# ModelSim Intel FPGA Starter Edition

*KAIST, SChool of EE*

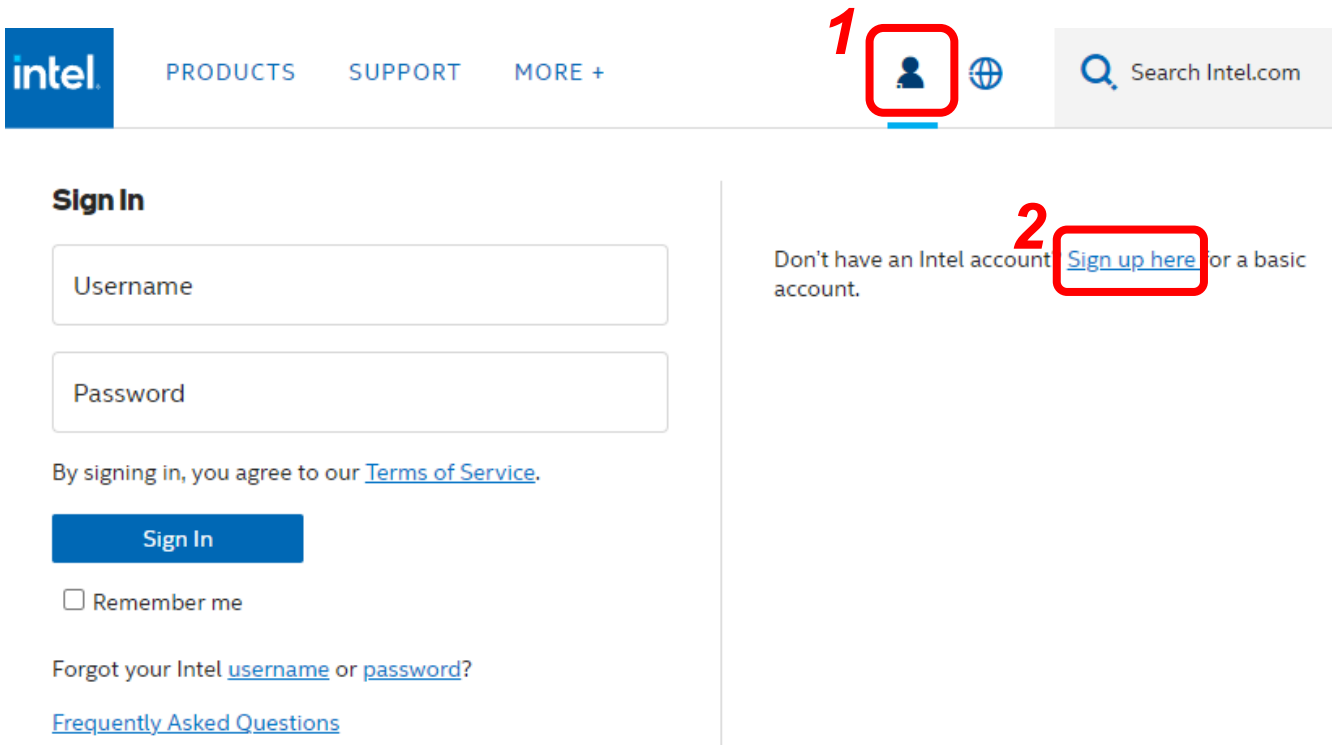
Computer Architecture and Memory systems Laboratory

