8 Códigos detectores y correctores de error

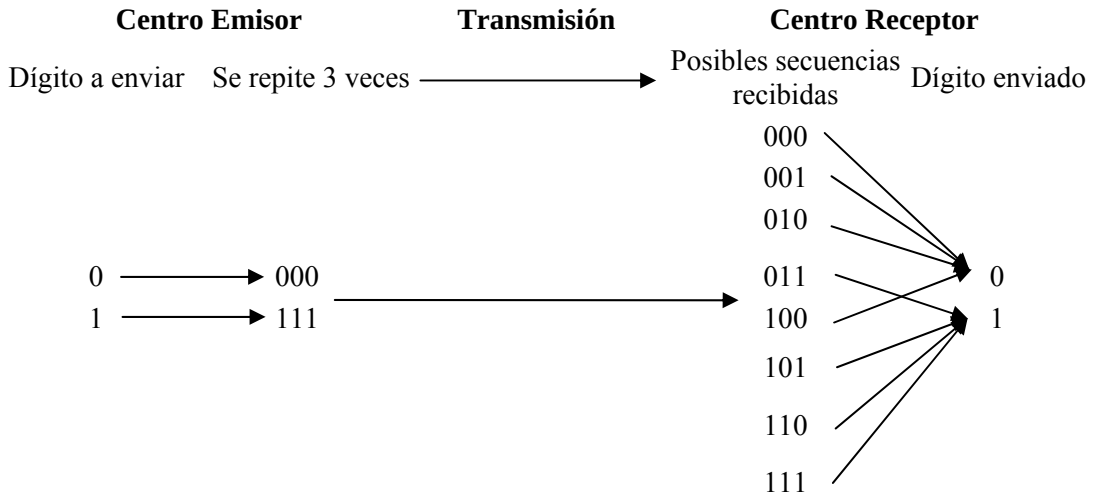
1.- INTRODUCCIÓN

Estudiada la fiabilidad de los canales de transmisión, vamos a centrarnos en este tema en la posibilidad de transmitir mensajes confiables por canales no confiables.

Una conclusión importante del tema anterior es que existe la posibilidad de que el mensaje recibido no sea el transmitido. Ningún canal real está libre de error , por lo que por muy pequeña que sea la probabilidad de éste en los canales actuales: del orden de 10^{-6}, 10^{-8} e inferiores, su existencia impone la necesidad de mecanismos ajenos al medio de transmisión que mejoren la calidad de ésta.

El mecanismo más inmediato es la simple repetición de cada símbolo transmitido un número n de veces. Si los n símbolos recibidos son iguales, se supondrá que en ninguna de las transmisiones ha existido error y, por lo tanto, el centro receptor identificará como símbolo transmitido el n veces repetido. Si los n símbolos recibidos no son todos iguales, implicará error en alguna de las transmisiones y se identificará como símbolo transmitido más probable al que aparezca repetido más veces en la recepción. Es fácil observar como éste simple mecanismo mejora la fiabilidad de la transmisión. Veámoslo con un ejemplo:

Ej.1.- Consideremos un canal BSC con una probabilidad de error p de 0.01, es decir, el 99% de los símbolos binarios transmitidos es recibido correctamente. Con objeto de aumentar la fiabilidad del canal repetimos el envío de cada dígito binario 3 veces. El proceso se representa en el siguiente esquema de emisión-transmisión de un dígito binario:



La probabilidad de que no exista error en la transmisión es: $(1 - p)^3 = (\overline{p})^3$.

La probabilidad de un solo error es: $3\ p\ \overline{p}^2$.

La probabilidad de 2 errores es: $3\ p^2\ \overline{p}$.

Y la probabilidad de que los tres símbolos binarios recibidos sean erróneos es: p^3.

Aplicando la regla de decisión indicada se tiene que la probabilidad de interpretar el mensaje erróneamente es igual a la suma de las probabilidades de que varíen dos o tres símbolos binarios durante la transmisión, o sea: $p^3 + 3\ p^2\ \overline{p} \approx 3*10^{-4}$.

La probabilidad de error ha pasado de 10^{-2} *si no existe repetición en la transmisión a* $3*10^{-4}$ *si repetimos 3 veces cada símbolo emitido.*

Vista la mejora de la fiabilidad de la transmisión, es fácil de inducir que la probabilidad de error disminuirá con el número de repeticiones, pero a costa de incrementar el tiempo de la transmisión, por lo que el procedimiento repetitivo descrito plantea un compromiso entre la velocidad de los mensajes y su fiabilidad. Por ejemplo, en el caso anterior se ha disminuido el porcentaje de error pero a costa de pasar de la transmisión de un mensaje por dígito binario a la de uno por cada tres dígitos binarios.

