**COE 561, Term 091**

**Digital System Design and Synthesis**

**HW# 3 Solution**

**Due date: Tuesday, Dec. 29**

# Consider the following function:

*X = ACE + BCE + AC’D’ + BC’D’ + DE + F*

## Compute all the kernels of *X* using the recursive kernel computation algorithm. Show all the steps.

## Compute all the kernels of X based on matrix representation. Compare your answer to the result obtained in (i).

## Find a quick factor of *X* by using the first level-0 kernel found. Assume that input variables are sorted in lexicographic order. Determine the number of literals obtained. Compare your solution with the result obtained by running the sis commands ***factor –q x; print\_factor; print\_stats –f.***

# Consider the following function:

## *X=AC+BC+AD+BD+A’B’C’+A’B’D’+AEF+BEF+AE’F’+BE’F’+A’B’EF’+A’B’E’F*

## Compute all double-cube divisors of *X* along with their bases and their weights. Show only double-cube divisors that have non-empty bases.

## Apply the fast extraction algorithm based on extracting double-cube divisors along with complements or single-cube divisors with two-literals. Show all steps of the algorithm. Determine the number of literals saved. Compare your solution with the result obtained by running the sis commands ***fx***.

# Consider the logic network defined by the following expressions:

*X = AB’ + A’B;*

*Y = XC + A B;*

*Z =Y +A';*

Inputs are {A, B, C} and output is {Z}.

## Compute the SDC set for node X.

## Compute the ODC set for node Y.

## Simplify the function of Y using both its ODC and SDC of node X.

## Compute the ODC set for node X based on the optimized network on (iii).

## Simplify the function of X using its ODC.

## Apply the sis command ***full\_simplify*** and compare the solution obtained with your obtained solution based in (iv).

