**Tema 2: Sistemas de numeración, aritmética y códigos**

**Operaciones en binario**

* **Suma:** Comenzando por la derecha y propagando el acarreo hacia la izquierda (1+1) = (10)
* **Resta:** Exige un prestamo en la columna de bits a la izquierda a la restada (en el caso de 0-1)
* **Producto:** Como en decimal, desplazando cada producto parcial una posición a la izquierda.
* Desbordamiento: Se produce cuando el resultado de una operación necesita un número de bits mayor que el de los operandos

**Codificación de números binarios con signo**

* **Signo-magnitud:** El bit más a la izquierda es el bit del signo, 1 negativo y 0 positivo.
  + Los bits de magnitud son el número sin signo.
  + Con n bits se pueden representar números en el intervalo {-(2n-1 -1), (2n-1 -1)}
  + Ej.: -16 = 110000
* **Complemento a 1:** El bit más a la izquierda es el bit del signo. Además, los números negativos son el complemento a 1 del correspondiente número positivo: intercambia los 0 y los 1.
  + Mismo intervalo que signo-magnitud
  + Dos representaciones posibles para el 0: 0000(+0) o 1111(-0)
  + Ej.: -16 = 101111
* **Complemento a 2:** El bit a la izquierda es el signo, los números negativos son el complemento a 2 del positivo:
  + Para calcular el complemento a 2 de un binario N de n bits, se calcula la diferencia 2^n-N y se toman los n bits más a la derecha.
    - Para los números que no son los extremos del intervalo de representación, se puede hacer el complemento a 1 y sumar 1.
    - Para los extremos del intervalo (ej. -16) se hace el complemento a 1, se suma 1 y sobra un bit a la izquierda que se elimina.
    - Ej.: -16 = ~~1~~**10000**
  + Intervalo: {-(2n-1), (2n-1 -1)} (**1 número negativo más** que signomagnitud)
  + 0 sólo tiene una representación, 0. Non existe o -0.
* **NOTA:** Para C1 y C2, los positivos son iguales que en signo magnitud.

**Operaciones en distintas notaciones**

* **Complemento a 2**
  + **Suma**: Se expresa cada operando en notación C2 y se suma igual que en binario, **suprimiendo** el acarreo final.
    - NOTA: Se deben expandir los sumandos (con 0s a la izquierda de ser positivos, o con 1s de ser negativos) para que tengan el mismo nº de bits que requiere la solución. Esos bits expandidos NO se suprimen.
  + De ser una resta, se opera como una suma con distintos signos.
  + **Producto:** Se expresan los números en positivo, se multiplica como si fuesen decimales y luego se cambia el signo del resultado para que coincida. De ser positivo se añade un 0 a la izquierda.
* **Complemento a 1:**
  + Suma: Se suma igual que en binario. El acarreo final se **suma** en el bit de la derecha (pero no el desbordamiento).
* **Signo-magnitud:**
  + Suma: Si son del mismo signo, se suman sólo los bits de magnitud. Si son de distinto signo, se coloca el número mayor (valor absoluto) encima y se restam. Luego se añade el bit del signo.
    - Como alternativa, se puede convertir a C1 los sumandos y luego volver a SM.
    - Ejemplo: 7-8 = 11000-00111 = 10001 = -1

**Sistema de numeración octal**

* Sistema de numeración posiciónal de base 8, con 8 dígitos.
  + Para convertir de octal a decimal, se multiplica cada dígito por 8^su posición.
  + De decimal a octal: convertir el decimal a binario, y de binario ao octal
  + Para convertir de octal a binario: convertir cada cifra individualmente, luego unirlas.
  + De binario a octal: Agrupar las cifras de 3 en 3 y convertirlas a octal.
    - Se comienza a agrupar desde el punto, tanto si hay decimales como no

**Sistema de numeración hexadecimal**

* Sistema de numeración posiciónal de base 16, con 16 dígitos: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
  + Para convertir de hexadecimal a binario, se sustituye cada dígito por los 4 bits correspondientes.
  + Para convertir de binario a hexadecimal, se agrupan los bits en 4 y se convierten a hexadecimal.

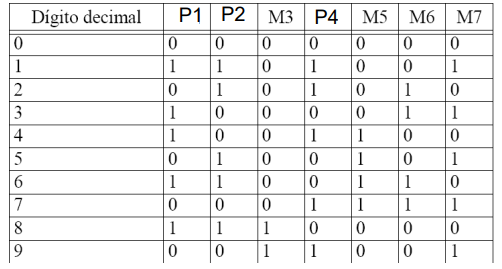
**Codificación en BCD**

* Paso directo de un número decimal a BCD: se analiza cada cifra del número decimal, se convierte a binario usando 4 bits y luego se vuelven a unir.
  + Ejemplo (51): 5 = 0101, 1=0001, 51=01010001
* Para sumar en BCD, se suman las cifras por separado.
  + Si la suma de los dígitos decimales es <=9 la suma es correcta.
  + Si es >9, habrá desbordamiento. Para corregirlo se suma **6** en binario al resultado final.
    - Ejemplo: 5+5 => 0101 + 0101 = 1010 + **0110** =>0001 0000=> 10

**Códigos con paridad**

* Códigos redundantes con un bit que indica si el número es par o impar. Se pueden usar para encontrar errores de 1 bit de distancia.
* Distancia mínima: número de bits que diferencian dos palabras distintas. Con distancia mínima=2 se detectan errores de 1 bit.

**Códigos Hamming**

* Código con distancia mínima 3 con paridad **par** (si se indica paridad impar, las combinaciones deberán tener un número de 1s impar).
* Ejemplo: Codificar las cifras del 0 al 9 en binario, pero con las posiciones 1,2 y 4 del número siendo bits de paridad.
* P1 es la paridad de M3,M5 y M7
* P2 es la paridad de M3,M6 y M7
* P4 es la paridad de M5,M6 y M7
* Si el código es correcto, el número de 1s en las combinaciones [P1,M3,M5,M7], [P2,M3,M6 y M7], [P3,M5,M6 y M7] debe ser **par**. De no serlo, conocemos que hay un error.
  + Los números de 1s en los conjuntos [P1,M3,M5,M7] y los otros conjuntos son los **bits de comprobación de paridad** C1, C2 y C4. Ci=1 si es impar, Ci=0 si es par.
  + Para un mensaje de tamaño m, se requieren k+m<=2k-1 bits de paridad. (en este caso, k=3 y m=4, lo cual cumple que 3+4<=7)

**Códigos Hamming SEC/DED**

* Código de distancia mínima = 4 que permite corrección de errores simples (en 1 bit) y detección de errores dobles.
* Para 4 bits de mensaje añade 3 bits de paridad y 1 bit de paridad total PT, que es la paridad de todos los demás bits del código.
* Situaciones posibles:
  + No hay error si PT=0 y todos los Ci=0
  + Error doble si PT=0 pero algún Ci=1
  + Error simple si PT=1 y algún Ci=1 (en ese caso, se puede localizar donde está el error por el Ci) o si CT=1 y todos los Ci=0 (error en el CT)

**Otras codificaciones**

* Codificación en exceso a 3: Se obtiene sumando 3 al número en BCD
* Código Grey: Código usado en Karnaugh, dígitos con distancia 1, sin peso
  + De binario a Grey:A cada bit se le suma el de su izquierda, sin acarreos
  + De Grey a binario: El primer bit queda igual, luego a cada bit de Grey se le suma el de su izquierda en binario.