Student ID: ____________

ICS 151
Final

Name : __________________ , __________________
(Last Name) (First Name)

Student ID : _______________

Signature : ________________

Instructions:

1. Please verify that your paper contains **19 pages** including this cover and 3 blank pages.
2. Write down your Student-Id on the top of each page of this quiz.
3. This exam is **closed book**. No notes or other materials are permitted.
4. Total credits of this quiz are **110 points.**
5. To receive the whole credit you must show your work clearly.
6. Calculators are **NOT** allowed.

	Q1	Q2	Q3	Q4	Total
Credit	40	30	10	30	110
Score					

Student ID: ____________

Q1: Combinational Logic Design & Optimization [40 points]

For function F(x, y, z) = x'y'z + x'yz + x'yz' + xyz' + xy'z

We want to design a circuit to implement function F(x,y,z) using three different methods.

First design is called **design MUX** . Function F(x,y,z) is implemented using Multiplexer. Second design is **design DEC**. F(x,y,z) is designed using Decoder. Then we want to optimize the function F(x, y, z), called **design OPT**, by two-level logic optimization using K-map and Implicants.

Finally, we want to compare three designs (**design MUX**, **design DEC**, and **design OPT**) in terms of area (number of 2-input gates) and critical path delay (number of 2-input gate levels). For computation of area and delay, refer to the following table:

Component	**Area (number of 2-input gates)**	**Delay (number of 2-input gate levels)**
2-to-1 Mux	5	2
4-to-1 Mux	11	4
3-to-8 Decoder	16	3
N-input gate (N>2)	(N-1)	(N-1)

[Note 1: Assume that the area and delay of inverter is negligible and is not considered. Also any link overhead is ignored.]

[Note 2: You are NOT supposed to minimize the function F for **design MUX** and **design DEC**]

Student ID: ____________

(a) Create the truth table (5 points)

(b) [**design MUX**] Implement F by means of only 4-to-1 Mux and 2-to-1 Mux. [Hint: You may use multiple Muxes in a hierarchical method] (5 points)

(c) [**design DEC**] Implement F by means of a 3-to-8 Decoder (5 points)

(d) Draw the K-map for this function F (5 points)

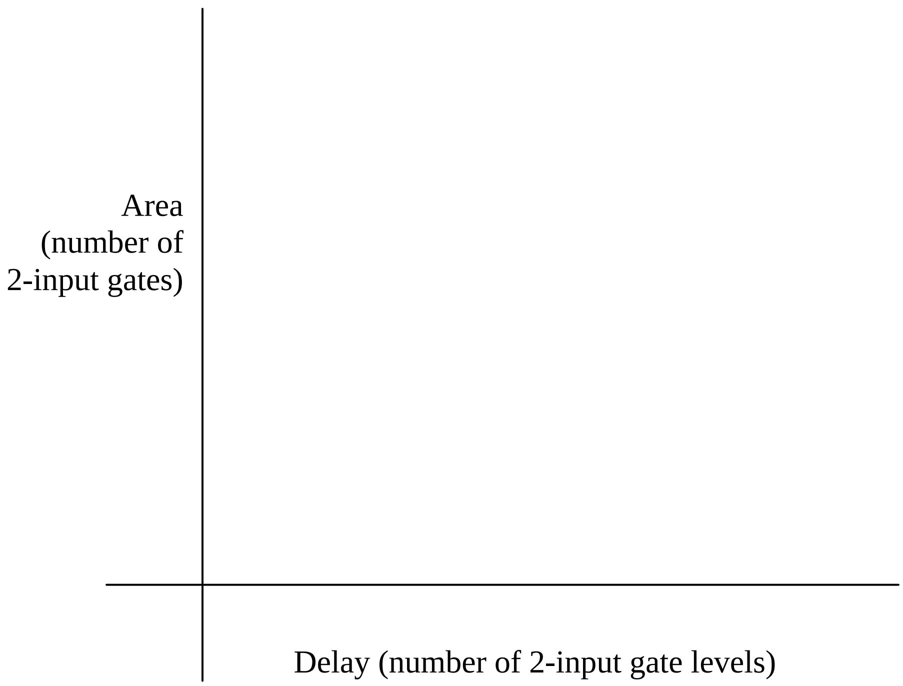
(e) Show all the "Prime Implicants" and "Essential Prime Implicants" from (d) (5 points)

(f) [**design <u>OPT</u>**] Simplify the function F(x, y, z) (5 points)

(g) Calculate the area (number of 2-input gates) and critical path delay (number of 2-input gate levels) for each design (**design** **<u>MUX</u>**, **design** **<u>DEC</u>**, and **design** **<u>OPT</u>**), and place each design with <u>correct values</u> of area and delay at the following graph (10 points)

Which design gives the best delay?

Which design gives the best area?



Student ID: ____________

Q2: FSM Design – Moore and Mealy Machines [30 points]

We want to design a non-resetting sequence detector using a finite state machine with one input X and one output Y. The FSM asserts its output Y when it recognizes the following input bit sequence: "**1101**". The machine will keep checking for the proper bit sequence and does not reset to the initial state after it has recognized the string.
[Note: As an example the input string X= "..1101101.." will cause the output to go high twice: Y = "..0001001.."]

(a) Capture the Moore FSM. (5 points)

(b) Create the architecture (3 points)

(c) Encode the states (use a simple binary encoding) (2 points)

(d) Create the state table (5 points)

(e) Implement the combinational logic (**NO NEED** to draw the gates, just write the equations) (5 points)

(f) Convert the Moore FSM into the Mealy FSM (10 points)

Q3: State Minimization [10 points]

Reduce the number of states in the following state table using **the implication table method** and tabulate the reduced state table.

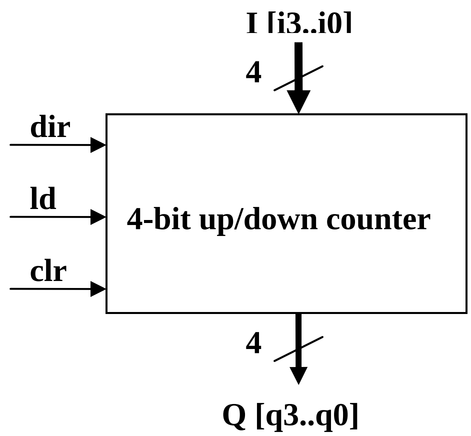
[Note: this state table is for a **Mealy** machine.]

Present state	Next State		Output	
	X=0	X=1	X=0	X=1
A	F	E	0	0
B	C	D	0	0
C	F	B	0	0
D	A	G	0	1
E	C	D	0	0
F	F	B	1	1
G	G	H	0	1
H	A	G	0	1

Student ID: ____________

Q4: RTL Design [30 points]

Using RTL design method, we want to design a 4-bit up/down counter as shown in the following block diagram, which can count either up or down. It requires an input signal **dir** to indicate the count direction, an input signal **ld** to load the input data from I [i3..i0], and an input signal **clr** to reset the counter. We'll let "**dir**=0" mean to count up and "**dir**=1" mean to count down. If "**ld** = 1", it loads from input data I [i3..i0] regardless of the value of **dir**. When "**clr** = 1", it clears the output Q [q3..q0] regardless of the value of **dir** or **ld**.



(a) Capture a high-level state machine (10 points)

(b) Create the datapath using the following components (10 points)

- 4-bit Register
- 2 to1 Mux
- 4-bit Adder

(c) Connect the datapath to the controller (5 points)

(d) Derive the controller's FSM (Moore machine) (5 points)