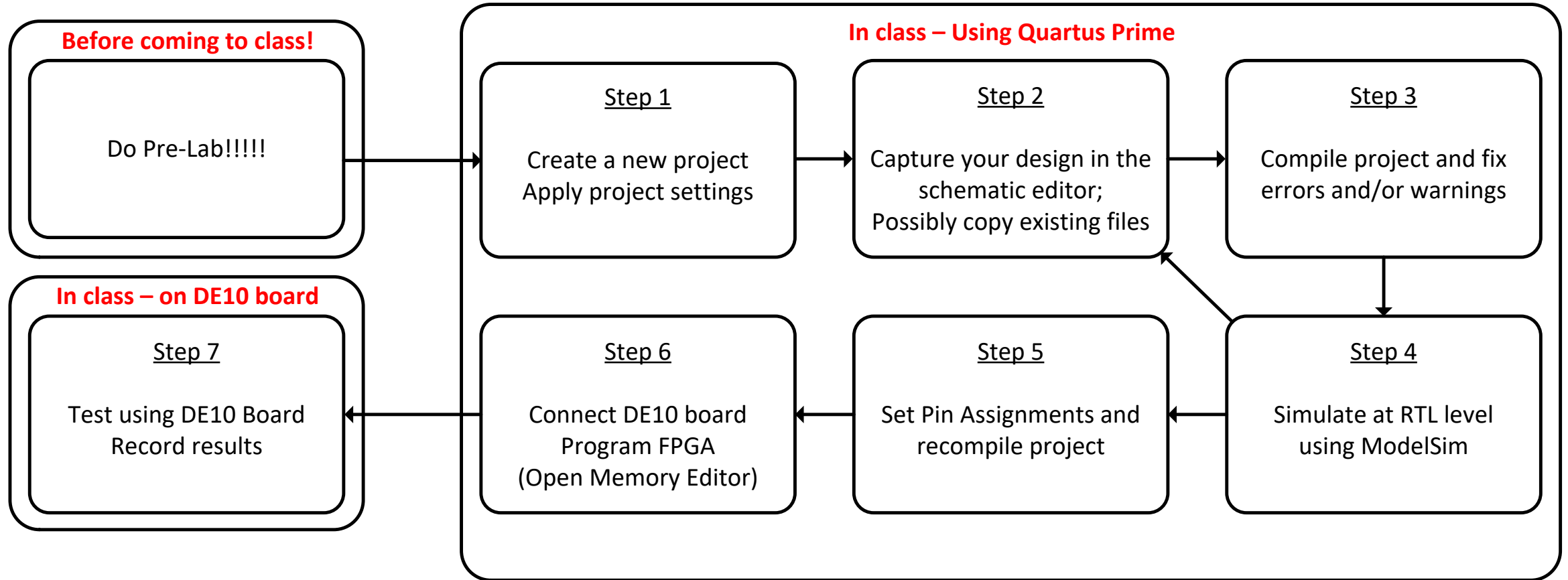


Announcements:

- Today: Modelsim
 - Short Slide Presentation
 - Lab 3
 - Tutorial
 - Memory
- Lab 4 “pre-lab” is last sheet of today’s lab.
 - Be sure to be ready for next week!

