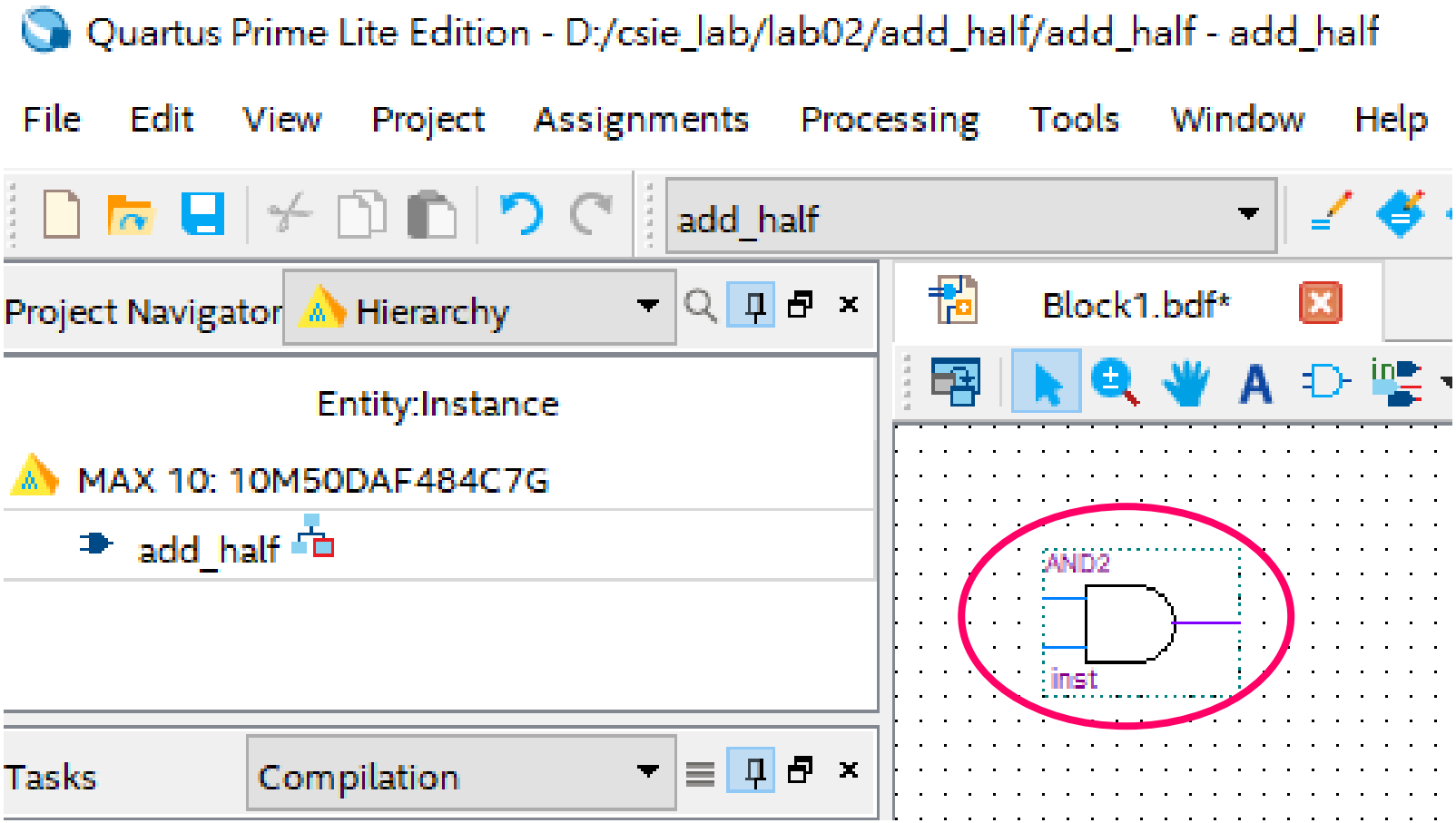
Import Logic Gates



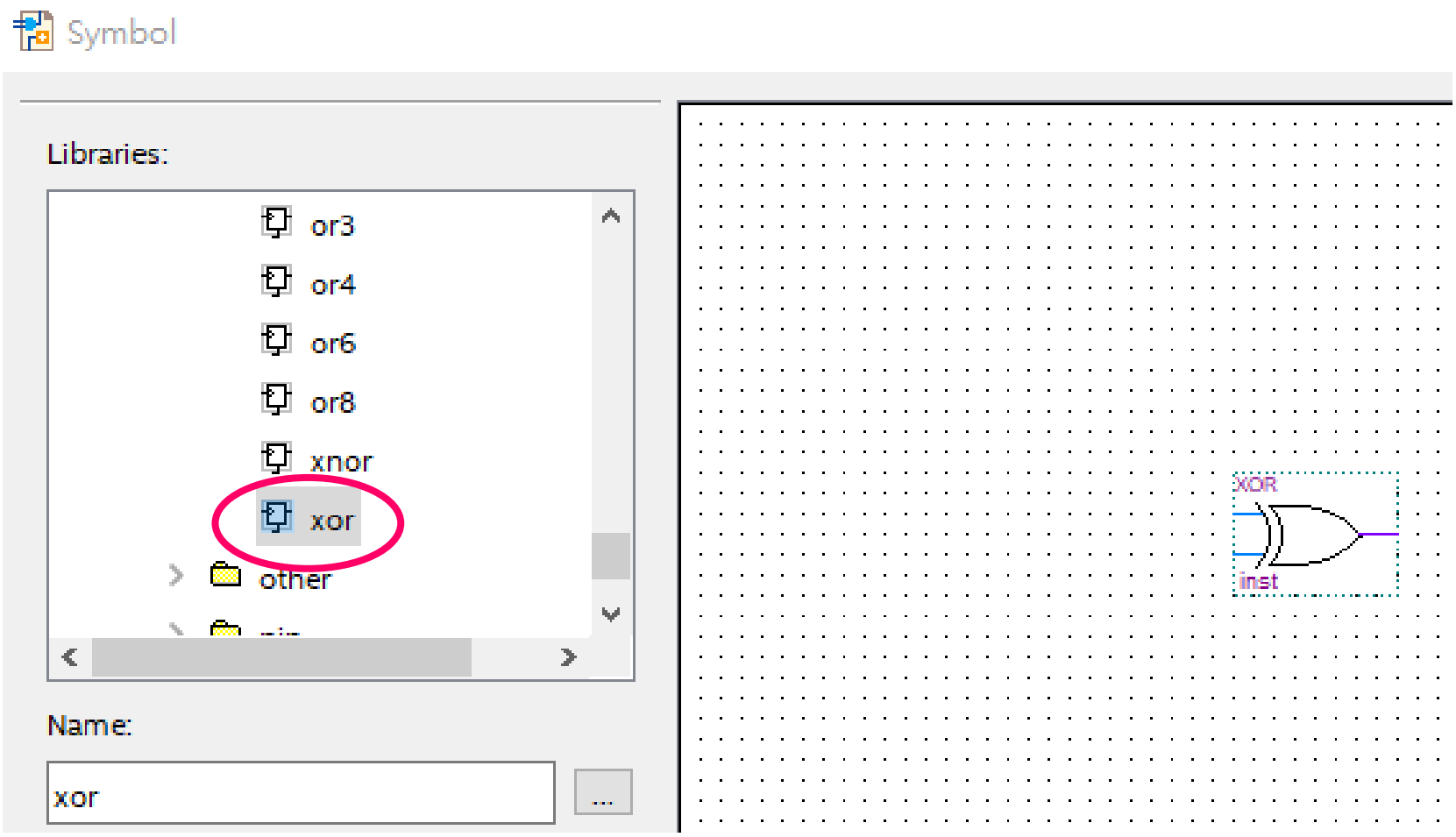
Select and Import an AND2 Gate



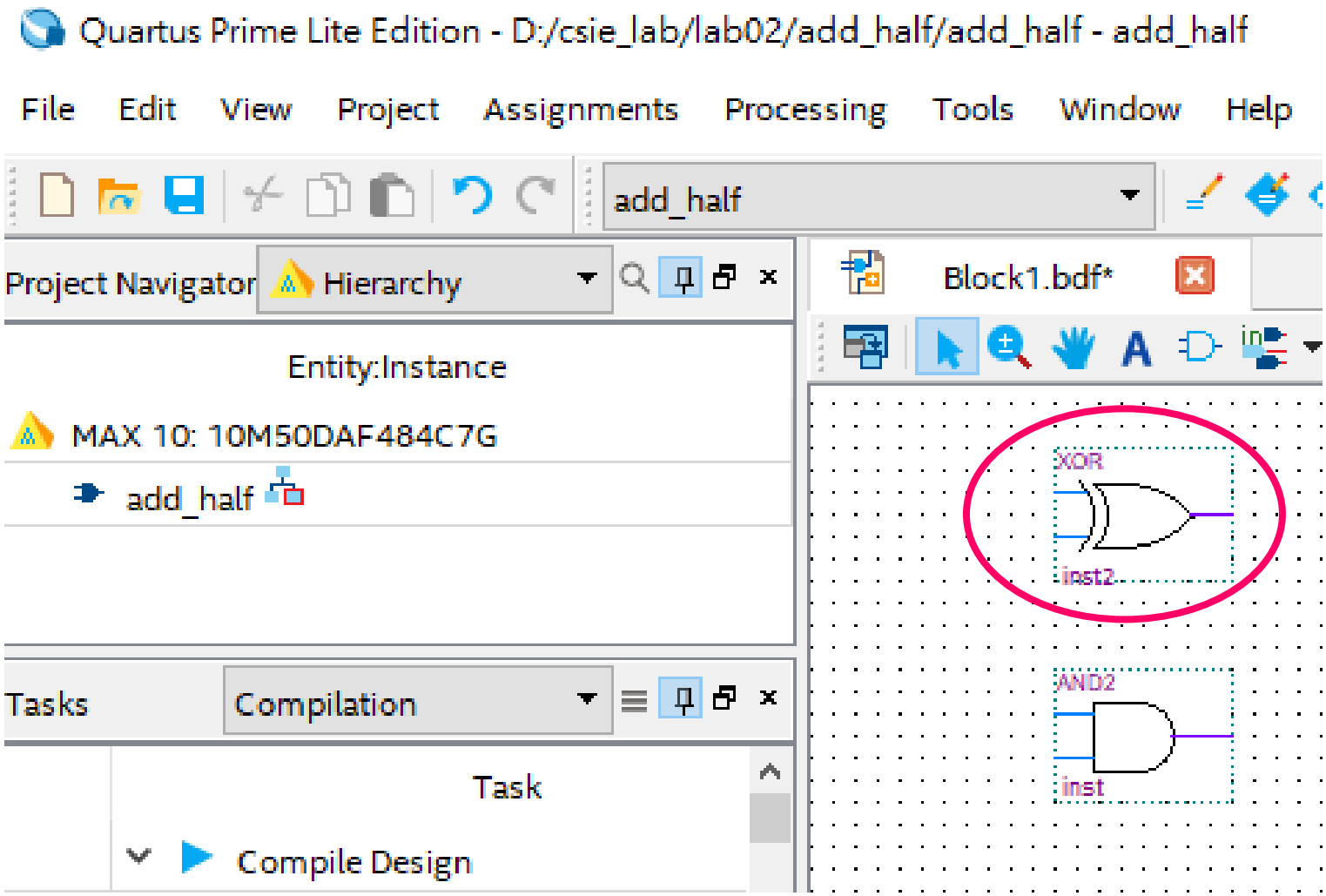
An AND2 Gate is Imported



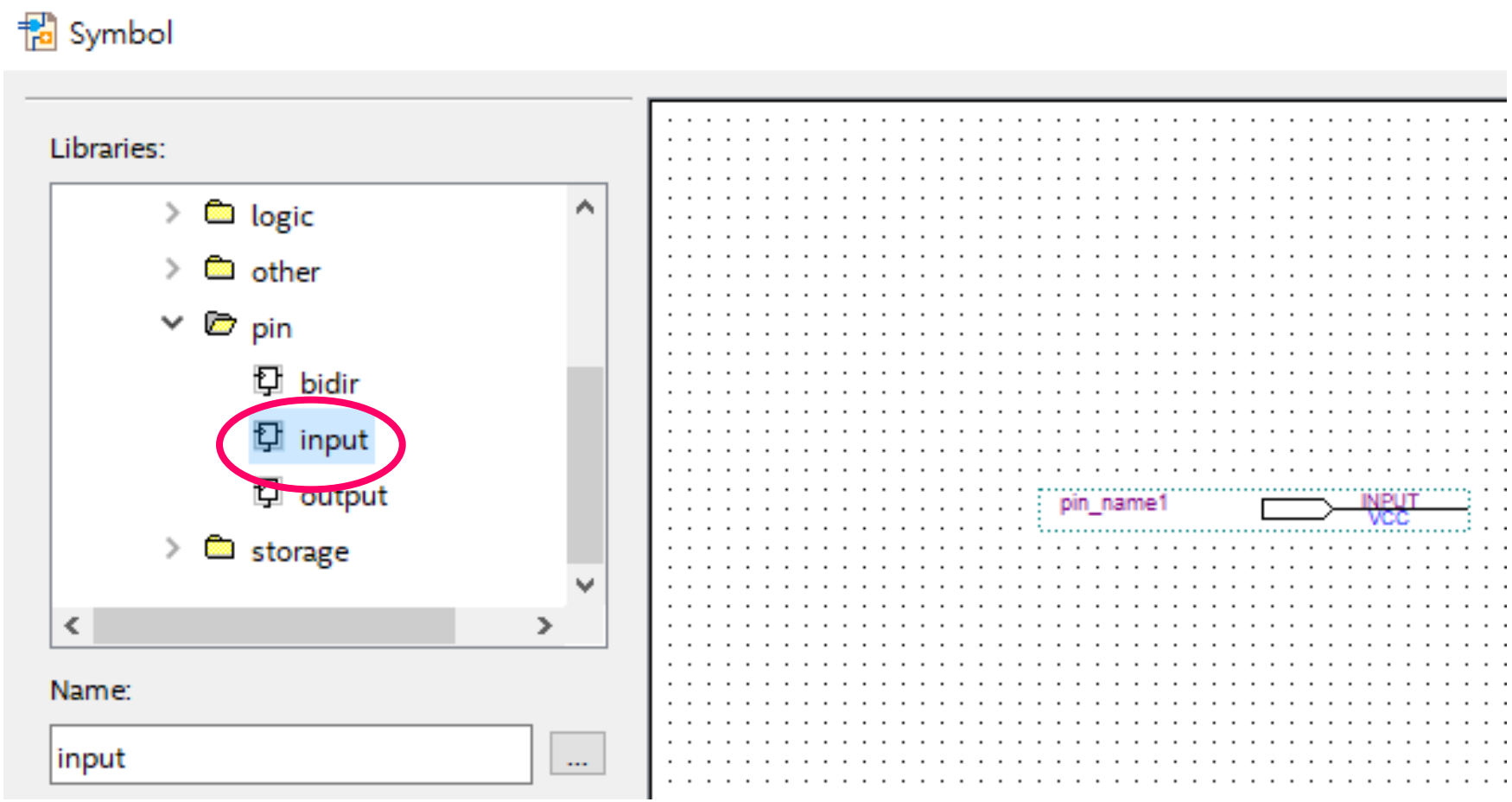
Select and Import an XOR Gate



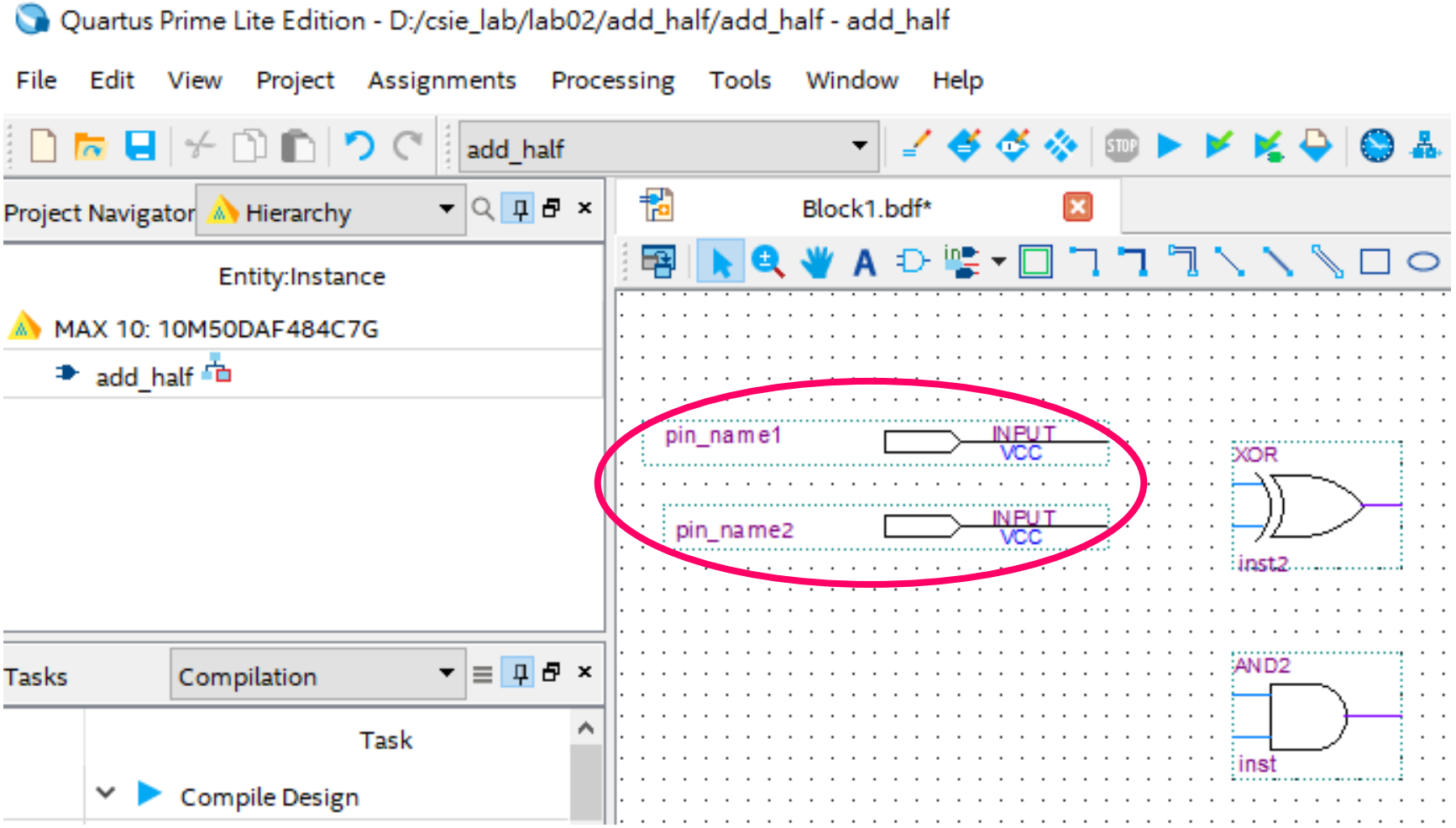
An XOR Gate is Imported



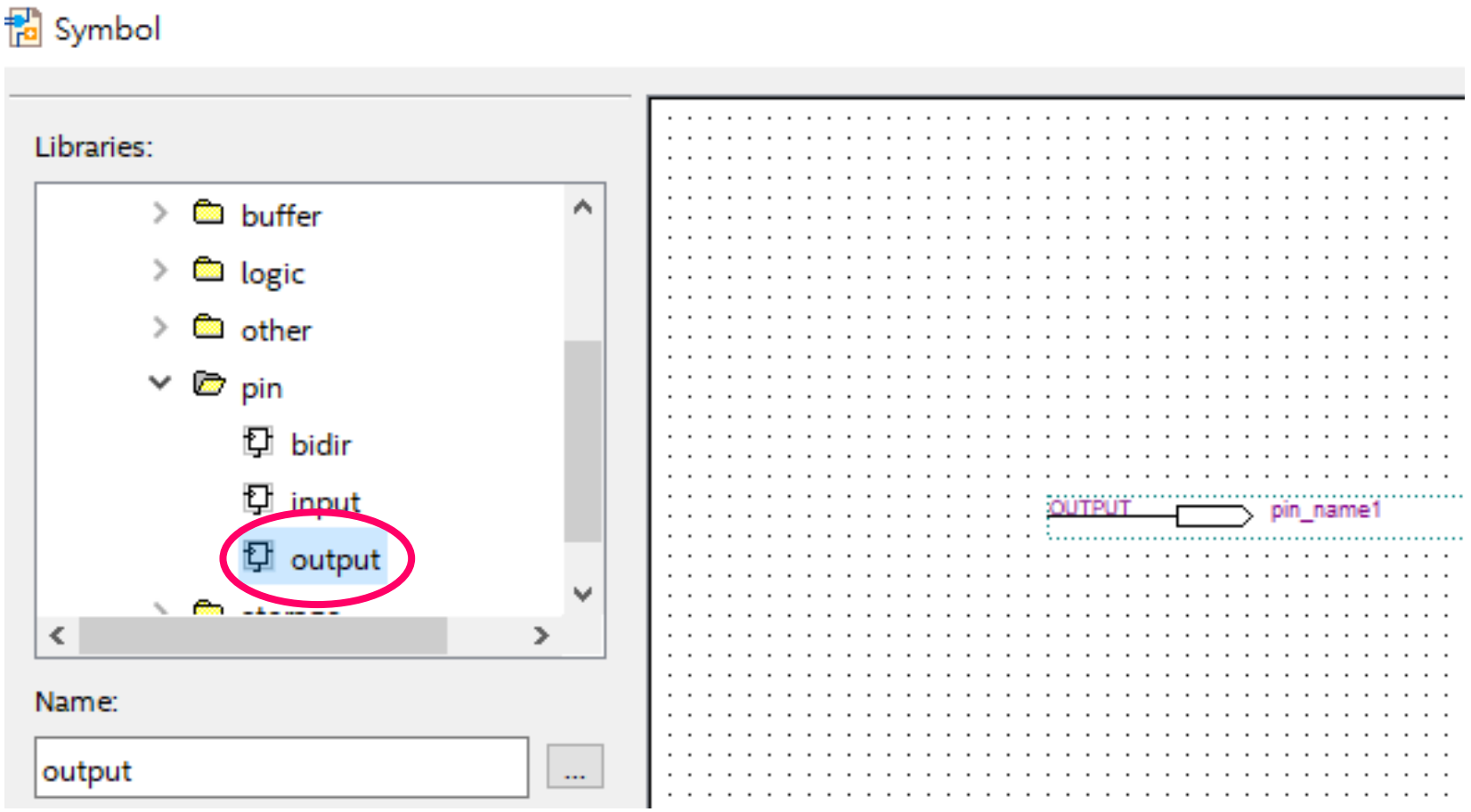
Select and Import an Input Pin



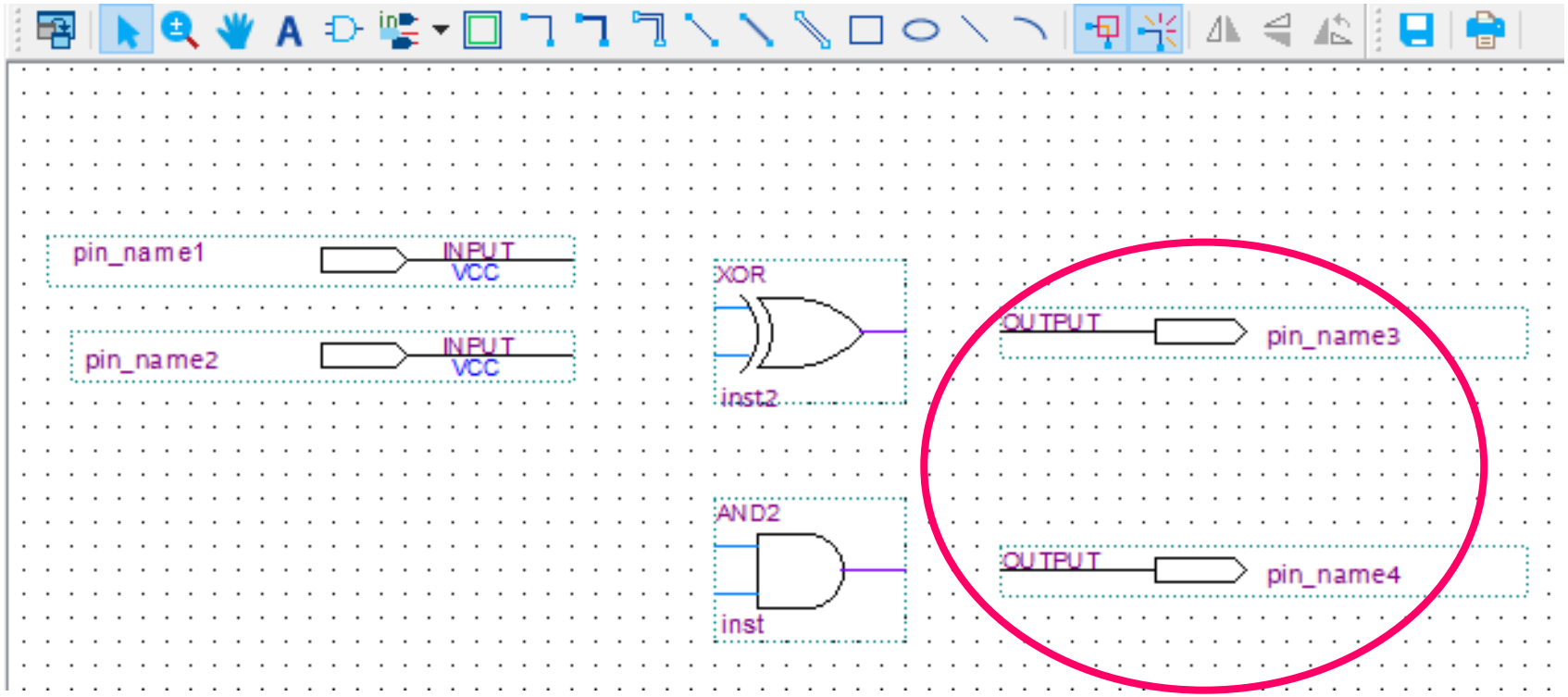
Two Input Pins are Imported



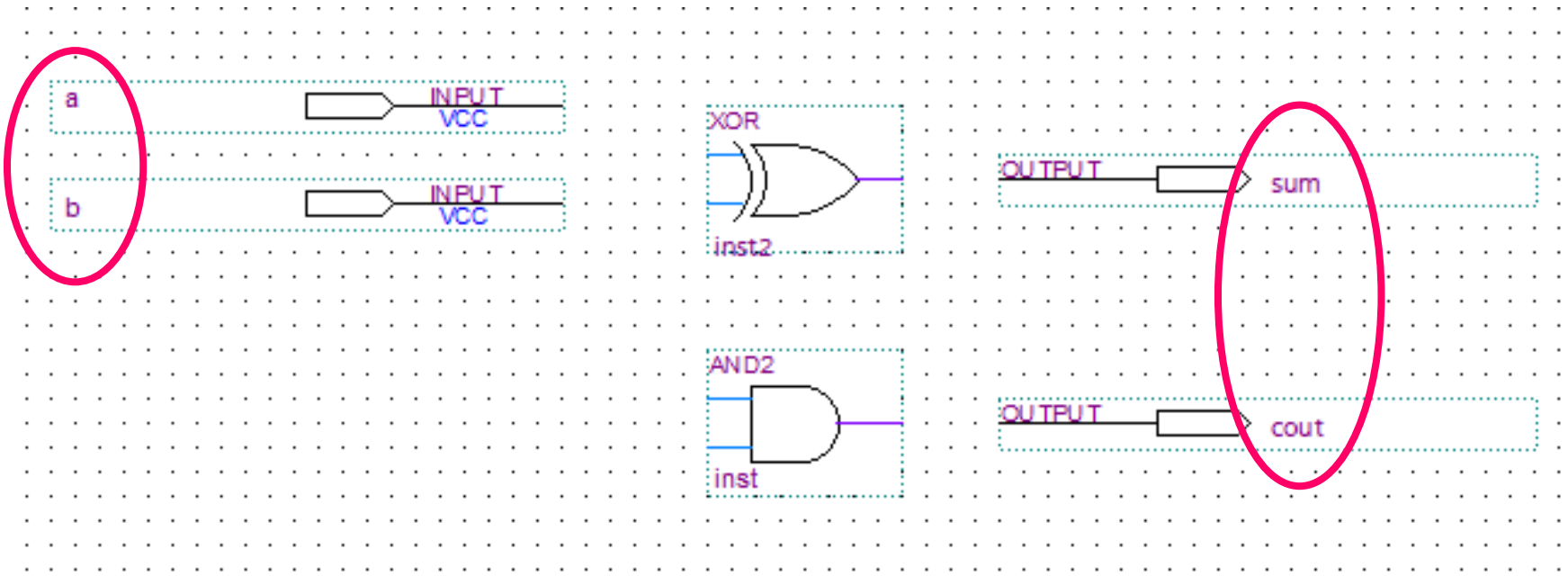
Select and Import an Output Pin



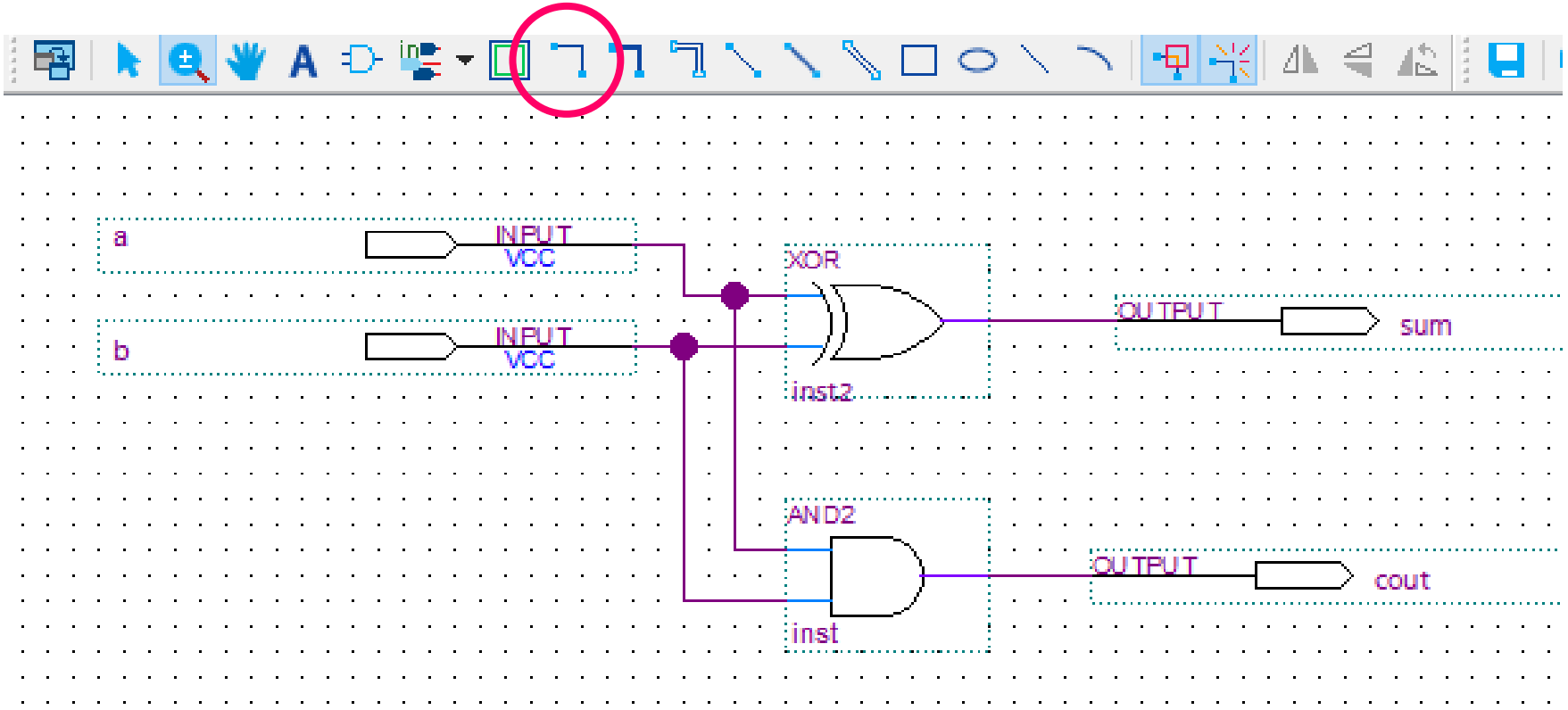
Two Output Pins are Imported



