COSE222, COMP212 Computer Architecture

MIPS CPU Synthesis and Simulation

This note summarizes how to do the hardware simulation. You are provided with a simple MIPS CPU design in Verilog. The CPU design has implemented only a few MIPS instructions. Its implementation is based on single-cycle execution, meaning that each machine code (instruction) is executed in one clock cycle. The design also comes with a simple testbench. The test (assembly) program is loaded in memory after compilation. For hardware simulation, we use a CAD (Computer Aided Design) tool called `ModelSim`, which is widely used in industry. Follow the steps below to run the MIPS CPU design with `ModelSim`.

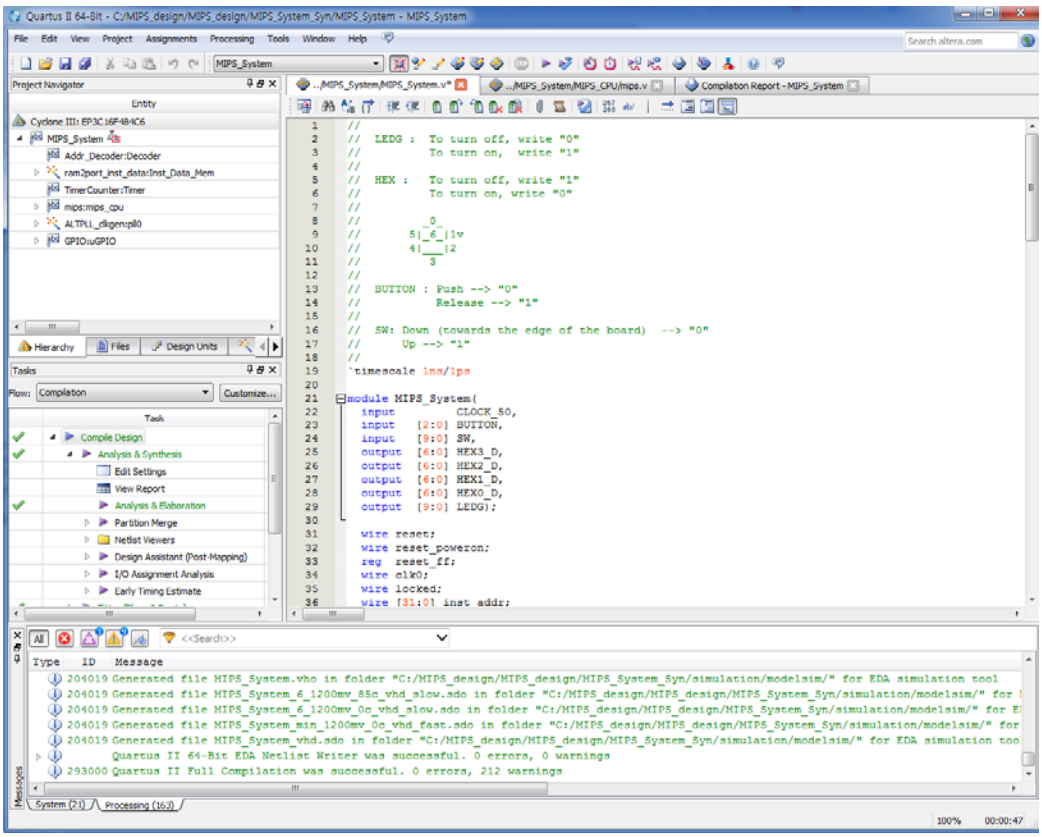
Prior to following the instructions below, install the EDA (Electronic Design Automation) tools linked on the class web: **ModelSim (v10.1d)** and **Quartus-II (v13.1)**.

Download the MIPS CPU design linked on the class web, unzip `MIPS_design.zip` and you will find 3 directories:

- **`MIPS_System`** contains the Verilog source code for MIPS CPU, peripheral devices (Timer and GPIO), etc. Traverse the directory structure to see what is where...
- **`MIPS_System_Sim`** contains the compiled binaries using ModelSim of the MIPS system source code for hardware simulation.
- **`MIPS_System_Syn`** contains the synthesized code of the MIPS system source code using Quartus-II.

