ECE199 Exam 3, Fall 2012

Tuesday, 13 November

Name and UIUC Net ID:

- **Be sure that your exam booklet has 10 pages.**
- **Write your name at the top of each page.**
- **This is a closed book exam.**
- **You are allowed THREE** 8.5×11**" sheets of notes.**
- **We have provided a scratch sheet at the back.**
- **Absolutely no interaction between students is allowed.**
- **Show all of your work.**
- **Challenge questions are marked with *****
- **Don't panic, and good luck!**

"Hence, about 10^5 flip flops or analogous elements would be required!
This would, or course, be entirely impractical."
—from "Logical Design of an Electronic Computing Instrument"
by A.W. Burkes, H.H. Goldstine, and J. von Neumann, 1946

Problem 1	15 points	_____
Problem 2	15 points	_____
Problem 3	15 points	_____
Problem 4	30 points	_____
Problem 5	25 points	_____
Total	100 points	_____

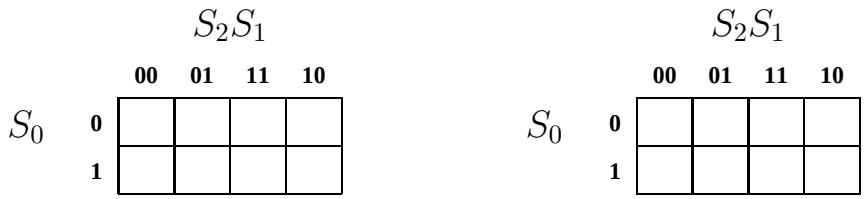
Problem 1 (15 points): Finite State Machine Implementation

Consider the Finite State Machine (FSM) defined by the next-state equations below. The FSM has three internal state bits, $S_2S_1S_0$, and three outputs, $Z_2Z_1Z_0$, that match the internal state bits. (That is, $Z_2 = S_2$, $Z_1 = S_1$, and $Z_0 = S_0$.) Boolean expressions for S_2^+, S_1^+ and S_0^+ are the following:

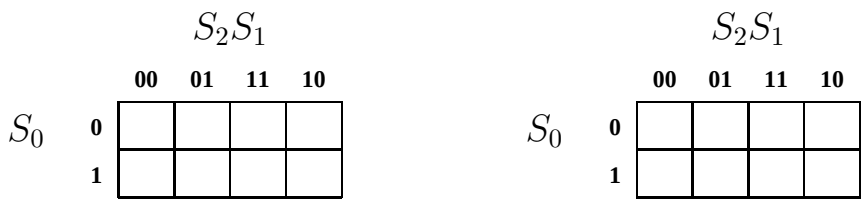
$$\begin{aligned} S_2^+ &= \bar{S}_2\bar{S}_1\bar{S}_0 + S_2S_0 + S_2S_1 \\ S_1^+ &= \bar{S}_1\bar{S}_0 + S_1S_0 \\ S_0^+ &= \bar{S}_0 \end{aligned}$$

Part A (5 points): For each of the next-state variables, S_2^+, S_1^+ and S_0^+, fill in one of the **two** blank K-maps below. **CROSS OUT the THREE K-maps that you DO NOT want graded.**

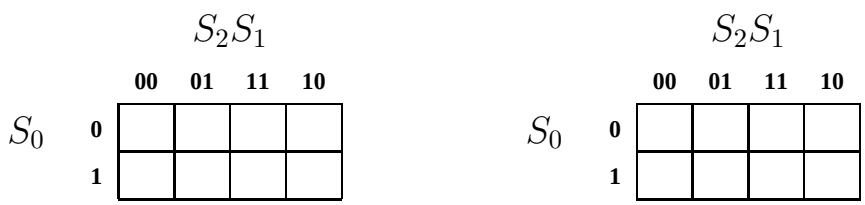
K-maps for S_2^+



K-maps for S_1^+



K-maps for S_0^+



Part B (5 points): Draw the state transition diagram and explain the function of this FSM.