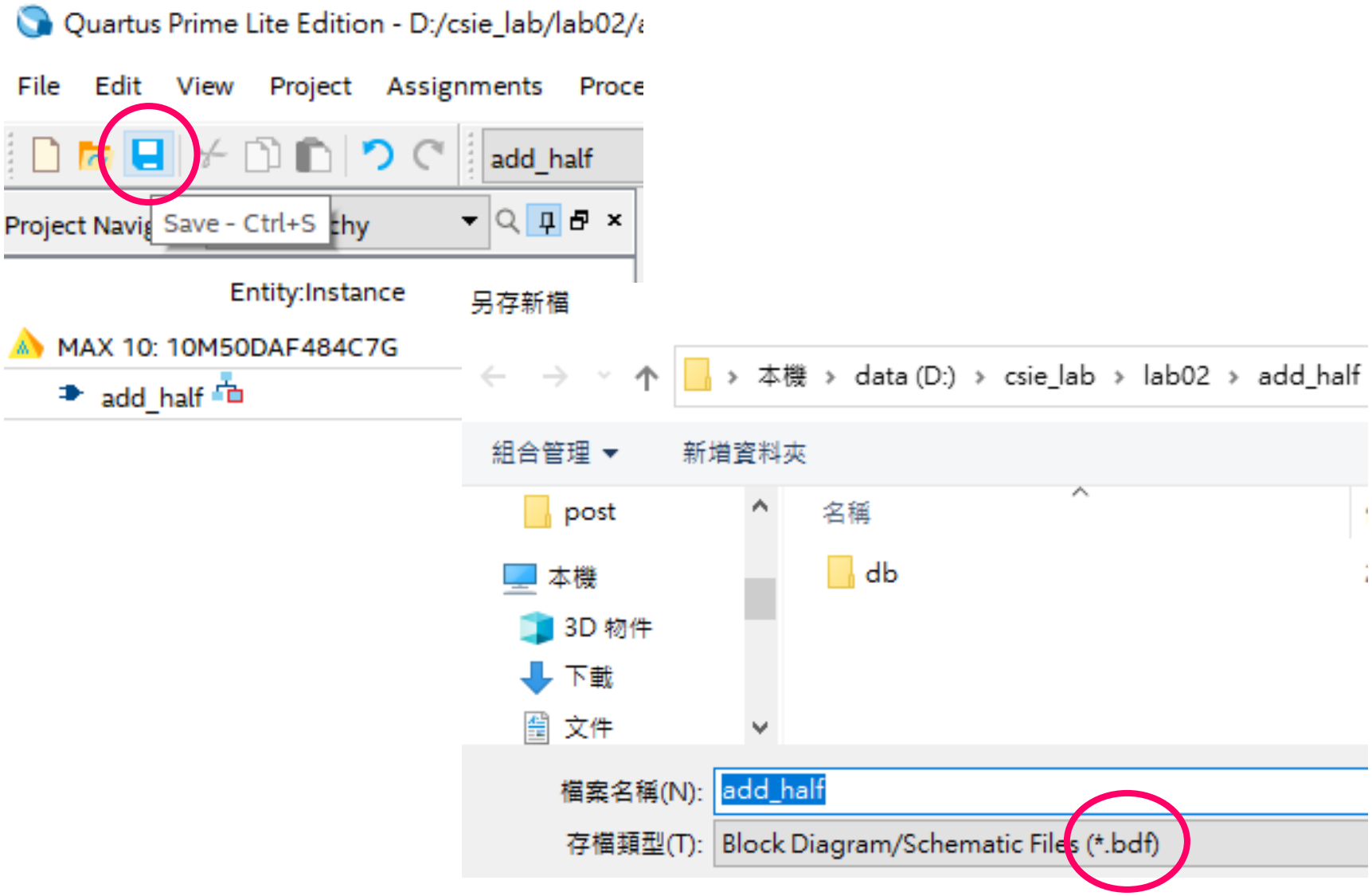
Assign Names to I/O Pins



Connect Nodes with Wires



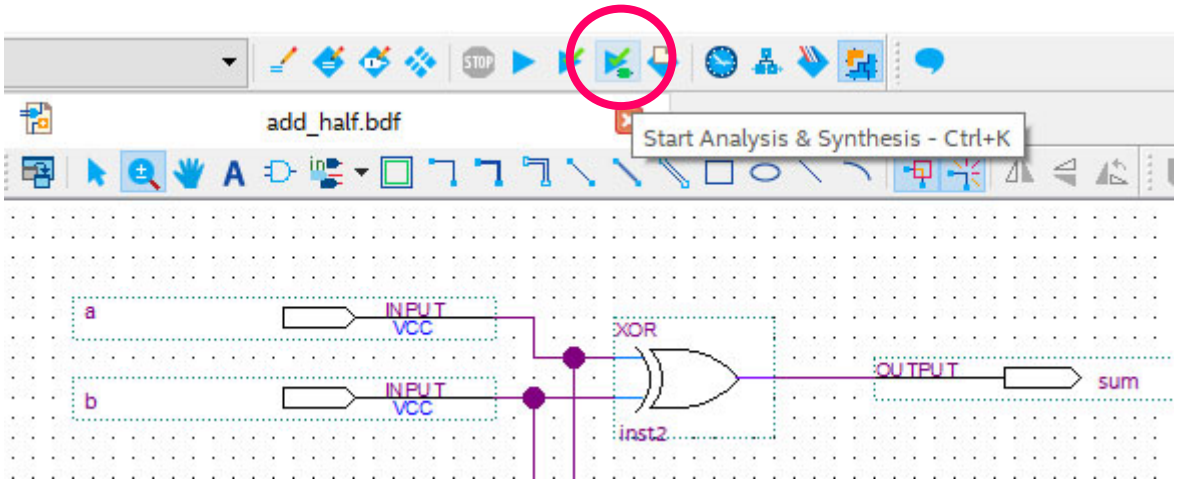
Save the Schematic File



Synthesize the Schematic Circuit

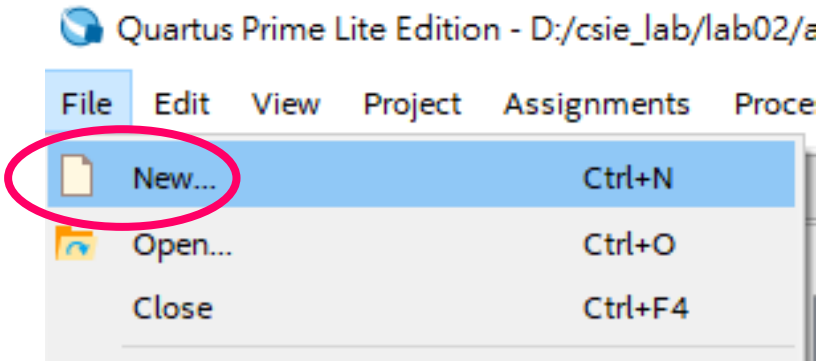
- The synthesis tool translates the schematic into logic expressions.
- Then the technology mapping step determines how each logic expression should be implemented in the logic elements (LEs) available in the target chip.
- The ***Analysis & Synthesis*** function performs the synthesis step.
- It produces a circuit of logic elements, where each element can be directly implemented in the target chip.

