UTN
Facultad Regional Córdoba

TUP

Tecnicatura Universitaria en Programación

SISTEMAS DE PROSESAMIENTO DE DATOS

Unidad Temática N°2:
Representación de Información

Material Teórico
1° Año – 1° Cuatrimestre

cc BY ND NC

Índice

UNIDAD N° 2: REPRESENTACIÓN DE LA INFORMACIÓN 2
INTRODUCCIÓN 3
REGLAS DE LOS SISTEMAS DE NUMERACIÓN POSICIONALES 4
SISTEMA DE NUMERACIÓN DECIMAL 6
Notación Polinomial 6
SISTEMA DE NUMERACIÓN BINARIO 8
Tabla de Pesos Binarios 9
Potencias positivas de dos (número entero) 9
Potencias negativas de dos (número fraccionario) 9
SISTEMA DE NUMERACIÓN OCTA 9
SISTEMA DE NUMERACIÓN HEXADECIMAL 10
TABLA DE EQUIVALENCIA ENTRE LOS DISTINTOS SISTEMAS DE NUMERACIÓN 11
CONVERSIÓN NUMÉRICA EN DISTINTAS BASES 12
Conversión de Sistema Numérico Decimal a otras Bases 12
Conversión de otras Bases a Sistema Numérico Decimal 15
Conversión entre Bases que son Potencias unas de otras (Binario-Octal-Hexadecimal) 16
OPERACIONES ARITMÉTICAS 18
Operaciones Aritméticas con Números Binarios 18
Complemento A 1 y Complemento A 2 de Números Binarios 20
Números Binarios con Bit de Signo 21
Resta de Números Binarios por Complemento 22
Suma y Resta en Sistema Octal 24
Suma y Resta en Sistema Hexadecimal 25
SISTEMAS DE CODIFICACIÓN 25
Códigos Binarios de 4 Bits de Números Decimales (BCD) 25
Códigos de Caracteres o Alfanuméricos 28
CÓMO SE UTILIZA LA TABLA ASCII8 29
BIBLIOGRAFÍA 32

UNIDAD N° 2: REPRESENTACIÓN DE LA INFORMACIÓN

Objetivo Principal

Orientar al alumno a representar los sistemas numéricos decimal, binario y hexadecimal empleando la notación posicional y polinomial.

Objetivos Específicos

Conocer los diferentes Sistemas de Numeración

Convertir números de base 10 a un equivalente en base 2 o 16 empleando las operaciones de división, para los enteros, y multiplicación, para las fracciones.

Convertir números en base 2 o 16 a base 10 empleando el polinomio de expansión numérica.

Convertir en forma directa números cuya base origen n es potencia de la base destino m.

Resolver operaciones aritméticas de suma, resta, multiplicación y división de números binarios; suma y resta de números en base hexadecimal, empleando los principios básicos de la aritmética decimal.

Restar números binarios utilizando complementos.

Gráfico 1: Elaboración propia.

INTRODUCCIÓN

Sistemas de Numeración

Un **sistema de numeración** es un conjunto de símbolos (números, letras…) y reglas que permiten construir todos los **números** válidos.

Los sistemas de numeración pueden clasificarse en dos grandes grupos: posicionales y no-posicionales:

- En los sistemas no-posicionales los dígitos tienen el valor del símbolo utilizado, que no depende de la posición (columna) que ocupan en el número. **Ejemplo:** Sistema egipcio, Romano
- En los sistemas de numeración posicionales el valor de un dígito depende tanto del símbolo utilizado, como de la posición que ese símbolo ocupa en el número. **Ejemplo:** Sistema Decimal, Binario, Octal y Hexadecimal.

En el Sistema de Numeración Binario en el que hay tan solo dos valores es el que realmente representa la importancia de los circuitos digitales y su comportamiento. Solo hay dos estados posibles o se es o NO se es, pero no hay intermedios. Encendido o apagado, funciona o no funciona, activado o desactivado, en cada caso existe un uno o existe un cero lógico.

Cuadro 1: Elaboración propia.

REGLAS DE LOS SISTEMAS DE NUMERACIÓN POSICIONALES

Los egipcios representan una de las civilizaciones más antiguas y desarrolladas del mundo. Gracias a la existencia de los papiros de *Rhind* y de sus múltiples jeroglíficos es que se sabe algo acerca de su aritmética. Aunque emplearon el sistema duodecimal en la subdivisión del año (en doce meses, correspondientes a sus doce dioses principales) y del día (en doce horas de claridad y doce de tinieblas), su numeración era decimal y contaba con signos jeroglíficos para las cifras del uno al diez y para cien, mil, diez mil, cien mil y un millón.

Imagen 1: Sistema de Numeración Egipcia.

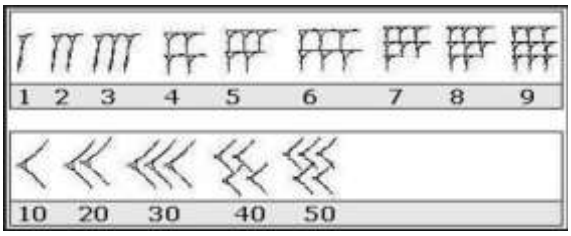


Imagen 2: Sistema de Numeración Babilónica.

Los babilonios, al igual que los egipcios, desarrollaron su propio sistema de numeración, ellos escribían sobre tablillas de arcilla, en donde utilizaban la escritura cuneiforme y no tenían ningún símbolo para representar el cero. Utilizaban un sistema de numeración de valor posicional a través de dos símbolos básicos en forma de cuña. Una en forma vertical para las unidades y otra en forma horizontal para las decenas.

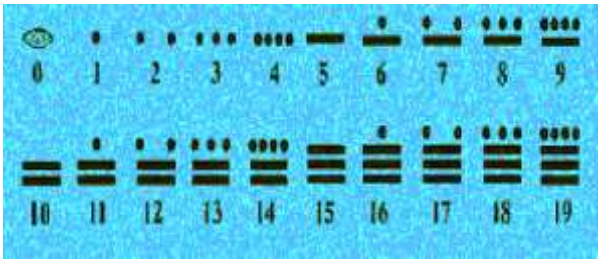


Imagen 3: Sistema de Numeración Maya.

Los mayas inventaron un sistema de numeración en donde aparece por primera vez el cero, además de que su base era el veinte, ya que se cree, que tal vez sea por el hecho de contemplar los dedos de pies y manos. Esta civilización representó cada cantidad por medio de símbolos que según la posición que ocupaban adquiría cierto valor, es decir el sistema maya así como el decimal es un sistema de posiciones.

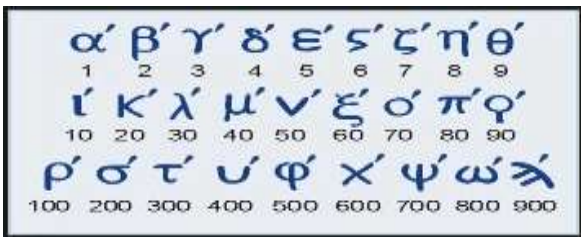


Imagen 4: Sistema de Numeración Griega

El símbolo del cero en cualquier posición indica ausencia de cantidad.

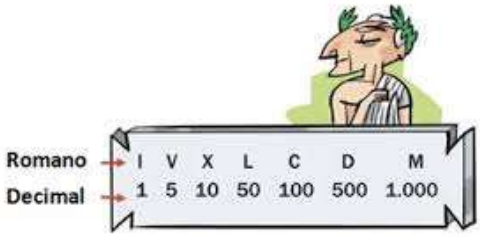


Imagen 5: Sistema de Numeración Romana.

Los hindúes representaron con nueve símbolos diferentes, uno por cada número del uno al nueve. Éstos han cambiado con el tiempo, pero llegaron a Europa en su forma actual en el siglo XVI.

