11. Generador/comprobador de paridad

En las transferencias de datos digitales (dentro de un sistema digital o en la transmisión de códigos de un sistema a otro), se pueden producir errores. Estos errores se manifiestan a través de cambios indeseados en alguno de los bits que conforman el paquete de información. La probabilidad de que ocurra un error en una transmisión de este tipo es muy pequeña y de que ocurran dos todavía menor. Para la detección de estos errores se utiliza un bit de paridad.

11.1. Bit de paridad

Es un bit que se añade a la izquierda del grupo de bits que forman el paquete de información a transmitir. El objetivo es conseguir que en todos los paquetes a transmitir, la cantidad de 1s sea par o impar según se establezca con anterioridad.

Ejemplo:

Acuerdo / **Paquete de información**	**Paridad par**	**Paridad impar**
ASCII A = 1000001	01000001	11000001
ASCII T= 1010100	11010100	01010100

Si un sistema trabaja con paridad par, todos los paquetes que recibe el sistema receptor deberan contener un número par de unos. Si no fuese así, en la transmisión habría ocurrido un error.

Puede concluirse que mediante este sistema, solo se detecta si hay un número impar de errores por paquete de información, es decir, si en la transmisión hubiera 2 errores, el receptor no detectaría dicho error.

11.2. Generación y comprobación de paridad

Así pues, es necesario diseñar un sistema que genere el bit de paridad a añadir al paquete de información y otro sistema que compruebe la paridad en el receptor. A este tipo de circuitos se les denomina Generador de paridad y Comprobador de paridad , respectivamante.

Generador de paridad

Supongamos que se desea transmitir un paquete d einformación compuesot por dos bits (A_1 A_0) y que el acuerdo preestablecido es la utilización de paridad par. En ese caso, la tabla de verdad y el circuito correspondiente son los mostrados en la Figura 125.

A_1	A_0	B_P
0	0	0
0	1	1
1	0	1
1	1	0

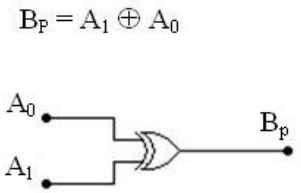


Figura 125

Siguiendo el mismo razonamiento, si el paquete de información a enviar debe contener tres bits (A_2 A_1 A_0), la tabla de verdad par ael diseño del circuito sería la desarrollada en la Figura 126.

A_2	A_1	A_0	B_P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

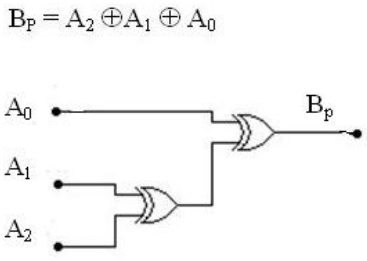


Figura 126

Transmisión

Una vez generado el bit de paridad, se añade al paquete a transmitir tal y como se muestra en la Figura 127.

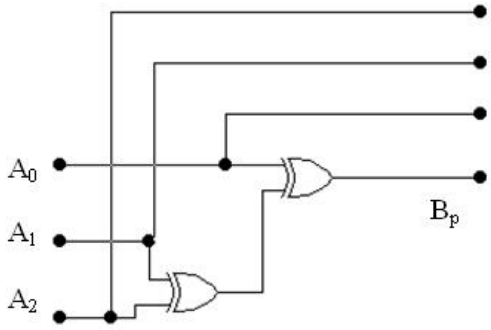


Figura 127

Comprobador de paridad

Siguiendo con el último ejemplo, transmisión de un paquete de información compuesto por tres bits con convenio de paridad par, la tabla de verdad correspondiente al circuito comprobador de paridad se presenta enla Figura 128 junto con la función lógica y el diagrama lógico obtenido apartir d ela misma.

A_2	A_1	A_0	B_P	E
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Donde E = 1 indica que ha habido algún error en la transmisión, es decir, ha detectado que el númeor de 1s recibido ha sido impar cuando debería haber sido par según el conveio preestablecido.

$E = A_2 \oplus A_1 \oplus A_0 \oplus B_P$

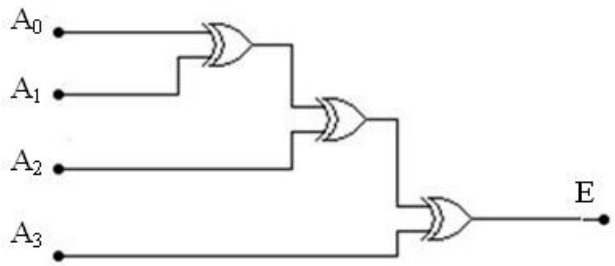


Figura 128

11.3. Circuito comercial

Un circuito comercial es el 74280 y puede utilizarse tanto como generador como comprobador de paridad haciendo un uso adecuado de sus entradas y salidas. Su tabla de funcionamiento y encapsulado se presentan en la Figura 129 y en el siguiente apartado se analizara su utidad a través de una aplicación.