# Consider the logic network defined by the following expressions:

# 

# e=a b d

# f =c d

# g = e + f

# h = a d

# i =a’ b’ d’

# j = h + i

# k= b d

# l =j + k

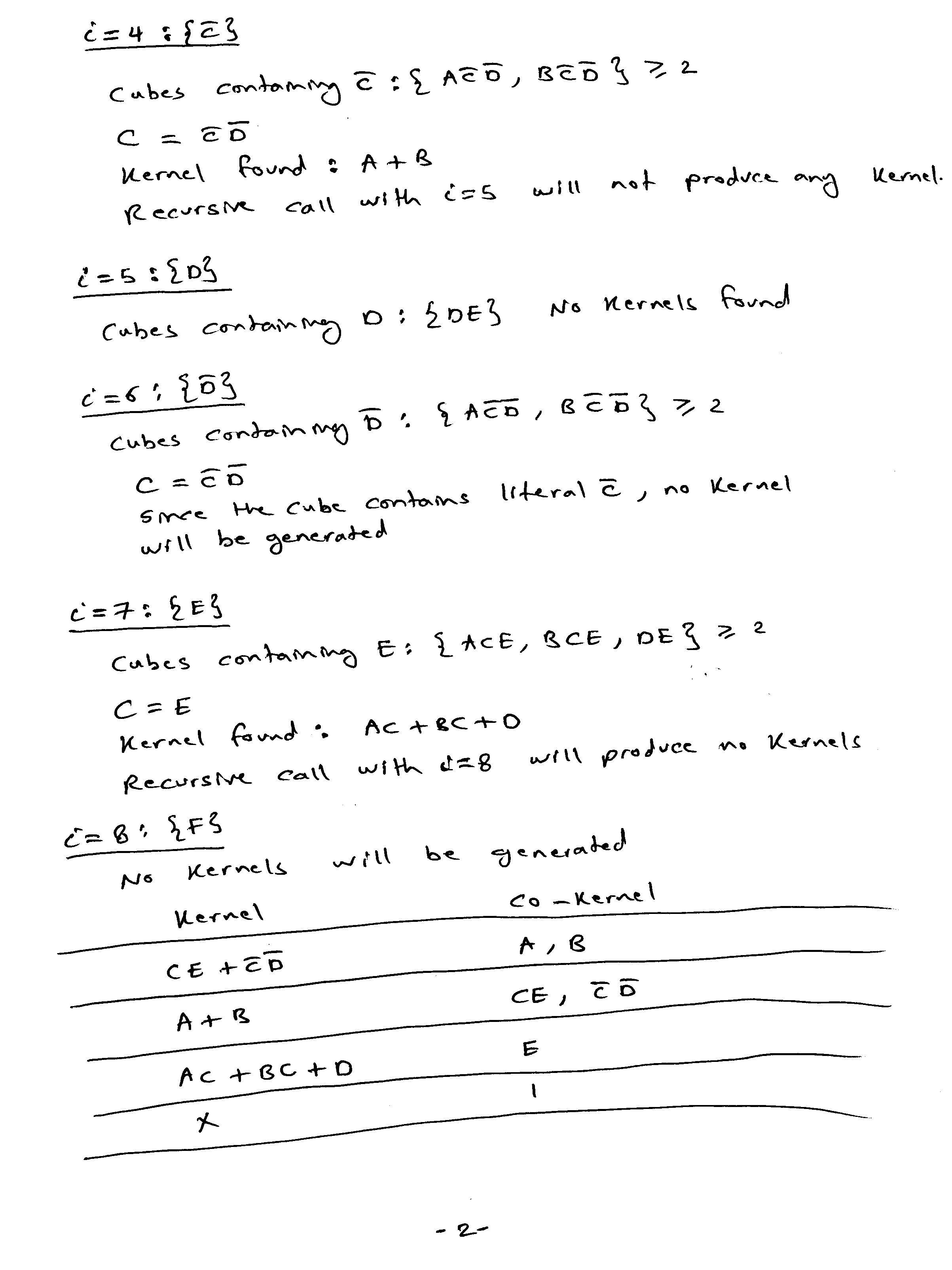
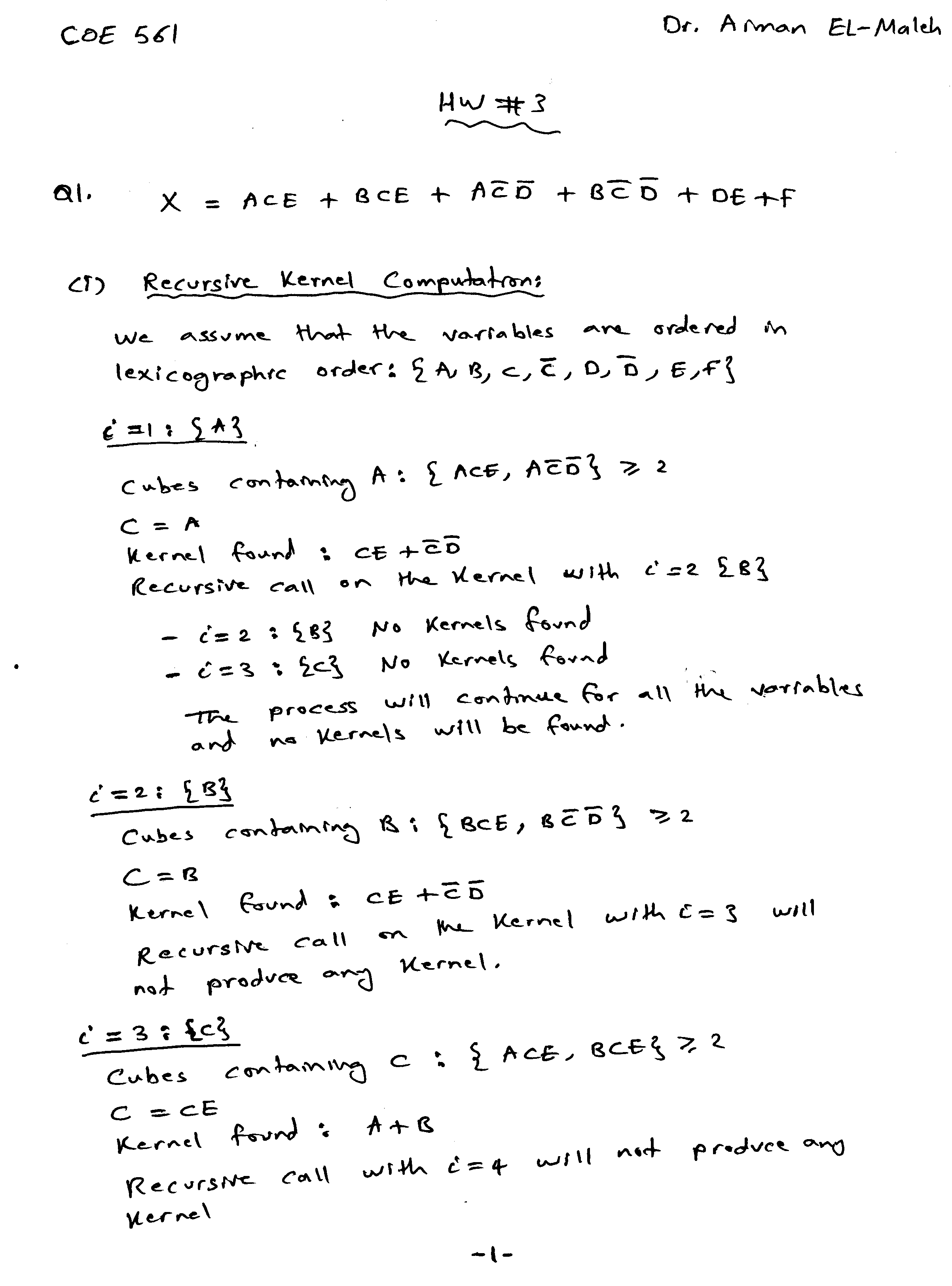
# x =g + l