Por su parte, los griegos y los hebreos, utilizaron nueve símbolos diferentes para estos números. En cada caso, los símbolos eran las primeras nueve letras de sus alfabetos.

El Imperio Romano desarrolló un sistema de numeración no posicional que se usó en Europa hasta el siglo XVII.

1. Elegido un número b >1 como base del sistema de numeración, se utilizan b símbolos, llamados cifras o guarismos (0, 1, 2, ..., b-1) que representan el cero y los primeros números naturales.
2. Los símbolos básicos se ordenan en forma monótona creciente, a partir del cero.
3. La ausencia de elementos se representa con un cero y la unidad con un uno.
4. La base del sistema de numeración (b), es 10 para todo sistema de numeración. Es un número finito.
5. En un Sistema de Numeración están definidas las operaciones de suma y producto.
6. En un sistema de base b, un número N cualquiera se puede representar mediante el Polinomio de Expansión Numérica. Es un polinomio de potencias de la base, multiplicados por un símbolo perteneciente al sistema.

Gráfico 2: Elaboración propia.

SISTEMA DE NUMERACIÓN DECIMAL

Es el más utilizado, cuenta con diez elementos o símbolo básicos: 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9. Las operaciones que en él se pueden presentar son las aritméticas (suma, resta, multiplicación, división, potenciación, etc.) y lógicas (Unión - Disyunción, Intersección - Conjunción, Negación, Diferencia, Complemento, etc.). Las relaciones entre los números del sistema decimal son mayor que, menor que, igual y a nivel lógico son pertenencia y contenencia.

BASE	PESO	FRACCIONARIOS
•El sistema de numeración decimal con sus diez dígitos, de 0 hasta 9, es un sistema en base diez.	•La posición de cada dígito en un número decimal indica la magnitud de la cantidad reservada, y se le puede asignar un **peso**. •Los pesos para los números enteros son potencias positivas de diez, que aumentan de derecha a izquierda, comenzando por $10^0 = 10^n$. ... 10^5 10^4 10^3 10^2 10^1 10^0	•Los pesos son potencias negativas de 10 que aumentan de izquierda a derecha, comenzando por 10^{-1} ·A partir de la coma decimal.Ej: 10^n....10^2 10^1 10^0 ,10^{-1} $10^{-2}10^{-3}10^{-4}$... •El valor de un número decimal es la suma de los dígitos después de haber multiplicado cada dígito por su peso.

Gráfico 3: Elaboración propia.

Notación Polinomial

En términos generales cualquier número *N,* puede ser escrito de acuerdo a la siguiente fórmula:

$$N = \sum_{i=-m}^{n-1} a_i r^i$$

Un número del sistema decimal tiene la siguiente representación Polinómica:

$$(N)_{10} = a_n*10^n + a_{n-1}*10^{n-1} + a_{n-2}*10^{n-2} +... + a_0*10^0 + a_{-1}*10^{-1} +... + a_{-p}*10^{-p}$$

Siendo:

N el número decimal,

ai el número relativo que ocupa la posición i-ésima,

n número de dígitos de la parte entera (menos uno),

p número de dígitos de la parte fraccionaria

Ejemplo

Expresar el número decimal 568,23 como suma de los valores de cada dígito.

Solución

El dígito 5 de la parte entera del número tiene un peso 100, es decir 10^2

El dígito 6 tiene un peso de 10, que corresponde a 10^1.

El dígito 8 tiene un peso de 1, que es 10^0.

El dígito 2 de la parte fraccionaria tiene un peso 0,1, es decir 10^{-1}.

Gráfico 4: Elaboración propia.

$$568{,}23 = (5 \times 10^2) + (6 \times 10^1) + (8 \times 10^0) + (2 \times 10^{-1}) + (3 \times 10^{-2})$$
$$= (5 \times 100) + (6 \times 10) + (8 \times 1) + (2 \times 0{,}1) + (3 \times 0{,}01)$$
$$= 500 + 60 + 8 + 0{,}2 + 0{,}03$$

Los dígitos pueden tener dos valores: un valor absoluto que indica el número de unidades que lo forman, y un valor relativo que es el que adquieren según la posición que ocupan dentro del numeral.

Ejemplo

El valor absoluto de los dígitos que forman **385** es**: 3, 8, 5.** Por su parte, el valor relativo es **300, 80 y 5.**

Las cifras que intervienen en un número se dividen en períodos de seis cifras cada uno de la siguiente forma:

Tercer periodo *Billones*						Segundo periodo *Millones*						Primer periodo *Unidades*					
Segundo grupo *Miles*			Primer grupo *Unidades*			Segundo grupo *Miles*			Primer grupo *Unidades*			Segundo grupo *Miles*			Primer grupo *Unidades*		
Tercer grupo *Centenas*	Segundo grupo *Decenas*	Primer grupo *Unidades*	Tercer grupo *Centenas*	Segundo grupo *Decenas*	Primer grupo *Unidades*	Tercer grupo *Centenas*	Segundo grupo *Decenas*	Primer grupo *Unidades*	Tercer grupo *Centenas*	Segundo grupo *Decenas*	Primer grupo *Unidades*	Tercer grupo *Centenas*	Segundo grupo *Decenas*	Primer grupo *Unidades*	Tercer grupo *Centenas*	Segundo grupo *Decenas*	Primer grupo *Unidades*

Tabla 1: Periodos de seis cifras.

SISTEMA DE NUMERACIÓN BINARIO

El sistema de numeración Binario es un conjunto de elementos formado por los símbolos básicos 0 y el 1. Se definen operaciones aritméticas (suma, resta, multiplicación, división) y lógicas (OR, AND y NOT) entre ellos. A través reglas permite establecer relaciones y operaciones entre sus dos elementos.

La utilización casi exclusiva de este sistema de numeración en los equipos de cálculo y control automático es debida a la seguridad y rapidez de respuesta de los elementos físicos que poseen dos estados diferenciados y a la sencillez de las operaciones aritméticas en este sistema (para representar una misma cantidad) que en los sistemas cuya base es mayor de dos.

BASE

- El sistema de numeración Binario sólo tiene 2 dígitos, 0 y 1, es un sistema en base dos.

PESO

- La posición de cada dígito binario (0 o 1), indica su peso o valor dentro del número.
- Los pesos de un número binario están basados en potencias de 2.
- Los pesos para los números enteros son potencias positivas de dos, que aumentan de derecha a izquierda, comenzando por $2^0 = 2^n$. ... 2^5 2^4 2^3 2^2 2^1 2^0

FRACCIONARIOS

- Los pesos son potencias negativas de 2 que aumentan de izquierda a derecha, comenzando por 2^{-1} ·A partir de la coma decimal.
- Estructura de pesos de Números binarios. Ej: 2^n...2^2 2^1 2^0 ,2^{-1} 2^{-2} 2^{-3} 2^{-4}

Gráfico 5: Elaboración propia.

Tabla de Pesos Binarios																	
Potencias positivas de dos (número entero)									**Potencias negativas de dos (número fraccionario)**								
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}			
256	128	64	32	16	8	4	2	1	1/2	1/4	1/8	1/16	1/32	1/64			
									0,5	0,25	0,125	0,625	0,03125	0,015625			

Tabla 2: Tabla de pesos Binarios.

SISTEMA DE NUMERACIÓN OCTA

El sistema octal es de base ocho, es decir para la representación de las cantidades utiliza ocho símbolos básicos diferentes que son los dígitos del **0** al **7**.

El interés de este sistema, es debido a que 8 es una potencia de 2 (2^3=8), por lo tanto resulta muy sencilla la conversión de los números del sistema binario al octal y viceversa, permitiendo una representación compacta de cantidades binarias. Cabe aclarar, que el CPU sólo trabaja en Sistema binario. Cada dígito octal se representa mediante un número binario de 3 bits.