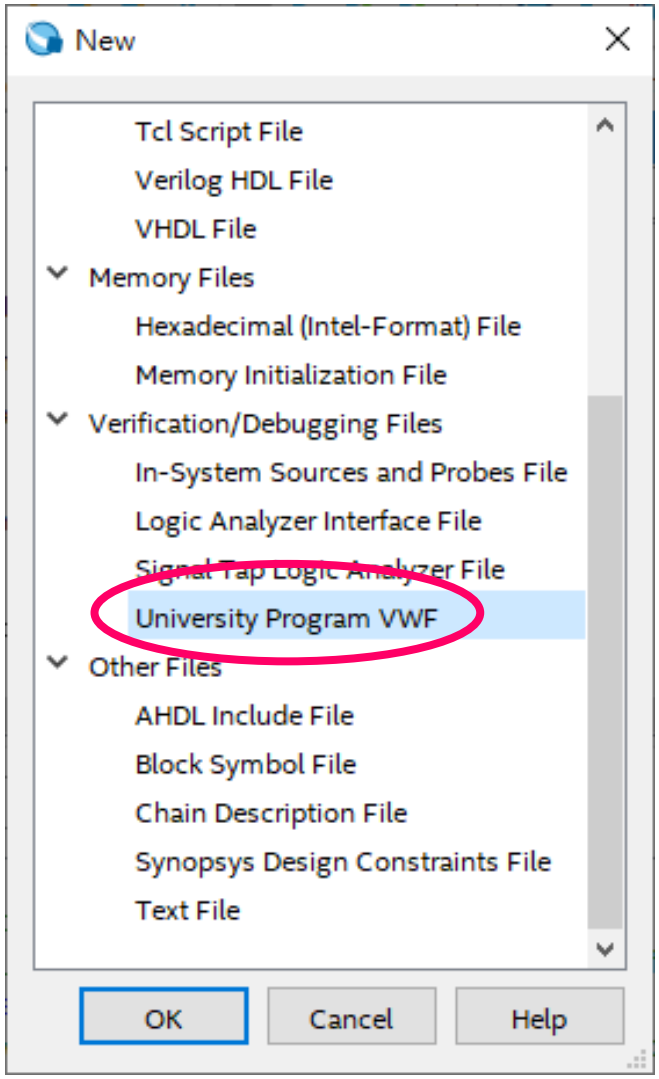
Run “Analysis & Synthesis” for Functional Simulation



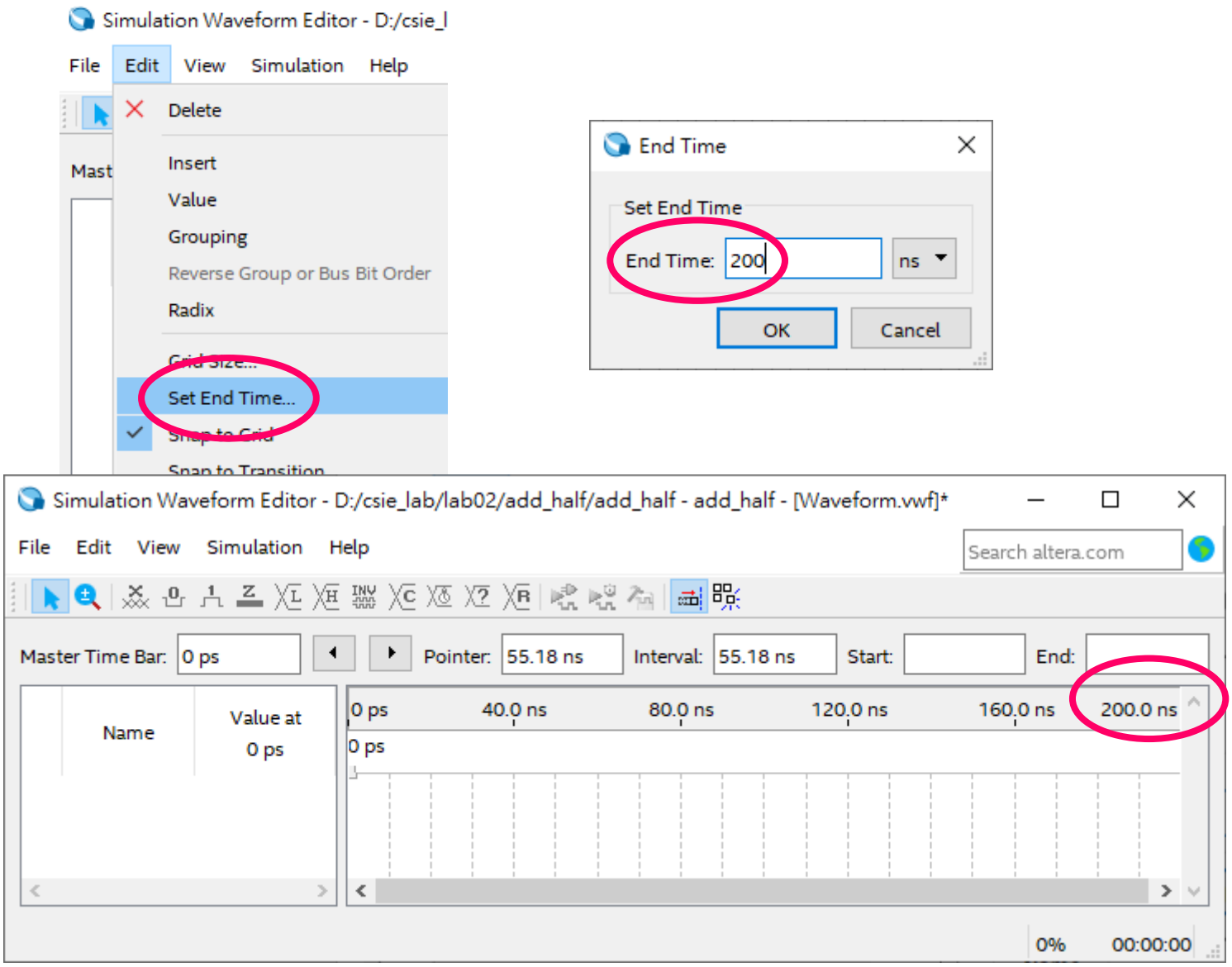
Type	ID	Message
		Running Quartus Prime Analysis & Synthesis
		Command: quartus_map --read_settings_files=on --write_settings_files=off add_half -c add_half
	18236	Number of processors has not been specified which may cause overloading on shared machines.
	20030	Parallel compilation is enabled and will use 8 of the 8 processors detected
	12021	Found 1 design units, including 1 entities, in source file add_half.bdf
	12127	Elaborating entity "add_half" for the top level hierarchy
	286030	Timing-Driven Synthesis is running
	16010	Generating hard_block partition "hard_block:auto_generated_inst"
	21057	Implemented 6 device resources after synthesis - the final resource count might be different
		Quartus Prime Analysis & Synthesis was successful. 0 errors, 1 warning

Create a Waveform File

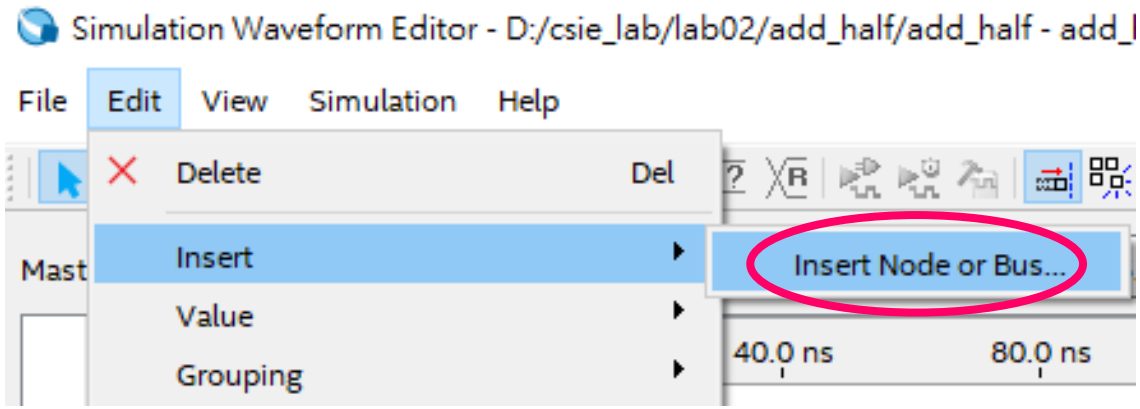


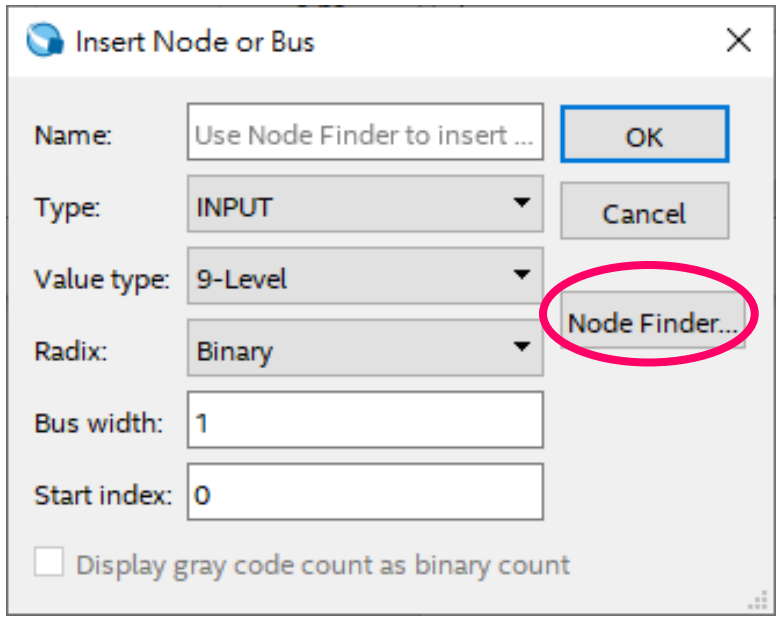


Set End Time for Simulation

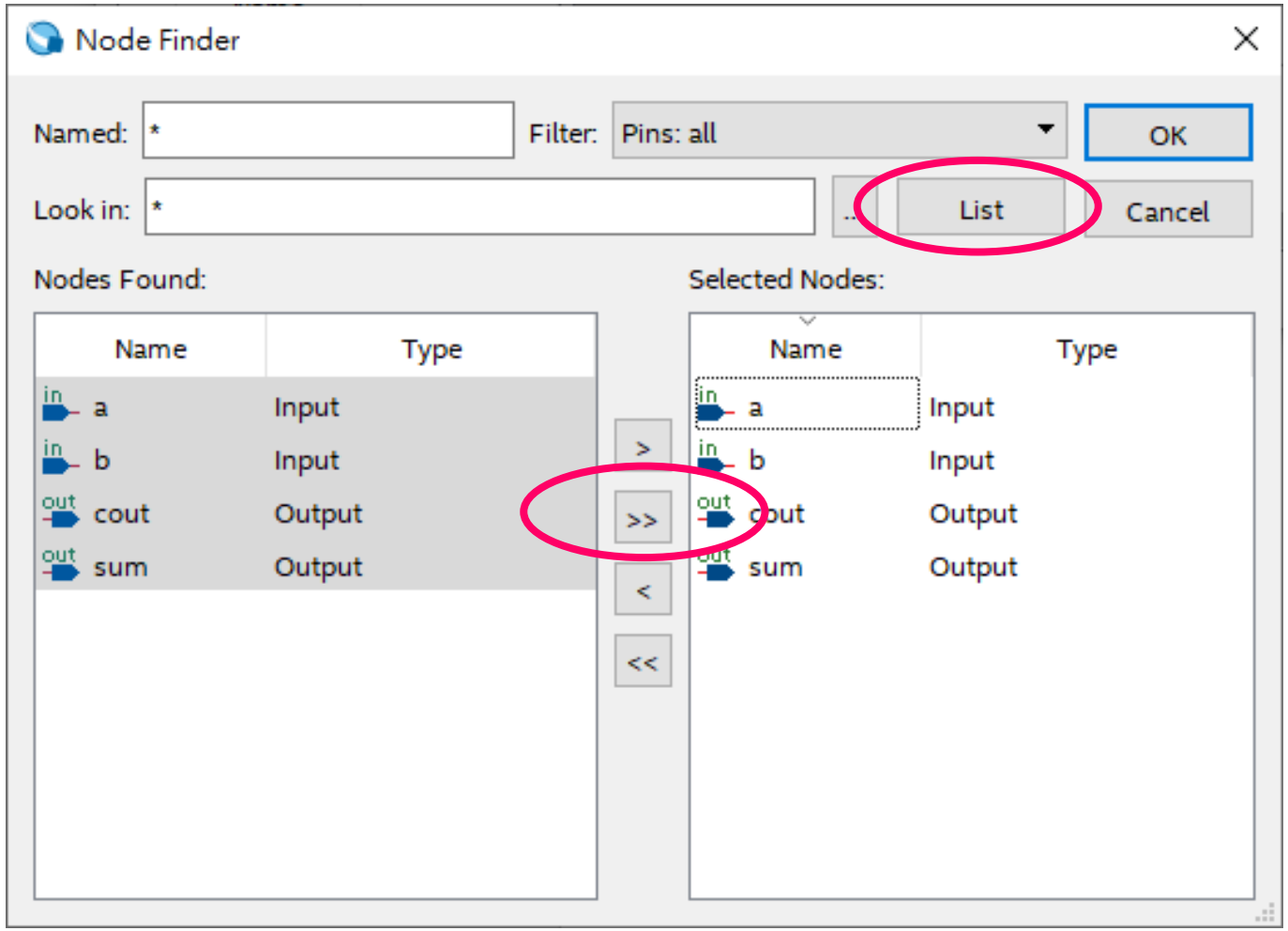


Insert Node for I/O Signals (1/3)

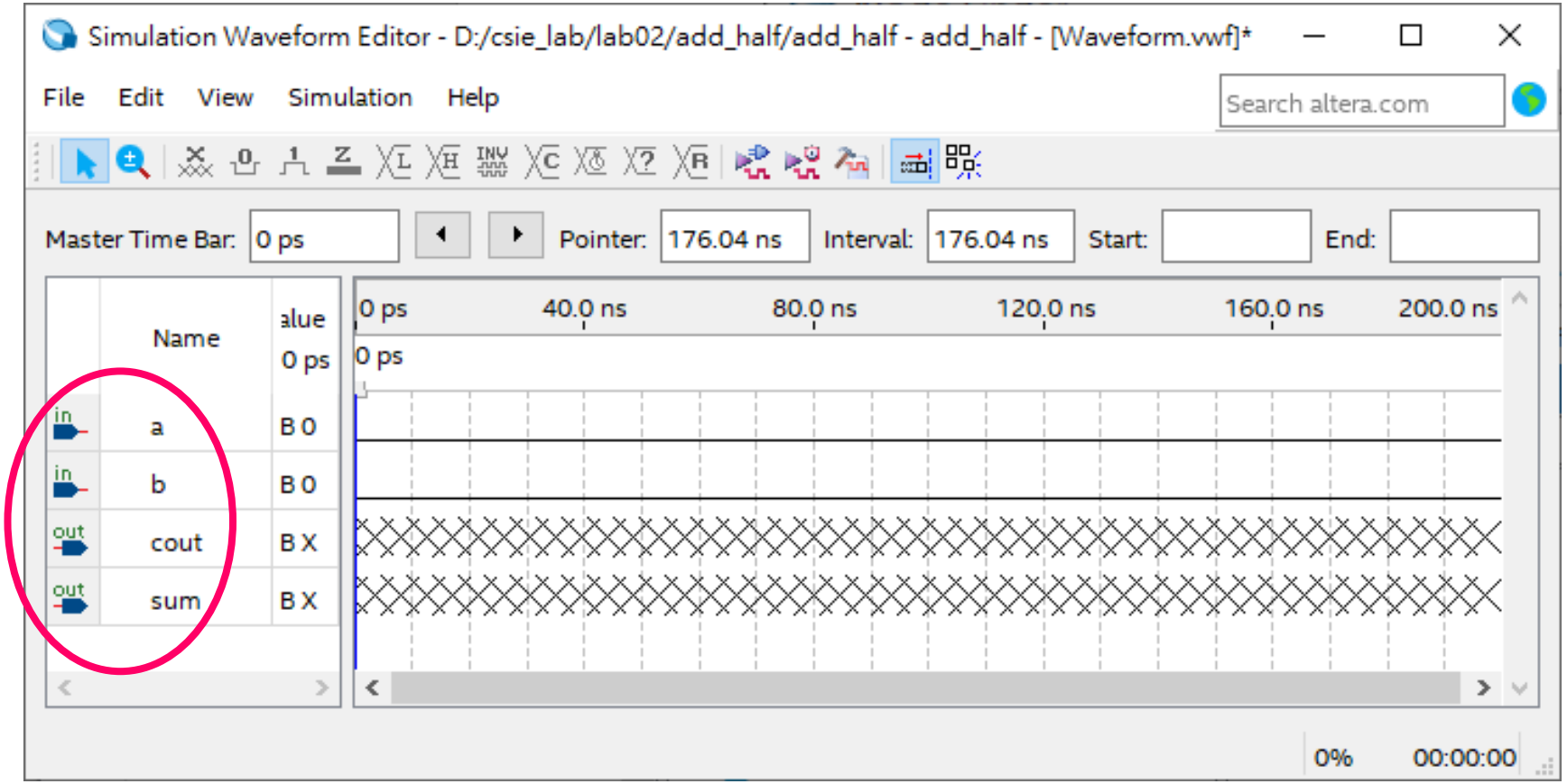




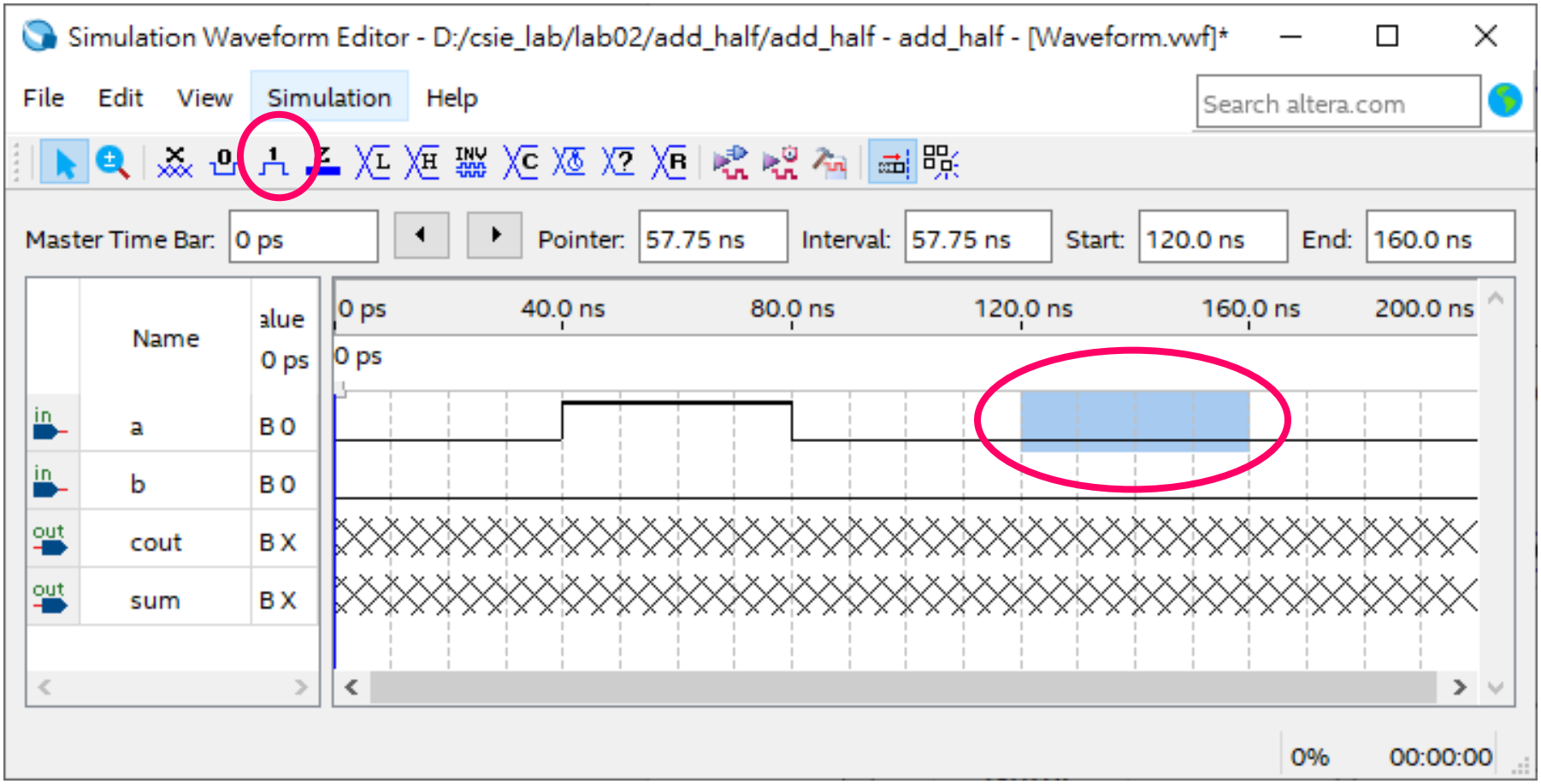
Insert Node for I/O Signals (2/3)