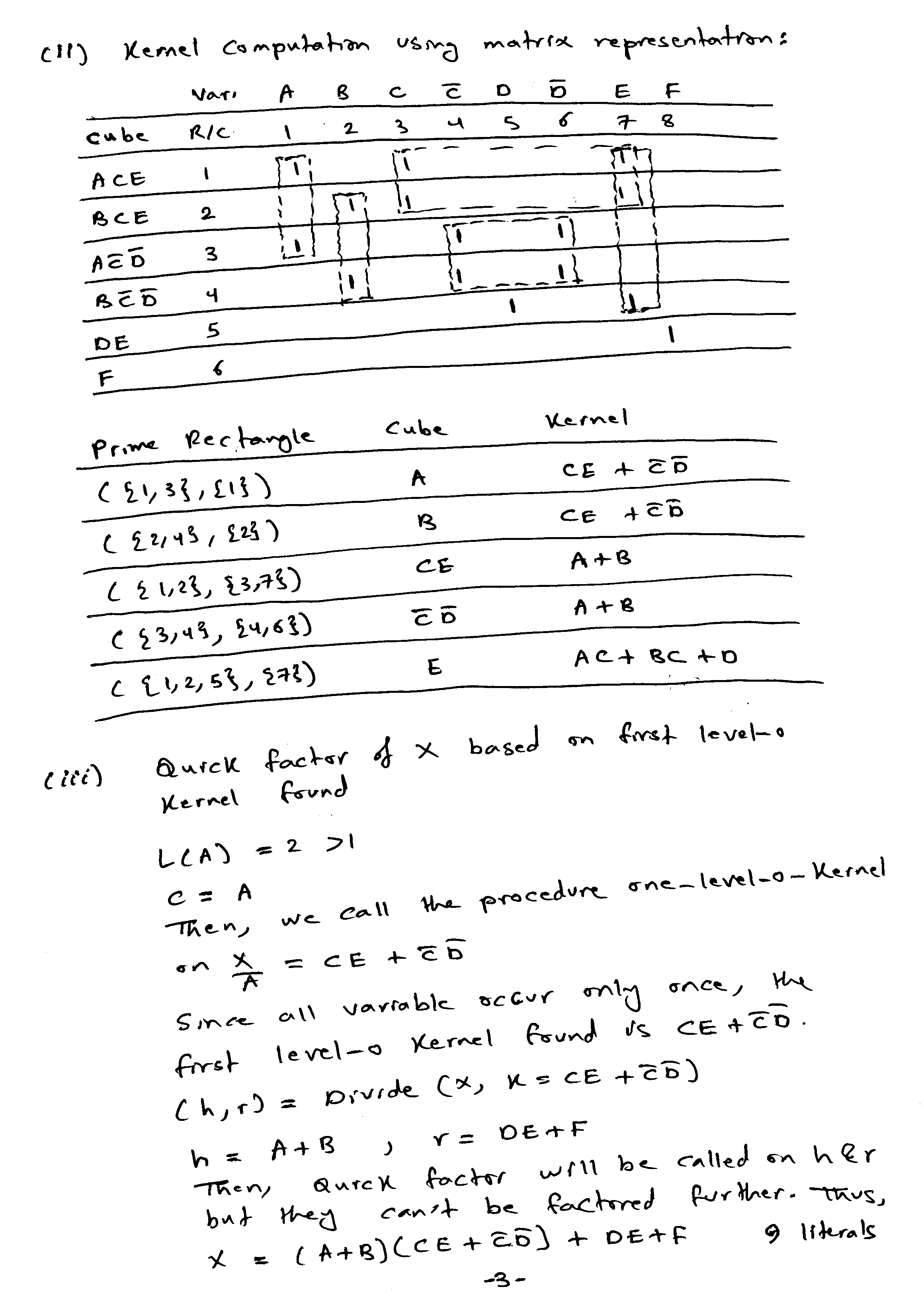
# Inputs are {a, b, c, d} and output is {x}. Assume that the delay of a gate is related to the number of its inputs. Also, assume that the input data-ready times are zero except for input d, which is equal to 2.