Problem 1, continued:

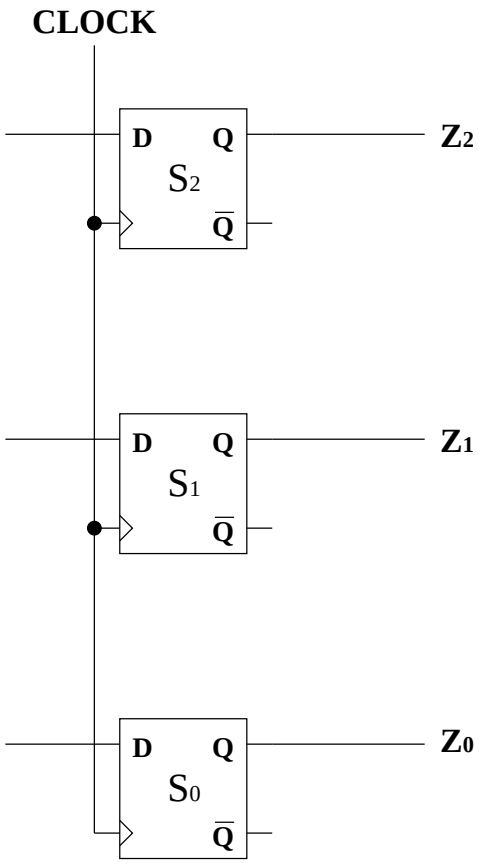
These equations are replicated from the previous page for your convenience.

Boolean expressions for S_2^+, S_1^+ and S_0^+ are the following:

$$\begin{aligned} S_2^+ &= \bar{S}_2\bar{S}_1\bar{S}_0 + S_2S_0 + S_2S_1 \\ S_1^+ &= \bar{S}_1\bar{S}_0 + S_1S_0 \\ S_0^+ &= \bar{S}_0 \end{aligned}$$

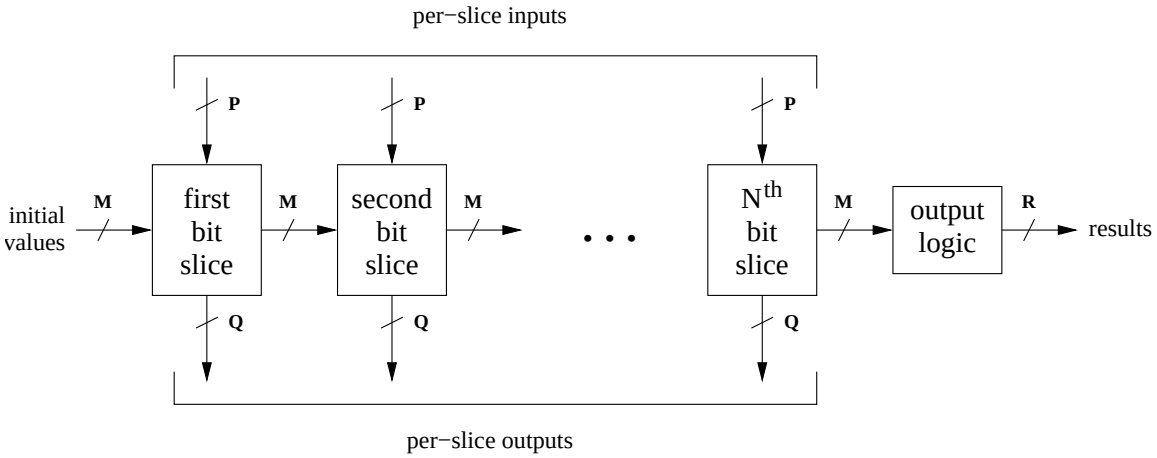
Part C (5 points): Add gates as necessary to complete the implementation the FSM using the three positive edge-triggered D flip-flops shown below.

Please label gate inputs with variable names rather than trying to draw feedback wires from the flip-flops back to the the next-state logic.



Problem 2 (15 points): Serial Designs and Sequence Recognizers

Parts A and **B** pertain to transforming a bit-sliced design into a serial design. The figure shown below represents a general bit-sliced design.

