COE 561, Term 081

Digital System Design and Synthesis

HW# 4

Due date: Tuesday, Jan. 27

Q.1. Consider a technology library containing the following cells:

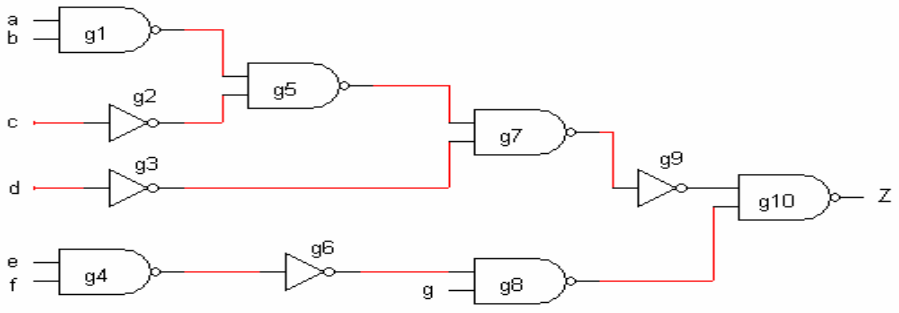
Cell	Area Cost
INV(x1) = x1'	1
NAND2(x1, x2) = (x1 x2)'	2
NAND3(x1, x2, x3) = (x1 x2 x3)'	3
NOR2(x1, x2) = (x1 + x2)'	2
AOI21(x1, x2, x3) = ((x1 x2) + x3)'	3
OAI21(x1, x2, x3) = ((x1+x2) x3)'	3

(i) Show the **pattern trees** of the library cells using **NAND2** and **INV** as base functions. Assume that symmetric representations do not need to be stored.

(ii) Decompose the function f = a + b' + c using NAND2 and INV as base functions into all possible **non-symmetric decomposition**s. Then, **map** the decomposed circuits using the given library and determine the decomposition that leads to a **lower area cost**.

(iii) Using the dynamic programming approach, **map** the circuit given below using the given library into the **minimum area** cost solution. Inputs are *{a, b, c, d, e, f, g}* and output is *{Z}*.

(iv) Using the given library, use the SIS command ***read_libray*** **q1.lib** to read the library. Then, map the circuit to the library using the sis command ***map –s –m 0***. Compare your solution to the solution obtained in (iii). You can save the mapped circuit using the sis command ***write_blif –n***. Why do you think the solution obtained by SIS is better than your solution?



(v) Assuming **Boolean matching**, determine the number of ROBDD's that need to be stored in the cell library for each of the following cells. Justify your answer.

a. f = a b + a c + b c
b. f = a b c + a' b' d

Q.2. Consider the incompletely-specified FSM that has 6 states, two inputs and one output, represented by the following state table:

Present State	**Next State, Output**			
	00	**01**	**11**	**10**
S1	S2, 0	–, –	S5, 1	–, –
S2	S1, 0	S3, –	–, –	–, –
S3	S3, –	S1, 1	S5, –	S4, 1
S4	–, –	S2, –	S1, –	–, –
S5	S3, –	S2, 1	–, 0	S6, –
S6	S6, 1	S1, –	S2, –	S5, –

(i) Determine the incompatible and the compatible states along with their implied pairs.

(ii) Compute the maximal compatible classes along with their implied state pairs.

(iii) Reduce the state table into the minimum number of states and show the reduced state table.

(iv) Apply the SIS command ***state_minmize*** and then save the minimized state table using the command ***write_kiss***. Compare your solution to that obtained by SIS.

Q.3. Consider the incompletely-specified FSM that has 4 states, two inputs and two outputs, represented by the following state table:

Product	**Input**	**Present State**	**Next State**	**Output**
P1	10	S1	S2	11
P2	00	S2	S2	11
P3	01	S2	S2	00
P4	00	S3	S2	00
P5	10	S2	S1	11
P6	10	S3	S1	11
P7	00	S1	S1	– –
P8	01	S3	S0	00
P9	11	S1	S1	10
P10	11	S3	S3	01
P11	11	S0	S0	11

(i) Assuming the following constraints, S0 = S1 OR S3, and that the code of S2 is covered by all other state codes, the state table can be reduced into the table shown below. Using implicant merging, covering and disjunctive relations show step by step how you can obtain the reduced state stable given below.

Input	Present State	Next State	Output
–0	S1, S2	S2	11
10	S2, S3	S1	11
00	S1	S1	– –
01	S3	S0	00
11	S0, S1	S1	10
11	S0, S3	S3	01

(ii) Show the encoding constraint matrix and compute all the seed dichotomies of the encoding constraint matrix. Then, eliminate seed dichotomies that violate the given covering and disjunctive constraints.

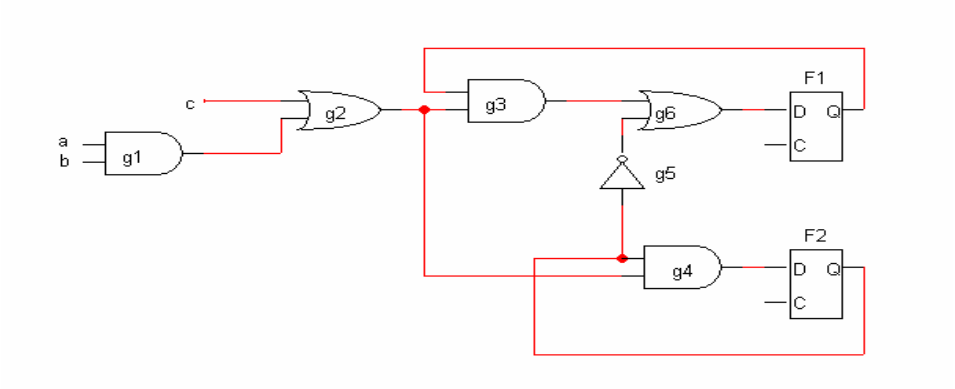
(iii) Compute all the prime dichotomies and eliminate those that violate the disjunctive constraints.

(iv) Find a state encoding satisfying the given constraints. Verify that your encoding satisfies all the constraints.

(v) Using K-MAP, obtain the equations for the output and flip-flops. Compare your solution to the solution obtained by running the SIS command ***stg_to_network*** using the state codes obtained in (iii).

(vi) Perform state assignment using the program nova by running the SIS command ***state_assign nova***. Compare the obtained solution to your solution in (v) in terms of number of literals.

Q.4. Consider the following circuit with inputs {a, b, c} and outputs {F1, F2}. Assume that the delay of an Inverter is 1 unit delay, the delay of a 2-input AND gate is 2 unit delays, and the delay of a 2-input OR gate is 2 unit delays. Consider the circuit given below:



(i) Determine the critical path of this circuit and the maximum propagation delay.

(ii) Using only the **Retiming** transformation, reduce the critical path of this circuit with the minimum number of flip-flops possible.

(iii) Read the library **q4.lib** using the command ***read_library q4.lib***. Then, map your design to the library using the command ***map –s***. Then, retime the circuit using the command ***retime -i***. Compare the maximum arrival time before and after retiming. Compare the obtained solution to the solution you obtained in (ii).