CPRE 281: Digital Logic
Extra Credit Homework (Optional)
Due: Friday Dec 11, 2015

8 problems x 0.5% each = 4%

What to submit: Submit your solutions on paper. Please write legibly and staple all your pages. Put you name, student ID number, and lab section number on the first page.

For all problems except 8, please use positive-edge triggered memory elements. Please provide the state diagram, state table, the state-assigned table, and the final implementation of the sequential circuit (and clearly label each part). Also, please provide the K-maps and the logic expressions for the next state logic and for the output logic. Please use the same naming conventions that were used in the class slides. In other words, use y_1 y_0 for current state, Y_1Y_0 for next state, z_1z_0 for outputs, and w_0 w_1 for inputs, just to avoid confusion and to make it easier for grading.

1. Inputs w_0 and w_1 provide two bit streams (the first is formed by the bits on w_0 and the second one by the bits on w_1 at every positive clock edge) to a Moore sequential machine. Design the machine to detect if the number represented by the stream of bits, from the first clock edge (LSB) to the current clock edge (MSB), on input w_0 is equal to, less than, or greater than the number on input w_1. The outputs should be set as follows: $z_1z_0 = 11$ if $w_1 = w_0$, $z_1z_0 = 01$ if $w_0 < w_1$, and $z_1z_0 = 00$ if $w_0 > w_1$.

   Example

   Clock

Clock	Initial	1	2	3	4	5	6	7	8
w_0	0	0	1	1	0	1	1	0	
w_1	0	0	1	1	1	0	1	1	
z_0	x	1	1	1	1	1	0	0	1
z_1	x	1	1	1	1	0	0	0	0
			$w_0 = w_1$	$w_0 = w_1$	$w_0 = w_1$	$w_0 < w_1$	$w_0 > w_1$	$w_0 > w_1$	$w_0 < w_1$

2. Input w_0 provides a stream of bits (with LSB first, one bit per clock edge) to a Moore sequential machine. Design the machine to output the 2's complement of the input number until the current active clock edge.

   Example

   Clock

Clock	Initial	1	2	3	4	5	6	7	8
w_0	0	0	1	1	0	1	1		
z_0	x	0	0	1	0	1	0	0	

3. Design and implement a Moore machine that detects the pattern 011 in its 1-bit serial input stream. Explain the logic behind your solution.

4. Design and implement Moore machine that detects the pattern 1001 in its 1-bit serial input stream. Explain the logic behind your solution.

5. Input w_0 provides a stream of 3-bit binary numbers (with LSB first, one bit per clock edge) to a Mealy sequential machine. Design the machine to output a 1 if and only if each 3-bit number is a ***Palindrome*** (Reads the same from LSB to MSB or vice versa. For example, 101 and 111 are Palindromes).

   Example

   Clock

Clock	1	2	3	4	5	6	7	8	9
w_0	0	0	1	1	0	1	0	0	0
z_0	0	0	0	0	0	1	0	0	1

6. Repeat problem 5, but assume that the 3-bit numbers ***overlap*** in successive clock cycles. Design the Mealy machine to output a 1 if and only if each overlapped 3-bit number is a Palindrome (Reads the same from LSB to MSB or vice versa. For example, 101 and 111 are Palindromes).

   Example

   Clock

Clock	1	2	3	4	5	6	7	8	9
w_0	0	0	1	0	1	0	1	0	0
z_0	0	0	0	1	1	1	1	1	0

7. Design and implement an Up/Down counter using a FSM model. This counter will count up (0, 1, 2, 3) if the input w is 0, and count down (0, 3, 2, 1) if the input is 1. In both cases, the counter will continuously cycle through the count without resetting it. Suppose that the counter is currently 2 and the counter was counting up and then switched to count down. Then after the next clock the count should be equal to 1. Use D flip--flops for the device's memory.

8. Design a circuit that can multiply an eight-bit number A=a7, a6, a5, a4, a3, a2, a1, a0 by either 1, 2, 3, or 4 to produce the result A, 2A, 3A, or 4A, respectively.

   Hint: Use a 10-bit adder (you can assume that you have that component so you don't have to design it), and multiple 4-to-1 multiplexers. The select lines of the multiplexers should select the multiplication factor. For example s1,s0 = 0,0 specifies multiplication by 1; s1,s0 = 1,1 specifies multiplication by 4. And so on. Hardcode the appropriate lines of the adder and the multiplexers to 0 or 1 as needed. You may also need some additional logic gates.

   Hint: You may find the example shown in Figure 3.6b in your textbook interesting.