## Draw the logic network graph and compute the data ready times and slacks for all vertices in the network.

## Determine the **maximum propagation delay** and the **topological critical path**.

## Suggest an implementation of the function ***x*** to reduce the delay of the circuit. What is the **maximum propagation delay** after the modified implementation?





SIS produces the same quick factor as shown below:

sis> read\_eqn hw3q1.eqn

sis> print

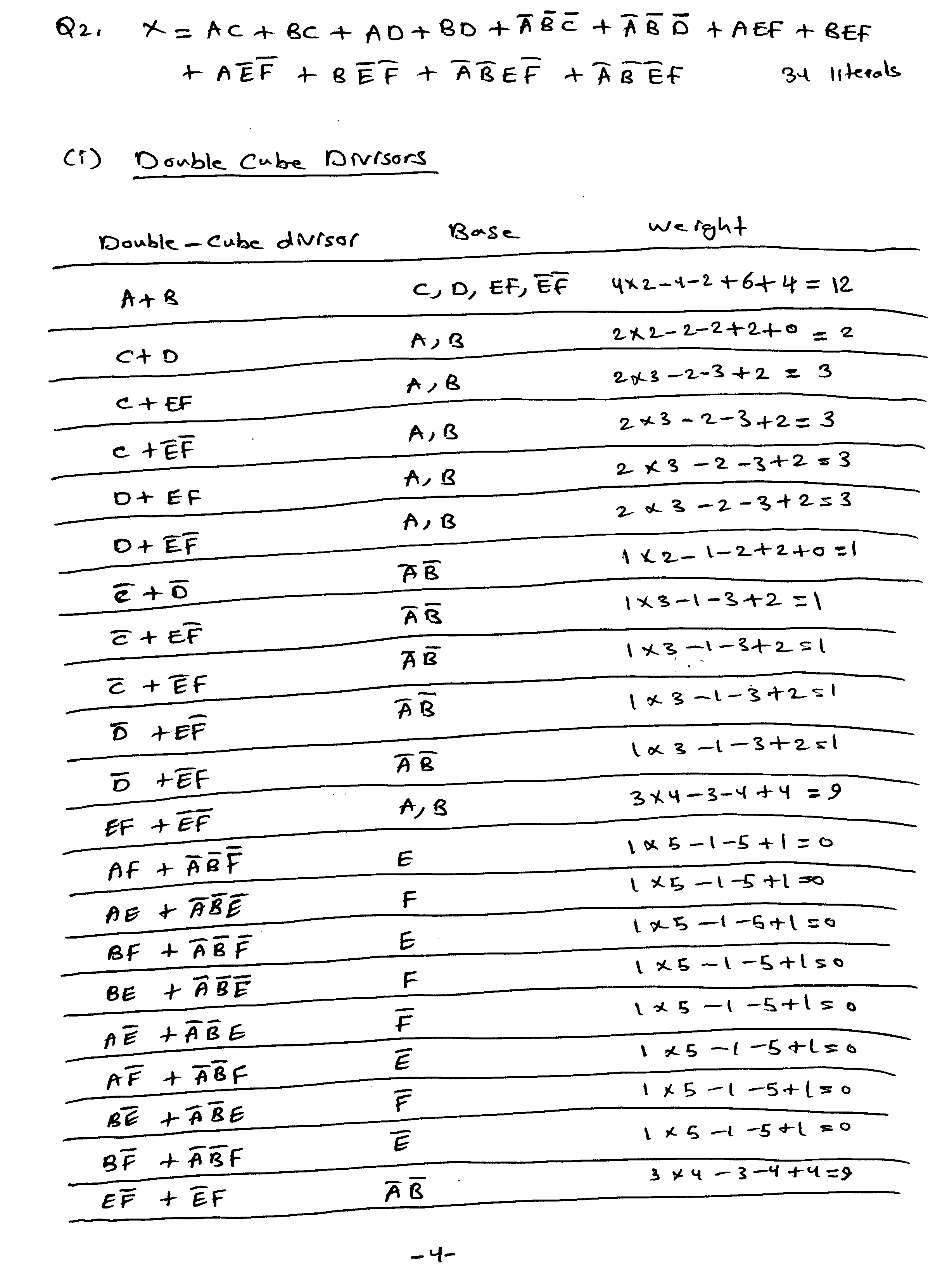
{X} = A C E + A C' D' + B C E + B C' D' + D E + F