- **Synthesis using Quartus-II and Download to DE0 Board**

1. Double-click the project file (`MIPS_System`) located at the `MIPS_System_Syn` directory
    - I have created a Quartus-II project for you. The project file contains all the necessary files for test-driving.

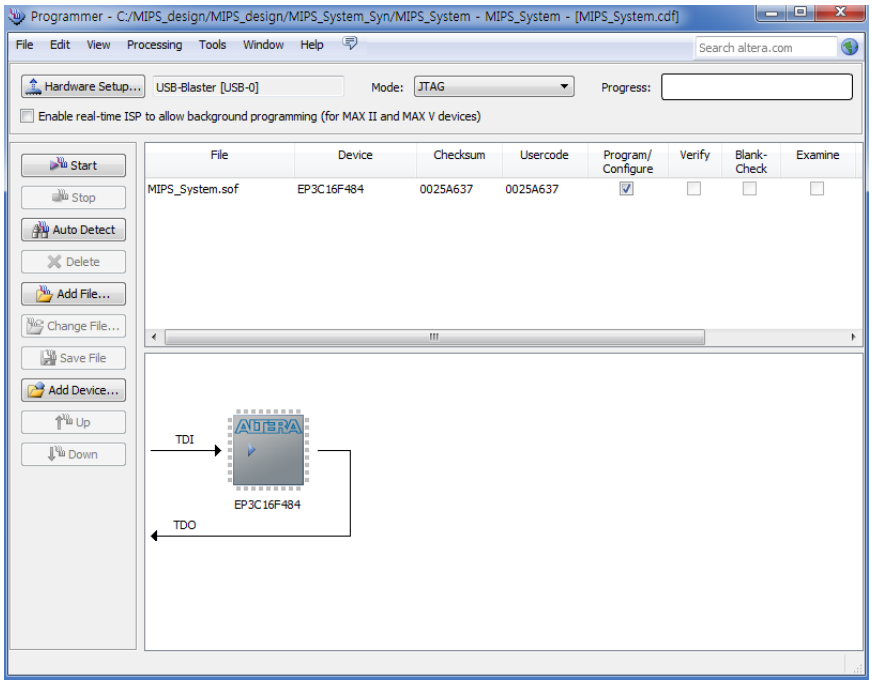


2. Synthesize the whole design again
    - Select `MIPS_System` in the **Project Navigator** pane
    - **Processing** → **Start Compilation.** It will take a little bit of time to compile the whole design

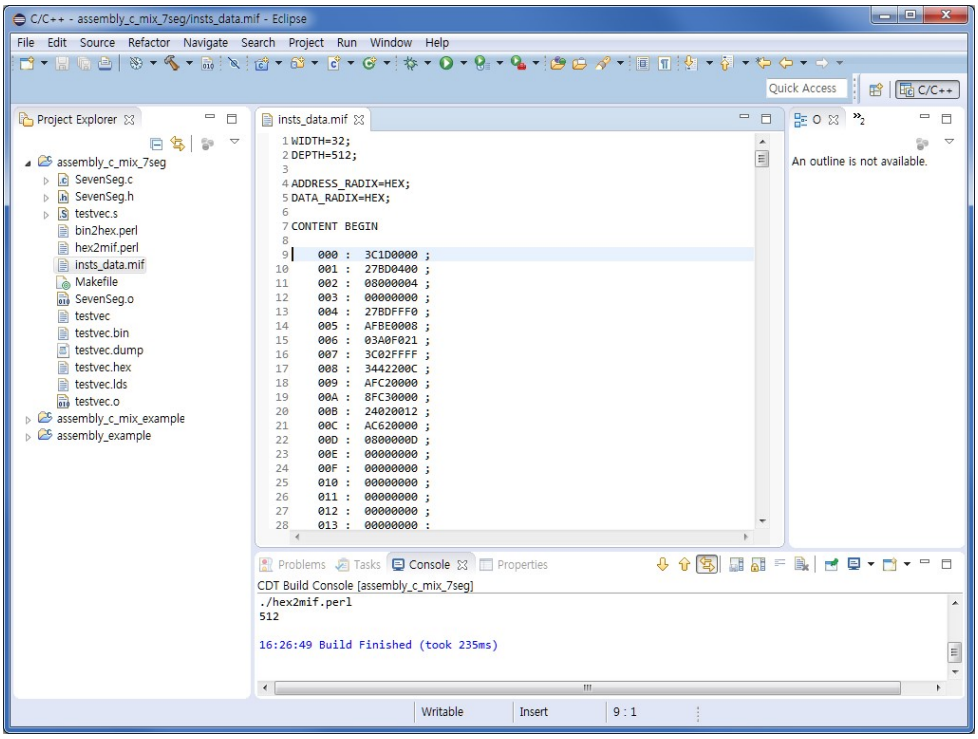
3. Download the design to the Cyclone-III FPGA on DE0 board
    - **Tools** → **Programmer,** Then click on **Start**
    - Make sure that you have USB-Blaster set up correctly on the upper left of the window. If it is not set up correctly, click on **Hardware Setup** button and add USB blaster. If you don't have the USB blaster listed with the Hardware Setup, you should install the USB device driver. The driver is located at the Quartus-II installation directory. In my case, the driver is located at