KAIST EE

CAMELab

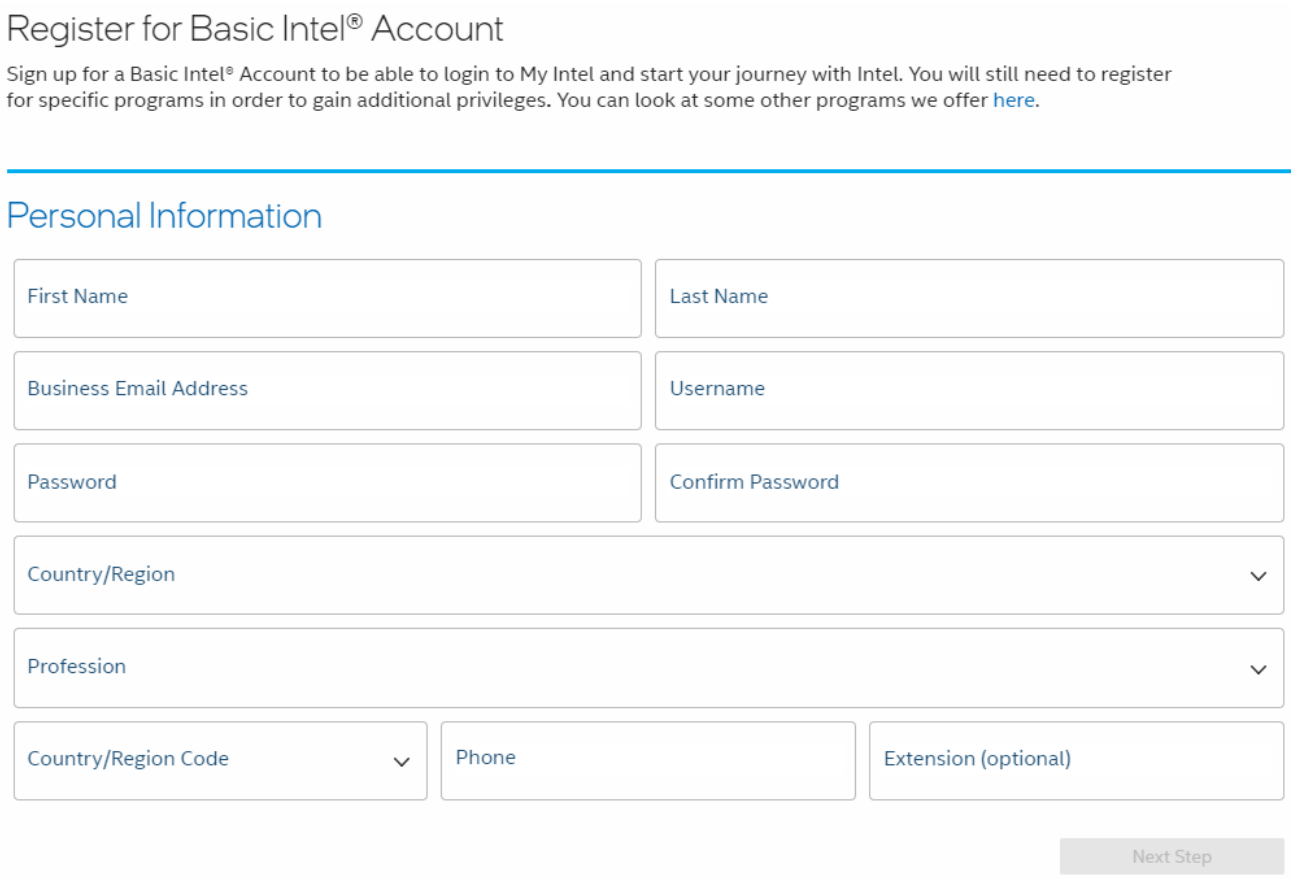
# Sign up Intel Website

- https://www.intel.com/content/www/us/en/homepage.html#



# Sign up Intel Website

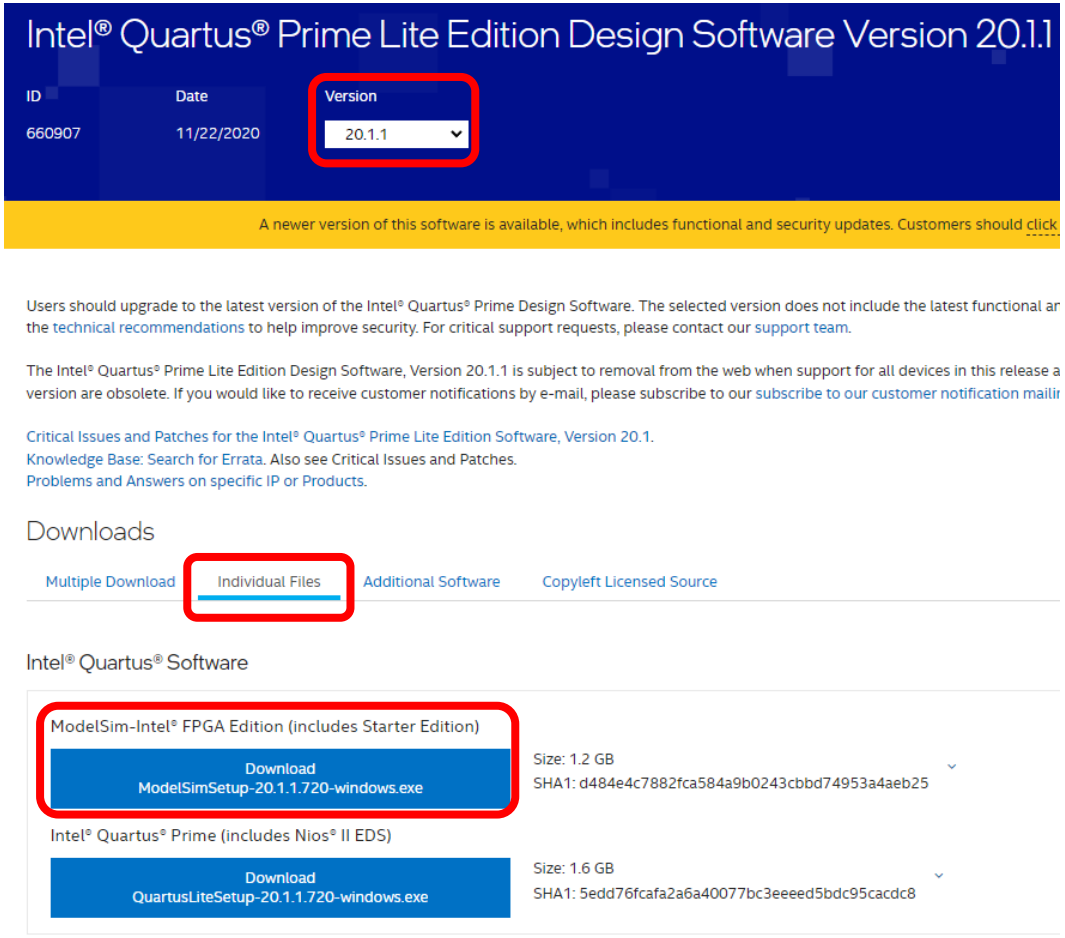
- https://www.intel.com/content/www/us/en/homepage.html#




