## Código de Hamming

Hoy, el **código de Hamming** se refiere al (7.4) que **Hamming** introdujo en 1950. El código de **Hamming** agrega tres bits adicionales de comprobación por cada cuatro bits de datos del mensaje.

El algoritmo de **Hamming** (7.4) puede corregir cualquier error de un solo bit, pero cuando hay errores en más de un bit, la palabra transmitida se confunde con otra con error en un sólo bit, siendo corregida, pero de forma incorrecta, es decir que la palabra que se corrige es otra distinta a la original, y el mensaje final será incorrecto sin saberlo. Para poder detectar (aunque sin corregirlos) errores de dos bits, se debe añadir un bit más, y el código se llama **Hamming extendido**. No daremos en esta ocasión dicho procedimiento, sólo el algoritmo de **Hamming** (no extendido).

El [algoritmo](http://es.wikipedia.org/wiki/Algoritmo) es el siguiente:

1. Todos los bits cuya posición es potencia de dos se utilizan como bits de paridad (posiciones 1, 2, 4, 8, 16, 32, 64, etc.).

2. Los bits del resto de posiciones son utilizados como bits de datos (posiciones 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, 17, etc.).

3. Cada bit de paridad se obtiene calculando la paridad de alguno de los bits de datos. La posición del bit de paridad determina la secuencia de los bits que alternativamente comprueba y salta, a partir de éste, tal y como se explica a continuación.

* Posición 1: salta 0, comprueba 1, salta 1, comprueba 1, etc.
* Posición 2: salta 1, comprueba 2, salta 2, comprueba 2, etc.
* Posición 4: salta 3, comprueba 4, salta 4, comprueba 4, etc.
* Posición 8: salta 7, comprueba 8, salta 8, comprueba 8, etc.
* Posición 16: salta 15, comprueba 16, salta 16, comprueba 16, etc.
* Regla general para la posición **n** es: **salta n-1 bits, comprueba n bits, salta n bits, comprueba n bits**...
* Y así sucesivamente.

En otras palabras, el bit de paridad de la posición 2*k* comprueba los bits en las posiciones que tengan al bit **k** en su representación binaria. Dicho a la inversa, el bit 13, por ejemplo, es chequeado por los bits 8, 4 y 1, al ser estos los de su representación binaria: 13=1101(2); 8=1000(2); 4=0100(2); 1=0001(2).

Así, por ejemplo, para los primeros términos se tiene:

* En la Posición 1 (2^0 = 1), comprobaríamos los bits: 3, 5, 7, 9, 11, 13...
* En la Posición 2 (2^1 = 2), los bits: 3, 6, 7, 10, 11, 14, 15...
* En la Posición 4 (2^2 = 4), los bits: 5, 6, 7, 12, 13, 14, 15, 20, 21, 22, 23...
* En la Posición 8 (2^3 = 8) tendríamos: 9, 10, 11, 12, 13, 14, 15, 24-31...

Siguiendo el algoritmo hasta completar la nueva cadena.

Los bits verifican los siguientes campos:

|  |  |  |
| --- | --- | --- |
| Bits | **Verifica los siguientes bits** | **Noten que** |
| **1** | 1 3 5 7 9 11 13 15 17 19 21 | 1 si; 1 no |
| **2** | 2 3 6 7 10 11 14 15 18 19 | 2 si; 2 no |
| **4** | 4 5 6 7 12 13 1 4 15 20 21 | 4 si; 4 no |
| **8** | 8 9 10 11 12 13 14 15 | 8 si; 8 no |
| **16** | 16 17 18 19 20 21 22 23 24 | 16 si; 16 no |

## Ejemplo 1

Consideremos la palabra de datos de 7 bits "0110101". Para ver cómo se generan y utilizan los códigos Hamming para detectar un error, observe las tablas siguientes. Se utiliza la **d** para indicar los bits de datos y la **p** para los de [paridad](http://es.wikipedia.org/wiki/Bit_de_paridad).

En primer lugar los bits de datos se insertan en las posiciones apropiadas y los bits de paridad calculados en cada caso usando la paridad par.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** |
| **Palabra de datos (sin paridad):** |  |  | **0** |  | **1** | **1** | **0** |  | **1** | **0** | **1** |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 1 |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 1 |
| **Palabra de datos (con paridad):** | **1** | **0** | 0 | **0** | 1 | 1 | 0 | **0** | 1 | 0 | 1 |
| Cálculo de los bits de paridad en el código Hamming | | | | | | | | | | | |

La nueva palabra de datos (con los bits de paridad) es ahora "10001100101". Consideremos ahora que el bit de la derecha, por error, cambia de 1 a 0. La nueva palabra de datos será ahora "10001100100".

