

CSE 140 Homework Three

February 19, 2013

*Only **Problem Set Part B** will be graded. Turn in only **Problem Set Part B** which will be due on March 6, 2013 (Wednesday) at 3:00pm.*

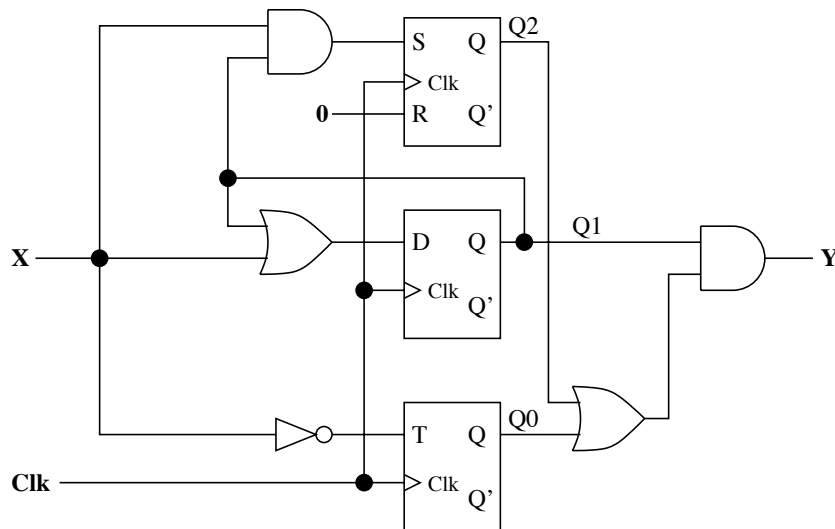
1 Problem Set Part A

- textbook 4.7(b)(d)
- textbook 4.9(b)(c)
- textbook 4.11(a)(c)
- textbook 4.17(b)
- textbook 4.19
- textbook 6.7
- textbook 6.8
- textbook 6.10
- textbook 6.11
- textbook 6.12
- textbook 6.14
- textbook 6.15
- textbook 6.17
- textbook 6.18
- textbook 6.19
- textbook 6.22

2 Problem Set Part B

1 (FSM Analysis and Optimization)

The following sequential circuit consists of three different flip-flops and five gates. The input and output signals are X and Y , respectively. Each state is encoded as $Q_2Q_1Q_0$, with **SR** being the most significant bit and T being the least significant bit of the state representation.



(Part A) In this part, you are asked to figure out the state diagram for the FSM. Assuming all the three flip-flops are initialized to 0 at the beginning, please specify all the transitions, as well as the input and output values associated with each transition. *Please note that you may not need to include all the states in your state diagram.*

000

010

100

110

001

011

101

111

(Part B) A group of CSE 140 students (from a previous batch) is having a hot discussion about the optimization opportunities for the given sequential circuit.

- Student **A** thinks that the number of flip-flops can be reduced by performing state minimization.
- Student **B** thinks that the original state diagram already has the minimal number of states.
- Student **C** thinks that while state minimization can be performed to reduce the number of states, the number of flip-flops can not be reduced.

Which of the three statements do you think is correct? Please give a brief reasoning.

(Part C) While the previous three students are struggling with the idea of state minimization, another pair of students are busy with implementing the given sequential circuit in *LogicWorks*. For some reason their implementation is not working correctly. After spending a lot of time debugging their circuit, they find that the problem is caused by a bug in *LogicWorks*: the inverter provided in the *LogicWorks* standard library is not working correctly.

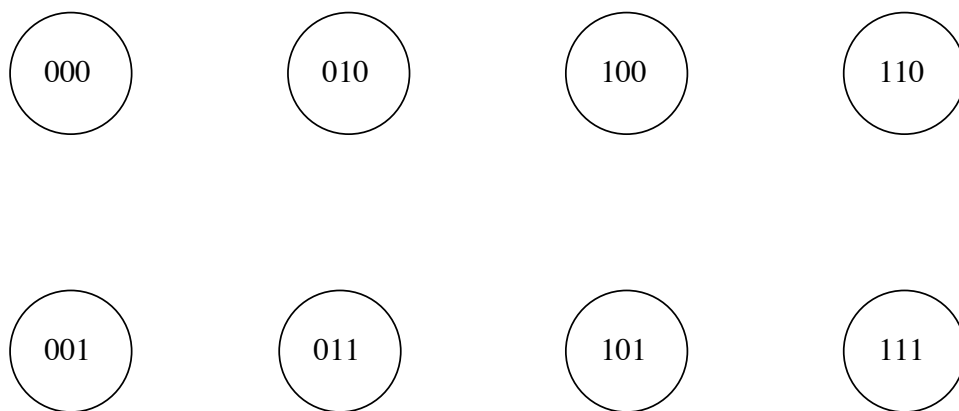
To make their implementation work correctly, Student **X** is wondering whether they should replace the inverter with an NAND gate or they should change the T flip-flop to a D flip-flop. Student **Y**, a somewhat smarter kid, proposes another idea. He has observed from the circuit that the inverter is only used by the T flip-flop and the next state logic of the other two flip-flops does not depend on the value of the T flip-flop. Therefore, he claims that without changing the flip-flop types, by only swapping the encodings of two states, the FSM can be implemented within only **two** 2-input **AND** and **two** 2-input **OR** gates and **no** inverters. Unfortunately, due to his highly limited communication skills, he cannot convince his partners about this idea. On the other hand, Student **X** does not believe in the statement of Student **Y**. Her argument is as follows: she has observed that some of the transformations on input '1' in the state diagram are self-loops, i.e., from state S_i to S_i . Therefore, she thinks that it is impossible to get rid of the inverter if the T flip-flop is still in use.