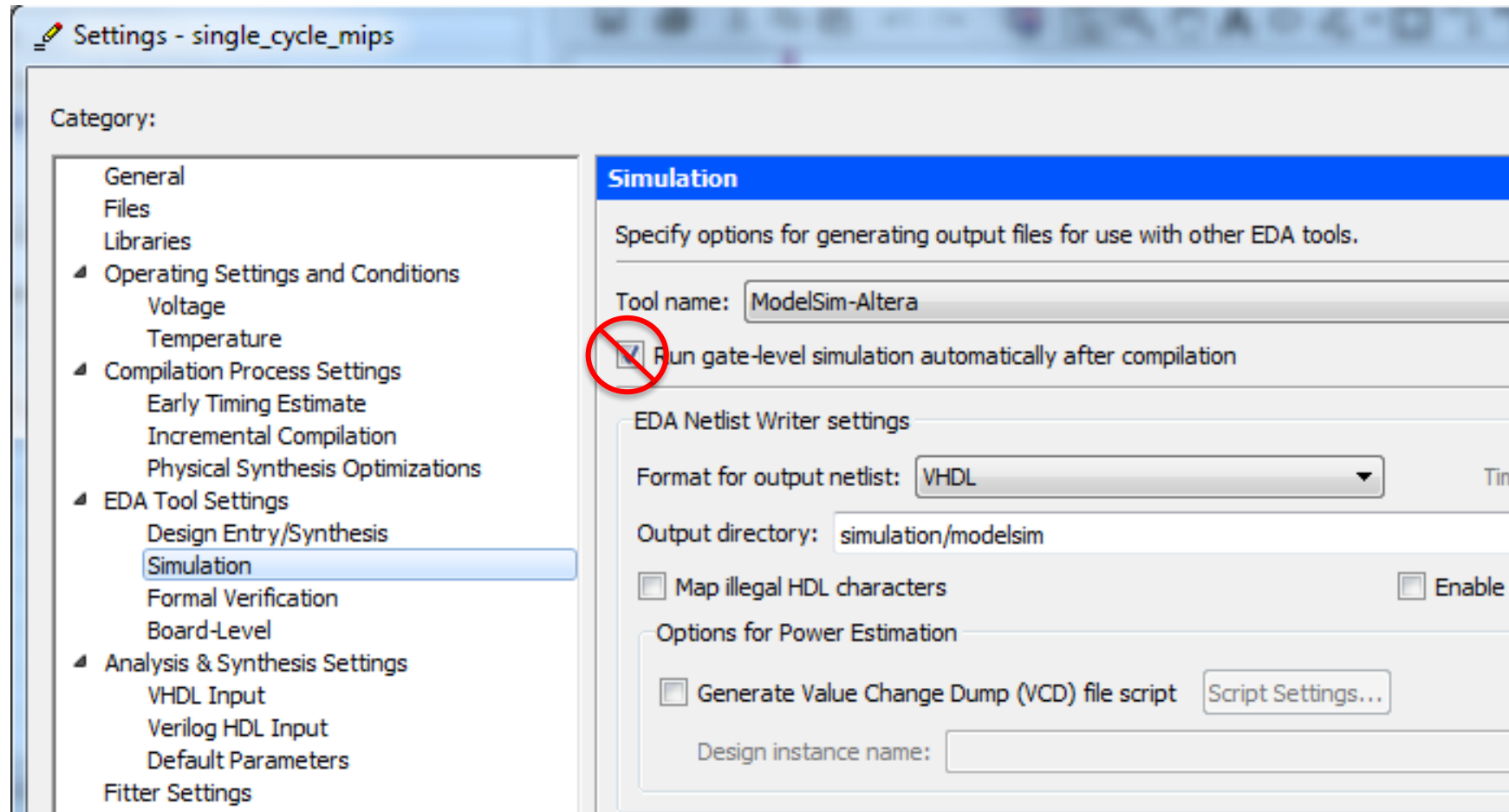
FPGA Design Flow



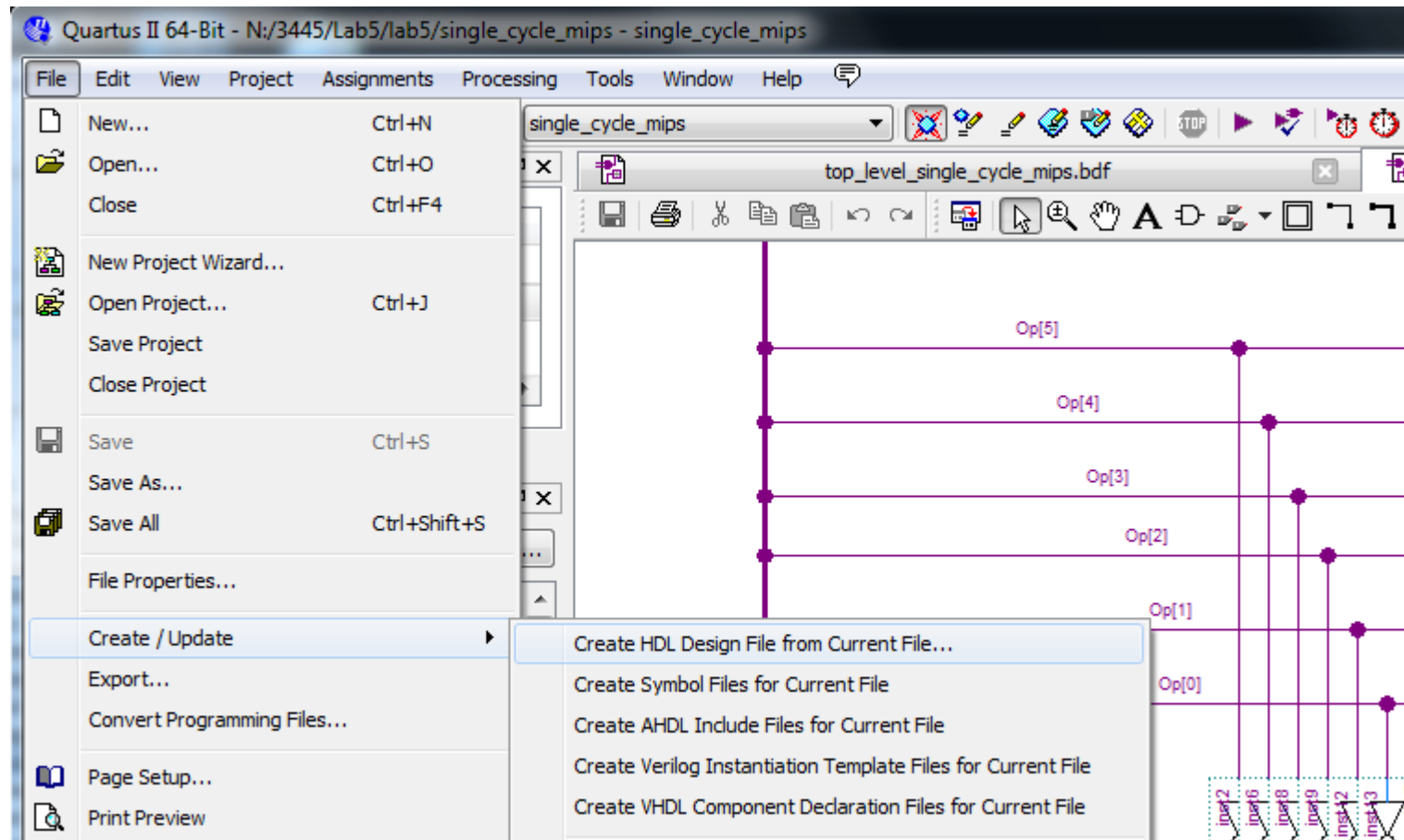
ModelSim Introduction

- A simulation and verification tool for designs using VHDL, Verilog, SystemVerilog, SystemC, etc.
 - Create VHDL version of your schematic (.bdf files) in Quartus
 - Let Quartus generate them automatically
 - Manually write VHDL models
- Apply stimuli to input signals and monitor output **and internal signals** in waveform window cycle by cycle.
 - All inputs should always be driven to some value!
 - Students often remember to “turn on” a control signal, but forget to “turn it off”!
- Will be used to test your lab designs before testing on the DE10 board
 - Much easier to view internal signals in simulation!
- GUI and commands file