BASE	PESO	FRACCIONARIOS
•El Sistema de Numeración Octal, es un sistema en base 8. Tiene 8 símbolos básicos: 0 - 7 dígitos.	•La posición de cada dígito octal, indica su peso o valor dentro del número. •Los pesos para los números enteros son potencias positivas de 8, que aumentan de derecha a izquierda, comenzando por $8^0 = 8^n$. ... 8^5 8^4 8^3 8^2 8^1 8^0	•Los pesos son potencias negativas de 8 que aumentan de izquierda a derecha, comenzando por 8^{-1} ·A partir de la coma decimal. •Estructura de pesos de Números Octales. Ej: 8^n...8^2 8^1 8^0 ,8^{-1} 8^{-2} 8^{-3} 8^{-4}

Gráfico 6: Elaboración propia.

Ejemplo:

Potencias de 8	...	8^3	8^2	8^1	8^0	8^{-1}	8^{-2}	...
Valor	...	512	64	8	1	0.125	0.016	...
		1	2	5	0 .	3	6	

$$1250.36_8 = 1 \cdot 512 + 2 \cdot 64 + 5 \cdot 8 + 0 \cdot 1 + 3 \cdot 0.125 + 6 \cdot 0.016 =$$
$$= 680.471_{10}$$

SISTEMA DE NUMERACIÓN HEXADECIMAL

El sistema hexadecimal es de base dieciséis, es decir para la representación de las cantidades utiliza dieciséis símbolos básicos diferentes que son los dígitos del **0** al **9** y las letras del alfabeto de la **A** a la **F**.

El interés de este sistema, es debido a que 16 es una potencia de 2 (2^4=16), por lo tanto resulta muy sencilla la conversión de los números del sistema binario al hexadecimal y viceversa, permitiendo una representación compacta de cantidades binarias. Cabe aclarar, que el CPU sólo trabaja en Sistema Binario. La mayoría de los sistemas digitales procesan grupos de datos binarios que son múltiplos de cuatro bits, lo que hace al número hexadecimal muy adecuado, ya que cada dígito hexadecimal se representa mediante un número binario de 4 Bits.

BASE	PESO	FRACCIONARIOS
• El Sistema de Numeración Hexadecimal, es un sistema en base 16. Tiene 16 símbolos básicos: 0 - 9 dígitos y A-F caracteres alfabéticos.	• La posición de cada dígito hexadecimal, indica su peso o valor dentro del número. • Los pesos para los números enteros son potencias positivas de 16, que aumentan de derecha a izquierda, comenzando por 16^0 = 16^n ... 16^5 16^4 16^3 16^2 16^1 16^0	• Los pesos son potencias negativas de 16 que aumentan de izquierda a derecha, comenzando por 16^{-1} ·A partir de la coma decimal. • Estructura de pesos de Números hexadecimales. Ej: 16^n...16^2 16^1 16^0 ,16^{-1} 16^{-2} 16^{-3} 16^{-4} .....

Gráfico 7: Elaboración propia.

Ejemplo:

Potencias de 16	...	16^2	16^1	16^0	16^{-1}	...
Valor	...	256	16	1	0.0625	...

C 2 A . 3

$$C2A.3_{16} = 12 \bullet 256 + 2 \bullet 16 + 10 \bullet 1 + 3 \bullet 0.0625 = 3114.1875_{10}$$

TABLA DE EQUIVALENCIA ENTRE LOS DISTINTOS SISTEMAS DE NUMERACIÓN

DECIMAL	BINARIO	OCTAL	HEXADECIMAL
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

Tabla 3: Equivalencias entre los distintos sistemas de numeración

CONVERSIÓN NUMÉRICA EN DISTINTAS BASES

Los números pueden representarse en diversos sistemas de numeración, que se diferencian por su base. Resulta importante cuestionarse la utilidad práctica de aplicar estos métodos de conversión numérica.

Por ejemplo, el CPU trabaja en Sistema de Numeración Binario. Si se quiere interpretar fácilmente las etiquetas de las direcciones de memoria RAM, cuando la unidad de control del microprocesador da una orden de *Memory Read o Memory Write* en una determinada celda de la misma, dicha dirección se representa en Sistema de Numeración Hexadecimal, a los fines de compactar las cifras binarias y resulte menos engorroso para el usuario, que en este caso se puede tratar de un desarrollador Assembler. Pero otro tipo de usuario puede requerir que se le muestren todas las especificaciones en el sistema numérico base 10.

Conversión de Sistema Numérico Decimal a otras Bases

El pasaje de un número decimal a otra base se realiza analizando por separado la parte entera (prescindiendo del signo) y la parte fraccionaria. **División sucesiva.**

Parte Entera: Se divide el número por la base a la que se desea convertir (división entera); luego se divide el cociente obtenido por la base, y así sucesivamente hasta que el cociente sea cero. Los restos enteros de cada división, que son siempre menores que la base, son los dígitos de la nueva representación del número, ordenados del menos significativo al más significativo.

Parte Fraccionaria: Se multiplica la parte fraccionaria por la base a la que se desea convertir.

La parte entera del resultado, que siempre es menor que la base, será el primer dígito fraccionario de la nueva representación. Se multiplica ahora la parte fraccionaria del resultado por la base y así sucesivamente.

Conversión de Sistema Decimal al Sistema Binario

Ejemplo: representar en binario número $524{,}625_{10}$ = $1000001100{,}101_2$

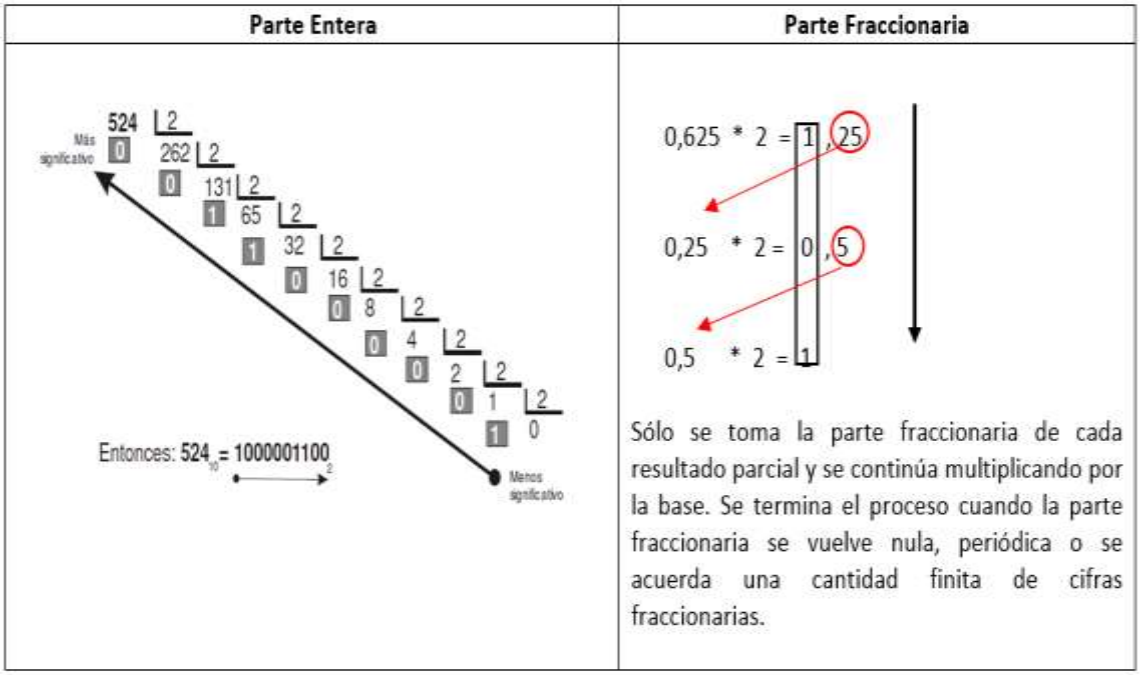


Tabla 4: Representación de conversión a números binarios

Conversión de Sistema Decimal Sistema Binario por el Método de Suma de pesos

Una forma de calcular el número binario equivalente a un número decimal dado es determinar el conjunto de pesos binarios, cuya suma es igual al número decimal.