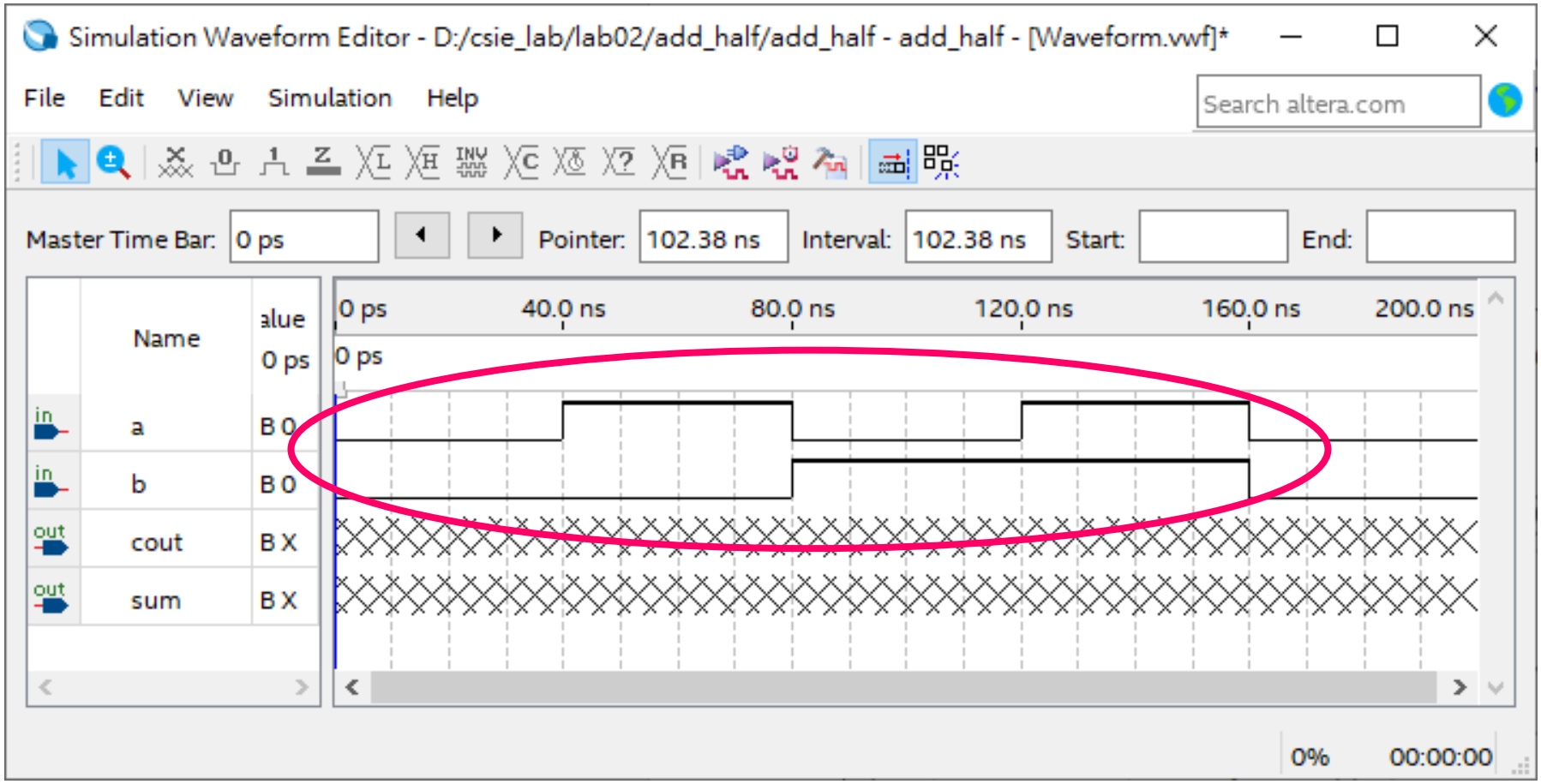
Insert Node for I/O Signals (3/3)



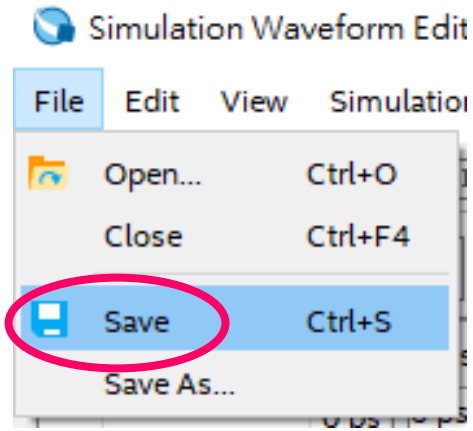
Set Values for Input Signals (1/2)

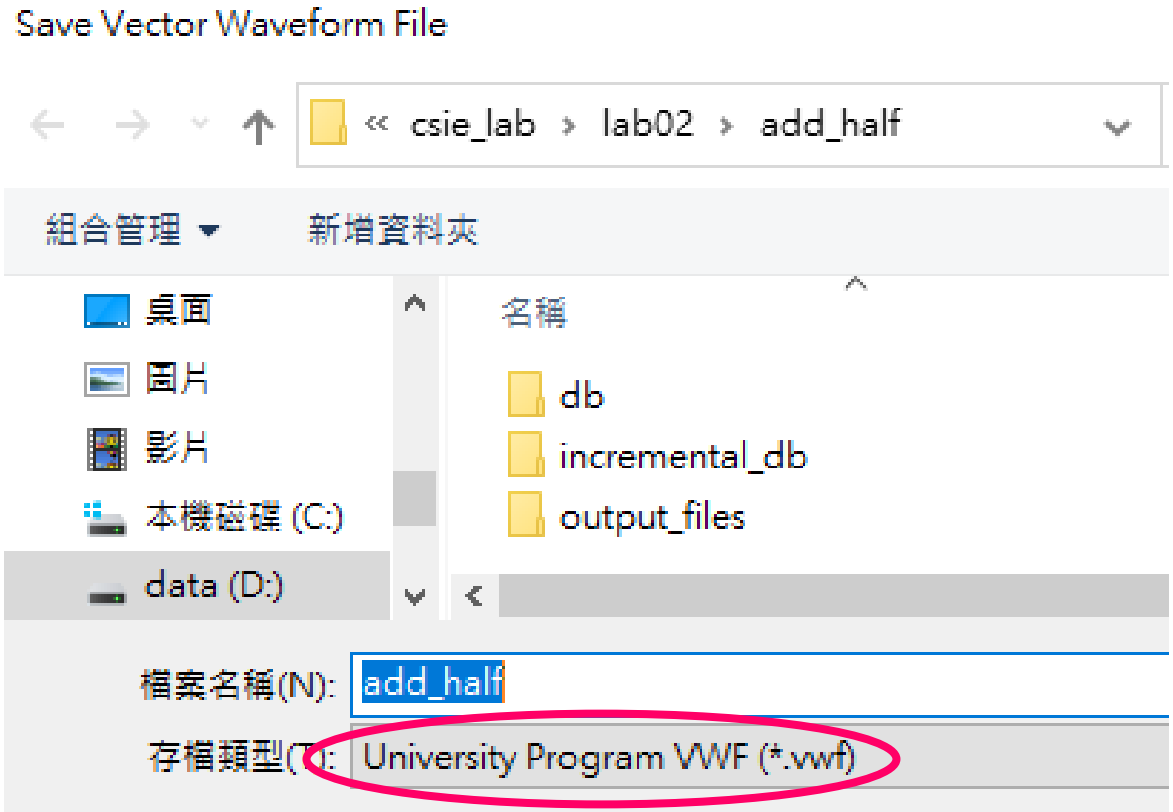


Set Values for Input Signals (2/2)



Save the Waveform File

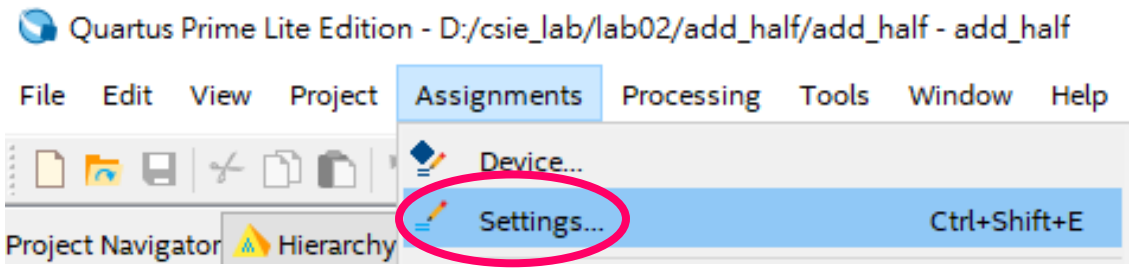


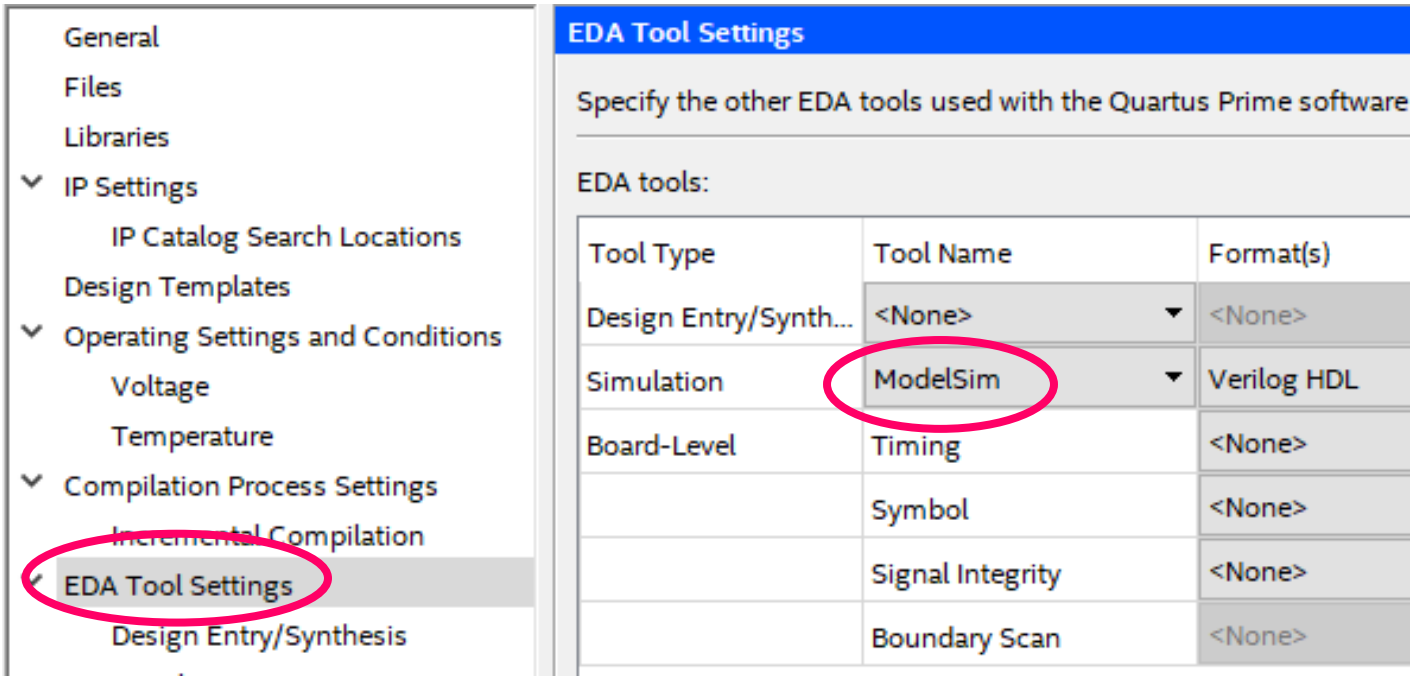


Simulation

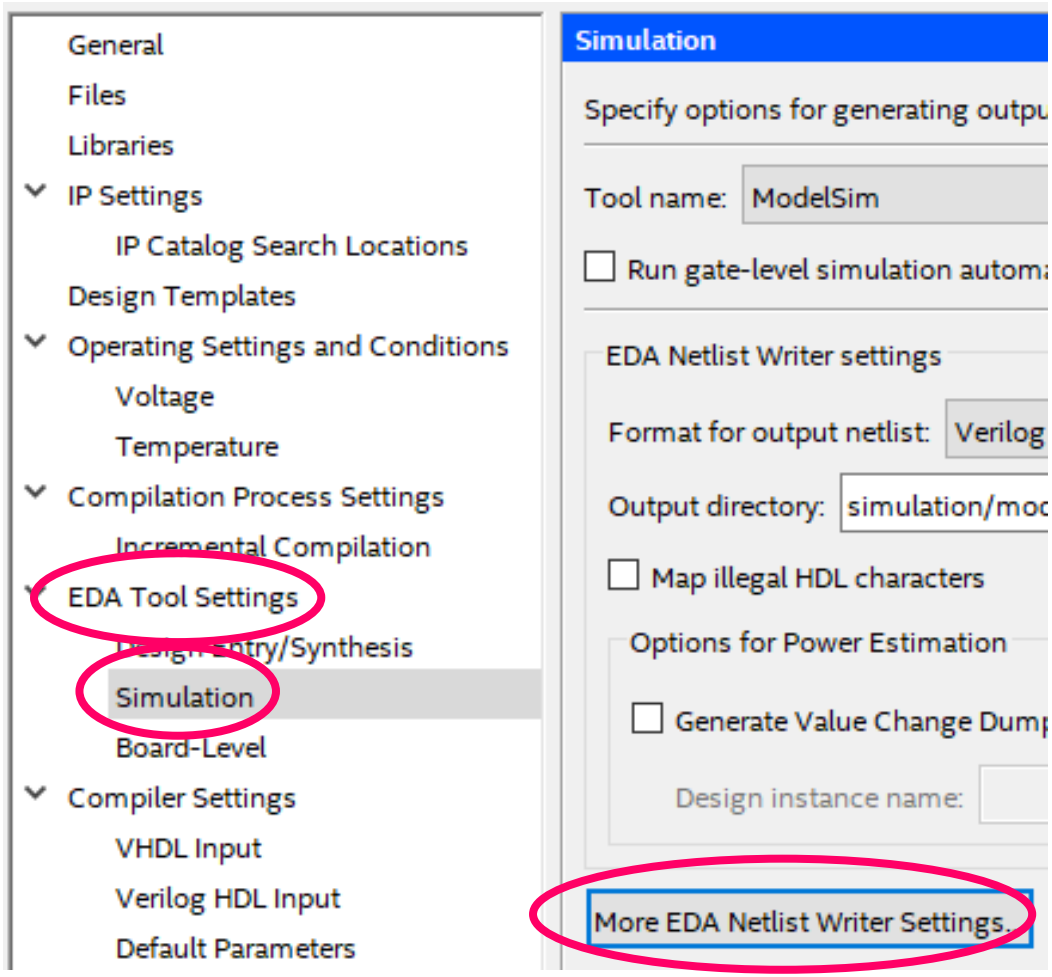
- Functional simulation
  - Logic elements and interconnection wires are perfect, causing no delay in propagation of signals through the circuit.
  - Used to verify the functional correctness of a circuit.
  - Takes much less time.
- Timing simulation
  - Takes all propagation delays into account.

Set Simulation Tool - ModelSim





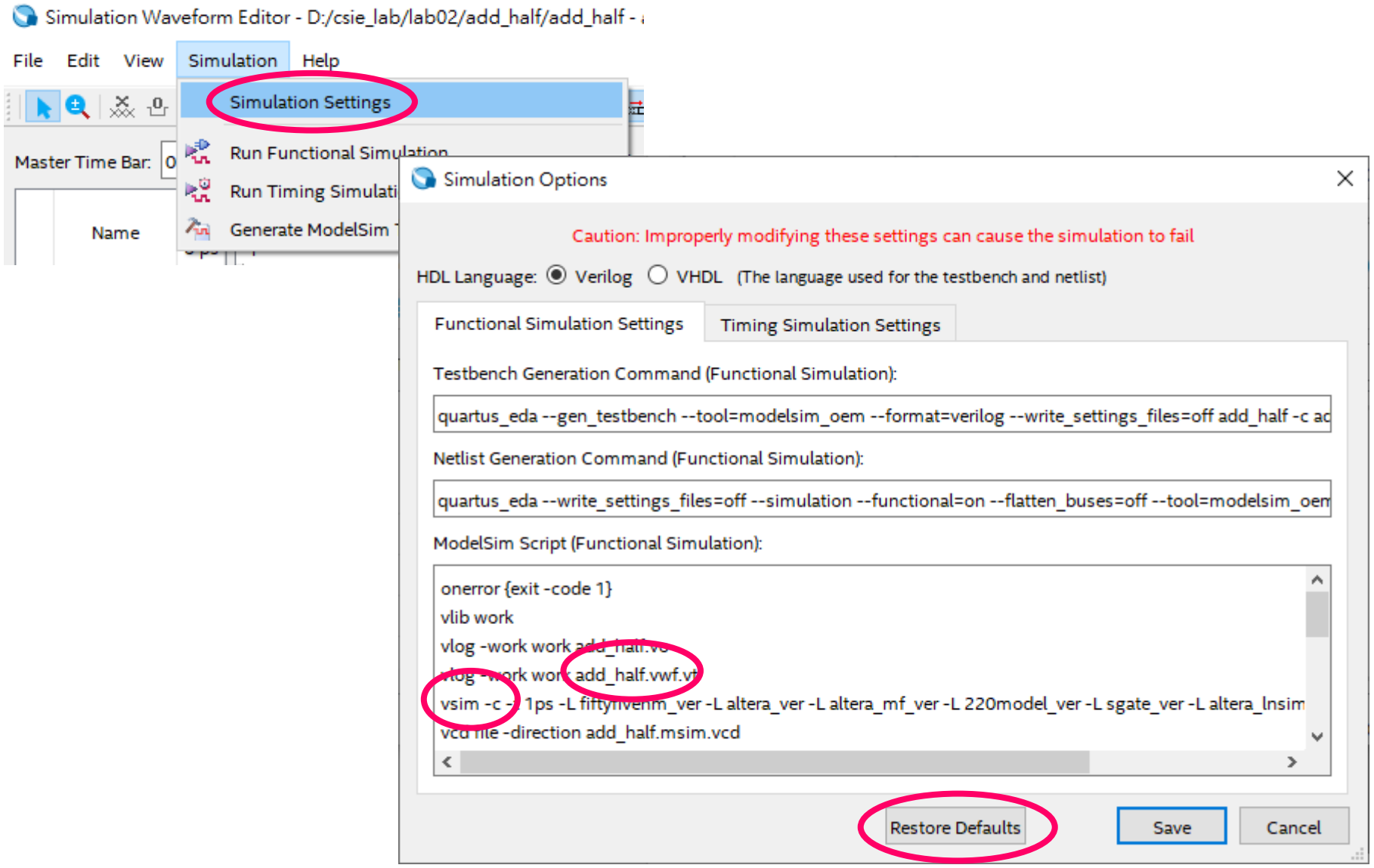
Check Functional Simulation Netlist (1/2)