Este método de corrección de errores aunque efectivo, no es el más eficaz que se puede utilizar; a continuación vamos a estudiar métodos más sofisticados que logran una mayor velocidad en la transmisión de los mensajes mediante una adecuada codificación de estos.

En los apartados siguientes hablaremos de la transmisión de palabras de código, para su mejor comprensión, y evitar así cualquier error, es conveniente señalar que en realidad los canales utilizados son binarios, por lo que la transmisión de una palabra de código supone la transmisión individual de cada uno de los n caracteres binarios que la componen. Dicho de otra forma, el proceso de transmisión de una palabra de código por un canal binario, hay que verlo como la extensión de orden n (n=longitud de la palabra de código) del canal binario.

2.- **CÓDIGOS DETECTORES DE ERROR**

Vamos a estudiar una serie de códigos cuya principal característica es que permiten detectar si se ha producido o no error en la transmisión de la palabra de código. Aunque en caso de producirse no son capaces de su corrección inmediata, su detección evita el uso de información incorrecta. El error se soluciona repitiendo la transmisión hasta que éste desaparezca. La ventaja de esta metodología frente a la anterior es inmediata: el mensaje sólo se repite en caso de error.

Si en un código binario de longitud constante (como sólo nos referiremos a este tipo de códigos, a partir de ahora obviaremos el indicar explícitamente esta característica) se utilizaran todas las combinaciones posibles de sus n dígitos binarios (2^n), resultaría imposible detectar si se ha producido error, ya que una combinación del código se transformaría en otra que también pertenece a él. Quiere esto decir, que la posibilidad de detectar errores se logra no utilizando todas la combinaciones posibles, de forma que al recibir una determinada combinación se puede identificar como errónea si no pertenece al código. Por ejemplo:

Ej. 2.- Queremos transmitir dígitos decimales para lo que usamos el código BCD natural. En éste no se emplean más que 10 de las 16 combinaciones posibles con 4 dígitos binarios. Supongamos que se emite la combinación 0011 y tras producirse error en un dígito binario durante la transmisión se recibe la 1011; como no pertenece al código utilizado es inmediato deducir que nunca ha podido ser transmitida, por lo que el error será detectado.

Sin embargo, esta condición es necesaria pero no suficiente. Veámoslo con un contraejemplo:

Ej. 3.- Retomemos el ejemplo anterior y supongamos ahora que al transmitir la combinación 0011 se produce el error en el primer dígito, recibiendo la 0010. Como también pertenece al código el centro receptor la considerará correcta, identificando, erróneamente, al dígito decimal 2 como el mensaje emitido.

De lo expuesto es fácil de deducir intuitivamente que la condición necesaria y suficiente para poder detectar error en un dígito binario es que toda combinación del código se transforme, al variar uno cualquiera de sus dígitos, en una combinación que no pertenezca al código. En la Figura 1 se puede observar gráficamente la condición expuesta.

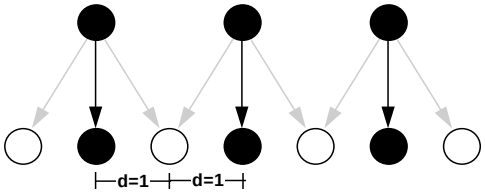


Figura 1. Los círculos sombreados representan combinaciones pertenecientes al código y los blancos combinaciones no pertenecientes al código. Las flechas negras indican transmisiones libres de error, mientras que con las grises se señalan las transmisiones con error en un dígito.

Los códigos que cumplen esta condición son aquellos cuya distancia mínima es mayor o igual a 2. Relacionando ambas afirmaciones se puede concluir que la condición necesaria y suficiente para que un código sea detector de error en un dígito binario es que su distancia mínima sea, como mínimo, dos.

Generalizando el razonamiento anterior se puede afirmar que la condición necesaria y suficiente para que un código permita detectar el error producido por la variación de un número de dígitos igual o menor que k durante la transmisión de la palabra de código, es que su distancia mínima sea, al menos, k+1. Gráficamente se puede ver en la Figura 2.

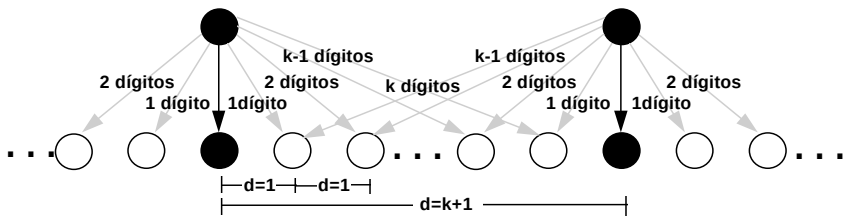


Figura 2. La simbología es la misma que en la figura anterior, sólo se añade, en las flechas grises, el número de dígitos que varían en la transmisión. Se puede ver como el código detector de k errores lo será también de un número de errores inferior a k.

