Jan. 27, 2011

COMPUTER ENGINEERING DEPARTMENT

COE 561

Digital System Design and Synthesis

Final Exam

(Open Book Exam)

First Semester (101)

Time: 7:00-10:00 PM

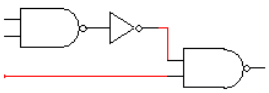
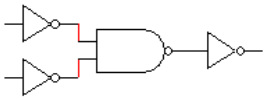
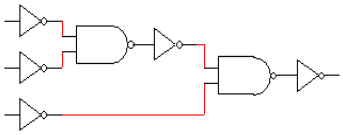
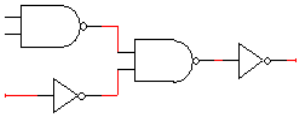
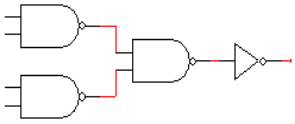
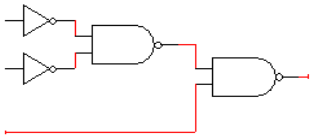
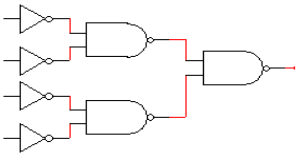
Student Name : KEY_____

Student ID. : _____

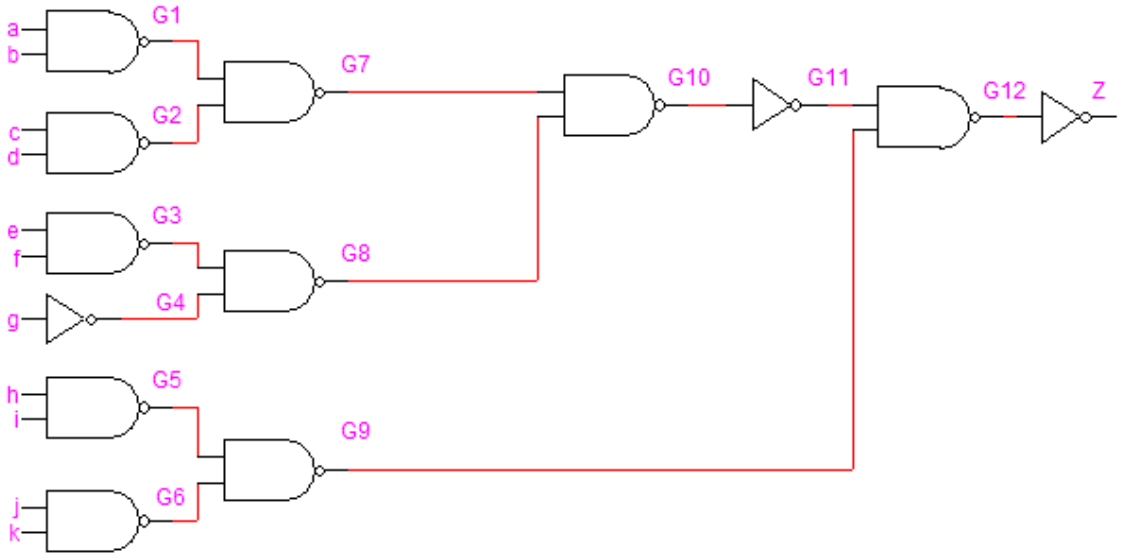
Question	Max Points	Score
Q1	20	
Q2	20	
Q3	23	
Q4	12	
Q5	25	
Total	100	

[20 Points]

(Q1) Consider a technology library containing the following cells:

Cell	Area Cost	Gate
INV(x1) = x1’	1	
NAND2(x1, x2) = (x1 x2)’	2	
NAND3(x1, x2, x3) = (x1 x2 x3)’	3	
NOR2(x1, x2) = (x1 + x2)’	2	
NOR3(x1, x2, x3) = (x1 + x2 + x3)’	3	
AOI21(x1, x2, x3) = ((x1 x2) + x3)’	3	
AOI22(x1, x2, x3, x4) = ((x1 x2) + (x3 x4))’	4	
OAI21(x1, x2, x3) = ((x1+x2) x3)’	3	
OAI22(x1, x2, x3, x4) = ((x1+x2) (x3+x4))’	4	

(i) Consider the circuit given below with inputs *{a, b, c, d, e, f, g, h, i, j, k}* and output *{Z}*. Using the dynamic programming approach and **Structural Matching**, **map** the circuit using the given library into the **minimum area** cost solution.



