UCSD CSE140L Spring 2014

LAB#1 Report

Demonstration Date : 4 / 15 / 14 Student CID__________268_______________

Student Name: _______ Kieth__________________________________Vo___________________

first M.I. Last

Submission Date & Time :

(FILLED BY Student BEFORE DEMO)

Self-test Report

	Working	Not working
Part1:	_________	________
Part2:	_________	_________
Part3:	_________	_________
Part4:	_________	_________
Part5:	_________	________

(*** FILLED BY TUTOR/INSTRUCTOR ***)

Demo Reviewer
Name : ______________________________

Demo score	**Report** score
_________/1	a) ______/1
_________/1	b)_______/1
__________/1	c) _______/1
__________/1	d) _______/1
_________ /1	e) ________/1
Subtotal	**Subtotal**
___________**/5**	_____________**/5**

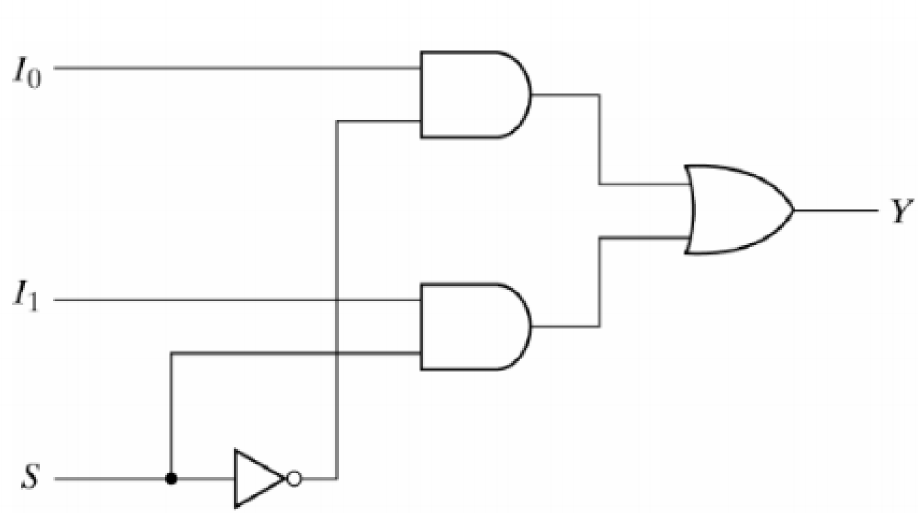
TOTAL Score: ________________**/10**

A)

1. **1.** For part 1 I used the input from SW0 and split the path into two. One if the switch is positive and one if the switch is negative. When it is negative the switch will have an output that takes off the middle segment in the 7 segment LED display in HEX 1 2 and 3 which will give us zeros. When the switch is positive, it will do the same things but this time the result would be my CID number which is 268 by blocking out the respective segments in the 7 segment LED display.
2. **2.** For part 2 I used the truth table given to create a sum of products equation. The equation I got was (SW2`SW1`SW0) + (SW2`SW1SW0`) + (SW2SW1`SW0) + (SW2SW1SW0). Then I created the appropriate logic gates that are equivalent to the equation.

SW2	SW1	SW0	LEDR1
0	0	0(=Down)	0(=OFF) <--- Initial condition
0	0	1(=Up)	1(=ON)
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

3. **3.** For part 3 I used the equivalent logic gates that mimic the MUX design. This was discussed in class and the professor showed us the circuit design for a MUX design using AND and OR gates and I just recreated the design.



