# Tema 2. Representación de la Información

## Objetivos

- Asimilar las características de los Sistemas de Numeración y Códigos de Representación empleados en los Sistemas Digitales.
- Comprender la metodología de conversión entre los diversos Sistemas y Códigos.
- Describir las características de los códigos detectores de errores.

# Tema 2. Representación de la Información

## Contenido

- Sistemas de numeración posicional.
  - Características.
  - Sistema binario.
  - Sistema octal.
  - Sistema hexadecimal.
- Conversión entre sistemas de numeración.
- Códigos binarios numéricos.
  - Binario natural y Binarios continuos y cíclicos (Gray)
  - BCD 8421.
- Códigos binarios alfanuméricos: ASCII.
- Códigos detectores de errores.
  - Concepto de distancia y distancia mínima.
  - Códigos de paridad.

# *Sistemas de Numeración Posicional*

- Un número se define mediante una cadena de dígitos, de forma que el valor de cada dígito depende de la posición que ocupe en la cadena, es decir tiene un peso.
- El peso se expresa en función de una potencia de un número concreto, que se denomina base (b).
- La base da nombre al sistema de numeración y define los valores que pueden tener los dígitos: 0 a b-1.
- Si consideramos un sistema de representación con n dígitos para parte entera y m para la fraccionaria, un número X se representa:

$$\mathbf{X = (a_{n-1}\ a_{n-2}\ \dots\ a_2\ a_1\ a_0\ ,\ a_{-1}\ a_{-2}\ \dots\ a_{-m})_b}$$

- Valor de los pesos:

$$\mathbf{P = b^{n-1}\ b^{n-2}\ \dots\ b^2\ b^1\ b^0\ ,\ b^{-1}\ b^{-2}\ \dots\ b^{-m}}$$

- Valor de los dígitos

$$\mathbf{0 \le a_i < b}$$

- Valor del número X

$$\mathbf{V(X) = a_{n-1}\ b^{n-1} + a_{n-2}\ b^{n-2} + \dots + a_1\ b^1 + a_0\ b^0 + a_{-1}\ b^{-1} + a_{-1}\ b^{-1} + \dots + a_{-m}\ b^{-m}}$$

# *Sistemas de Numeración Posicional*

- **Sistemas de numeración posicionales**

| Nombre | Base "b" | Dígitos utilizados |
|---|---|---|
| Binario | 2 | 0, 1 |
| Octal | 8 | 0, 1, 2, 3, 4, 5, 6, 7 |
| Decimal | 10 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 |
| Hexadecimal | 16 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F |

- **Tamaño de palabra**. En los sistemas digitales se trabaja con un nº finito de "n" dígitos llamado tamaño de palabra

# Conversión entre Sistemas de Numeración

- ***Conversión de binario, octal y hexadecimal a decimal.***
  - Se aplica la fórmula que determina el valor de un número.
  - Al operar en decimal se obtiene su equivalente decimal.
- ***Ejemplo de conversión de binario a decimal.***
  - $110010{,}011_2 = 1\cdot2^5 + 1\cdot2^4 + 1\cdot2^1 + 1\cdot2^{-2} + 1\cdot2^{-3}$
    $= 32 + 16 + 2 + 0{,}25 + 0{,}125 = 50{,}375$
  - **Conclusión.**
    - Se suman las potencias correspondientes a los bits con valor 1.
- ***Ejemplo de conversión de octal a decimal.***

$567{,}45_8 = 5\cdot8^2 + 6\cdot8^1 + 7\cdot8^0 + 4\cdot8^{-1} + 5\cdot8^{-2}$
$= 5\cdot64 + 6\cdot8 + 7\cdot1 + 4\cdot0{,}125 + 5\cdot0{,}015625$
$= 375{,}578125$

- ***Ejemplo de conversión de hexadecimal a decimal.***

$C8E{,}AB_{16} = C_{16}\cdot16^2 + 8_{16}\cdot16^1 + E_{16}\cdot16^0 + A_{16}\cdot16^{-1} + B_{16}\cdot16^{-2}$
$= 12\cdot256 + 8\cdot16 + 14\cdot1 + 10\cdot0{,}0625 + 11\cdot0{,}00390625$
$= 3214{,}66796875$

