EECS 150, Midterm 1, Fall 2005, Professor Randy Katz

Question 1. *Combinational Logic Design (10 Points)*

Your task is to design a combinational logic system to convert a four-bit "sign and magnitude number" ($SM_2M_1M_0$) to a twos complement number ($T_3T_2T_1T_0$).

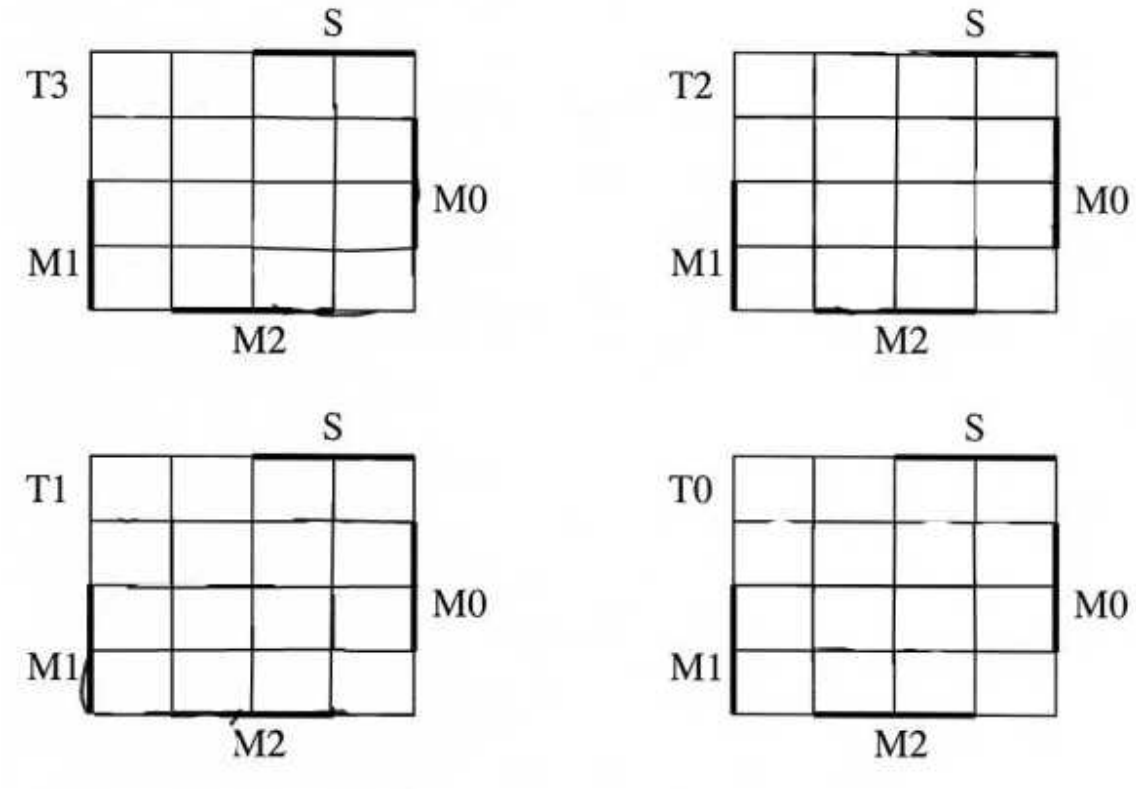
For example, 0111_2, 1111_2 are the representations of +7 and -7 in sign and magnitude from respectively. In twos complement form, positive numbers are represented just like the sign and magnitude scheme, but the negative numbers are formed as follows. Take the positive form, complement the bits, and add 1. Thus, the representations for -7 in twos complement is formed as follows: 0111_2 -> 1000_2 -> +1 = 1001_2.

Note that sign and magnitude form has two representations for zero (1000_2 and 0000_2) while twos complement has only one zero (0000_2).

(a) To make sure that you understand the function to be implemented, complete the following truth table (2 points).

	S	M_2	M_1	M_0	T_3	T_2	T_1	T_0
+0	0	0	0	0	0	0	0	0
+1	0	0	0	1				
+2	0	0	1	0				
+3	0	0	1	1				
+4	0	1	0	0				
+5	0	1	0	1				
+6	0	1	1	0				
+7	0	1	1	1				
-0	1	0	0	0				
-1	1	0	0	1				
-2	1	0	1	0				
-3	1	0	1	1				
-4	1	1	0	0				
-5	1	1	0	1				
-6	1	1	1	0				
-7	1	1	1	1	1	0	0	1

(b) Fill in each of the four k-maps and minimize for sum of products implementation (8 points).



