

5

Codificando Información con 0s y 1s

Uno de los primeros pasos de abstracción que logramos desde niños son las primeras operaciones aritméticas. Cuando necesitamos conocer cuántos objetos, por ejemplo manzanas, tenemos en total, como se muestra en la Figura 5.1 no imaginamos las manzanas, sino que nos abstraemos en una *representación simbólica* (números) y aplicamos operaciones en estos símbolos para *calcular* el resultado.

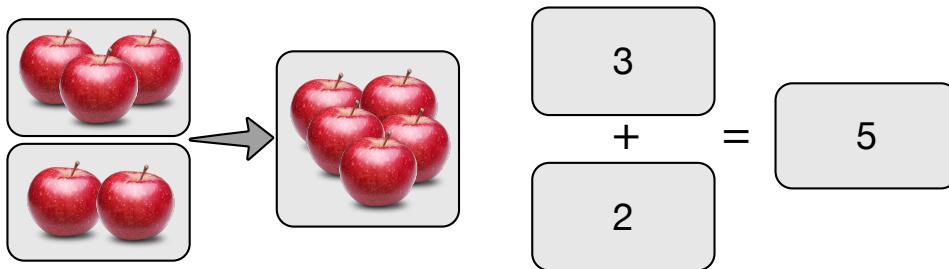


Figura 5.1: Abstracción

Otro ejemplo de nuestro uso cotidiano de la abstracción, son los mapas, una representación gráfica de mundo real que nos permite orientarnos y recorrer el mundo. Las nociones de puntos cardinales (Norte, Sur, Este y Oeste) más dibujos (líneas, rectángulos, etc.) permiten *codificar* el espacio real donde habitamos a un pedazo de papel o pantalla, de definitiva a un plano. Luego utilizando esa representación (información abstracta) podemos, sin haberlo hecho antes, llegar a un lugar de manera precisa, determinar cuánto tiempo o recursos necesitamos para conseguirlo, etc.

Otra circunstancia de debemos tener en cuenta son los límites

5.1. Sistemas Numéricos

Antes de comprender las diferentes formas de representar números naturales y racionales, es indispensable repasar las nociones de sistemas numéricos y sobre todo, la importancia de los sistemas posicionales para poder operar *algorítmicamente* sobre ellos.

¿Qué es un *Sistema Numeración*? , según la *Real Academia Española*[1], una *numeración* es “*Sistema para expresar de palabra o por escrito todos los números con una cantidad limitada de vocablos y de caracteres o guarismos*”. Dicho de otra manera y en la dirección de nuestro interés, un sistema de numeración nos permite expresar de manera *abstracta*, a través de símbolos y reglas, la noción de cantidad.

El producto de un sistema de numeración es el conjunto de *números* y la relación unívoca que establece con sus respectivas cantidades.

Como se puede observar en la Fig. 5.2

Sistemas Numéricos Posicionales

Si bien en nuestra vida cotidiana, en general, utilizamos sistemas posicionales, repasemos qué significa que un sistema numérico sea *posicional*. Los sistemas numéricos posicionales, como lo

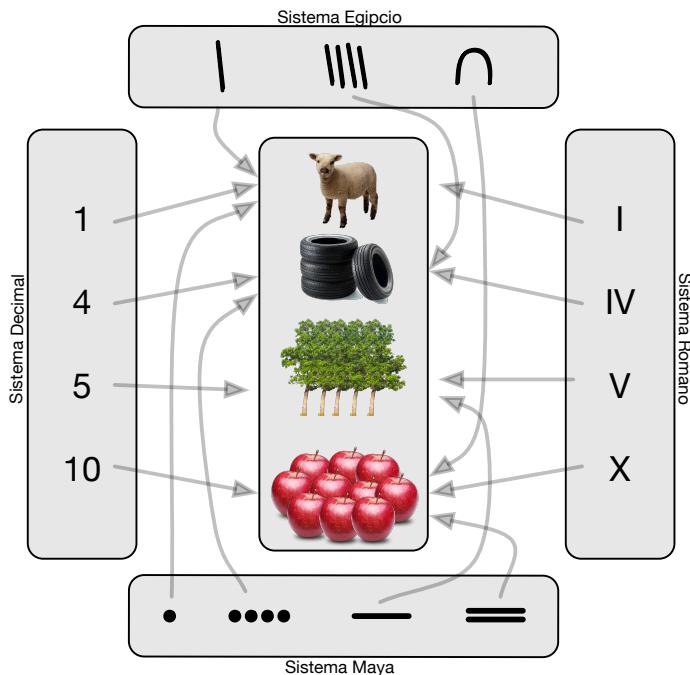


Figura 5.2: Relación entre números (símbolos) y su significado (cantidad)

expresa su nombre, son aquellos donde el *significado* de un número depende, no sólo del *símbolo*, sino también de su *posición* en el texto.

A la *cantidad de símbolos* que disponemos para representar los números se lo denomina *base*. Por ejemplo, en un sistema *decimal* los símbolos que podemos utilizar son *diez*: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

La característica particular de este tipo de sistemas numéricos, la que deriva su nombre, es su única regla elemental para construir los *números*: “Una vez que he agotado las combinaciones ordenadas de símbolos (hasta la posición utilizada), avanza una posición”. Si bien parece una regla simple, está cargada de conceptos que es necesario definir.

