UNIVERSITY OF TEHRAN
Electrical and Computer Engineering Department
Object-Oriented Modeling of Electronic Circuits, Spring 1400
Computer Assignment 3, Week 7-8
Logic Minimization and SystemC Simulation

Name:
Date:

The focus of this assignment is on pre-synthesis simulation (in SystemC), synthesis with a SOP synthesis tool that you develop, and post synthesis simulation using the simulation tool that you developed in Computer Assignment 1. The main part is the synthesis tool that we refer to as expression optimization and synthesis tool (EOST). This tool takes Verilog expressions, uses the QM tabular minimization for minimizing the expressions and produces a netlist of gates.

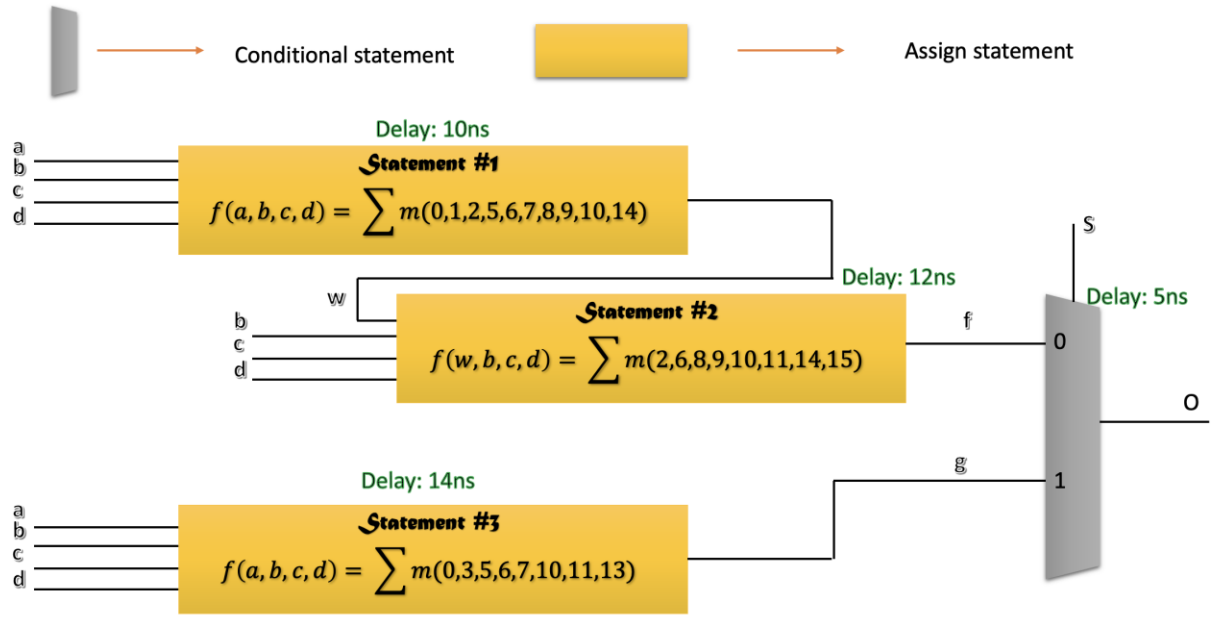
The starting point is a Verilog description that consists of a feedback-free interconnection of Verilog **assign** statements. The LHS of an **assign** statement may be used on the RHS of another, and the order in which the assign statements appear is not necessarily according to the flow of data from circuit inputs to the circuit outputs.

SOP and conditional **assign** statements can be handled by your synthesis tool. The SOP expressions use &, |, and ~ operations of Verilog to describe a two-level AND-OR circuit. Parentheses are not allowed in such expressions. The condition expressions are simple expressions with scalars for the condition and the two choices. Each of which may be LHS of another **assign** statement. No nesting of condition statements needs to be handled.

The SOP type of expressions are to be minimized using the QM method before they are converted to gates. After reading an expression from the input file, if it is recognized as a SOP type expression, it is passed on to the QM implementation procedure, and then the minimized gate-list is generated. This minimization is only limited to individual **assign** statements and does not go across multiple **assign** statements. The conditional **assign** statements translate to simple AND, OR, and NOT gates.

1. Manually translate the input Verilog description to SystemC for pre-synthesis simulation. Perform verification of this description in SystemC using a SystemC testbench as discussed in class.
2. Write a C++ program for implementation of EOST. This program generates a netlist with the format discussed in Computer Assignment 1.
3. Manually insert delay values in your output netlist (Part 2) and perform post-synthesis simulation using your gate-level simulator. Use #5, #5, and #3 for AND, OR, and NOT gates, respectively.
4. Compare simulation results of Part 1 and Part 3.
5.

Use the description shown below as an input for your toolset.




```
module main(input a, b, c, d, s, output o);

assign #10 w = ~a&~b&~c&~d | ~a&~b&~c&d | ~a&~b&c&~d | ~a&b&~c&d | ~a&b&c&~d | ~a&b&c&d | a&~b&~c&~d | a&~b&~c&d | a&~b&c&~d|
               a&b&c&~d;

assign #12 f = ~w&~b&c&~d | ~w&b&c&~d | w&~b&~c&~d | w&~b&~c&d | w&~b&c&~d | w&~b&c&d | w&b&c&~d | w&b&c&d;

assign #14 g = ~a&~b&~c&~d | ~a&~b&c&d | ~a&b&~c&d | ~a&b&c&~d | ~a&b&c&d | a&~b&c&~d | a&~b&c&d | a&b&~c&d;

assign #5 o = s ? g : f;

endmodule;
```