# Conversión entre Sistemas de Numeración

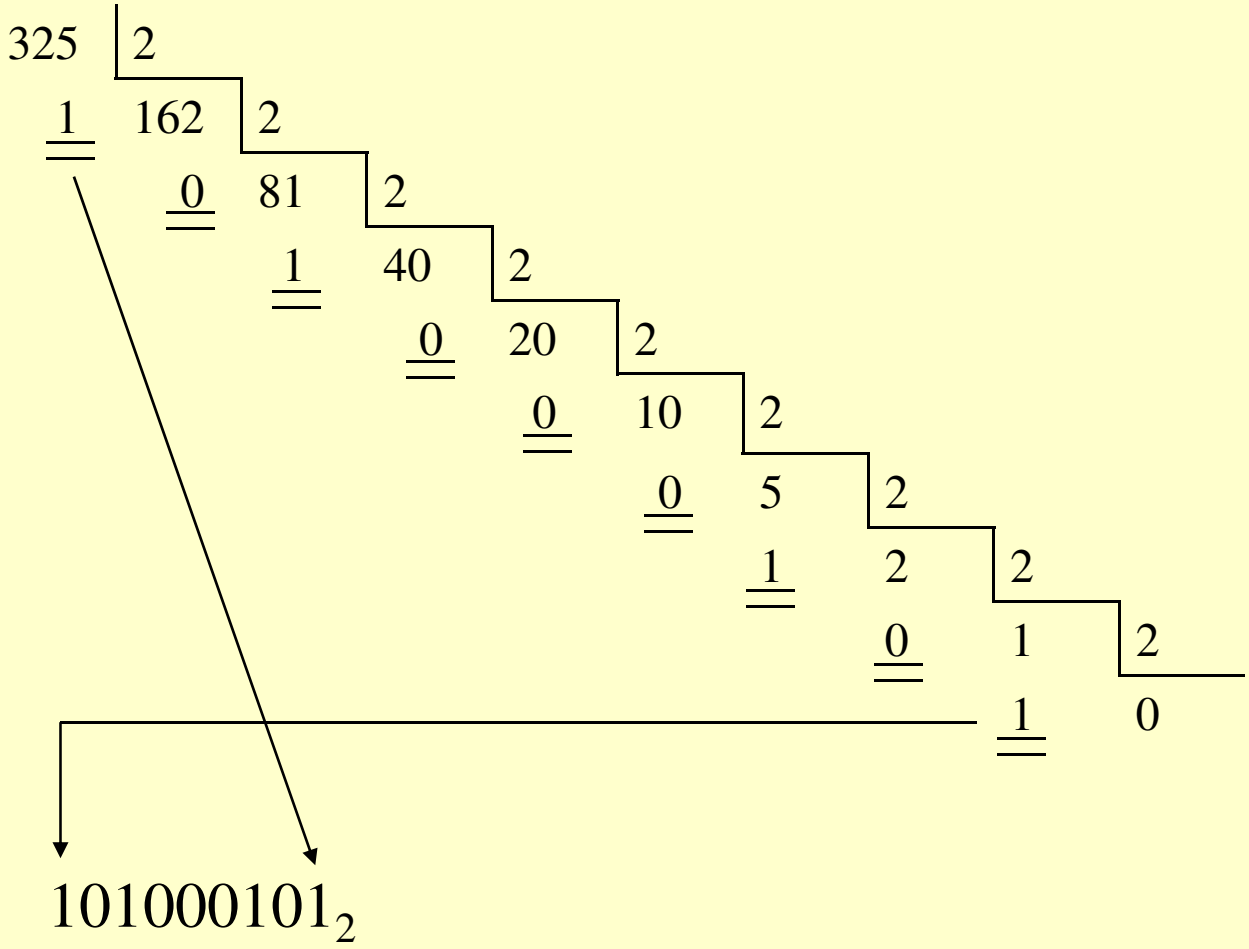
- Conversión de decimal a binario, octal y hexadecimal.
  - Se convierte por separado la parte entera y la fraccionaria.
  - Parte entera.
    - Divisiones sucesivas por la base a la que se convierte.
  - Parte fraccionaria.
    - Multiplicaciones sucesivas por la base a la que se convierte.