De los múltiples códigos detectores de error cabe destacar, de entre los detectores de un solo error, a los de paridad y a los de peso constante.

2.1.- **Códigos de paridad**

Concepto de paridad: Se dice que una combinación binaria tiene paridad par si el número de unos de esa combinación es par. De igual forma se dice que una combinación tiene paridad impar si su número de unos es impar.

Síntesis del código

Un código de paridad se obtiene añadiendo a las palabras de un código de distancia mínima uno, un dígito que se denomina de paridad. Si el código que se desea obtener es de paridad par, este dígito debe adquirir un valor tal que la paridad de cada combinación sea par. Igual criterio se aplica si el código deseado es de paridad impar.

Ej. 4.- Vamos a crear un código de paridad para los dígitos decimales a partir del código BCD natural

Dígito decimal	Código de paridad	
	Código BCD natural	Dígito de paridad
0	0000	0
1	0001	1
2	0010	1
3	0011	0
4	0100	1
5	0101	0
6	0110	0
7	0111	1
8	1000	1
9	1001	0

Para verificar que los códigos de paridad son detectores de error en un dígito binario basta con probar que su distancia mínima es dos, para lo que se demostrarán las siguientes dos proposiciones:

a) Un código de paridad par posee una distancia mínima estrictamente mayor que uno.

Razonamos por reducción al absurdo. Supóngase que existen dos combinaciones $b_k\, b_{k-1} \ldots b_1\, b_0$ y $a_k\, a_{k-1} \ldots a_1\, a_0$, con una distancia entre ellas de uno. Esto implica que existe un bit, por ejemplo el j-ésimo, que hace que:

$$b_i = a_i \quad \forall i \in [0, k] / i \neq j$$
$$\overline{b_j} = a_j$$

De aquí se infiere que si la primera combinación posee m 1´s, la segunda tiene m+1 ó m-1. En cualquier caso si m es par m+1 ó m-1 son números impares, con los cual la segunda combinación no pertenecería al código. Este razonamiento lleva a un absurdo, con lo que queda demostrada la proposición.

b) En un código de paridad par existen siempre dos combinaciones que distan dos unidades.

Como el código de partida es de distancia mínima uno, existirán al menos dos subcombinaciones $b_{k-1} \ldots b_1 b_0$ y $a_{k-1} \ldots a_1 a_0$, en las que solamente un índice j hace que:

$$b_i = a_i \quad \forall i \in [0, k] / i \neq j$$
$$\overline{b}_j = a_j$$

Si la primera combinación posee m 1´s, la segunda tendrá m+1 ó m-1. Supóngase que m es par. Entonces los respectivos bits de paridad serán b_k=0 y a_k=1. Si m fuera impar los bits de paridad serían b_k=1 y a_k=0. En cualquier caso se cumple que $\overline{b}_k = a_k$. Por lo tanto, la distancia de estas dos combinaciones resultante es dos ($\overline{b}_k = a_k$ y $\overline{b}_j = a_j$).

En esta demostración se ha trabajado con códigos de paridad par. El razonamiento es el mismo si fuese de paridad impar.

Análisis del código

Para detectar si existe o no error en la palabra de código recibida, se comprueba si ésta cumple el criterio de paridad preestablecido, si es así se supondrá que no ha existido error en la transmisión, si no es así es que algún dígito ha variado de valor, no podemos saber cual es, pero sí que ha existido error.

Ej. 5.- Supongamos que el criterio de paridad del código usado es par. Entonces si se recibe:

10001 Su paridad es par $\Rightarrow$ será correcta.
10000 Su paridad es impar $\Rightarrow$ será incorrecta.
01110 Su paridad es impar $\Rightarrow$ será incorrecta.
01100 Su paridad es par $\Rightarrow$ será correcta.

Como comentario final se puede añadir que por la forma de operar de estos códigos, permitirán la detección de error siempre que el número de dígitos binarios que varíen sea impar.

2.2.- **Códigos de peso constante**

Se denomina peso de una combinación binaria al número de unos que posee. Entonces, los códigos de peso constante serán aquellos cuyas combinaciones tienen siempre la misma cantidad de unos.