(ii) Can you obtain a better mapping than the one obtained in (i). If the answer is yes, show the better solution and explain how it is obtained.

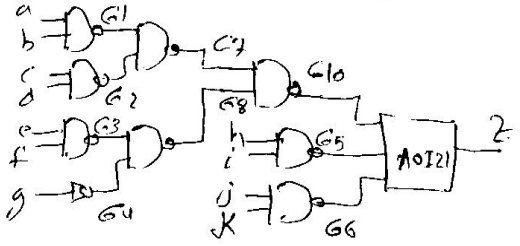
(iii) Assuming **Boolean Matching**, determine the number of ROBDD's that need to be stored in the cell library for the following cell. Justify your answer.

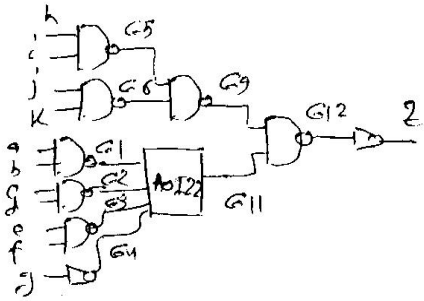
Y = a b' c d' + a' b c + d e f

(1)

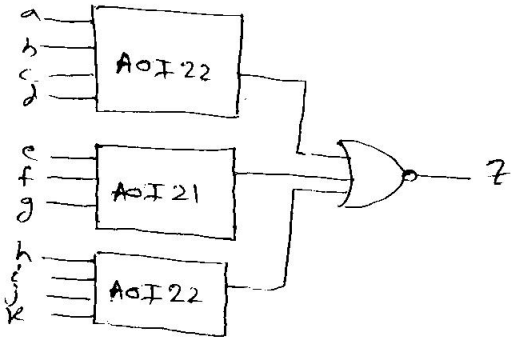
Vertex	gate	cost
G1	Nand2 (a,b)	2
G2	Nand2 (c,d)	2
G3	Nand2 (e,f)	2
G4	INV (g)	1
G5	Nand2 (h,i)	2
G6	Nand2 (j,k)	2.
G7	Nand2 (G1, G2)	2+2+2 = 6
G8	Nand2 (G3, G4)	2+1+2 = 5
G9	Nand2 (G5, G6)	2+2+2 = 6
G10	Nand2 (G7, G8)	6+5+2 = 13
G11	INV (G10)	13+1 = 14
	AOI22(G1, G2, G3, G4)	2+2+2+1+4 = 11
G12	Nand2 (G11, G9)	11+6+2 = 19
	Nand3 (G7, G8, G9)	6+5+6+3 = 20
Z	INV (G12)	19+1 = 20
	AOI21(G10, G5, G6)	13+2+2+3 = 20

Thus, there are two minimal solutions with the same cost of 20 as follows:





(ii) Yes, a better mapping can be obtained by inserting a pair of inverters at the outputs of gates G7, G8 and G9 before mapping resulting in the following mapping with an area cost of 14.



(iii) $c1 = \{(a)^b, (b)^b, (c)^{\mu}, (d)^b\}$

$c2 = \{(e, f)^{\mu}\}$

This, the number of ROBDD's needed to be stored in the cell library is $3! = 6$

[20 Points]

(Q2) Consider the incompletely-specified FSM that has 5 states, one input (X) and two outputs (Z1Z0), represented by the following state table:

Present State	Next State		Output Z1Z0
	X=0	X=1	
S0	S0	–	–
S1	S2	S1	1 0
S2	S3	–	0 1
S3	–	S1	–
S4	S0	S2	0 1

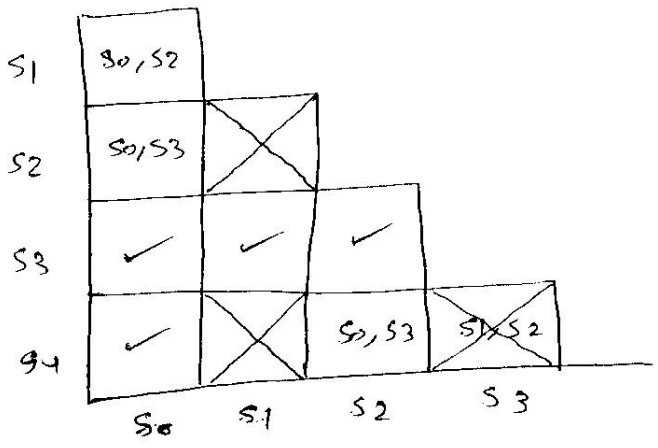
(i) Determine the incompatible states and the compatible states along with their implied pairs.