      C:₩altera₩13.1₩quartus₩drivers₩usb-blaster

- **Number 5 should be displayed on the HEX0 7 Segment**



4. To change the program you want to run on MIPS, compile your code with Eclipse.
    - It will generate a file called **`insts_data.mif`**: `mif`
    - mif (memory initialization file) is for initializing memory supported by Altera FPGAs)
    - I have created a memory for the class project via **Tools → MegaWizard Plugin Manager** on Quartus-II

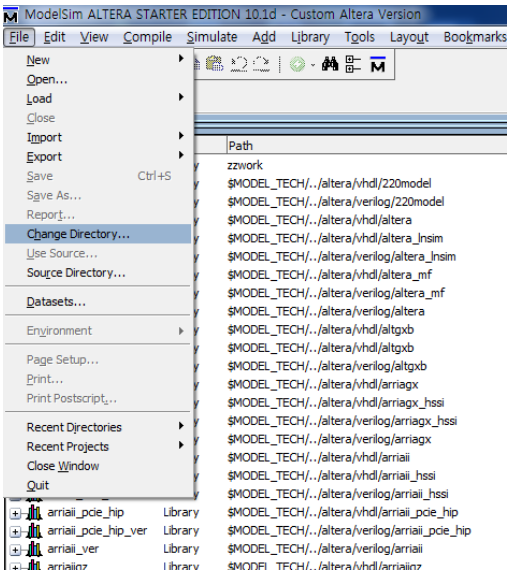


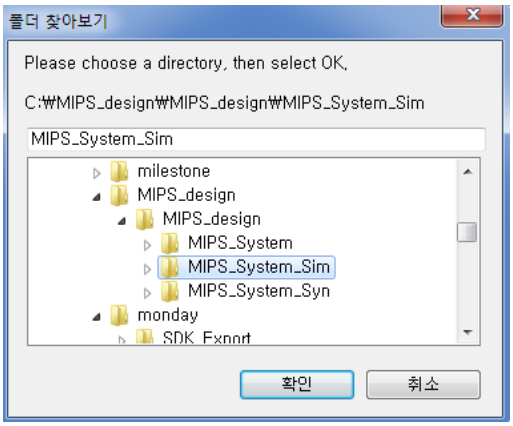
5. To download your program (mif) with the hardware design to the DE0 board
    - Download an example program (Assembly and C mixed program) at http://esca.korea.ac.kr/teaching/cose222_CA/milestones/assembly_c_mix_7seg.zip . The program will display a number 5 on HEX0 7 Segment.
        - ✓ Create an Eclipse project to compile the code
    - Copy **`insts_data.mif`** to the `MIPS_System_Syn` directory
    - Then, you have 2 options for downloading
        - ✓ 1st option: follow from step 1 to step 3
        - ✓ 2nd option: Execute the `quartus_mem_update_download.bat` by double-clicking the file (you don't have to start Quartus-II). The file is located under the `MIPS_System_Syn` directory
    - Change the example code, so it can display 7 on the 7 Segment

