Diseño con PROMs

Los componentes con los que se realiza el diseño condiciona el tipo de diseño a realizar. En este caso queremos diseñar con PLDs, por lo tanto, si tenemos en cuenta el tipo de circuito que vamos a usar, nos damos cuenta que debemos conseguir representar las expresiones del circuito a implementar en función de los términos mínimos de las variables de entrada y que, en este caso, no tiene sentido su minimización, sino todo los contrario. Es decir, para diseñar con PLDs lo que tenemos que hacer es expandir las funciones en vez de minimizarlas. Si se realiza el diseño a partir de la tabla de verdad las expresiones obtenidas se usan directamente, sin minimizar.

El ejercicio de autoevaluación 2.3 consiste en la síntesis de un sumador de acarreo adelantado para palabras de 4 bits y usando una PROM. Dado que el diseño para palabras de 4 bits es muy laborioso y que no se pierde el valor pedagógico perseguido con su resolución, lo vamos a resolver para palabras de 2 bits.

Los pasos a seguir para realiza el diseño son:

1. En este caso concreto podemos partir de las expresiones 5.9 y 5.10 de la pag.277 del texto que son las expresiones recursivas que tenemos que ir desarrollando para los distintos bits de las palabras binarias.

   $P_i = A_1 \oplus B_i \ , \ G_i = A_1 B_i$

   $S_i = P_1 \oplus C_i \ , \ C_{i+1} = G_i + P_1 C_i$

2. Particularizamos estas expresiones para los dos bits de las palabras que queremos sumar y que son los datos del problema junto con el acarreo de entrada, C_0. Es decir, debemos poner las expresiones en función de los bits de las palabras $A(A_1\ A_0)$ y $B(B_1\ B_0)$ y del acarreo de entrada, C_0, siendo A_1 y B_1 los bits más significativos. Así,

   $P_0 = A_0 \oplus B_0, \quad G_0 = A_0 B_0 \qquad P_1 = A_1 \oplus B_1, \quad G_1 = A_1 B_1$

   $S_0 = P_0 \oplus C_0 \ , \quad C_1 = G_0 + P_0 C_0 \qquad S_1 = P_1 \oplus C_1 \ , \quad C_2 = G_1 + P_1 C_1$

3. Sustituimos los valores de las P_i y G_i por sus expresiones en función de los bits de las palabras:

   $S_0 = A_0 \oplus B_0 \oplus C_0 \ , \quad C_1 = A_0 B_0 + (A_0 \oplus B_0) C_0$

   $S_1 = A_1 \oplus B_1 \oplus C_1 = A_1 \oplus B_1 \oplus [A_0 B_0 + (A_0 \oplus B_0) C_0]$,
   $C_2 = A_1 B_1 + (A_1 \oplus B_1)[A_0 B_0 + (A_0 \oplus B_0) C_0]$

4. Operando resulta:

   $S_0 = (\overline{A}_0 B_0 + A_0 \overline{B}_0) \oplus C_0 = \overline{(\overline{A}_0 B_0 + A_0 \overline{B}_0)} C_0 + (\overline{A}_0 B_0 + A_0 \overline{B}_0) \overline{C}_0$
   $= (\overline{A}_0 \overline{B}_0 + A_0 B_0) C_0 + (\overline{A}_0 B_0 + A_0 \overline{B}_0) \overline{C}_0 = \overline{A}_0 \overline{B}_0 C_0 + A_0 B_0 C_0 + \overline{A}_0 B_0 \overline{C}_0 + A_0 \overline{B}_0 \overline{C}_0$
   $C_1 = A_0 B_0 + (A_0 \oplus B_0) C_0 = A_0 B_0 + (\overline{A}_0 B_0 + A_0 \overline{B}_0) C_0 = A_0 B_0 + \overline{A}_0 B_0 C_0 + A_0 \overline{B}_0 C_0$