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

(iii) Compute the prime compatibility classes along with their implied state pairs.

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

(i) Compatibility Table :



The incompatible states are :

(S1, S2) , (S1, S4) , (S3, S4)

The compatible states with their implied pairs:

$(S_0, S_1) \Leftarrow (S_0, S_2)$
$(S_0, S_2) \Leftarrow (S_0, S_3)$
(S_0, S_3)
(S_0, S_4)
(S_1, S_3)
(S_2, S_3)
$(S_2, S_4) \Leftarrow (S_0, S_3)$

(ii) Maximal Compatible Classes:

From the incompatible state pairs we have:

$$(\bar{S}_1 + \bar{S}_2)(\bar{S}_1 + \bar{S}_4)(\bar{S}_3 + \bar{S}_4)$$
$$= (\bar{S}_1 + \bar{S}_2\bar{S}_4)(\bar{S}_3 + \bar{S}_4)$$
$$= \bar{S}_1\bar{S}_3 + \bar{S}_1\bar{S}_4 + \bar{S}_2\bar{S}_3\bar{S}_4 + \bar{S}_2\bar{S}_4$$
$$= \bar{S}_1\bar{S}_3 + \bar{S}_1\bar{S}_4 + \bar{S}_2\bar{S}_4$$

Thus, the maximal compatible classes along with their implied pairs are:

$(S_0, S_2, S_4) \Leftarrow (S_0, S_3)$
(S_0, S_2, S_3)
$(S_0, S_1, S_3) \Leftarrow (S_0, S_2)$

(iii) Prime Compatibility Classes:

In addition to the maximal compatible classes, we have the following prime classes:

(S0, S4)

(S1, S3)

(iv) The following minimum cover can be used which satisfies the closure:

{ (S0, S2, S4), (S0, S1, S3) }

Thus, the state machine can be reduced to two states as follows:

P.S.	Next State X = 0	Next State X = 1	output Z1 Z0
S0, 2, 4	S0, 1, 3	S0, 2, 4	0 1
S0, 1, 3	S0, 2, 4	S0, 1, 3	1 0

[23 Points]

(Q3) Consider the given FSM which has 6 states, one input and one output, represented by the following state table:

Product	Input	Present State	Next State	Output
P1	0	S0	S2	0
P2	1	S0	S4	0
P3	0	S1	S2	0
P4	1	S1	S2	0
P5	0	S2	S3	1
P6	1	S2	S2	0
P7	0	S3	S1	0
P8	1	S3	S0	0
P9	0	S4	S1	0
P10	1	S4	S0	0
P11	0	S5	S5	0
P12	1	S5	S4	0

(i) Assuming the following constraints: S3 covers S2, S4 covers S2 and that the code of S0 is covered by all other state codes, the state table can be reduced into the table given below. Using implicant merging and covering relations show step by step how you can obtain the reduced state stable given below:

Input	Present State	Next State	Output
–	S0, S1, S2	S2	0
0	S3, S4	S1	0
1	S0, S5	S4	0
0	S2	S3	1
0	S5	S5	0

(ii) Compute all the seed dichotomies and construct their conflict graph. Find a minimum cover for the seed dichotomies. Based on the found cover, derive an encoding satisfying the given constraints with minimal bit length.

(i) P1 and P3 can be merged using implicant merging into the following row:

r1 0 S0, S1 S2 0

P4 and P6 can be merged using implicant merging into the following row:

r2 1 S1, S2 S2 0

Since in P5, 0 S2 S3 1, and S3 covers S2, we can add S2 to r1 resulting in:

r3 0 S0, S1, S2 S2 0

Since in P2, 1 S0 S4 0, and S4 covers S2, we can add S0 to r2 resulting in:

r4 1 S0, S1, S2 S2 0

Next, r3 and r4 can be merged using implicant merging into the following row;

r5 — S0, S1, S2 S2 0

P7 and P9 can be merged into the following row:

r6 0 S3, S4 S1 0

P2 and P12 can be merged into the following row:

r7 1 S0, S5 S4 0

P8 and P10 can be removed since S0 is covered by all states since its code will be all 0's and since the output is 0.

