Verilog Homework Assignment

This homework assignment is for those of you who need to get familiar with Verilog HDL and is intended to improve your understanding of the course materials and Verilog coding skills and prepare you to carry out the first project of the course smoothly.

If you have none or little experience with Verilog HDL, you may want to visit the following website and learn some basic points about Verilog before starting this homework. Also, this website is a good reference point for checking the details even if you are coding as an experienced Verilog code developer.

http://www.asic-world.com/verilog/

1) Consider the following chunk of Verilog code. It implements a D-type flip-flop. Answer the following questions regarding the code.

```
***** Declaration part *****
**************************
always @(posedge x)
begin
        w = z;
end

endmodule
```

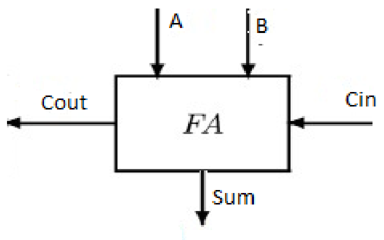
(a) Specify the role of each signal (x, w, and z). Roles are *clock*, *D*, and *Q*.
(b) Complete the declaration of this flip-flop module. Use this chunk in a Verilog module with proper port declaration. Add all the details needed for the code to be compilable. Don't change the name of the signals.

---

2) Provide the Verilog code for a Full Adder, shown below.

Recall that a full adder has 3 inputs: A, B, and Cin; and 2 outputs: Sum and Cout. Implement this full adder using Verilog primitive gates described in the class. Name your module myFA and save it in the "FA.v". Feel free to define new intermediate signals (in the form of wires) as needed but for the ports, your module should only have those 5 mentioned ports. So, module declaration for this entity should look like the following:

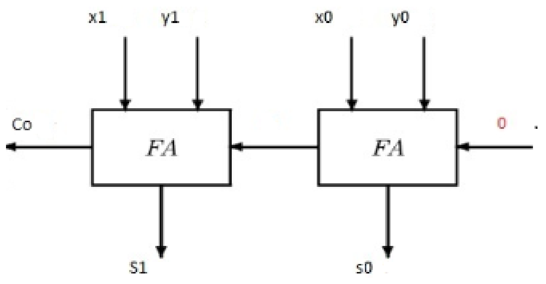
```
module myFA (A, B, Cin, Sum, Cout);
```



3) Provide the Verilog code for a 2-bit Ripple Carry Adder.

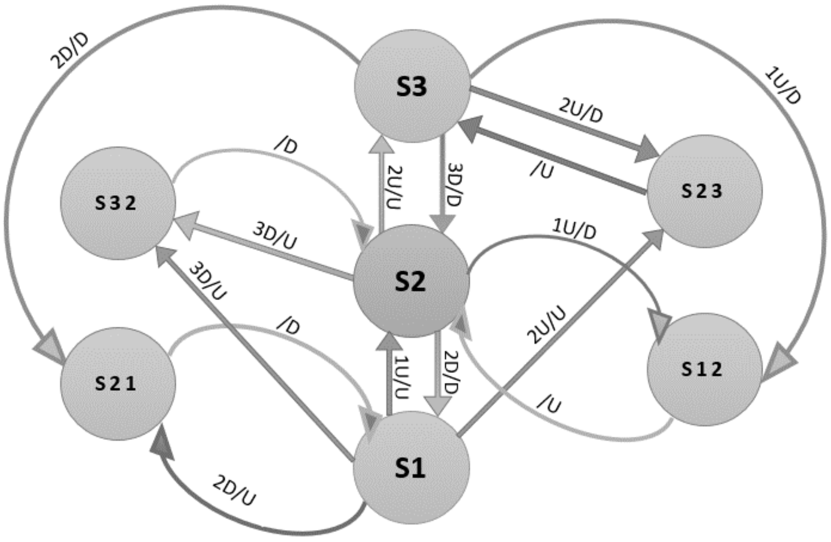
The block diagram of a 2-bit ripple carry adder is shown below. In a new file, describe this structure by instantiating two copies of your myFA modules developed in problem 2. Note that, there is no Cin port in this design and you should connect the Cin port of the first full adder directly to 0. (When instantiating the full adder, in port connection, use 0 for the Cin port.) Name this new module as RippleCarryAdder. It will have 2 input ports: x and y where each of them is a 2-bit wide vector. It will also have 2 output ports: s and Co where s is a 2-bit wide vector and Co is a single bit output. So, the module declaration for this entity should look like the following:

module RippleCarryAdder (x, y, s, Co);



---

4) Write the module declaration and declarations of the input and output variables for the following FSM (just the module and I/O declarations and NOT the whole module). Name the module PrjFSM.



Note: Once you get familiar with the software tools that compile and simulate Verilog HDL designs, you will be able to ensure your code is correct (by compiling it) and write additional Verilog code (called a Testbench for each design) to ensure the correct functionality of your design. You will learn more about these software tools during next week lecture.