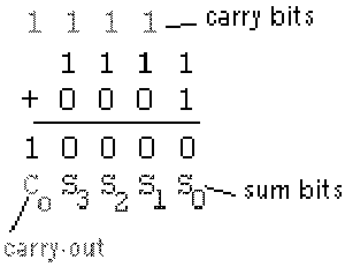
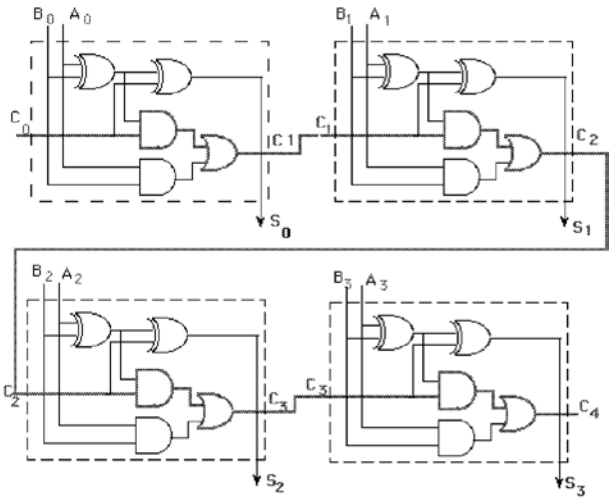
EECS 318 HW1 ***Due: Sept 24***

Problem 1 (20 points): One type of circuit where the effect of gate delays is particularly important, is an ADDER. In this homework you are ask to design and measure the delay of two different types of adder circuits; the 4-bit ripple-carry adder and the Carry Look Ahead adders. In a ripple-carry adder the result of an addition of two bits depends on the carry generated by the addition of the previous two bits. Thus, the Sum of the most significant bit is only available after the carry signal has rippled through the adder from the least significant stage to the most significant stage. This can be easily understood if one considers the addition of the two 4-bit words: $111_2 + 0001_2$, as shown in the figure.

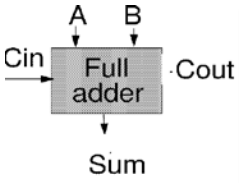


In this case, the addition of ($1+1 = 10_2$) in the least significant stage causes a carry bit to be generated. This carry bit will consequently generate another carry bit in the next stage, and so on, until the final carry-out bit appears at the output. This requires the signal to travel (ripple) through all the stages of the adder as illustrated in the Figure below. As a result, the final Sum and Carry bits will be valid after a considerable delay. The carry-out bit of the first stage will be valid after 4 gate delays (2 associated with the XOR gate and 1 each associated with the AND and OR gates). From the schematic of Shown below, one finds that the next carry-out (C_2) will be valid after an additional 2 gate delays(associated with the AND and OR gates) for a total of 6 gate delays. In general the carry-out of an N-bit adder will be valid after 2N+2 gate delays. The Sum bit will be valid an additional 2 gate delays after the carry-in signal. Thus the sum of the most significant bit S_{N-1} will be valid after 2(N-1) + 2 +2 = 2N +2 gate delays. This delay may be in addition to any delays associated with interconnections.



a. Design an 8-bit ripple-carry adder, describe your design in structural Verilog (A gate delay is 10).
b. Derive an input which exhibit the largest delay on the most significant bit (explain your reasoning) and find the actual delay for the derived input.
c. Derive a test-bench and verify your result using ModelSim.

Problem 2 (20 points): In order to overcome the propagation delay, several approaches including the 'look ahead' carry have been developed. In a typical structure the adder is divided into several bit-slices. Each slice produces its own sum and ``Propagate'' (P) = A ⊗ B , ``Generate'' (G) = A B signals for use in the ``lookahead'' unit.