T_3 =

T_2 =

T_1 =

T_0 =

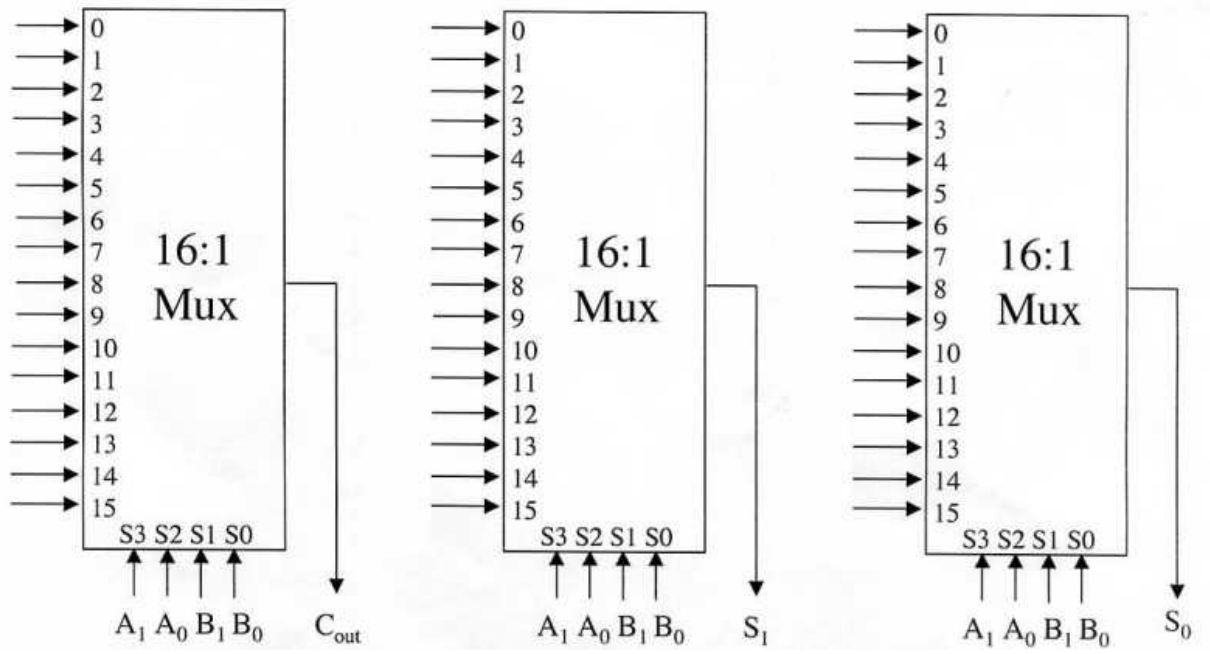
Question 2. *Combinational Logic Design (10 Points)*

Your task is to implement a "two x two adder" using three 16:1 multiplexers. The inputs are two two-bit magnitudes (no sign!) A_1A_0 and B_1B_0 and the carry in C_{in}. The adder generates a two-bit sum output S_1S_0 and a carry out C_{out}.

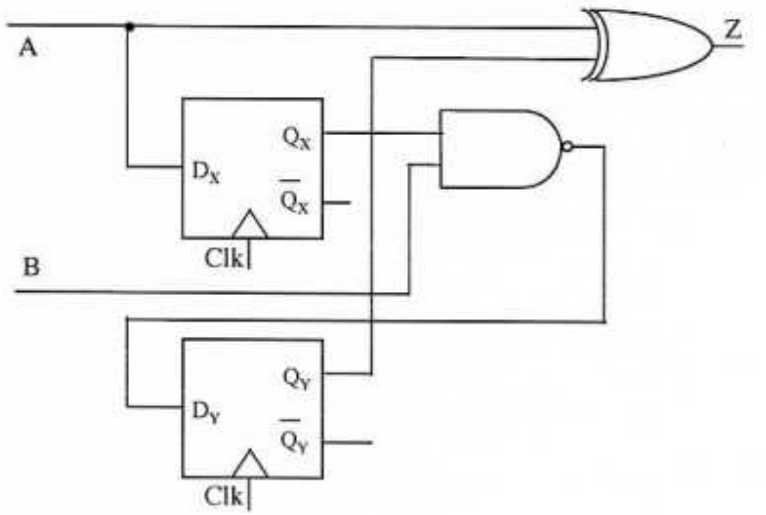
(a) Complete the truth table (4 points):