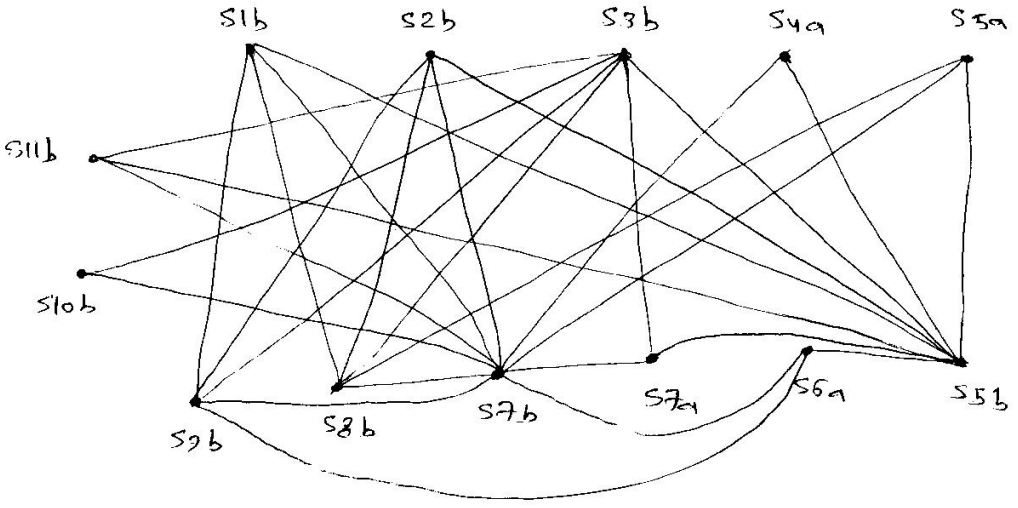
This results in the given reduced table.

(ii) Seed Dichotomies

X s1a : (s0, s1, s2), (s3)
X s2a : (s0, s1, s2), (s4)
X s3a : (s0, s1, s2), (s5)
s4a : (s3, s4), (s0)
s5a : (s3, s4), (s1)
s6a : (s3, s4), (s2)
s7a : (s3, s4), (s5)
X s8a : (s0, s5), (s1)
X s9a : (s0, s5), (s2)
X s10a : (s0, s5), (s3)
X s11a : (s0, s5), (s4)

s1b : (s3), (s0, s1, s2)
s2b : (s4), (s0, s1, s2)
s3b : (s5), (s0, s1, s2)
X s4b : (s0), (s3, s4)
s5b : (s1), (s3, s4)
X s6b : (s2), (s3, s4)
s7b : (s5), (s3, s4)
s8b : (s1), (s0, s5)
s9b : (s2), (s0, s5)
s10b : (s3), (s0, s5)
s11b : (s4), (s0, s5)

Conflict Graph



Based on the conflict graph, a minimum cover can be found as follows since these nodes can be covered using two colors:

P1 : s1b (S3), (S0, S1, S2)
s2b (S4), (S0, S1, S2)
s3b (S5), (S0, S1, S2)
s4a (S3, S4), (S0)
s5a (S3, S4), (S1)
s6a (S3, S4), (S2)
$\Rightarrow$ (S3, S4, S5), (S0, S1, S2)

P2 : s7a (S3, S4), (S5)
s8b (S1), (S0, S5)
s9b (S2), (S0, S5)
s10b (S3), (S0, S5)
s11b (S4), (S0, S5)
$\Rightarrow$ (S1, S2, S3, S4), (S0, S5)