*Fill out the registeration form*

# Download

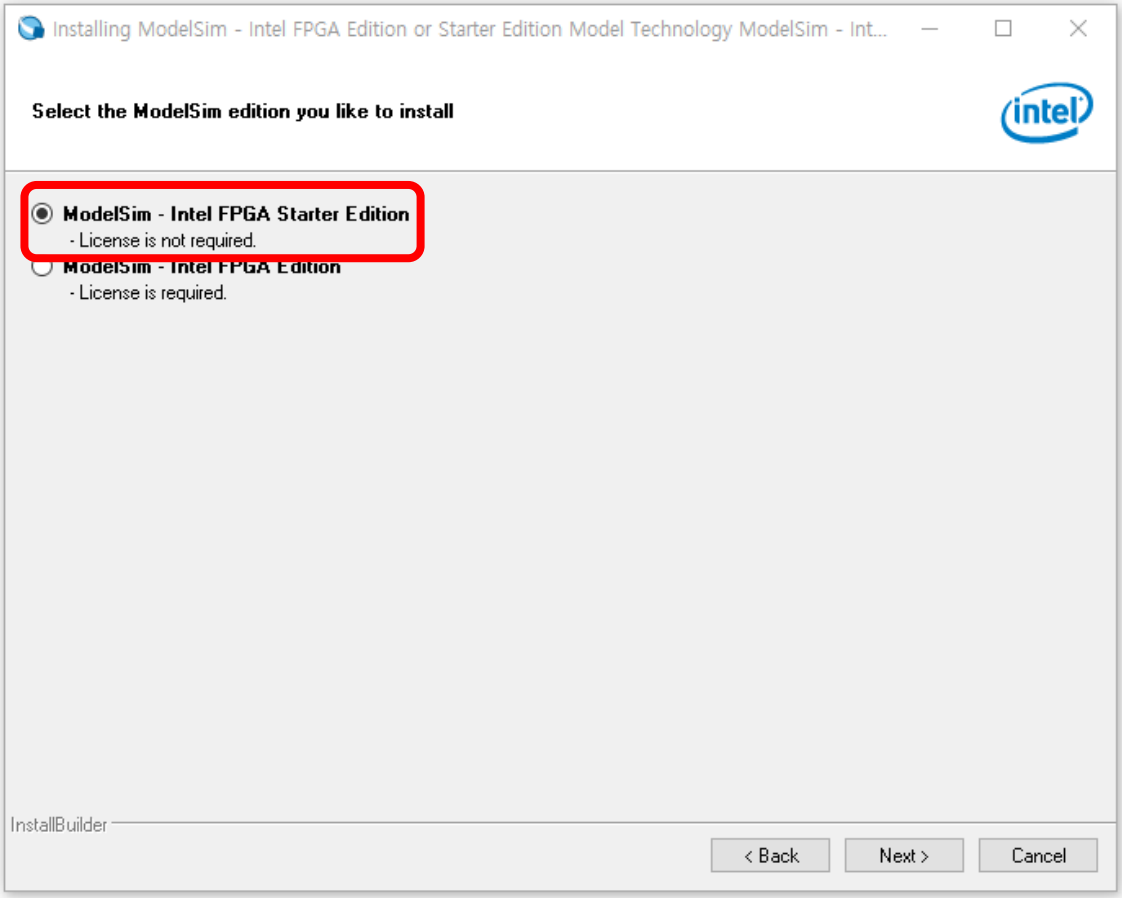
- https://www.intel.com/content/www/us/en/collections/products/fpga/software/downloads.html?s=Newest&q=lite



