

CSE 140 Homework Four

February 27, 2008

*Only **Problem Set Two** will be graded. Turn in only **Problem Set Two** which will be due on March 12, 2008 (Wednesday) at 11:00am.*

1 Problem Set One

- textbook 5.3
- textbook 5.9
- textbook 5.10
- textbook 5.13(c)
- textbook 5.16
- textbook 5.17(a)(c)
- textbook 6.2
- textbook 6.6
- textbook 6.10
- textbook 6.11
- textbook 6.13
- textbook 6.14
- textbook 6.19
- textbook 6.20

2 Problem Set Two

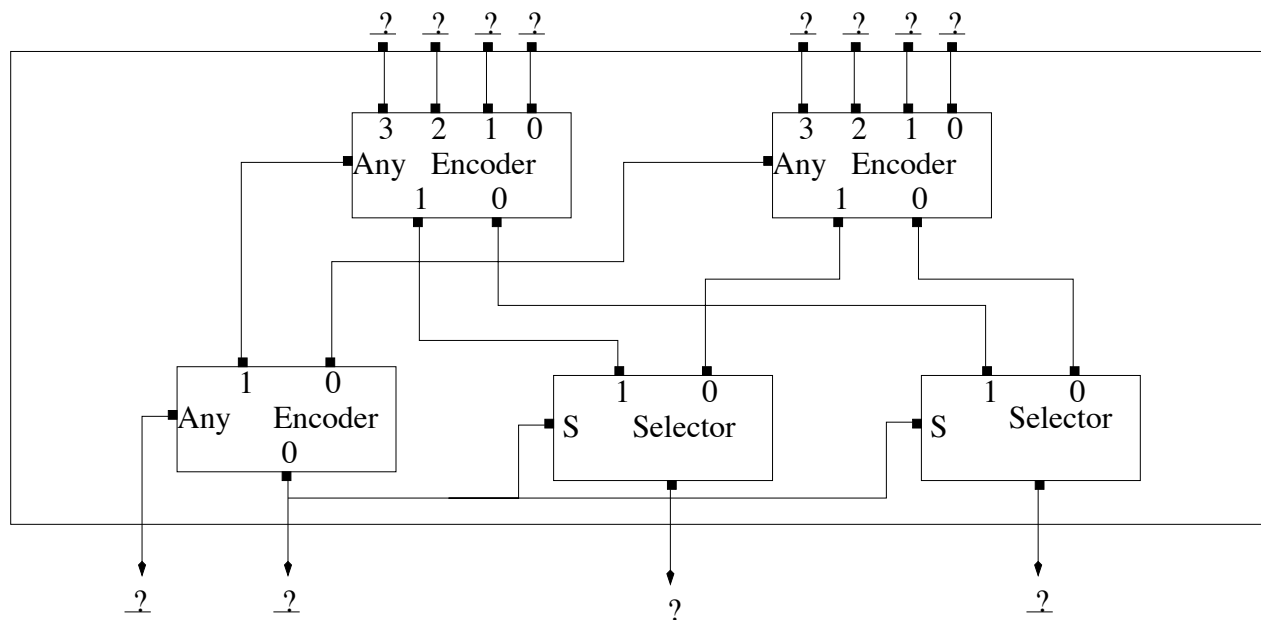
1 (Priority Encoders)

The priority encoder that has been discussed in class implements an increasing priority; when both D_i and D_j are on, where $i > j$, it returns i . In this question, you are asked to implement a new priority scheme where the inputs with odd indices have priority over the ones with even indices. When there is a tie, the larger index gets the higher priority. For example, if only D_3 and D_6 are on, the input with odd index, D_3 , has the higher priority and 3 is returned as the encoder output. If only D_6 and D_4 are on, the input with the larger index, D_6 , has the higher priority and 6 is returned as the encoder output.

(PART A) Please fill in the missing entries in the following table. You can (and should) use *don't cares* for some of the D_i 's in order to represent the requested case fully.