- **Hardware Simulation with ModelSim**

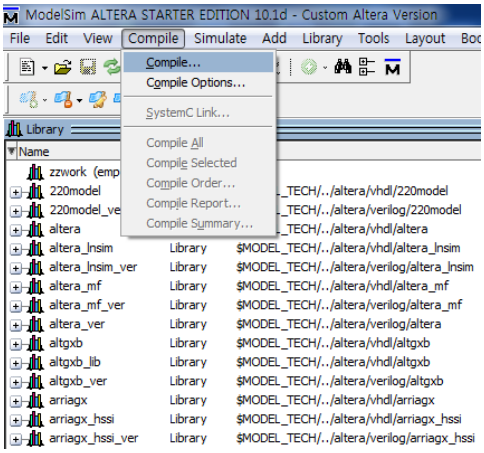
1. Invoke ModelSim

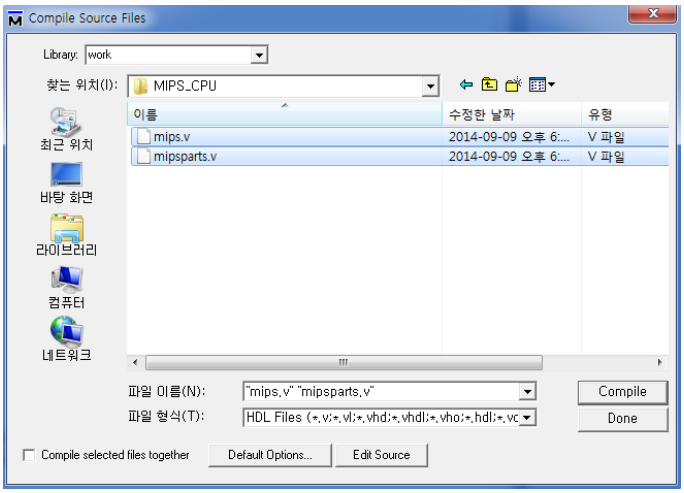
2. Change to the `MIPS_System_Sim` directory
   - **File → Change Directory**



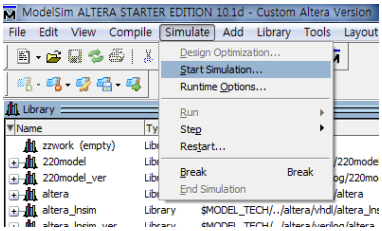


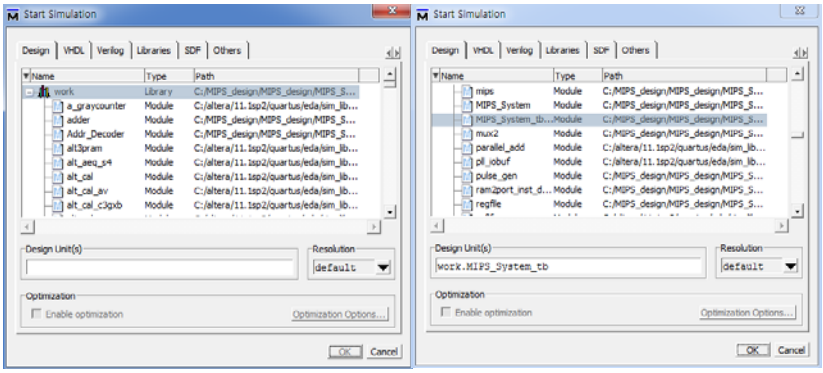
3. I have created a library (called **work**) and compiled the Verilog design
   - So, the **work** directory contains the compiled version of the Verilog code.
   - Remember that every time you change and/or add the Verilog code, you have to compile it before simulation.
   - Let's compile the CPU source as an example (I already did it though)
     - ✓ **Compile → Compile**
     - ✓ In the **Look in** tab, change the directory to where the CPU source code is (`MIPS_System/MIPS_CPU`)
     - ✓Select all the files and click on **Compile**