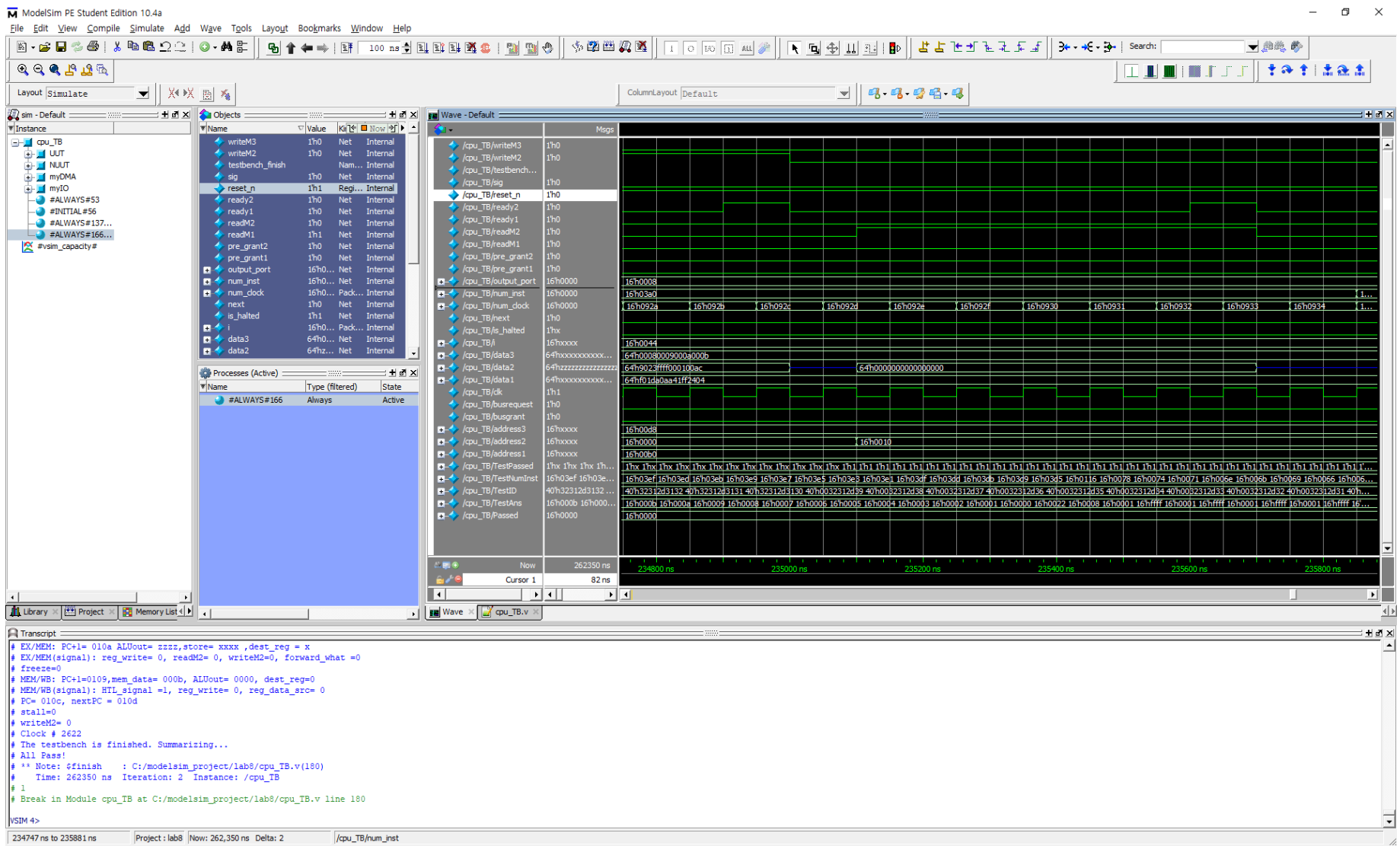

Note that we select “Lite” version (It does not require a license.)

# ModelSim Installation

- Select “Intel FPGA Starter Edition” and click next, next, ... until it finishes the installation

