Entradas: 4 bits que son **a**, **b**, **c** y **d**
Salidas: 2 bits que son **B1** y **B2**
Significado de las clausulas:

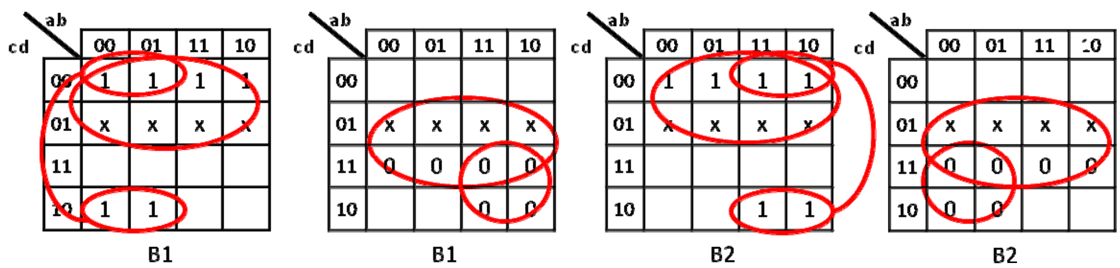
1) Si c=1 y d=0 → Si a=0 → B1=1, B2=0
   Si c=1 y d=0 → Si a=1 → B1=0, B2=1
   Al leer la segunda parte de la clausula se interpreta que la primera parte era con a=0, y que la bomba B" estaba apagada.
2) Si c=1 y d=1 → B1=0, B2=0
3) Si c=0 y d=0 → B1=1, B2=1

*(ya se puede observar que no dependerá de **b** la solución ni para B1, ni para B2)*

	a	b	c	d		B1	B2	
3)	0	0	0	0		1	1	
	0	0	0	1		X	X	absurdo
1)	0	0	1	0		1	0	
2)	0	0	1	1		0	0	
3)	0	1	0	0		1	1	
	0	1	0	1		X	X	absurdo
1)	0	1	1	0		1	0	
2)	0	1	1	1		0	0	
3)	1	0	0	0		1	1	
	1	0	0	1		X	X	absurdo
1)	1	0	1	0		0	1	
2)	1	0	1	1		0	0	
3)	1	1	0	0		1	1	
	1	1	0	1		X	X	absurdo
1)	1	1	1	0		0	1	
2)	1	1	1	1		0	0	

Por tanto las soluciones para B1 y B2 son

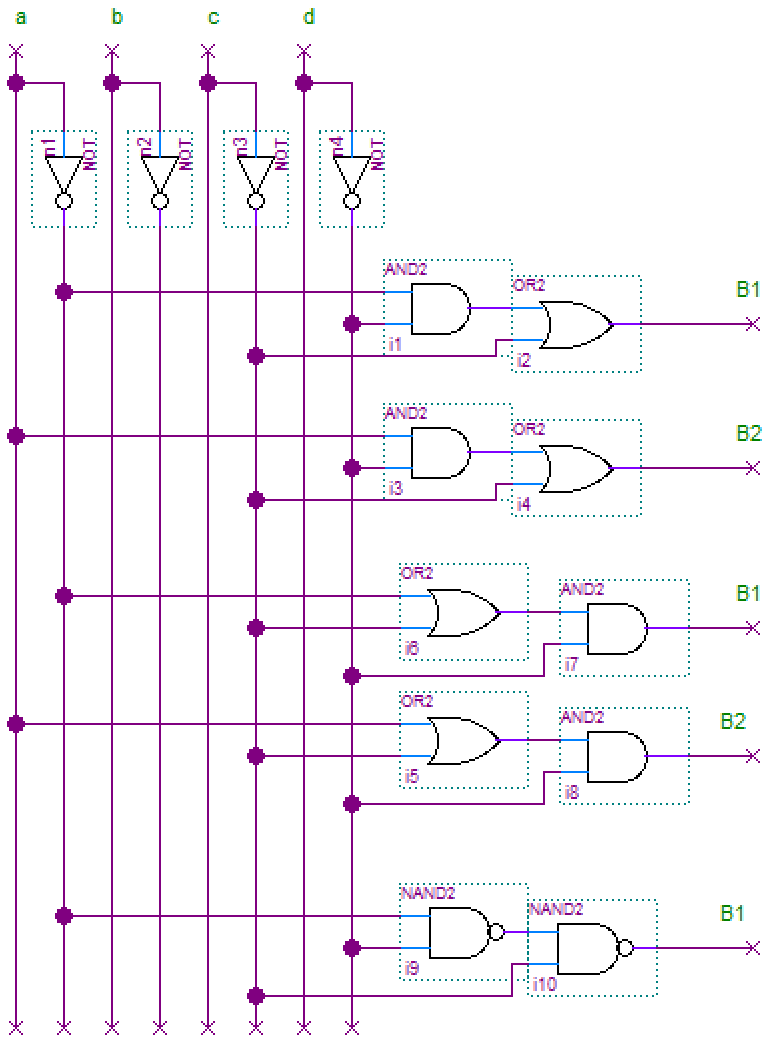
$B1 = \Sigma_m(0,2,4,6,8,12) + \Phi(1,5,9,13)$ o $B1 = \Pi_M(3,7,10,11,14,15) \cdot \Phi(1,5,9,13)$
$B2 = \Sigma_m(0,4,8,10,12,14) + \Phi(1,5,9,13)$ o $B2 = \Pi_M(2,3,6,7,11,15) \cdot \Phi(1,5,9,13)$



B1 = /c + (/a·/d) o B1 = /d · (/a+/c) B2 = /c + (a·/d) o B2 = /d · (a+/c)

Para resolver B1 en puertas NAND de 2 entradas lo mejor es partir de la solución en SOP.

B1 = /c + (/a·/d) = //(/c + (/a·/d)) = /(//c · /(/a·/d)) = /(c·/(/(a·a)·/(d·d)))


a
b
c
d
n1
NOT
n2
NOT
n3
NOT
n4
NOT
AND2
i1
OR2
i2
B1
AND2
i3
OR2
i4
B2
OR2
i6
AND2
i7
B1
OR2
i5
AND2
i8
B2
NAND2
i9
NAND2
i10
B1