Sin errores

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** | **Prueba de paridad** | **Bit de paridad** |
| **Palabra de datos recibida:** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **0** | **1** | **1** |  |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 | **Correcto** | **0** |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 1 | **Correcto** | **0** |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  | Correcto | 0 |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 1 | **Correcto** | **0** |
| Comprobación de los bits de paridad (con primer bit de la derecha sin cambiar) | | | | | | | | | | | | | |

Con errores

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** | **Prueba de paridad** | **Bit de paridad** |
| **Palabra de datos recibida:** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **0** | **0** | **1** |  |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 0 | **Error** | **1** |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 0 | **Error** | **1** |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  | Correcto | 0 |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 0 | **Error** | **1** |
| Comprobación de los bits de paridad (con primer bit de la derecha cambiado) | | | | | | | | | | | | | |

Si se analiza en la tabla anterior la paridad que se debe obtener a la derecha tras la llegada del mensaje sin errores debe ser siempre 0 (por cada fila), pero en el momento en que ocurre un error esta paridad cambia a 1, de allí el nombre de la columna "prueba de paridad 1". Se observa que en la fila en que el cambio no afectó la paridad es cero y llega sin errores.

El paso final es evaluar los bits de paridad (recuerde que el fallo se encuentra en **d7**). El valor entero que representan los bits de paridad es 11 (si no hubieran ocurrido errores este valor seria 0), lo que significa que el bit décimo primero de la palabra de datos (bits de paridad incluidos) es el erróneo y necesita ser cambiado.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **p4** | **p3** | **p2** | **p1** |  |
| **Binario** | 1 | 0 | 1 | 1 |  |
| **Decimal** | **8** |  | **2** | **1** | **Σ = 11** |

Cambiando el bit décimo primero 1000110010**0** se obtiene de nuevo 1000110010**1**. Eliminando los bits de patrón de la paridad no se tienen en cuenta los bits de paridad. Si el error se produjera en uno de ellos, en la comprobación sólo se detectaría un error, justo el correspondiente al bit de paridad causante del mismo.

## Ejemplo 2

Tomemos la palabra en memoria 1100111000010111 y apliquemos el código de Hamming para paridad par.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nro.Bits | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | Cantidad |
| Hamming | H | H | 1 | H | 1 | 0 | 0 | H | 1 | 1 | 1 | 0 | 0 | 0 | 0 | H | 1 | 0 | 1 | 1 | 1 |  |
| **Para 16** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| **Para 8** |  |  |  |  |  |  |  | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |  |  |  |  |  |  | 3 |
| **Para 4** |  |  |  | 1 | 1 | 0 | 0 |  |  |  |  | 0 | 0 | 0 | 0 |  |  |  |  | 1 | 1 | 3 |
| **Para 2** |  | 0 | 1 |  |  | 0 | 0 |  |  | 1 | 1 |  |  | 0 | 0 |  |  | 0 | 1 |  |  | 4 |
| **Para 1** | 1 |  | 1 |  | 1 |  | 0 |  | 1 |  | 1 |  | 0 |  | 0 |  | 1 |  | 1 |  | 1 | 7 |
| **Final** | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 12 |

Si se produce un cambio en el bit 5 deberá ser tomado por el control de los bits 1 y 4 donde se verificará el cambio o error producido en la transmisión o por una variación de corriente eléctrica.

Tabla con error

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nro.Bits | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | Paridad | Bit | Cantidad |
| Hamming | H | H | 1 | H | 1 | 0 | 0 | H | **0** | 1 | 1 | 0 | 0 | 0 | 0 | H | 1 | 0 | 1 | 1 | 1 | 1 |  |  |
| **Para 16** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 0 | 1 | 0 | 1 | 1 | 1 | bien | 0 | 4 |
| **Para 8** |  |  |  |  |  |  |  | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |  |  |  |  |  |  | mal | 1 | 3 |
| **Para 4** |  |  |  | 1 | 1 | 0 | 0 |  |  |  |  | 0 | 0 | 0 | 0 |  |  |  |  | 1 | 1 | bien | 0 | 3 |
| **Para 2** |  | 0 | 1 |  |  | 0 | 0 |  |  | 1 | 1 |  |  | 0 | 0 |  |  | 0 | 1 |  |  | bien | 0 | 4 |
| **Para 1** | 1 |  | 1 |  | 1 |  | 0 |  | 0 |  | 1 |  | 0 |  | 0 |  | 1 |  | 1 |  | 1 | mal | 1 | 7 |
| **Final** | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | bien | 0 | 12 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **P5** | **p4** | **p3** | **p2** | **p1** |  |
| **Binario** | 0 | 1 | 0 | 0 | 1 |  |
| **Decimal** |  | **8** |  |  | **1** | **Σ = 9** |

No querramos controlar con los valores de 1 y 4 porque no es así como se debe hacer. Esto se debe hacer de la misma manera en que se construye cada bit. Iniciamos por el 16 y contamos la cantidad de unos que este bit controla y luego seguimos con el 8, luego el cuatro, el 2 y por último el 1. De la misma manera se hace con el error. Si da impar sabremos que ese control está mal.

