Homework Assignment 3
(Due Feb. 10th at the beginning of the class)

***** Submission policy: Please zip your source code and waveform screenshots into a single file and send it to daehyun@eecs.wsu.edu. The file name should be *firstname_lastname*.zip (or .tar.gz or .tar …)

(1) **[Synthesis, 10 points]** Click "Labs" in the course webpage and read "tut-dc.pdf". It explains how to run Design Compiler. Then, download the following file into your working directory.

- http://eecs.wsu.edu/~ee434/Homework/hw03.zip
- Unzip it.
  - > unzip hw03.zip
- You will see the following files.
  - dlatch.vhd (VHDL code for D-latches)
  - dlatch_tb.vhd (VHDL code to test D-latches)
  - add8.vhd (VHDL code for 8-bit adders)
  - add8_tb.vhd (VHDL code to test 8-bit adders)
- Synthesize the d-latch code (use the same std. cell library file).
- Save the synthesized netlist into dlatch_syn.v.
- Open dlatch_syn.v in a text editor and see the netlist. It will have a single D-latch cell.
- Run ModelSim, create a new project, and add the following files to the project:
  - dlatch_tb.vhd
  - dlatch_syn.v
  - NangateOpenCellLibrary.v
- Compile them and simulate it (until 12ns). Does the synthesized netlist work well?
- [**Submit**] The synthesized netlist (dlatch_syn.v) and a snapshot of your test waveform.

```
module myDLatch ( D, EN, Y );
  input D, EN;
  output Y;

  DLH_X1 Y_reg ( .G(EN), .D(D), .Q(Y) );
endmodule
```

(2) **[Synthesis, 10 points]** Synthesize the 8-bit adder (add8.vhd) and save the netlist into add8_syn.v. Run ModelSim, create a new project (or you can just open an existing project), and add (add8_syn.v, add8_tb.vhd, NangateOpenCellLibrary.v) to the project. Simulate test_myADD8 until 5ns. Verify the output result against the expected output values in add8_tb.vhd.

- [**Submit**] The synthesized netlist (add8_syn.v) and a snapshot of your test waveform.

```
module myADD8_DW01_add_0 ( A, B, CI, SUM, CO );
  input [8:0] A;
  input [8:0] B;
  output [8:0] SUM;
  input CI;
  output CO;
  wire   n1;
  wire   [8:1] carry;

  FA_X1 U1_7 ( .A(A[7]), .B(B[7]), .CI(carry[7]), .CO(SUM[8]), .S(SUM[7]) );
  FA_X1 U1_6 ( .A(A[6]), .B(B[6]), .CI(carry[6]), .CO(carry[7]), .S(SUM[6]) );
  FA_X1 U1_5 ( .A(A[5]), .B(B[5]), .CI(carry[5]), .CO(carry[6]), .S(SUM[5]) );
  FA_X1 U1_4 ( .A(A[4]), .B(B[4]), .CI(carry[4]), .CO(carry[5]), .S(SUM[4]) );
  FA_X1 U1_3 ( .A(A[3]), .B(B[3]), .CI(carry[3]), .CO(carry[4]), .S(SUM[3]) );
  FA_X1 U1_2 ( .A(A[2]), .B(B[2]), .CI(carry[2]), .CO(carry[3]), .S(SUM[2]) );
  FA_X1 U1_1 ( .A(A[1]), .B(B[1]), .CI(n1), .CO(carry[2]), .S(SUM[1]) );
  AND2_X1 U1 ( .A1(B[0]), .A2(A[0]), .ZN(n1) );
  XOR2_X1 U2 ( .A(B[0]), .B(A[0]), .Z(SUM[0]) );
endmodule


module myADD8 ( A, B, Y );
  input [7:0] A;
  input [7:0] B;
  output [8:0] Y;


  myADD8_DW01_add_0 add_15 ( .A({1'b0, A}), .B({1'b0, B}), .CI(1'b0), .SUM(Y)
        );
endmodule
```