FUNCTION TABLE

NUMBER OF INPUTS A THRU I THAT ARE HIGH	OUTPUTS	
	Σ EVEN	Σ ODD
0, 2, 4, 6, 8	H	L
1, 3, 5, 7, 9	L	H

H = high level, L = low level

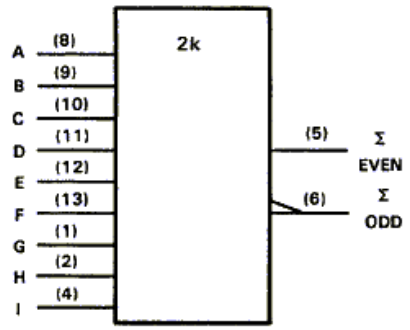


Figura 129

11.4. Aplicación

En la Figura 130 se muestra un sistema de transmisión de datos. En él se han utilizado un multiplexor, un demultiplexor, el generador/comprobador de paridad y un sistema de almacenamiento. La función de este último bloque consiste en guardar los datos hasta tener disponibles los 8 datos del paquete.

Para utilizarlo el 74280 como generador de paridad par utilizaríamos la salida Impar y para utilizarlo como comprobador de paridad par también la salida Impar. Esta salida determina si hay error en los datos de sus entradas pero esa señal de error solo será válida en el momento en que se hayan transmitido los 8 bits. Para ello, se utiliza la puerta AND que transmitirá el error cuando las entradas de selección sean todas 1.