Es importante indicar que el código de Hamming sólo puede detectar el cambio de 1 bit. Si fueran dos los que cambiaron sería casi imposible y si fueran 3 totalmente imposible más aún si se compensan.

Andrew S. Tanembaum[[1]](#footnote-1) desarrolla el siguiente ejercicio, que traducimos en un desarrollo igual que el nuestro para que pueda ser seguido detalladamente

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nro.Bits | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | Cantidad |
| Hamming | H | H | 1 | H | 1 | 1 | 1 | H | 0 | 0 | 0 | 0 | 1 | 0 | 1 | H | 0 | 1 | 1 | 1 | 0 |  |
| **Para 16** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 | 0 | 1 | 1 | 1 | 0 | 3 |
| **Para 8** |  |  |  |  |  |  |  | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |  |  |  |  |  |  | 2 |
| **Para 4** |  |  |  | 0 | 1 | 1 | 1 |  |  |  |  | 0 | 1 | 0 | 1 |  |  |  |  | 1 | 0 | 6 |
| **Para 2** |  | 0 | 1 |  |  | 1 | 1 |  |  | 0 | 0 |  |  | 0 | 1 |  |  | 1 | 1 |  |  | 6 |
| **Para 1** | 0 |  | 1 |  | 1 |  | 1 |  | 0 |  | 0 |  | 1 |  | 1 |  | 0 |  | 1 |  | 0 | 6 |
| **Final** | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 10 |

**Llega errado el bit 19**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nro.Bits | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | Paridad | Bit | Cantidad |
| Hamming | H | H | 1 | H | 1 | 1 | 1 | H | 0 | 0 | 0 | 0 | 1 | 0 | 1 | H | 0 | 1 | 0 | 1 | 0 | 1 |  |  |
| **Para 16** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 | 0 | 1 | 0 | 1 | 0 | mal | 1 | 3 |
| **Para 8** |  |  |  |  |  |  |  | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |  |  |  |  |  |  | bien | 0 | 2 |
| **Para 4** |  |  |  | 0 | 1 | 1 | 1 |  |  |  |  | 0 | 1 | 0 | 1 |  |  |  |  | 1 | 0 | bien | 0 | 6 |
| **Para 2** |  | 0 | 1 |  |  | 1 | 1 |  |  | 0 | 0 |  |  | 0 | 1 |  |  | 1 | 0 |  |  | mal | 1 | 6 |
| **Para 1** | 0 |  | 1 |  | 1 |  | 1 |  | 0 |  | 0 |  | 1 |  | 1 |  | 0 |  | 0 |  | 0 | mal | 1 | 6 |
| **Final** | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | bien | 0 | 10 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **P5** | **p4** | **p3** | **p2** | **p1** |  |
| **Binario** | 1 | 0 | 0 | 1 | 1 |  |
| **Decimal** | **16** |  |  | **2** | **1** | **Σ = 19** |

## Ejercicio 1:

Consideremos la palabra de datos de 7 bits "0110101". Muestre cómo se generan y utilizan los códigos Hamming para detectar un error (calculando los bits de paridad). Si se produce un error en el 6to bit de la cadena original (0110101) Explique detalladamente cómo encuentra el lugar del error.

## Solución:

Palabra 0110101

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** |
| **Palabra de datos (sin paridad):** |  |  | **0** |  | **1** | **1** | **0** |  | **1** | **0** | **1** |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 1 |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 1 |
| **Palabra de datos (con paridad):** | **1** | **0** | 0 | **0** | 1 | 1 | 0 | **0** | 1 | 0 | 1 |
| Cálculo de los bits de paridad en el código Hamming | | | | | | | | | | | |

La nueva palabra de datos (con los bits de paridad) es ahora "10001100101". Consideremos ahora que el segundo bit de la derecha, por error, cambia de 1 a 0. La nueva palabra de datos será ahora "10001100111".

