Appendix - The detailed design worksheet

inputs and outputs

4 bit input
7 bit output

encoding scheme

input

decimal digit	binary bits
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

output

LED state	binary bit
on	1
off	0

~~truth table~~

~~input~~ ~~output~~
~~$x_1 x_2 x_3 x_4$~~ ~~a b c d e f g~~

truth table

input				output						
X_3	X_2	X_1	X_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	-	-	-	-	-	-	-
1	0	1	1	-	-	-	-	-	-	-
1	1	0	0	-	-	-	-	-	-	-
1	1	0	1	-	-	-	-	-	-	-
1	1	1	0	-	-	-	-	-	-	-
1	1	1	1	-	-	-	-	-	-	-

Minimization

$a = M(1,4)$

x_0

1	0	1	1
0	1	1	1
–	–	–	–
1	1	–	–

x_1; x_2; x_3

$a = (x_2' + x_1 + x_0)(x_3 + x_2 + x_1 + x_0')$

$b = M(5,6)$

x_0

1	1	1	1
1	0	1	0
–	–	–	–
1	1	–	–

x_1; x_2; x_3

$b = (x_2' + x_1 + x_0')(x_2' + x_1' + x_0)$

$c = M(2)$

x_0

1	1	1	0
1	1	1	1
–	–	–	–
1	1	–	–

x_1; x_2; x_3

$c = x_3 + x_2 + x_1' + x_0$

$d = M(1,4,7)$

x_0

1	0	1	1
0	1	0	1
–	–	–	–
1	1	–	–

x_1; x_2; x_3

$d = (x_2' + x_1 + x_0)(x_2' + x_1' + x_0')(x_3 + x_2 + x_1 + x_0')$

Minimization of output g

Using K-maps

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	–	–	–	–
10	1	1	–	–

Prime implicates

$(x_3+x_2+x_1)(x_2'+x_1'+x_0')$

essential prime implicates

$(x_3+x_2+x_1)(x_2'+x_1'+x_0')$

$$g=(x_3+x_2+x_1)(x_2'+x_1'+x_0')$$

Minimization of output f

using K-maps

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	–	–	–	–
10	1	1	–	–

Prime implicates

$(x_1'+x_0')$

(x_2+x_1')

$(x_3+x_2+x_0')$

essential prime implicates

$(x_1'+x_0')$, (x_2+x_1'),

$(x_3+x_2+x_0')$

$$f=(x_1'+x_0')(x_2+x_1')(x_3+x_2+x_0')$$

Minimization of output e

Using K-maps

prime implicates

$(x_0')(x_2'+x_1)$

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	–	–	–	–
10	1	0	–	–

essential prime implicates

$(x_0')(x_2'+x_1)$

$$e=(x_0')(x_2'+x_1)$$

x_1'

transformation procedure by demorgan's law

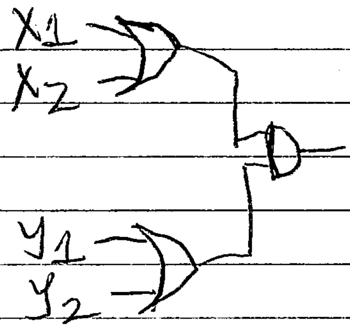
$X \cdot Y = (X' + Y')'$

OR-AND to NOR-NOR

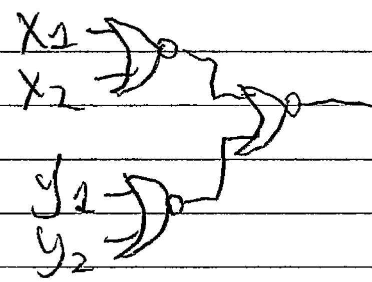
$X = X_1 + X_2$

$Y = Y_1 + Y_2$

$(X_1 + X_2)(Y_1 + Y_2) = ((X_1 + X_2)' + (Y_1 + Y_2)')'$



equal to

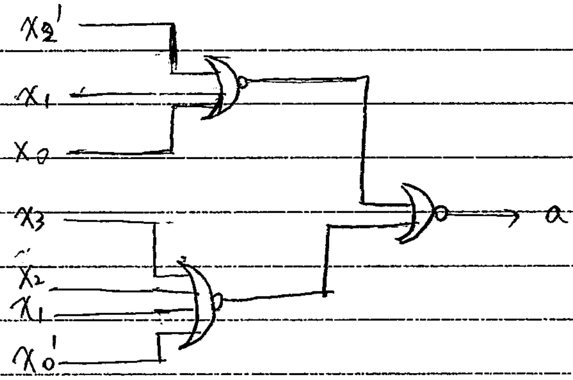


final minimal expressions

a. origional

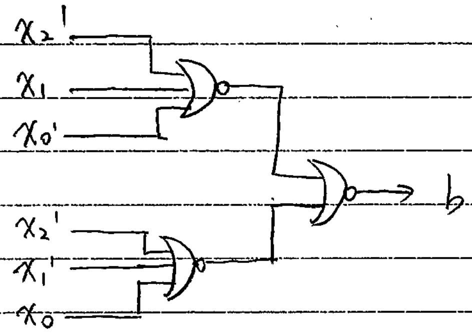
$a = (x_2' + x_1 + x_0)(x_3 + x_2 + x_1 + x_0')$