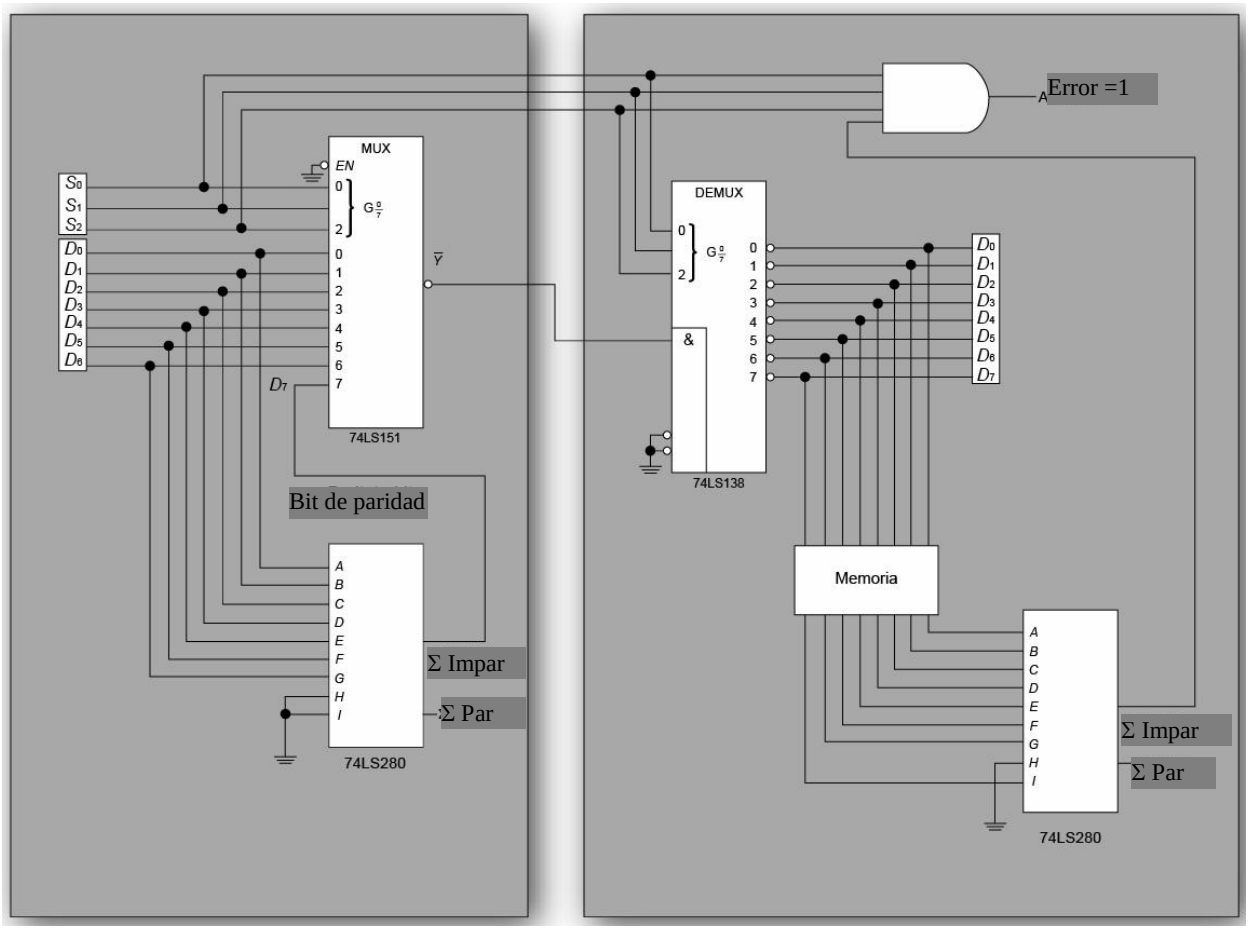

MUX
EN
S0
S1
S2
D0
D1
D2
D3
D4
D5
D6
D7
74LS151
Bit de paridad
74LS280
Σ Impar
Σ Par
DEMUX
&
74LS138
Error =1
Memoria
74LS280
Σ Impar
Σ Par

Figura 130

11.5. Método de paridad para la detección y corrección de errores: códigos de *Hamming*

Además de los bits de paridad, existen otros códigos específicos que también permiten realizar la detección de errores. Uno de estos es el código Hammning mediante el cual se puede detectar un error y corregirlo. El código Hamming extendido permite detectar dos errores.

En el código Hammnig se emplean varios bits de paridad (BP) en lugar de un único bit para todo el paquete de datos (o palabra) a enviar.

Cada uno de los bits de paridad se genera a partir de un grupo de bits (un subgrupo) de la palabra (palabra: D_0, D_1, D_2, ...D_{n-1})) de datos. Por ello, el primer paso será decidir el número de bits de paridad que habrá que añadir. Este numero debe cumplir la siguiente inecuación:

$$2k \geq n + k + 1 \Rightarrow B_{p1}, B_{p2},\ldots B_{pk} \text{ (Bits de paridad)}$$

Donde *k* es el número de bits de paridad a añadir a la palabra de *n* bits. Por lo tanto, la palabra nueva tendrá *k* + *n* bits, y cada *k* bit es un bit de paridad de un subgrupo de bits de la palabra a transmitir. Cada bit de paridad debe ocupar unaposición concreta en la nueva palabra a transmitir. Esa posición se define utilizando la expresión:

$$P_{Bpi} = 2^{i-1}\ (i=1,..k) \Rightarrow \text{posición del bip de paridad i-ésimo}$$

Posición	1	2	3	4	5	6	7
Posición	P_1	P_2	P_3	P_4	P_5	P_6	P_7
Bit en la posición	B_{P1}	B_{P2}	D_0	B_{P3}	D_1	D_2	D_3

El segundo paso será definir los grupos de bits (o subgrupos) a paritr de la nueva palabra de datos, es decir la compuesta por los bits de información y los bits de paridad. También ahbrá que asignar a cada subgrupo uno de los bits de paridad creados B_{pk} ($i=1,..k$). Para ello, habrá que generar una lista de números desde 0 hasta (2^k-1). A continuación, esos números se deben expresar en binario natural. Teniendo en cuanta ese formato, se analiza el valor de cada uno de los bits de todos los números de la lista, es decir, los los bis de la posición 2^0 , los de la posición 2^1 y los de la posición 2^2 . En cada caso, se anota el número que contiene un 1. Los números obtenidos así indican la posición de los bits que formarán cada subgrupo. Un ejmeplo de aplicación se muestra en la Figura 131. para un paquete de infrmación de 4 bits, es decir, *n* = 4.