Ejemplo

Convertir los siguientes números decimales a binario:

$12 = 8 + 4 = 2^3 + 2^2 \longrightarrow 1100$

$25 = 16 + 8 + 1 = 2^4 + 2^3 + 2^0 \longrightarrow 11001$

$58 = 32 + 16 + 8 + 2 = 2^5 + 2^4 + 2^3 + 2^1 \longrightarrow 111010$

$82 = 64 + 16 + 2 = 2^6 + 2^4 + 2^1 \longrightarrow 1010010$

Conversión de Sistema Decimal al Sistema Octal

Ejemplo: representar en Hexadecimal el número $4665{,}65625_{10} = 11071{,}52_{8}$

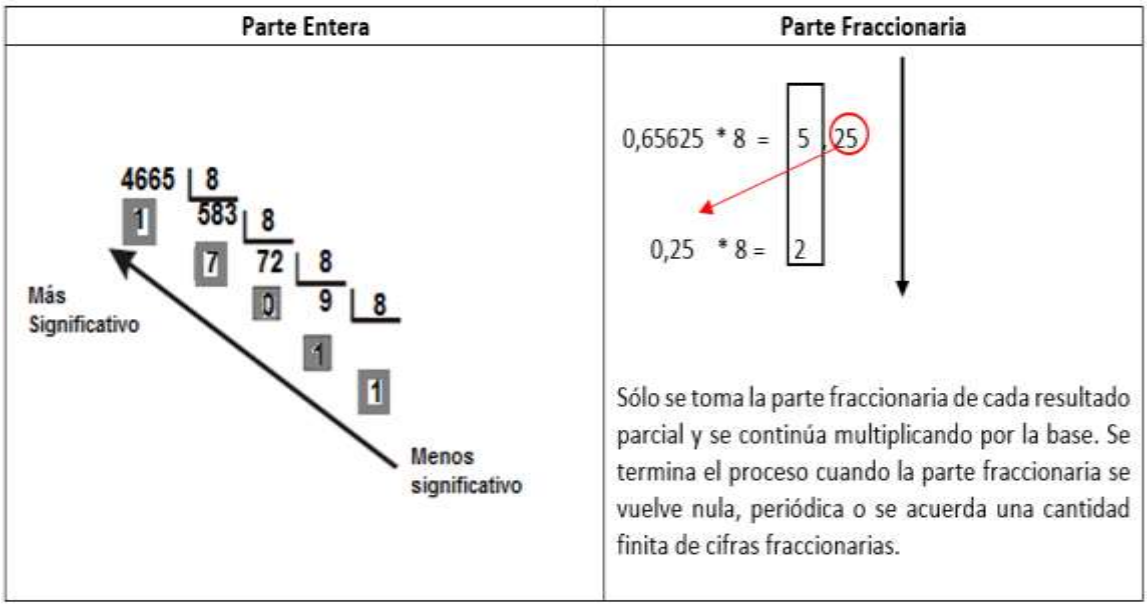


Tabla 5: Representación de conversión a número octal

Conversión De Sistema Decimal al Sistema Hexadecimal

Ejemplo: representar en Hexadecimal el número $3511{,}65625_{10} = DB7{,}A8_{16}$

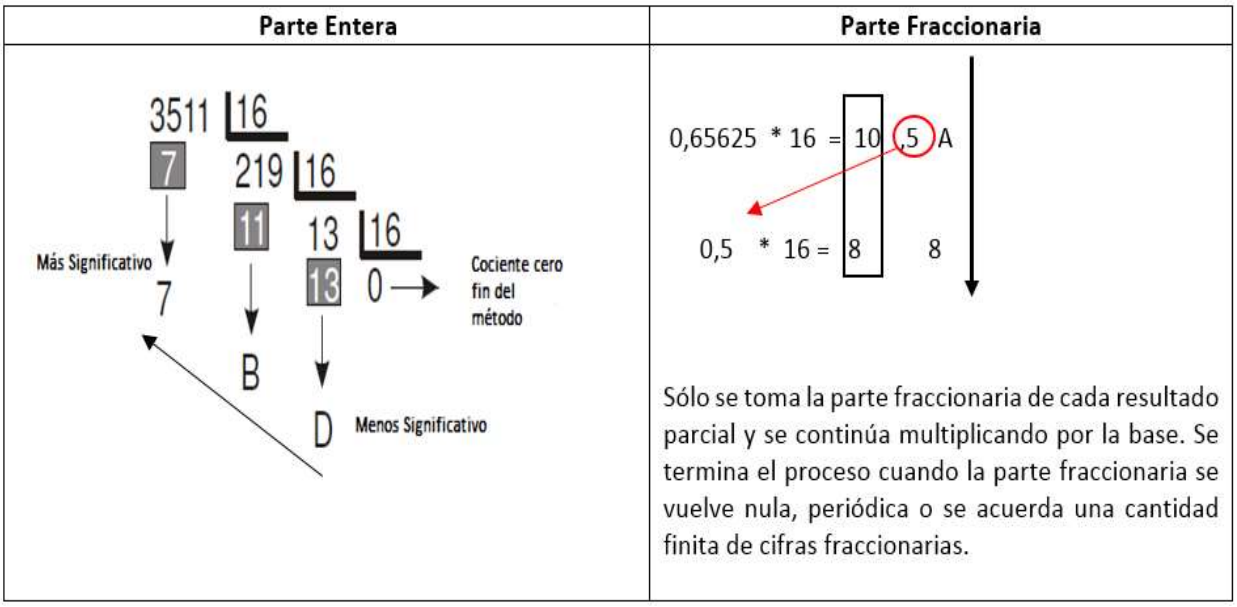


Tabla 6: Representación de conversión a número Hexadecimal

Conversión de otras Bases a Sistema Numérico Decimal

A los efectos de hallar la representación decimal de un número expresado en otra base cualquiera ***b***, se utiliza el **polinomio de expansión numérica** como método de conversión.

- **CONVERSIÓN DE SISTEMA NUMERICO BINARIO A SISTEMA NUMERICO DECIMAL:** Se opera internamente en decimal. Es decir que al representar la base binaria en el polinomio, esta se expresa con el numeral 2.
- **CONVERSIÓN DE SISTEMA NUMERICO OCTAL A SISTEMA NUMERICO DECIMAL:** Se opera internamente en decimal. Es decir que al representar la base octal en el polinomio, esta se expresa con el numeral 8. Entonces se multiplica el valor decimal de cada dígito octal por su peso y luego se efectúa la suma de estos productos.
- **CONVERSIÓN DE SISTEMA NUMERICO HEXADECIMAL A SISTEMA NUMERICO DECIMAL:** Se opera internamente en decimal. Es decir que al representar la base hexadecimal en el polinomio, esta se expresa con el numeral 16, y los caracteres alfabéticos con su correspondiente decimal. Entonces se multiplica el valor decimal de cada dígito Hexadecimal por su peso y luego se efectúa la suma de estos productos. Para un número hexadecimal de 4 dígitos los pesos son:

16^3	16^2	16^1	16^0
4096	256	16	1

Ejemplo:

Binario a decimal $101.01_2 = 1*2^2 + 0*2^1 + 1*2^0 + 0*2^{-1} + 1*2^{-2} = 4 + 0 + 1 + 0 + 1/4 = 5.25_{10}$

Octal a decimal $34.1_8 = 3*8^1 + 4*8^0 + 1*8^{-1} = 24 + 4 + 1/8 = 28.125_{10}$

Hexadecimal a decimal $D56.A2_{16} = 13*16^2 + 5*16^1 + 6*16^0 + 10*16^{-1} + 2*16^{-2} = 3414.6328_{10}$

Conversión entre Bases que son Potencias unas de otras (Binario-Octal-Hexadecimal)

Algunos procedimientos de conversión pueden resultar relativamente sencillos siempre y cuando se cumpla con algunos requisitos mínimos, tales como la existencia de una relación de potencia entera y positiva entre las bases origen B1, y destino B2. A continuación se explican esos procedimientos.