Generate VHDL and Loads Modelsim Automatically from Quartus

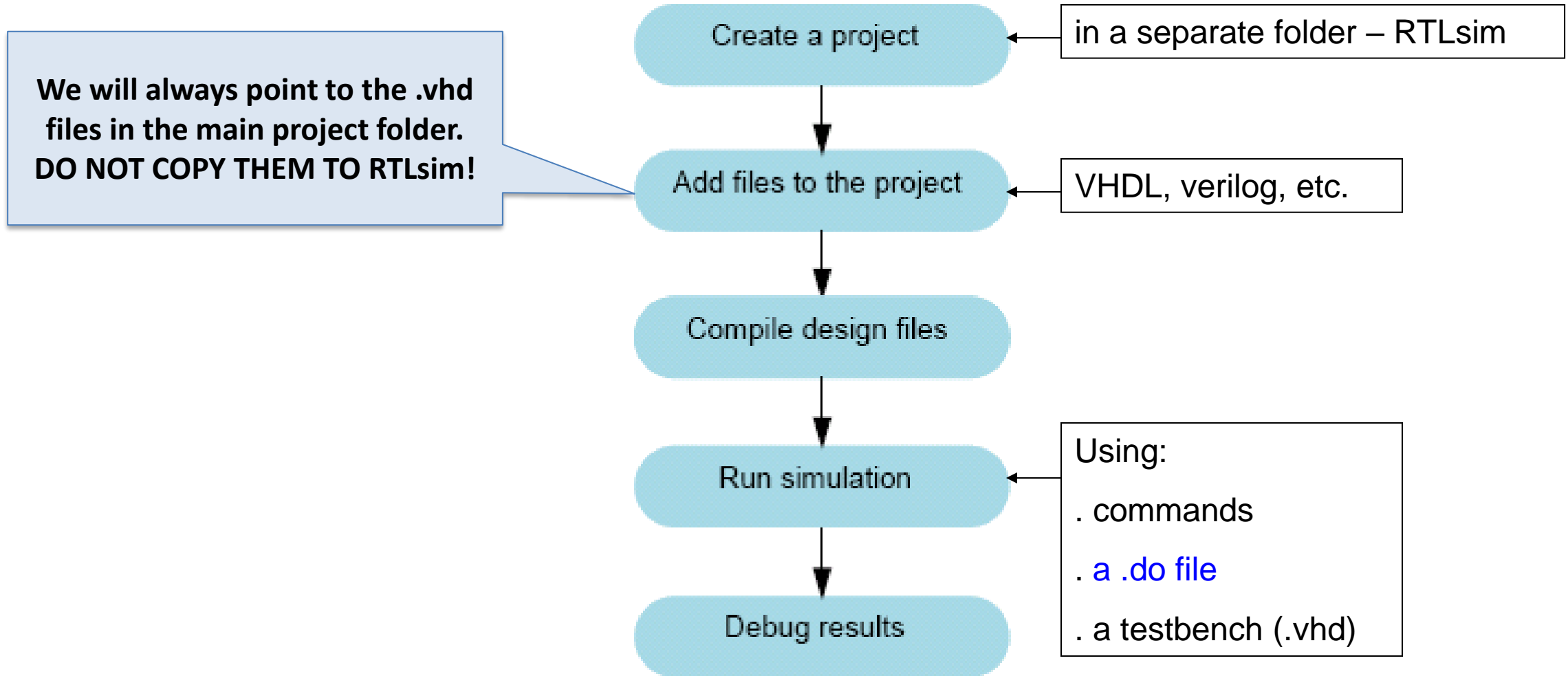


Generate VHDL from Schematic

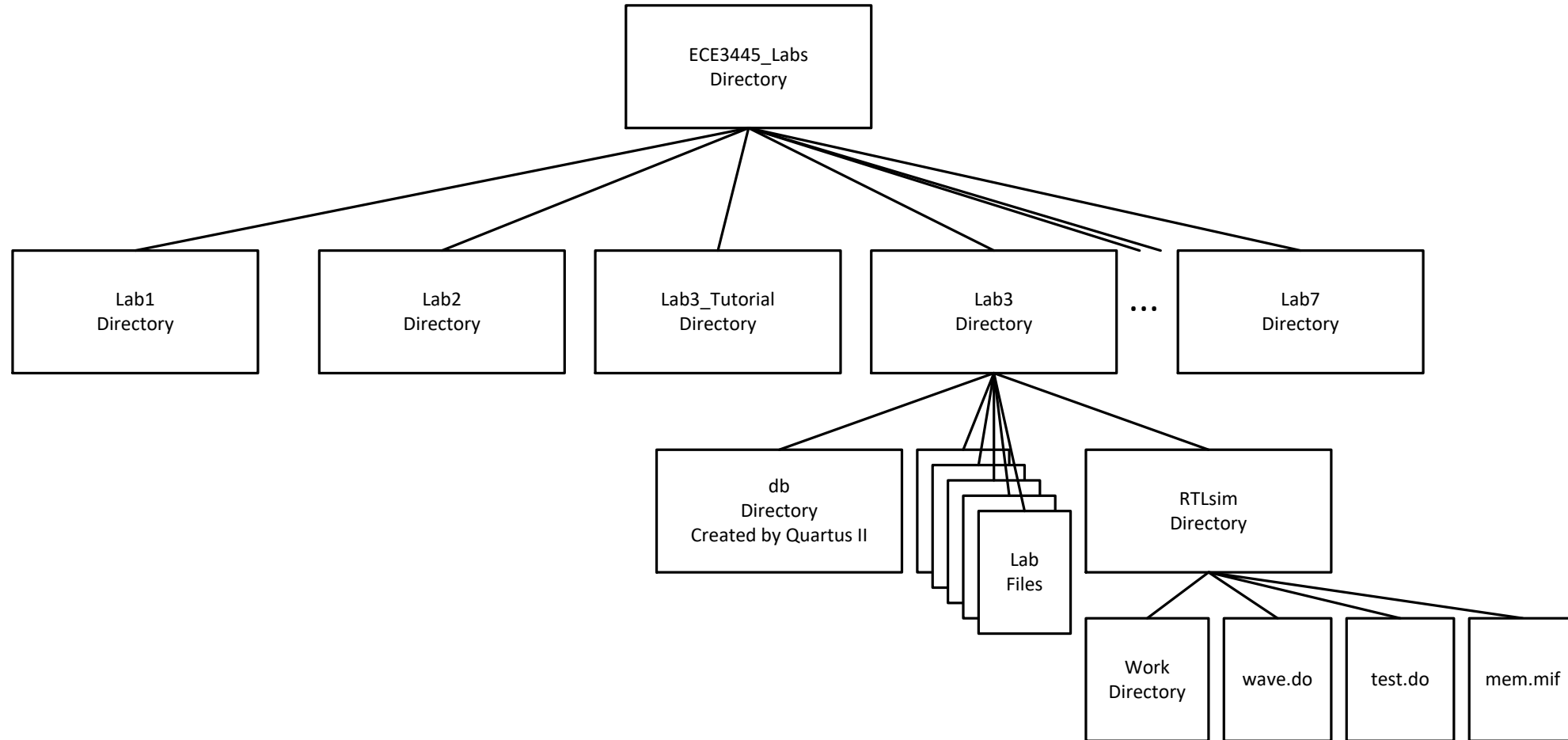


Don't forget to select “VHDL” for model output!!!