Sin errores

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** | **Prueba de paridad** | **Bit de paridad** |
| **Palabra de datos recibida:** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **0** | **1** | **1** |  |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 | **Correcto** | **0** |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 1 | **Correcto** | **0** |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  | Correcto | 0 |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 1 | **Correcto** | **0** |
| Comprobación de los bits de paridad (con segundo bit de la derecha sin cambiar) | | | | | | | | | | | | | |

Con errores

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** | **Prueba de paridad** | **Bit de paridad** |
| **Palabra de datos recibida:** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **1** | **1** |  |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 | **Correcto** | **0** |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 1 | 1 | **Error** | **1** |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  | Correcto | 0 |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 1 | 1 | **Error** | **1** |
| Comprobación de los bits de paridad (con primer bit de la derecha cambiado) | | | | | | | | | | | | | |

Si se analiza en la tabla anterior la paridad que se debe obtener a la derecha tras la llegada del mensaje sin errores debe ser siempre 0 (por cada fila), pero en el momento en que ocurre un error esta paridad cambia a 1, de allí el nombre de la columna "prueba de paridad 1". Se observa que en la fila en que el cambio no afectó la paridad es cero y llega sin errores.

El paso final es evaluar los bits de paridad (recuerde que el fallo se encuentra en **d7**). El valor entero que representan los bits de paridad es 11 (si no hubieran ocurrido errores este valor seria 0), lo que significa que el bit décimo primero de la palabra de datos (bits de paridad incluidos) es el erróneo y necesita ser cambiado.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **p4** | **p3** | **p2** | **p1** |  |
| **Binario** | 1 | 0 | 1 | 0 |  |
| **Decimal** | **8** |  | **2** |  | **Σ = 10** |

## Ejercicio 2:

Consideremos la palabra de datos de 7 bits "0110101". Muestre cómo se generan y utilizan los códigos Hamming para detectar un error (calculando los bits de paridad). Si se produce un error en el 7mo bit de la cadena original (0110101) Explique detalladamente cómo encuentra el lugar del error.

## Solución:

Palabra 0110101

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** |
| **Palabra de datos (sin paridad):** |  |  | **0** |  | **1** | **1** | **0** |  | **1** | **0** | **1** |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 1 |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 1 |
| **Palabra de datos (con paridad):** | **1** | **0** | 0 | **0** | 1 | 1 | 0 | **0** | 1 | 0 | 1 |
| Cálculo de los bits de paridad en el código Hamming | | | | | | | | | | | |

La nueva palabra de datos (con los bits de paridad) es ahora "10001100101". Consideremos ahora que el bit de la derecha, por error, cambia de 1 a 0. La nueva palabra de datos será ahora "10001100100".

Sin errores

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** | **Prueba de paridad** | **Bit de paridad** |
| **Palabra de datos recibida:** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **0** | **1** | **1** |  |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 1 | **Correcto** | **0** |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 1 | **Correcto** | **0** |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  | Correcto | 0 |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 1 | **Correcto** | **0** |
| Comprobación de los bits de paridad (con primer bit de la derecha sin cambiar) | | | | | | | | | | | | | |

Con errores

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **p1** | **p2** | **d1** | **p3** | **d2** | **d3** | **d4** | **p4** | **d5** | **d6** | **d7** | **Prueba de paridad** | **Bit de paridad** |
| **Palabra de datos recibida:** | **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **0** | **0** | **1** |  |
| **p1** | **1** |  | 0 |  | 1 |  | 0 |  | 1 |  | 0 | **Error** | **1** |
| **p2** |  | **0** | 0 |  |  | 1 | 0 |  |  | 0 | 0 | **Error** | **1** |
| **p3** |  |  |  | **0** | 1 | 1 | 0 |  |  |  |  | Correcto | 0 |
| **p4** |  |  |  |  |  |  |  | **0** | 1 | 0 | 0 | **Error** | **1** |
| Comprobación de los bits de paridad (con primer bit de la derecha cambiado) | | | | | | | | | | | | | |

Si se analiza en la tabla anterior la paridad que se debe obtener a la derecha tras la llegada del mensaje sin errores debe ser siempre 0 (por cada fila), pero en el momento en que ocurre un error esta paridad cambia a 1, de allí el nombre de la columna "prueba de paridad 1". Se observa que en la fila en que el cambio no afectó la paridad es cero y llega sin errores.

El paso final es evaluar los bits de paridad (recuerde que el fallo se encuentra en **d7**). El valor entero que representan los bits de paridad es 11 (si no hubieran ocurrido errores este valor seria 0), lo que significa que el bit décimo primero de la palabra de datos (bits de paridad incluidos) es el erróneo y necesita ser cambiado.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **p4** | **p3** | **p2** | **p1** |  |
| **Binario** | 1 | 0 | 1 | 1 |  |
| **Decimal** | **8** |  | **2** | **1** | **Σ = 11** |

1. Andrew S. Tanembaum en su libro **“Organización de Computadoras un Enfoque Estructurado”** Editorial Prentice Hall 3ra. Edición páginas 50 a 54. Aconsejamos leer. [↑](#footnote-ref-1)