¿Qué es un *orden* en los símbolos?

Según su definición, es la manera de colocar los símbolos siguiendo un determinado criterio. El *criterio* en este caso es la importancia o significado de cada símbolo o composición de los mismos. Por ejemplo, si hablamos del Sistema Decimal que utilizamos con frecuencia, como se muestra en la Fig 5.2, tiene *mayor* significado el símbolo “5” que el número “4”; como así también en su composición el número “10” es mayor que el “1”. En ambas situaciones expresan una magnitud más grande de cosas que cuentan.

¿Qué es una *combinación* de símbolos?

Una combinación entre símbolos, en el contexto de sistemas numéricos, son todas las posibilidades o formas de mezclar dichos símbolos. En este caso particular, la combinación de dos símbolos en el Sistema Decimal serían: 00, 01, 02, 03, ..., 54, 55, 56, ..., 97, 98, 99. Cabe destacar que en el listado anterior, que si bien para una comprensión intuitiva están ordenados según el Sistema Decimal, una combinación no implica que deban seguir un orden como se muestra en la combinación de lápices de la Fig 5.3.

Una pregunta interesante y que utilizaremos más adelante, es conocer cuántas combinaciones podemos tener. Es decir, si tengo n cosas diferentes, cuántas combinaciones de m cosas diferentes puedo tener. Esta pregunta cobra más sentido cuando la hacemos dentro de un sistema numérico posicional, ya que me permite conocer *cuántas cosas puedo representar (contar)* si dispongo de n símbolos y m posiciones.

Para construir una respuesta a esta pregunta, razonemos siguiendo el ejemplo de los lápices completando la siguiente tabla, donde las filas representan diferentes colores, y las columnas, diferentes cantidades:

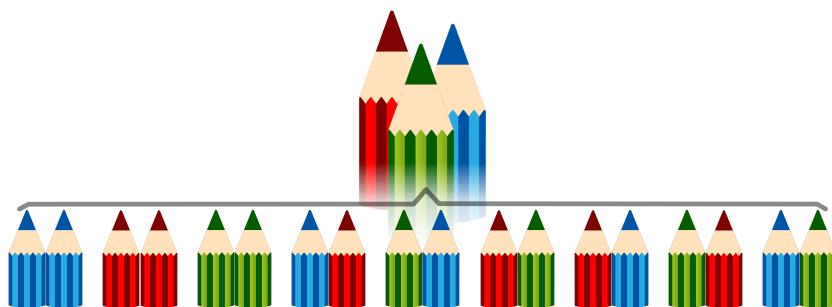


Figura 5.3: Combinación de 2 (dos) lápices de 3(tres) diferentes colores

Cant. colores	Cant. lápices				
	I (1)	II (2)	III (3)	IIII (5)	IIIIIIII (10)
1 color	1	2	3	5	10
2 colores	2	4	8	32	1024
3 colores	3	9	27	243	59049
4 colores	4	16	64	1024	1048576

Tabla 5.1: Cantidad de combinaciones de lápices, variando cantidad de lápices y colores

En la Tabla 5.1 podemos observar que si disponemos de un solo color, el único incremento en la cantidad de combinaciones es por agregando mayor cantidad de lápices. Por otro lado, si disponemos de un solo lápiz para combinar, la cantidad de posibilidades es igual a la cantidad de colores. Ahora bien, para calcular las combinaciones posibles cuando tenemos más de un color y más de un lápiz, podemos analizar de la siguiente manera, consideremos el caso en el cual disponemos de 3 (tres) colores. Si tenemos un lápiz, como lo analizamos anteriormente, tenemos tres posibilidades, los tres colores. Si agregamos un lápiz más, tenemos la combinación de los 3 colores que contábamos con cualquier posible color del segundo lápiz, esto es $3 * 3 = 9$ (nueve) combinaciones. Nuevamente, si además agregamos otro lápiz, podemos combinar las 9 existentes (con dos lápices) con los posibles colores del tercero $(3 * 3) * 3 = 27$. Así, en la medida que vamos incorporando lápices, calculamos las combinaciones multiplicando las obtenidas por la cantidad de colores, de manera genérica, si tenemos l lápices:

$$\underbrace{3 * 3 * \dots * 3}_{l \text{ veces}} = 3^l$$

Finalmente, si variamos la cantidad de colores c , las combinaciones se pueden calcular como c^l .

Ejercicio 1: Combinatoria

¿ Cuántos libros deberíamos conseguir (menor cantidad posible) si queremos mantener un estante de la biblioteca ordenado de manera diferente (sin repetición) cada día de mi vida ? Por ejemplo, con 3 libros, nos alcanza para ordenarlos de manera diferente durante 27 días.

Con este simple razonamiento, reemplazando *cantidad colores* por *cantidad símbolos(base)* y *cantidad de lápices* por *cantidad de posiciones* (tamaño), podemos calcular de la misma manera, cuántos números diferentes de ese tamaño puedo construir son esos símbolos. Dicho de otra manera, cuántas cosas diferentes puedo representar o contar.