# Conversión entre Sistemas de Numeración

- Conversión por divisiones y multiplicaciones sucesivas.
  - Parte entera.
    - Se divide sucesivamente la parte entera del número decimal por la base hasta obtener un cociente igual a 0.
    - Los restos forman los dígitos en orden creciente de peso.
      - Es decir, el primer resto corresponde al dígito menos significativo y el último al más significativo.
  - Parte fraccionaria.
    - Se multiplica la parte fraccionaria del número decimal por la base. La parte entera obtenida corresponde al dígito más significativo.
    - Se vuelve a repetir el proceso con la parte fraccionaria hasta que sea 0, se repita o se de por buena la aproximación.
    - Se obtienen los dígitos en orden decreciente de peso.

# Conversión entre Sistemas de Numeración

- Ejemplo: 325,625



# *Conversión entre Sistemas de Numeración*

- Ejemplo: 325,625

$0{,}625 \times 2 = 1{,}25 \Rightarrow 1$ Bit más significativo

$0{,}25 \times 2 = 0{,}5 \Rightarrow 0$

$0{,}5 \times 2 = 1{,}0 \Rightarrow 1$ Bit menos significativo

$$325{,}625 = 101000101{,}101_2$$

# *Conversión entre Sistemas de Numeración*

- Conversión de decimal a octal. 478,58

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 478 | 8 | | 0,58 x 8 | = 4,64 | ⇒ | 4 | + signif. |
| 6 | 59 | 8 | 0,64 x 8 | = 5,12 | ⇒ | 5 | |
| | 3 | 7 8 | 0,12 x 8 | = 0,96 | ⇒ | 0 | |
| | | 7 0 | 0,96 x 8 | = 7,68 | ⇒ | 7 | - signif. |

$478{,}58 = 736{,}4507_8$

- Conversión de decimal a hexadecimal. 478,58

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 478 | 16 | | 0,58 x 16 | = 9,28 | ⇒ | 9 | + signif. |
| 14 | 29 | 16 | 0,28 x 16 | = 4,48 | ⇒ | 4 | |
| | 13 | 1 16 | 0,48 x 16 | = 7,68 | ⇒ | 7 | |
| | | 1 0 | 0,68 x 16 | = 10,88 | ⇒ | A | - signif. |

$478{,}58 = 1DE{,}947A_{16}$

# *Conversión entre Sistemas de Numeración*

- Conversión de binario a octal.
  - Como $8 = 2^3$, cada dígito octal consta de 3 bits.
    1. Se forman grupos de 3 bits por separado en la parte entera y fraccionaria, comenzando a partir de la coma.
    2. Si el número de bits de la parte fraccionaria no es múltiplo de 3 bits se completa el grupo de la derecha con los ceros necesarios.
    3. Se obtiene el dígito octal correspondiente a cada grupo de 3 bits.
  - Ejemplo.

$$1100101{,}1101_2 = \underline{1}\ \underline{100}\ \underline{101}{,}\underline{110}\ \underline{100}_2$$

$$1\quad 4\quad 5\ ,\ 6\quad 4_8$$

# *Conversión entre Sistemas de Numeración*

- Conversión de octal a binario.
  - Como $8 = 2^3$, cada dígito octal consta de 3 bits.
    - Se obtiene el equivalente de 3 bits de cada dígito octal.
  - Ejemplo.

