TEMA 1: Representación y comunicación de la información

Índice de contenidos

1.- Introducción

El ordenador digital es una máquina capaz de procesar datos para resolver un problema determinado. Para llevar a cabo esta tarea, existen dos elementos fundamentales:

- **Datos** → sobre los que realiza operaciones
- **Programas** → compuestos por un conjunto de instrucciones, que recibirán unos datos de entrada y producirán otros datos de salida.

Para ello, un ordenador digital debe ser capaz de manejar información que está representada mediante dos símbolos (0 y 1), que corresponden a dos estados electrónicos. Esto facilita la construcción y fiabilidad de los circuitos internos del ordenador, que es capaz de manejar:

- Carácteres numéricos: 0, 1, 2, 3, 4, 5, 6,, 7, 8, 9
- Carácteres alfabéticos: A,B..Z, a,b..z
- Carácteres especiales: ( ) $%& …
- Carácteres gráficos
- Información de control: saltos de línea, control de comunicaciones, control de escape, …

Y para representar toda esta variedad de carácteres y de información de control que puede manejar un ordenador, es necesario disponer de un sistema de codificación y decodificación.

Desde hace ya muchos años y en nuestras actividades cotidianas, llevamos a cabo expresión de información, almacenamiento de información, que a su vez manejamos de la forma más conveniente para resolver los problemas que se presenten. Para tal fin, empleamos el sistema DECIMAL para representar números, y para representar el idioma en el que nos comunicamos, el ALFANUMÉRICO. Desde el punto de vista del ordenador, se necesita transformar la información representada en estos sistemas a un sistema fundamentado en elementos digitales y electrónicos: el paso o no paso de corriente eléctrica, para lo que el sistema binario encaja perfectamente.

2.- La información y su representación

Un sistema de numeración es un conjunto de reglas, convenios y símbolos combinados con palabras que nos permiten expresar verbal y gráficamente los números. Estos pueden ser:

- **Posicionales**: se contempla el valor relativo a la cifra dentro del número. Ejemplo sería el sistema de numeración DECIMAL.
- **No posicionales**: independientemente de la posición, la cifra siempre tiene el mismo valor. Por ejemplo el sistema de numeración ROMANA.

Tomaremos como ejemplo el sistema decimal o de base 10, en el que se emplean diez símbolos (0..9). El conjunto de símbolos está ordenado y el valor del símbolo de cada posición se multiplica por la base elevada a la posición del símbolo, empezando por cero, de derecha a izquierda.

Codificación: es la transformación o correspondencia unívoca de los elementos de un conjunto en los elementos de otro conjunto distinto, siguiendo un método determinado, de forma que se pueda efectuar el proceso inverso, denominado **decodificación**. Según el número de elementos que necesitemos codificar, necesitaremos más o menos bits, sabiendo que un bit es la unidad mínima de información que se puede representar en el sistema binario, el que emplea el ordenador.

Ya que los ordenadores funcionan con impulsos eléctricos, solo son capaces de interpretar si un elemento está cargado o no de electricidad (elemento biestable). A los efectos prácticos, este se representa mediante 1 (cargado) y 0 (sin carga). Por lo tanto, toda la información que se maneja o almacena en un ordenador está representada mediante bits, utilizando para ello el sistema binario. Y podemos definir un **bit** (Binary digIT) como la unidad mínima de información que es capaz de representar un ordenador: 0 o 1; siendo a su vez un **byte** la agrupación de 8 bits la unidad mínima inteligible. Definimos como una **palabra** a la unidad mínima que maneja el microprocesador: 8, 16, 32 o 64 bits (longitud de palabra).

2.1.- Sistemas numéricos

Un sistema numérico o sistema de numeración es el conjunto de símbolos utilizados para la representación de cantidades, así como las reglas que rigen dicha representación. Se distinguen por su **base**, que es el número de símbolos que se emplean y que determina el valor de cada símbolo según la posición que ocupe. Para indicar en que base está representado un número se emplea la **nomenclatura: número)$_{base}$** , como por ejemplo $101)_2$

- **Teorema Fundamental de la Numeración**:

Cualquier cantidad expresada en cualquier sistema numérico puede representarse en sistema decimal mediante la siguiente fórmula:

$$N = \sum_{i=-d}^{n} X_i x B^i$$

Donde:
N = valor decimal
X = valor decimal del símbolo o dígito
B = valor de la base
d = número de dígitos a la derecha de la coma
n= número de dígitos a la izquierda de la coma -1