Fórmula 1: Representación

$$\text{base}^{\text{tamaño}}$$

Por ejemplo, en nuestro Sistema Decimal (10 símbolos) con tamaño 3, podemos construir $10^3 = 1000$ números diferentes, del 000 al 999.

¿Qué entendemos por avanzar una posición?

Antes de analizar qué es *avanzar* en este contexto, repasemos la noción de *posición*. Para ello, tomemos como ejemplo el número en Sistema Decimal 999. Si por un momento nos olvidamos que estamos observando el número *novecientos noventa y nueve* en nuestro usual Sistema Decimal, estaríamos observando 3 *símbolos iguales* uno al lado del otro. Esto es disponemos de tres posiciones y en cada una de ellas está el símbolo 9. Ahora bien, en un sistema posicional, el valor que representa un símbolo, en este caso el 9, depende del *símbolo* y de su *posición*. Siguiendo con el ejemplo y de derecha a izquierda (sigue una regla precisa), el primer 9 representa el valor *nueve*, el segundo *noventa* y último *novecientos*.

Un ejemplo de sistemas NO posicionales, es el sistema *Romano*3 que es *semi-posicional*. En él, por ejemplo el número *XCIX* representa el *noventa y nueve*. Acudiendo al mismo ejercicio de observar los símbolos, el X que se encuentra en dos posiciones distintas, en la primera y en la última; en ambas posiciones su significado es el mismo: *diez*. En la primera ocasión para indicar el valor *diez* y en la segunda para indicar *diez* (menos que cien).

En los sistemas posicionales, el *valor* de cada símbolo depende de lugar donde se encuentra, siguiendo las reglas de conformación de números del sistema.

Retomando la pregunta, ¿ a qué nos referimos con *avanzar* ? . En los sistemas posicionales, y en particular los utilizados actualmente de *base constante* como el Sistema Decimal, asociamos la noción de avanzar una posición a la situación en la cual agotamos los símbolos (y sus combinaciones) hasta la posición en uso. En esta situación, para construir un nuevo número, cambiamos el símbolo de la siguiente posición por el de siguiente valor, *restaurando a nada* las posiciones inferiores a ella.

Por ejemplo, como se muestra en la Fig5.4 en el Sistema Decimal, suponiendo tres posiciones, para ir construyendo los números en orden, comenzando con el número 1, vamos cambiando la primera posición por símbolos del siguiente valor 2,3,4,...,9, hasta agotar los símbolos disponibles. Una vez agotados, cambiamos el símbolo de la siguiente posición, en este caso 0 por 1, volviendo las posiciones inferiores a 0.

Con este razonamiento, en cada avance de posición (cambio en el símbolo de la siguiente posición) significa que hemos agotado toda nuestra *base* (cantidad) de símbolos en las anteriores. En el caso del Sistema Decimal, *diez* símbolos. Con esto, la posición no permite asociar un significado al símbolo que esta contiene en relación al su valor. Siguiendo el ejemplo de la Fig5.4, cada vez que incrementamos el símbolo de la segunda posición, hemos pasado por diez de la primera, por ello, el valor que representa el símbolo en la *segunda posición* es *diez veces su valor*. De manera análoga, el de la *tercera posición cien veces*, etc.. Finalmente podemos concluir en la siguiente fórmula que expresa el valor de un símbolo *s* en la posición *i* (comenzando desde la posición cero) en un Sistema Posicional de *base b*:

Fórmula 2: Valor del símbolo *s* en la posición *i* en un sistema posicional de base *b*

$$s_i = s * \underbrace{b * b * \dots * b}_{i \text{ veces}} = s * b^i$$

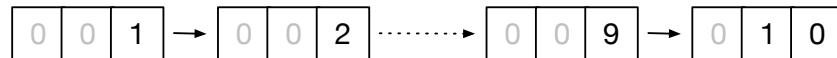


Figura 5.4: Construcción de números en Sistema Decimal, del 1 al 10.

Por ejemplo, si tenemos en cuenta el número 436 en el Sistema Decimal, su valor, *cuatrocientos treinta y seis* es la acumulación (suma) de los valores de cada símbolo en cada posición, como lo muestra la Figura 5.5

Un apreciación importante a tener en cuenta es que si bien en su uso habitual no se considera un tamaño (longitud) del número más allá que el utilizado, esto es con símbolos significativos, si uno extendiera la conformación del número serían con 0. Tomando el ejemplo del número 436, si lo escribiéramos con longitud 10, sería 0000000436. Siguiendo la fórmula 2, todas las posiciones desde la 3 hasta la 9, su valor es cero, coincidiendo su valor con el primero. Es por ello que no se frecuenta su escritura.

Además de su simpleza en cuanto a reglas de conformación de números, una diferencia fundamental que hizo prevalecer los sistemas posicionales por sobre los otros, son los mecanismos abstractos para realizar *operaciones*, por ejemplo, la *suma*, *resta*, *multiplicación* y *división*.

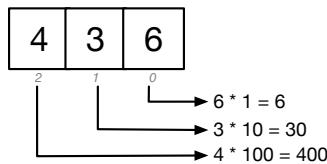


Figura 5.5: Descomposición del número 436 en Sistema Decimal.