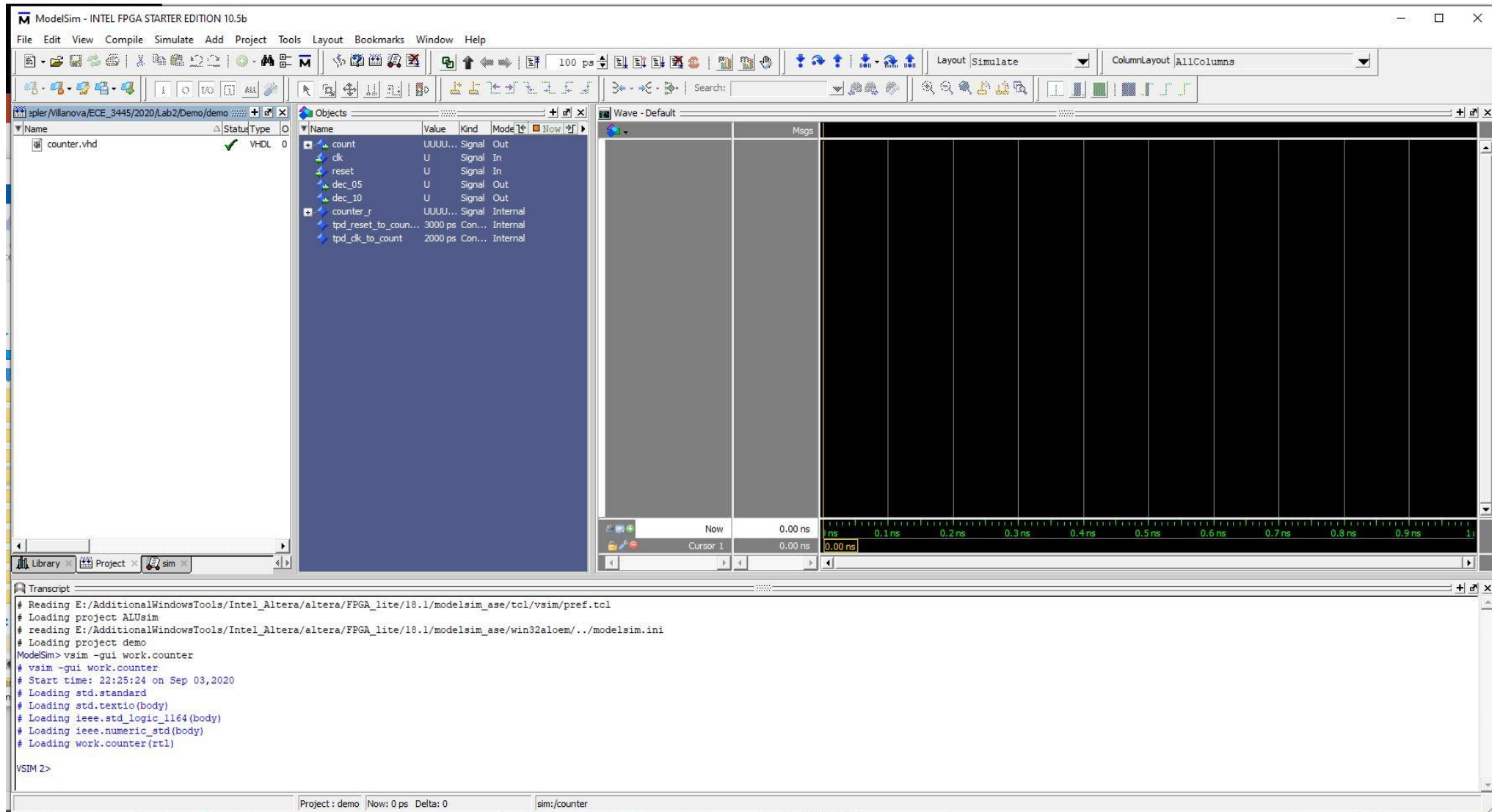
General ModelSim Simulation Flow



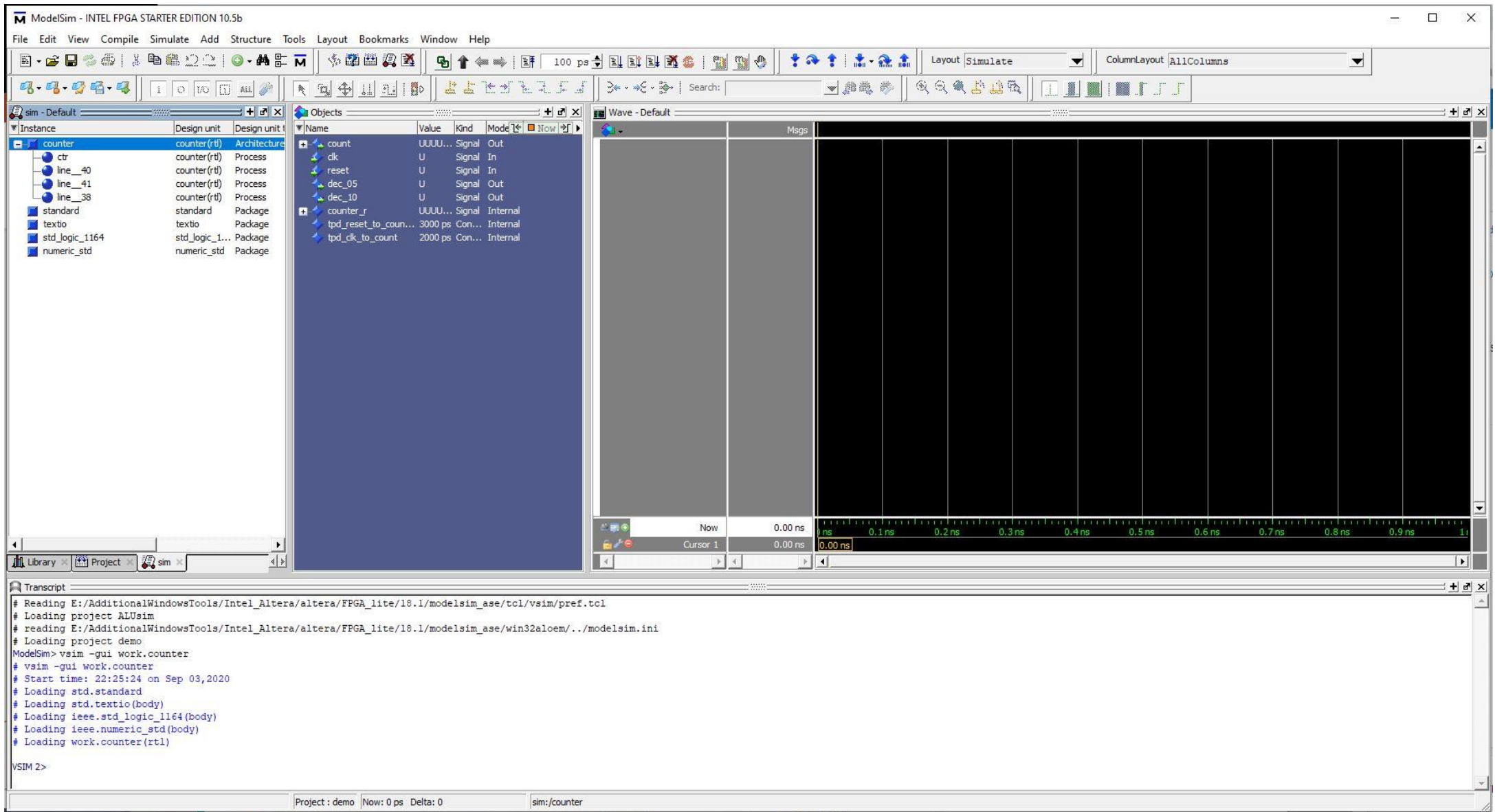
