EXAMPLE OF A FUNCTION MINIMISATION USING **MINILOG**

1. Symbol for the circuit to be designed



1. Truth table

The tbl extension file “HEX\_7SEG\_basic.tbl” (which is a text file to be opened with Notepad or the Minilog text file editor):

 ------------------------------------------------------------------------------------------

table HEX\_7SEG\_basic

input D C B A " Code inputs active high

output a\_L b\_L c\_L d\_L e\_L f\_L g\_L " Segment outputs active low for

 " common anode display

" Truth table for the basic hexadecimal to 7 segment decoder

" INPUTS OUTPUTS

" ========== ============================

" D C B A a\_L b\_L c\_L d\_L e\_L f\_L g\_L

" =========== =========================== Displayed symbol

"

 0 0 0 0 0 0 0 0 0 0 1 " 0

 0 0 0 1 1 0 0 1 1 1 1 " 1

 0 0 1 0 0 0 1 0 0 1 0 " 2 -- a --

 0 0 1 1 0 0 0 0 1 1 0 " 3 | |

 0 1 0 0 1 0 0 1 1 0 0 " 4 f b

 0 1 0 1 0 1 0 0 1 0 0 " 5 | |

 0 1 1 0 0 1 0 0 0 0 0 " 6 -- g --

 0 1 1 1 0 0 0 1 1 1 1 " 7 | |

 1 0 0 0 0 0 0 0 0 0 0 " 8 e c

 1 0 0 1 0 0 0 1 1 0 0 " 9 | |

 1 0 1 0 0 0 0 1 0 0 0 " A -- d --

 1 0 1 1 1 1 0 0 0 0 0 " b

 1 1 0 0 0 1 1 0 0 0 1 " C

 1 1 0 1 1 0 0 0 0 1 0 " d

 1 1 1 0 0 1 1 0 0 0 0 " E

 1 1 1 1 0 1 1 1 0 0 0 " F

End

-------------------------------------------------------------------------------

1. Minimisation

Preparing the minimisation parameters:

- (MOM) Multiple output mode; (the minimisation of all outputs together)

- (SOP) Sum Of Product (method of ones)

- Output format: as a table



Running the minimisation



1. Examining and analysing the output table

Output file “HEX\_7SEG\_basic.min”

 M I N I L O G

 ==============================================================

 = =

 = LOGIC MINIMIZATION PACKAGE version 4.6-ß =

 = =

 = (C)1988...2004 W.M.J. de Valk, Oss, the Netherlands =

 = =

 ==============================================================

FUNCTION TABLE

==============

 ABCDEFG

 \_\_\_\_\_\_\_

DCBA LLLLLLL

==============

0000 | 0000001

0001 | 1001111

0010 | 0010010

0011 | 0000110

0100 | 1001100

0101 | 0100100

0110 | 0100000

0111 | 0001111

1000 | 0000000

1001 | 0001100

1010 | 0001000

1011 | 1100000

1100 | 0110001

1101 | 1000010

1110 | 0110000

1111 | 0111000

CHECKING COMPLETENESS OF INPUT TABLE

MINIMIZING FUNCTION TABLE

MINIMIZATION RESULT STATISTICS

==============================

FOUND 15 ESSENTIAL PRODUCT TERMS

MAXIMUM FANIN: 15

TOTAL LITERAL COUNT: 82

MAXIMUM PRODUCT TERM SIZE: 4

MAXIMUM OUTPUT FUNCTION SIZE: 5

The output table

**==============**

 **ABCDEFG**

 **\_\_\_\_\_\_\_**

**DCBA LLLLLLL**

**==============**

**1010 | ...1...**

**0001 | 1......**

**0010 | ..1..1.**

**1011 | 11.....**

**1101 | 1....1.**

**1100 | .11...1**

**000- | ......1**

**-110 | .1.....**

**0101 | .1..1..**

**0100 | 1..11..**

**-111 | ...1...**

**0111 | ....111**

**111- | .11....**

**-001 | ...11..**

**00-1 | ....11.**

---------------------------------------------------

So, it can be deduced that:

A\_L = D’C’B’A + DC’BA + DCB’A + D’CB’A’

B\_L = DC’BA + DCB’A’ + CBA’ + D’CB’A+DCB

 ...