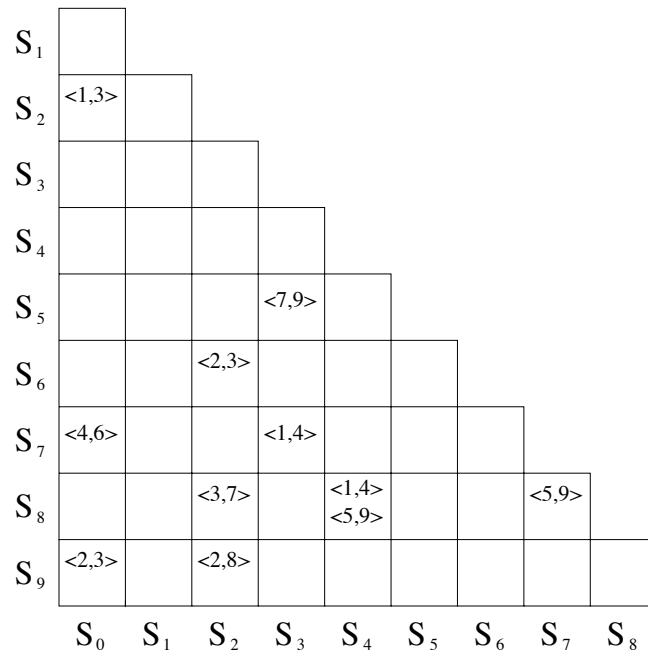
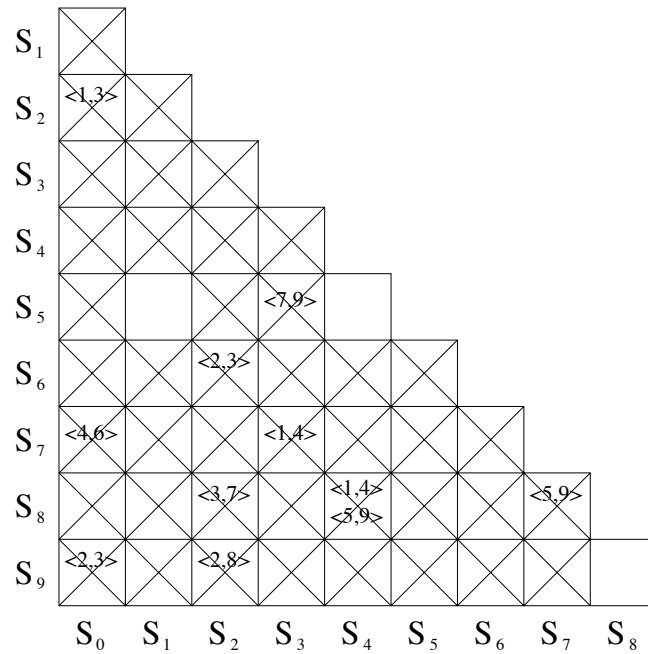
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0	ANY
								0	0	0	0
								0	1	1	
								1	0	0	
			1					0	1		
	0							1	1		

(PART B) An implementation of an odd-even priority encoder is given in the following graph by using the 4-to-2 and 2-to-1 increasing priority encoders and 2-to-1 selectors. In the implementation given below, the input and output signals are missing. Please label the inputs and outputs by using $D_7, D_6, D_5, D_4, D_3, D_2, D_1, D_0$ and $A_2, A_1, A_0, \text{ANY}$, respectively.



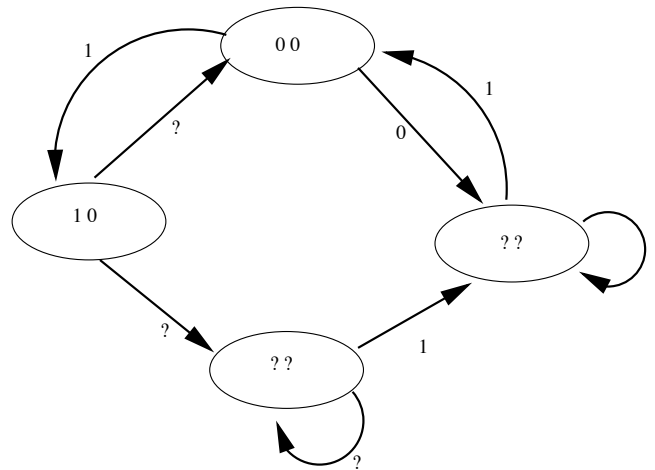
2 (State Minimization)

Our technical staff has been working on the following state minimization problem. In a moment of bleary-eyed, need more coffee time, they made a mistake and put an 'X' where it does not belong. Then they continued filling in the chart. Rather than start from scratch, we would like you to fix the problem on this diagram. Please identify the 'X' that was inadvertently placed and provide a reason for your answer. Next, please correct the ensuing inconsistencies that result from your fix. Finally, tell us the equivalent state classes of the final answer.



3 (Finite State Machines)

Consider the following scenario. An FSM was implemented using T-flip flops using the corresponding next state logic. The actual implementation charts were lost; the only thing that partially survived was the FSM state diagram, together with the next state logic. Only one input exists in this FSM and 2 bits are used to represent all 4 states. Following is the partial FSM state diagram that was obtained by using T flip flops.



As the designers who found the next state logic together with the partial FSM did not know what flip flops the machine originally was designed for, they went ahead and used the recovered next state logic with the D flip flop, as D is the most commonly used flip flop. A smart engineer was able to figure out what the machine corresponding to the original got transformed to when the original next state logic got coupled to the D-flip flops. Can you help him finish his job by filling in the missing parts (denoted by "?") in both diagrams.

Hint: Both machines are using the same “next state logic”. All the T-flip flops from the previous machine have simply been replaced by D flip flops. What effect will this have?