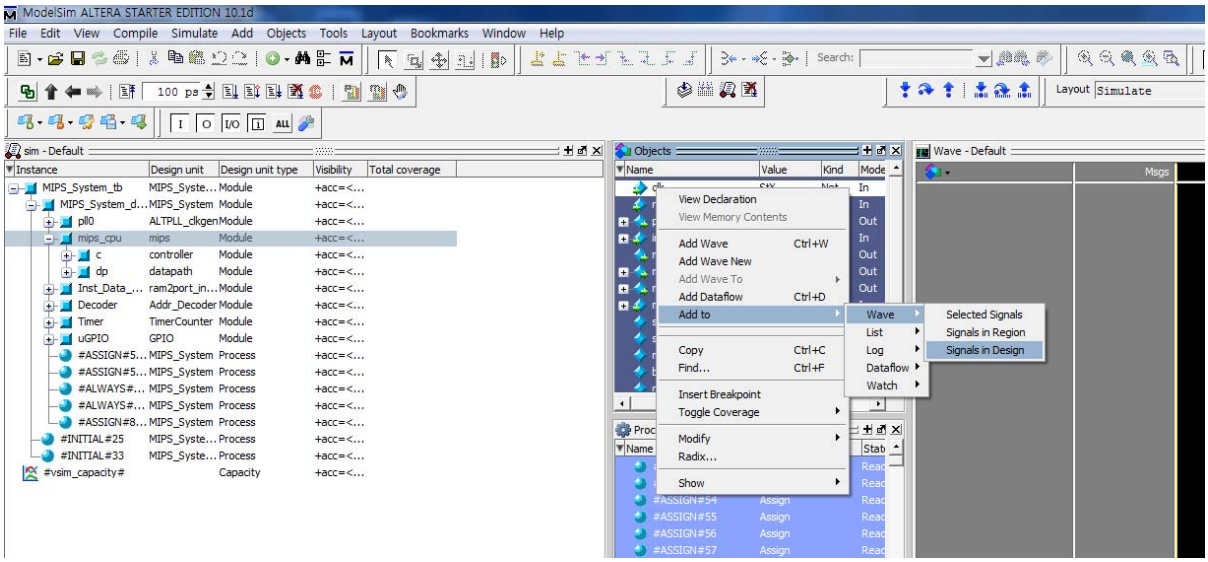


4. Start Simulation
    - **Simulate → Start Simulation**
    - Select `MIPS_System_tb` (testbench) under **work** and click on **OK**
        - ✓ Check out the testbench source code, which is located at `MIPS_System/MIPS_System_tb.v`