Entre estos códigos de peso constante se encuentran los BCD 2 entre 5 y biquinario o 2 entre 7, que es, además ponderado:

Dígito decimal	Código 2 entre 5	Código biquinario *peso* 5 0 4 3 2 1 0
0	0 1 1 0 0	0 1 0 0 0 0 1
1	1 1 0 0 0	0 1 0 0 0 1 0
2	1 0 1 0 0	0 1 0 0 1 0 0
3	1 0 0 1 0	0 1 0 1 0 0 0
4	0 1 0 1 0	0 1 1 0 0 0 0
5	0 0 1 1 0	1 0 0 0 0 0 1
6	1 0 0 0 1	1 0 0 0 0 1 0
7	0 1 0 0 1	1 0 0 0 1 0 0
8	0 0 1 0 1	1 0 0 1 0 0 0
9	0 0 0 1 1	1 0 1 0 0 0 0

Se puede comprobar que estos códigos poseen una distancia mínima de dos. El error se detecta cuando la combinación recibida tenga un número de unos distinto al peso del código usado.

Se puede añadir el mismo comentario que a los de paridad.

3.- **CÓDIGOS CORRECTORES DE ERRORES**

Estos códigos permiten, además de detectar el error, corregirle sin necesidad de repetir la transmisión. De forma general, se puede decir que la técnica empleada para la corrección consiste en identificar como combinación correcta, a la perteneciente al código que sea más cercana a la errónea recibida, o sea, aquella cuya distancia de la combinación incorrecta sea menor.

Veamos, al igual que antes, las propiedades de estos códigos partiendo del caso más sencillo:

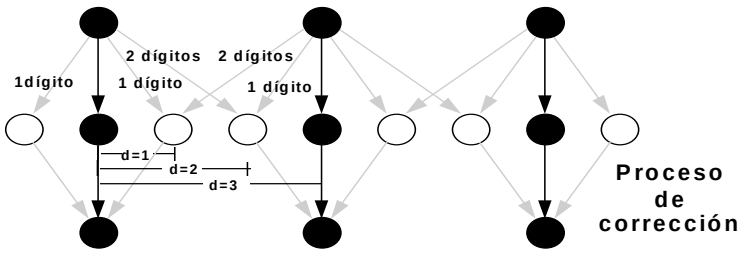


Figura 3. Se utiliza la misma simbología que en figuras anteriores. Conviene fijarse como un código corrector de orden 1 sólo corrige correctamente si hay un error en la transmisión. Esta afirmación es extrapolable a códigos correctores de cualquier orden.

códigos capaces de corregir el error cometido al variar un dígito binario.

Puesto que se ha de detectar el error, no deben de usarse todas las combinaciones posibles (como se vio en la pregunta anterior), pero, además, como el proceso de corrección ha de ser no ambiguo, sólo debe de haber una posible combinación del código que diste una unidad de cada

combinación no usada. En la Figura 3 se puede observar esquemáticamente el proceso de corrección. Se puede concluir, que la condición necesaria y suficiente para que un código sea corrector de orden uno (corrija correctamente errores producidos al variar un dígito en la transmisión) es que su distancia mínima sea tres. Veamos un ejemplo que ilustra lo expuesto:

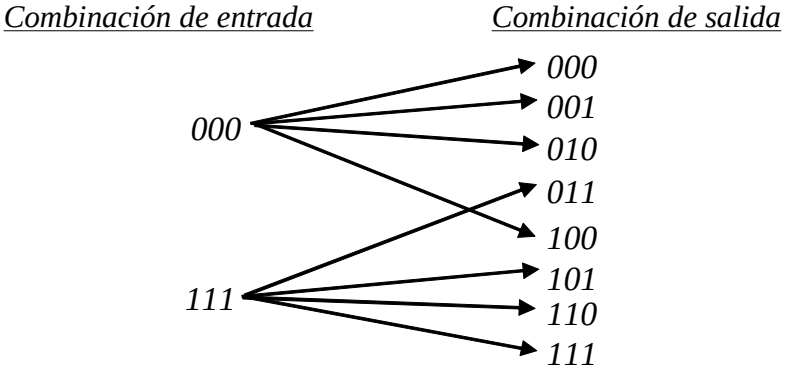
Ej. 6.- Queremos transmitir dos mensajes A y B. Si la distancia mínima del código usado es menos que tres, es imposible una corrección no ambigua:

código I: *A.....00*
B.....11

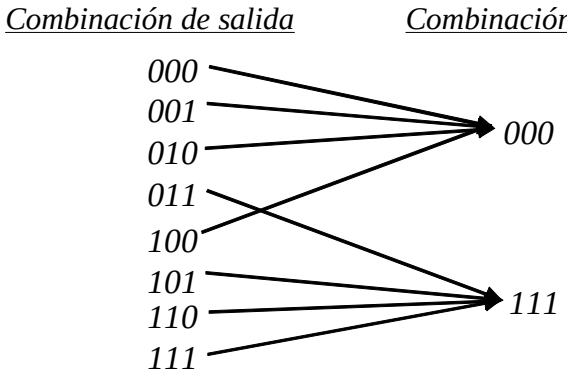
Supongamos que usando este código se recibe la combinación 01, evidentemente ha existido un error, y suponiendo que ha sido en un solo dígito es imposible discernir si la combinación enviada ha sido la 00 o la 11 ya que ambas distan una unidad de la recibida. Creemos ahora un código de distancia mínima tres:

código II: *A.....000*
B.....111

Las entradas y salidas del canal serían (suponiendo que no puede haber más de un error en la transmisión de cada combinación):



Es inmediato ver como en el proceso de corrección no existe ninguna ambigüedad:



Aunque la restricción de que no existe más de un error en la transmisión de cada combinación es aplicable en numerosos canales reales, es conveniente generalizar el proceso de corrección a un orden k: Si durante la transmisión de una combinación varían un número de dígitos igual o inferior a k, se recibirá una combinación no perteneciente al código; para que este error pueda se corregido, debe de cumplirse que la combinación emitida sea la única perteneciente al código cuya distancia de la recibida sea igual o inferior a k. Generalizando el razonamiento a cualquier combinación transmitida se puede concluir que la condición necesaria y suficiente para que un código sea corrector de orden k es que su distancia mínima sea de 2k+1; sólo en este caso el proceso de corrección será no ambiguo. En la Figura 4 se puede observar una esquematización gráfica del proceso de corrección de orden k que puede ayudar a la comprensión de lo anteriormente expuesto.

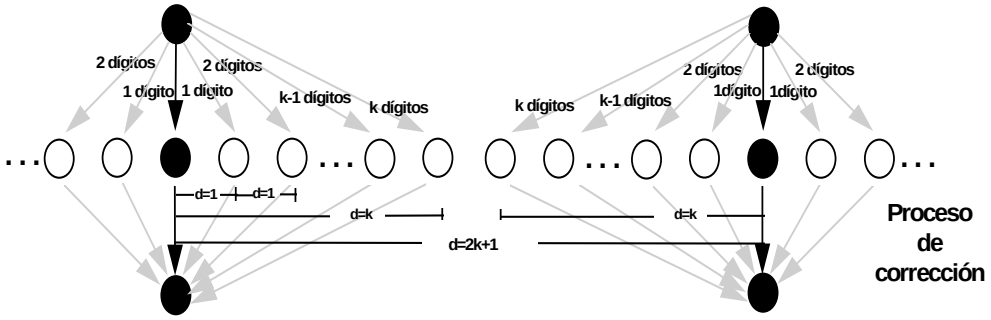


Figura 4. Representación simbólica de un proceso de corrección de k errors.

De los posibles códigos correctores de errores vamos a ver únicamente los denominados códigos de Hamming.

3.1.- **Códigos de Hamming**

Con este nombre se conoce a un conjunto de códigos correctores en k dígitos binarios. En esta lección solamente se tratará el de orden uno: k=1.

A la hora de trabajar con este tipo de códigos podemos distinguir dos operaciones:

a) Construcción, que se realizará en el centro emisor.
b) Interpretación, que se realizará en el centro receptor.

a) Construcción. Se parte de un código de n dígitos de distancia mínima uno[1]. Estos n dígitos son conocidos dentro del código de Hamming como "dígitos de datos". A continuación se le añaden p (c_{p-1}, ..., c_2, c_1, c_0) dígitos denominados de control o paridad. Así pues, el nuevo código tendrá una longitud de palabra de l=n+p. La numeración de los dígitos es la habitual (de derecha a izquierda) pero comenzando por uno: d_{n+p} d_{n+p-1} ... d_2 d_1.

[1] Esta condición de partida, código de distancia mínima uno, no es estrictamente imprescindible, pero de no ser así, se añade redundancia innecesaria al código resultante (existen palabras de código innecesariamente no utilizadas).

Cada uno de estos p dígitos que añadimos al código original va a afectar a unas determinadas posiciones de la nueva palabra de código de n+p dígitos, de forma que tomaran el valor adecuado para que se cumpla el criterio de paridad (par o impar) preestablecido en las subcombinaciones afectadas por cada uno. Se tiene, entonces, que en la construcción del código los p dígitos añadidos actúan como dígitos de paridad.

b) Interpretación. Recibida una combinación de un código de Hamming hay que comprobar si es correcta, y de no ser así habrá que detectar el dígito que varió en la transmisión.