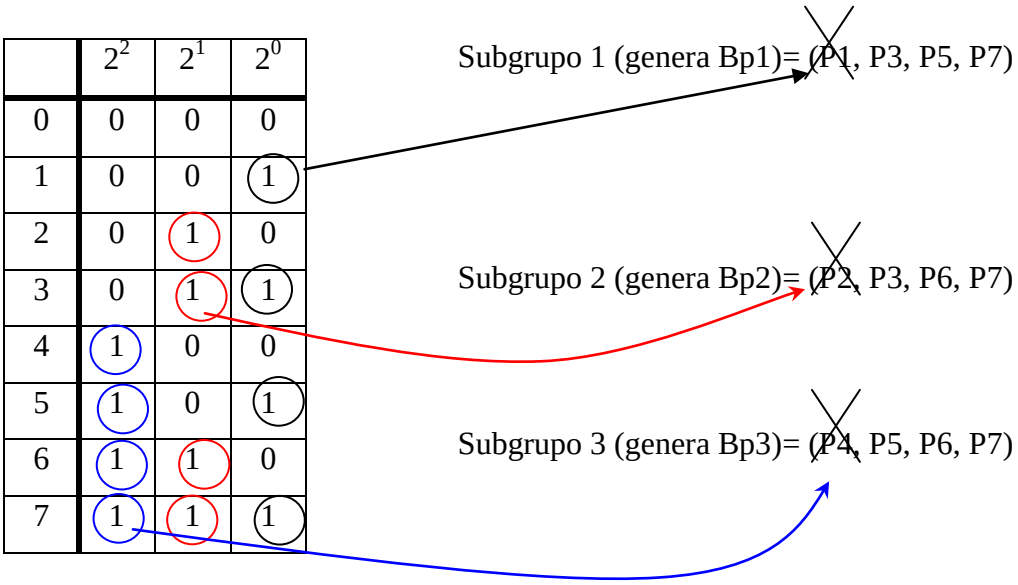


Figura 131

Se observa que cada subgrupos incluye la posición correspondiente aun bit de paridad, lo cual no tiene ningún sentido, ya que, estos grupos son precisamnete para generar los bits de paridad. Por ello, habrá que eliminar de cada subgrupo ese bit y ese grupo será el que genere el bit de paridad que ha habido que eliminar.

11.6. Aplicación: transmisión de palabras de 4 bits

Ene ste apartado se va a diseñar el sistema de generación/comprobación/corrección aplicando los códigos Hammning para un paquete de información de $n = 4$.

En primer lugar se obtienen los datos de partida:

- Número de bits de paridad = 3 que denominaremos $B_{p1}B_{p2}B_{p3}$ y ocupan las posiciones, P_0, P_1, P_4.
- Bits que corresponden en cada posicion:

$$B_{p1}B_{p2}D_0B_{p3}D_1D_2D_3 = P_1P_2P_3\ P_4P_5P_6P_7$$

- Agrupación de los bits (subgrupos que generan los bits de paridad):

$$\mathbf{B_{p1}} \propto (P_3P_5P_7);\ \mathbf{B_{p2}} \propto (P_3P_6P_7);\ \mathbf{B_{p3}} \propto (P_5P_6P_7)$$

A continuación se deben diseñenar los circuitos para generr los bits de paridad, detector de error y el corrector de error.

Circuito generador de bits de paridad

El diagrama de bloques para el circuito gnerador será el representado en la Figura 132:

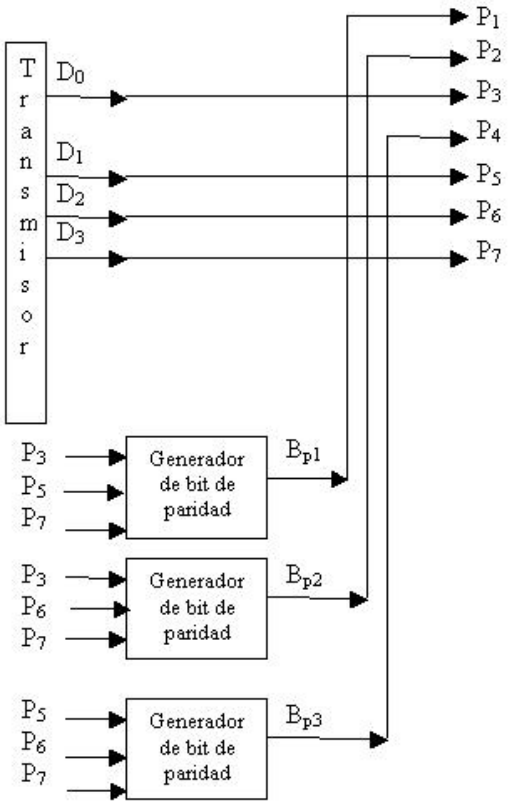


Figura 132

Observando el diagrama de bloques se concluye que es necesario del diseño de tres circuitos generadores de paridad, sin embargo, es evidente que basta con diseñar un único bloque ya que la funcionalidad de los tres es la misma y únicamente varian los nombres de las entradas y salidas. Así pues, la tabla de verdad será:

P_7	P_6	P_5	$P_4=B_{P3}$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Tabla 32

La función lógica que se obtiene es:

$$B_{P3} = P_7 \oplus P_6 \oplus P_5$$

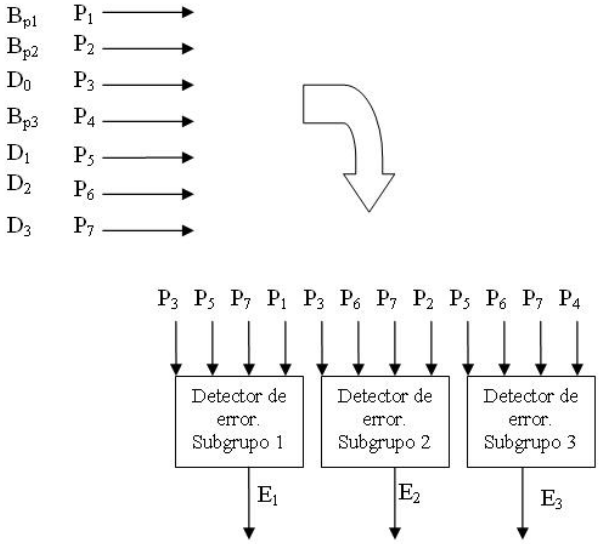
Aplicando esa lógica a los otros dos bloques

$$B_{P2} = P_7 \oplus P_6 \oplus P_3$$

$$B_{P1} = P_7 \oplus P_5 \oplus P_3$$

Circuito detector de error

Una vez transmitida la información, hay que chequear si ha ocurrido algún error en la transmisión. El diagrama de bloques correspondiente será:



Igual que en apartado anterior, basta con diseñar un único de los tres bloques que aparecen en el diagrama ya que la funcionalidad de los tres es la misma y únicamente varian los nombres de las entradas y salidas. Así pues, la tabla de verdad será:

P_7	P_6	P_5	$P_4=B_{P3}$	E_3
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Tabla 33

Donde $E_3 = 1$ indicará que ha ahbido error, es decir, un número impar de 1s en las entradas.

La función lógica que se obtiene es:

$$E_3 = P_7 \oplus P_6 \oplus P_5 \oplus P_4$$

Aplicando esa lógica a los otros dos bloques

$$E_2 = P_3 \oplus P_6 \oplus P_7 \oplus P_2$$

$$E_1 = P_3 \oplus P_5 \oplus P_7 \oplus P_1$$

Circuito corrector de error

Para implementar el circuito correcto de error, analicemos la información de las salidas E_1, E_2, E_3. Teniendo en cuenta que $E_i = 0$ indica que no hay error en el grupo correspondiente, la combinación $E_1= 0$ $E_2 = 0$ y $E_3 = 1$, significa que un bit del grupo E ha conmutado y además ese bit no está en el grupo de E_1 ni en el de E_2. Por lo tanto, deberá ser el bit de la posición 4.

Estudiando todas las combinaciones posibles de E_1, E_2, E_3.componemos la Tabla 34 donde B.E. indica la posición del Bit Erroneo. Se observa que el número binario que componen E_3 E_2 E_1 indica la posición del bit erroneo.

MSB	E_3	0	0	0	0	1	1	1	1
	E_2	0	0	1	1	0	0	1	1
LSB	E_1	0	1	0	1	0	1	0	1
	B.E	-	1	2	3	4	5	6	7

Tabla 34

$$E = E_3\, E_2\, E_1 \quad (E = 0, 1, 1, 3...7) \Rightarrow E \neq 0 \text{ ERROR}$$

Así pues, con la información contenida en E, se puede conocer el bit erroneo y por lo tanto corregirlo, es decir, complementarlo. El diagrama de bloques para este diseño se presenta en la Figura 133 donde C_{pi} indica que se debe corregir el bit de la posición p_i con i =3,5,6,7 (los bits de estas posicione son los únicos que conllevan la información que se deseaba transmitir). Las salidas D_{ic} se corresponden con los datos transferidos y corregidos.

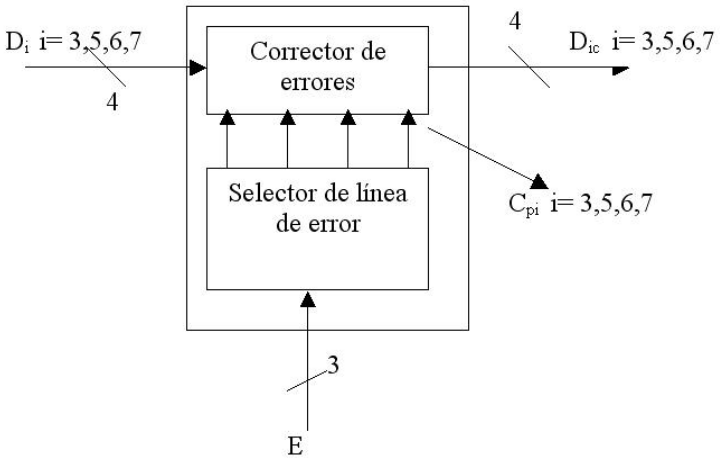


Figura 133

Las tablas de verdad para los bloques del diagrama son las siguientes:

1) Selector de línea erronea

La tabla de verdad será:

E_3	E_2	E_1	C_{p3}	C_{p5}	C_{p6}	C_{p7}
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	1	0	0	0
1	0	0	0	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Observando dicha tabla puede concluirse que este circuito puede implementarse utilizando puertas lógicas o un decodificador con tres entradas.