Sistema Decimal

El Sistema Decimal, es un sistema numérico posicional que, como lo expresa su nombre, tiene *base* 10, es decir, dispone de *diez* símbolos para la conformación de sus números: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. En este sistema posicional, el valor de un número, como lo muestra la figura 5.5, es la suma de los valores de los símbolos según su posición i , es decir, multiplicado por 10^i .

Como lo mencionamos anteriormente, una gran ventaja que presentan los sistemas posicionales, son los mecanismos (algoritmos) para realizar operaciones. Comencemos por repasar la operación de *suma*(+), estos es, cuando tenemos dos cantidades expresadas como números en el sistema y queremos expresar con un número, el valor que representa si juntamos ambas cantidades.

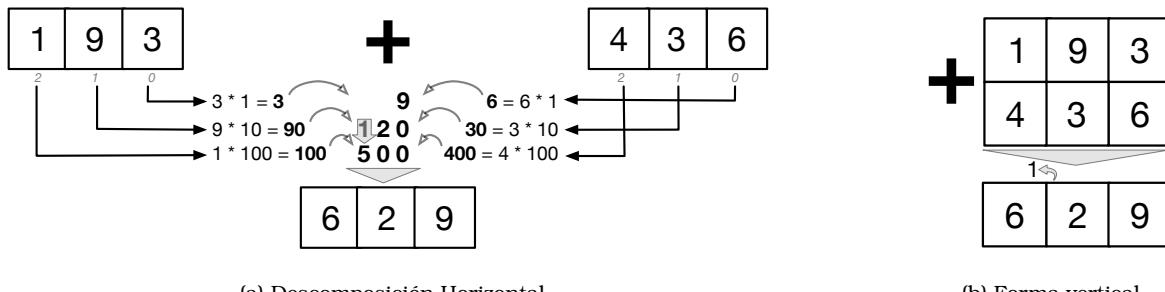


Figura 5.6: Suma en Sistema Decimal.

La Figura 5.6 (a) muestra un ejemplo de *suma* por descomposición de 2 números decimales, en particular el 193 y el 436. Dado que es un sistema posición, una mecanismo utilizado para resolver la suma, es la *descomposición*, esto es, partimos ambos números según sus posiciones y sumamos cada una de ellas de manera independiente. De manera análoga a cuando mencionamos el concepto de avanzar una posición, en caso de que la suma de las cantidades de una posición, excede la *base*, en el ejemplo $9+3$, avanzamos una posición, según sea necesario. Esta última situación es conocida usualmente con la frase '*me llevo uno*'. Cabe notar, que en la suma de dos símbolos, de cualquier base sólo puede, en caso de exceder, *incrementar en uno* la siguiente posición. Por ejemplo, en sistema decimal los dos símbolos más grandes que puedo sumar son $9+9=18$, es decir, lo que excede de la suma no puede ser más que uno. Para finalizar, este algoritmo, frecuentemente lo realizamos posicionando un número sobre el otro como se muestra en la Figura 5.6 (b).

Otra importante operación es la *multiplicación*(*), esto es obtener el número que *expresa cuántas veces es otro número*. Por ejemplo, si necesitamos saber *cuánto es tres veces seis*, multiplicamos $6*3$, cuyo resultado es 18. En la Figura 5.7(a), podemos observar el razonamiento de la multiplicación de $436*2$, es decir, *dos veces 436*. La forma intuitiva de razonar la multiplicación es la de una secuencia de sumas, es decir, dado un valor v (*multiplicando*) si queremos saber qué valor representa n *veces* (*multiplicador*) puede obtenerse sumando n veces el valor v .

Fórmula 3: Definición de la multiplicación en término de secuencia de sumas

$$v * n = \underbrace{v + v + \dots + v}_{n \text{ veces}}$$

En la medida que el *multiplicador* crece, como se muestra en la Figura 5.7 (b), un técnica para achicar este proceso es la de operar utilizando las *bases* de cada símbolo. Si tenemos que sumar

32 veces un número n , es igual que sumar por un lado 2 veces n y luego 30 veces n ; para finalizar sumamos las sumas parciales. ¿ Cuál es la ventaja de hacer esto ?.

Para responder a esta pregunta, primero debemos pensar qué sucede en términos de un número (visto como una secuencia de símbolos) cuando la cantidad de veces que queremos expresar coincide con su base. En términos del Sistema Decimal esta pregunta sería, ¿ qué sucede cuando multiplicamos por 10 ? . Como surge de la Fórmula 2, cada vez que multiplicamos por la base nos corremos un lugar hacia la izquierda (más significativos), a esta acción de avanzar una posición, la llamaremos en adelante como *shift* en este caso a izquierda.

Teniendo en cuenta este razonamiento y volviendo a la pregunta anterior, para expresar el valor de 30 veces un número n , en vez de sumar treinta veces, utilizando la multiplicación por la base, sumamos 3 (tres) veces n y luego avanzamos un lugar (multiplicamos por la base).