Directory Structure



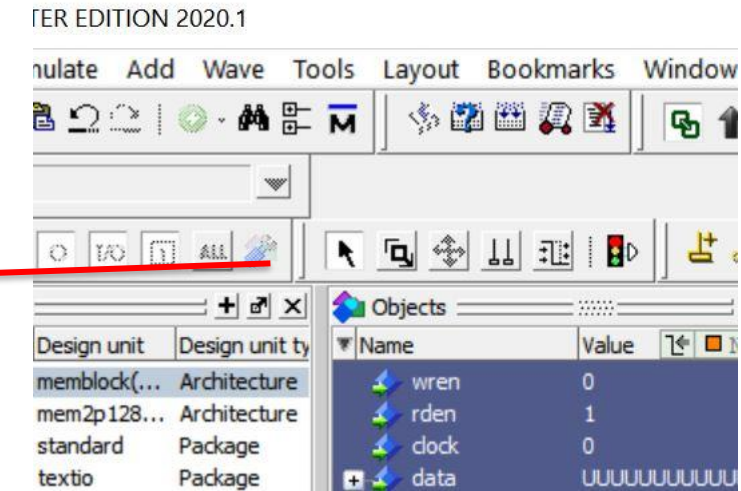
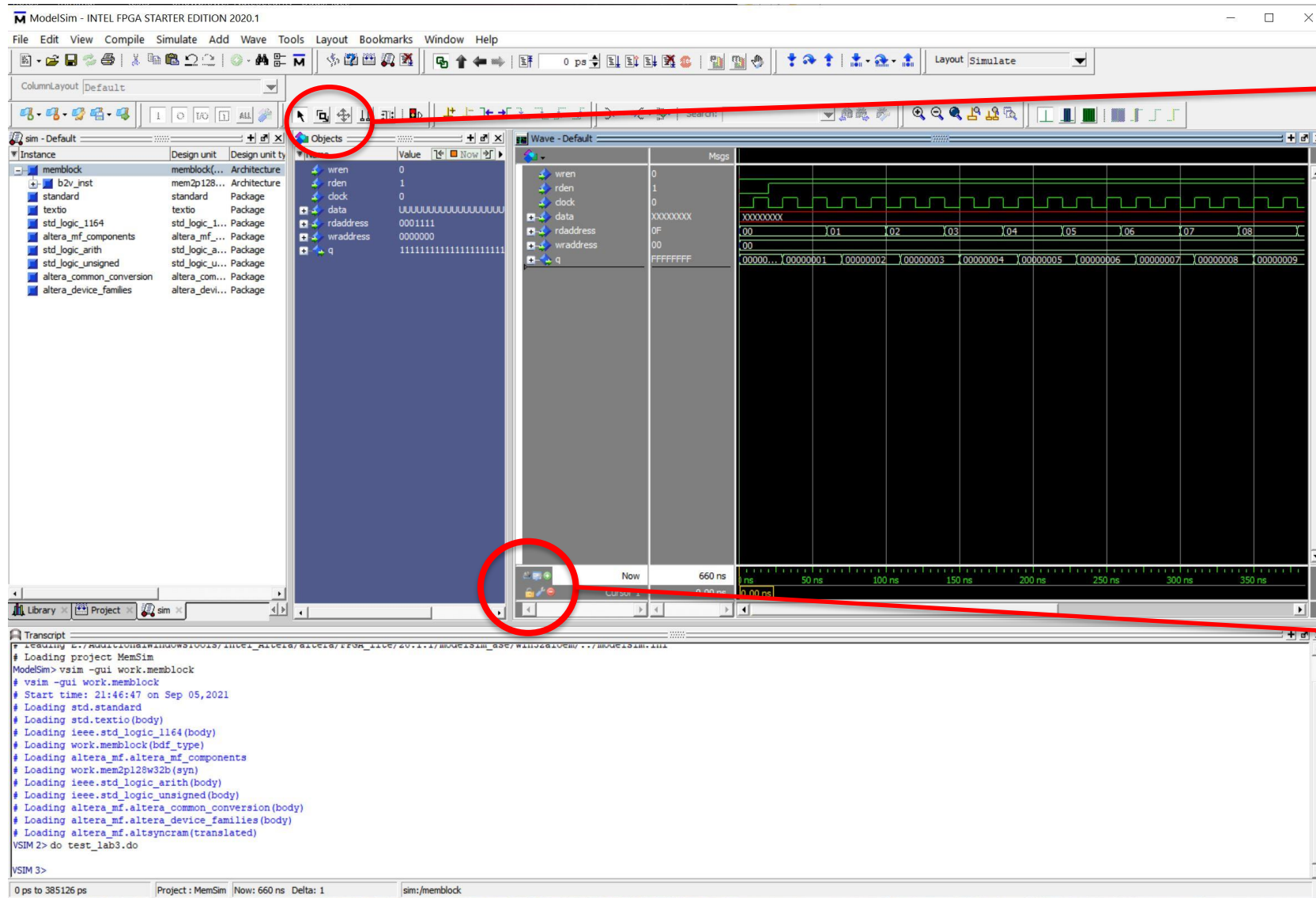
ModelSim Windows (1)