Ahora los p dígitos añadidos actúan como dígitos de control y con ellos formamos una palabra binaria. Cada uno de los dígitos de esta palabra toma el valor 0 ó 1 dependiendo de si el número de unos de las posiciones de la palabra de código por el afectadas cumplen o no el criterio de paridad establecido. Interpretando la combinación resultante en binario natural, tendremos dos posibilidades:

- Que se corresponda con el 0. Entonces quiere decir que la transmisión ha sido correcta.

- Que se corresponda a un número distinto del 0. Entonces en la transmisión ha variado el dígito situado en la posición indicada por ese número.

Quedan varias cuestiones por resolver:

I.- Cómo calcular p.
II.- A que posiciones afecta cada uno de los p dígitos de control o paridad.
III.- Dónde se colocan estos dígitos dentro de la palabra de código.

I.- Dada la forma de calcular la posición errónea, con p dígitos binarios se tiene que poder detectar el error en todas y cada una de la n+p posiciones de la palabra de código. Como la combinación formada por los p dígitos de control se interpreta en binario natural, se debe cumplir que:

$$2^p - 1 \geq n + p$$

Donde 2^p-1 es el mayor número que se puede representar en binario natural con p dígitos.

II.- Construyamos todas las combinaciones posibles con p dígitos de control, e interpretemos cada una en binario natural:

c_{p-1} ... c_2 c_1 c_0	Posición
0 ... 0 0 0	0
0 ... 0 0 1	1
0 ... 0 1 0	2
0 ... 0 1 1	3
0 ... 1 0 0	4
0 ... 1 0 1	5
. . .	

Cada dígito de control ha de afectar a aquellas posiciones en las que sea capaz de detectar error, o sea, va a afectar a las posiciones de la tabla anterior para las que ese dígito valga 1:

Dígito		Posiciones
c_o	.....	1, 3, 5, 7, 9, 11, 13, ...
c_1	.....	2, 3, 6, 7, 10, 11, 14, 15, ...
c_2	.....	4, 5, 6, 7, 12, 13, 14, 15, ...
	. . .	
c_p	.....	2^p, 2^p+1, 2^p+2, ...

III.- Han de colocarse en aquellas posiciones en las que no se vean afectados por otro dígito de control, así no existirán ambigüedades a la hora de otorgarles valor en la creación del código. Estas posiciones han de ser entonces:

Dígito		Posición		Combinación en binario natural
c_0	.....	2^0	.....	0 ... 001
c_1	.....	2^1	.....	0 ... 010
c_2	.....	2^2	.....	0 ... 100
	. . .			

Veamos un ejemplo de creación e interpretación de un código de Hamming de paridad par:

Ej. 7.- Partimos del código binario natural de 4 bits.

a) Creación. Dividimos el proceso en una secuencia lógica de pasos:

1.- Cálculo del número de dígitos de control necesarios:

$2^p - 1 \geq n+p$

$\Rightarrow 2^p \geq 5 + p \Rightarrow p = 3$

$n = 4$

La palabra de código tendrá, entonces, una longitud l=4+3=7 dígitos: $d_7d_6d_5...d_1$. Para identificar los dígitos de control les denominamos c_2, c_1 y c_0.

2.- Hallamos las posiciones de la palabra de código afectadas por cada dígito de control.

c_2	c_1	c_0	Posición
0	0	1	 1
0	1	0	 2
0	1	1	 3
1	0	0	 4
1	0	1	 5
1	1	0	 6
1	1	1	 7

Los controles de paridad se efectúan sobre las siguientes subcombinaciones:

c_0 .- d_1, d_3, d_5, d_7.
c_1 .- d_2, d_3, d_6, d_7.
c_2 .- d_4, d_5, d_6, d_7.

3.- Cada dígito de control estará situado en las siguientes posiciones de la palabra de código:

c_o .- d_1 *c_1 .- d_2* *c_2 .- d_4*

4.- Construcción del código de Hamming:

				c_2		c_1	c_0					c_2		c_1	c_0
	d_7	d_6	d_5	d_4	d_3	d_2	d_1		d_7	d_6	d_5	d_4	d_3	d_2	d_1
0	0	0	0	**0**	0	**0**	**0**	8	1	0	0	**1**	0	**1**	**1**
1	0	0	0	**0**	1	**1**	**1**	9	1	0	0	**1**	1	**0**	**0**
2	0	0	1	**1**	0	**0**	**1**	10	1	0	1	**0**	0	**1**	**0**
3	0	0	1	**1**	1	**1**	**0**	11	1	0	1	**0**	1	**0**	**1**
4	0	1	0	**1**	0	**1**	**0**	12	1	1	0	**0**	0	**0**	**1**
5	0	1	0	**1**	1	**0**	**1**	13	1	1	0	**0**	1	**1**	**0**
6	0	1	1	**0**	0	**1**	**1**	14	1	1	1	**1**	0	**0**	**0**
7	0	1	1	**0**	1	**0**	**0**	15	1	1	1	**1**	1	**1**	**1**

b) Interpretación. Analicemos todos los casos posibles de error en un bit.