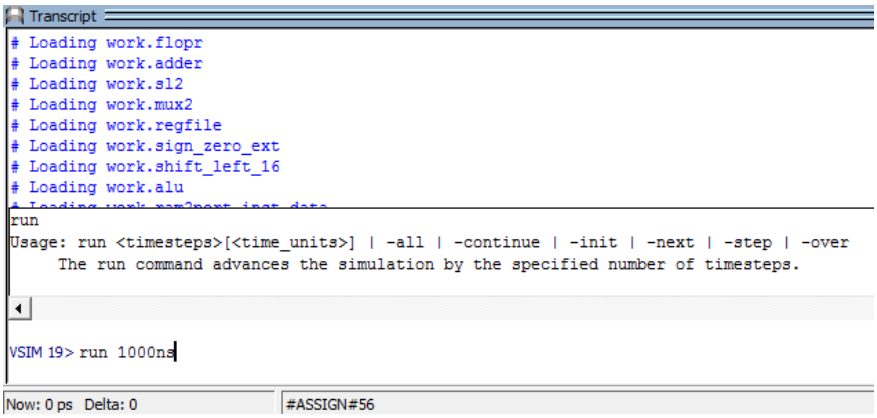




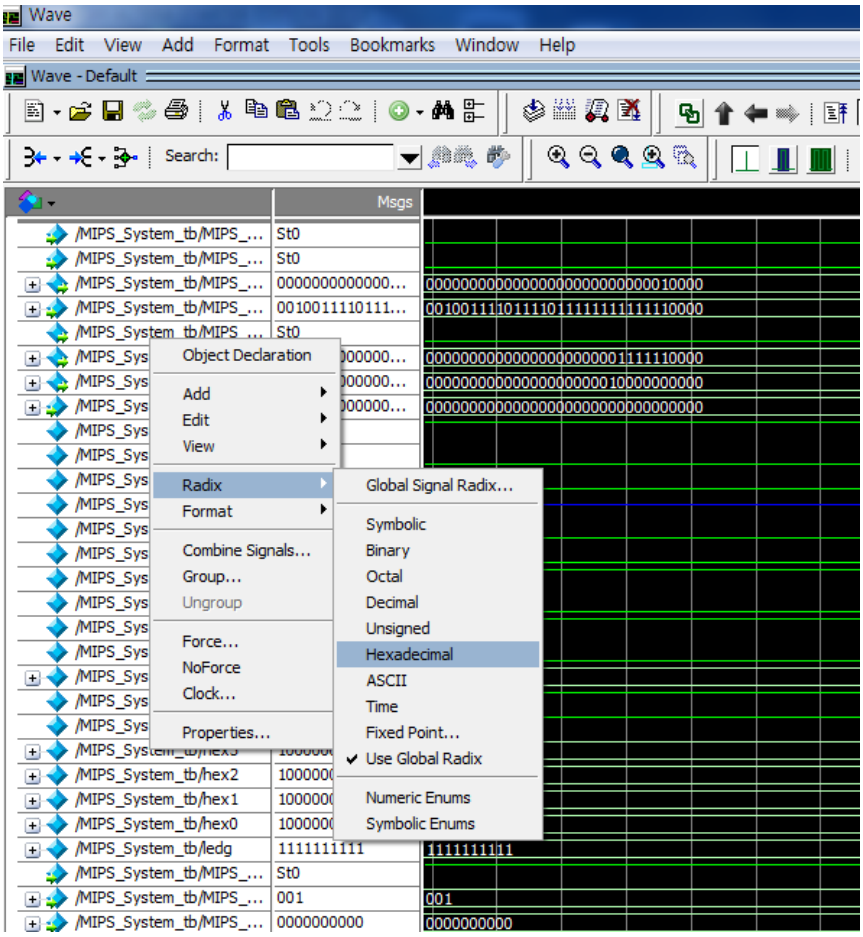
5. Add the signals you want to watch in the waveform
    - Select **mips_cpu** on the left pane
    - Right-click on the **Objects** pane
        - ✓ **Add To → Wave → Signals in Region**



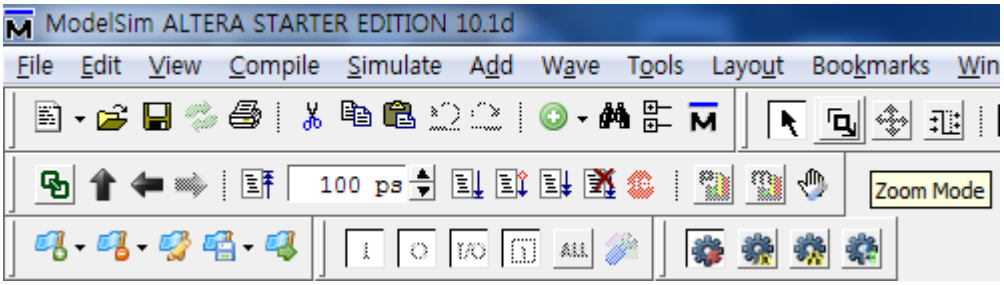
6. Run simulation
   - Type **run 1000ns** in the Transcript pane
     - ✓ It means that the simulation will run for 1000ns