$$3\ \ 2\ \ 7\ ,\ \ 6\ \ 2_8$$

$$11\ 010\ 111,110\ 010_2 = 11010111,11001_2$$

# *Conversión entre Sistemas de Numeración*

- Conversión de binario a hexadecimal.
  - Como $16 = 2^4$, cada dígito hexadecimal consta de 4 bits.
    1. Se forman grupos de 4 bits por separado en la parte entera y fraccionaria, comenzando a partir de la coma.
    2. Si el número de bits de la parte fraccionaria no es múltiplo de 4 bits se completa el grupo de la derecha con los ceros necesarios.
    3. Se obtiene el dígito hexadecimal correspondiente a cada grupo de 4 bits.
  - Ejemplo.

$$1101111{,}11011_2 = \underline{110}\ \underline{1111}{,}\underline{1101}\ \underline{1000}_2$$

$$6\quad F\ ,\ D\quad 8_{16}$$

# *Conversión entre Sistemas de Numeración*

- Conversión de hexadecimal a binario.
  - Como $16 = 2^4$, cada dígito hexadecimal consta de 4 bits.
    - Se obtiene el equivalente de 4 bits de cada dígito hexadecimal.
  - Ejemplo.

$$\begin{array}{cccccc} A & 4 & E & , & B & 4_8 \\ \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow \end{array}$$

$$1010\ 0100\ 1110,1011\ 0100_2 = 101001001110,101101_2$$

# *Conversión entre Sistemas de Numeración*

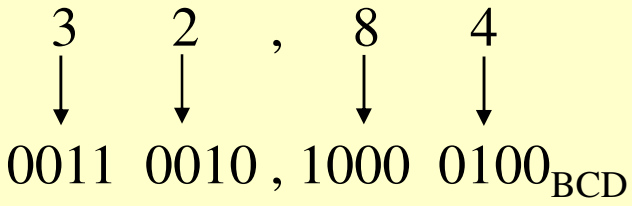
- Conversión de hexadecimal a octal.
  - Se convierte a través del sistema decimal o el binario.
  - Hexadecimal ⇒ Binario ⇒ Octal
    Ó
  - Hexadecimal ⇒ Decimal ⇒ Octal
- Conversión de octal a hexadecimal.
  - Se convierte a través del sistema decimal o el binario.
  - Octal ⇒ Binario ⇒ Hexadecimal
    Ó
  - Octal ⇒ Decimal ⇒ Hexadecimal

# Códigos Binarios

- ***Concepto de código***
  - ✓ Representación biunívoca de los números y caracteres alfanuméricos mediante una combinación de símbolos determinados.
  - ✓ Si los símbolos son los dígitos binarios, el código estará formado por combinaciones de bits o combinaciones binarias, y éste se denomina ***código binario***.
  - ✓ Las combinaciones binarias pertenecientes al código se denominan palabras.
  - ✓ ***Código binario natural***.

# BCD (Decimal Codificado Binario)

- Código binario ponderado de 4 bits con peso 8421.
- Representa los dígitos decimales (0 al 9) en su equivalente binario de 4 bits.
- Conversión de decimal a BCD 8421.
  - Se convierte cada dígito decimal a su equivalente BCD de 4 bits.




- Conversión de BCD 8421 a decimal
  1. Se forman grupos de 4 bits en la parte entera y en la fraccionaria a partir de la coma fraccionaria.
  2. Se obtiene el equivalente decimal de cada grupo de 4 bits.

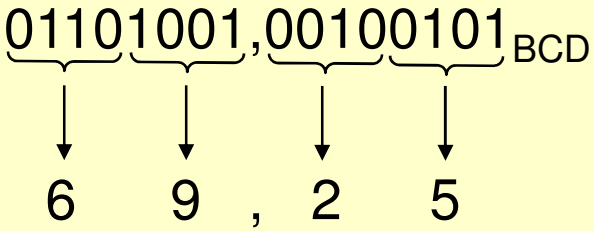



Tabla de conversión de decimal al código BCD

| Decimal | Código BCD 8421 |
|---|---|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |

# Códigos Binarios Numéricos sin Peso

## Códigos continuos y cícliclos

- ***Código continuo y cíclico.***
  - **Continuo.**
    - Un código binario en el que las combinaciones binarias (palabras) correspondientes a los números decimales consecutivos son adyacentes (sólo cambia un bit).
    - La representación del 0 y el 1 en BCD 8421son adyacentes, pero no la del 1 y el 2 ⇒ ***El código BCD 8421 no es un código continuo.***
      - 0000 y 0001 son adyacentes.
      - 0001 y 0010 no son adyacentes.
  - **Cíclico.**
    - Es un código binario continuo en el que la última combinación binaria es adyacente a la primera.
- **Uno de los códigos binarios continuos y cíclicos más común es el código Gray.**

# Códigos Binarios Numéricos sin Peso

## Código Gray

- Código continuo y cíclico sin peso.
- La capacidad de codificación es la misma que en el código binario natural. $C = 2^n$
- También se denomina código reflejado.
- **Obtención del código Gray**
  - El código Gray de *n* bits se forma a partir del de *n-1* bits haciendo una reflexión especular y añadiendo por la izquierda un bit con valor 0 en las $2^{n-1}$ primeras combinaciones y un 1 en las $2^{n-1}$ siguientes.
    - Cada combinación es **adyacente** (se diferencian en un solo bit) con la que le sigue y la precede. La primera y la última también son adyacentes.

| 1 bit | 2 bits | 3 bits |
|---|---|---|
| 0 | 00 | 000 |
| 1 | 01 | 001 |
| | 11 | 011 |
| | 10 | 010 |
| | | 110 |
| | | 111 |
| | | 101 |
| | | 100 |

# Códigos Binarios Numéricos sin Peso

## Código Johnson

- Código Johnson
  - Para un código de n bits la capacidad de codificación es de 2n números diferentes.
  - Código Johnson para representar los dígitos decimales.
    1. Primero se calcula el número de bits.

       Nº valores = 2 . n

       $10 = 2 . n \quad \Rightarrow n = 5$
    1. La combinación correspondiente al 0 es 00000.
    2. El resto de combinaciones se obtienen realizando la operación de rotación a la izquierda del bit situado más a la izquierda complementado.

| Decimal | Johnson |
|---|---|
| 0 | 00000 |
| 1 | 00001 |
| 2 | 00011 |
| 3 | 00111 |
| 4 | 01111 |
| 5 | 11111 |
| 6 | 11110 |
| 7 | 11100 |
| 8 | 11000 |
| 9 | 10000 |

# Códigos utilizando representación en exceso

## Código BCD en exceso 3

- **Sistema de representación en exceso a M.**
  - Primero se le suma M a los números y después se representan en binario natural.
    - $A(XSM) = A + M = (A + M)_2$
  - **Conversión de XSM a decimal**
    - Se pasa a decimal y se le resta M.
  - **Ejemplo:**
    - Código BCD exceso 3

| Decimal | BCD | $BCD_{EXC3}$ |
|---|---|---|
| 0 | 0000 | 0011 |
| 1 | 0001 | 0100 |
| 2 | 0010 | 0101 |
| 3 | 0011 | 0110 |
| 4 | 0100 | 0111 |
| 5 | 0101 | 1000 |
| 6 | 0110 | 1001 |
| 7 | 0111 | 1010 |
| 8 | 1000 | 1011 |
| 9 | 1001 | 1100 |

# Códigos Alfanuméricos

## Código ASCII

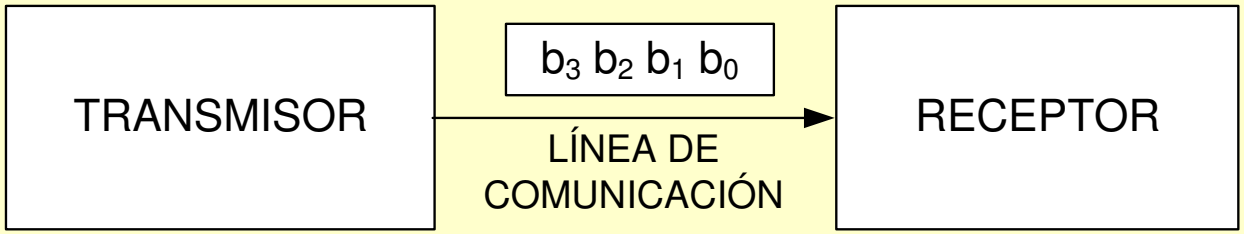
- En los computadores y otros sistemas digitales se procesa información no numérica: letras, símbolos de puntuación, caracteres de control, etc.
- Los códigos que representan esta información se denominan alfanuméricos.