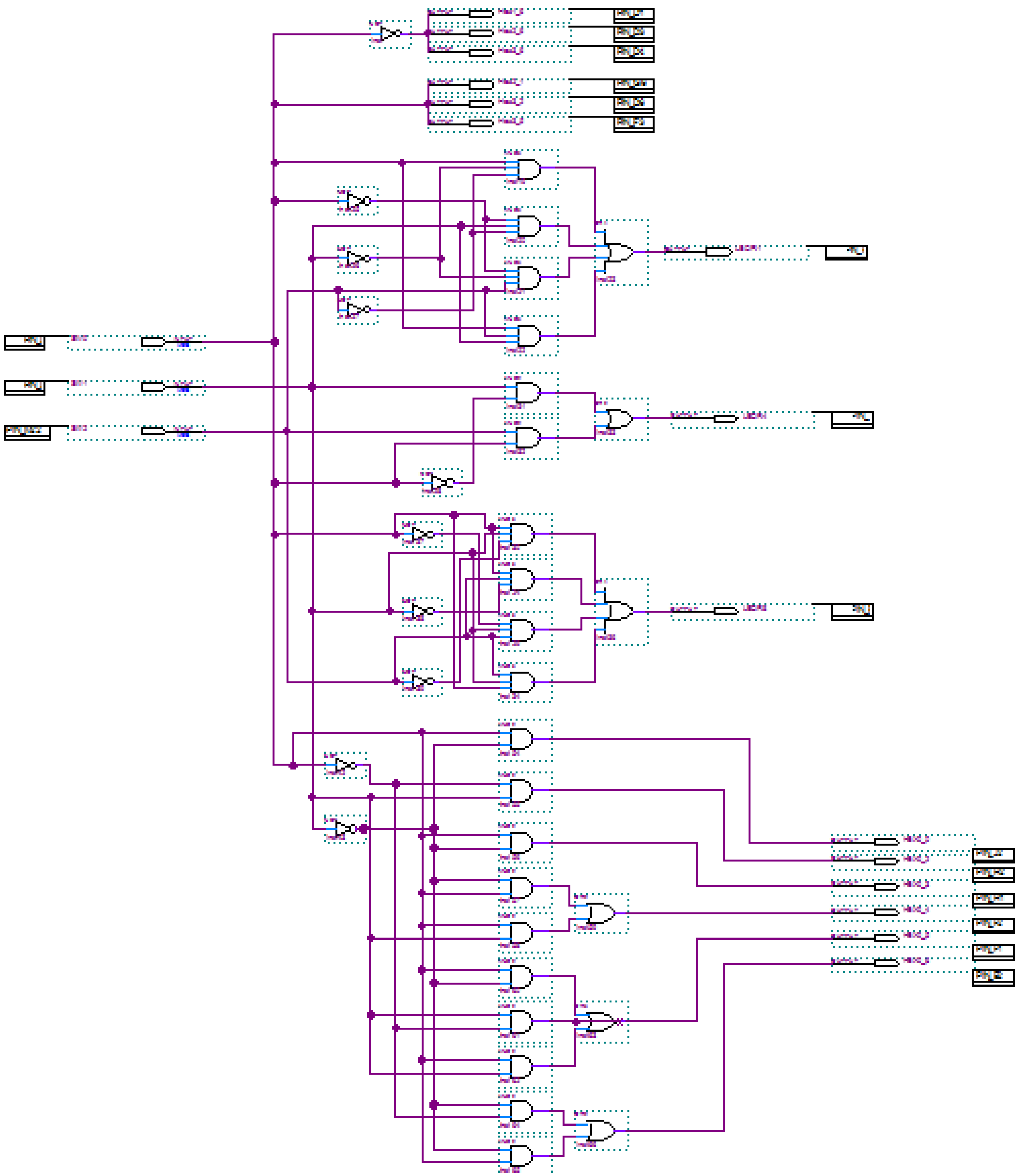
4. **4.** For part 4 I used a similar truth table like in part 2 but this time the equation was different. Since the output only equaled 1 when there is a majority of 1's. I figured out the sums of products equation which was (SW2`SW1SW0) + (SW2SW1`SW0) + (SW2SW1SW0`) + (SW2SW1SW0). Again, I just used logic gates to create replicate the equation.