Para convertir un número N de una base origen B1, a una destino B2, en donde se cumple la relación:

B2 = B1k siendo k un número natural.

Se procede a agrupar el número N en grupos de tamaño k, tanto hacia la izquierda como hacia la derecha del punto decimal. Una vez generadas estas agrupaciones se procede a sustituir cada una de ellas por su representación equivalente, según la base destino B2.

En el caso de conversión de B2 a B1 se procede a expresar cada dígito de B2 por su equivalente en B1, utilizando k dígitos para cada uno de ellos.

Conversión de Sistema Numérico Binario a Octal y viceversa

BINARIO A OCTAL **Se agrupan los bits enteros y fraccionarios en grupos de tres a partir del punto decimal y se convierte cada grupo independientemente.** **Para completar el último grupo se añaden los ceros que sean necesarios.** Sea por ejemplo el número **1011.10101_2**, hay que completarlo con 2 ceros a la izquierda y 1 cero a la derecha	$0\,0\,1\,011.101\,010_2 = 1\,3\,,5\,2_8$ 100 110 011 010 ↓ ↓ ↓ ↓ 4 6 3 2 = 4632_8
OCTAL A BINARIO **La base del Sistema Octal es 8, que es una potencia de 2, donde $2^3 = 8$. Por lo tanto, cada dígito Octal se sustituye por 3 dígitos binarios equivalentes.**	3 0 7 4 2 1 4 2 1 4 2 1 $0\,1\,1\,0\,0\,0\,1\,1\,1_{(2)}$

Tabla 7: Representación de conversión de binario a octal y viceversa.

Conversión de Sistema Numérico Binario a Hexadecimal y viceversa

BINARIO A HEXADECIMAL **Se agrupan los bits enteros y fraccionarios en grupos de cuatro a partir del punto decimal y se convierte cada grupo independientemente.** **Para completar el último grupo se añaden los ceros que sean necesarios.** Sea por ejemplo el número **100111.10101_2**, hay que completarlo con 2 ceros a la izquierda y 3 ceros a la derecha	Ceros agregados para completar los grupos de 4 dígitos binarios. Número original. $\underbrace{0\,0}\,\underbrace{1\,0\,0\,1\,1\,1}.\underbrace{1\,0\,1\,0\,1\,0\,0\,0}_{2}$ 2_{16} 7_{16} A_{16} 8_{16} *Equivalentes hexadecimales* Resultando por lo tanto: $100111.10101_2 = 27.A8_{16}$ Otro ejemplo: $-11.01_2 = -\underbrace{001}_{3}1.\underbrace{0100}_{4}{}_2 = -3.4_{16}$
HEXADECIMAL A BINARIO **La base del Sistema Hexadecimal es 16, que es una potencia de 2, donde $2^4 = 16$. Por lo tanto, cada dígito hexadecimal se sustituye por 4 dígitos binarios equivalentes.**	Sea el número **$81AE_{16}$** el equivalente binario será: $81AE_{16} = \underbrace{1000}_{8}\underbrace{0001}_{1}\underbrace{1010}_{A}\underbrace{1110}_{E}{}_2$

Tabla 8: Representación de conversión de binario a hexadecimal y viceversa. Elaboración propia.

Conversión de Sistema Numérico Octal a Hexadecimal y viceversa

HEXADECIMAL A OCTAL Como la base de origen y la magnitud de la base destino, son potencias de 2, lo recomendable es convertir primero a binario y posteriormente convertir el número binario a su equivalente en base 8.	**$FAD,B9_{16} = 7655,562_8$** Se utiliza el "puente binario ", es decir, primero se convierte a binario. **F A D , B 9** 1111 1010 1101 , 1011 1001 Ahora se agrupan en ternas, respetando el lugar de la coma y se completa con ceros las ternas de los extremos, si quedasen incompletas. 111 110 101 101 , 101 110 010 7 6 5 5 , 5 6 2 **R = $7655,562_8$**

Tabla 9: Representación de conversión de hexadecimal a octal y viceversa. Elaboración propia.

OCTAL A HEXADECIMAL	$7410,73_8 = F08,E6_{16}$
Como la base de origen y la magnitud de la base destino, son potencias de 2, lo recomendable es convertir primero a binario y posteriormente convertir el número binario a su equivalente en base 16.	Se utiliza el "puente binario ", es decir, primero se convierte a binario. 7 4 1 0 , 7 3 111 100 001 000 , 111 011 Ahora se agrupan en cuartetas, respetando el lugar de la coma y se completa con ceros las cuartetas de los extremos, si quedasen incompletas. 1111 0000 1000 , 1110 0110 F 0 8 , E 6 R = $F08,E6_{16}$

Tabla 10: Representación de conversión de octal a hexadecimal y viceversa.

OPERACIONES ARITMÉTICAS

Operaciones Aritméticas con Números Binarios

La Unidad Aritmético Lógica, en la CPU del procesador, es capaz de realizar operaciones aritméticas, con datos numéricos expresados en el sistema binario. Las operaciones aritméticas básica (suma, resta, multiplicación y división) en Sistema Binario siguen las mismas reglas de ejecución que en el Sistema Decimal, aunque por la sencillez de su sistema de representación, las tablas de comportamiento son más sencillas. Las operaciones aritméticas entre números representados en una base cualquiera siguen las reglas similares a las de la aritmética decimal.

SUMA BINARIA

La suma binaria toma el valor uno cuando uno solo de los sumandos tiene dicho valor. Cuando ambos sumandos tienen el valor uno, la suma es cero y se produce un acarreo (me llevo 1).

a	b	a+b
0	0	0
0	1	1
1	0	1
1	1	0 y me llevo 1

Ejemplos

```
  1010        1 101010        1   1 1
  +101        +001001        00111001
  ----        -------       +00100011
  1111        110011        --------
                            01011100
```

Cuando la suma de dos dígitos excede los símbolos numéricos disponibles en la notación, se lleva un 1 (**acarreo**) a la posición de dígito inmediatamente superior.

RESTA BINARIA

Restar en binario es, nuevamente, igual que la misma operación en el sistema decimal. Los términos que intervienen en la resta se llaman minuendo, sustraendo y diferencia.

a	b	a-b
0	0	0
0	1	1 y debo 1
1	0	1
1	1	0

Ejemplos:

```
     0 10                10 1
     1 1 0 1 0 1        0 0 10 10
   - 1 0 1 0 0 1        ~~1 1 0 0~~ 0 1 1
     0 0 1 1 0 0      - 1 0 1 0 0 1
                        0 1 1 1 0 1 0
```

La resta 0 - 1 se resuelve, igual que en el sistema decimal, tomando una unidad prestada de la posición siguiente, con acarreo negativo de 1: 10 -1

Tabla 11: Representación de conversión de octal a hexadecimal y viceversa.

PRODUCTO BINARIO

La multiplicación en binario es más fácil que en cualquier otro sistema de numeración.
Como los factores de la multiplicación sólo pueden ser CEROS o UNOS, el producto sólo puede ser CERO o UNO.

a	b	axb
0	0	0
0	1	0
1	0	0
1	1	1

Ejemplo

```
     110101
  x      10
  ---------
     000000
    110101
  ---------
    1101010
```

En una computadora, la operación de multiplicar se realiza mediante sumas repetidas.