| | $b_6b_5b_4$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $b_3b_2b_1b_0$ | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| 0000 | NUL | DLE | SP | 0 | @ | P | ‘ | p |
| 0001 | SOH | DC1 | ! | 1 | A | Q | a | q |
| 0010 | STX | DC2 | ” | 2 | B | R | b | r |
| 0011 | ETX | DC3 | # | 3 | C | S | c | s |
| 0100 | EOT | DC4 | $ | 4 | D | T | d | t |
| 0101 | ENQ | NAK | % | 5 | E | U | e | u |
| 0110 | ACK | SYN | & | 6 | F | V | f | v |
| 0111 | BEL | ETB | ’ | 7 | G | W | g | w |
| 1000 | BS | CAN | ( | 8 | H | X | h | x |
| 1001 | HT | EM | ) | 9 | I | Y | i | y |
| 1010 | LF | SUB | * | : | J | Z | j | z |
| 1011 | VT | ESC | + | ; | K | [ | k | { |
| 1100 | FF | FS | , | < | L | \ | l | \| |
| 1101 | CR | GS | – | = | M | ] | m | } |
| 1110 | SO | RS | . | > | N | ^ | n | ~ |
| 1111 | SI | US | / | ? | O | _ | o | DEL |

# *Códigos detectores y correctores de errores.*

## *Fundamentos.*

- **Justificación.**
  - Los sistemas digitales intercambian información entre sí mediante alguno de los códigos analizados: BCD, ASCII, etc.
  - El sistema que envía la información se denomina transmisor.
  - El sistema que recibe la información se denomina receptor.
  - Generalmente los sistemas pueden ser tanto transmisores como receptores.
  - Se puede producir errores en la comunicación debido a las interferencias eléctricas o por avería de alguno de los componentes.
  - Se han desarrollados códigos que permitan detectar si la información recibida es errónea e incluso otros que pueden corregir el error.



# Códigos detectores de errores.

## Fundamentos.

- Requisitos para que un código pueda detectar errores.
  - Un código se dice que es un código de detección de errores si tiene la propiedad de detectar combinaciones binarias que no son del código.
  - La condición necesaria y suficiente para que un código pueda detectar errores viene dada por su distancia mínima.
  - Concepto de distancia.
    - La distancia entre dos combinaciones binarias es el número de bits que hay que cambiar en una para obtener la otra.

      D(0000, 0001) = 1  D(0000, 1111) = 4
- Concepto de distancia mínima (Dm).
  - La menor de las distancias de todas las posibles parejas de palabras de ese código.
  - Para que un código pueda detectar errores en un bit su Dm debe ser mayor que 1.
    - Ejemplo: Sea el código de tres bits cuyas palabras son 000, 011, 100, 110. Los caracteres 001, 010, 111, 101 no pertenecen al código. Si transmitimos la palabra 100 podríamos recibir, si se produce cambios en un bit (error en un bit), las palabras 000 o 110 que si pertenecen al código, no pudiéndose detectar el error. La Dm en este caso es 1.

# Códigos detectores y correctores de errores.

## Códigos de paridad

- ***Códigos de paridad***
  - Se forman añadiendo a los códigos de Dm unidad un bit denominado de ***paridad***.
  - Hay dos tipos de paridad:
    - ***Par.*** El bit de paridad toma el valor necesario para que el número de bits con valor 1 en la palabra del nuevo código sea par. El número 0 se considera par.
    - ***Impar.*** El bit de paridad toma el valor necesario para que el número de bits con valor 1 en la palabra del nuevo código sea impar.
  - Código de paridad a partir del código binario natural de 3 bits.
    - **Paridad par**
      - Formato

| $b_2$ $b_1$ $b_0$ | P |
|---|---|

      - P es 1 si el número de 1 en los bits de datos es impar.
      - P es 0 si el número de 1 en los bits de datos es par.
    - **Paridad impar**
      - Formato

| $b_2$ $b_1$ $b_0$ | I |
|---|---|

      - I es 1 si el número de 1 en los bits de datos es par.
      - I es 0 si el número de 1 en los bits de datos es impar.