SW2	SW1	SW0	LEDR8
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

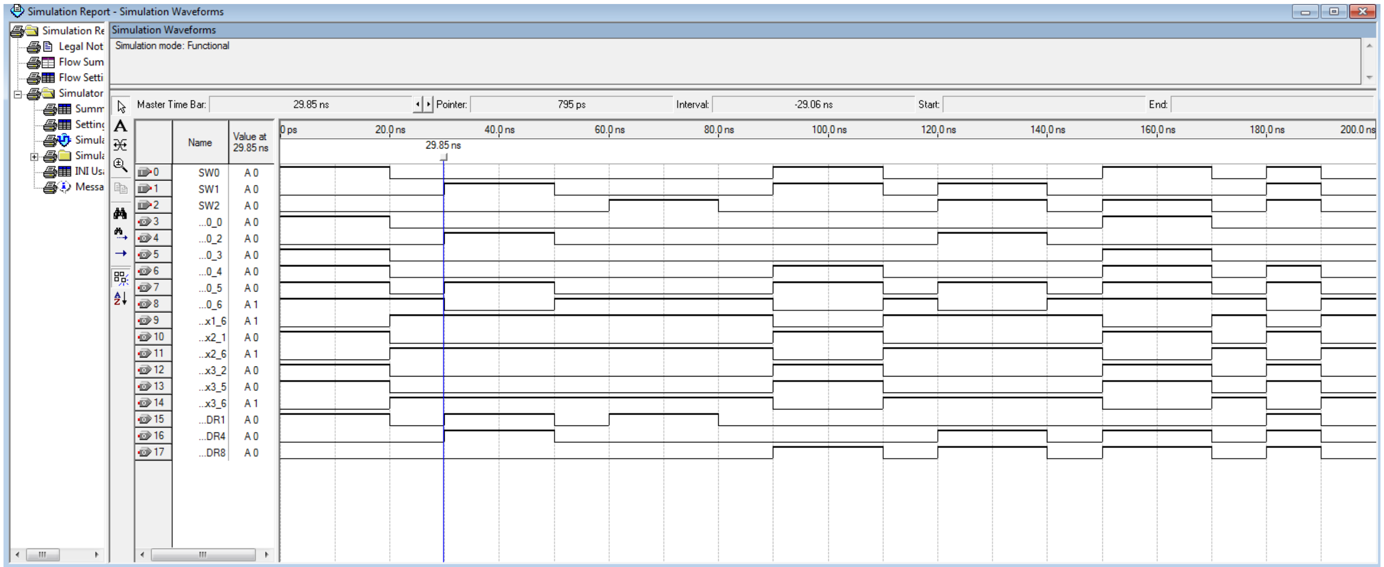
5. For part 5 I used a truth table that had SW1 and SW0 with the different segments in the 7 segment LED display of HEX0 as outputs so I knew which ones needed to be turned off or on. This gave me six equations so I just implemented them as logic gates. The equations are: 0 = SW1`SW0 2 = SW1SW0` 3 = SW1`SW0 4 = (SW1`SW0) + (SW1SW0) 5 = (SW1`SW0) + (SW1SW0`) + (SW1SW0) 6 = (SW1`SW0`) + (SW1`SW0)

SW1	SW0	HEX0_1	HEX0_2	HEX0_3	HEX0_4	HEX0_5	HEX0_6
0	0	0	0	0	0	0	1
0	1	1	0	1	1	1	1
1	0	0	1	0	0	1	0
1	1	0	0	0	1	1	0

B)

