Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería Eléctrica

IEE2713 — Sistemas Digitales

Pauta T3

1. Expandimos la expresión para luego simplificar directamente:

$$\begin{aligned}
f &= x_1x_3 + x_1\bar{x_2} + \bar{x_1}x_2x_3 + \bar{x_1}\bar{x_2}\bar{x_3} \\
&= x_1x_3(x_2 + \bar{x_2}) + x_1\bar{x_2}(x_3 + \bar{x_3}) + \bar{x_1}x_2x_3 + \bar{x_1}\bar{x_2}\bar{x_3} \\
&= x_1x_2x_3 + x_1\bar{x_2}x_3 + \cancel{x_1\bar{x_2}x_3} + x_1\bar{x_2}\bar{x_3} + \bar{x_1}x_2x_3 + \bar{x_1}\bar{x_2}\bar{x_3} \\
&= (x_2x_3 + \bar{x_1}x_2x_3) + x_1\bar{x_2}x_3 + x_1\bar{x_2}\bar{x_3} + \bar{x_1}\bar{x_2}\bar{x_3} \\
&= x_2x_3 + (x_1\bar{x_2}x_3 + x_1\bar{x_2}\bar{x_3}) + (x_1\bar{x_2}\bar{x_3} + \bar{x_1}\bar{x_2}\bar{x_3}) \\
&= x_2x_3 + x_1\bar{x_2} + \bar{x_2}\bar{x_3}
\end{aligned}$$

   También es correcta la solución:

$$f = x_2x_3 + x_1x_3 + \bar{x_2}\bar{x_3}$$

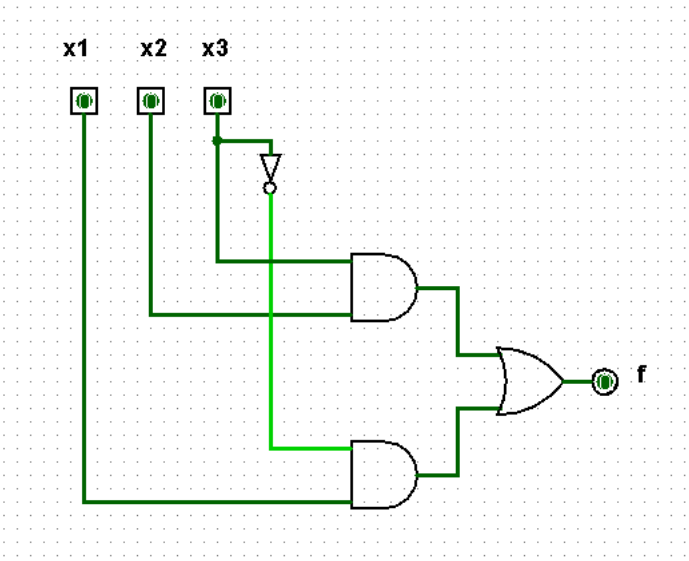
2. Expandimos cada término para luego simplificar usando la propiedad (12b) $(x + y)(x + z) = x + yz$.

$$\begin{aligned}
f &= (x_1 + x_3 + x_4)(x_1 + \bar{x_2} + x_3)(x_1 + \bar{x_2} + \bar{x_3} + x_4) \\
&= (x_1 + x_2 + x_3 + x_4)(x_1 + \bar{x_2} + x_3 + x_4)\cancel{(x_1 + \bar{x_2} + x_3 + x_4)} \\
&(x_1 + \bar{x_2} + x_3 + \bar{x_4})(x_1 + \bar{x_2} + \bar{x_3} + x_4) \\
&= (x_1 + x_4 + (\bar{x_2} + x_3)(\bar{x_2} + \bar{x_3}))(x_1 + x_3 + (\bar{x_2} + x_4)(x_2 + x_4)) \\
&(x_1 + \bar{x_2} + (x_3 + x_4)(x_3 + \bar{x_4})) \\
&= (x_1 + \bar{x_2} + x_4)(x_1 + x_3 + x_4)(x_1 + \bar{x_2} + x_3)
\end{aligned}$$