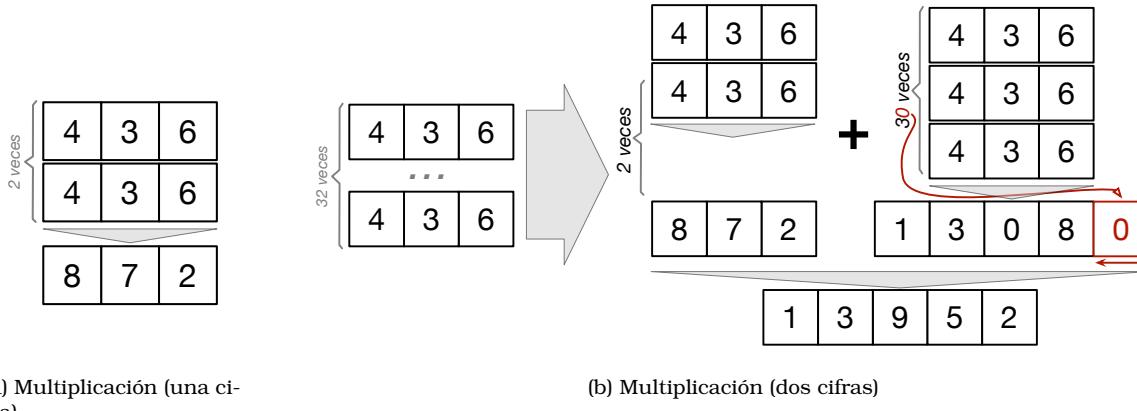


Figura 5.7: Multiplicación en Sistema Decimal.

Podemos considerar a la *resta*($-$) como la operación contraria a la *suma*. Esto es, dado un valor v queremos calcular el valor que nos queda luego de quitarle otro valor s , escrito como $v - s$. Con un razonamiento similar al de la suma podemos realizar esta operación, operando con un símbolo a la vez (descomposición en término de sus bases), calculamos el número que resulta de restar s a v , escrito como $v - s$. Una situación particular sucede cuando intentamos quitarle al valor de un símbolo uno más grande, siguiendo el ejemplo que se muestra en la Figura 5.8 cuando a 2 le restamos 3. En estos casos, y nuevamente en sentido contrario de la suma, en vez de '*llevarme uno*', '*pedimos uno*' al símbolo en la siguiente base. En el ejemplo, consideramos el 2 como 12, obteniendo como resultado de la resta $12 - 3 = 9$ para esa posición.

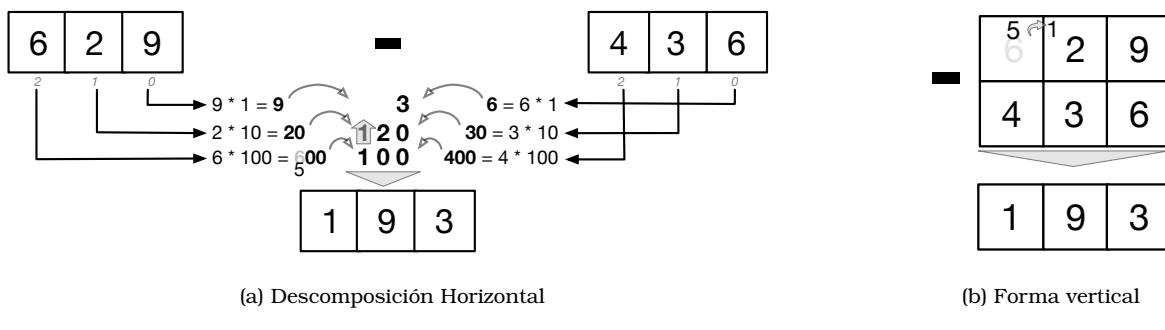


Figura 5.8: Resta en Sistema Decimal.

Una pregunta importante surge de la operación de *resta* si generalizamos la situación anterior, ¿ qué sucede si a un valor le quitamos uno más grande ? . Necesitamos *expresar valores negativos*. Si bien existen muchas notaciones utilizadas para tal fin, la más frecuente, es la de anteponer al número el signo $-$. Por ejemplo si a 436 le restamos 629, obtenemos como resultado -193 . Como puede observarse, para realizar tal operación, no podemos realizarlo siguiendo estrictamente el procedimiento anterior. Para poder aplicarlo, debemos invertir el orden, es decir, siempre restar un valor a uno más grande que él.

De manera similar a la resta, la *división* (\div) es la operación inversa de la multiplicación. Esta operación expresa cuántas veces un valor s (*divisor*) puede ser parte de otro v (*dividendo*), escrito como $v \div s$. Dicho en términos de operaciones, la división expresa cuántas veces podemos restar a v el valor s hasta agotarlo. Como podemos concluir de la definición anterior, la división tiene dos resultados importantes, el *cociente* y el *resto*. El *cociente* representa la respuesta a la *pregunta de cuántas veces*, mientras que el *resto*, como lo define su término, expresa lo que sobró del valor luego de la secuencia de restas. Podemos razonar el cociente y el resto de la división en términos de la multiplicación y suma como se define en la Fórmula 4

Fórmula 4: Definición de cociente y resto de la división en términos de la multiplicación y suma

$$n \div d = c, \text{ con resto } r \Leftrightarrow (c * d) + r = n$$

Si bien existen muchas técnicas algorítmicas para calcular una división, vamos razonar de manera análoga a la de la multiplicación. Como lo mencionamos anteriormente, la división es una *secuencia restas* del dividendo sobre el divisor el objetivo de calcular cuántas veces es uno del otro. Una vez que llegado al punto que no podemos restarlo más, lo que me queda es el *resto* y la cantidad de veces que puede restarlo es el *cociente*. Cuando aprendemos a dividir y con el objetivo de acelerar este proceso, aplicamos técnicas similares a las utilizadas para la multiplicación, descomponiendo al número en sus bases como lo muestra la Figura 5.9.