Alternatively, if Single Output Mode (SOM) is selected, the following output table is produced because each output is simplified independently

**==============**

 **ABCDEFG**

 **\_\_\_\_\_\_\_**

**DCBA LLLLLLL**

**==============**

**0100 | 1..1...**

**1011 | 1......**

**0001 | 1......**

**1101 | 1....1.**

**0101 | .1.....**

**1-11 | .1.....**

**11-0 | .11....**

**-110 | .1.....**

**0010 | ..1....**

**111- | ..1....**

**1010 | ...1...**

**-001 | ...11..**

**-111 | ...1...**

**010- | ....1..**

**0--1 | ....1..**

**001- | .....1.**

**00-1 | .....1.**

**0-11 | .....1.**

**1100 | ......1**

**0111 | ......1**

**000- | ......1**

---------------------------------------------------

So, it can be deduced that:

A\_L = D’CB’A’ + DC’BA + D’C’B’A + DCB’A

····

G\_L = DCB’A’ + D’CBA + D’C’B’

If you want to produce a more complex design, as the following one:



Then, produce the “HEX\_7SEG\_ALL.tbl” file

table HEX\_7SEG\_ALL

input BI\_L " Blanking input active low

input LT\_L " Lamp Test active low

input RBI\_L " Ripple blanking output active low

input D C B A " Code inputs active high

output a\_L b\_L c\_L d\_L e\_L f\_L g\_L " Segment outputs active low for

 " common anode display

output RBO\_L " Ripple blankig output active low

" Truth table for the basic hexadecimal to 7 segment decoder

" INPUTS OUTPUTS

" ======================= ===================================

" BI\_L LT\_L RBI\_L D C B A a\_L b\_L c\_L d\_L e\_L f\_L g\_L RBO\_L Symbol

" ===== ================= ===================================

 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 " 0

 1 1 - 0 0 0 1 1 0 0 1 1 1 1 1 " 1

 1 1 - 0 0 1 0 0 0 1 0 0 1 0 1 " 2 -- a --

 1 1 - 0 0 1 1 0 0 0 0 1 1 0 1 " 3 | |

 1 1 - 0 1 0 0 1 0 0 1 1 0 0 1 " 4 f b

 1 1 - 0 1 0 1 0 1 0 0 1 0 0 1 " 5 | |

 1 1 - 0 1 1 0 0 1 0 0 0 0 0 1 " 6 -- g --

 1 1 - 0 1 1 1 0 0 0 1 1 1 1 1 " 7 | |

 1 1 - 1 0 0 0 0 0 0 0 0 0 0 1 " 8 e c

 1 1 - 1 0 0 1 0 0 0 1 1 0 0 1 " 9 | |

 1 1 - 1 0 1 0 0 0 0 1 0 0 0 1 " A -- d --

 1 1 - 1 0 1 1 1 1 0 0 0 0 0 1 " b

 1 1 - 1 1 0 0 0 1 1 0 0 0 1 1 " C

 1 1 - 1 1 0 1 1 0 0 0 0 1 0 1 " d

 1 1 - 1 1 1 0 0 1 1 0 0 0 0 1 " E

 1 1 - 1 1 1 1 0 1 1 1 0 0 0 1 " F

" When Blankng Input is activated, then all segment OFF

 0 - - - - - - 1 1 1 1 1 1 1 0

" When Lamp Test is activated, then all segment ON

 1 0 - - - - - 0 0 0 0 0 0 0 1

" When Ripple Blanking Input is activated and DCBA = '0000', then all segment OFF

 1 1 0 0 0 0 0 1 1 1 1 1 1 1 0

End

And obtain for (MOM) the following table:

**==================**

 **R R**

**BLB B**

**ITI ABCDEFGO**

**\_\_\_ \_\_\_\_\_\_\_\_**

**LLLDCBA LLLLLLLL**

**==================**

**-1-1010 | ...1....**

**111000- | ......11**

**-1-1011 | 11......**

**-1-1101 | 1....1..**

**-1-0001 | 1.....1.**

**-1-1100 | .11...1.**

**-1--111 | ...1....**

**11-0010 | ..1..1.1**

**-1-111- | .11.....**

**11-0101 | .1..1..1**

**-1--001 | ...11...**

**11--110 | .1.....1**

**-100000 | 1111111.**

**11-0100 | 1..11..1**

**11-0111 | ....1111**

**11-00-1 | ....11.1**

**10----- | .......1**

**1--1--- | .......1**

**0------ | 1111111.**

Where the three-levels of gates circuit using NOT-AND-OR can be implemented as:

A\_L = LT\_L·D·C’·B·A + LT\_L·D·C·B’·A + LT\_L·D’·C’·B’·A +

+ LT\_L· RBI\_L’·D’·C’·B’·A’ + BI\_L·LT\_L·D’·C·B’·A’ + BI\_L’

·······