transformation: $a = ((x_2' + x_1 + x_0)' + (x_3 + x_2 + x_1 + x_0')')'$



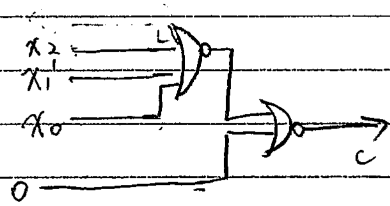
b origional $(x_2' + x_1 + x_0')(x_2' + x_1' + x_0)$

transformation $((x_2' + x_1 + x_0')' + (x_2' + x_1' + x_0)')'$



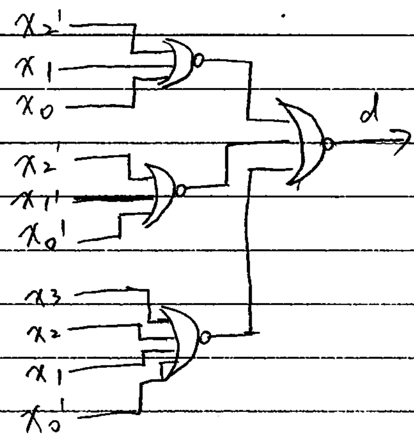
c origional $(\quad x_2 + x_1' + x_0)$

transformation $(\quad + x_2 + x_1' + x_0) \cdot 1 = ((\quad x_2 + x_1' + x_0)' + 0)'$



d origional $(x_2'+x_1+x_0)(x_2'+x_1'+x_0')(x_3+x_2+x_1+x_0')$

transformation $((x_2'+x_1+x_0)'+(x_2'+x_1'+x_0')'+(x_3+x_2+x_1+x_0')')'$

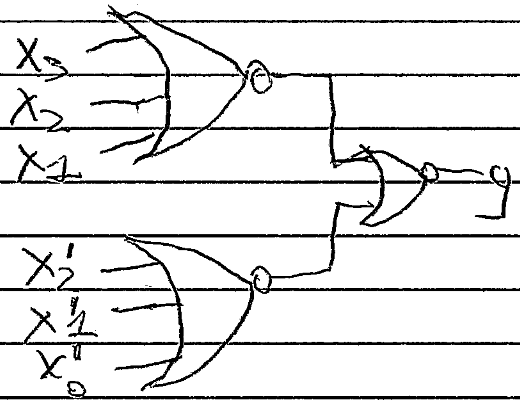


final minimal expressions

Output y

OR-AND $y = (x_3+x_2+x_1)(x_2'+x_1'+x_0')$

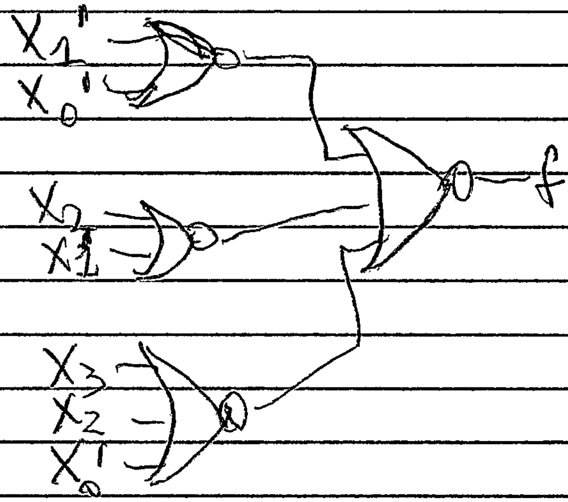
NOR-NOR $y = ((x_3+x_2+x_1)' + (x_2'+x_1'+x_0')')'$



Output f

OR-AND $f = (x_1'+x_0')(x_2+x_1')(x_3+x_2+x_0')$

NOR-NOR $f = ((x_1'+x_0')' + (x_2+x_1')' + (x_3+x_2+x_0')')'$



output e

OR-AND $e = (X_0')(X_2' + X_1)$

NOR-NOR $e = (X_0 + (X_2' + X_1)')'$