El ejemplo consiste en dividir a 436 por 12. Para ello, y de manera inversa a la multiplicación, comenzamos a operar por los símbolos más grandes, es decir, los números desde la izquierda hacia la derecha (mayor base). En el primer paso Figura 5.9(a), intentamos dividir el número más significativo de 436, este es 4, por el divisor 12. Como no podemos, en la base correspondiente del cociente, 2 en este caso, anotamos el número 0; es decir, el resultado no va a tener números significativos en la base dos, por lo tanto, seguro será un número menor que 99.

Dado el caso anterior, aún no hemos restado nada, tomamos el segundo número significativo en composición con el primero, en este caso 43 (Figura 5.9(b)). Ahora sí calculamos cuántas veces (la mayor cantidad) podemos restar 12 a 43. La respuesta para esta operación es 3, ya que tres veces 12 es 36, lo cual es menor que 43; además cuatro veces ya no podríamos porque es mayor que 43. Con este cálculo, anotamos en la base correspondiente del cociente el número 3 del cálculo anterior, y como resto, lo que sobró de dicha operación anotamos 7. Cotejando con la Fórmula ??, $(3 * 12) + 7 = 43$ es correcto.

Analizando en términos absolutos, según sus posiciones, acabamos de calcular que puedo quitarle (dividir) treinta veces ($3 * 10^1$) el valor doce a cuatrocientos treinta ($4 * 10^2 + 3 * 10^1$) y obteniendo como resto setenta ($7 * 10^1$). Es necesaria esta reflexión para notar que estamos operando en una base más alta 10^1 . Ahora debemos continuar con la siguiente base teniendo en cuenta el resto del paso anterior. Como se muestra en la Figura 5.9(c), tenemos que dividir 76, setenta del resto del paso anterior más seis, del dividendo original en la siguiente base con la cual operar, por el divisor 12. En esta caso, la respuesta es 6 y con resto 4 ya que $(6 * 12) + 4 = 76$. Anotamos 6 en la base correspondiente del cociente, y el resto de la operación es 4.

Finalmente, como agotamos el dividendo ya no quedan más bases de la descomposición por tratar, hemos finalizado la operación obteniendo como *resultado* el valor treinta y seis, y como *resto* el valor cuatro. Nuevamente para cotejarlo podemos verificar la fórmula:

$$\begin{array}{rcl} & \text{divisor} & \text{dividendo} \\ & \overbrace{(36 * 12)}^{\text{cociente}} + \overbrace{4}^{\text{resto}} & = \overbrace{436}^{\text{}} \end{array}$$

Para finalizar, tanto en la multiplicación como en la división, el cálculo de cantidad de veces es ayudado por la memorización de las *tablas de multiplicación* que se muestran en Tabla 5.2.

Sistema Binario

El Sistema Binario es un sistema numérico posicional que consta de *dos símbolos* para conformar sus números. Esta característica lo convierte en el sistema ideal para representar y manipular información en dispositivos tecnológicos que funcionen con electricidad, que como vimos en el Capítulo ??, posee dos posibles valores, *con/sin presencia de electricidad, encendido/apagado, ..., 1/0*.

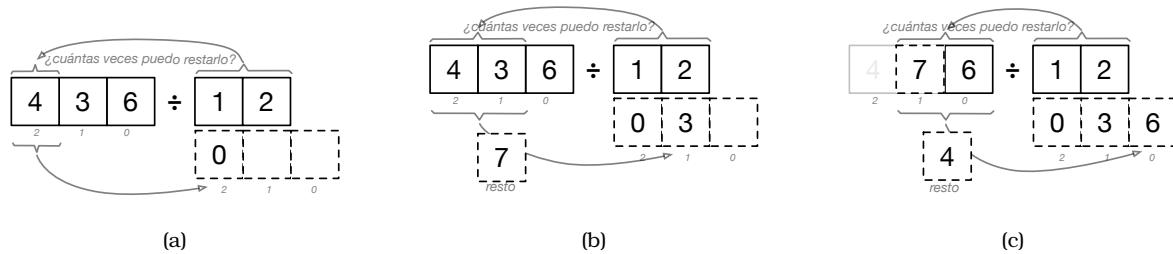


Figura 5.9: División en Sistema Decimal.

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	18	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

Tabla 5.2: Tablas de multiplicación del Sistema Decimal

Analicemos el Sistema Binario con un recorrido similar al que realizamos para el Sistema Decimal. Comenzando con la construcción de sus números y luego las operaciones entre los mismos.

Siguiendo la Fórmula 2 podemos graficar los valores de cada posición como se muestra en la Figura 5.10(a). En cada posición de un número binario o bien puede haber 0 o bien 1. Al igual que con el Sistema Decimal, el valor de un número es la suma de los valores de cada posición, así le valor del número binario 101011 es cuarenta y tres (Figura 5.10(b)).