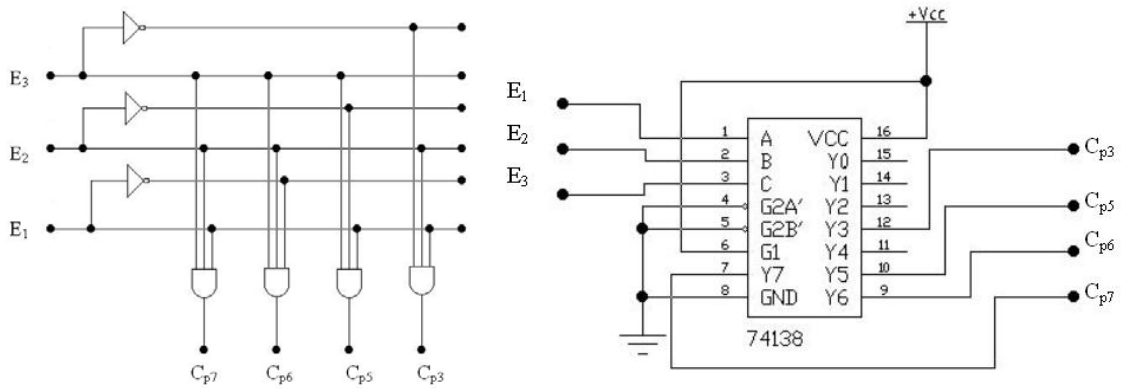


Figura 134

2) Corrector de errores

Si el dato D_i debe corregirse, es decir, $Z_{pi} = 1$, entonces, ese dato debe invertirse. La tabla de verdad correspondientea esa lógica y la función lógica obtenida son:

D_i	C_{pi}	D_{ic}
0	0	0
0	1	1
1	0	1
1	1	0

$$D_{ic} = D_i \oplus C_{Di}$$

Circuito completo

El circuito completo para la transmisión de una palabra de cuatro bits utlizando los códigos Hammnig se presenta en la Figura 135.

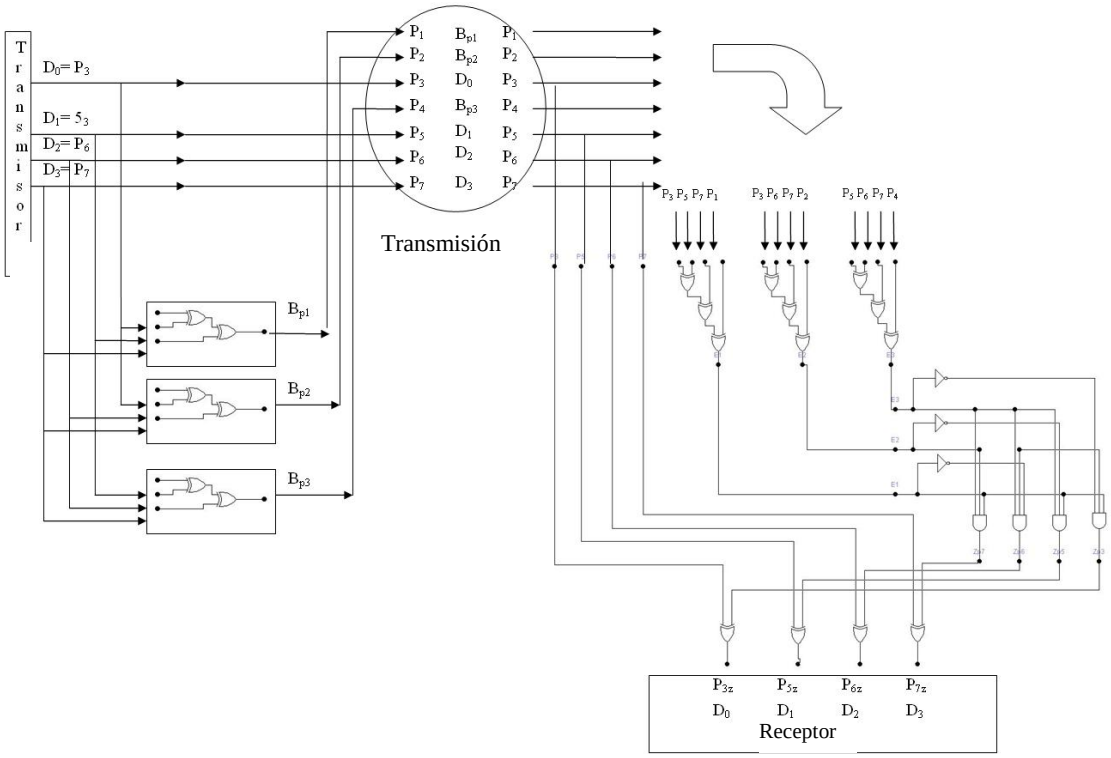


Figura 135

Circuito detector de doble error

La detección del doble error es muy importante ya que de ocurrir, el circuito receptor no detectaría error alguno puesto que el número de 1s leído sería par. Para detectar doble error, basta con añadir otro bit de paridad sobre el grupo de bits compuesto por: los *n* bits de la palabra inicial que contiene la información a transferir y, los *k* bits de paridad generados. Por lo tanto, la nueva palabra está compuesta por $n + k + 1$ bits. Este bit ocuparará la posición 8 y le denominamos B_{p8}

$$P8 = B_{p8} \propto (P7\ P6\ P5\ P4\ P3\ P2\ P1).$$

En la Figura 136 se representan los diagramas de bloques para el emisor y el receptor donde se han añadido los circuitos que generan B_{p8} y E_8. Este último circuito evaluará la paridad sobre los $n + k + 1$ bits