Así, para obtener el valor decimal del número $201)_3$ (expresado en base 3) desarrollamos la fórmula como sigue:

$$N^0 = (s_2 \times b^2) + (s_1 \times b^1) + (s_0 \times b^0)$$
$$N^0 = (2 \times 3^2) + (0 \times 3^1) + (1 \times 3^0)$$
$$N^0 = 18 + 0 + 1 = 19$$

ç

Otro método que podemos emplear, esta vez para el cambio a base 10, es el **método Ruffini.** Siguiendo el ejemplo en el que pasamos el número $1234)_5$ a sistema decimal (base 10):

	1	2	3	4
5		5	35	190
	1	7	38	194

Este método también se utiliza para otros fines en matemáticas, como puede ser el cálculo con polinomios.

2.1.1.- Decimal

El sistema decimal, es el usado normalmente por el ser humano. Se emplea la **base 10** y utiliza los símbolos {0,1,2,3,4,5,6,7,8,9}. Normalmente, no se suele aplicar el teorema de la numeración al sistema decimal ya que el resultado es el propio número.

Los múltiplos y submúltiplos se representan como potencias de 10 ($1000=10^3=1K$; $0.1=10^{-1}$)

2.1.2.- Binario

El sistema binario es el que utiliza el hardware del ordenador ya que solo usa dos símbolos {0,1} para la representación de cualquier número. Por tanto su **base es 2.** A cada símbolo usado en este sistema se le denomina bit.

El factor para los múltiplos en binario es 1024 (2^{10}). Así que un kilobit serán 1024 bits y un kilobyte (KB) 1024 bytes.

Aplicando el Teorema Fundamental de la Numeración (TFN), el número $1001)_2$, será:

$$N^0 = (1x2^3)+(0x2^2)+(0x2^1)+(1x2^0)$$
$$N^0=8+0+0+1=9_{)10}$$

¿Cuántos bits necesitamos para codificar cierto número de símbolos?

> $2^n>=m$
> $n>=\log_2 m$
> siendo **m** el número de símbolos a codificar y n el número de bits que necesitaremos para codificarlos. Siendo **n** número entero.

2.1.3.- Octal

Emplea los símbolos {0,1,2,3,4,5,6,7}, siendo, por tanto, **base 8**.

De este modo, el número $1001)_8$, en decimal sería:

$$N^0 = (1x8^3)+(0x8^2)+(0x8^1)+(1x8^0)$$
$$N^0=512+0+0+1=513_{)10}$$

2.1.4- Hexadecimal

El sistema hexadecimal o de base 16, emplea los siguientes símbolos {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}. Debido a que la conversión entre binario-hexadecimal es directa, se emplea bastante en el software informático, al igual que el sistema octal. Cada símbolo tiene un valor asociado en base 10, como se muestra en la siguiente tabla:

Decimal	Binario	Hexadecimal	Octal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

Tabla 1: de correspondencia entre los sistemas decimal-binario,hexadecimal, octal

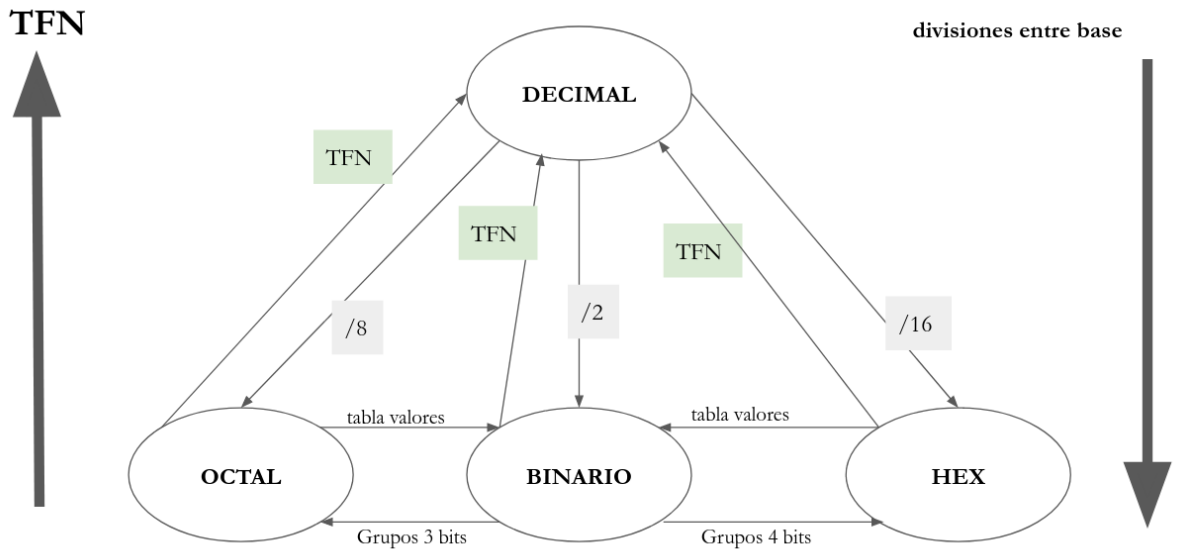
Así, como en los sistemas anteriores y aplicando el TFN, el número $2CA)_{16}$, en decimal sería:

$$N^0 = (2x16^2)+(Cx16^1)+(Ax16^0)$$
$$N^0 = (2x16^2)+(12x16^1)+(10x16^0)$$
$$N^0=512+192+10=714_{)10}$$

2.2.- Conversión entre sistemas numéricos

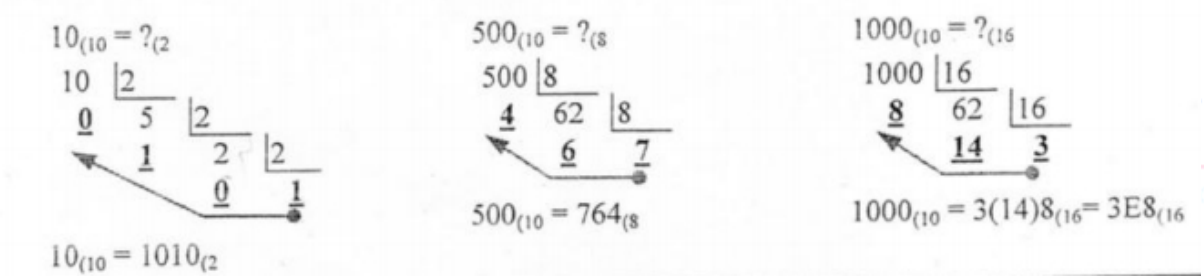
Suele ser necesario hacer cambios de base, de forma que podamos ser capaces de convertir la representación de un número en un sistema numérico a otro sistema numérico diferente, con la finalidad de un mejor entendimiento sobre el funcionamiento interno del ordenador de como este representa la información.

Para ello presentaremos las conversiones entre los sistemas binario - octal - hexadecimal y decimal, viendo todos los caminos posibles para transformar un número de una base a otra.



Esquema de transformación entre distintos sistemas numéricos

- **Cualquier sistema a decimal**: se aplica el TFN.
- **Decimal a otro sistema**: se realiza mediante divisiones sucesivas usando como divisor la base a la que queremos convertir el número. Se toma para la siguiente división solo la parte entera del cociente y se van guardando los restos obtenidos. Cuando el último cociente es menor que la base (ya no se puede seguir dividiendo), se toma este como primer dígito y a continuación se van situando los restos obtenidos en orden inverso.



Ejemplo de transformación de sistema decimal a binario, octal y hexadecimal

- **Decimal con decimales a binario**: si para la parte entera se han empleado divisiones sucesivas, para la parte fraccional se harán multiplicaciones por 2, hasta obtener cero decimales o tengamos el número de decimales deseado. Se repite el proceso con el resultado de la operación anterior despreciando la parte entera. Se toman las partes

enteras (que siempre serán 0 o 1) en el orden obtenido (desde la coma hacia fuera):

$$0.828125_{(10} = ?_{(2}$$
$$0.828125 \times 2 = 1.65625$$
$$0.65625 \times 2 = 1.3125$$
$$0.3125 \times 2 = 0.625$$
$$0.625 \times 2 = 1.25$$
$$0.25 \times 2 = 0.5$$
$$0.5 \times 2 = 1.0$$
$$0.828125_{(10} = 0.110101_{(2}$$

- **Hexadecimal a binario y viceversa:**

  Hexadecimal → binario: se sustituye cada dígito por el cuarteto correspondiente de dígitos binarios a su valor absoluto. Hay que respetar los ceros a la izquierda.