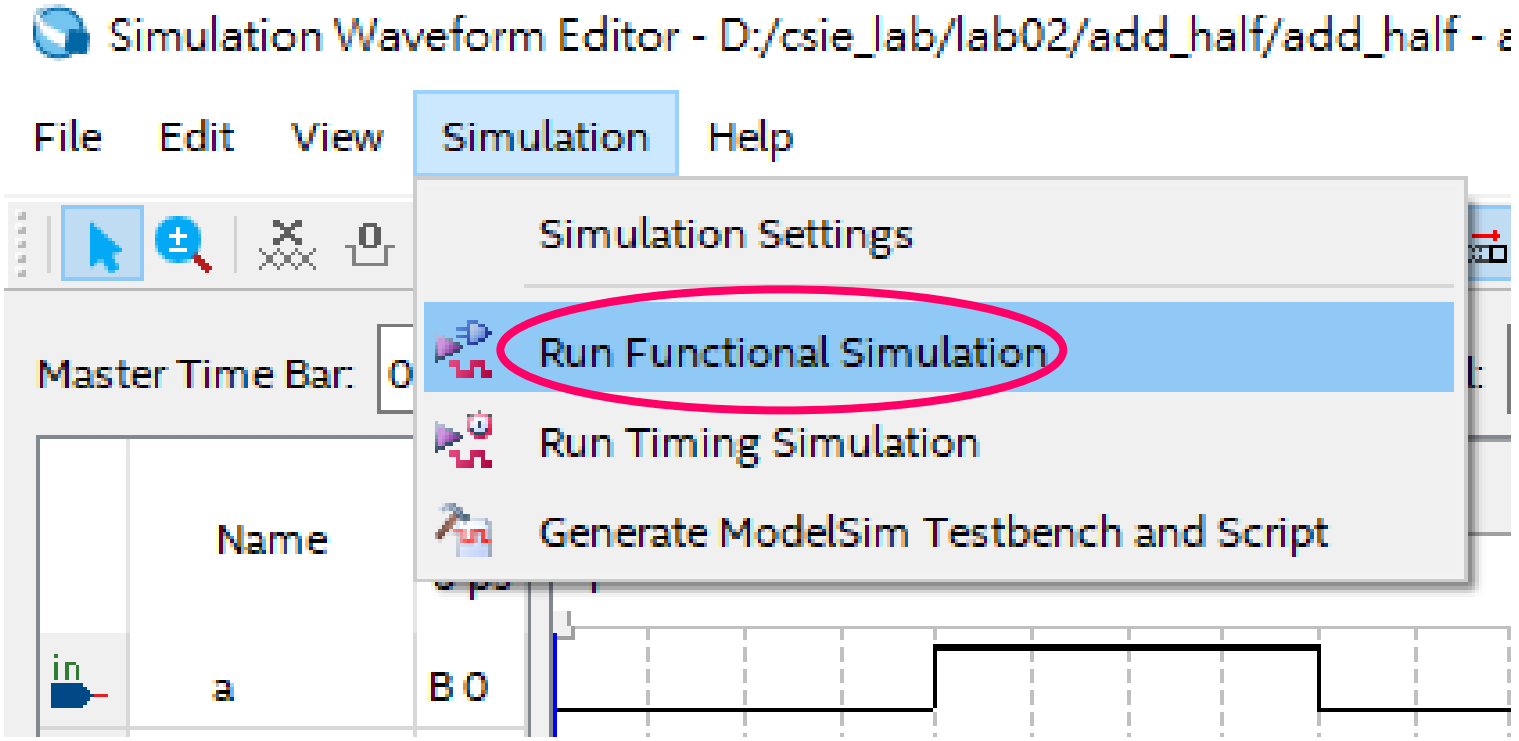
Check Functional Simulation Netlist (2/2)

Name:	Setting:
Architecture name in VHDL output netlist	structure
Bring out device-wide set/reset signals as ports	Off
Disable detection of setup and hold time violations in the input registers of bi-directional pins	Off
Do not write top level VHDL entity	Off
Flatten buses into individual nodes	Off
Generate functional simulation netlist	On
Generate third-party EDA tool command script for RTL functional simulation	Off
Generate third-party EDA tool command script for gate-level simulation	Off
Maintain hierarchy	Off
Truncate long hierarchy paths	Off

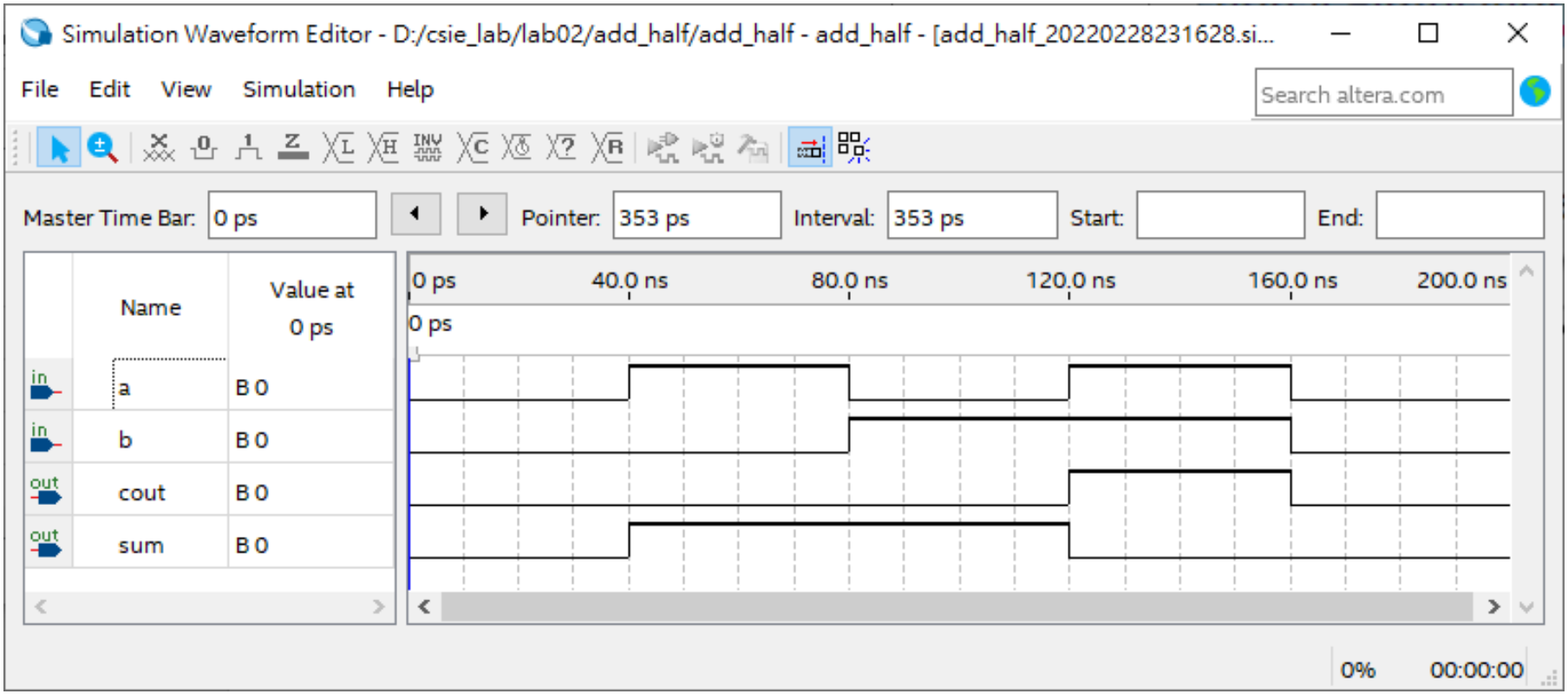
Modify ModelSim Script



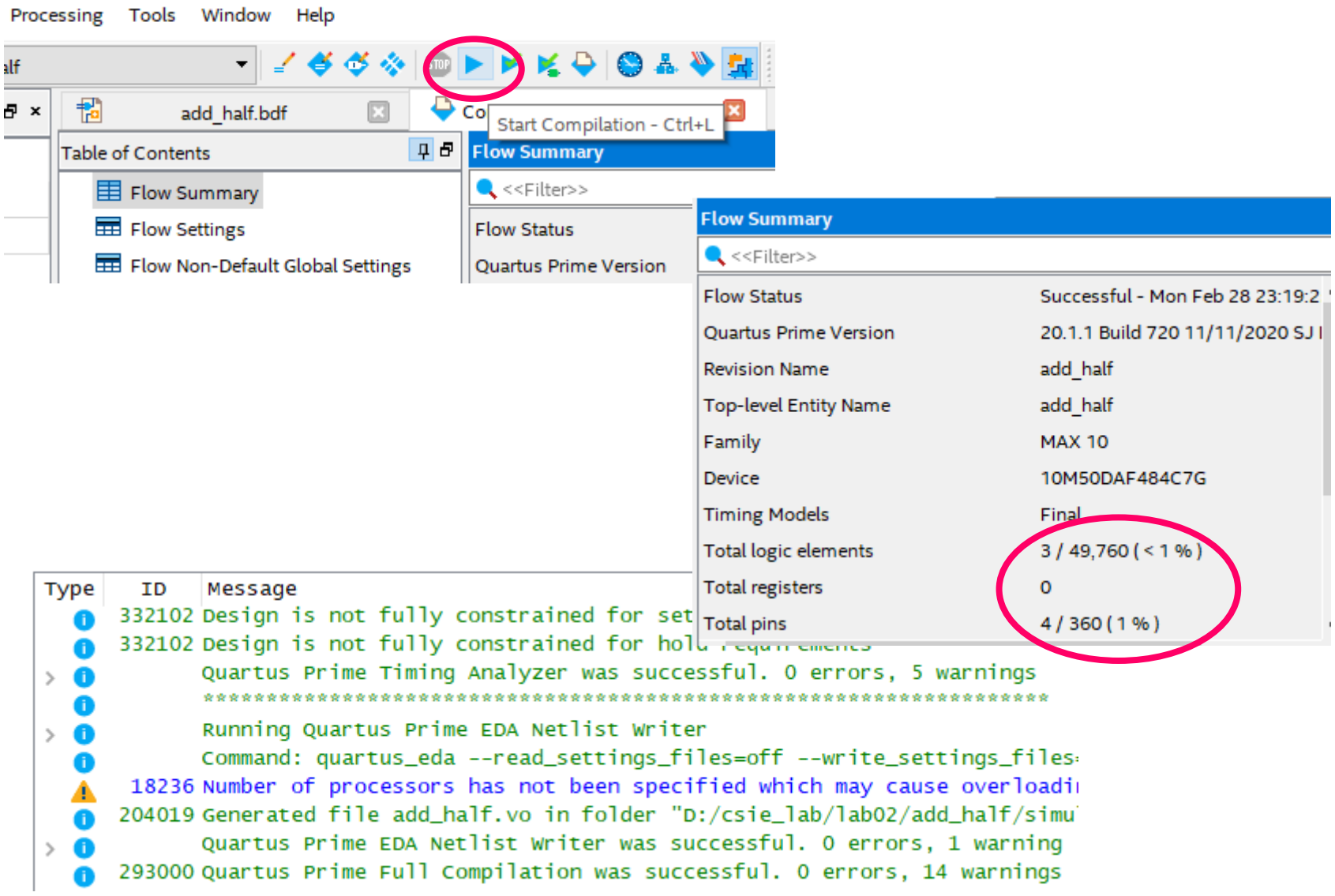
Run Functional Simulation



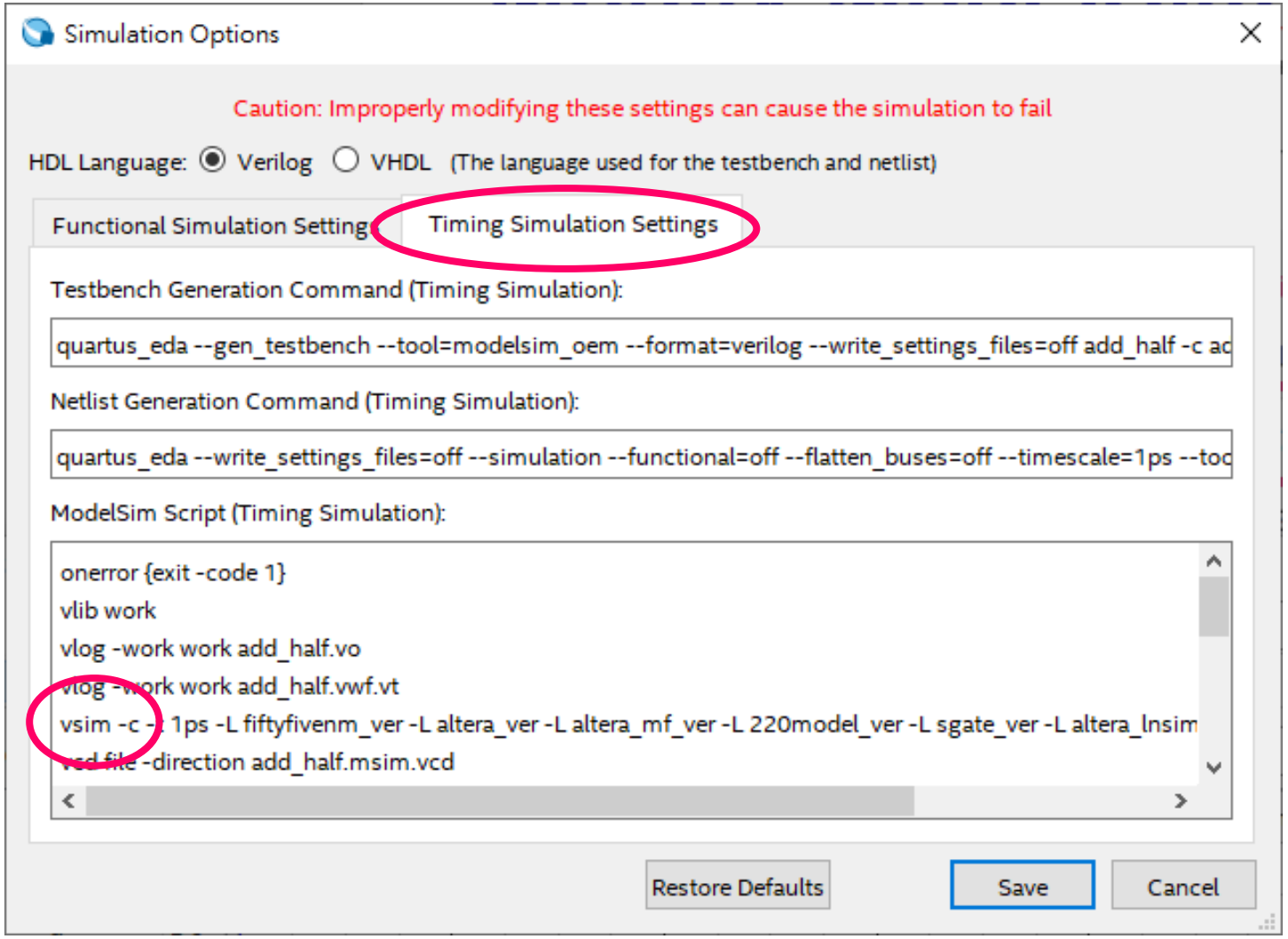
Functional Simulation Result



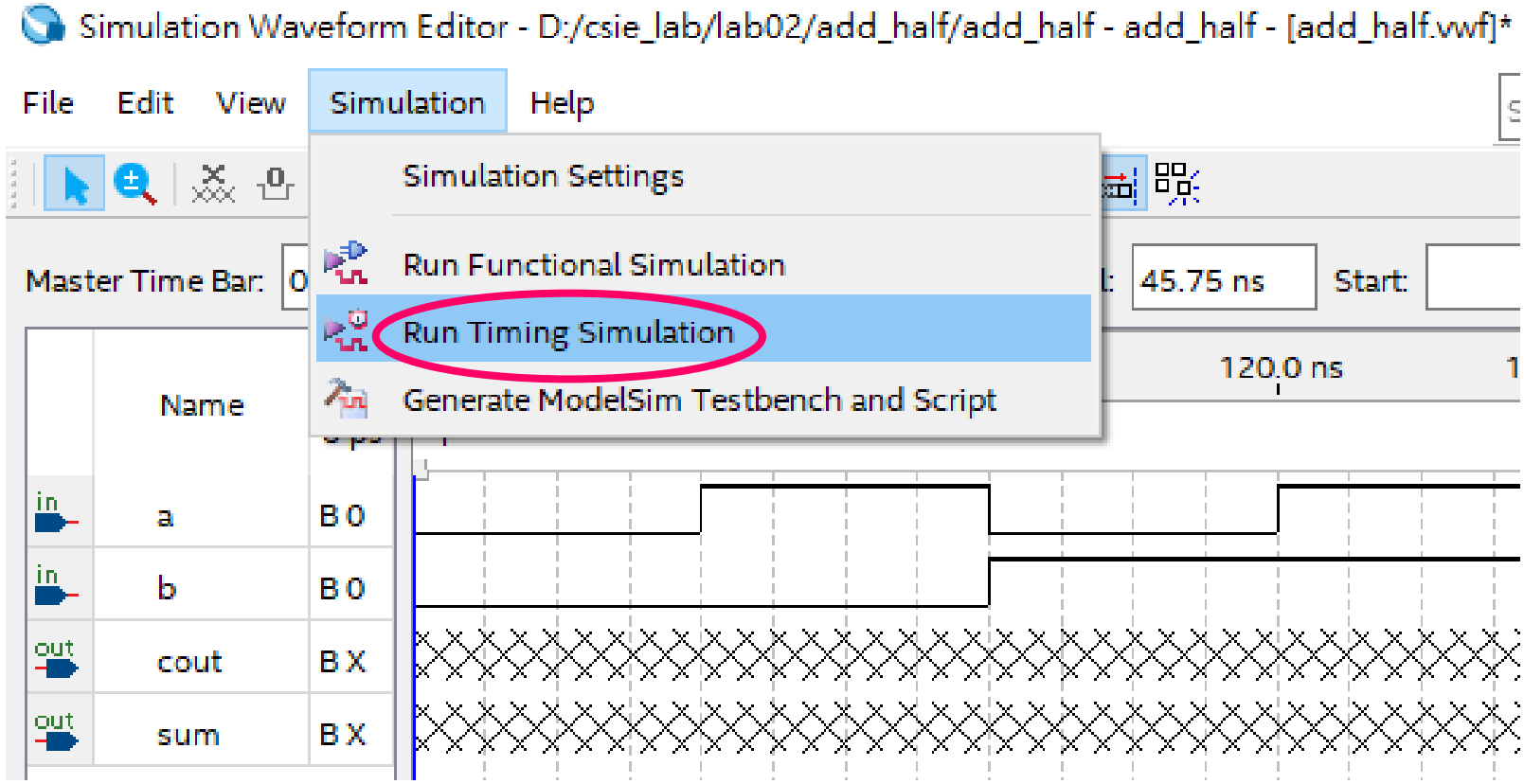
Perform Advanced Compilation for Timing Simulation