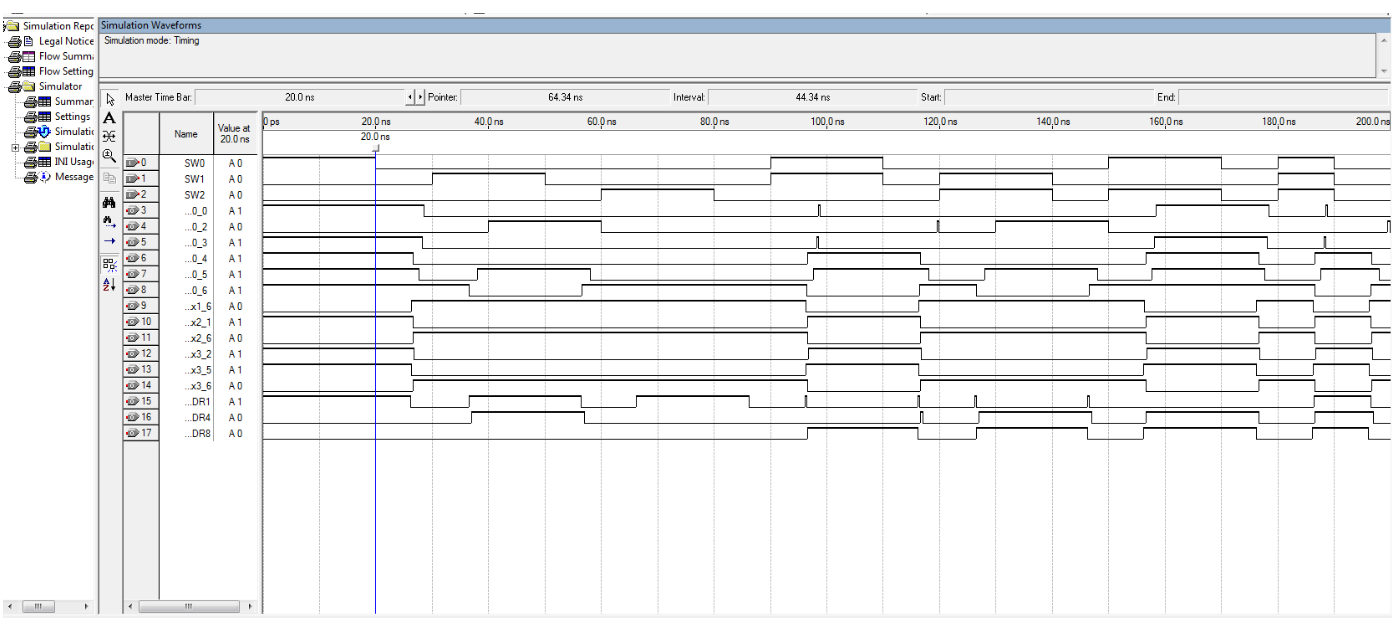


C)

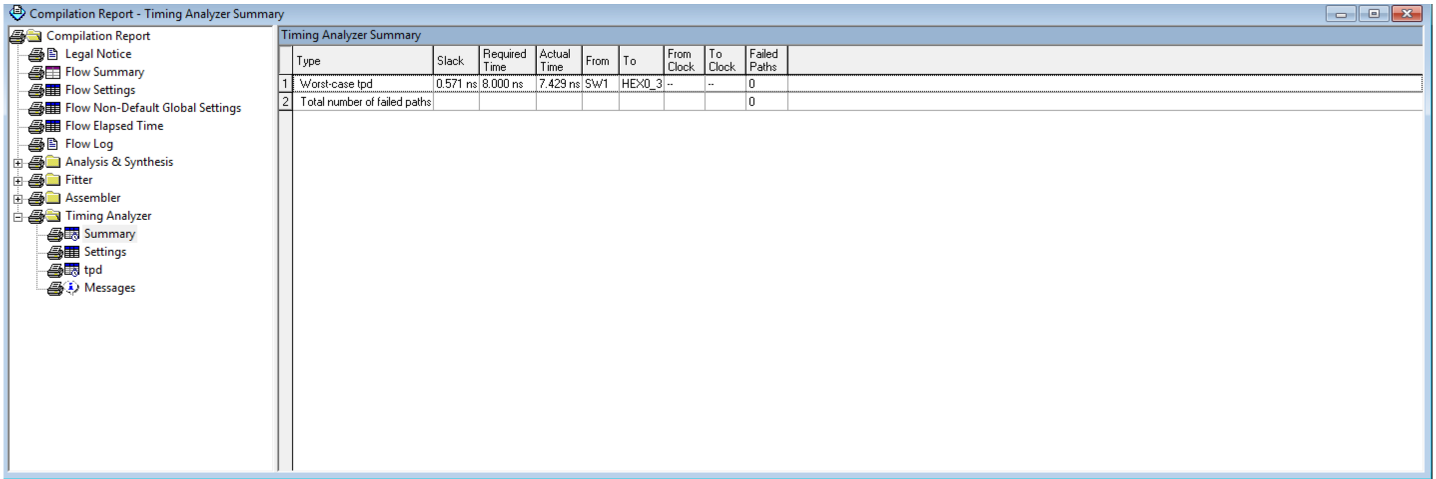


D)

|___| ← 8 NS delay



E)



	Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
1	Worst-case tpd	0.571 ns	8.000 ns	7.429 ns	SW1	HEX0_3	--	--	0
2	Total number of failed paths								0