Since this group is in a deadlock situation with no one able to convince the other, they are asking for your help. As a student who is steeped in digital logic understanding, you immediately find that although the concern of Student **X** is well-founded, the statement of Student **Y** is correct. Please help Student **Y** to convince his partner by 1) enunciating clearly the two states whose encodings need to be swapped so as to get rid of the inverter, and 2) making appropriate modifications to the FSM that of course preserve the semantics yet can handle the self-loop problem. Furthermore, please help this group of students to quickly finish their implementation by figuring out the new state diagram and the new schematic of the circuit. (*Please clearly show your answers on the next page.*)

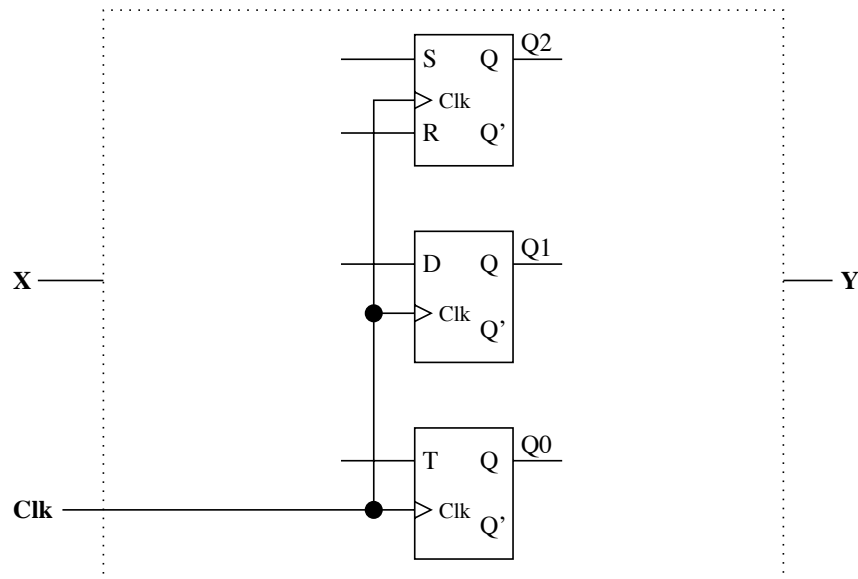
The two states whose encodings need to be swapped:

The way to handle the self-loop problem:

The new state diagram:



The new circuit implementation:



2 (State Minimization)

This question concerns the state minimization issue for both completely specified and incompletely specified FSMs.

If the next state and the output corresponding to any input value is specified for every state in an FSM, then such an FSM is a completely specified one. As you might have noticed, the FSMs discussed in the textbook are all completely specified FSMs. In *Parts A and B* of this question, you are asked to explore a set of interesting properties in the implication table based state minimization for completely specified FSMs.

However, in practical designs, we sometimes encounter FSMs with output values associated with a few transition edges unspecified (incompletely specified FSMs). These unspecified output values become *Don't Cares* in the next-state and output table. Since these unspecified outputs do not impact the functionality of the circuit, they can be safely ignored during the checking of state compatibility in the state minimization process. In other words, the only time when a pair of states with compatible next states fails to be merged into a single state is if both of them have specified outputs and the specified outputs conflict. In Parts C through E of this question, you are asked to investigate the issue of state minimization utilizing *Don't Cares*.

(Part A) A completely specified FSM has n groups of equivalent states identified by the state minimization process. Among the n groups, only two of them contain more than one state. More concretely, there is one group that contains **three** equivalent states and another group that contains **four**. All the other $n - 2$ groups each contain exactly one state. For the FSM outlined above, how many boxes in the implication table should **NOT** end up being crossed out during the state minimization process?

(Part B) A m -state (m is an **even** number which is larger than 3) completely specified FSM has two groups of equivalent states and thus can be finally reduced to a 2-state FSM by the state minimization process. Please identify the distribution of the m states in the two groups which can minimize/maximize the number of boxes that are crossed out from the implication table.

Distribution that **minimizes** the boxes that are crossed out:

of states in group 1 = ()
of states in group 2 = ()
of boxes that are crossed out = ()

Distribution that **maximizes** the boxes that are crossed out:

of states in group 1 = ()
of states in group 2 = ()
of boxes that are crossed out = ()

(Part C) The next-state and output table below defines an incompletely specified FSM, within which the output of state S_2 under input value 0 is a *Don't Care* (denoted by “-”). Please specify whether the number of states of this FSM can be further reduced. If your answer is **Yes**, please specify how the states are merged and draw the minimized FSM diagram. If your answer is **No**, please provide a reasoning for full credit.

Present state	Next state/output	
	$x = 0$	$x = 1$
S_1	$S_1/0$	$S_2/0$
S_2	$S_2/-$	$S_1/0$
S_3	$S_3/1$	$S_1/1$

(Part D) The next-state and output table below defines another incompletely specified FSM. Again, please specify whether the number of states of this FSM can be further reduced. If your answer is **Yes**, please specify how the states are merged and draw the minimized FSM diagram. If your answer is **No**, please provide a reasoning for full credit.

Present state	Next state/output	
	$x = 0$	$x = 1$
S_1	$S_3/0$	$S_2/0$
S_2	$S_2/-$	$S_3/0$
S_3	$S_3/1$	$S_2/0$

(Part E) The next-state and output table below defines yet another incompletely specified FSM. Again, please specify whether the number of states of this FSM can be further reduced. If your answer is **Yes**, please specify how the states are merged and draw the minimized FSM diagram. If your answer is **No**, please provide a reasoning for full credit.

Present state	Next state/output	
	$x = 0$	$x = 1$
S_1	$S_3/0$	$S_2/0$
S_2	$S_2/-$	$S_1/0$
S_3	$S_1/1$	$S_2/0$