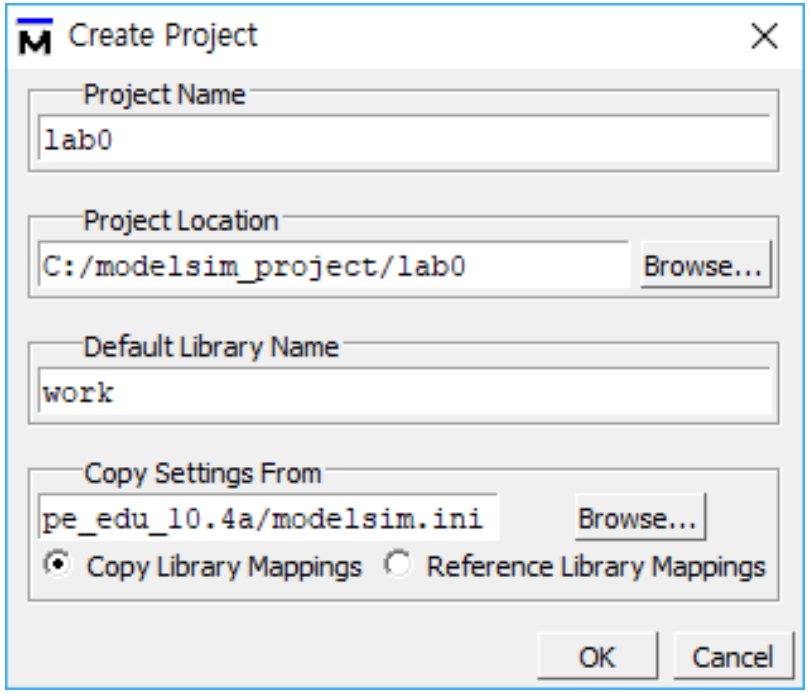


# Ready to use ModelSim!



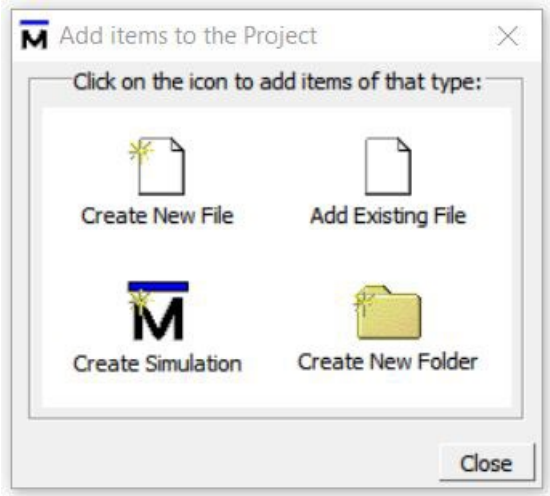
# Create new project

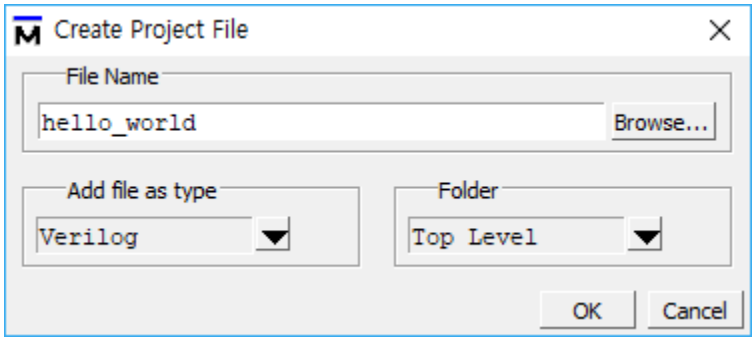
- #1. Create “lab0” directory in C:\modelsum_project
- #2. Run ModelSim
- #3. File ➔ New ➔ project



# Create new project

- #1. “Add items to the Project” window will pop up
- #2. Click “Create New File”
- #3. Create “hello_world” file with the type of Verilog



# Hello World!

- 4-bit synchronous up counter
- Update at clock posedge

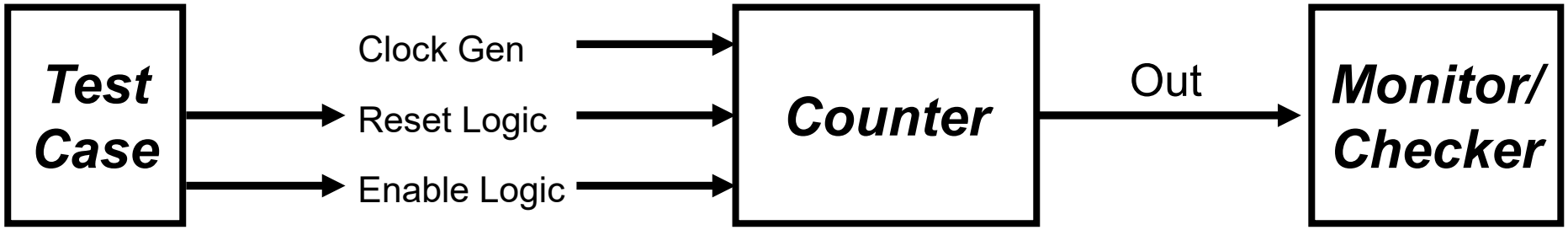
# Hello World!

- Copy & Paste Compile!

```
`timescale 1ns / 100ps
module counter (clock, reset, enable, out);
        input       clock;
        input       reset;
        input       enable;
        output      [0:3]  out;
        reg         [0:3]  out;
        initial begin
        end
        always @(posedge clock) begin
                if (enable == 1'b1) begin
                        if (reset == 1'b1)
                                out <= 4'b0000;
                        else begin
                                out <= out+1;
                        end
                end
        end
endmodule
```