1.- Alteración de un bit de datos:

	d_7	d_6	d_5	d_4	d_3	d_2	d_1
Combinación transmitida	*0*	*1*	*1*	*0*	*1*	*0*	*0*
Combinación recibida	*1*	*1*	*1*	*0*	*1*	*0*	*0*

- Control de paridad de c_0 $\rightarrow d_7\, d_5\, d_3\, d_1$; 3 unos $\rightarrow$ impar $\Rightarrow c_o = 1$.
- Control de paridad de c_1 $\rightarrow d_7\, d_6\, d_3\, d_2$; 3 unos $\rightarrow$ impar $\Rightarrow c_1 = 1$.
- Control de paridad de c_2 $\rightarrow d_7\, d_6\, d_5\, d_4$; 3 unos $\rightarrow$ impar $\Rightarrow c_2 = 1$.

El bit erróneo es: $c_2\, c_1\, c_0 = 1\ 1\ 1_{(2} = 7$.

2.- Alteración de un bit de control:

	d_7	d_6	d_5	d_4	d_3	d_2	d_1
Combinación transmitida	*0*	*1*	*1*	*0*	*1*	*0*	*0*
Combinación recibida	*0*	*1*	*1*	*1*	*1*	*0*	*0*

- Control de paridad de c_0 $\rightarrow d_7\, d_5\, d_3\, d_1$; 2 unos $\rightarrow$ impar $\Rightarrow c_o = 0$.
- Control de paridad de c_1 $\rightarrow d_7\, d_6\, d_3\, d_2$; 2 unos $\rightarrow$ impar $\Rightarrow c_1 = 0$.
- Control de paridad de c_2 $\rightarrow d_7\, d_6\, d_5\, d_4$; 3 unos $\rightarrow$ impar $\Rightarrow c_2 = 1$.

El bit erróneo es: $c_2\, c_1\, c_0 = 1\ 0\ 0_{(2} = 4$.

3.- No hay alteración:

	d_7	d_6	d_5	d_4	d_3	d_2	d_1
Combinación transmitida	*0*	*1*	*1*	*0*	*1*	*0*	*0*
Combinación recibida	*0*	*1*	*1*	*0*	*1*	*0*	*0*

- Control de paridad de $c_0 \rightarrow d_7\, d_5\, d_3\, d_1$ *; 2 unos* $\rightarrow$ *impar* $\Rightarrow c_o = 0$.

- Control de paridad de $c_1 \rightarrow d_7\, d_6\, d_3\, d_2$ *; 2 unos* $\rightarrow$ *impar* $\Rightarrow c_1 = 0$.

- Control de paridad de $c_2 \rightarrow d_7\, d_6\, d_5\, d_4$ *; 2 unos* $\rightarrow$ *impar* $\Rightarrow c_2 = 0$.

Como $c_2\, c_1\, c_0 = 0\ 0\ 0_{(2} = 0$*, entonces no existe error en la combinación recibida.*

4.- **BIBLIOGRAFÍA**

Norman Abramson, "Teoría de la Información y Codificación", Paraninfo, 1986.

Enrique Mandado, "Sistemas Electrónicos Digitales", Marcombo, 1987.

J.-P. Meinadier, "Estructura y Funcionamiento de las Computadores Digitales", AC, 1986.

Pedro de Miguel Anasagasti, "Fundamentos de los computadores", Paraninfo, 1988.

Ejercicios propuestos

1. Crear un código binario que represente los dígitos 0 a 3 que tenga distancia mínima 3. ¿Cuántos errores podemos detectar y corregir?.

2. ¿Cuál es la distancia mínima necesaria de un código para poder detectar errores de 3 bits?. ¿Y para corregirlos?.

3. Crear un código binario que permita representar

   a.) los siete días de la semana,
   b.) los caracteres del alfabeto.
   c.) los caracteres del alfabeto distinguiendo entre mayúsculas y minúsculas.

   . Modifique los códigos anteriores para obtener otros que permitan detectar errores de un dígito binario.
   . ¿Qué debería hacerse para convertirlos en códigos de distancia mínima 3?.
   . Si estos códigos se utilizan en un sistema informático en el que una vez enviada la información es imposible repetirla, construya los respectivos códigos que solucionen este problema para errores de 1 dígito binario.