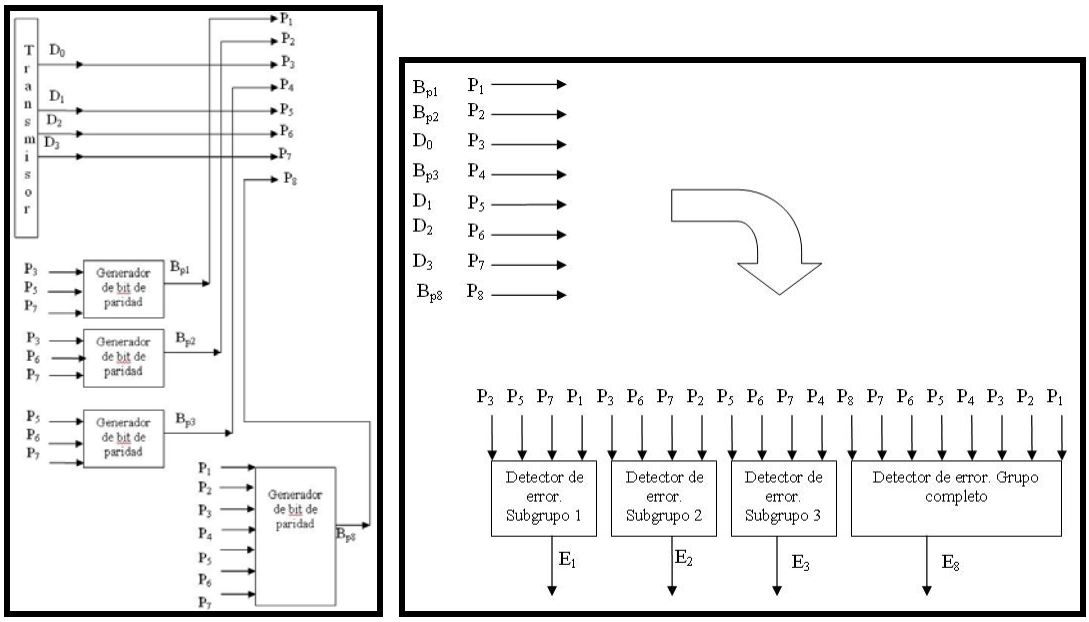


Figura 136

En caso de detectar doble error, ante la imposibilidad de detectar qué dos bits han mutado (ya que los ciruitos anteriores sólo son validos para el caso de un único error) y por lo tanto corregirlos, deberá diseñarse un circuito que solicite el reenvio de la información. Este circuito tendrá como entradas E_1, E_2, E_3. E_8 . Analizando la combinación de esta información se concluye:

Si E = 0 y E_8 = 0 Þ R = no ha habido error. no hay que solicitar el reenvio de datos

Si E ≠ 0 y E_8 = 1 Þ R = ha habido un error en alguno de los subgrupos. Entonces, se corrige y no hay que solicitar el reenvio de datos.

Si E ≠ 0 y E_8 = 0 Þ R = ha habido doble error. Debe solicitarse el reenvio de datos.

Si E = 0 y E_8 = 1 Þ R = ha habido error en el bit de paridad B_{p8}. No hay que solicitar el reenvio de datos

A partir de las líneas anteriores, la tabla de verdad correspondiente al circuito que debe dar la señal de reenvio de información será:

E_8	E_3	E_2	E_1	R
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

R = 1 significará que se debe reenviar la información.

La función lógica que se obtiene es:

$$R = \overline{E}_8\left(E_1 + E_2 + E_3\right) + E_8\overline{E}_3\overline{E}_2\overline{E}_1$$

Si al circuito de la Figura 135. se le añade el circuito detector de dos errores el resultado sería el presentado en la Figura 137.