A_1	A_0	B_1	B_0	C_{in}	C_{out}	S_1	S_0
0	0	0	0	0			
0	0	0	0	1			
0	0	0	1	0			
0	0	0	1	1			
0	0	1	0	0			
0	0	1	0	1			
0	0	1	1	0			
0	0	1	1	1			
0	1	0	0	0			
0	1	0	0	1			
0	1	0	1	0			
0	1	0	1	1			
0	1	1	0	0			
0	1	1	0	1			
0	1	1	1	0			
0	1	1	1	1			
1	0	0	0	0			
1	0	0	0	1			
1	0	0	1	0			
1	0	0	1	1			
1	0	1	0	0			
1	0	1	0	1			
1	0	1	1	0			
1	0	1	1	1			
1	1	0	0	0			
1	1	0	0	1			
1	1	0	1	0			
1	1	0	1	1			
1	1	1	0	0			
1	1	1	0	1			
1	1	1	1	0			
1	1	1	1	1			

(b) Finish the wiring of the multiplexers to C_{in}, C_{in}'s complement (write these as C, $\overline{C}$), 0 and 1, in order to implement the functions C_{out}, S_1, and S_0 (6 points):



Question 3. *Reverse Engineering (10 Points)*
Your task is to derive a PORTION of the state diagram associated with the following sequential logic circuit diagram:



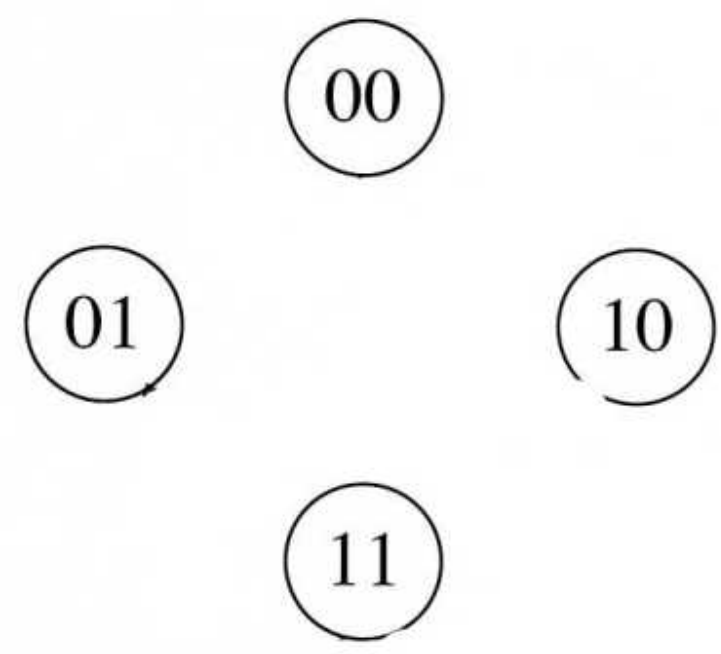
(a) Is this a MEALY MACHINE or a MOORE MACHINE (circle one!) (1 point).

(b) Next, Write Boolean equations for the following circuit nodes (3 points):
$Z\ (A, B, Q_X, Q_Y) =$
$D_X\ (A, B, Q_X, Q_Y) =$
$D_Y\ (A, B, Q_X, Q_Y) =$

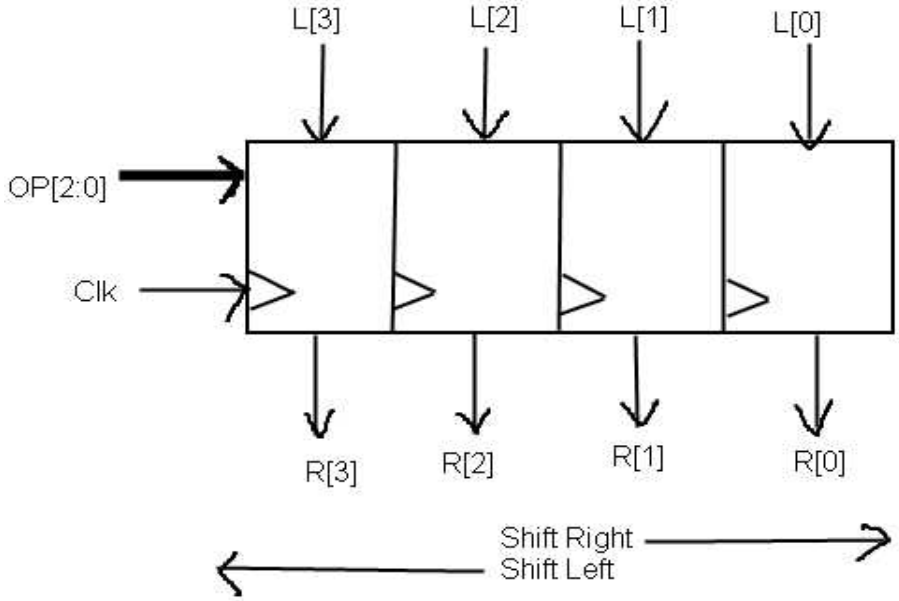
(c) Next fill in the encoded state transition table (2 points):