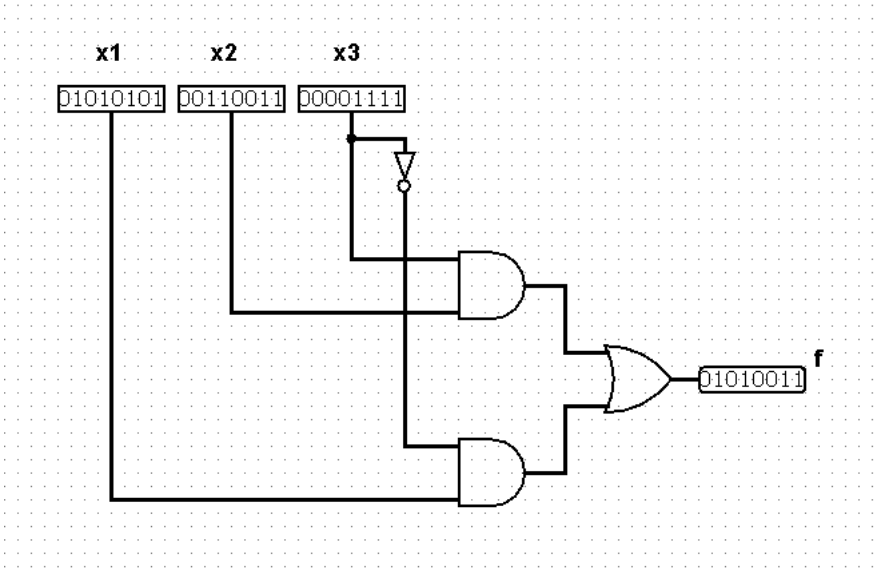
3. Antes de diseñar el circuito simplificamos la ecuación:

$$\begin{aligned}
f &= \bar{x_1}x_2x_3 + x_1\bar{x_2}\bar{x_3} + x_1x_2\bar{x_3} + x_1x_2x_3 \\
&= x_2x_3 + x_1\bar{x_3}
\end{aligned}$$

   Así obtenemos el siguiente circuito, que si nos damos cuenta es el mismo que el de un multiplexor:



4. Simulamos todas las combinaciones:



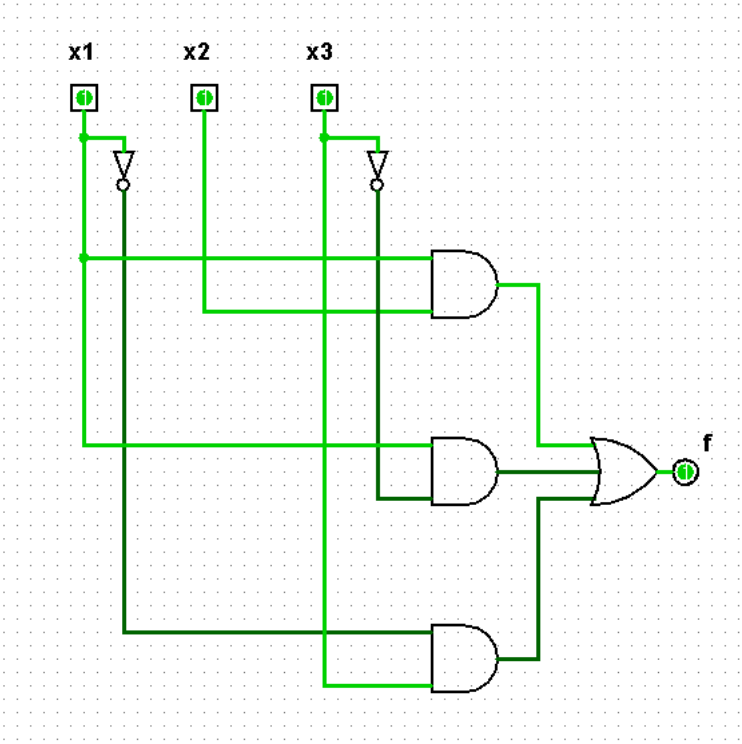
5. Nuevamente simplificamos la función usando el teorema del consenso:

$$\begin{aligned} f &= (x_1 + x_2 + x_3)(x_1 + \bar{x_2} + x_3)(\bar{x_1} + x_2 + \bar{x_3}) \\ &= (x_1 + x_3)(\bar{x_1} + x_2 + \bar{x_3}) \\ &= x_1x_2 + x_1\bar{x_3} + \bar{x_1}x_3 + x_2x_3 \\ &= x_1x_2 + x_1\bar{x_3} + \bar{x_1}x_3 \end{aligned}$$

