Gavin M. (+ 3 Days Late)

CPSC3300 – Computer Systems Organization

Homework #2 – Boolean Algebra, Combinational Logic and Sequential Logic

Due: 11:59PM September 23

Total 100pts

1. [25pts] For the Boolean function E and F, as given in the following truth table:

Input			Output	
x	**y**	**z**	**O1**	**O2**
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	0	1

   a. List the minterms for a three-variable function with variables x, y, and z.

0: x'y'z
1: x'y'z
2: x'yz'
3: x'yz
4: x'yz
5: xy'z'
6: xy'z
7: xyz'
8 xyz

   b. Express O1 and O2 in sum-of-product algebraic form.

O1 = x'y'z' + x'y'z + x'yz' + x'yz + xyz'
O2 = xy'z' + xy'z + xyz

2. [25] In class, we learned the implementation for a 4-bit carry lookahead adder. We can use the same idea and extend to build a 16-bit carry lookahead adder. Denote this implementation one-level carry lookahead adder.

   In the textbook, Figure 8.6.3 shows a two-level implementation of a 16-bit carry lookahead adder. This adder uses 4-bit carry lookahead adders at the lower level, and uses a carry lookahead unit at the higher level.

   Compare these two implementations and provide your explanation why the two-level implementation could be preferred.

The single-level 16-bit carry-lookahead adder using a single 16-bit carry-lookahead unit has the fewest gate delays. However, because each higher bit generates signals from all lower bits, more gates plus wider gates are used with each additional bit. This dramatically lengthens the transistor count and is inefficient as well as unproductive to implement in hardware.

Alternatively, having four 4-bit carry-lookahead adders in series, with the highest carry bit being the carry-in bit for the next higher 4-bit CLA, will require much less hardware but is noticeably slower. The two-level 16-bit carry-lookahead adder introduces an additional 4-bit carry-lookahead stage that uses a "super" propagator to generate a signal from the four 4-bit carry-lookahead adders. At the first stage, the input generates the carry-in signals for the three higher first-stage carry-lookahead adders. This extra second-stage carry-lookahead unit that uses "super" propagates and saves time.

3. [25] To provide proper rounding for floating-point addition, three extra bits are included in right shifts of fractions: guard, round, and sticky. The least-significant of these, the sticky bit, stays one whenever a 1 bit is shifted through it. In designing the logic for a sticky bit, let the following state transition table define the actions for R (reset) and I (input) on a D flip-flop.

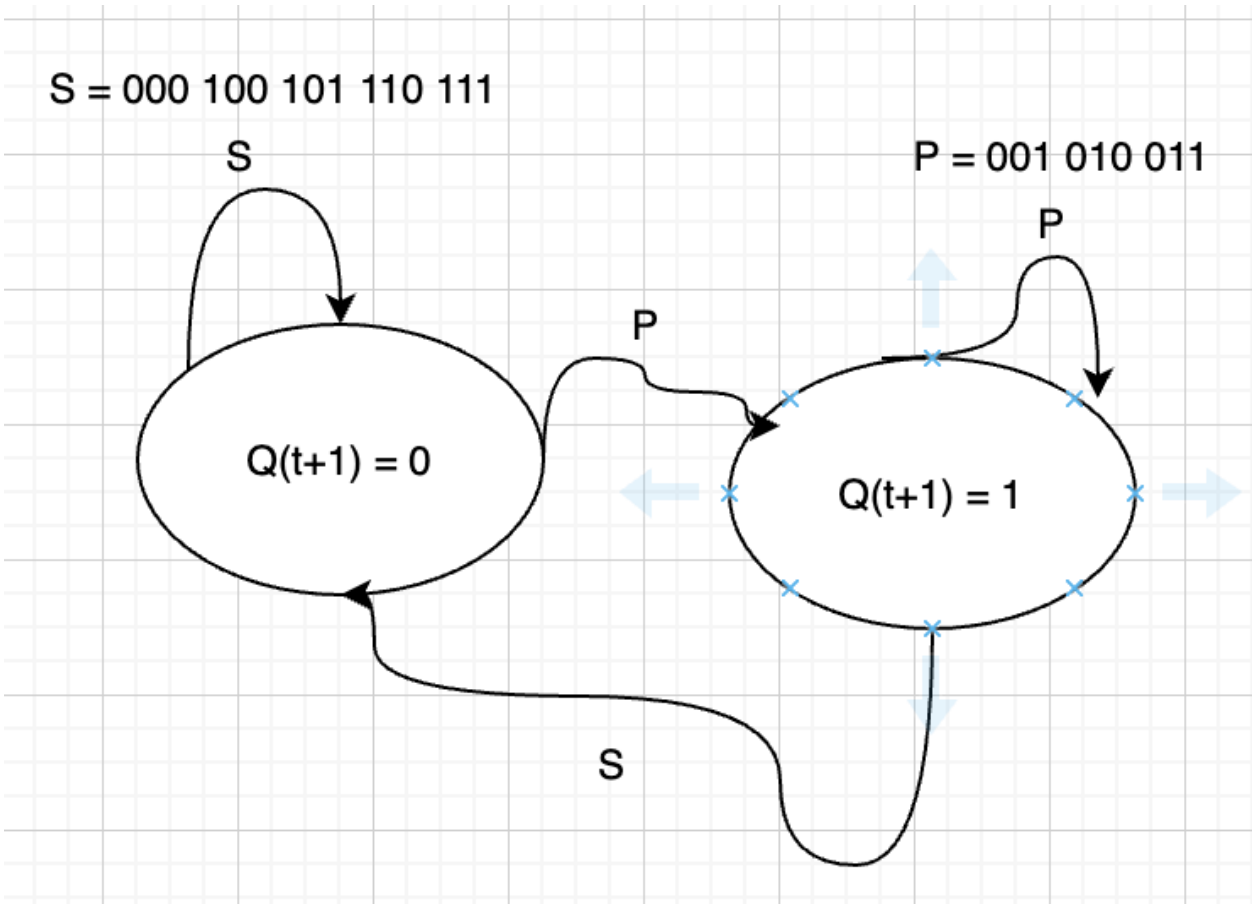
R	I	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	d	d	0

Five rows are shown (using d = don't care in the last row). The last row expands to four rows when a complete enumeration is done.

(a) Using a sum-of-products, give the input to the D flip-flop in terms of R, I, and Q(t).

D = R'I'Q+R'IQ'+R'IQ

(b) Draw the state diagram for the sticky bit logic from the table above. There is no separate output signal.

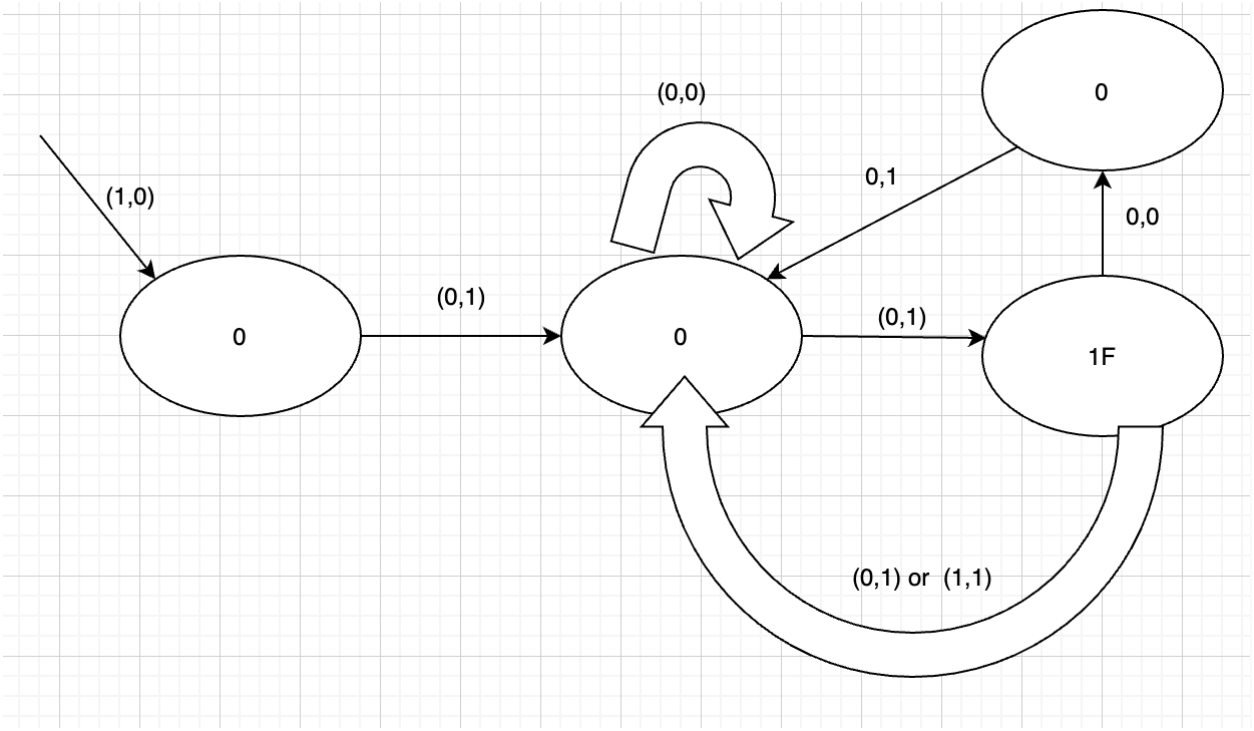


4. [25] Consider a state machine with two inputs, R and I (reset and in), that, after a reset R, outputs a 1 on each second 1 that it receives on input I. Reset causes any count of previous and current I=1 inputs to be lost, and the counting of I=1 inputs starts in the subsequent clock cycle after the reset signal goes back to 0.

That is, the state machine behaves like this

input R: 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0
input I: 0 1 0 1 1 1 1 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1
output O: 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 1

(a) Give the state diagram. (There should be only two states.)



(I wasn't 100% sure how to properly draw this. I know there's more than two states but if you follow the input it works)

(b) Give the state transition table with inputs R, I, current state Q(t), next state Q(t+1), and output O.

Extra credits [10]
Implement the above state machine using a circuit.
(Notice – the sum of the HW1, HW2, and extra credits will be less than or equal to 250.)