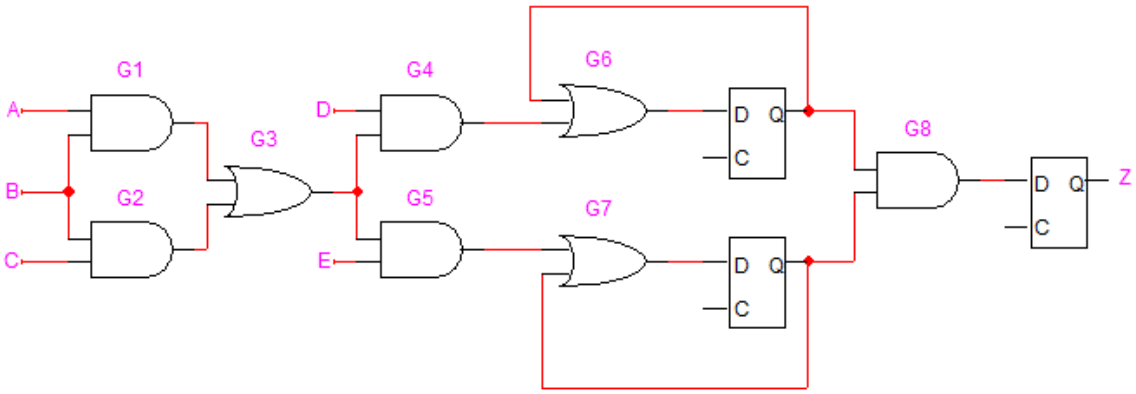
Thus, we obtain the following encoding satisfying the given constraints:

	P1	P2	
S0	0	0	0
S1	0	1	1
S2	0	1	0
S3	1	1	0
S4	1	1	1
S5	1	0	0

Additional column to differentiate between the codes of S2 & S1 and S3 & S4.

[12 Points]

(Q4) Consider the sequential circuit given below having 5 inputs {A, B, C, D, E} and one output {Z}. Assume that the delay of a gate is related to the number of inputs i.e. 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.



(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. Determine the maximum propagation delay after retiming.

(i) The maximum propagation delay is 8 and there are 8 critical paths as follows:

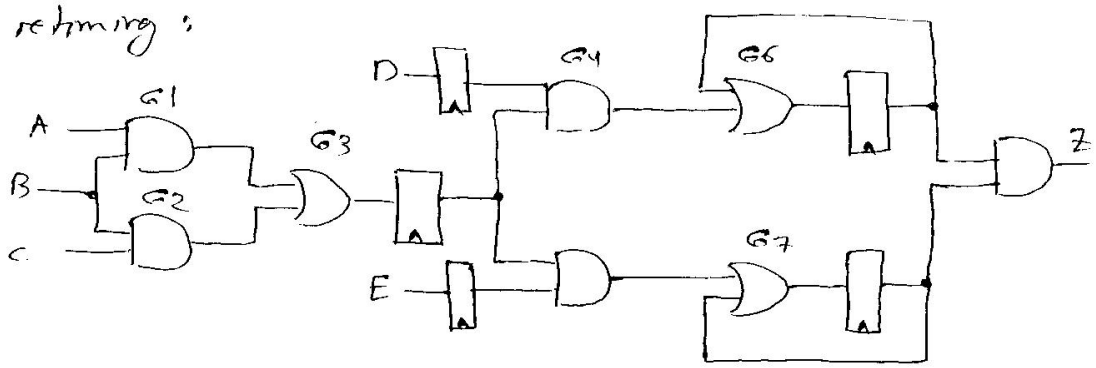
{A, G1, G3, G4, G6}, {A, G1, G3, G5, G7}
{B, G1, G3, G4, G6}, {B, G1, G3, G5, G7}
{B, G2, G3, G4, G6}, {B, G2, G3, G5, G7}
{C, G2, G3, G4, G6}, {C, G2, G3, G5, G7}

(ii) We can apply the following retiming transform. to reduce the critical path:

- retime G6 by +1
- retime G7 by +1
- retime G8 by +1
- retime the stem on fanout of G6 by +1
- retime the stem on fanout of G7 by +1

- retime G4 by +1
- retime G5 by +1
- retime the stem on the fanout of G3 by +1

This results in the following circuit after retiming:



The maximum propagation delay in the resulting circuit is 4. The number of flip-flops has increased from 3 to 5.