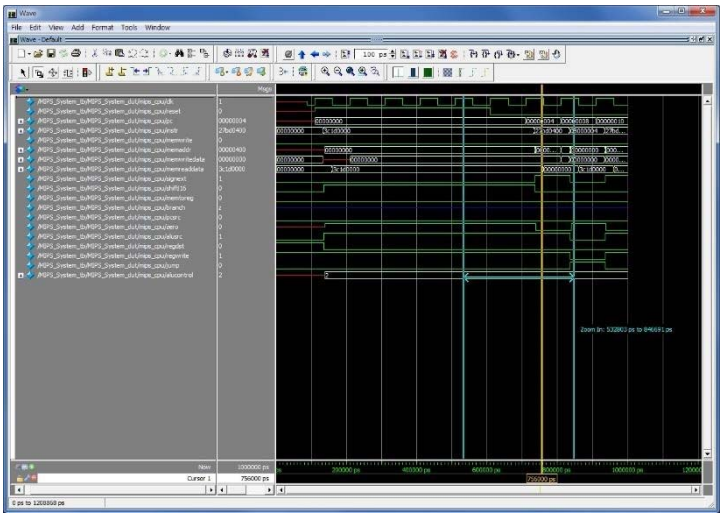


   - Click on the Wave window and observe the signals (such as pc, fetched instructions, control signals, ALU output etc) you want to watch
     - ✓ **View → Zoom → Zoom Full**
     - ✓ Change the radix to Hexadecimal for buses
     - ✓ Run simulation for 1000ns more (type **run 1000ns** in the Transcript pane)

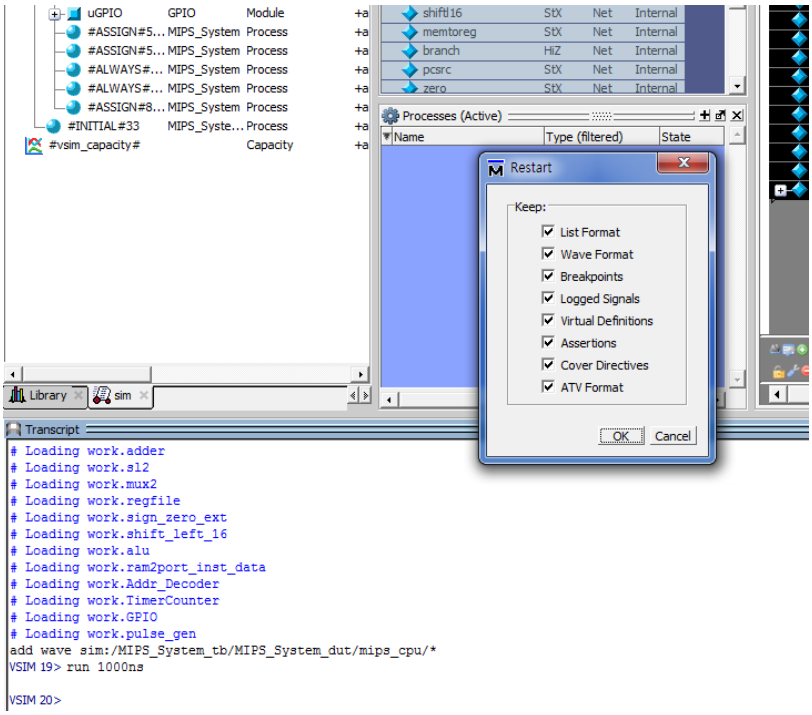


✓Use the Zoom Mode to zoom in the area you are interested in





7. To restart the simulation
    - Type **restart** in the Transcript pane and click on **Restart**
    - Add more signals to the wave if you want to add more signals
        - ✓Go back to step 6 to see how to add signals to the wave
    - Run simulation
        - ✓For example, type **run 2000ns** in the Transcript pane as you did in the step 7



8. To run the simulation of a different program
    - Copy **`insts_data.mif`** you generated with Eclipse to the `MIPS_System_Sim` directory, like you did in the synthesis instruction
    - Run the simulation again: Probably you only need to restart the simulation without exiting the tool.