   En todos los casos calcule su eficacia y redundancia, suponiendo símbolos equiprobables.

4. Partiendo de los códigos siguientes, construir los correspondientes códigos de Hamming de paridad impar e impar correctores de errores de un dígito binario:

   a.) Código binario natural de 3 y 5 dígitos binarios,
   b.) Código BCD exceso a 3.

5. Interpretar las siguientes combinaciones binarias de un código de Hamming de paridad impar (diciendo la información contenida en ellos), indicando si hay error en qué posición y corregirlo:

Código Gray de 4 bits	ASCII extendido 8 bits
i. 100 0000	iv. 0011 1011 1011
ii. 000 1100	v. 0011 1000 0000
iii. 000 0100	vi. 1011 1000 0000

6. Interprete la siguiente secuencia de 32 dígitos binarios

   0100 0010 1000 0010 0110 0000 1010 0000

   como datos del código 2 entre 7 con dígito de paridad par en último lugar (ccccccP; c = bit de código, P = bit de paridad), Indicar cuando el dato almacenado es erróneo. Si los datos representados son numéricos, escriba su equivalente en base diez.

7. a.) Un dato y su dígito de paridad se copian de una parte a otra de un ordenador. La subsiguiente comprobación de paridad falla. ¿Es cierto que el dato es ahora incorrecto? Explicar la respuesta.

   b.) Si, en el caso anterior, la comprobación de paridad no falla, ¿es cierto que el dato es correcto? Explicar la respuesta.

   c.) A la luz de sus respuestas en los apartados a y b, comentar brevemente la utilidad de las comprobaciones de paridad.

8. Sea un código instantáneo se le añade un dígito binario de paridad a la derecha de cada palabra de código, ¿seguirá siendo un código instantáneo?

9. A un código de distancia mínima 1 se le añade un dígito de paridad par. Al código así resultante se le agrega un nuevo dígito ahora de paridad impar. Hallar y justificar la distancia mínima del nuevo código.

10. Dadas las siguientes secuencias de dígitos binarios:

    i) 000011111111001101101 ii) 10001110101101.01101001011100

interpretarlas como secuencias de palabras del código de Hamming de paridad par formado a partir del BCD natural. Corregir cada palabra en el caso de que haya algún dígito erróneo, obtener el dígito decimal representado y obtener las cantidades representadas en cada secuencia.

11. Interpretar la siguiente secuencia binaria perteneciente a un código corrector de Hamming construido a partir jdel código BCD natural, con criterio de paridad par.:

    00000001011110010101010111

12. Los dígitos decimales son representados en las calculadoras deigitales mediante 7 segmentos, de la forma descrita en la figura. A esta forma de representación se la puede considerar como una codificación de los dígitos decimales.
a) Identificar el alfabeto código.
b) ¿Será un código detector de un error?

13. Se desea transmitir los resultados de un experimento realizado en el laboratorio A hacia el B.

a) Sabiendo que el experimento tiene dos únicos posibles resultados, y que estos son equiprobables: crear, para estos resultados, un código corrector de un error, de longitud media mínima. Indicar claramente por qué el código creado es instantáneo y su longitud media es mínima.
b) Una vez codificado el resultado obtenido la palabra de código es enviada hacia el centro de cálculo. Para comprobar el funcionamiento del canal binario usado se realiza la siguiente prueba: se toma un número suficientemente grande de entradas (las palabras del código creado en el apartado a) escogidas al azar, y se observa que a la salida 2 de cada 9 salidas contienen un error en uno cualquiera de sus dígitos, que 1 de cada 9 contiene dos errores, no observándose variación de mayor número de dígitos. En ambos casos, la probabilidad de que un 1 se convierta en 0, y a la inversa (0 en 1) es la misma. Calcular la entropía de la fuente de salida del canal.
c) El centro de cálculo, una vez recibida una combinación, procede a comprobar si ésta es correcta, aplicando el proceso de corrección oportuno en caso de combinación incorrecta. Viendo este proceso como un canal:
   I. ¿Qué tipo de canal será?
   II. Calcular la entropía de la fuente de salida del proceso de corrección.

14. Para transmitir la salida de la fuente de información de memoria nula $S=\{S_1,S_2,S_3,S_4\}$, por un canal de alfabeto de entrada $A=\{a_1,a_2,a_3,a_4\}$, se utiliza un código compacto.
a) Crear el código compacto identificando claramente los alfabetos fuente y código, e indicando por qué el código creado es compacto.
b) Supongamos que la línea de transmisión es unidireccional (el centro receptor no puede comunicarse con el emisor), proponer algún procedimiento que permita mejorar la fiabilidad en la transmisión.