  Binario → Hexadecimal: se divide el número en cuartetos de bits, empezando por la derecha, sustituyendo cada cuarteto por su correspondiente símbolo equivalente en hexadecimal.

  (Véase tabla 1)

2.3.- Representación interna de la información

El hombre ha venido usando un sistema de códigos, bien sean números (normalmente en decimal) o palabras (compuestas por caracteres: A-Z, a-z, 0-9, +,-, … etc) y el ordenador utiliza los bits para representar la información. Para la conversión entre dichos sistemas se impone el uso de las llamadas **codificaciones**. Así, para representar los números, emplea una serie de codificaciones mientras que para las letras usa otra distinta. A partir de aquí, entendemos por número una cifra que va a ser empleada para cálculos matemáticos, habiendo otros tipos de datos que sean contemplados como caracteres al no realizarse cálculos matemáticos sobre ellos (número de póliza, número de teléfono, etc).

2.3.1.- Números enteros

Se emplea el binario puro con la excepción de que se añade un dígito a la izquierda para representar el signo (0 para positivos y 1 para negativos). El resto de los bits componen el llamado módulo (valor absoluto). Este método se denomina **módulo y signo** (MS). Usando un byte para representar un número entero, se puede representar cualquier número comprendido entre 127 y -127. Si se emplean 2 bytes, el rango de representación estaría entre 32767 y -32767.

Con el método de **complemento a 1**, también se reserva el bit de la izquierda para el signo. En cambio, en el módulo se representan los números positivos en su binario puro y los

negativos complementados a 1 (se cambian los 0 por 1 y viceversa, incluidos 0 a la izquierda). Los rangos de representación son los mismos que en MS.

En **complemento a 2**, es como en complemento a 1, pero obteniendo el módulo de los negativos en dos pasos:

1º) se obtiene el complemento a 1

2º) se le suma 1 al número.

Si hay acarreo en el último dígito, se desprecia. La diferencia fundamental con los otros métodos es que el rango de representación es asimétrico, de forma que para un byte estaría entre 127 y -128. Para dos bytes entre 32767 y -32768.

2.3.2.- Números reales

Al codificar números hay que tener en cuenta que el ordenador usa un número finito de bits para ello. De esta forma, los números reales sufren un truncamiento en su parte decimal. Ni siquiera el método de coma flotante nos ofrece garantías absolutas de precisión. Excepto en cálculos aritméticos de alta precisión, este truncamiento no suele suponer problemas esenciales para el normal funcionamiento de un ordenador.

- **Representación en coma fija.**

Al representar un número real, empleamos un conjunto de bits para la parte entera, un carácter adicional (la coma) para separarla de la parte fraccionaria o real, igual que haríamos al trabajar en base decimal. De esta forma tendremos un número fijo de bits para la parte entera y un número de bits para la parte fraccionaria, determinando este último la precisión capaz de obtener.

$$101{,}1101_{)2} = 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 + 1\cdot2^{-1} + 1\cdot2^{-2} + 0\cdot2^{-3} + 1\cdot2^{-4} =$$
$$= 4 + 1 + 0{,}5 + 0{,}25 + 0{,}0625 = 5{,}8125_{)10}$$

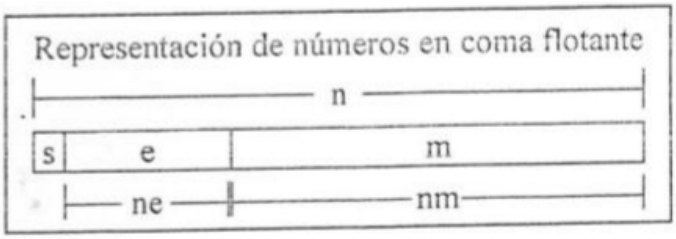
Como vemos en el ejemplo anterior, la máxima precisión que se consigue es de 0,0625. De esta forma la precisión viene dada por el número de bits que maneje la parte fraccionaria, quedando fuera de alcance los números que no se puedan representar por requerir de más dígitos en la parte fraccionaria. En cuanto a la ventaja, debemos resaltar la simplicidad de las operaciones ya que se pueden emplear los mismos circuitos hardware que realizan las operaciones con la parte entera.