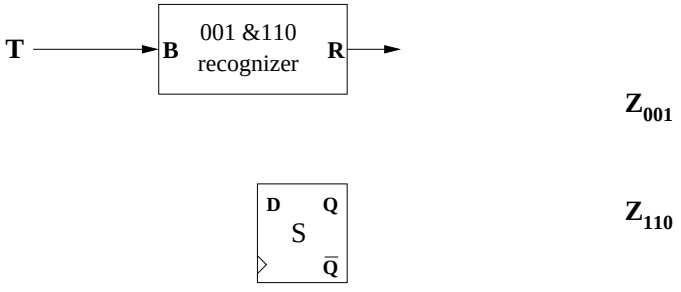


Part A (5 points): In terms of the parameters given in the diagram above, how many flip-flops are needed to construct a serial design (using a Moore machine model)?

Part B (5 points): If you add a counter to control the serial design, how many flip-flops do you need to implement the counter?

Part C (5 points): You are charged with designing FSMs to recognize the bit sequences 001 and 110. Fortunately, you find a component that recognizes both sequences. The component is the box marked "001 & 110 recognizer" in the diagram below. This component outputs $R = 1$ if and only if the B input for the last three cycles matches one of the two recognized bit patterns.

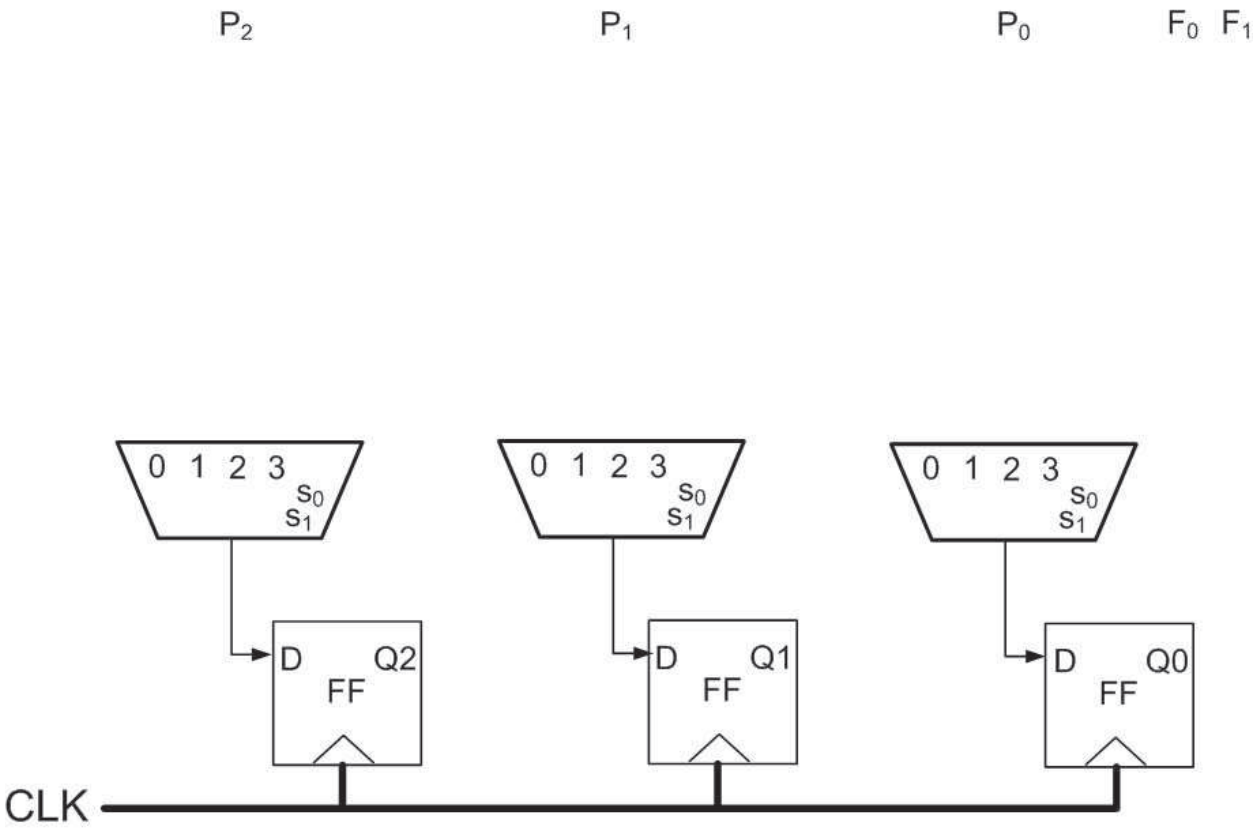
Using one additional flip-flop (drawn for you) and no more than three extra gates, build a system that recognizes the two sequences independently. In other words, Z_{001} should be 1 whenever the last three values from input T are 001, and Z_{110} should be 1 whenever the last three values from input T are 110.