[25 Points]

(Q5) Consider the network given below with inputs *{a, b, c, d, e, f}* and output *{y}*:

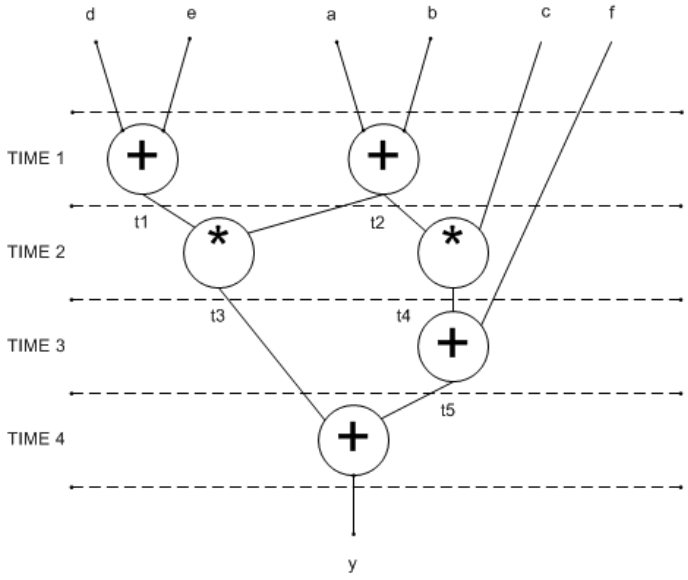
[1]=a+b; [2]=[1]*c; [3]=d+e; [4]=[2]+f;
[5]=[1]*[3]; y=[4]+[5];

(i) Draw the **sequencing graph** for the above network.

(ii) Assuming that the delay of both the Adder and the Multiplier fit within one clock cycle, show the **ASAP** and **ALAP** scheduling of the sequencing graph assuming a latency of **5 clock cycles**. Compute the **mobility** of each operation.

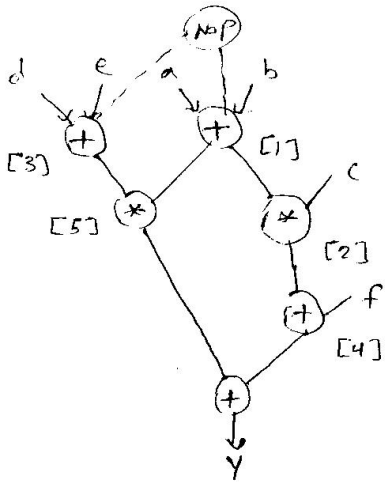
(iii) Using **List Scheduling** for minimum resource usage algorithm LIST_R, schedule the sequencing graph under the latency constraint of **5 clock cycles** minimizing the number of resources required. Show the details of the algorithm step by step and the resulting scheduled sequencing graph.

(iv) Consider the scheduled sequencing graph below assuming that the input values will be available to the circuit for only one clock cycle.

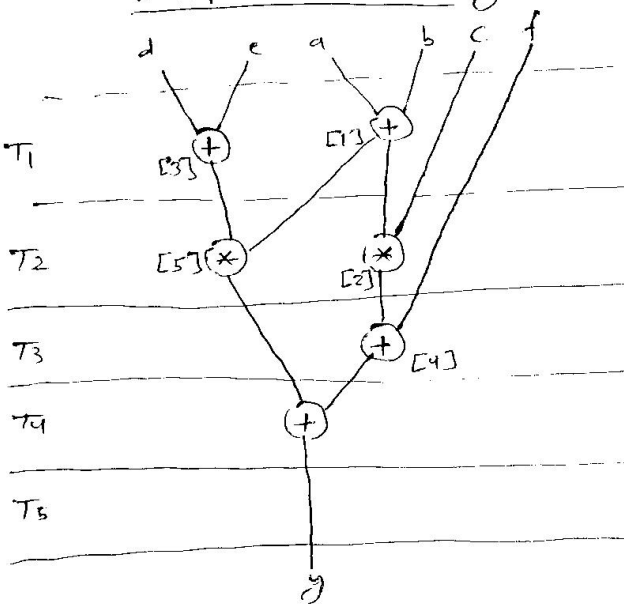


a. Show the life-time of all variables.
b. Determine the minimum number of registers that are required to store all the variables. Show the mapping of variables to registers. Select a mapping that **minimizes the number of multiplexers and interconnect area** as much as possible.
c. Draw the **data-path** implementing the scheduled sequencing graph based on the variable-register mapping that you obtained in (b).