Q_X	Q_Y	A	B	D_X	D_Y	Z
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

(d) Based on your state transition table in part (c), complete the PORTION of the state diagram FOR THOSE TRANSITION ORIGINATING IN STATE 11 ONLY (i.e., there is no need to include transitions FROM states 00, 01, 10) (4 points):



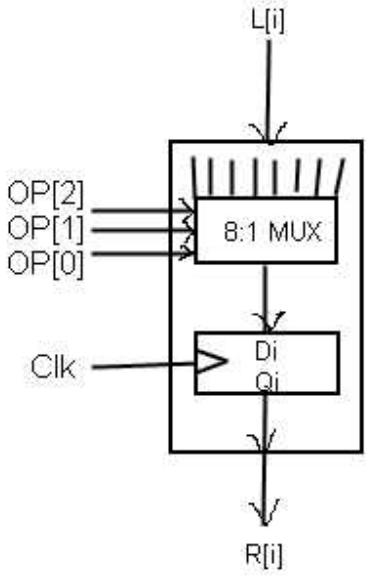
Question 4. *Verilog Specification (10 Points)*
You are to write partial behavioral Verilog for a four-bit shifter subsystem to the following specification. The subsystem has four load inputs L[3:0], four register outputs R[3:0], a clock CLK, and a 3-bt operation input OP[2:0]. Here is a block diagram of the subsystem:



OP[2:0] is defined as follows:

000: Hold current value
001: Logical shift right (shift right plus highest bit is filled with zero)
010: Logical shift left (shift left plus lowest bit is filled with zero)
011: Circular shift right (shift right plus lowest bit wraps around to the highest bit)
100: Store zeros in all register bits (reset)
101: Arithmetic shift right (shift right plus highest bit retains its value)
110: Circular shift left (shift left plus highest bit wraps around to the lowest bit)
111: Load register from inputs L[3:0]

Internally, a bit slice looks like this:



Your job is to write the portion of the behavioral Verilog that determines the correct inputs to the multiplexer.

Fill in the following behavioral Verilog fragments:

(a) The high order bit slice D[3] (4 points):

```
always @(posedge clk)
   case (OP)
      3'b000: D[3] <=
      3'b001: D[3] <=
      3'b010: D[3] <=
      3'b011: D[3] <=
      3'b100: D[3] <=
      3'b101: D[3] <=
      3'b110: D[3] <=
      3'b111: D[3] <=
```

(b) The low order bit slice D[0] (3 points):

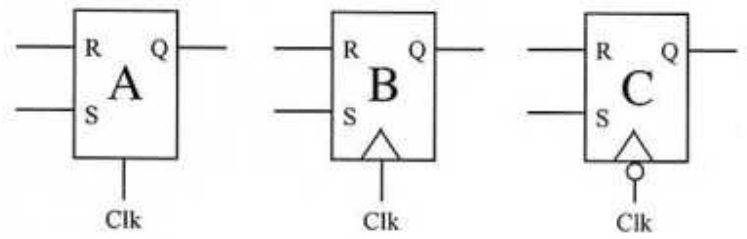
```
always @(posedge clk)
   case (OP)
      3'b000: D[0] <=
      3'b001: D[0] <=
      3'b010: D[0] <=
      3'b011: D[0] <=
      3'b100: D[0] <=
      3'b101: D[0] <=
      3'b110: D[0] <=
      3'b111: D[0] <=
```

(c) A middle bit slice D[i]. Write you answer in terms of D[i], D[i+1], and D[i-1]. (3 points)

```
always @(posedge clk)
   case (OP)
      3'b000: D[i] <=
      3'b001: D[i] <=
      3'b010: D[i] <=
      3'b011: D[i] <=
      3'b100: D[i] <=
      3'b101: D[i] <=
      3'b110: D[i] <=
      3'b111: D[i] <=
```

Question 5. *Latch vs. Edge-Triggered Storage Element Behavior (10 points)*

Device A is a clock-level sensitive R-S latch (i.e., it reacts to the R-S inputs only when the clock is high). Device B is an R-S Flip-Flop that is positive edge triggered. Device C is an R-S flip-flop that is negative edge triggered.



Assume 0 set-up and hold times, and 0 propagation delays. All devices treat R and S as active high signals (i.e., Reset when R is true and Set when S is true), and are implemented using NOR gates. Initially the devices have 0 stored in them.

Complete the timing diagram below for the signals Q_A, Q_B, Q_C, showing the behavior of the three different device to the same R and S input changes: