University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Science

EECS 150
Spring 2007

R. H. Katz, Instructor

FIRST MIDTERM EXAMINATION
Thursday, 15 February 2007

INSTRUCTIONS—READ THEM NOW! This examination is CLOSED BOOK/CLOSED NOTES. There is no need for calculations, and so you will not require a calculator, PDA, laptop computer, iPhone, Nintendo DS, or other calculation aid. Please put away your (camera) cell phones too! You MAY use one 8.5" by 11" double-sided crib sheet, as densely packed with notes, formulas, and diagrams as you wish. The examination has been designed for 50 minutes/50 points (1 point = 1 minute, so pace yourself accordingly). Nevertheless, you will have the full 80 minutes to work on it. All work should be done on the attached pages.

In general, if something is unclear, write down your assumptions as part of your answer. If your assumptions are reasonable, we will endeavor to grade the question based on them. If necessary, of course, you may raise your hand, and a TA or the instructor will come to you. Please try not to disturb the students taking the examination around you.

We will post solutions to the examination as soon as possible, and will grade the examination as soon as practical, usually within a week. Requests for regrades should be submitted IN WRITING, explaining why you believe your answer was incorrectly graded, within ONE WEEK of the return of the examination. We try to be fair, and do realize that mistakes can be made during the grading process. However, we are not sympathetic to subjective arguments of the form "I got half the problem right, why did I get a quarter of the points?"

Answer Key
(Signature)

SID: 77777777

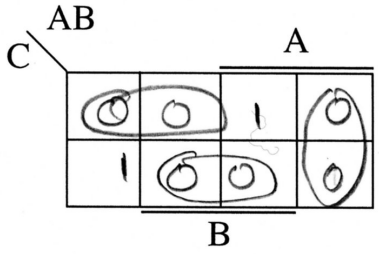
Smart T. Pants
(Name—Please Print!)

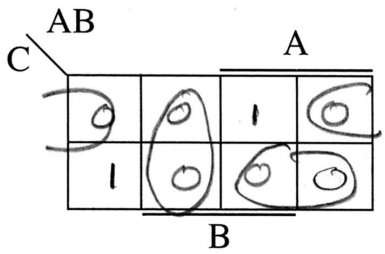
QUESTION	POINTS ASSIGNED	POINTS OBTAINED
1	10	
2	10	
3	10	
4	10	
5	10	
TOTAL	50	

Student Name: _____ SID: _____

Question 1. *Combinational Logic Canonical Forms* (10 points)

(a) (6 points) Fully fill in the following two 3-variable K-maps with an identical patterns of 0's and 1's (your chosen pattern need not have the same number of 0's and 1's in it). Then derive two *different* minimized *Product of Sums* (PoS) forms for the same function by circling the 0's in an alternative way in each of the two K-maps. Write the minimized equations derived from your two choices of circling. Indicate the number of literals and sum terms in each answer.





F(A,B,C) = $(A+C)(\bar{B}+\bar{C})(\bar{A}+B)$

# of Literals = 6

# of Sum terms = 3

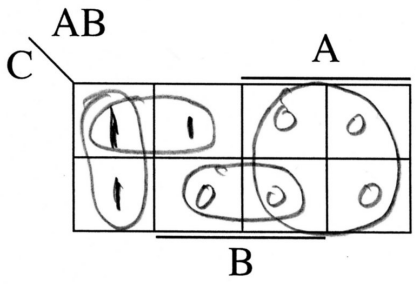
F(A,B,C) = $(A+\bar{B})(\bar{A}+\bar{C})(B+C)$

# of Literals = 6

# of Sum terms = 3

Are they the same? If not, why not? If so, why? Must be the same! Same truth table, equivalent minimized equations as result

(b) (2 points) Fully fill in the following 3-variable K-map with a pattern of 0's and 1's such that the minimized *Product of Sums* (PoS) form has FEWER literals than the minimized *Sum of Products* (SoP) form for the same function. Your pattern must have at least one 0 and one 1! Write the minimized equations in each case and indicate the number of literals.



Min PoS F(A,B,C) = $\bar{A}(\bar{B}+\bar{C})$

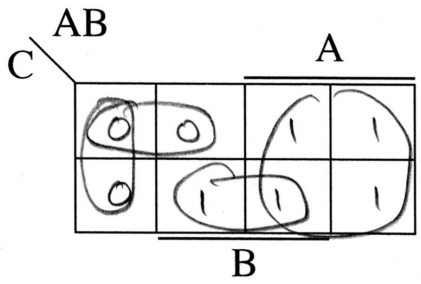
# of Literals = 3

Min SoP F(A,B,C) = $\bar{A}\bar{B} + \bar{A}\bar{C}$

# of Literals = 4

Student Name: _____ SID: _____

(c) (2 points) Fully fill in the following 3-variable K-map with a pattern of 0's and 1's such that the minimized *Sum of Products* (SoP) form has FEWER literals than the minimized *Product of Sums* (PoS) form for the same function. Your pattern must have at least one 0 and one 1! Write the minimized equations in each case and indicate the number of literals.



use complement of function from part (b)!

Min SoP F(A,B,C) = $A + BC$

# of Literals = 3

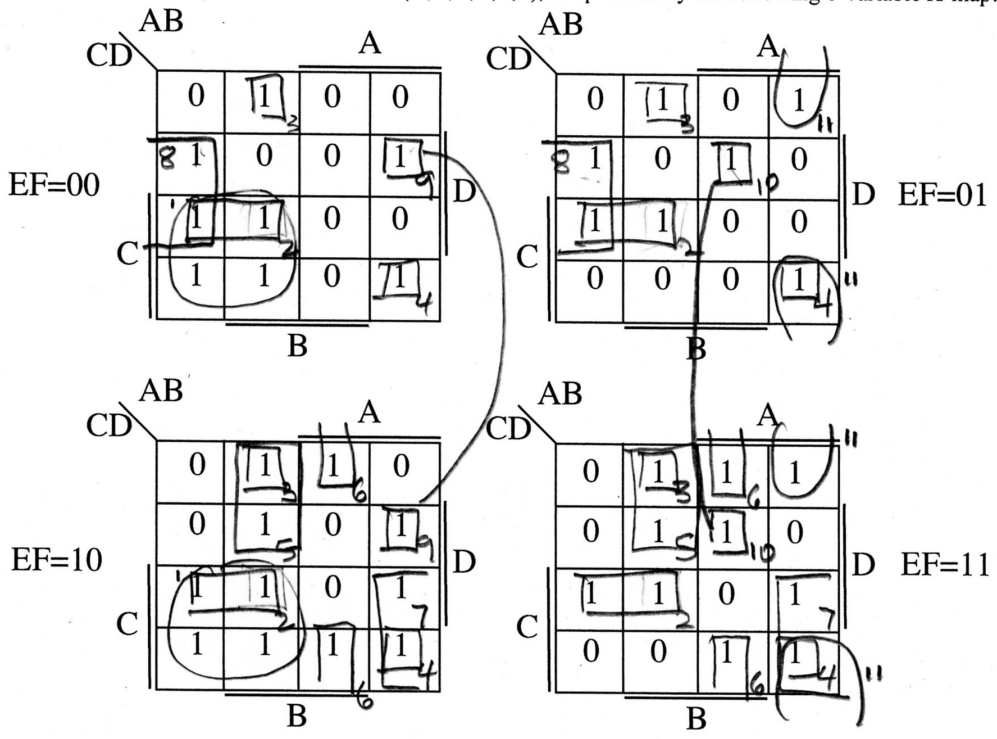
Min PoS F(A,B,C) = $(A+B)(A+C)$

# of Literals = 4

must be minimized correctly to receive full points

Student Name: _____ SID: _____

Question 2. *6-Variable K-maps and Mux-based Logic Design* (10 Points)
Consider the following Boolean function G(A,B,C,D,E,F), as specified by the following 6-variable K-map:



(a) (4 points) Circle 1's in the above K-map to derive the minimized *Sum of Products* (SoP) form for G. Write down the number of literals and product terms in your answer:

G(A,B,C,D,E,F) = $\bar{A}C\bar{F} + \bar{A}CD + \bar{A}B\bar{C}D + AB\bar{C}\bar{D} + \bar{A}B\bar{C}E + AB\bar{D}E + A\bar{B}CE + \bar{A}B\bar{D}\bar{E} + AB\bar{C}D\bar{F} + AB\bar{C}DF + A\bar{B}\bar{D}F$

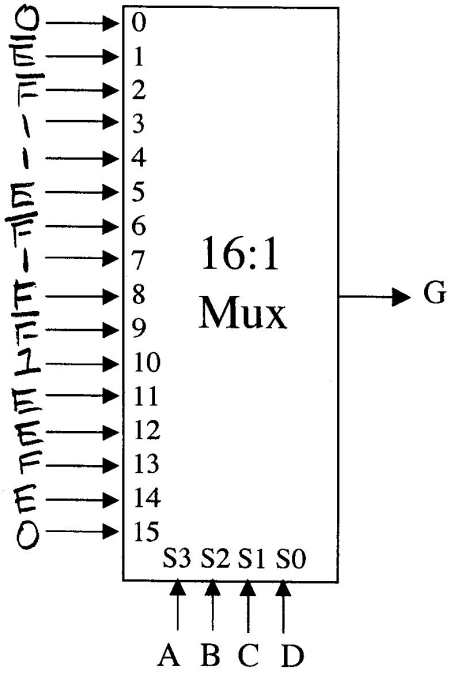
# of Literals = 44

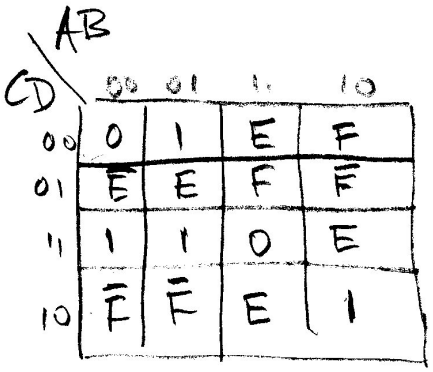
# of Product Terms = 11

It is possible that a better solution exists. This is the best I could come up with.

(b) (6 points) A general 6-variable function can always be implemented with a 64:1 multiplexer with only 0's and 1's on the mux's data inputs. It can also be implemented with a 32:1 multiplexer with 0, 1, and a single variable or its complement on the data inputs (this is the variable not used on the selection inputs).

Some 6-variable functions can be implemented using even simpler multiplexers. G is such a function. Show how to implement G using a single 16:1 mux, with ABCD on the selection inputs, and 0, 1, E, $\overline{E}$, F, $\overline{F}$ wired to the data inputs:



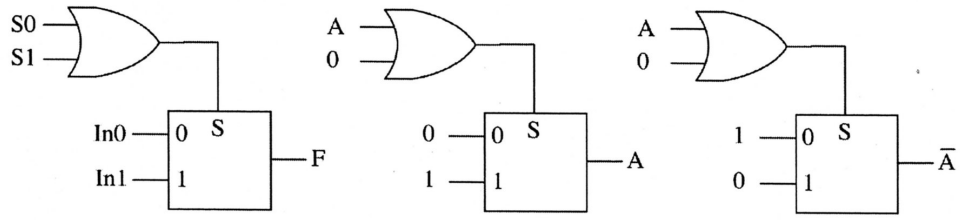


← Can be filled in directly by looking at the K-map.

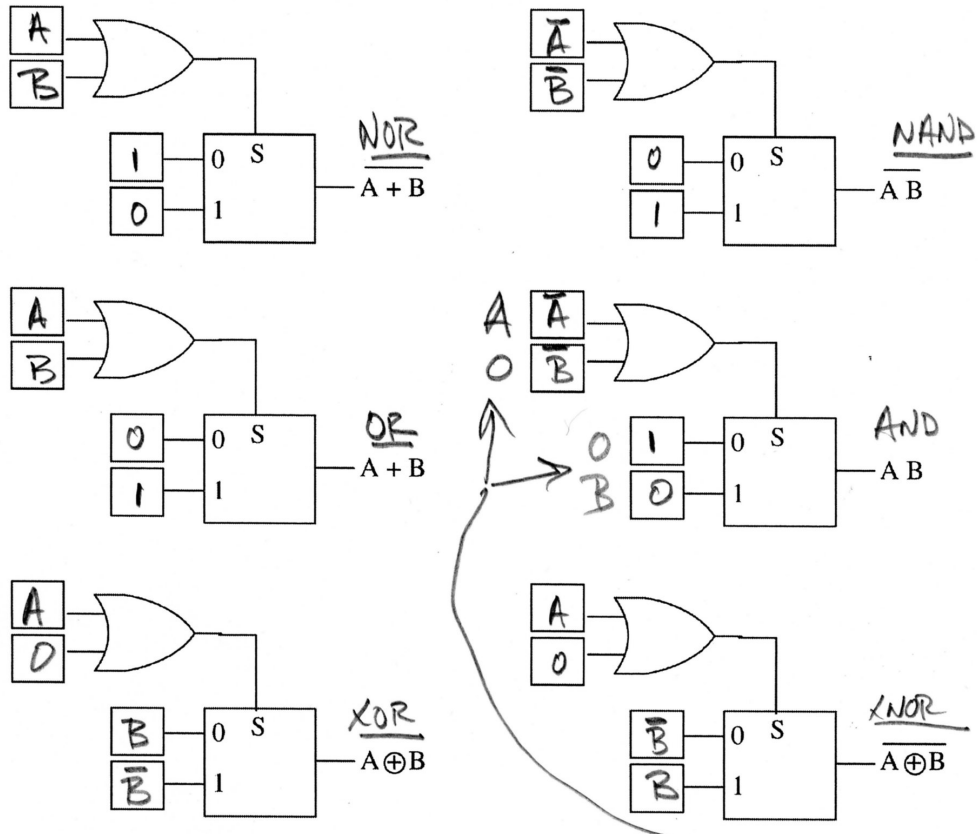
If you drew a 64 line truth table somewhere you belong at Stanford!

Student Name: _____ SID: _____

Question 3. *Programmable Logic* (10 Points)
A former CS150 student has started a competitor to Xilinx using the following as the basic programmable logic building block, shown at the left below. It is a slightly modified 2:1 multiplexer. In the middle and at the right demonstrate ways to connect the inputs to implement the functions A and the complement of A.



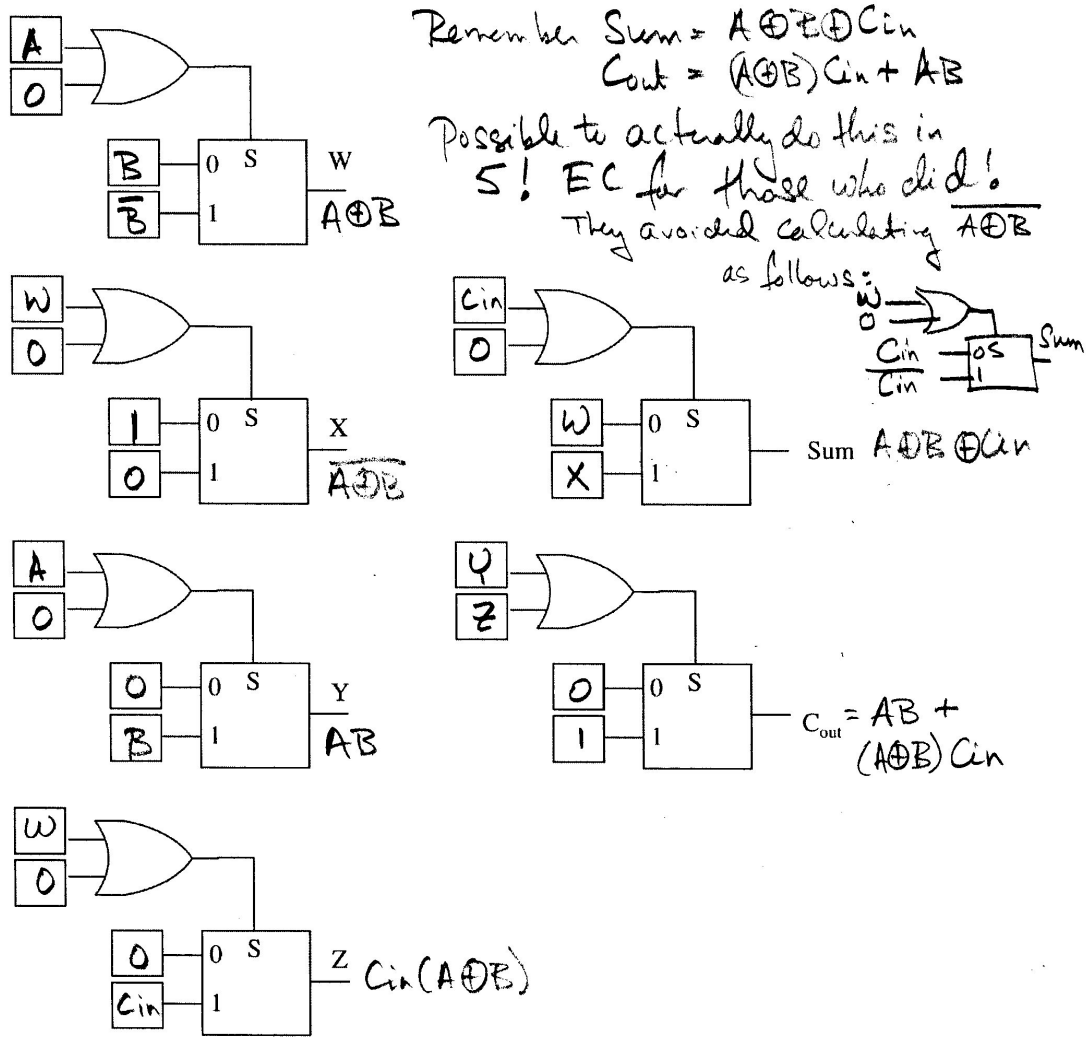
(a) (6 points) Fill in the boxes in the figure below to show how to connect the building block to inputs A and B, their complements, and the constants 0 and 1, to implement the indicated logic functions:



Other solutions possible, for example

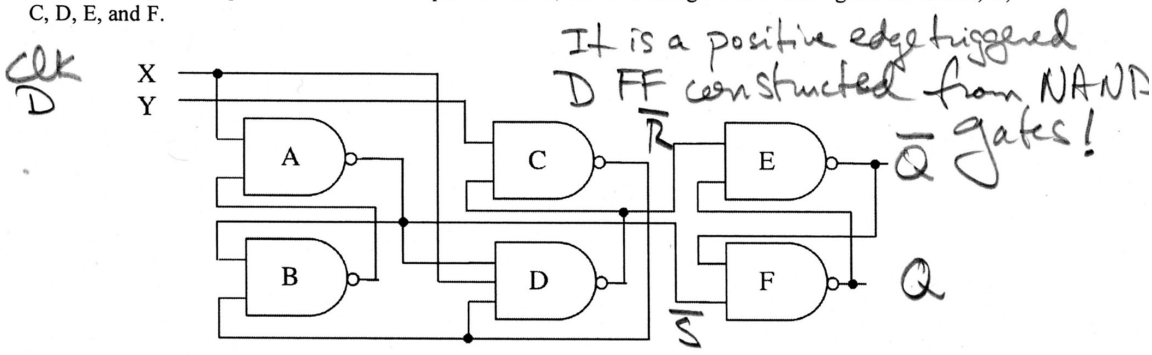
Student Name: _____ SID: _____

(b) (4 points) Using this building block, and the template below, show how to implement the full adder functions Sum(A,B,C_{in}) and C_{out}(A,B,C_{in}) using exactly six of the basic building block. You may fill in the boxes with the constants 0 and 1, the inputs A, B, C_{in}, and their complements, and the outputs from the first four building blocks in the template, labeled W, X, Y, and Z.

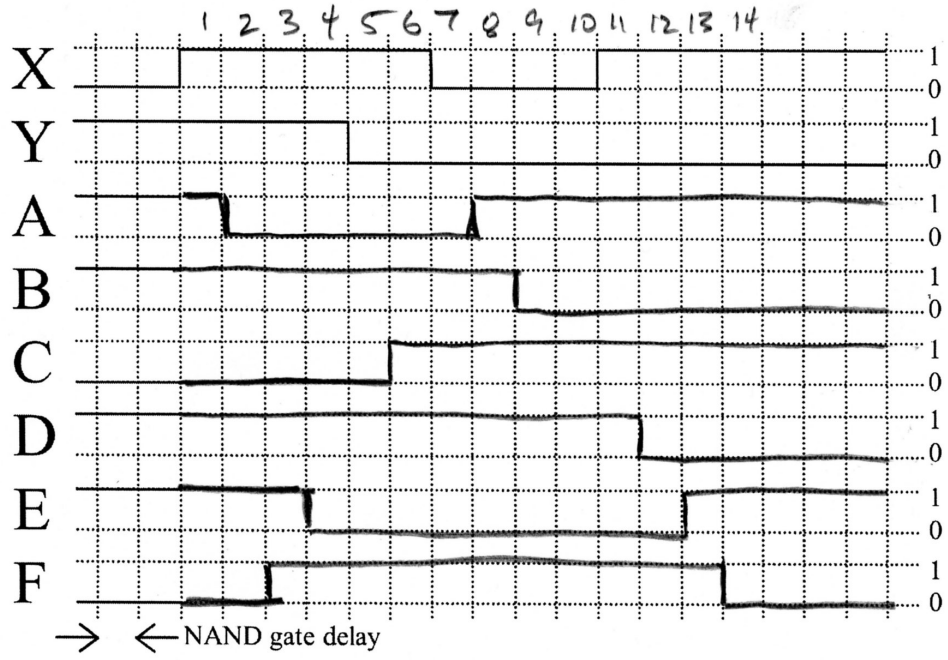


Student Name: _____ SID: _____

Question 4. *Timing Behavior* (10 Points)
The following is a sequential circuit with inputs X and Y, and consisting of six NAND gates labeled A, B, C, D, E, and F.



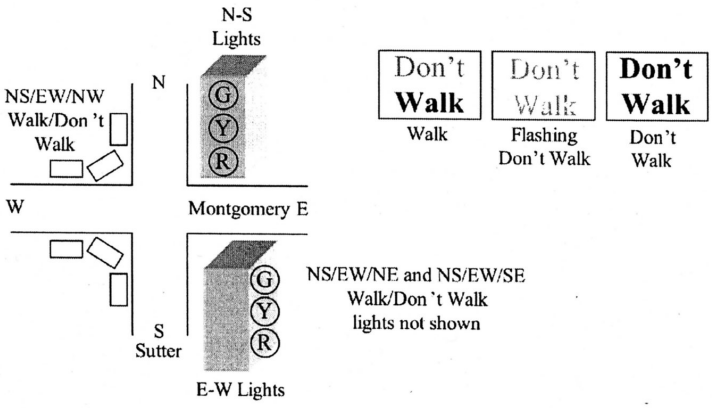
Below is a timing waveform that shows transitions in X and Y and the initial values at the output of each of the labeled NAND gates. The time between vertical grid lines is a single NAND gate delay. Complete the waveform for the circuit nodes at the outputs of the NAND gates labeled A, B, C, D, E, and F.