A	*B*	C_i	*S*	C_o	*Carry status*
0	0	0	0	0	delete
0	0	1	1	0	delete
0	1	0	1	0	propagate
0	1	1	0	1	propagate
1	0	0	1	0	propagate
1	0	1	0	1	propagate
1	1	0	0	1	generate
1	1	1	1	1	generate

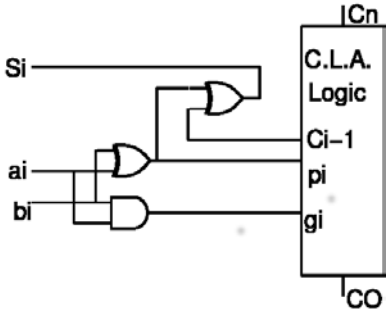
The function of carry generate signal is to indicate that a carry was generated internally by this stage. Carry propagate indicates that if carry existed at the carry-in, it would propagate across that stage. The signals associated with the adder are:

``generate'' signal $g_i = a_i \cdot b_i$

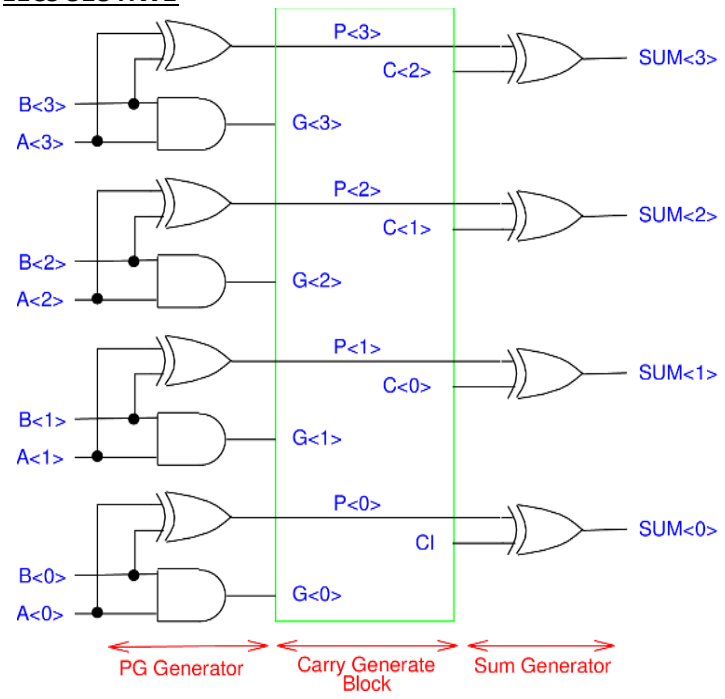
``propagate'' signal $p_i = a_i \otimes b_i$

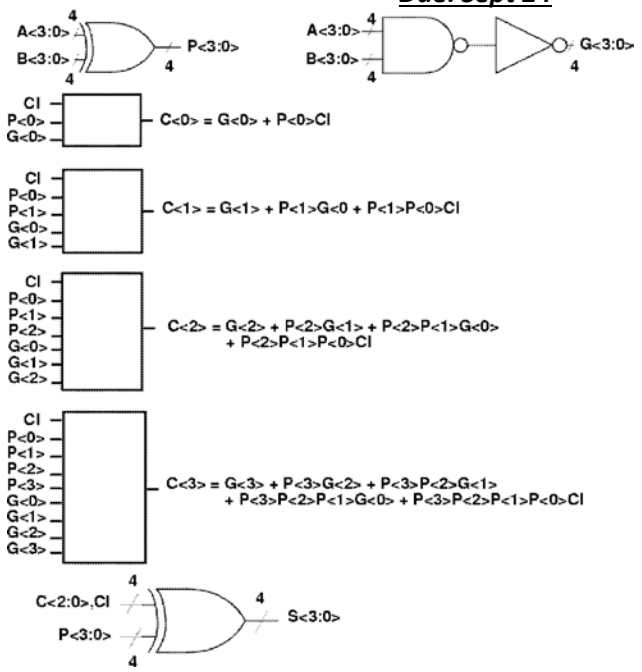
``carry'' signal $C_i = (C_0 \cdot p_0 \cdot p_1 \ldots\ldots p_i) + (g_0 \cdot p_1 \cdot p_2 \ldots p_i) + (g_1 \cdot p_2 \cdot p_3 \ldots p_i) + (g_{i-1}\, p_i) + g_i$

`sum'' signal $S_i = C_{i-1} \otimes a_i \otimes b_i = C_{i-1} \otimes p_i$



A 4-bit CLA adder is shown below:





a. Design an 4-bit Carry-Look-ahead CLA adder and derive a gate-level model for your design. Develop a Verilog model for the gate-level design (gate delay is 10).

b. Derive an input which exhibits the largest delay on the most significant bit and find the actual delay for the derived inputs.

c. Use ModelSim to verify your computation.