ModelSim Windows (2)



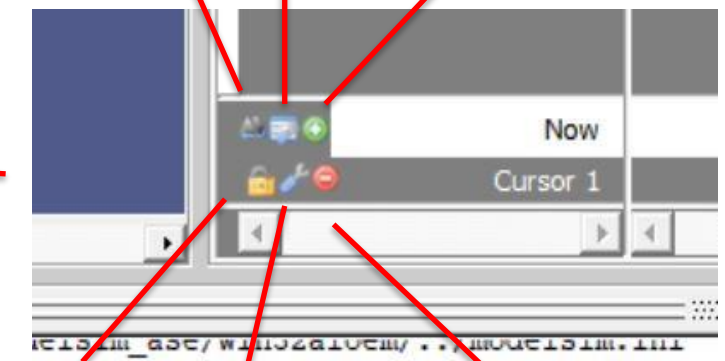
ModelSim Windows (3)



Names

Settings

Add Cursor



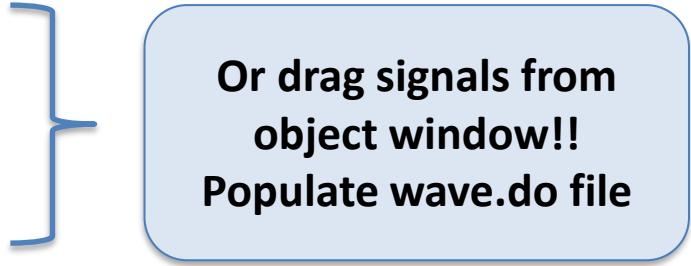
Lock

Edit Cursor

Delete Cursor

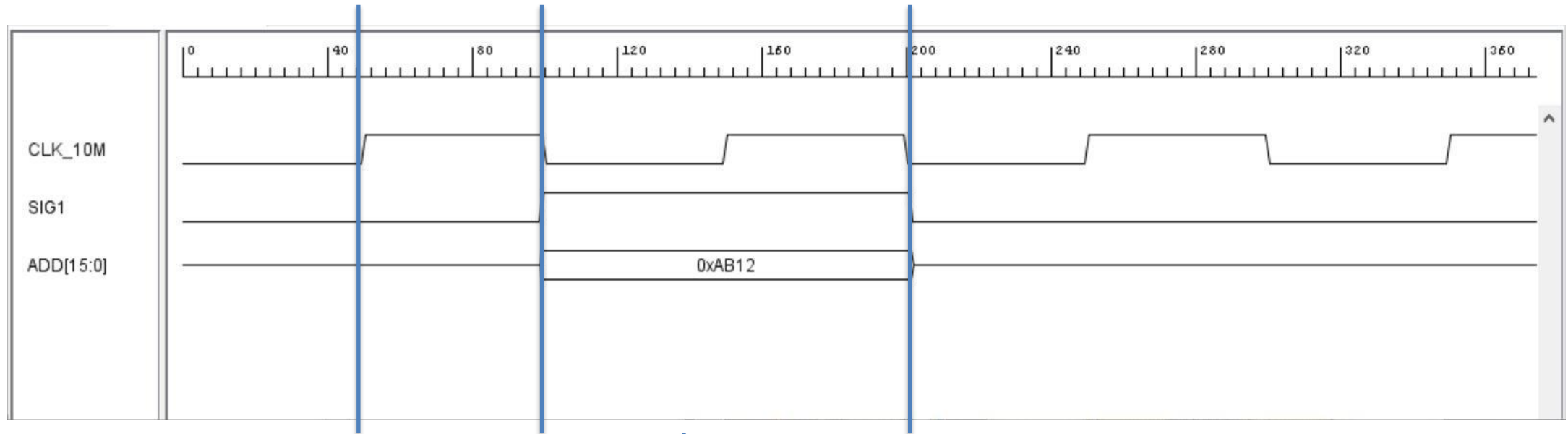
Stimulus: .do file

- A text file containing Modelsim commands to control the simulation
- Comments: start with “#”
- **restart -f** command forces a reset of the simulation
- **delete wave *** # delete all signals
- “**add wave**” command adds signals into the wave window
 - **add wave /count** # add signal “count” (default binary)
 - **add wave -radix hexadecimal /count** # add signal “count” (force to hexadecimal)
 - **add wave * -r** # add all signals
 - **add wave -divider divider_title** # add section title
- **force <signal> <value> <start time>**
 - Drives (forces) the specified signal to specific value at (optionally) a given time
 - Examples:
 - Wires:
 - **force abc 1** # for std_logic signals (1 bit)
 - **force reset 1 0 ns, 0 40 ns** # for std_logic signals (1 bit) (multiple values; last one “forever”)
 - Buses:
 - **force bus1 2#00001000** # binary, for std_logic_vector signals
 - **force bus1 16#08** # hexadecimal, for std_logic_vector signals
 - Clock signals:
 - **force clk 0 0 ns, 1 20 ns -repeat 40 ns** # “repeated” signal (1 bit) forever
 - **force clk 0 0 ns, 1 20 ns -r 40 ns -cancel 1000 ns** # “repeated” signal (1 bit) limited time
- **run 1000 ns** # command asks to simulate the design (advance simulation time) for 1000 ns
- Consult the [ModelSim command reference](#) for more details.



Or drag signals from
object window!!
Populate wave.do file

Sample Stimulus



force CLK_10M 0 0 ns, 1 50 ns –repeat 100 ns
force SIG1 0 0 ns, 1 100 ns, 0 200 ns
force ADD 16#0 0 ns, 16#AB12 100 ns, 16#0 200 ns
run 300 ns

**“Schedule” all signal changes,
then Run simulation**

force CLK_10M 0 0 ns, 1 50 ns –repeat 100 ns
force SIG1 0
force ADD 16#0
run 100 ns
force SIG1 1
force ADD 16#AB12
run 100 ns
force SIG1 0
force ADD 16#0
run 100 ns

**“Schedule” clock,
Set signals
Run
Set signals
Run**

Typical .do file

- #####
- # A simple .do file template #
- # Assume simulation module is already loaded #
- #####
- # Reset the simulator
- restart -f
- # Remove all signals from wave window
- delete wave *
- # Add signals into wave window (using script)
- # wave.do file must already exist
- do wave.do
- # Define "required" signals
- # clock = 0 at 0 ns, 1 at 10 ns, and repeat the pattern every 20 ns
- force clk 0 0 ns, 1 10 ns -r 20 ns