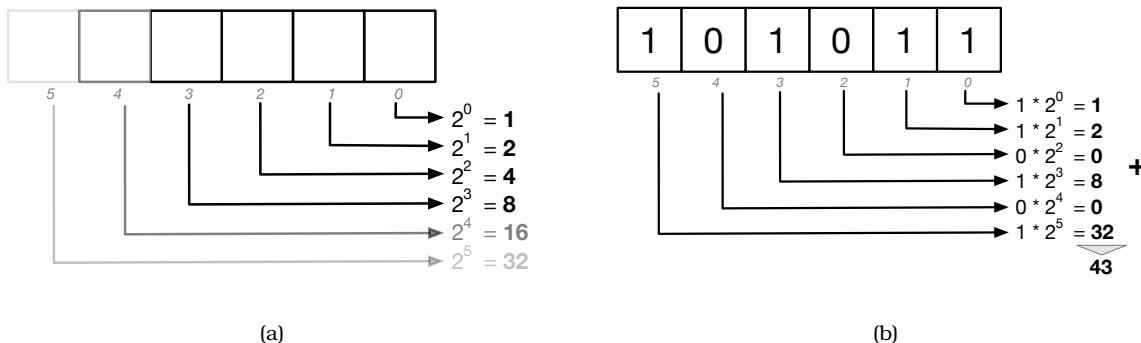


Figura 5.10: Descomposición de un número en Sistema Binario.

Como puede observarse, dado que la base del Sistema Binario es dos, la cantidad de números que puedo representar con el mismo tamaño es inferior que en el Sistema Decimal, por ejemplo, si disponemos de 5 posiciones, siguiendo la Fórmula 1 con el Sistema Decimal puedo representar 100,000 números, desde el 0 al 99,999, en cambio en el Sistema Binario, tan solo 32, del 0 al 31. Esta aparente desventaja con respecto al Sistema Decimal se ve compensada con la sencillez de las operaciones aritméticas entre números binarios. Para comenzar y comprender el por qué, observemos la Tabla 5.3, tabla de multiplicación del Sistema Binario.

Comencemos analizando las operaciones de *suma* y *resta* en el Sistema Binario. Como se trata de un sistema posicional, el razonamiento es el mismo que para el Sistema Decimal. Como lo muestran las Figuras 5.11(a,b), suma y resta respectivamente, en ambos casos podemos realizar las operaciones de a una posición a la vez. En el caso de la suma, si agotamos los símbolos de cada

	0	1
0	0	0
1	0	1

Tabla 5.3: Tablas de multiplicación del Sistema Binario.

posición, que por tener sólo 2 es algo frecuente, acarreamos uno a la siguiente posición. En caso de la resta, la única situación conflictiva es cuando a 0 le quiero quitar 1 en alguna posición. En esta situación hay que tener en cuenta el '*'pedir uno'* me convierte en 0 en 10, pero recordemos que este último *es un número binario* cuyo valor es 2; es decir, tener en cuenta que en binario $10 - 1 = 1$ (dos menos uno es uno).

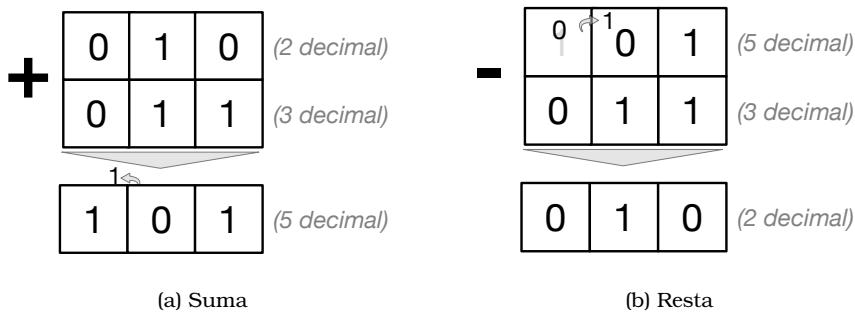


Figura 5.11: Operaciones con números binarios.

Otra situación a tener en cuenta es cuando sumamos más de dos o tres números binarios, es que podemos acarrear no a la posición inmediata siguiente, sino la siguiente a esta. Es decir, *llevarnos uno saltando una posición*. El siguiente ejercicio propone esta situación.

Ejercicio 2: Suma de números binarios

Realizar la siguiente suma de números binarios: $0111 + 0101 + 1011 + 1101$.

Como lo mencionamos anteriormente, las operaciones aritméticas con números binarios son más simples, no por la técnica en sí, sino porque al disponer solamente de dos símbolos, las posibilidades son binarias. Para comprender esto, pensemos en la multiplicación de números binarios. Esto es, dado un valor v , ahora expresado en términos de un número binario, queremos calcular cuál es el valor que representan n veces v .

Aplicando el mismo razonamiento que para números decimales, para calcular la n veces de un valor v , descomponemos a n en sus bases n_i y operamos con cada una de ellas sumando parcialmente n_i veces el valor v '*shiftreado*' (corrido) a la base i . Luego, sumamos todas las sumas parciales. Este proceso se simplifica cuando lo realizamos con el sistema binario, ya que como se puede anticipar de su tabla de multiplicación, Tabla 5.3 sólo tenemos dos opciones por cada suma parcial: o bien sumamos v o no.