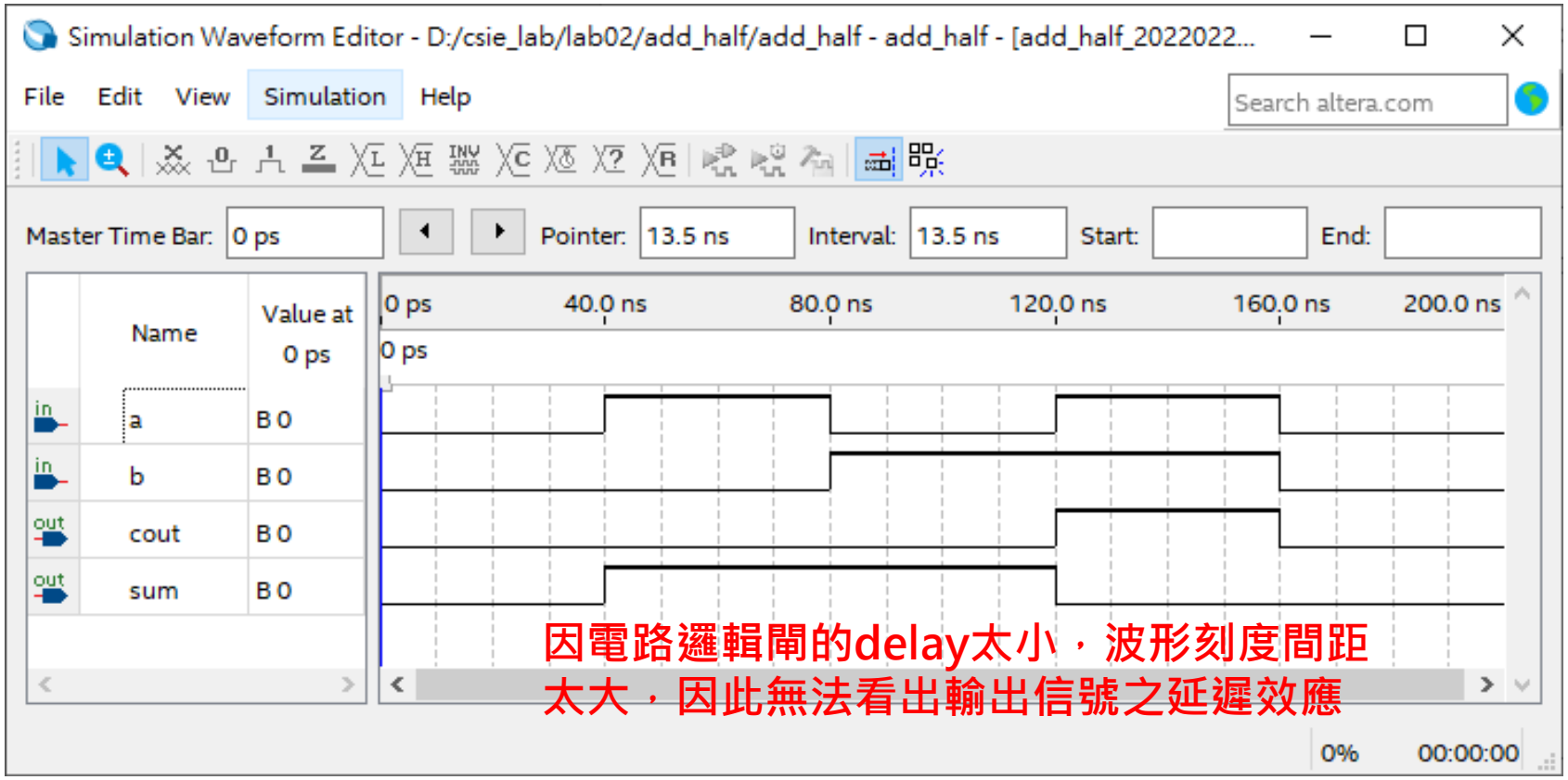
Modify ModelSim Script



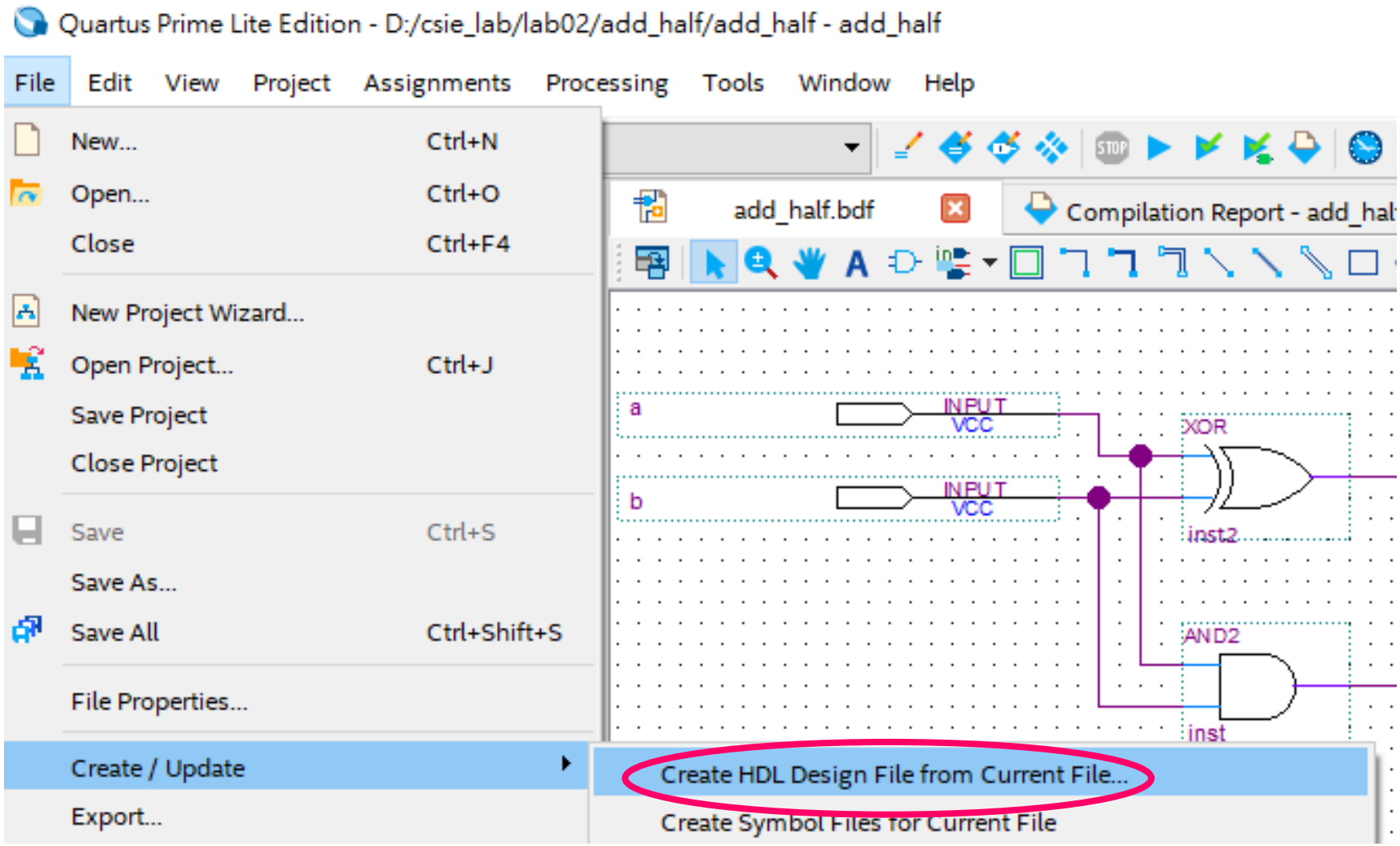
Run Timing Simulation



Timing Simulation Result



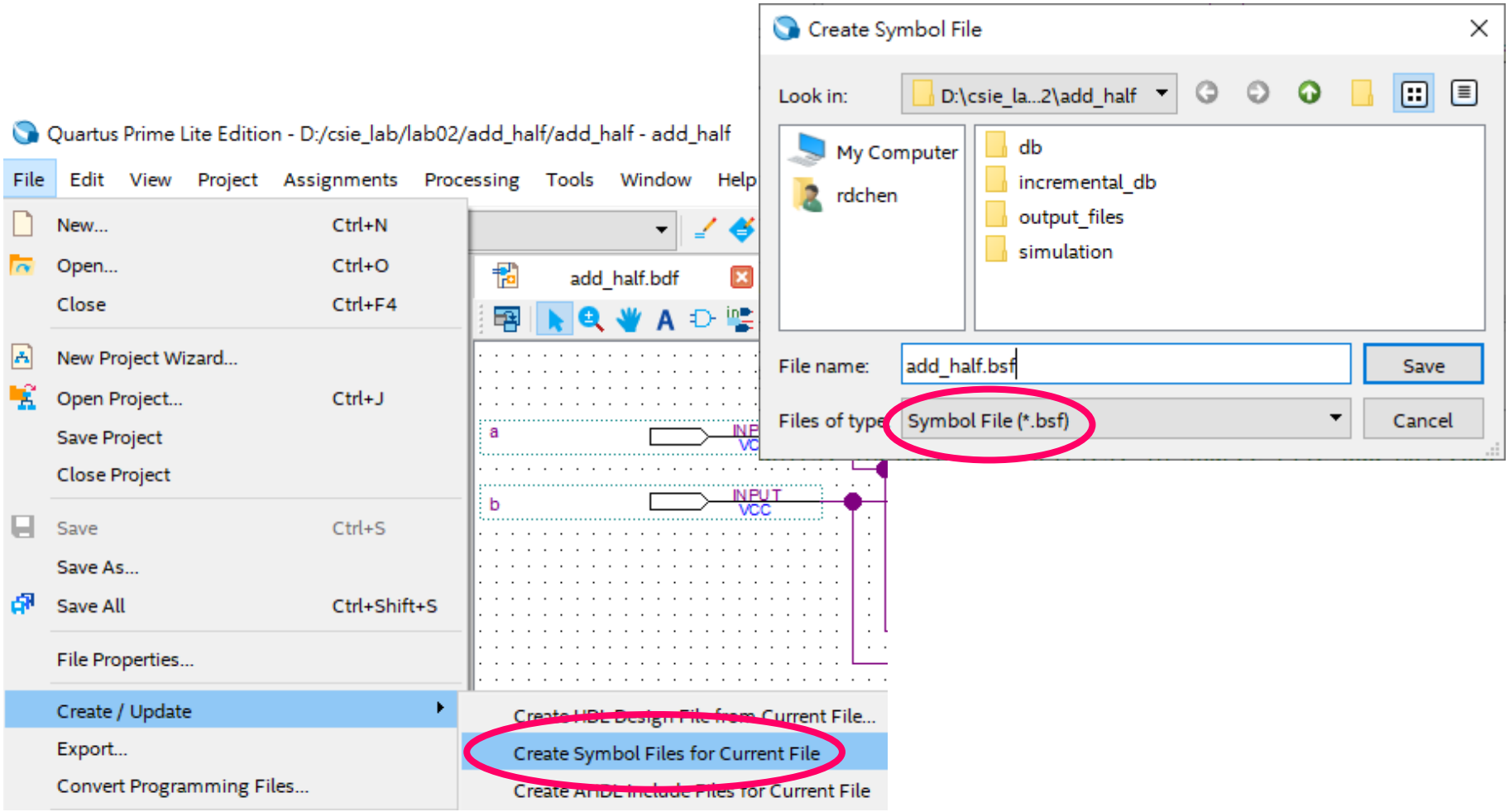
Create the HDL (Verilog) File



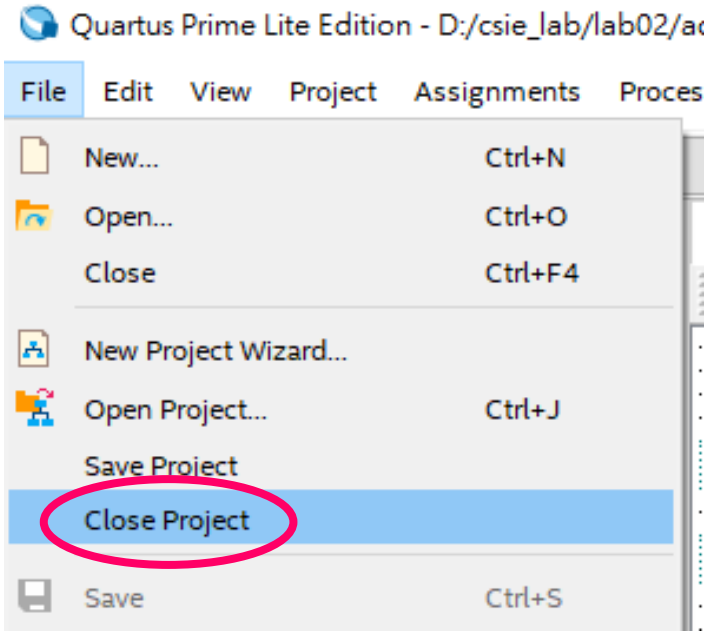
Content in add_half.v (Generated by Tool)

```
module add_half(
    a,
    b,
    cout,
    sum
);

input wire  a;
input wire  b;
output wire cout;
output wire sum;

assign  cout = a & b;
assign  sum = a ^ b;

endmodule
```

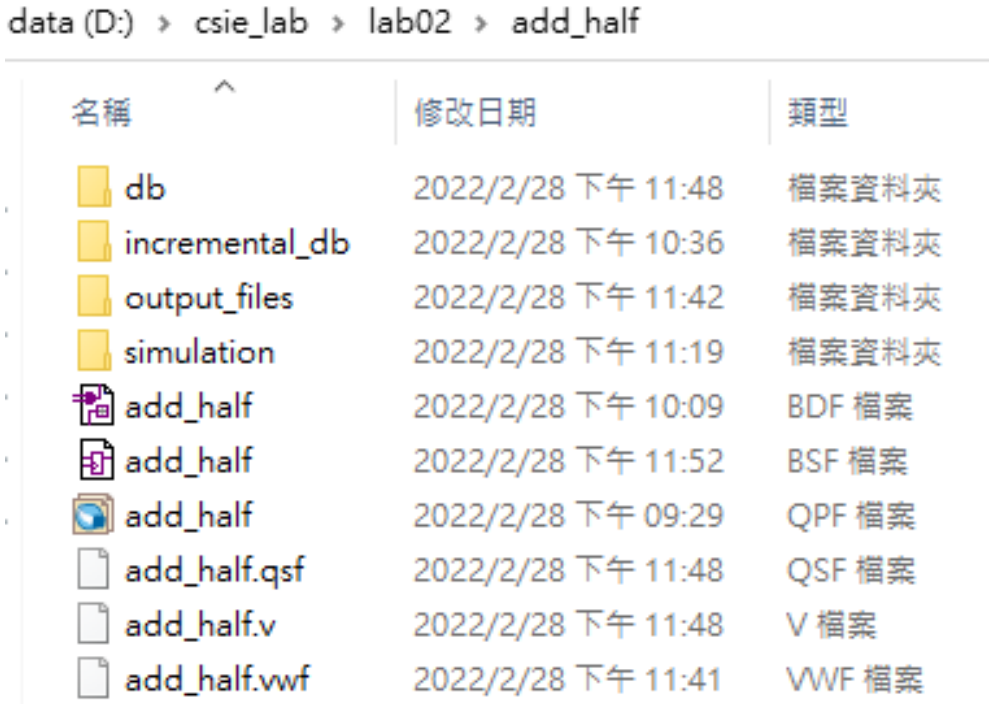
Create the Symbol File



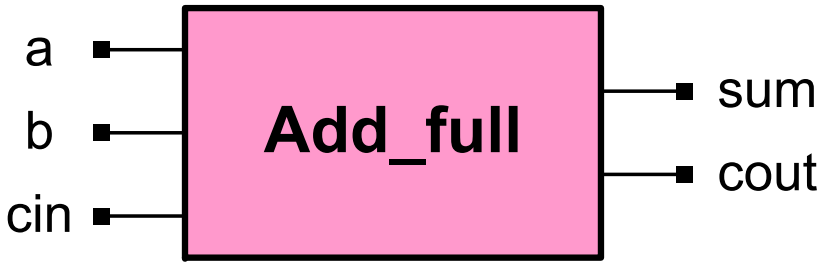
Close the Current Project



All add_half files



Design Example: Full Adder (1/2)



