Assignment #2
EECS 303: Advanced Digital Logic Design

Problem 0. Work through the Xcelium Tutorial to complete the RTL simulation of the example design of the conventional Arithmetic Logic Unit (alu_conv.v) provided in the Assignment folder, using the testbench file (alu_conv_test.v) also provided in the Assignment folder. Then, work through the Genus Tutorial using the same alu design.

Problem 1. Minimizing Two-level Logic Functions Using the QM Method (30 Points):

Simplify the following Boolean functions using the Quine-McCluskey algorithm. You need to (1) find out all the prime implicants (2) find out a minimum cover using techniques from the class: remember to first attempt to reduce the covering table by trying to identify essential columns and using row/column elimination rules. If a cyclic core is obtained, then use branch and bound)

(1) f (a, b, c, d) = S (0, 5, 6, 10, 11, 13)+d(4, 8, 14)

(2) f (a, b, c) = S (0, 2, 3, 4, 5, 7)

Problem 2. Value Representations: (20 Points) For each of the following fill the table with the corresponding information:

	Signed Or Unsigned	Width	Base: Decimal, Binary, Octal, Hexadecimal, etc.	Equivalent Integer Value
8'b00001000				
16'hABCD				
4'sb1110				
4'd12				
4'b1100				
8'shFF				
8'sb00011100				
12'h2A8				
6'sb111101				
12'o3456				

Problem 3 Verilog Module Statements (30 Points):

Write a Verilog module in RTL style for the combinational circuit in the figure shown below. Write a testbench and simulate the Verilog code using Excelium for all input combinations to match the truth table shown below. Include all Verilog codes and screenshots of the simulation output.

Inputs			Outputs			
A2	A1	A0	D3	D2	D1	D0
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Table 6-1
Truth table for a 3-input, 4-output combinational logic function.

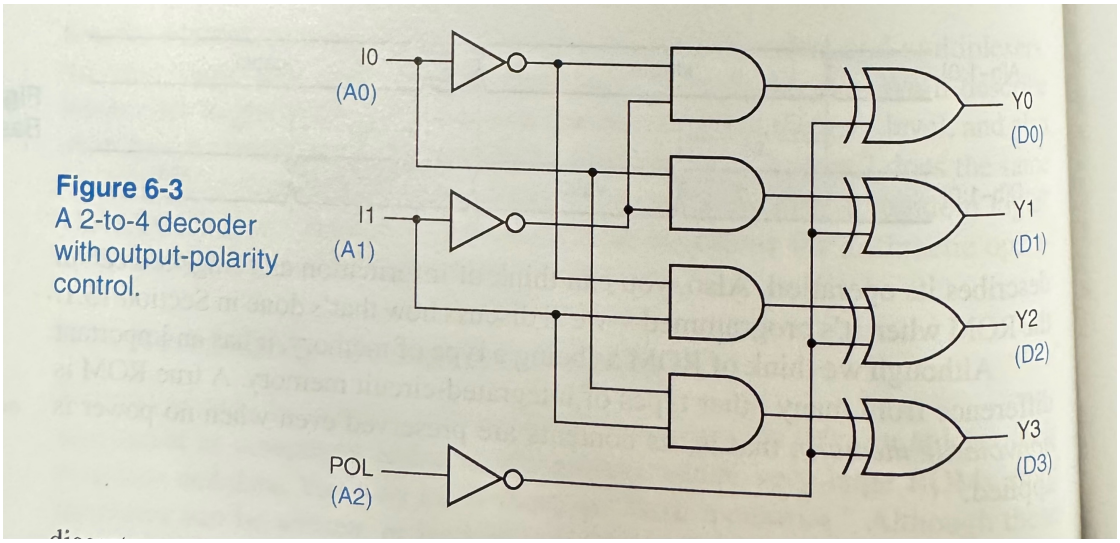

Figure 6-3
A 2-to-4 decoder with output-polarity control.

Problem 4 Verilog Module Statements (20 Points):

Write RTL-style Verilog module corresponding to the full adder circuit in Figure 2.1. Then, instantiate multiple copies of it to create a structural model for a 4-bit ripple carry adder structure of Figure 2.2. Create a testbench for the adder, apply the following input combinations

A = 1, B = 3
A = 11, B = 9
A = 7, B = 13
A = 15, B = 1

Attach all codes and screenshots of the simulation showing correct behavior.

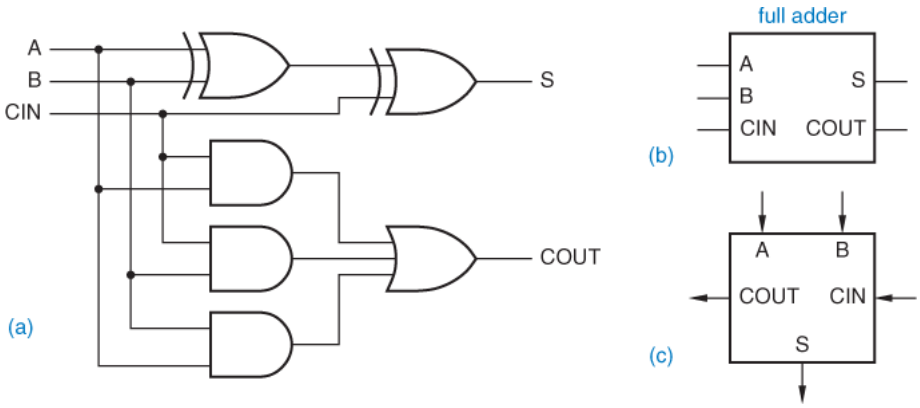

Figure 2-1 Full adder: (a) gate-level logic diagram; (b) logic symbol; (c) alternate logic symbol suitable for cascading.

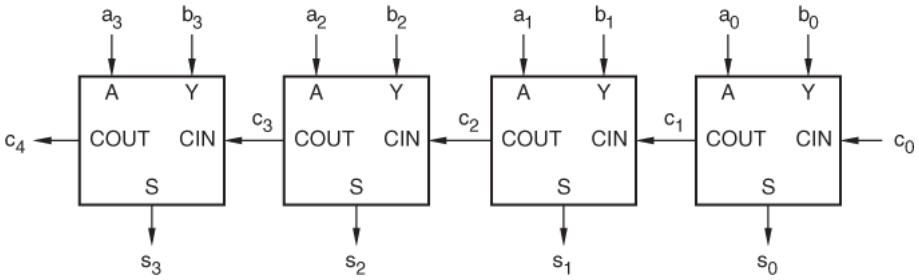

Figure 2-2 A 4-bit ripple adder.