DIVISION BINARIA

La división es muy fácil de realizar, porque no son posibles en el cociente otras cifras que UNOS y CEROS.

a	b	a/b
0	0	indeterminado
0	1	0
1	0	infinito
1	1	1

Ejemplo:

```
  110101      | 101
- 101         | 1010,1...
  00110
-   101
  00110
-    0101
     0001 ...
```

Si la división es posible, entonces, el divisor sólo podrá estar contenido **una vez** en el dividendo, es decir, la primera cifra del cociente es un UNO. En ese caso, el resultado de multiplicar el divisor por 1 es el propio divisor. Se Restan las cifras del dividendo del divisor y se baja la cifra siguiente. La división se efectúa a través de restas reiteradas.

Tabla 12: Operaciones binarias. Elaboración propia.

Complemento A 1 y Complemento A 2 de Números Binarios

El **complemento a 1** y el **complemento a 2** de un número binario son importantes porque permiten la **representación de números negativos**. La aritmética en complemento a 2 se usa comúnmente en las computadoras para manipular los números negativos.

Complemento a 1 (o a la base menos 1)	Complemento a 2 (o a la base)
El complemento a 1 de un número N, es el número que resulta de restar cada una de las cifras de N a la base menos uno del sistema de numeración que se esté utilizando. Es equivalente a intercambiar unos y ceros, así el complemento a 1 de 1011 es 0100. El complemento a 1 de un número binario se obtiene cambiando todos los 1 (unos) por 0 (ceros) y todos los 0 (ceros) por 1 (unos). 1 0 1 1 0 0 1 0 Número binario ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ 0 1 0 0 1 1 0 1 Complemento a 1	El complemento a 2 de un número binario se obtiene sumando 1 al complemento a 1 de dicho número binario. **Complemento a 2 = (Complemento a 1) + 1** **Método alternativo** **1.** En el número a complementar, se posiciona en la derecha se rastrean los dígitos y se escriben los bits como están hasta encontrar el primer 1, incluido éste. **2.** Se calcula el complemento a 1 de los bits restantes. **Ejemplo:** Calcular el complemento a 2 de 10111000, utilizando el método alternativo. **Solución.** 10111000 Número binario Complemento a 1 de los bits originales — 01001000 Complemento a 2 Estos bits no varían
Ejemplo: Hallar el complemento a 2 de 10110010 *Solución.* 10110010 Número Binario 01001101 Complemento a 1 + 1 Se suma 1 01001110 Complemento a 2	

Tabla 13: Complementos A1 y A2. Elaboración propia.

Números Binarios con Bit de Signo

Los sistemas digitales, tales como la computadora, deben ser capaces de manejar números positivos y negativos. Un número binario con signo queda determinado por su magnitud y su signo. El signo indica si un número es positivo o negativo, y la magnitud es el valor del número.

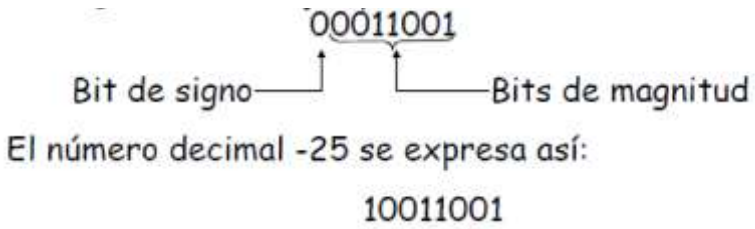
El bit de signo

Se reserva un dígito para representar el signo del número En general, el bit más a la izquierda en un número binario con signo es el **bit de signo**, que indica si el número es positivo o negativo. El significado suele ser:

0, número positivo y 1, número negativo.

Ejemplo:

Representar el número decimal 25 en binario, con signo y magnitud tanto positiva como negativa.



Resta de Números Binarios por Complemento

La Unidad Aritmética y Lógica del CPU, sólo cuenta con un circuito que es el sumador. Razón por la cual, utiliza un método para realizar sustracciones que consiste en sumar al minuendo el complemento a 1 o el complemento a 2 del sustraendo.

Método

1. Determinar el rango, denotado por **n,** o la cantidad de bits que tendrá el minuendo y el sustraendo.
2. Balancear las cantidades, es decir que ambas cantidades deben tener igual cantidad de bits, completando con ceros el número con menor cantidad de dígitos.
3. Se obtiene el complemento a 1 o a 2 del sustraendo o la cantidad que está precedida con el signo –.
4. El complemento obtenido en el paso anterior se suma al minuendo.
5. Se analiza el bit de signo (BS):

BS= 0 el resultado es positivo y corresponde a los bits de magnitud, Existe Overflow y se desprecia.

BS= 1 el resultado es negativo y hay que recomplementar los bits correspondientes a la magnitud.

No existe bit de overflow.

- Caso A (minuendo) > B (sustraendo)

Ejemplo:

Opere en complemento a 1 y a 2: **A: 101010_2 - B: 10101_2**

Rango n = 7 (cantidad de bits del número más grande, en este caso A, + el Bit de signo).

Balanceo de cantidades: A: 0101010_2 y B: 0010101_2 completando con ceros.

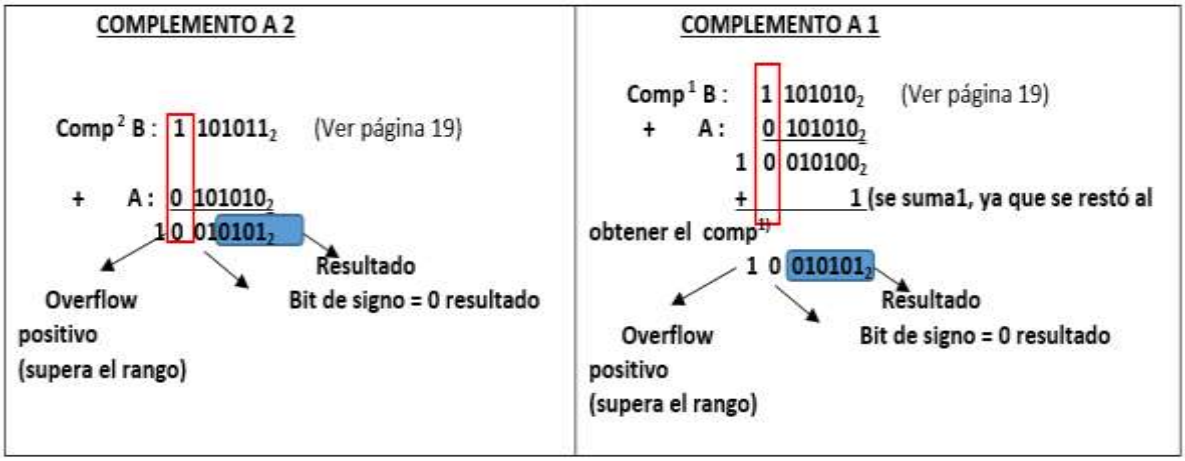


Tabla 14: Resta de números binarios por complemento sin signo.

- Caso A (minuendo) < B (sustraendo):

Ejemplo

Opere en complemento a 1 y a 2: A: 10101_2 - B: 101010_2

Rango n = 7 (cantidad de bits del número más grande, en este caso A, + el Bit de signo).

Balanceo de cantidades: completando con ceros A: 0010101_2 y B: 0101010_2

COMPLEMENTO A 2	COMPLEMENTO A 1
Comp^2 B : 1 010110_2 (Ver página 19) + A : 0 010101_2 1 101011_2 No es el Resultado Bit de signo = 1 resultado hay que No hay Overflow negativo (no supera el rango) recomplementarlo C^b Resultado = - 010101_2	Comp^1 B : 1 010101_2 (Ver página 19) + A : 0 010101_2 1 101010_2 + 1 (se suma1, ya que se restó al obtener el comp$^{1)}$ 1 0 101011_2 Resultado Bit de signo = 1 resultado No hay Overflow negativo (no supera el rango) hay que recomplementarlo C^b Resultado = - 010101_2

Tabla 15: Complementos A1 y A2 con signo

Suma y Resta en Sistema Octal

SUMA OCTAL	RESTA OCTAL
Ejemplo: Aquí se muestra el Método de "restar la base" donde se opera internamente en decimal. Cabe mencionar que existe otro método muy utilizado denominado "el reloj". 1 1 1 1 5 7 3 3 2_8 + 2 3 4 5 1 0_8 4 9 12 8 4 2 -8 -8 -8 4 1 4 0 4 2_8 Se opera internamente en decimal Cuando la suma es mayor a 7 se resta la base 8, y se produce un acarreo positivo	Ejemplo: Se utiliza el mismo método que se emplea en la resta en el sistema decimal. Internamente se opera en decimal, cuando no se puede efectuar la resta le pide la base al numeral anterior y se produce un acarreo negativo. 10 8 pide la base 1 2 12 4 0 8 acarreo negativo 2 3 4 5 1 0_8 1 5 7 3 3 2_8 0 5 5 1 5 6_8 Se opera internamente en decimal. Cuando el numeral del minuendo es menor que el numeral del sustraendo le pide la base al numeral anterior y se suma con éste, luego se produce un acarreo negativo.