- **Representación en coma flotante.**

Cuando se trata de representar números muy grandes se suele usar la notación exponencial. Esto sería, tomando de partida el número 13257.3285 = $13257.3285 \cdot 10^0 = 1.32573285 \cdot 10^4$. Es decir, $\mathbf{N = M \cdot B^E}$, donde B es la base, M la mantisa y E el exponente. La notación exponencial también se denomina coma flotante ya que el punto (o la coma) decimal "flota" tantas posiciones hacia la izquierda como indica el exponente (o hacia la derecha si E es

negativo).

Para la representación de números en coma flotante en el ordenador se usa el estándar IEEE 754[1] (desarrollado entre 1977 y 1985) en el que se usa como base (B) el 2 y que al estar predeterminado, no es necesario almacenar. Así solo hay que representar el signo del número(s), el exponente (E) y la mantisa (M), donde:



n = número de bits usados en la representación del número
s = bit de signo: 0 para positivos y 1 para negativos
ne = número de bits que se usan para representar el exponente
nm = número de bits usados para representar la mantisa

e = campo de exponente: almacena el exponente en formato de entero sesgado, es decir, **al exponente se le suma un valor constante llamado sesgo** (S). De esta forma podemos incluir exponentes positivos o negativos sin tener que reservar un bit extra para su signo. El valor del sesgo depende del tamaño de ne y se obtiene con la fórmula $\mathbf{S=2^{ne-1}-1}$. Así por ejemplo, para ne=8, $S=2^{8-1}-1 = 127$.

m = campo de la mantisa: la mantisa se representa mediante la notación exponencial normalizada, es decir, en la posición de las unidades quedará representado el primer 1 significativo del número. En la mantisa normalizada se cumple que 1<= M <=2.

Todos los valores estarían representados mediante unos y ceros, es decir en binario. Hay que tener en cuenta que la mantisa solo almacena la parte fraccionaria (a la derecha del punto decimal) de M normalizada. Con esto se consigue ahorrar espacio (se dice que el 1. está implícito o que el número está empaquetado). De forma que al realizar cualquier operación con el número almacenado mediante esta representación, la ALU debe restituir el 1. (desempaquetando el dato). Luego **M=1.m**, siendo m la parte que se almacena.

Ejemplo: usando 8 bits para ne y 23 para nm, la representación quedaría:

Número	Notación exponencial	Repr coma flotante (32 bits)
0,0100 1110 0110	$1{,}0011\ 1001\ 10 \cdot 2^{-2}$	0 **01111101** 00111001100000000000000
-1101 101	$-1{,}101101 \cdot 2^{6}$	1 **10000101** 10110100000000000000000

1 Institute of Electrical and Electronics Engineers, instituto que fomenta el desarrollo de estándares. De los relacionados con la informática se ocupa la IEEE Computer Society.

El estándar IEEE 754 considera dos tamaños o precisiones posibles de datos:

- **Simple precisión**: usa 32 bits para la representación de los que 8 son para ne y 23 para nm. El sesgo S usado es 127 por lo que el mayor exponente que puede representar es 127 y el menor -126 (-126<=E<=127).
- **Doble precisión**: usa 64 bits para la representación, de los que 11 son para ne y 52 para nm. El sesgo S usado es 1023 (-1022<=E<=1023).

2.3.3.- Datos alfanuméricos

Para esta finalidad se utilizan los llamados códigos externos o códigos de Entrada/Salida. Estos códigos se emplean para traducir la información que introduce el usuario a una forma inteligible por el ordenador, pasando previamente por una traducción binaria. Algunos de los códigos externos empleados son:

- **BCD** (*Binary Coded Decimal*, 4 bits) → cada dígito decimal con 4 bits
- **EBCDIC** (*Extended Binary Coded Decimal Interchange Code* - 8 bits). Contempla un total de 256 caracteres y venía siendo usado en ordenadores mainframe de IBM.
- **ASCII** (*American Standard Code for Information Exchange*), inicialmente representado mediante 7 bits. ASCII extendido emplea 8 bits para incluir letras acentuadas, la ñ, caracteres semigráficos y algunos símbolos más.
- **UNICODE**. Hoy en día empleado para codificar símbolos utilizados en los distintos idiomas alrededor del mundo, existiendo diferentes formas de llevar a cabo esta codificación: **UTF**-8 (8-bit Unicode Transformation Format) siendo de longitud variable, pudiendo emplear de 1 a 4 bytes (UTF-8, 16 y 32).