Student Name: _____ SID: _____

Question 5. *State Machine Design* (10 Points)
Professor Katz lives in San Francisco and is fascinated by the traffic lights there. At the intersection of Montgomery and Sutter Streets in the Financial District the traffic lights and Walk/Don't Walk signs are arranged so that at one point in the sequence, pedestrians can cross in all directions at once, allowing them to cross diagonally across the street if they so choose! Your task is to design a simple state machine for this traffic light system.

The figure below shows the Montgomery-Sutter intersection, with Green-Yellow-Red lights facing North-South (along Sutter) and East-West (along Montgomery). In addition, there are Walk/Don't Walk signs that face N-S and E-W, but also NE-SW and NW-SE (the NE and SE lights are not shown in the figure).



The state machine must adhere to the following specifications and constraints:

1. The traffic lights must pass through the sequence RED (R), followed by GREEN (G), followed by YELLOW (Y) before they can return to R. We use R, G, and Y to indicate that the RED, GREEN, and YELLOW lights are illuminated respectively. You must never transition from G to R without first passing through Y.
2. The WALK/DON'T WALK pass through the sequence WALK (W), DON'T WALK flashing ("DW"), DON'T WALK (DW) before they can return to W. We use W, "DW," and DW to indicate that WALK, flashing DON'T WALK, and DON'T WALK are illuminated respectively. You must never transition directly from W to DW without first passing through "DW".
3. Normally, when the traffic lights are G in a given direction, the walk/don't walk signs in that direction are W.
4. Normally, when the traffic lights are Y in a given direction, the walk/don't walk signs are "DW".
5. Normally, when the traffic lights are R in a given direction, the walk/don't walk signs in that direction are DW.
6. HOWEVER, there must be a point in the R-G-Y light sequence where traffic is stopped in all directions and pedestrians are permitted to cross the street in all directions: E-W, N-S, NW-SE, and NE-SW.
7. A synchronous reset signal puts the state machine in an initial state that is R on Sutter, G on Montgomery, with the N-S, NE-SW, NW-SE pedestrian signs showing DW, and the E-W signs showing W.
8. For simplicity, assume that the machine stays in every state for the same amount of time. A clock signal, which is twenty seconds long, advances the machine from one machine state to the next.

Normally does not imply always

Student Name: _____ SID: _____

Describe a state machine for this traffic light controller by completing the symbolic state table shown below. You must come up with a solution *with the fewest possible states* that meet the above specifications. Add more rows to the table if you need them. Write down any additional assumptions you feel you need to make in the space below the table.

Reset	Current State	Next State	N-S Lights	E-W Lights	N-S W/DW	E-W W/DW	NE-SW W/DW	NW-SE W/DW
1	X	Montgomery Green	R	G	DW	W	DW	DW
0	Montgomery Green	Montgomery Yellow	R	Y	DW	W	DW	DW
0	Montgomery Yellow	Red ALL	R	R	W	W	W	W
0	Red ALL	Sutter Green	R	R	W	"DW"	"DW"	"DW"
0	Sutter Green	Sutter Yellow	G	R	W	DW	DW	DW
0	Sutter Yellow	X	Y	R	"DW"	DW	DW	DW

Adding these states to the basic 4 state traffic light controller is the key

Just six states!

6 - Y - R

W - "DW" - DW

Additional assumptions (WARNING: it is left to the grader to determine if your assumptions are reasonable!):