Problem 3 (15 points): Shift Register Design

Complete the design of a 3-bit register that performs the operations listed in the table below. Parallel load inputs are labeled and indexed as P_i. When arithmetic operations are performed, assume that the numbers are in 2's complement representation. You may use Q_i inputs without drawing the wire from the outputs of the flip-flops (just write the appropriate Q_i label).

F_1	F_0	Operation
0	0	No change
0	1	Circular shift right
1	0	Arithmetic shift left
1	1	Parallel load



Problem 4 (30 points): Parking Garage FSM

A small parking garage has parking spaces for three cars. Your job is to design an electronic admission control system that only admits cars when there are empty spaces. There are two binary inputs to the system; the signal E is 1 when a car enters the garage, and 0 otherwise, and the signal L is 1 when a car leaves the garage, and 0 otherwise. There is an output signal, V, that your system should set to 1 when there is a vacancy (that is, when there are less than three cars currently in the garage), and reset to 0 otherwise. (The V signal controls the signage and a gate that prevents entry if the garage is full.) *Hint: in Part B, you will need to represent your system with two internal state bits.*

Part A (10 points): Design a state machine implementing this system. Determine the states you need, **giving them meaningful names** so that we can comprehend your design intent, and sketch the complete state transition diagram for your FSM.

You should not need this much space!

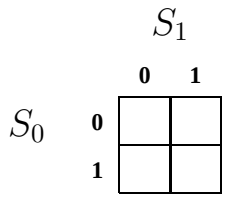
Problem 4, continued:

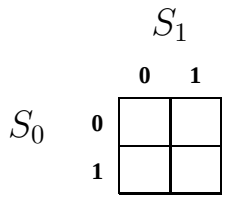
Part B (8 points): Fill in the truth table below by assigning internal state bit representations S_1S_0 and output value V for each of your states from **Part A** and then filling in the values for the next-state variables S_1^+ and S_0^+ for each combination of S_1S_0EL.

Part C (6 points): Using the truth table that you wrote in **Part B**, fill in the K-maps for V, S_1^+, and S_0^+, and derive Boolean expressions for these variables. We have provided you with two copies of blank K-maps for each variable. **CROSS OUT the THREE K-maps that you DO NOT want graded.**

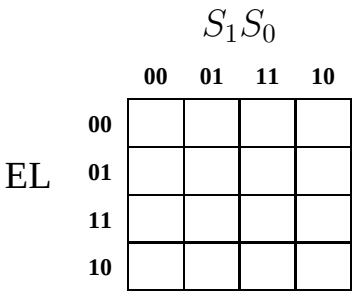
State Name	S_1	S_0	E	L	V	S_1^+	S_0^+
			0	0			
			0	1			
			1	0			
			1	1			
			0	0			
			0	1			
			1	0			
			1	1			
			0	0			
			0	1			
			1	0			
			1	1			
			0	0			
			0	1			
			1	0			
			1	1			

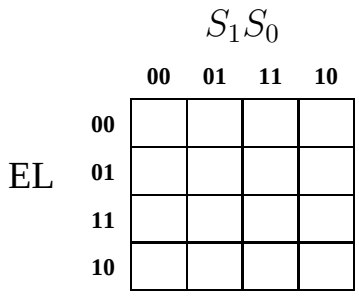
$V =$



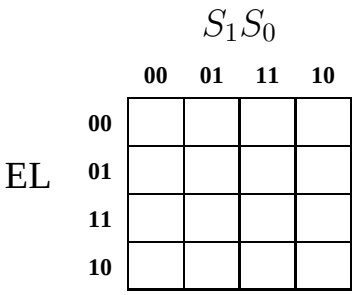


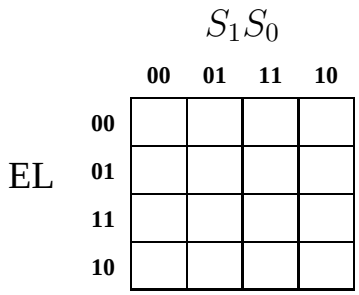
$S_1^+ =$





$S_0^+ =$





Problem 4, continued:

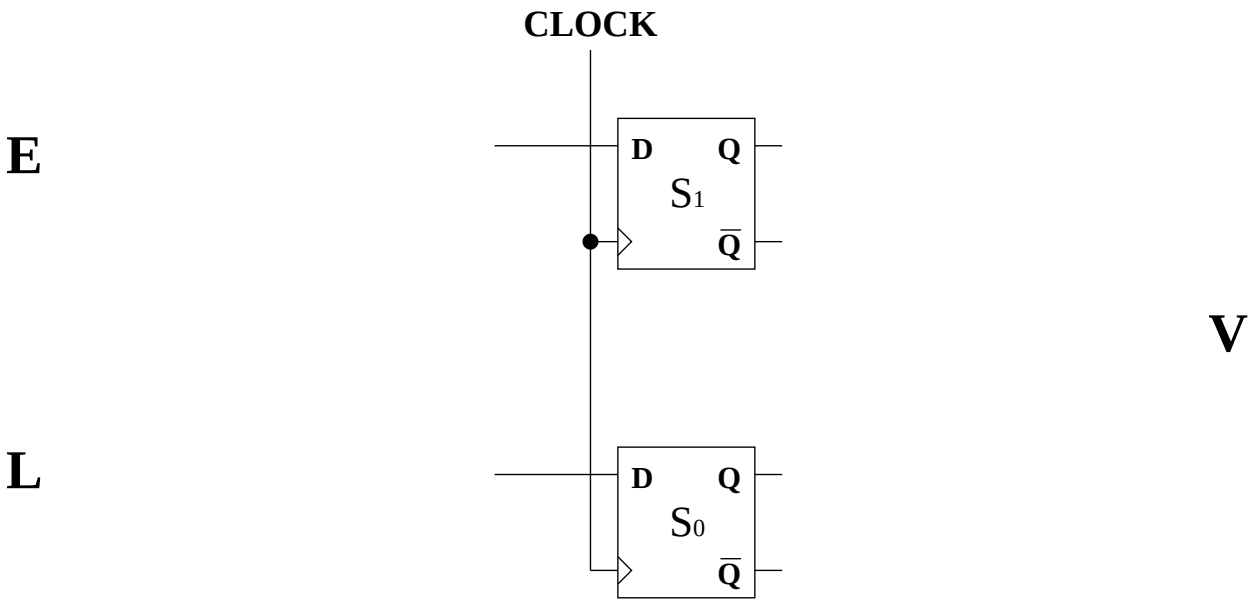
Copy the equations that you derived in **Part C** here.

$$V =$$

$$S_1^+ =$$

$$S_0^+ =$$

Part D (6 points): Add gates as necessary to complete the implementation of your FSM using the two positive edge-triggered D flip-flops shown below.



Problem 5 (25 points): Short-Answer Questions

Part A (10 points): The following components can be found in which major units of a von Neumann computer?

Circle the correct answer(s).

MAR	Control	Memory	Processing	Input/Output
PC	Control	Memory	Processing	Input/Output
Register file	Control	Memory	Processing	Input/Output
IR	Control	Memory	Processing	Input/Output
MDR	Control	Memory	Processing	Input/Output

Part B (5 points): If a memory has a 20-bit address space and has byte (8-bit) addressability, what will be its addressability if we convert this memory (same total number of bits) to a 16-bit address space?

Part C (5 points): What is the difference between a Mealy machine and Moore machine?

Part D*** (5 points): A "Harvard architecture" has separate program and data memories, as opposed to the "Princeton architecture," which has a single memory shared by both program and data. Assume that a Harvard architecture also has separate address and data busses for each of these memories.

In **25 or fewer words**, explain one one advantage of a Harvard architecture with respect to a Princeton architecture.

This page provided as scratch paper. If you need us to look at this page when grading, indicate this need **on the page of the corresponding problem** (not here!).