3.- Comunicación de la información

3.1.- Elementos de un sistema de comunicación

Un sistema de comunicación es aquel que transmite información desde un lugar (emisor o transmisor) a otro lugar (receptor). La información transmitida es conocida como mensaje. El **emisor** transforma el mensaje original en señal apropiada según el **canal**, por medio de un transductor y la adecúa y amplifica obteniendo otra señal apta para ser eficientemente emitida a través del canal y captada por el **receptor**. En el receptor se realiza el proceso inverso: la señal transmitida se convierte en la señal-mensaje de forma que esta sea inteligible para el receptor final.

Los sistemas electrónicos en los que nos vamos a centrar son los que tanto el emisor como el receptor son equipos electrónicos. La señal transmitida puede ser de naturaleza eléctrica, electromagnética u óptica.

En los sistemas de comunicación para transmisión de datos en el emisor y receptor suelen distinguirse dos equipos:

- Equipo terminal de datos o DTE, que son los elementos origen y destino finales de la

información.
- Equipo terminal de la línea de comunicaciones o DCE, que son el emisor receptor, encargado específicamente de adaptar la señal o mensaje para transmitirlo o recibirlo convenientemente.

Los equipos terminales de datos también se denomina estaciones o sistemas huésped o, sencillamente, terminales.

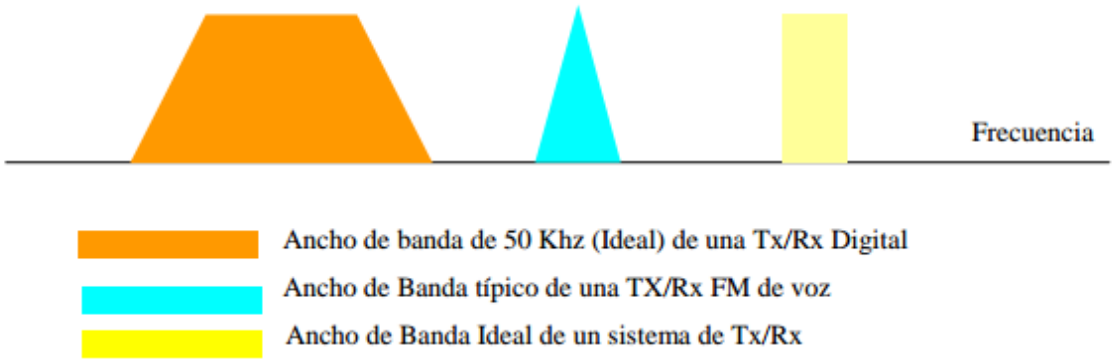
El **canal o medio de transmisión** puede ser guiados y no guiados. Dichos canales se pueden componer de:
- Hilos o cables eléctricos, para transmisión de señales de tensión o corriente.
- Fibras ópticas para transmisión de señales electromagnéticas en el rango o guías ondas, para transmisión de señales ópticas.
- La atmósfera y espacio libre para transmisión de señales electromagnéticas.

3.2.- Características de la comunicación

Las señales transmitidas en el medio **pueden ser alteradas** indeseablemente por 2 motivos:
- **Atenuación**. En efecto, las señales con la distancia se debilitan y deforman, siendo muchas veces necesario incluir en el canal, a ciertos intervalos de distancia, circuitos repetidores que reconstruyan y amplifiquen la señal original.
- **Ruido** que interfiere con la señal transmitida, pudiendo incluso imposibilitar al receptor recuperar el mensaje original.

Desde un punto de vista técnico, existen dos aspectos relevantes en un sistema de comunicación son:
- **Fiabilidad** en la transmisión del mensaje. Es decir, el sistema debe proporcionar en el elemento destino el mensaje sin errores, tal como se introdujo en el emisor.
- **Velocidad de transferencia** de información. Cuanto más mejor. En cualquier caso, ésta debe adecuarse a la naturaleza de los medios y recursos empleados. Así, la velocidad de transferencia de información entre dos ordenadores o sistemas debe ser superior a la que se establezca entre un ordenador y una impresora, por ejemplo.