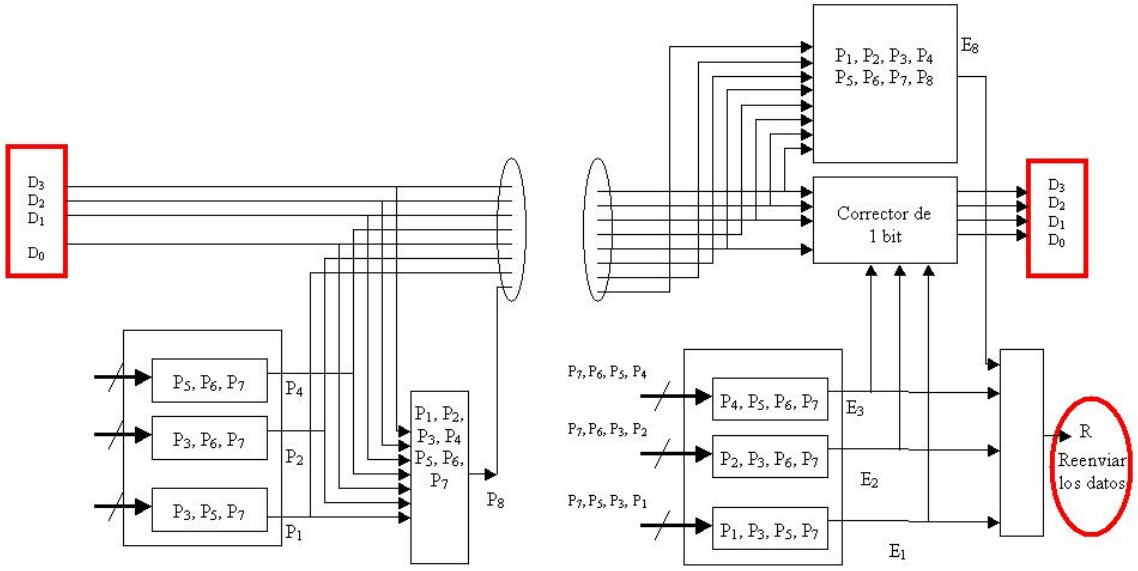


Figura 137

11.7. Ejemplo de aplicación de código Hamming

Se utilizará paridad par.

Datos a transmitir: 1010 , y su posición 3 5 6 7

Bits de paridad: P_1 $(D_3D_5D_7) = 1$

P_2 $(D_3D_6D_7) = 0$

P_4 $(D_5D_6D_7) = 1$

Paquete completo a transmitir

Posición	1	2	3	4	5	6	7
Palabra	1	0	1	1	0	1	0

Veamos cómo aplicando las funciones lógicas obtenidas, funciona correctamente la codificación.

Supongamos error en la posición 3: $1 \rightarrow 0$

Posición	1	2	3	4	5	6	7
Palabra	1	0	0	1	0	1	0

$C_4 = D_7 \oplus D_6 \oplus D_5 \oplus B_{P4} = 0$

$C_2 = D_3 \oplus D_6 \oplus D_7 \oplus B_{P2} = 1$ ⇒ B. E. = 3, el bit erroneo está en la posición 3

$C_1 = D_3 \oplus D_5 \oplus D_7 \oplus B_{P1} = 1$

Supongamos error en la posición 2: $0 \rightarrow 1$

Posición	1	2	3	4	5	6	7
Palabra	1	1	1	1	0	1	0

$C_4 = D_7 \oplus D_6 \oplus D_5 \oplus B_{P4} = 0$

$C_2 = D_3 \oplus D_6 \oplus D_7 \oplus B_{P2} = 1$ ⇒ B. E. = 2, el bit erroneo está en la posición 2

$C_1 = D_3 \oplus D_5 \oplus D_7 \oplus B_{P1} = 0$

Supongamos error en la posición 3: $1 \rightarrow 0$ y en laposición 2: $0 \rightarrow 1$

Posición	1	2	3	4	5	6	7
Palabra	1	1	0	1	0	1	0

$C_4 = D_7 \oplus D_6 \oplus D_5 \oplus B_{P4} = 0$

$C_2 = D_3 \oplus D_6 \oplus D_7 \oplus B_{P2} = 0$ ⇒ B. E. = 1, la ecuación indica que el. bit erroneo está en la posición 1, lo cual es incorrecto

$C_1 = D_3 \oplus D_5 \oplus D_7 \oplus B_{P1} = 1$

Si se añade $P_8 = 0$ a los ejemplos anteriores

Posición	1	2	3	4	5	6	7	8
Palabra	1	0	1	1	0	1	0	0

Supongamos error en la posición 3: $1 \rightarrow 0$

Posición	1	2	3	4	5	6	7	8
Palabra	1	0	0	1	0	1	0	0

$C_4 = D_7 \oplus D_6 \oplus D_5 \oplus B_{P4} = 0$

$C_2 = D_3 \oplus D_6 \oplus D_7 \oplus B_{P2} = 1$ ⇒ B. E. = 3, el bit erroneo está en la posición 3

$C_1 = D_3 \oplus D_5 \oplus D_7 \oplus B_{P1} = 1$

$C_8 = 1$

Supongamos error en la posición 3: $1 \rightarrow 0$ y en la posición 2: $0 \rightarrow 1$

Posición	1	2	3	4	5	6	7	8
Palabra	1	1	0	1	0	1	0	0

$C_4 = D_7 \oplus D_6 \oplus D_5 \oplus B_{P4} = 0$

$C_2 = D_3 \oplus D_6 \oplus D_7 \oplus B_{P2} = 0$ ⇒ Las ecuaciones indican que ha habido dos errores

$C_1 = D_3 \oplus D_5 \oplus D_7 \oplus B_{P1} = 1$

$C_8 = 0$