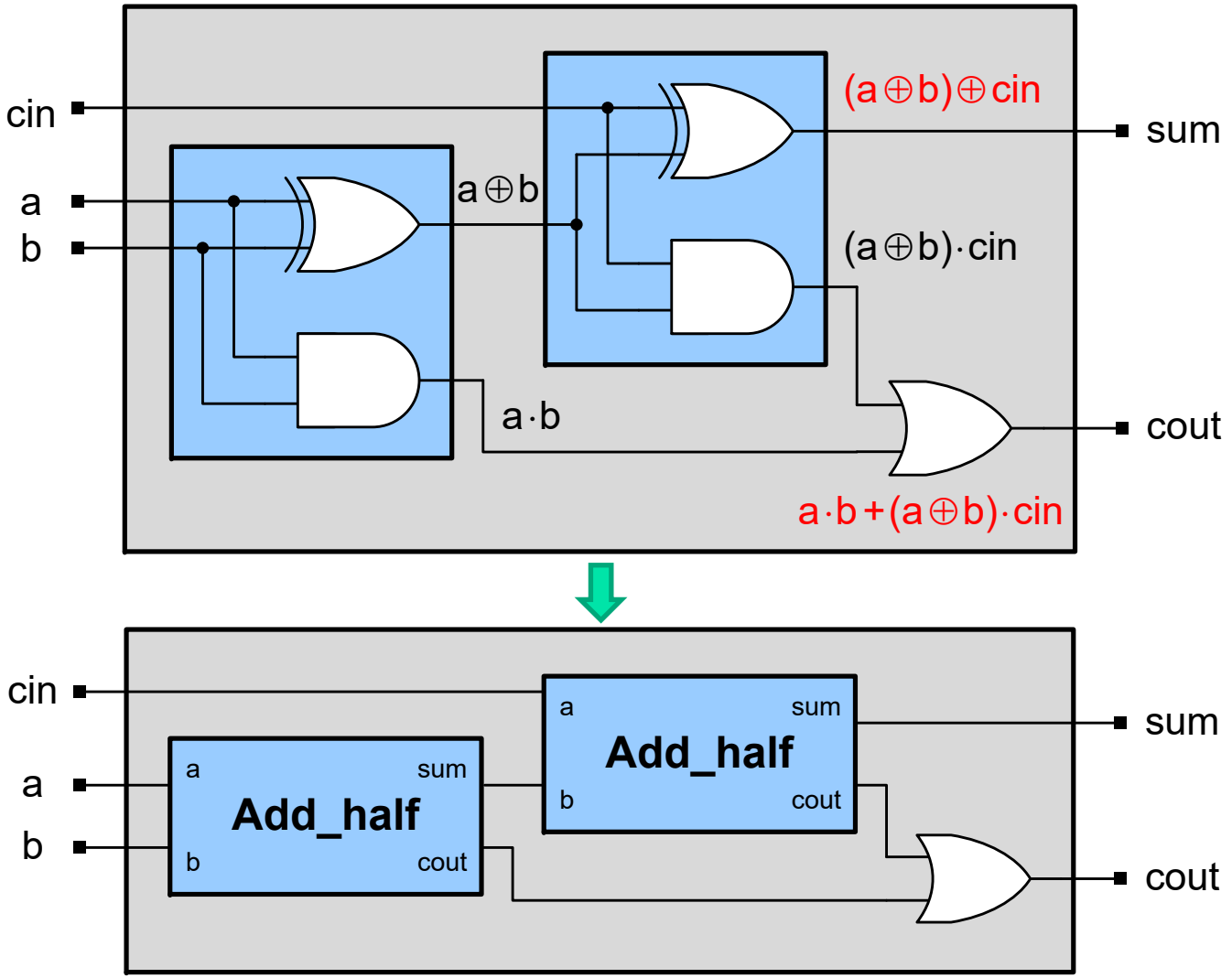
ab\cin	0	1
00	0	1
01	1	0
11	0	1
10	1	0

$sum = (a \oplus b) \oplus cin$

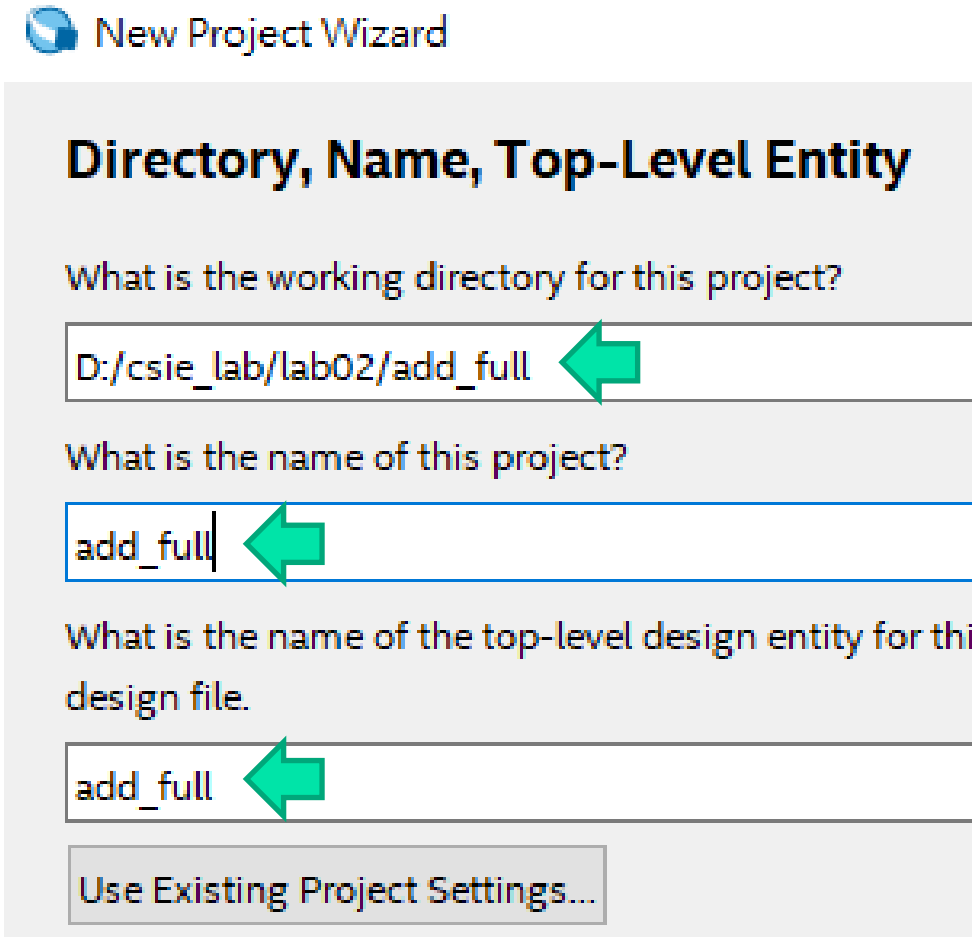
ab\cin	0	1
00	0	0
01	0	1
11	1	1
10	0	1

$cout = a \cdot b + (a \oplus b) \cdot cin$

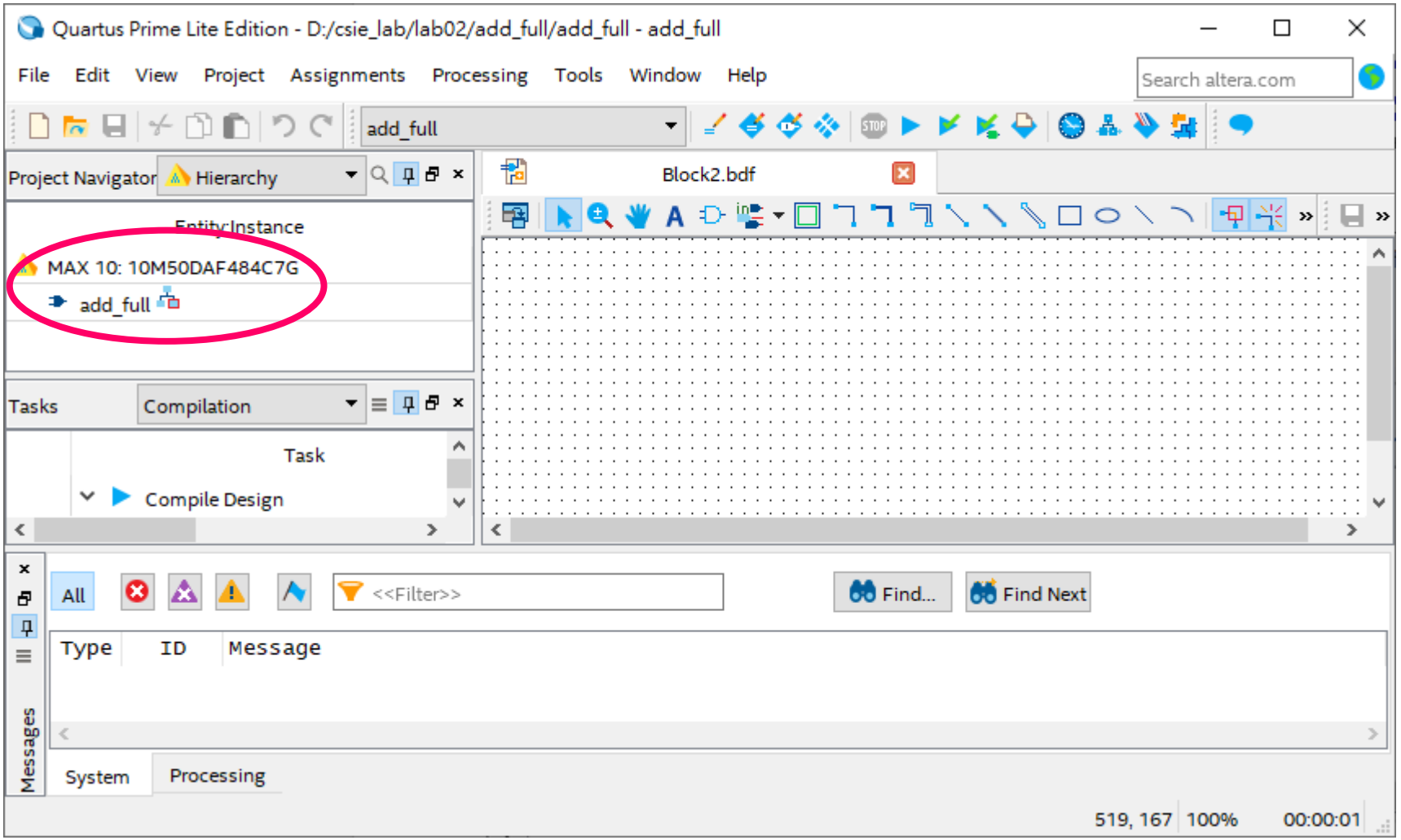
Design Example: Full Adder (2/2)