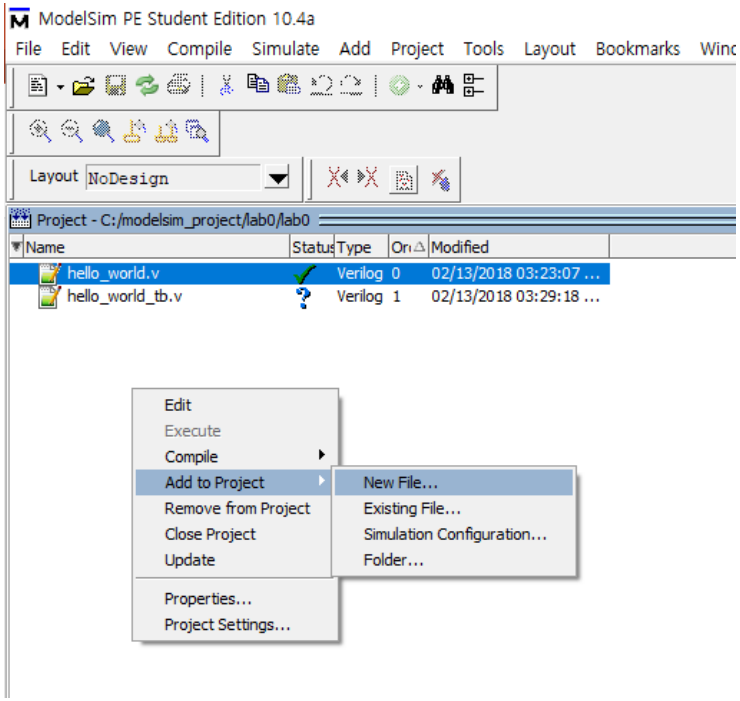
# Hello World!

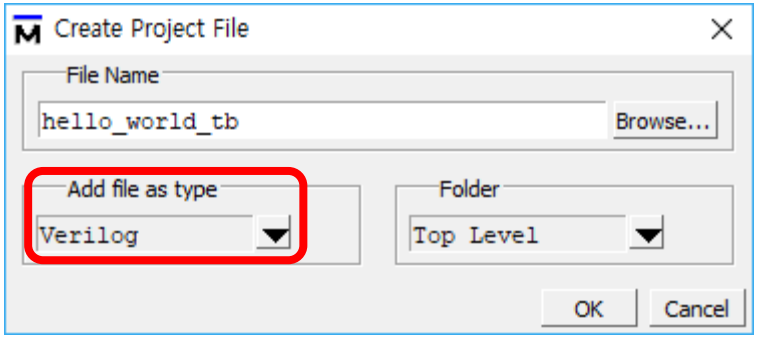
- We need testbench to test our module



# Hello World!

- Add new file for testbench to the project, and don't forget the type is Verilog



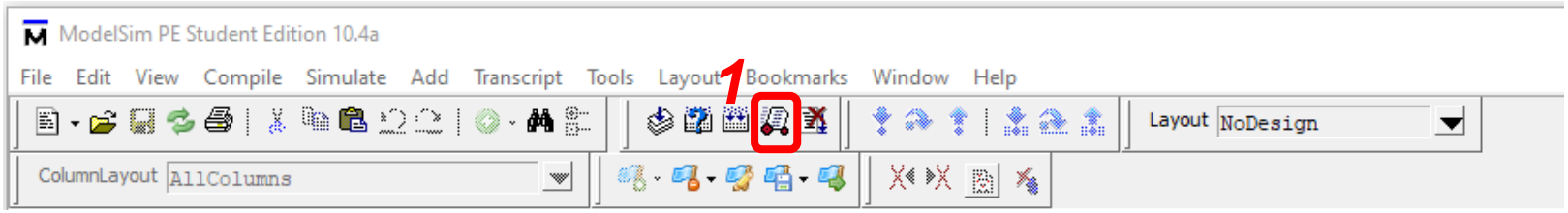
# Hello World!