Tabla 16: Operaciones en sistemas octal.

Suma y Resta en Sistema Hexadecimal

SUMA HEXADECIMAL	RESTA HEXADECIMAL
Ejemplo: Aquí se muestra el Método de "restar la base" donde se opera internamente en decimal. Cabe mencionar que existe otro método muy utilizado denominado "el reloj". 1 1 1 1 1 9 7 A B 2_{16} + 2 7 C F 1 E_{16} 4 17 20 25 D 16 -16 -16 -16 -16 4 1 4 9 D 0_{16} Se opera internamente en decimal Cuando la suma es mayor a 15 se resta la base 16 y se produce un acarreo positivo	**Ejemplo:** Se utiliza el mismo método que se emplea en la resta en el sistema decimal. Internamente se opera en decimal, cuando no se puede efectuar la resta le pide la base al numeral anterior y se suma con éste, luego se produce un acarreo negativo. pide la base 1 23 E 17 acarreo negativo ~~2~~ ~~7~~ C ~~F~~ ~~1~~ E_{16} - 1 9 7 A B 2_{16} 0 E 5 4 6 C_{16} Se opera internamente en decimal. Cuando el numeral del minuendo es menor que el numeral del sustraendo le pide la base al numeral anterior y se suma con éste, luego se produce un acarreo negativo.

Tabla 17: operaciones en sistemas Hexadecimal

SISTEMAS DE CODIFICACIÓN

Códigos Binarios de 4 Bits de Números Decimales (BCD)

Los números binarios son los más apropiados para los cálculos internos en un Sistema Digital, pero la mayoría de la gente todavía prefiere trabajar con los números decimales.

El Código Decimal Binario (BCD, Binary Coded Decimal) es una forma de expresar cada uno de los dígitos decimales con un código binario.

- Como resultado, las interfaces externas de un sistema digital pueden leer o exhibir números decimales.
- Un conjunto de cadenas de n bits en que las diferentes cadenas de bits representan diferentes números u otras cosas se llama código.
- Una combinación particular de valores de n bits se llama palabra del código
- La facilidad de conversión entre los números en código 8421 y los números decimales es la principal ventaja.

- El BCD sigue siendo ampliamente utilizado para almacenar datos, en aritmética binaria o en electrónica. Los números se pueden mostrar fácilmente en visualizadores de siete segmentos enviando cada cuarteto BCD a un visualizador.
- La BIOS de una computadora personal almacena generalmente la fecha y la hora en formato del BCD, probablemente por razones históricas se evitó la necesidad de su conversión en ASCII.
- La ventaja del código BCD frente a la representación binaria clásica es que no hay límite para el tamaño de un número.
- Los números que se representan en formato binario están generalmente limitados por el número mayor que se pueda representar con 8, 16, 32 o 64 bits.
- Por el contrario utilizando BCD añadir un nuevo dígito sólo implica añadir una nueva secuencia de 4 bits.

El Código Decimal Binario significa (BCD) que cada dígito decimal, de 0 hasta 9, se representa mediante un código binario sin signo de 4 bits, del 0000 al 1001. Entonces codifica los números decimales dígito a dígito. En un byte caben dos dígitos en BDC.

Los códigos BCD más usados son

- **BCD Natural (8421):** se usa codificación ponderada, en la cual, se le dan a los bits, de izquierda a derecha, los pesos 8-4-2-1, respectivamente. Como estos pesos son precisamente los valores de posición en el sistema binario, un dígito decimal está codificado como su representación binaria, según se muestra en la tabla.

- **BCD Aiken (2421):** es un código autocomplementario y ponderado; o sea, la palabra código para el complemento a 9 de cualquier dígito puede obtenerse al complementar los bits individuales de la palabra código del dígito.

- **BCD Exceso 3 (no ponderado):** tiene una relación aritmética con el BDC; ya que la palabra código para cada dígito decimal es la correspondiente a la de BCD más 0011 (+3).

DECIMAL	BCD Natural	BCD Aiken	BCD Exceso 3
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	0100	0111
5	0101	1011	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

Tabla 18: Códigos BCD más utilizados. Elaboración propia.

Ejemplos:

Codificar en BCD Natural, en BCD Aiken y en BCD Xs-3 el siguiente número decimal: **3078592_{10}**	3 0 7 8 5 9 2 0011 0000 0111 1000 0101 1001 0010 BCD Natural
	3 0 7 8 5 9 2 0011 0000 1101 1110 1011 1111 0010 BCD Aiken
	3 0 7 8 5 9 2 0110 0011 1010 1011 1000 1100 0101 BCD XS-3

Decodificar a decimal los números expresados en BCD Natural, en BCD Aiken y en BCD Xs-3	0111 0100 0101 0000 0110 1001 0001 BCD Natural = 7450691_{10}
	1011 0010 1101 0011 1100 1110 0001 BCD Aiken = 5273681_{10}
	0111 1011 1010 1001 1000 1100 0101 BCD XS-3 = 4876592_{10}

Tabla 19 y 20: Elaboración propia.

Códigos de Caracteres o Alfanuméricos

Para la comunicación, no sólo se necesitan números, sino también letras y otros símbolos.

- En sentido estricto, los códigos alfanuméricos son códigos que representan números y caracteres alfabéticos (letras).
- Sin embargo, la mayoría de estos códigos también representan otros caracteres tales como símbolos y distintas instrucciones para la transferencia de información.
- En muchas aplicaciones, para completar la comunicación, son necesarios otros símbolos además de los números y las letras. Se necesitan espacios, puntos, dos puntos, punto y coma, signo de interrogación, etc.
- También se necesitan instrucciones para comunicar al sistema receptor qué hacer con la información.
- De este modo, con códigos con una longitud de seis bits, se pueden manejar números decimales, el alfabeto y otros 28 símbolos. El ASCII es el código alfanumérico más común. Otros ejemplos de códigos son: Videotext y EBCDIC.

ASCII 8: (American Standard Code for Information Interchange, Código Estándar Americano para el Intercambio de Información) es un código alfanumérico universalmente aceptado, que se usa en la mayoría de las computadoras y otros equipos electrónicos.

El código ASCII surgió originalmente como estandarización en 7 bits de varias codificaciones especiales y luego se extendió a una codificación de 8 bits. Se usa especialmente para la transmisión de datos y ha sido adoptado actualmente por la gran mayoría de los fabricantes de teclados. Utiliza únicamente 7 bits para la representación de los datos y están almacenados justificados por la derecha dentro de un campo estándar de 8 bits. El octavo bit se utiliza para la paridad de los anteriores.

La mayor parte de los teclados de computadora se estandarizan de acuerdo con el código ASCII, y cuando se pulsa una letra, un número o un comando de control, es el código ASCII el que se introduce en la computadora.

CÓMO SE UTILIZA LA TABLA ASCII8

Para codificar caracteres, números y caracteres especiales y de control a se utiliza la tabla que se muestra a continuación.