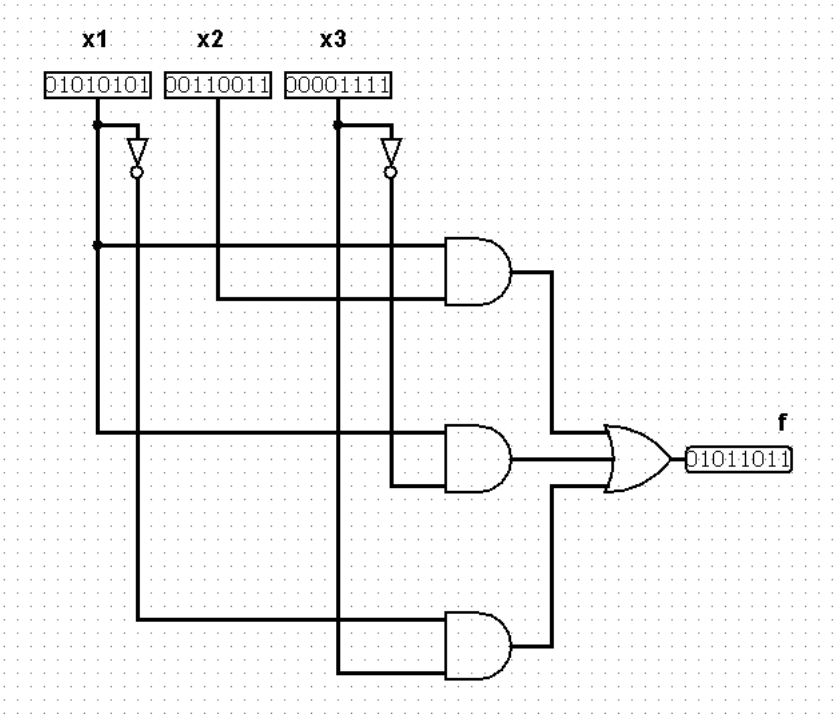
También es válida la solución:

$$f = x_1\bar{x_3} + \bar{x_1}x_3 + x_2x_3$$

El circuito implementado:



6. Probando todas las combinaciones comprobamos que los únicos valores en cero están en la posición M_0, M_2, M_5



7. a) La tabla de verdad es la siguiente:

x_1	x_0	y_1	y_0	f
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

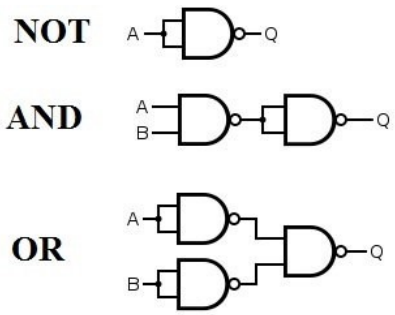
b) La suma de productos obtenida directamente de los *miniterms* es la siguiente:

$$
\begin{aligned}
f &= \sum m(0,4,5,8,9,10,12,13,14,15) \\
&= \bar{x_1}\bar{x_0}\bar{y_1}\bar{y_0} + \bar{x_1}x_0\bar{y_1}\bar{y_0} + \bar{x_1}x_0\bar{y_1}y_0 + x_1\bar{x_0}\bar{y_1}\bar{y_0} + x_1\bar{x_0}\bar{y_1}y_0 + x_1\bar{x_0}y_1\bar{y_0} + \\
& x_1x_0\bar{y_1}\bar{y_0} + x_1x_0\bar{y_1}y_0 + x_1x_0y_1\bar{y_0} + x_1x_0y_1y_0
\end{aligned}
$$

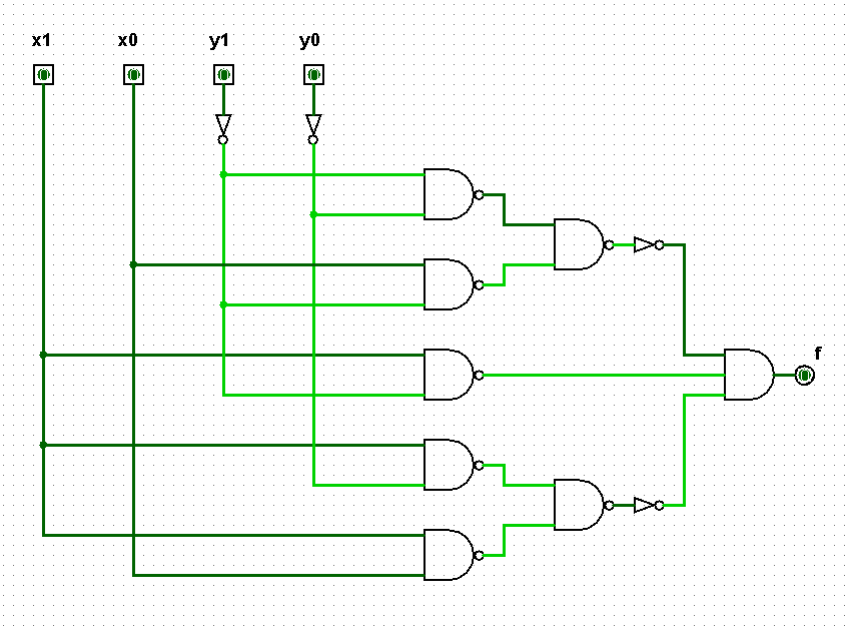
c) Simplificando la ecuación anterior:

$$f = \bar{y_1}\bar{y_0} + x_0\bar{y_1} + x_1\bar{y_1} + x_1\bar{y_0} + x_1x_0$$

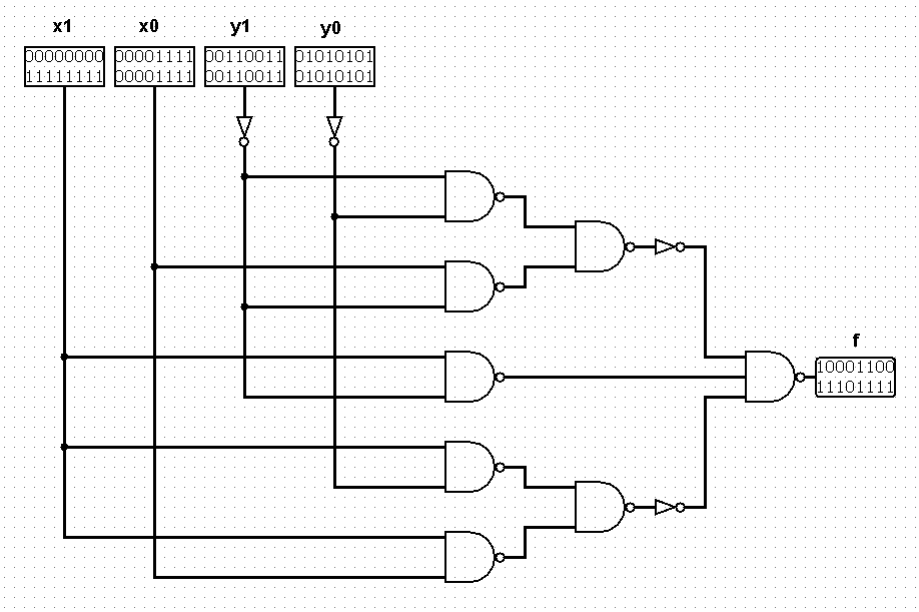
d) Para hacer el circuito necesitamos hacer un OR por medio de NANDs. Esto se puede hacer de la siguiente manera:



De esta forma obtenemos el siguiente circuito:



e) Finalmente simulamos:



8. pregunta_8.sv

```
1          //------------------------------------------------
2          // Nombre del modulo    : pregunta_8
3          // Nombre del archivo   : pregunta_8.sv
4          // Funcion              : f = sum{m(1,2,3,4,5,6)}
5          //------------------------------------------------
6          module pregunta_8(x1, x2, x3, f);
7
8            input  x1, x2, x3;
9            output f;
10
11           assign f = (~x1 & ~x2 & x3) | (~x1 & x2 & ~x3) |
12                      (~x1 & x2 & x3)  | (x1 & ~x2 & ~x3) |
13                      (x1 & ~x2 & x3)  | (x1 & x2 & ~x3);
14
15         endmodule
```

9. pregunta_8_tb.sv

```
1          //------------------------------------------------
2          // Nombre del modulo    : pregunta_8_tb
3          // Nombre del archivo   : pregunta_8_tb.sv
4          // Funcion              : testbench para pregunta_8
5          //------------------------------------------------
6          `timescale 1ns/1ps
```

```
7
8           module pregunta_8_tb();
9
10            reg  x1, x2, x3;
11            wire f;
12            integer i;
13
14            pregunta_8 dut(x1, x2, x3, f);
15
16            initial begin
17
18                $dumpfile("pregunta_8_tb.vcd");
19                $dumpvars(0, pregunta_8_tb);
20
21                for(i=0; i<8; i=i+1) begin
22                    {x3, x2, x1} = i;
23                    #2;
24                end
25
26                $finish;
27
28            end
29
30          endmodule
```