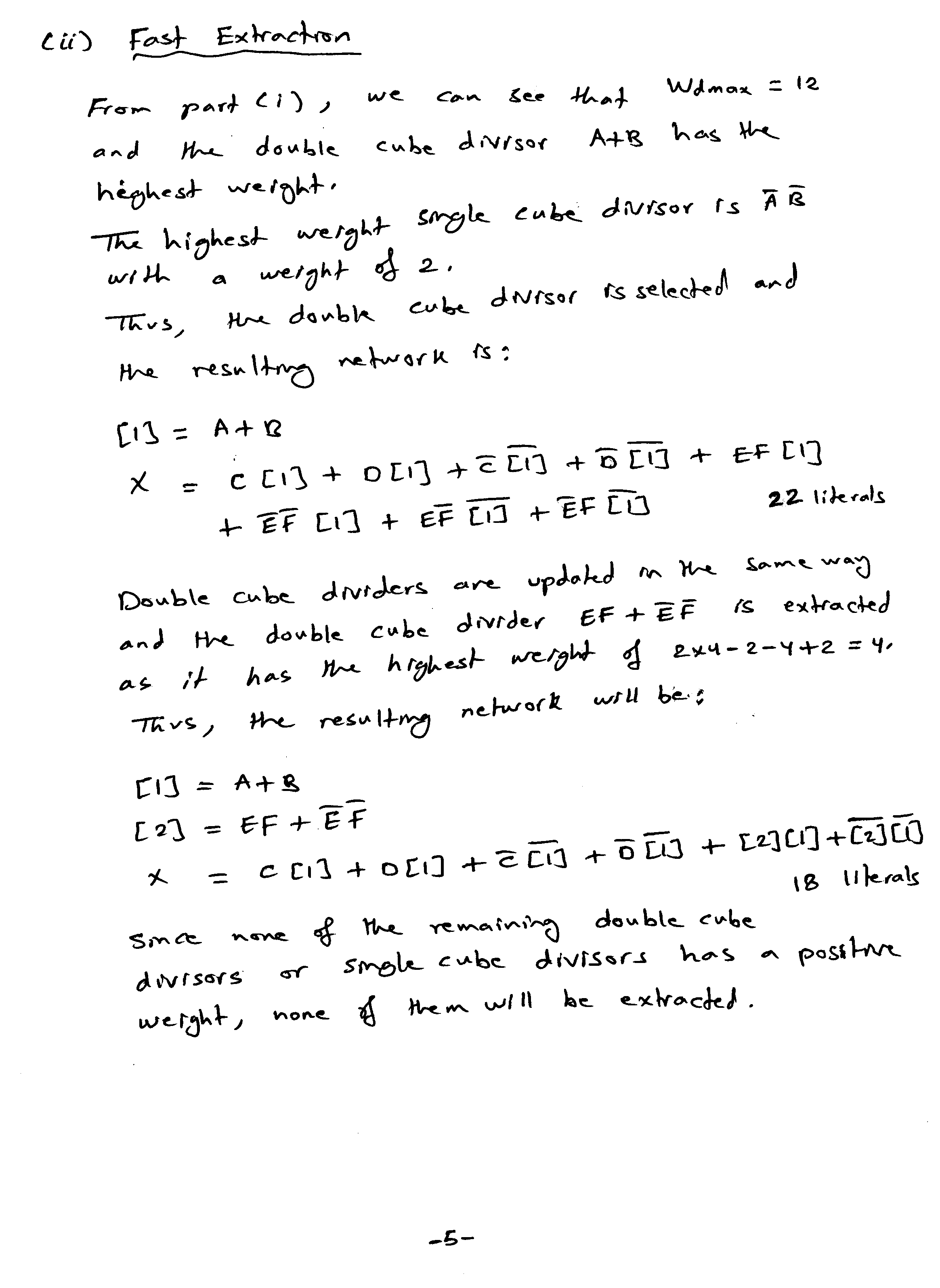
sis> factor -q X; print\_factor; print\_stats -f

{X} = (C E + C' D') (B + A) + D E + F

hw3q1.eqn pi= 6 po= 1 node= 1 latch= 0 lits(sop)= 15 lits(ff)= 9

sis>





SIS has resulted in the same extracted network as shown below:

sis> read\_eqn hw3q2.eqn

sis> print

{X} = A C + A D + A E F + A E' F' + A' B' C' + A' B' D' + A' B' E F' + A' B

' E' F + B C + B D + B E F + B E' F'

sis> print\_stats

hw3q2.eqn pi= 6 po= 1 node= 1 latch= 0 lits(sop)= 34 lits(ff)= 16

sis> fx

sis> print

{X} = C [1] + C' [1]' + D [1] + D' [1]' + [1] [2] + [1]' [2]'