La misma cuenta con columnas: “Byte Cod y Char”, a los efectos de codificar caracteres ingresados por el teclado se selecciona el carácter a codificar y luego se transcribe el correspondiente valor numérico indicado por el número binario en la columna Byte.

Ejemplo:

Codificar en ASCII8.

Asiento	Cine	Precio	Fecha
8921	Hoyts	354	30 / 09 / 18

00111000	00111001	00110010	00110001	00100000			
8	9	2	1	space			
01001000	01101111	01111001	01110100	01110011	01000000		
H	o	y	t	s	space		
00110011	00110101	00110100	01000000				
3	5	4	space				
00110011	00110000	00101111	00110000	00111001	00101111	00110001	00111000
3	0	/	0	9	/	1	8

Tabla 21:::ASCII 8

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
00000000	0	Null	00100000	32	Spc	01000000	64	@	01100000	96	`
00000001	1	Start of heading	00100001	33	!	01000001	65	A	01100001	97	a
00000010	2	Start of text	00100010	34	"	01000010	66	B	01100010	98	b
00000011	3	End of text	00100011	35	#	01000011	67	C	01100011	99	c
00000100	4	End of transmit	00100100	36	$	01000100	68	D	01100100	100	d
00000101	5	Enquiry	00100101	37	%	01000101	69	E	01100101	101	e
00000110	6	Acknowledge	00100110	38	&	01000110	70	F	01100110	102	f
00000111	7	Audible bell	00100111	39	'	01000111	71	G	01100111	103	g
00001000	8	Backspace	00101000	40	(	01001000	72	H	01101000	104	h
00001001	9	Horizontal tab	00101001	41	)	01001001	73	I	01101001	105	i
00001010	10	Line feed	00101010	42	*	01001010	74	J	01101010	106	j
00001011	11	Vertical tab	00101011	43	+	01001011	75	K	01101011	107	k
00001100	12	Form Feed	00101100	44	,	01001100	76	L	01101100	108	l
00001101	13	Carriage return	00101101	45	-	01001101	77	M	01101101	109	m
00001110	14	Shift out	00101110	46	.	01001110	78	N	01101110	110	n
00001111	15	Shift in	00101111	47	/	01001111	79	O	01101111	111	o
00010000	16	Data link escape	00110000	48	0	01010000	80	P	01110000	112	p
00010001	17	Device control 1	00110001	49	1	01010001	81	Q	01110001	113	q
00010010	18	Device control 2	00110010	50	2	01010010	82	R	01110010	114	r
00010011	19	Device control 3	00110011	51	3	01010011	83	S	01110011	115	s
00010100	20	Device control 4	00110100	52	4	01010100	84	T	01110100	116	t
00010101	21	Neg. acknowledg	00110101	53	5	01010101	85	U	01110101	117	u
00010110	22	Synchronous idle	00110110	54	6	01010110	86	V	01110110	118	v
00010111	23	End trans. block	00110111	55	7	01010111	87	W	01110111	119	w
00011000	24	Cancel	00111000	56	8	01011000	88	X	01111000	120	x
00011001	25	End of medium	00111001	57	9	01011001	89	Y	01111001	121	y
00011010	26	Substitution	00111010	58	:	01011010	90	Z	01111010	122	z
00011011	27	Escape	00111011	59	;	01011011	91	[	01111011	123	{
00011100	28	File separator	00111100	60	<	01011100	92	\	01111100	124	|
00011101	29	Group separator	00111101	61	=	01011101	93	]	01111101	125	}
00011110	30	Record Separator	00111110	62	>	01011110	94	^	01111110	126	~
00011111	31	Unit separator	00111111	63	?	01011111	95	_	01111111	127	Del

Tabla 22: a) Tabla de código ASCII para 8 bits

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
10000000	128	Ç	10100000	160	á	11000000	192	+	11100000	224	Ó
10000001	129	ü	10100001	161	í	11000001	193	-	11100001	225	ß
10000010	130	é	10100010	162	ó	11000010	194	-	11100010	226	Ô
10000011	131	â	10100011	163	ú	11000011	195	+	11100011	227	Ò
10000100	132	ä	10100100	164	ñ	11000100	196	-	11100100	228	õ
10000101	133	à	10100101	165	Ñ	11000101	197	+	11100101	229	Õ
10000110	134	å	10100110	166	ª	11000110	198	ã	11100110	230	µ
10000111	135	ç	10100111	167	º	11000111	199	Ã	11100111	231	þ
10001000	136	ê	10101000	168	¿	11001000	200	+	11101000	232	Þ
10001001	137	ë	10101001	169	®	11001001	201	+	11101001	233	Ú
10001010	138	è	10101010	170	¬	11001010	202	-	11101010	234	Û
10001011	139	ï	10101011	171	½	11001011	203	-	11101011	235	Ù
10001100	140	î	10101100	172	¼	11001100	204	¦	11101100	236	ý
10001101	141	ì	10101101	173	¡	11001101	205	-	11101101	237	Ý
10001110	142	Ä	10101110	174	«	11001110	206	+	11101110	238	¯
10001111	143	Å	10101111	175	»	11001111	207	¤	11101111	239	´
10010000	144	É	10110000	176	_	11010000	208	ð	11110000	240	-
10010001	145	æ	10110001	177	_	11010001	209	Ð	11110001	241	±
10010010	146	Æ	10110010	178	_	11010010	210	Ê	11110010	242	‗
10010011	147	ô	10110011	179	¦	11010011	211	Ë	11110011	243	¾
10010100	148	ö	10110100	180	¦	11010100	212	È	11110100	244	¶
10010101	149	ò	10110101	181	Á	11010101	213	i	11110101	245	§
10010110	150	û	10110110	182	Â	11010110	214	Í	11110110	246	÷
10010111	151	ù	10110111	183	À	11010111	215	Î	11110111	247	¸
10011000	152	ÿ	10111000	184	©	11011000	216	Ï	11111000	248	°
10011001	153	Ö	10111001	185	¦	11011001	217	+	11111001	249	¨
10011010	154	Ü	10111010	186	¦	11011010	218	+	11111010	250	·
10011011	155	ø	10111011	187	+	11011011	219	_	11111011	251	¹
10011100	156	£	10111100	188	+	11011100	220	_	11111100	252	³
10011101	157	Ø	10111101	189	¢	11011101	221	¦	11111101	253	²
10011110	158	×	10111110	190	¥	11011110	222	Ì	11111110	254	■
10011111	159	ƒ	10111111	191	+	11011111	223	_	11111111	255	

Tabla 23 b) Tabla de código ASCII para 8 bits

BIBLIOGRAFÍA

Ginzburg, M.C. - La PC por dentro, Apéndice 1, 2ª edición -. Editorial: Biblioteca Técnica Superior. Biblioteca

Ginzburg, M.C. - Operación y programación de computadoras, Capítulo 3.Representación y aritmética y Apéndice 2.Sist. numéricos, 4ª edición. Editorial: Biblioteca Técnica Superior. Biblioteca Central Ubicación: 005.1 GIN o

Ginzburg M.C. - Introducción a las técnicas digitales con circuitos integrados, Capítulo 2.Codif. bin. núm. y car., y Apéndice 1.Enteros y punto flotante IEEE, 7ª edición. Editorial: Biblioteca Técnica Superior. Biblioteca Central Ubicación: 621.395 GIN i

Morris Mano - Arquitectura de computadores, Capítulo 1.Sistemas binarios. Editorial Prentice Hall - 3ª edición. Biblioteca Central Ubicación: 004.22 MAN a

Atribución-No Comercial-Sin Derivadas

Se permite descargar esta obra y compartirla, siempre y cuando no sea modificado y/o alterado su contenido, ni se comercialice. Referenciarlo de la siguiente manera:
Universidad Tecnológica Nacional Facultad Regional Córdoba (S/D). Material para la Tecnicatura Universitaria en Programación, modalidad virtual, Córdoba, Argentina.