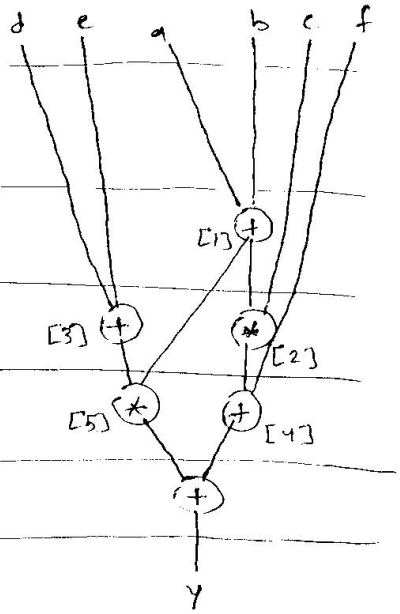
(i) Sequencing Graph :



(ii) ASAP Scheduling



ALAP Scheduling



operation	[1]	[2]	[3]	[4]	[5]	y
mobility	1	1	2	1	2	1

(iii) List scheduling for minimum resources under a latency of 5 clock cycles:

$\lambda = 5$, $a = [1, 1]$

$\underline{l = 1}$

$U_{1,1} = \{\}$

$U_{1,2} = \{1, 3\}$

operations with zero slack $= \{\}$

Since we have one adder available, we can schedule either [1] or [3]. Let us schedule [1], as it has lower slack.

$\underline{l = 2}$

$U_{2,1} = \{2\}$ and we schedule it.

$U_{2,2} = \{3\}$ and we schedule it.

$\underline{l = 3}$

$U_{3,1} = \{5\}$ and we schedule it.

$U_{3,2} = \{4\}$ and we schedule it

$\underline{l = 4}$

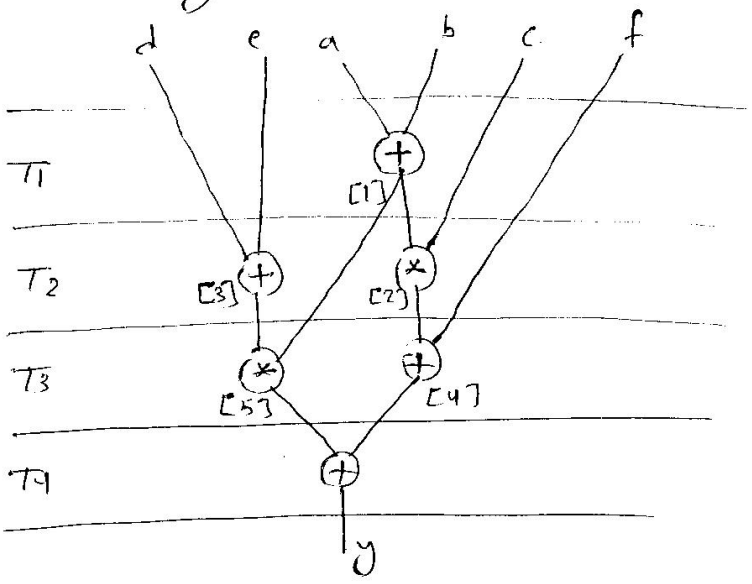
$U_{4,1} = \{\}$

$U_{4,2} = \{y\}$ and we schedule it.

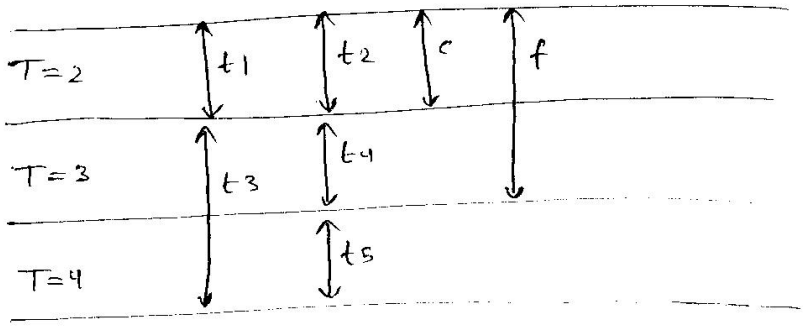
Thus, we can see that it is possible to have a schedule with latency of 4 and using one adder and one multiplier.

Note that if we scheduled [3] in $l = 1$ we will endup with a schedule of latency of 5 cycles.

The resulting scheduled graph is as follows:



(iv) a. Life time of variables:



b. Based on the lifetime of all varrables, it is obvious that 4 registers are needed to store all the variables.

First, we will assign operation [3] to one adder, while operations [1], [4] and y to the other adder.

We will assign varrables to registers as follows:

R1 : t1, t3
R2 : t2, t4, t5
R3 : c
R4 : f

c: Data Path