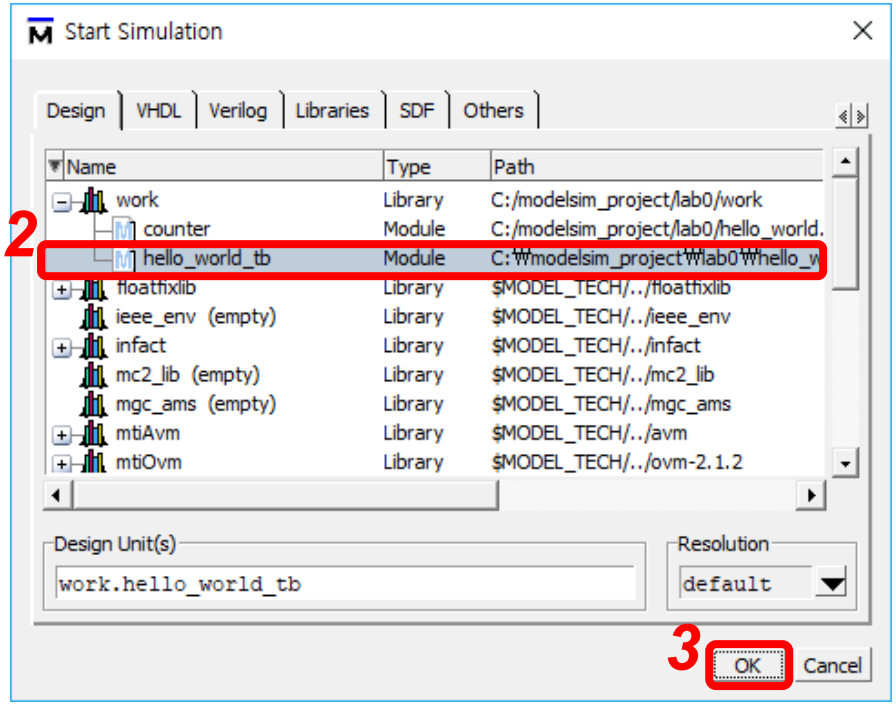
- Copy & Paste Compile!

```
`include "hello_world.v"
module hello_world_tb();
        reg clock_r, reset_r, enable_r;
        wire [0:3] counter_out;
        initial begin
                $display ("time\t clk reset enable counter");
                $monitor ("%T\t %b    %b     %b        %b", $time, clock_r, reset_r, enable_r, counter_out);
                clock_r <= 1;
                reset_r <=0;
                enable_r <= 0;
                #5 enable_r <= 1;
                #5 reset_r <=1;
                #10 reset_r <= 0;
                #100 enable_r <= 0;
                #5 $finish;
        end
        always begin
                #5 clock_r <= ~clock_r;
        end
        counter U_counter (clock_r, reset_r, enable_r, counter_out);
endmodule
```

# Simulation

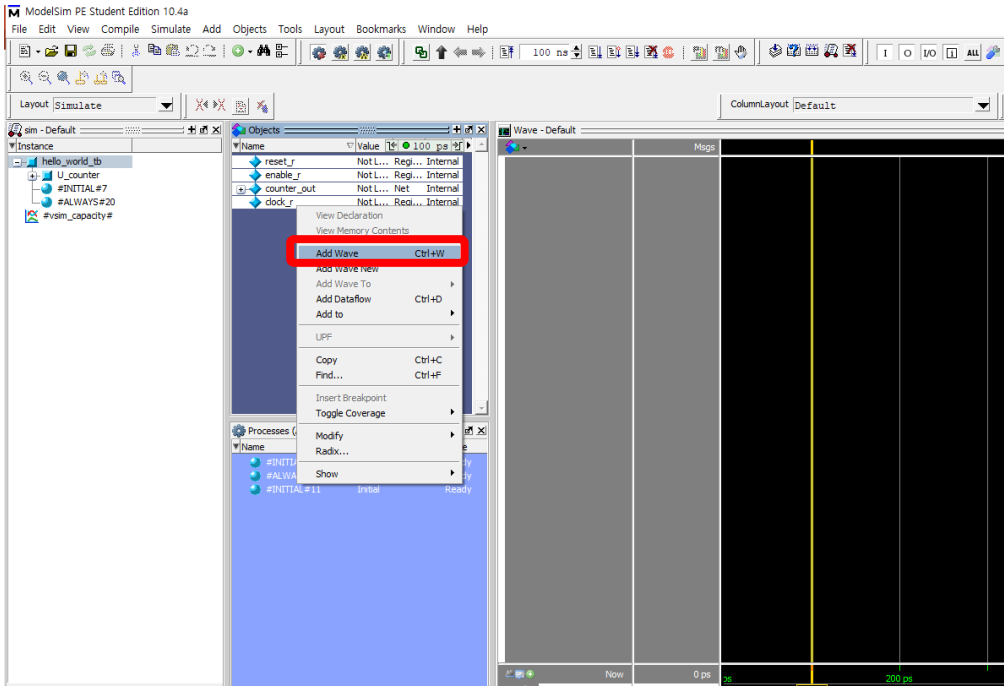
- After compile, let's simulate your code



