Johnny poet

Combinatorial Circuits; Storage Elements

CS 350: Computer Organization & Assembler Language Programming

Due Tue Nov 29 (no late assignments)

A. Why?

- Combinatorial logic circuits correspond to pure (state-free) calculations on booleans.
- Storage elements are the basic circuits that store data.

B. Outcomes

After this lab you should be able to

- Translate between simple combinatorial logic circuits and boolean expressions.
- Recognize whether a circuit can be said to remember or store data.
- Describe how *R-S* and *D* latches work as storage elements

C. Problems [50 points total]

1. [14 = 7 + 7 pts] Let's implement 2's complement negative ($Y = -X$) using a modular design. At position i, we have bit $Y_i = \overline{X}_i$ or $Y_i = X_i$ depending on whether bit X_i is or is not in the trailing (i.e., rightmost) 10…0 section of X. Let $T_i = 1$ if position i contains a rightmost zero or is the 1 bit just before the rightmost zeros, and let $T_i = 0$ otherwise. Note T_0 always = 1.

    a. Give equations for outputs Y_i and T_{i+1} from inputs X_i and T_i.

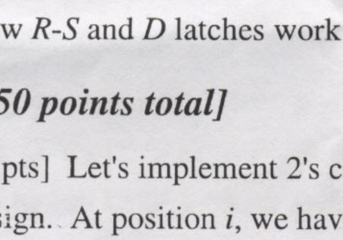
    b. Give logic gate implementations of Y_i and T_{i+1} using inputs X_i and T_i. You can give a PLA-based implementation or you can simplify using any/all of the standard gates (*AND*, *OR*, *NAND*, *NOR*, *XOR*, *XNOR*, and *NOT*), your choice.

X_i	T_i	Y_i	T_{i+1}
0	0	1	0
0	1	0	0
1	0	0	1
1	1	1	0

1a) $Y_i = \overline{X_i \veebar T_i}$

$T_{i+1} = \overline{X}_i \wedge T_i$

1b)



2. [16 = 4 + 4 + 4 + 6 pts] Consider the following partial statements about *R*-*S* latches. Map each of (a) – (d) to all of (1) – (5) that apply.

$\boxed{Q_{n+1} = d_n}$ →

(a) If $R\,S$ then …

(b) If $\overline{R}\,S$ then … $\boxed{1,5}$

(c) If $R\,\overline{S}$ then … $\boxed{2,4}$

(d) If $\overline{R}\;\overline{S}$ then … $\boxed{3,6,7}$

(1) $Q = 01$ is stable

(2) $Q = 10$ is stable

(3) $Q = 11$ is stable

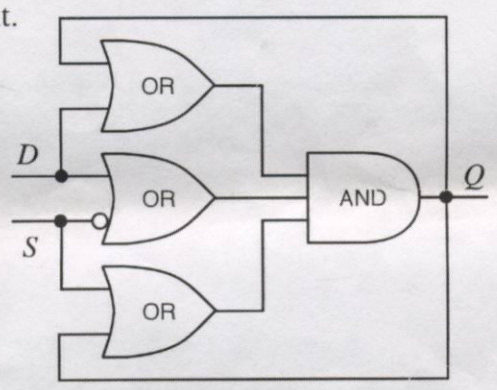
(4) If $Q = 01$ then $Q = 11$ then 10 and is stable

(5) If $Q = 10$ then $Q = 11$ then 01 and is stable

(6) If $Q = 01$ then $Q = 11$ and is stable

(7) If $Q = 10$ then $Q = 11$ and is stable

3. [18 = 4 + 6 + 4 + 4 pts] Study this logic circuit.



a. Translate the circuit to an equation for new Q = a boolean expression over D, S, and (the current value of) Q. (Make the translation direct.)

b. Simplify your equation from part (a) to get new Q = a DNF expression. (Doesn't have to be full DNF.) Show your work.

c. When does this circuit have logically stable or unstable values for Q?

d. How this circuit be used to store a bit? I.e., how can we set / reset / maintain the value of Q?

b) $Q = \boxed{(d \wedge S) \vee \neg Q}$

a) $\boxed{Q = (Q \vee D) \wedge (D \vee \neg S) \wedge (S \vee Q)}$

c) When

d) Q can be set if (d and S)==1.
Q gets reset to 0 if (State==1) and (S==1) and (d==0)

2. [16 = 4 + 4 + 4 + 6 pts] Consider the following partial statements about *R*-*S* latches. Map each of (a) – (d) to all of (1) – (5) that apply.

+1 = d_n

(a) If $R\,S$ then …

(b) If $\overline{R}\,S$ then … 1,5

(c) If $R\,\overline{S}$ then … 2,4

(d) If $\overline{R}\;\overline{S}$ then … 3,6,7

(1) $Q = 01$ is stable

(2) $Q = 10$ is stable

(3) $Q = 11$ is stable

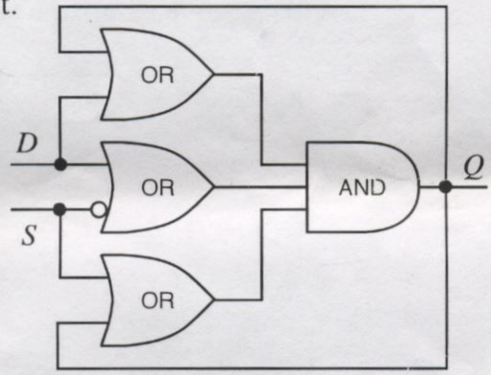
(4) If $Q = 01$ then $Q = 11$ then 10 and is stable

(5) If $Q = 10$ then $Q = 11$ then 01 and is stable

(6) If $Q = 01$ then $Q = 11$ and is stable

(7) If $Q = 10$ then $Q = 11$ and is stable

3. [18 = 4 + 6 + 4 + 4 pts] Study this logic circuit.



a. Translate the circuit to an equation for new *Q* = a boolean expression over *D*, *S*, and (the current value of) *Q*. (Make the translation direct.)

b. Simplify your equation from part (a) to get new *Q* = a DNF expression. (Doesn't have to be full DNF.) Show your work.

c. When does this circuit have logically stable or unstable values for *Q*?

d. How this circuit be used to store a bit? I.e., how can we set / reset / maintain the value of *Q*?

$= \boxed{(d \wedge S) \vee \neg Q}$

a) $\boxed{Q = (Q \vee D) \wedge (D \vee \neg S) \wedge (S \vee Q)}$

Q can be set if (d and S) == 1.

Q gets reset to 0 if (State == 1) and (S == 1) and (d == 0)