Entre los parámetros que distinguen a distintos medios de transmisión se encuentran la **velocidad de transferencia** (en Mbps - Mega bits por segundo), el ancho de banda en el que debe moverse la señal (en MHz) y distancia entre repetidores



Representación de anchos de banda

Cuando se transmite a grandes distancias es necesario efectuar una transformación en el mensaje denominada **modulación**. La modulación permite transmitir a grandes distancias con atenuación muy baja y utilizar el mismo canal para transferir varios mensajes simultáneamente, mediante un proceso electrónico denominado multiplexación.

Mediante la multiplexación, las señales se propagan "mezcladas" y pueden separarse en los receptores por medio de una de-multiplexación además de una de-modulación, recuperando la señal original. La modulación consiste en variar linealmente uno de los parámetros de otra señal denominada portadora, con la señal o mensaje a transmitir. La portadora puede ser una señal sinusoidal (modulación analógica) o un tren de pulsos (modulación digital). De igual forma el mensaje puede ser analógico (voz, sonidos) o digital (datos a transmitir entre ordenadores o sistemas informáticos). Las modulaciones analógicas pueden ser en amplitud, frecuencia o fase y la modulación digital por anchura de pulso, por posición de pulso o modulación codificada por pulsos.

Las distintas variantes de transmisión son: transmisión analógica de una señal analógica (radio, televisión), transmisión analógica de una señal digital (se utiliza el módem), transmisión digital de una señal digital y transmisión digital de una señal analógica (son las más fiables y por ello las redes de comunicaciones modernas lo utilizan para transmitir señales de voz, datos e imágenes).

En cuanto a la multiplexación existen dos modalidades: multiplexación en frecuencia y en el tiempo.

La comunicación se puede establecer: **simplex, semiduplex (o half-duplex) y duplex**

En una comunicación **simplex** existe un solo canal unidireccional: el origen puede transmitir al destino, pero el destino no puede comunicarse con el origen. Por ejemplo, la radio y la televisión. En una comunicación **half-duplex** existe un solo canal que puede transmitir en los dos sentidos pero no simultáneamente: las estaciones se tienen que turnar. Esto es lo que ocurre con las emisoras de radioaficionados Por último, en una comunicación **full-duplex** existen dos canales, uno para cada sentido: ambas estaciones pueden transmitir y recibir a la vez. Por ejemplo, el teléfono.

4.- Redundancia

La codificación binaria es una codificación redundante, donde el **rendimiento (r)** se define como la relación entre la cantidad de información en bits referida a las posibilidades del código redundante, y la cantidad de información en bits referida a las posibilidades totales de ese mismo código.

$(\%)r = (In / It) \cdot 100$	Donde: In = cantidad de información útil
rendimiento	It = cantidad de información total

Partiendo de esto, para un código BCD que usa 4 símbolos para representar 10 símbolos distintos:

$(\%)R = 10/2^4 \cdot 100 = 10/16 \cdot 100 = 0.625 \cdot 100 = 62.5\%$

Al ser R menor que 100, se aprecia que el código es redundante, además su redundancia se calcula de la siguiente forma:

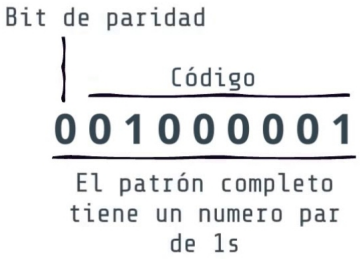
$$R=(1-r)\cdot 100 = (1-0.625)\cdot 100 = 37.5\%$$

Por lo tanto, la redundancia del código BCD es del 37.5%

Con esto podemos decir que un código poco o nada redundante es el que aprovecha al máximo todas las posibilidades que tiene de representar la información. Pero algunas veces es útil que un código sea redundante, ya que esto puede facilitar la detección de errores.

4.1.- Bit de paridad

Se trata de un método para detectar errores y que consiste en añadir un bit de paridad. Existen criterios distintos para este bit: paridad par y paridad impar. El método consiste en añadir un bit más, justo antes de la transmisión, que será 1 o 0, en función de si el dato contendrá un número par de unos (paridad par) o impar (paridad impar). Este método no es capaz de detectar el cambio en más de un bit.