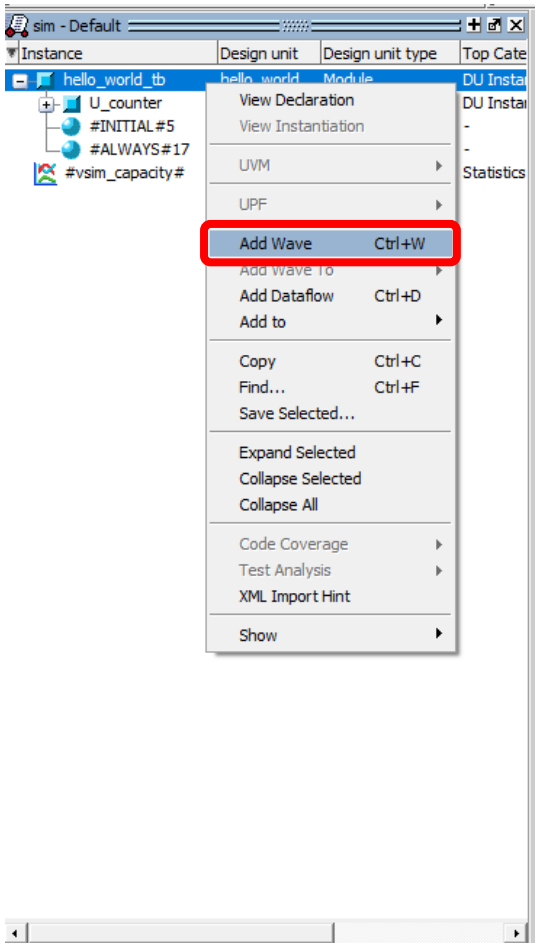



# Simulation

- Add waveform using one of the following ways

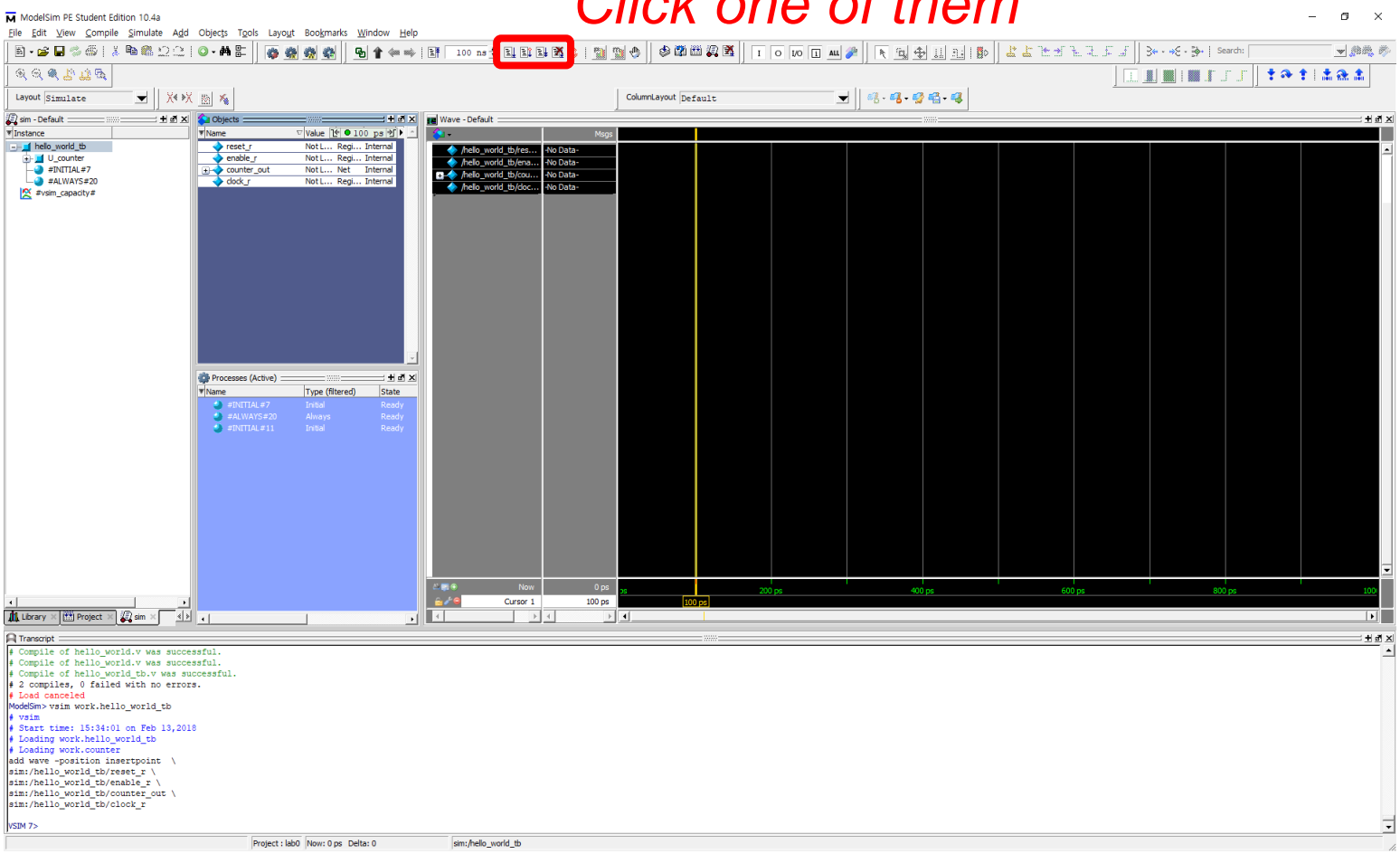



*or*



# Simulation

- Click one of buttons to run or break the simulation



# Simulation

- Now, you can see the waveform and output on the “Transcipt” tab