# Códigos detectores de errores

## Códigos de paridad

- Código de paridad a partir del código binario natural de 3 bits.

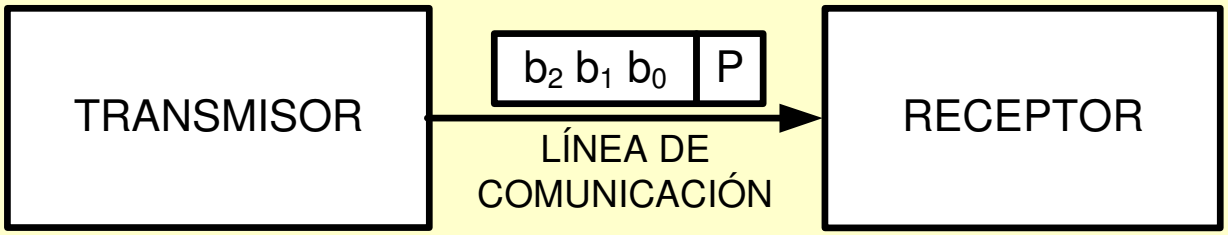
| Binario 3 bits | | | Paridad Par | | | | Paridad Impar | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $b_2$ | $b_1$ | $b_0$ | $b_2$ | $b_1$ | $b_0$ | P | $b_2$ | $b_1$ | $b_0$ | I |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

- La Dm del nuevo código es 2.
  - El nº de palabras del código es igual al de las que no son del código.

# Códigos detectores de errores

## Códigos de paridad

- Comprobación de paridad.
  - El transmisor y el receptor usan el mismo código de paridad.
  - Supongamos el código binario de 3 bits con paridad par.




  - El receptor comprueba la paridad volviendo a generar el bit de paridad par de la palabra recibida ($b_2$, $b_1$, $b_0$, P).
    - ✓ Si el bit de paridad es 0 la palabra es correcta.
    - ✓ Si el bit de paridad es 1 la palabra es incorrecta.
  - El transmisor envía el 2 (0101).
    - ✓ El receptor lo recibe correctamente y genera el bit de paridad par.
      - ✓ PP(0101) = 0 Correcto
    - ✓ Se produce un error en $b_2$ y el receptor recibe la palabra 1101.
      - ✓ PP(1101) = 1 Incorrecto
  - Un código de paridad puede detectar cambios en un nº impar de bits, pero no detecta cambios que impliquen un nº par de bits.

# Bibliografía detallada
## Representación de la Información

- **Las diapositivas se han confeccionado utilizando como fuentes:**
  - Para la conversión entre sistemas:
    - "Introducción al diseño lógico", Hayes
      Apartados: 2.1, 2.2, 2.3, 2.4, 2.8
    - "Sistemas electrónicos digitales", Mandado.
      Apartados: 1.1, 1.2, 1.3, 1.4.
  - Para los códigos binario y BCD
    - "Sistemas electrónicos digitales", Mandado.
      Apartado: 1.5.
    - "Diseño Lógico". A. Lloris, A. Prieto. Mc-Graw Hill. 1996.
      Apartados. 4.3.2, 4.3.4.
  - Para los códigos alfanuméricos:
    - "Introducción al diseño lógico", Hayes.
      Apartado: 2.10
    - "Diseño Lógico". A. Lloris, A. Prieto. Mc-Graw Hill. 1996.
      Apartados: 4.3.0, 4.3.1
  - Para los códigos detectores de errores:
    - "Sistemas electrónicos digitales", Mandado
      Apartados: 1.7. 1.8