4.2.- Paridad vertical

Consiste en añadir un bit de paridad vertical para cada grupo de bytes, de forma que cada grupo de bytes lleva asociado un bit de paridad por cada columna. Añadiendo esta paridad, podemos además de detectar un error en el cambio de un bit, localizar donde se ha producido el error y así corregirlo. Si el error alcanza a más de un bit, se podría detectar aunque no corregir.

Detección de bit erróneo al añadir paridad vertical

4.3.- Código Hamming

Es capaz de detectar errores múltiples y corregir errores sencillos utilizando en proporción menos bits de paridad. Consiste en añadir al dato varios bits de paridad colocados en las posiciones que son potencia de dos, de forma que cada uno de estos proteja a varios bits del dato.

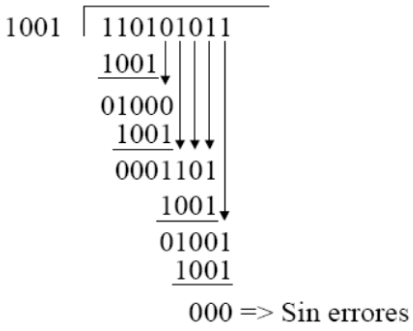
4.4.- Códigos de redundancia cíclica (CRC)

Pretenden la detección de errores en la transmisión serie, cuyos errores suelen afectar a varios bits consecutivos. Consiste en añadir a cada bloque su residuo (módulo) con respecto a un valor concreto representado por un polinomio, de forma que estos cálculos sean simples y se realicen mientras se envían y reciben datos. Estos códigos se basan en el uso de un **polinomio generador G(X)** de grado r, y en el principio de que **n** bits de datos binarios se pueden considerar como los coeficientes de un polinomio de orden n-1. Por ejemplo, los datos 10111 pueden tratarse como el polinomio $x^4 + x^2 + x^1 + x^0$. A estos bits de datos se añaden **r bits de redundancia** de forma que el polinomio resultante sea divisible por el polinomio generador, sin generar resto.

Ejemplo: El emisor quiere enviar la trama 110101 siendo r = 3 y G = 1001. Entonces, $M \cdot 2^r$ =110101000 que dividido (división de módulo 2) entre G produciría R = 011

```
          110011
      ┌──────────
1001  │ 110101000
        1001
        01000
        1001
        0001100
           1001
           01010
           1001
           011 = R (3 bits)
```

En el receptor se recibirá T' y procederá a dividirlo entre G



5.- Conclusiones

Desde la aparición de la informática, maś bien de los primeros ordenadores, se hicieron esenciales otros sistemas de numeración que se adaptaran al diseño arquitectónico de los ordenadores de cada época, siendo el sistema binario el que predomina y en el que está basado, al menos hasta el momento, toda la capacidad de cómputo de los ordenadores y de almacenamiento de la información.

Entre tanto, también se emplean otros sistemas de numeración y de representación de la información de diferente naturaleza a la que se le ha dado distintas soluciones según los casos, mediante códigos externos o de E/S.

A todo esto se le suma la opción de transmitir la información entre sistemas, donde los métodos existentes para asegurar la calidad de la transmisión y sobre todo en asegurar el contenido de la misma, han sido mejorados a lo largo de los años, estableciendo una estrecha relación con las matemáticas.

5.1.- Otras consideraciones

Una posible ampliación a este tema podría ser mediante las operaciones aritmético-lógicas que se pueden dar en el sistema binario, por un lado viendo la suma, resta, multiplicación y división, y por otro lado las operaciones lógicas, algo que se debe dejar para temas posteriores, que sí tratan las puertas lógicas y la lógica binaria.

5.2.- Ámbitos de aplicación docente

- Ciclos formativos de grado superior de ASIR (ISO), DAW (SI), DAM (SI)
- Ciclos formativos de grado medio SMR: SO
- Bachillerato (TIC I, Programación y computación)
- ESO (TIC, Computación y Robótica)

6.- Referencias bibliográficas

- Introducción a la Informática.Alberto Prieto. Antonio Lloris. Juan Carlos Torres. Ed. McGraw Hill. 2ª Edición. 1995.
- Redes de computadoras. Andrew S. Tanenbaum. Ed. Prentice Hall. 3ª Edición. 1997