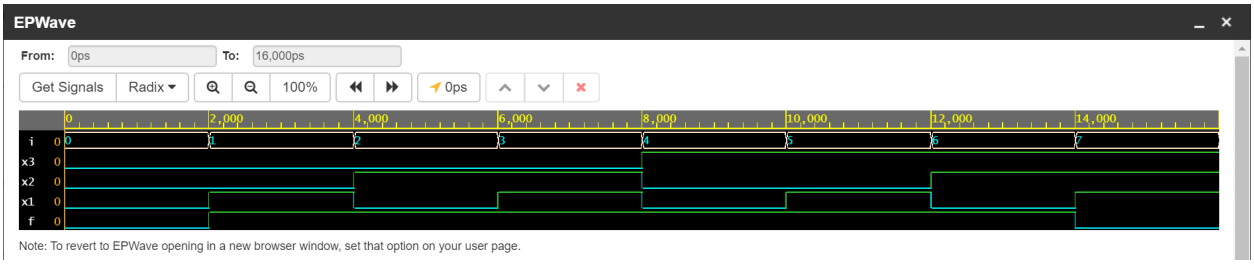


Figura 1: Diagrama de tiempo para pregunta 8.

10. mux_4x1_mod.sv

```
1           //-------------------------------------------------
2           // Nombre del modulo     : mux_4x1_mod
3           // Nombre del archivo    : mux_4x1_mod.sv
4           // Funcion               : Multiplexor 4:1
5           //-------------------------------------------------
6           `include "mux_2x1_mod.v";
7
8           module mux_4x1_mod(x0, x1, x2, x3, s0, s1, dout);
```

```
9
10            input x0, x1, x2, x3;
11            input s0, s1;
12            output dout;
13
14            wire a, b;
15
16            mux_2x1_mod mux1(x0, x1, s0, a);
17            mux_2x1_mod mux2(x2, x3, s0, b);
18
19            mux_2x1_mod mux3(a, b, s1, dout);
20
21          endmodule
```

La implementación del multiplexor anterior se basa en el siguiente diagrama:

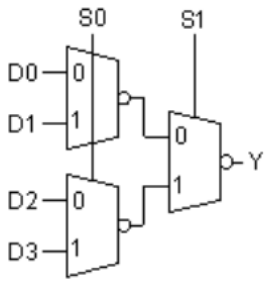


Figura 2: Multiplexor 4:1 modularizado.

11. mux_4x1_mod_tb.sv

```
1           //-------------------------------------------------
2           // Nombre del modulo     : mux_4x1_mod_tb
3           // Nombre del archivo    : mux_4x1_mod_tb.sv
4           // Funcion               : testbench para mux_4x1_mod
5           //-------------------------------------------------
6           `timescale 1ns/1ps
7
8           module mux_4x1_mod_tb();
9
10            reg  x0, x1, x2, x3;
11            reg  s0, s1;
12            wire dout;
13            integer i;
14
15            mux_4x1_mod dut(x0, x1, x2, x3, s0, s1, dout);
```

```
16
17              initial begin
18
19                  $dumpfile("mux_4x1_mod_tb.vcd");
20                  $dumpvars(0, mux_4x1_mod_tb);
21
22                  for(i=0; i<64; i=i+1) begin
23                      {s1, s0, x3, x2, x1, x0} = i;
24                      #2;
25                  end
26
27                  $finish;
28
29              end
30
31          endmodule
```

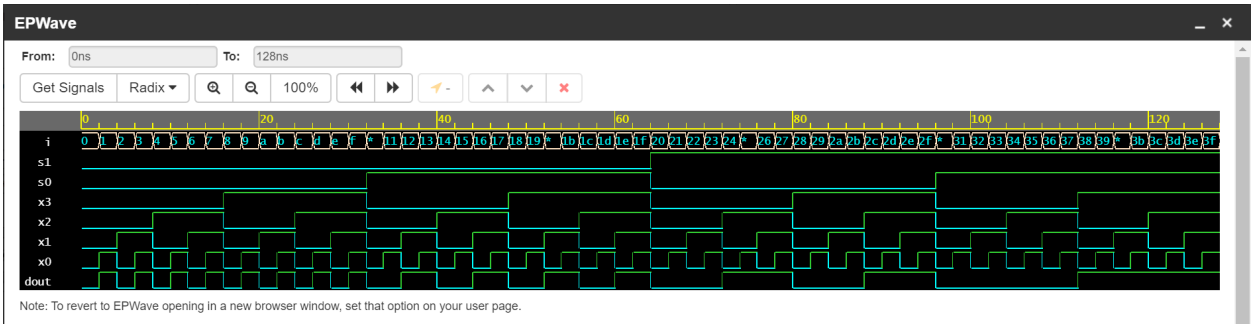


Figura 3: Diagrama de tiempo para pregunta 11.