   Análogamente,

   $S_1 = (\overline{A}_1 B_1 + A_1 \overline{B}_1) \oplus C_1 = \overline{A}_1 \overline{B}_1 C_1 + A_1 B_1 C_1 + \overline{A}_1 B_1 \overline{C}_1 + A_1 \overline{B}_1 \overline{C}_1$

   pero, $\quad C_1 = A_0 B_0 + \overline{A}_0 B_0 C_0 + A_0 \overline{B}_0 C_0 = A_0 B_0 + B_0 C_0 + A_0 C_0$

   y $\quad \overline{C}_1 = \overline{A_0 B_0 + \overline{A}_0 B_0 C_0 + A_0 \overline{B}_0 C_0} = \overline{A}_0 \overline{B}_0 + \overline{B}_0 \overline{C}_0 + \overline{A}_0 \overline{C}_0$

por tanto, operando obtenemos:

$$S_1 = \left(\overline{A}_1\,\overline{B}_1 + A_1\,B_1\right)C_1 + \left(\overline{A}_1\,B_1 + A_1\,\overline{B}\right)_1\,\overline{C}_1 =$$
$$= \left(\overline{A}_1\,\overline{B}_1 + A_1\,B_1\right)\left(A_0\,B_0 + B_0\,C_0 + A_0\,C_0\right) + \left(\overline{A}_1\,B_1 + A_1\,\overline{B}_1\right)\left(\overline{A}_0\,\overline{B}_0 + \overline{B}_0\,\overline{C}_0 + \overline{A}_0\,\overline{C}_0\right) =$$
$$= \overline{A}_1\,\overline{B}_1 A_0\,B_0 + A_1\,B_1 A_0\,B_0 + \overline{A}_1\,\overline{B}_1\,B_0\,C_0 + A_1\,B_1 B_0\,C_0 + \overline{A}_1\,\overline{B}_1 A_0\,C_0 + A_1\,B_1 A_0\,C_0 +$$
$$+ \overline{A}_1\,B_1\,\overline{A}_0\,\overline{B}_0 + A_1\,\overline{B}_1\,\overline{A}_0\,\overline{B}_0 + \overline{A}_1\,B_1 \overline{B}_0\,\overline{C}_0 + A_1\,\overline{B}_1\,\overline{B}_0\,\overline{C}_0 + \overline{A}_1\,B_1\,\overline{A}_0\,\overline{C}_0 + A_1\,\overline{B}_1 \overline{A}_0\,\overline{C}_0$$

De igual forma calcularíamos C_2

5. Para implementar estas expresiones mediante una PROM tenemos que ponerlas en función de los términos mínimos de las señales de entrada o datos del problema, A_1, B_1, A_0, B_0 y C_0.

Si observamos estas expresiones vemos que S_0 y C_1 son funciones sólo de A_0, B_0 y C_0, y que las expresiones de S_1 y C_2 son funciones de A_0, B_0, C_0, A_1, y B_1, pero los productos que forman parte de ellas no son los términos mínimos de las 5 variables de entrada. Por tanto, el siguiente paso es expandirlas, o sea, expresarlas en función de los términos mínimos de las 5 variables de entrada. La forma de conseguir esto es ver la/s variable/s que falta/n en cada uno de los sumandos de cada una de las expresiones y multiplicarlos por la suma de esta/s variable/s y su/s complementada/s. Así, como en la expresión de S_0 faltan las variables A_1 y B_1 en todos los sumandos, deberemos multiplicarla por $\left(A_1 + \overline{A}_1\right)\left(B_1 + \overline{B}_1\right)$. En realidad, lo que estamos haciendo es dejarla como está, puesto que la estamos multiplicando por la unidad, pero la estamos cambiando de representación. Justo, estamos haciendo lo contrario de lo que hacemos al minimizar, o sea, estamos aplicando Karnaugh al revés, ya que en vez de agrupar los sumandos, lo que hacemos es expandirlos. Así,