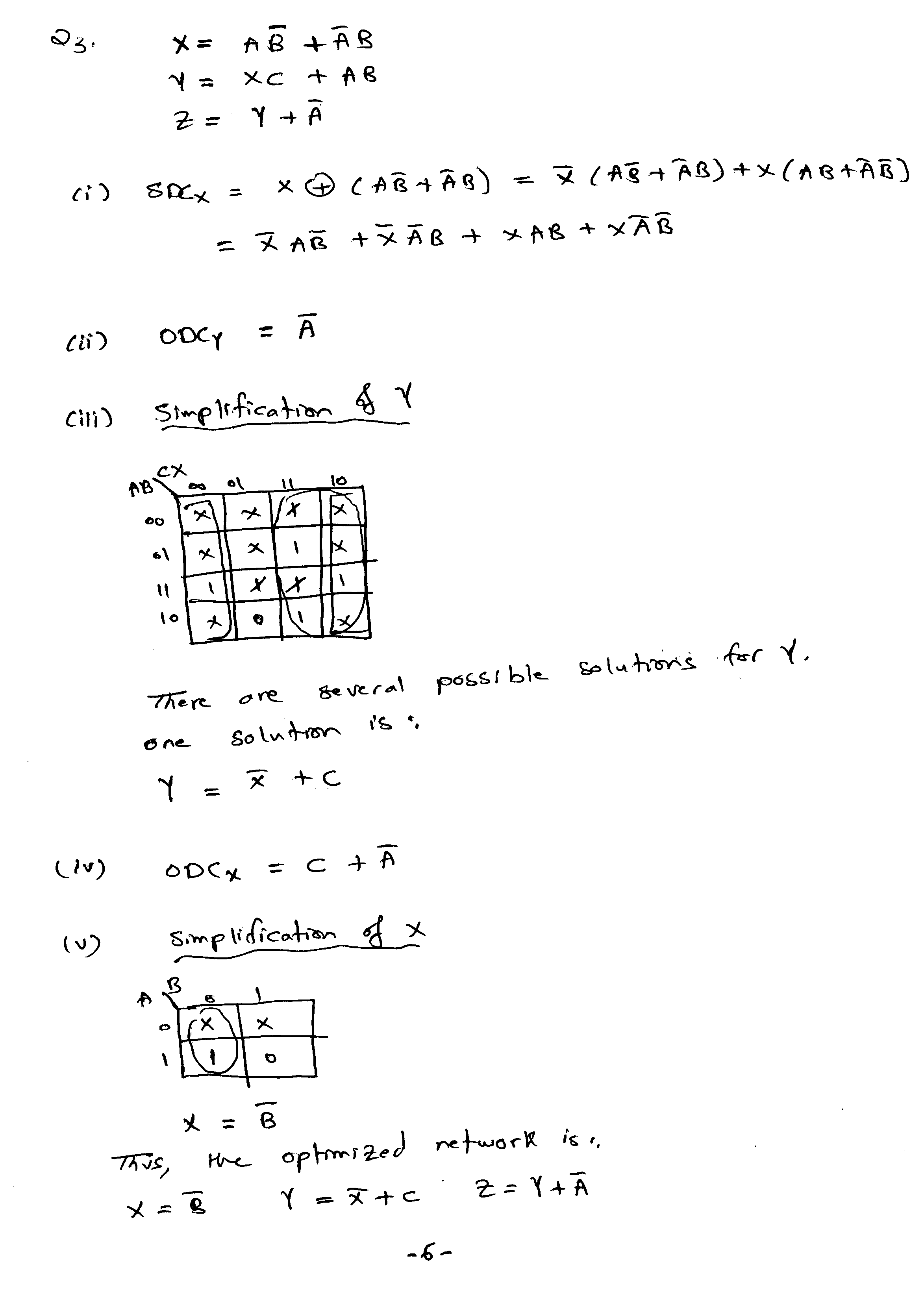
[1] = A + B

[2] = E F + E' F'

sis> print\_stats

hw3q2.eqn pi= 6 po= 1 node= 3 latch= 0 lits(sop)= 18 lits(ff)= 14

sis>



SIS has resulted in the same simplified network as shown below:

sis> read\_eqn hw3q3.eqn

sis> print

{Z} = A' + Y

X = A B' + A' B

Y = A B + C X

sis> print\_stats

hw3q3.eqn pi= 3 po= 1 node= 3 latch= 0 lits(sop)= 10 lits(ff)= 10

sis> full\_simplify

sis> print

{Z} = A' + Y

X = B'

Y = C + X'

sis> print\_stats

hw3q3.eqn pi= 3 po= 1 node= 3 latch= 0 lits(sop)= 5 lits(ff)= 5

sis>