Problem 3(20 points): Multiplication of signed operands, which generate a double-length product in the 2's-complement number system. The general strategy is the accumulated partial products by adding versions of the multiplicand as selected by the multiplier bits. First, consider the case of a positive multiplier and a negative multiplicand, when we add a negative multiplicand to a partial product, we must extend the sign-bit value of the multiplicand to the left as far as the product will extend (as shown in the figure below). The sign extension of the multiplicand is hand-written. For a negative multiplier, a straightforward solution is to form the 2's complement of both multiplier and multiplicand and proceed as in the first case.

					1	0	0	1	1	MULTIPLICAND (-13)
					0	1	0	1	1	MULTIPLIER (+11)
	1	**1**	**1**	**1**	1	0	0	1	1	
	1	**1**	**1**	1	0	0	1	1		
	0	**0**	0	0	0	0	0			
	1	1	0	0	1	1				
	0	0	0	0	0					
1	1	0	1	1	1	0	0	0	1	PRODUCT

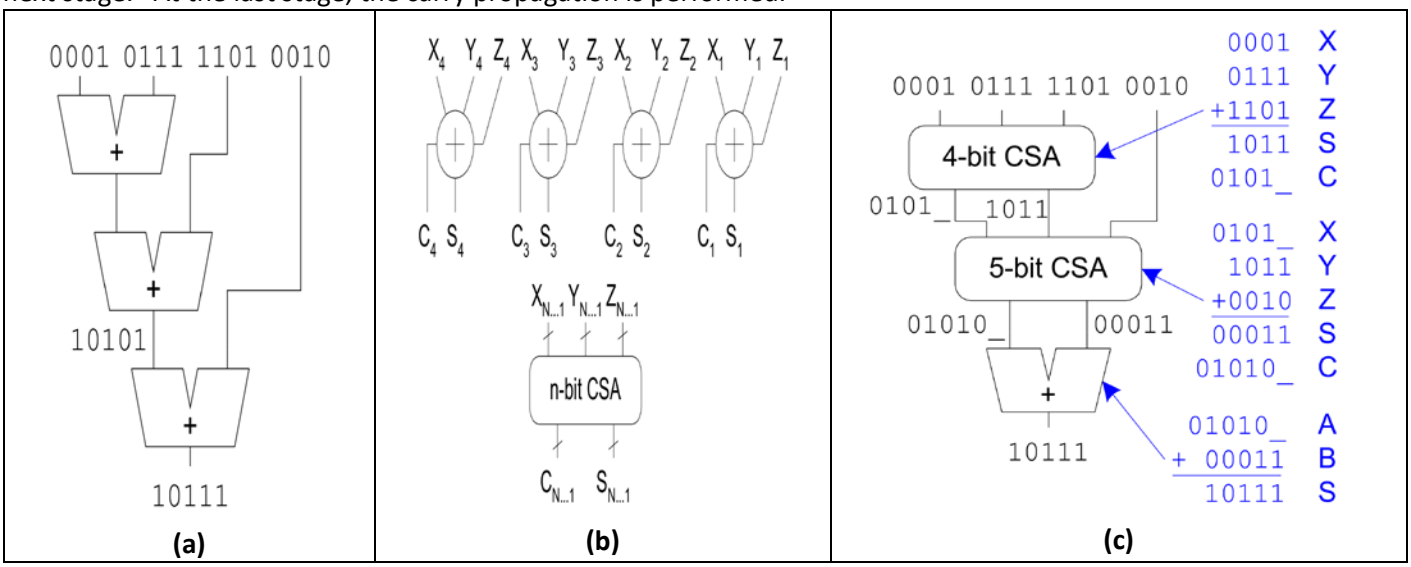
a) Write a Structural Verilog design to model the above behavior for 5 bits signed multiplier.

b) Use Modelsim to simulate your multiplier with the following cases: (-10*4, 11*-3, -10*-11)

Problem 4(20 points): To add k N-bit words, you need k-1 N-bit adder. For example to add 0001 + 0111 + 1101 + 0010 = 10111 you need a structure similar the one shown below (a). Classical full adder sums 3 inputs to produce 2 outputs Carry (C) and Sum (S) where the C has twice the weight of the sum output. If you design N full adder in parallel shown in (b), then this produces N Sums and N Carry outs. This is called Carry Save Adder (CSA).