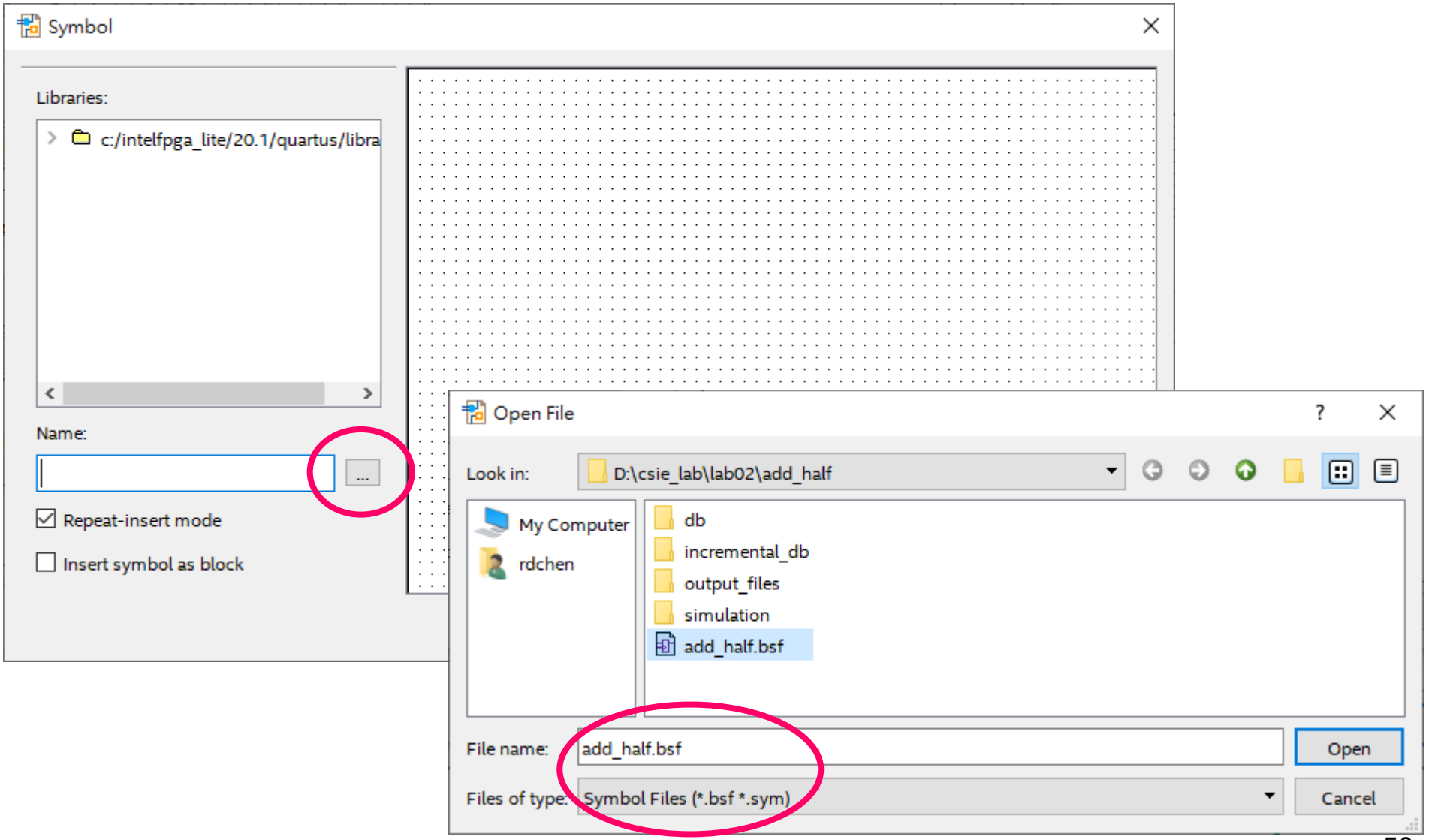
Create a New Project - add_full



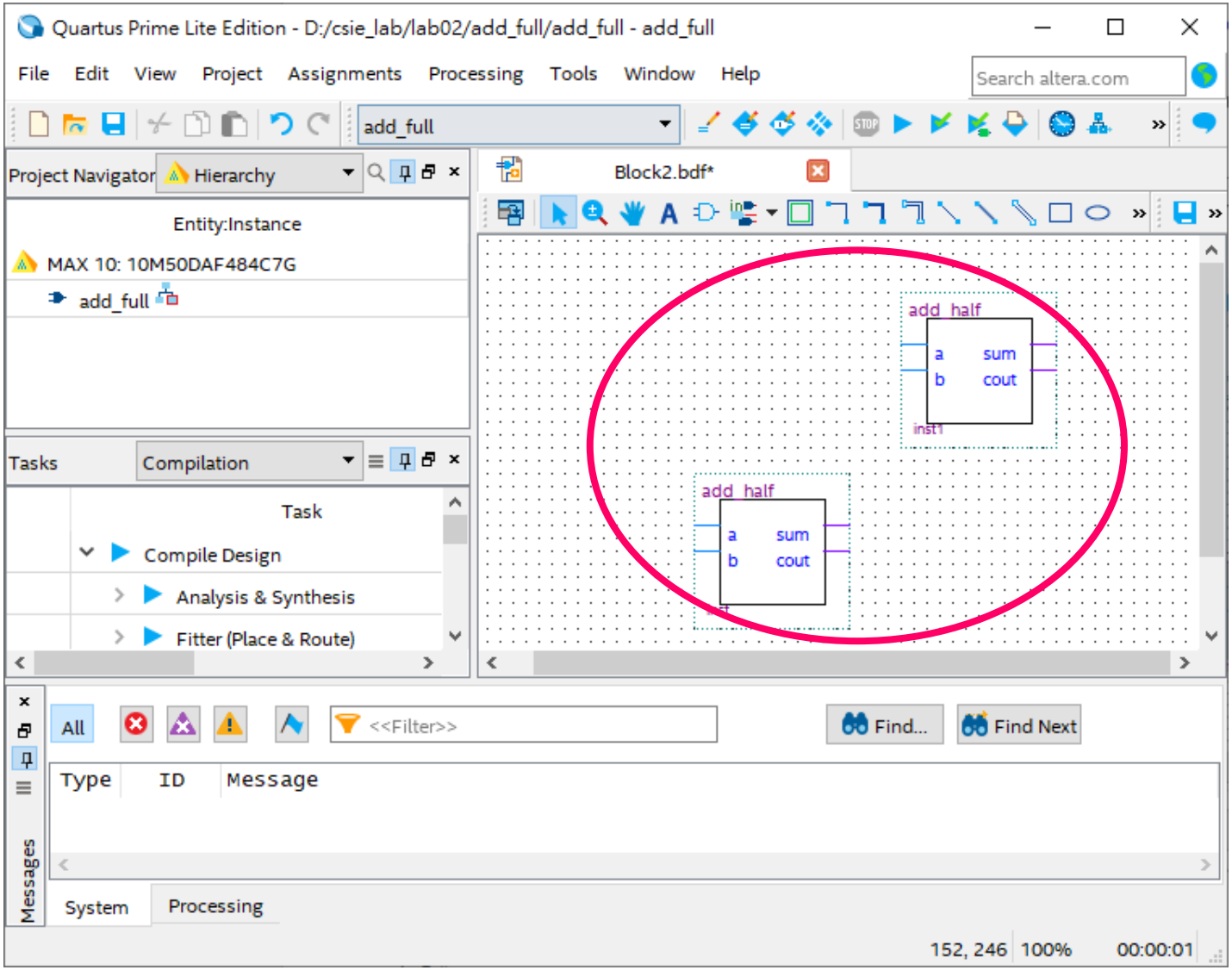
Create a Schematic Design for add_full



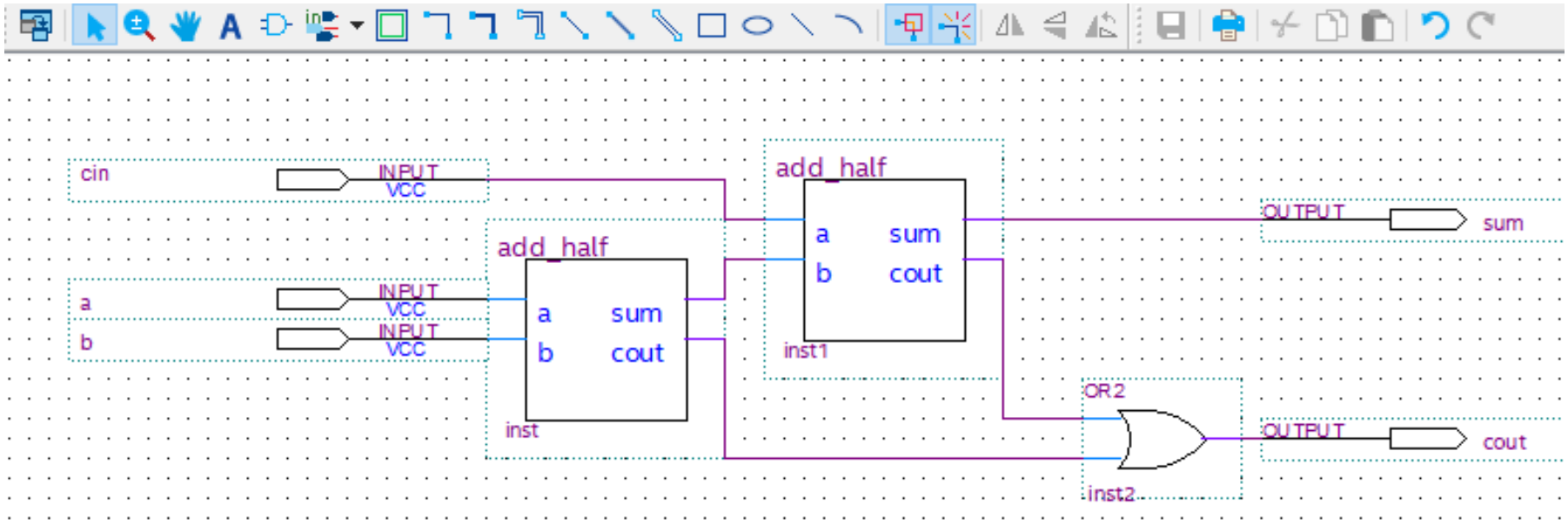
Select and Import a Half Adder



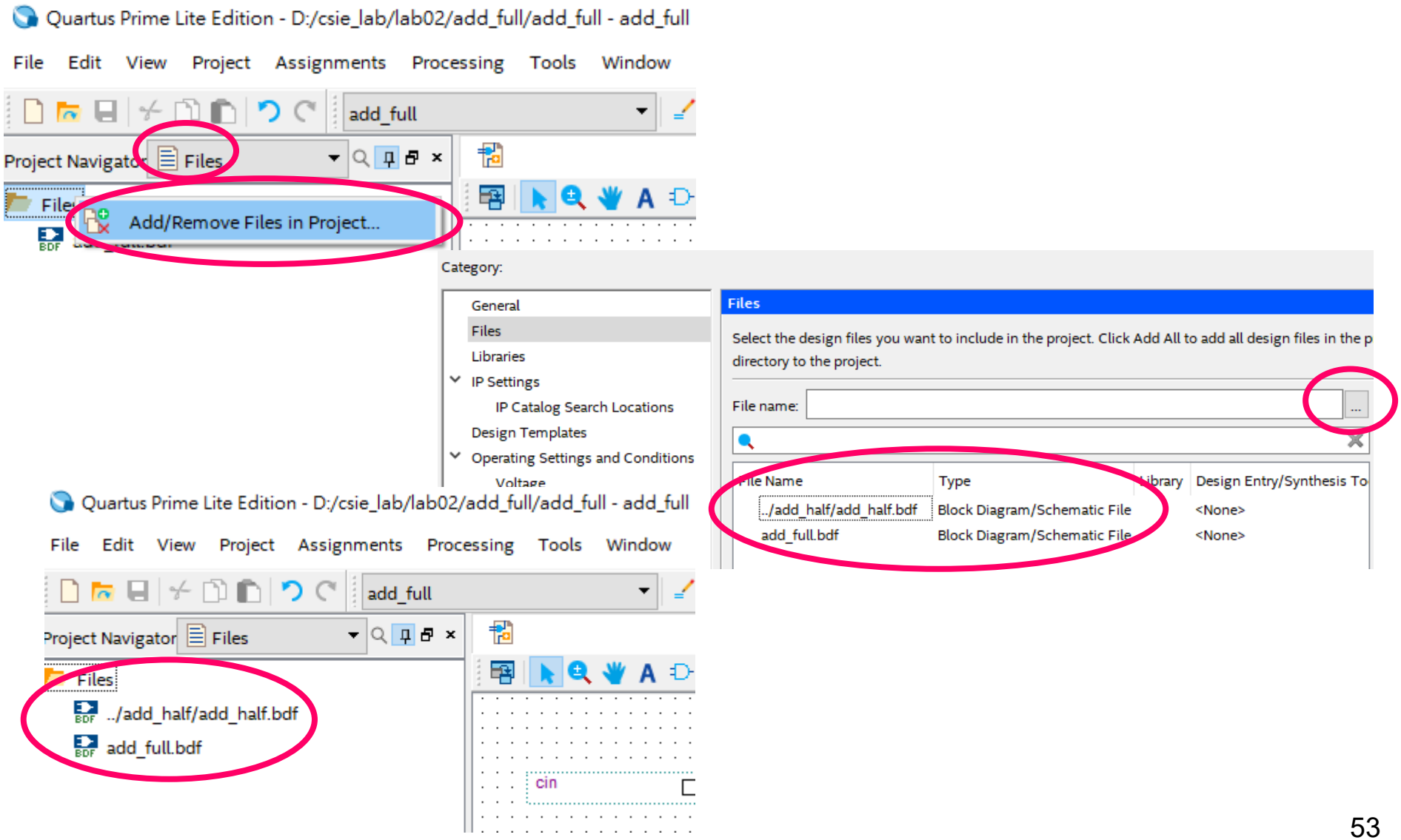
Two Half Adders are Imported



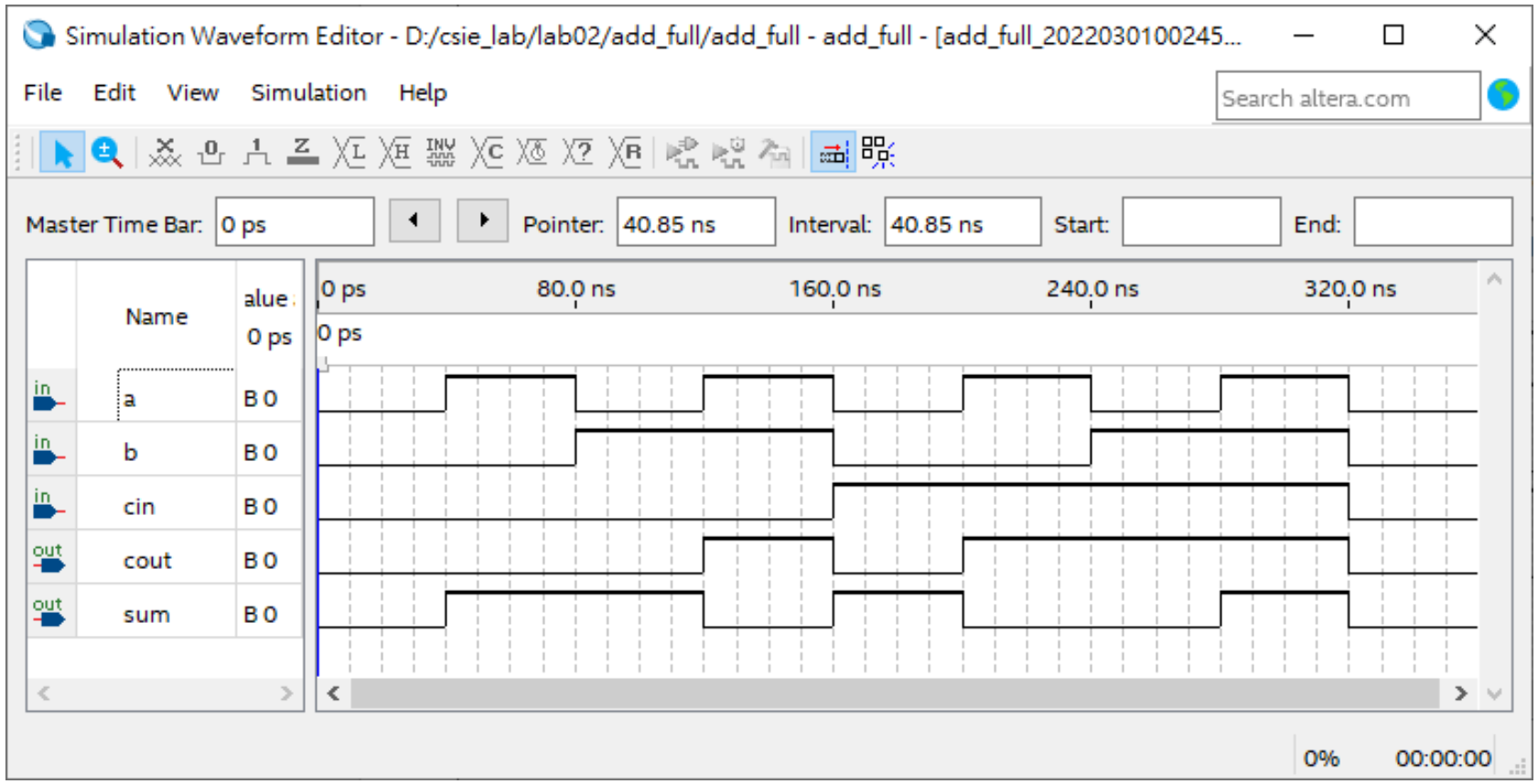
Import OR2 Gate and I/O Pins



Add the add_half Schematic File



Functional Simulation Result



Content in add_full.v (Generated by Tool)

```
module add_full(
    a,
    b,
    cin,
    cout,
    sum
);

input wire  a;
input wire  b;
input wire  cin;
output wire cout;
output wire sum;

wire    SYNTHESIZED_WIRE_0;
wire    SYNTHESIZED_WIRE_1;
wire    SYNTHESIZED_WIRE_2;

add_half    b2v_inst(
    .a(a),
    .b(b),
    .cout(SYNTHESIZED_WIRE_2),
    .sum(SYNTHESIZED_WIRE_0));

add_half    b2v_inst1(
    .a(SYNTHESIZED_WIRE_0),
    .b(cin),
    .cout(SYNTHESIZED_WIRE_1),
    .sum(sum));

assign  cout = SYNTHESIZED_WIRE_1 | SYNTHESIZED_WIRE_2;

endmodule
```

Pin Assignments

Inputs

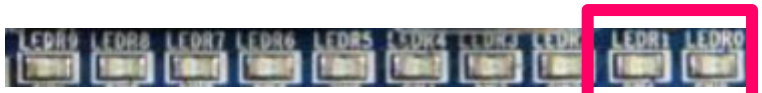
SW2 SW1 SW0

1
0

a b cin

Outputs

LED1 LED0

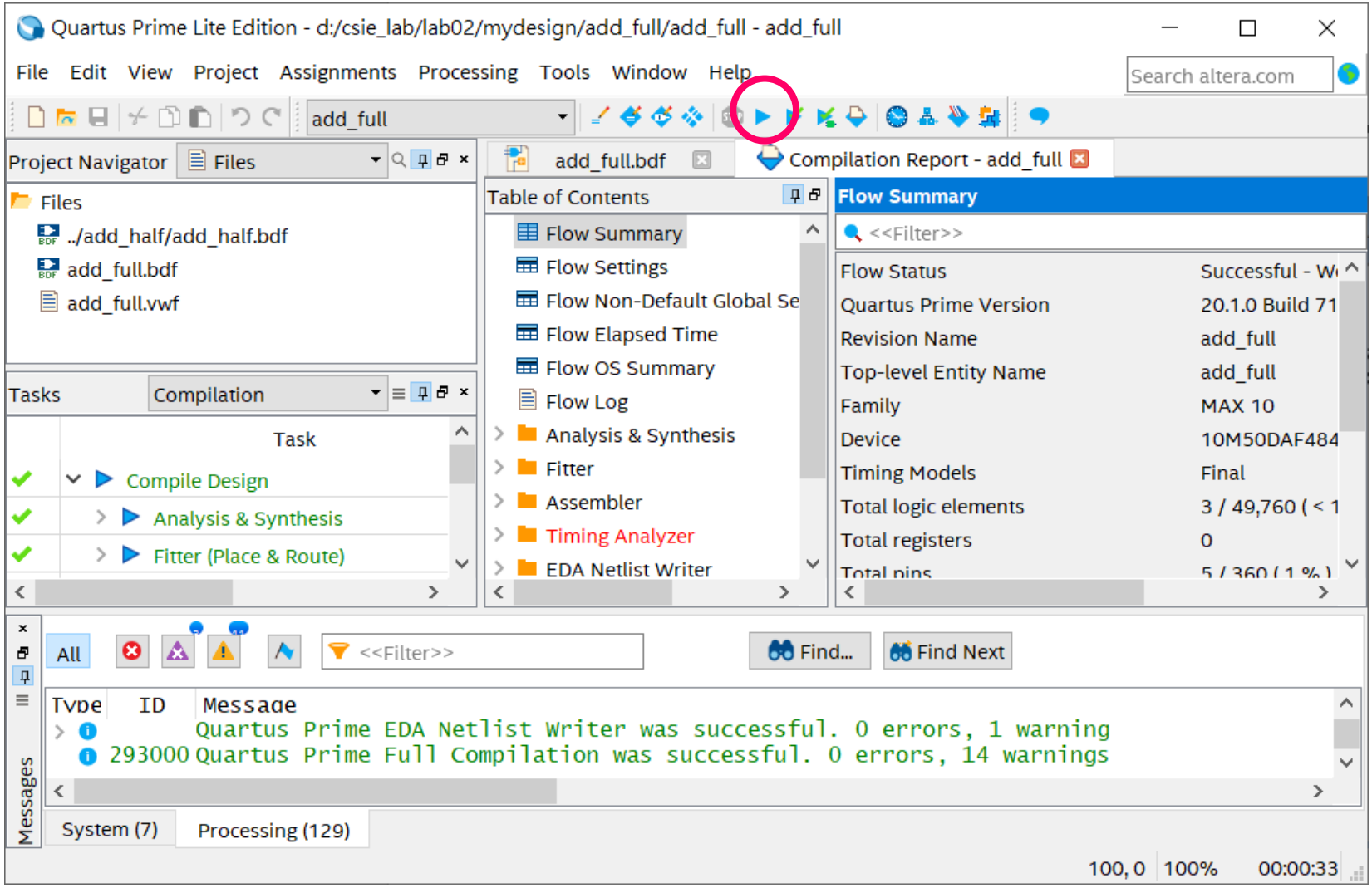


1: LED on
0: LED off

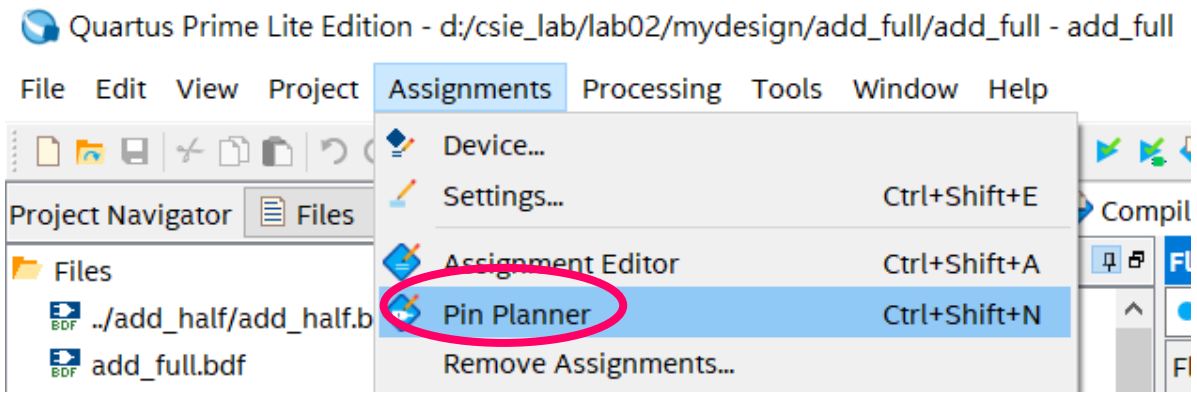
cout sum

{cout, sum} = a + b + cin

Start Compilation

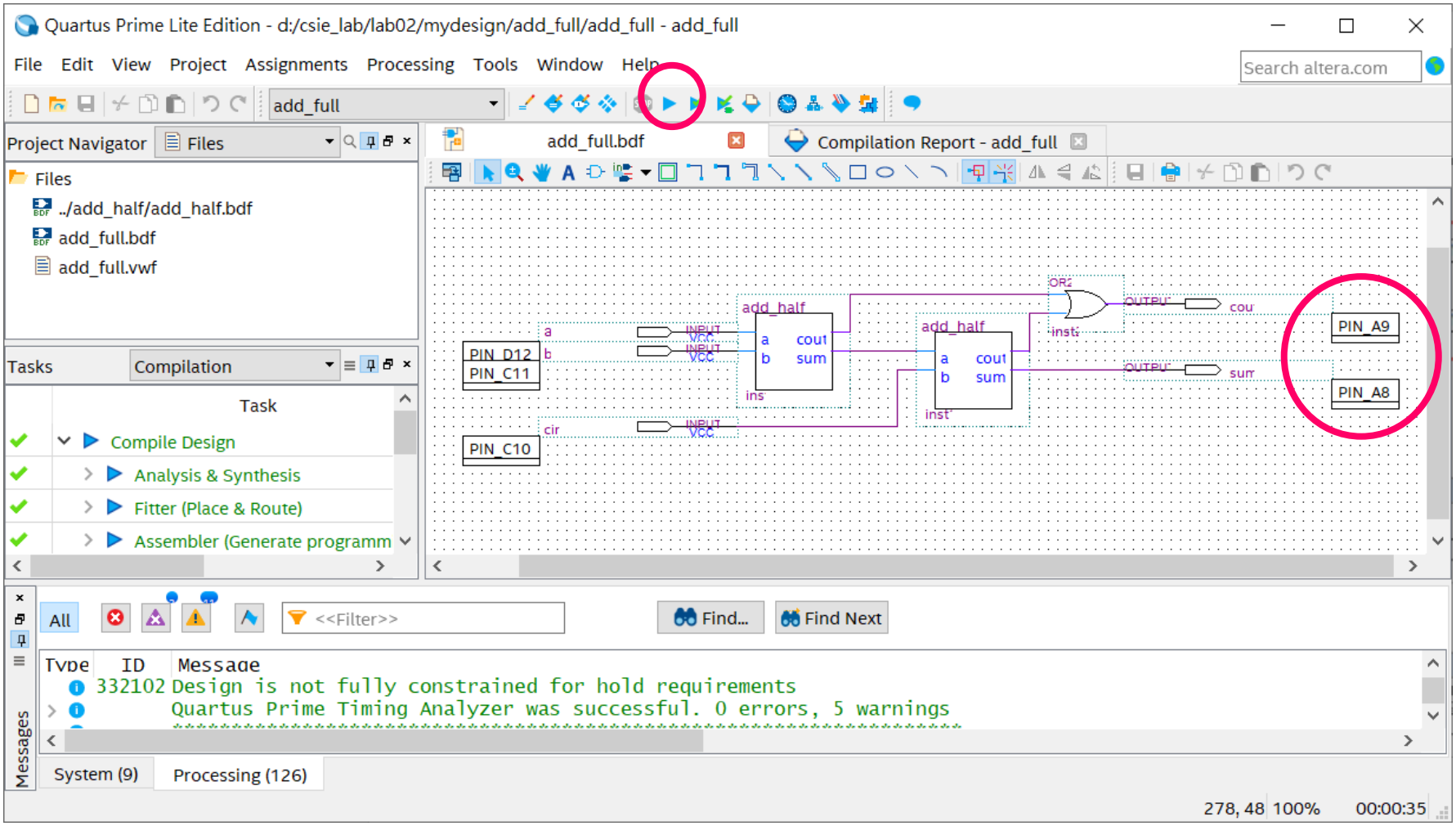


Make Pin Assignments

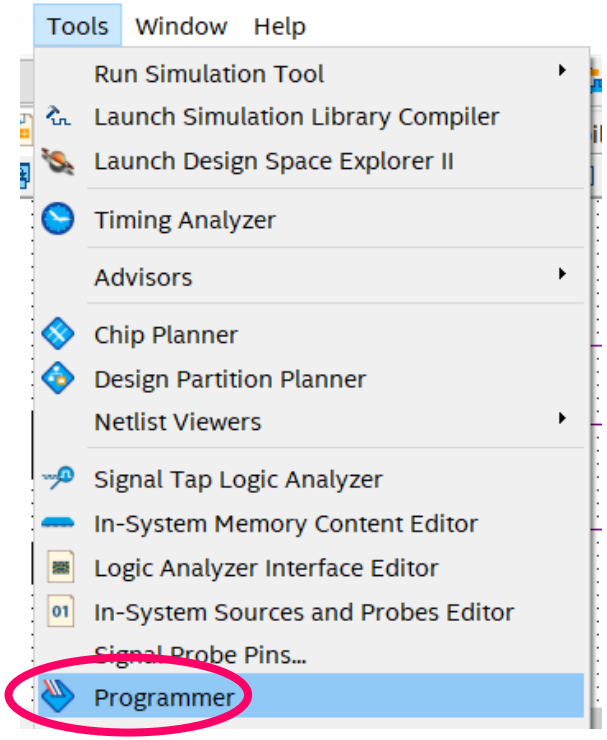


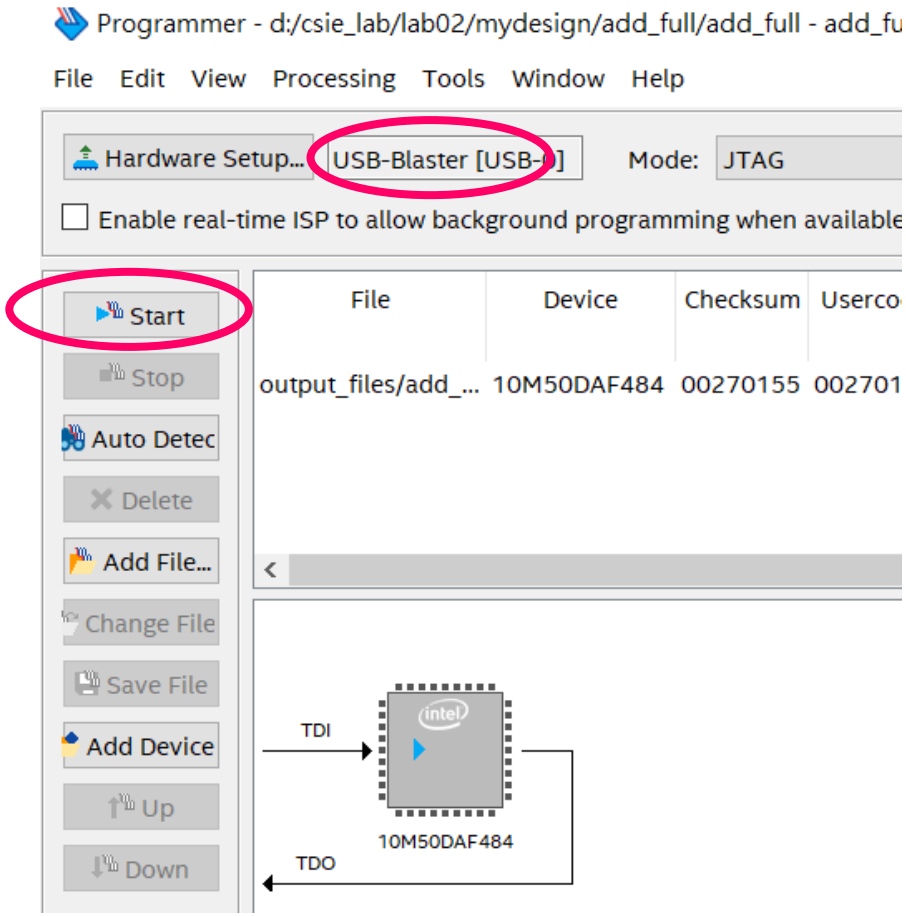
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location
a	Input	PIN_D12	7	B7_N0	PIN_Y5
b	Input	PIN_C11	7	B7_N0	PIN_AA2
cin	Input	PIN_C10	7	B7_N0	PIN_AA1
cout	Output	PIN_A9	7	B7_N0	PIN_W3
sum	Output	PIN_A8	7	B7_N0	PIN_U6

Start Compilation



Program into Device





Verify Your Design

- sw2/sw1/sw0 : dn/dn/dn (0/0/0) => L1/L0: off/off (0/0)
- sw2/sw1/sw0 : dn/dn/up (0/0/1) => L1/L0: off/on (0/1)
- sw2/sw1/sw0 : dn/up/dn (0/1/0) => L1/L0: off/on (0/1)
- sw2/sw1/sw0 : dn/up/up (0/1/1) => L1/L0: on/off (1/0)
- sw2/sw1/sw0 : up/dn/dn (1/0/0) => L1/L0: off/on (0/1)
- sw2/sw1/sw0 : up/dn/up (1/0/1) => L1/L0: on/off (1/0)
- sw2/sw1/sw0 : up/up/dn (1/1/0) => L1/L0: on/off (1/0)
- sw2/sw1/sw0 : up/up/up (1/1/1) => L1/L0: on/on (1/1)

實驗報告

- 請老師or助教驗證**add_full**電路之模擬波形及實驗板之行為是否正確