- # reset = 1 at 0 ns, 0 after 20 ns (High active reset)
- force reset 1 0 ns, 0 20 ns
- # Force a set of signals
- force sig1 0
- force sig2 0
- force sig3 0
- # Run simulator for some period of time
- run 40 ns
- # Force signals some other way
- force sig1 1
- force sig2 0
- force sig3 1
- # Run simulator for some more time
- run 40 ns
- # Repeat as needed...

Simple ModelSim Simulation Steps (included in Lab3 description)

- Create a new directory (Typically RTLsim).
- Create a new Modelsim project, then add your files to the project
 - Make sure new project is in “RTLsim”
 - “new” -> “project”, a new library (typically “work”) is also created here
 - In ModelSim, all designs are compiled into a library.
 - "Work" is the library name used by the compiler as the default destination for compiled design units. ModelSim library format is compatible across all supported platforms.
 - Click on “Add existing file” to add the design files or use “Project” -> “add to project” -> new file or existing files
 - Browse to where the .vhd files exist (typically one directory up). **“Reference”** them – DO NOT COPY THEM into the RTLsim directory!
 - Use “pwd” and “cd” to check and change current working directory if necessary
- Compiling Your Design into the library
 - “compile” -> “compile all” or other options, check the results, a green checkmark indicates a successful compilation
- Loading the Simulator with your Design and Running the Simulation
 - Invoke the simulator on a top-level module or a configuration of entity/architecture pair : “simulate” -> “start simulation” -> select the top entity in the work library
 - Open the wave window: “View” -> “waves” or add wave in .do file
 - Three ways to add signals to Wave window: a. Right click on the top module and select “Add” -> “Add All Signals to Wave”; b. “view” -> “objects”, then drag the signals into the wave window; c. add the signals in .do file
 - In the command window, run: do test_file.do
 - “Restart” clears previous simulation results
- Debugging Your Results
 - Use ModelSim’s debugging environment to track down the cause of the problem (tracing signal flows).
 - A common problem: uninitialized signals cause others involved in an operation to have unknown values.
- Save waveforms and End simulation
 - File -> Export Image
 - “Simulate” -> “End Simulation”

Or use Snipping Tool

What you should demonstrate

- Create new cursors
- Move the cursors
- Zoom into a location
- wave.do file
- test_lab3.do file

Directory Structure