EECS 318 HW1 ***Due: Sept 24***

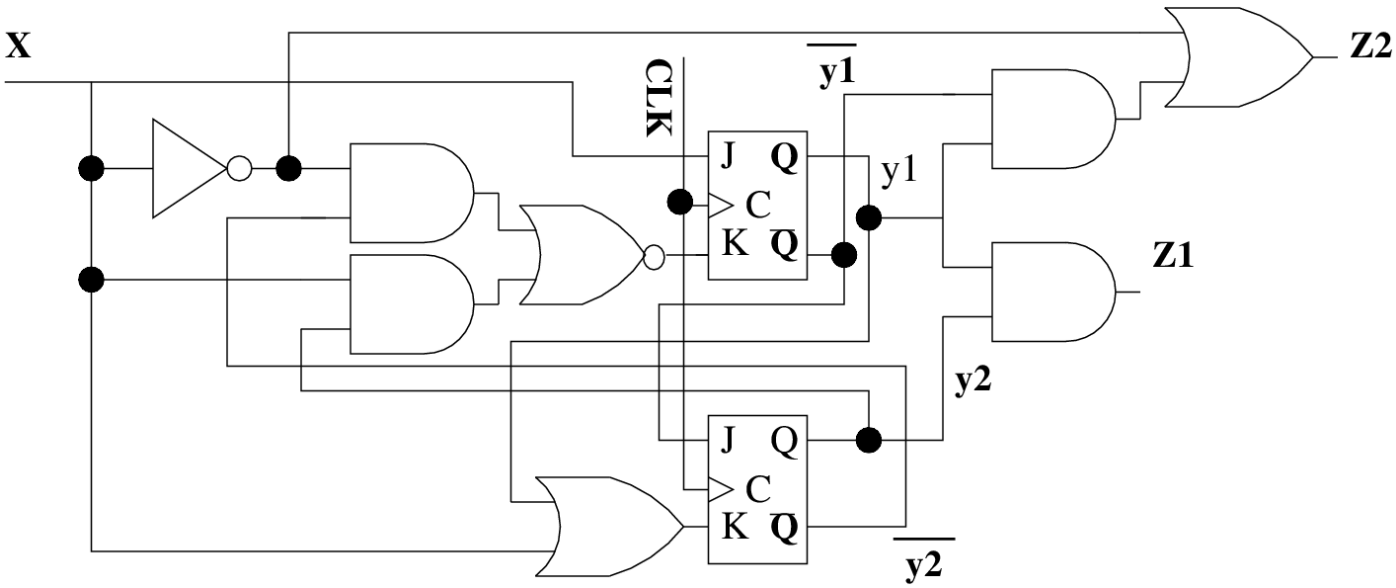
To add N-bit words, a 2-stage of CSA followed by a regular adder as shown. In this structure, the carry bits are logically shifted by one bit after each stage to reflect the weight of the carry. The size of the CSA is increased by a single bit at the next stage. At the last stage, the carry propagation is performed.



a) Design a CSA to perform add a sequence of 10 8-bit binary numbers. How many CSA stages are needed (explain). Write a Verilog model for your design.

b) Use Modelsim to simulate your design with the following sequence: (11,2,13,4,5,6,7,8,9,10) and (3,14,5,6,7,8,19,10).

Problem 5(20 points): Analyze the logic circuit given below.



a) Derive a state diagram for the circuit.

b) Write a structural Verilog Model for this design and simulate your circuit to verify correctness (Explain).

c) Write a behavioral Verilog model based on the derived state-diagram and simulate to verify correctness.

d) Using ModelSim, can you verify that the model in a) and c) are equivalent (Explain).