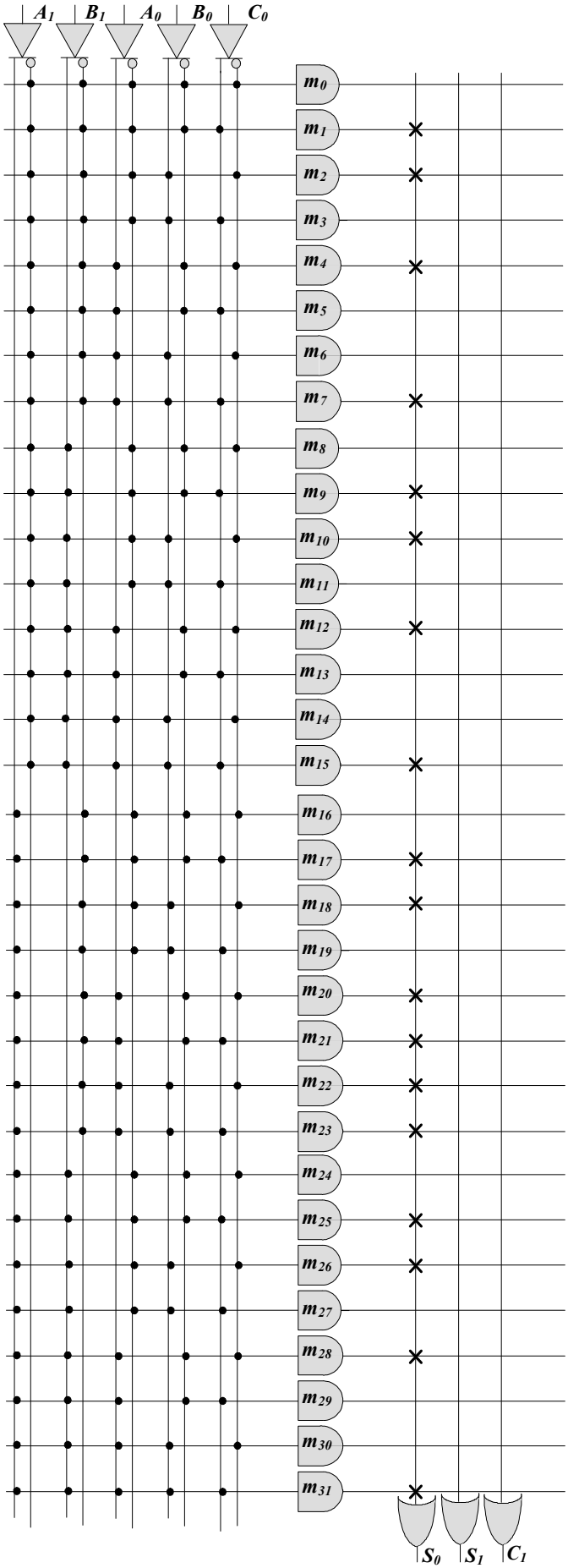
$$S_0 = \overline{A}_0\,\overline{B}_0\,C_0 + A_0\,B_0 C_0 + \overline{A}_0\,B_0\,\overline{C}_0 + A_0\,\overline{B}_0\,\overline{C}_0 =$$
$$= \left(\overline{A}_0\,\overline{B}_0\,C_0 + A_0\,B_0 C_0 + \overline{A}_0\,B_0\,\overline{C}_0 + A_0\,\overline{B}_0\,\overline{C}_0\right)\left(A_1 + \overline{A}_1\right)\left(B_1 + \overline{B}_1\right)$$

Si operamos, la expresamos en función de los términos mínimos de forma simplificada, o sea en función de m_i, siendo i el equivalente en decimal del término mínimo (que puede tomar 2^5-1 valores, desde 0 hasta 31) y tomamos para el orden de los bits de las entradas el mismo que después consideraremos en las entradas de la PROM, A_1(MSB)$B_1A_0B_0C_0$(LSB), obtenemos:

$$S_0 = m_1 + m_2 + m_4 + m_7 + m_9 + m_{10} + m_{12} + m_{15} + m_{17} + m_{18} + m_{20} + m_{23} + m_{25} + m_{26} + m_{28} + m_{31}$$

De igual forma operaríamos con las expresiones S_1 y C_2.

Una vez obtenidas las 3 expresiones, S_0, S_1 y C_2 las implementamos en la PROM. Así el circuito resultante para S0 es:


A1
B1
A0
B0
C0
m0
m1
m2
m3
m4
m5
m6
m7
m8
m9
m10
m11
m12
m13
m14
m15
m16
m17
m18
m19
m20
m21
m22
m23
m24
m25
m26
m27
m28
m29
m30
m31
S0
S1
C1

****