La Figura 5.12 grafica un proceso para multiplicar $13 * 6$ en Sistema Binario $1101 * 0110$. Como se puede observar, a diferencia que para el Sistema Decimal, para cada base del *multiplicador*, o bien, si es 1 sumamos el multiplicando corrido a la correspondiente base, o si es 0 no.

En cuanto a la división en el Sistema Binario, siguiendo el mismo razonamiento aplicado para el Sistema Decimal, también notamos una simplificación. Dado que ahora, como se muestra en la Figura 5.13, la pregunta es *¿puedo restarlo?*, porque o bien, lo puedo restar *una vez* o no puedo restarlo.

Estos procedimientos para multiplicar y dividir en Sistema Binario son el fundamento detrás de los algoritmos que implementan los procesadores para realizar dichas operaciones.

Sistema Hexadecimal

El Sistema Hexadecimal, es un sistema numérico posicional, a diferencia del Sistema Decimal y Binario, su base es 16. Esto es, dispone de *dieciséis símbolos* para conformar sus números: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Como es de esperarse, esta propiedad, en contraste con el Sistema

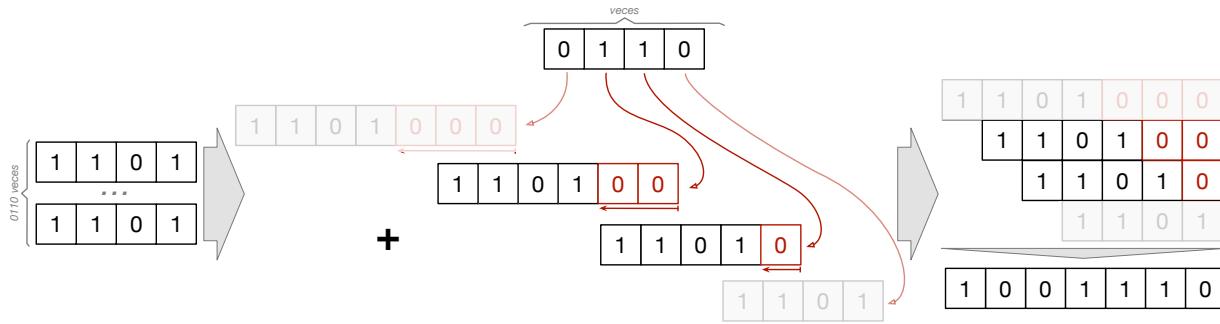


Figura 5.12: Multiplicación de números binarios.

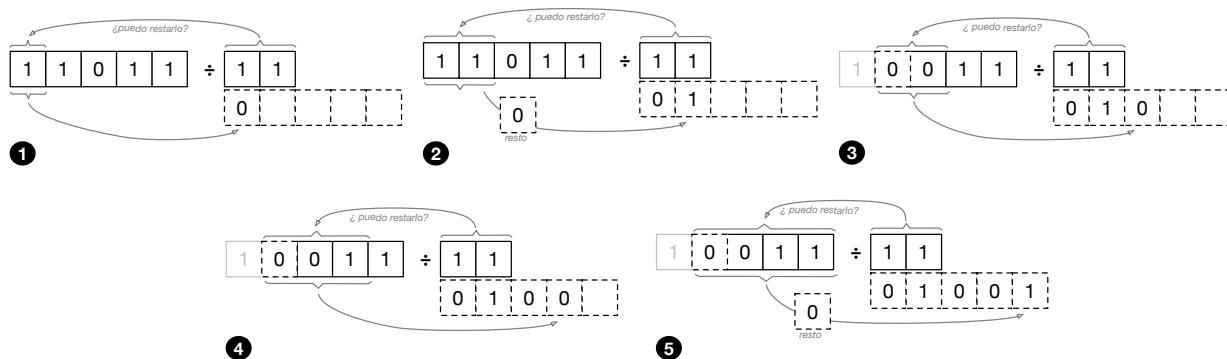


Figura 5.13: División de números binarios.

Binario, permite escribir números de manera más compacta, es decir, con el mismo tamaño podemos representar una mayor cantidad de números, incluso, más que con el Sistema Decimal. Por ejemplo, con tamaño 5 en Sistema Hexadecimal podemos conformar *un millón cuarenta y ocho mil quinientos setenta y seis* números, con el Decimal *cien mil* y con el Binario tan solo *treinta y dos*.

Precisamente, una de los usos más frecuentes del Sistema Hexadecimal, es el poder expresar información (generalmente codificada de manera binaria) de manera compacta. Un ejemplo de ellos son los mensajes en el que se informa una dirección de memoria como se muestra en la Figura 5.14. En él, las dos direcciones que figuran están escritas en Sistema Hexadecimal, si estuviese expresadas en Sistema Binario serían 110111000100110011111110101001 y 100011101111111111111111000 respectivamente.



Access violation at address 77133FA9 in module 'ntdll.dll'. Read of address 477FFFF8.

Figura 5.14: Mensaje de error en dirección de memoria expresada en Sistema Hexadecimal.

Conversiones entre Sistema Binario, Decimal y Hexadecimal

Anteriormente presentamos tres Sistemas Numéricos para expresar valores: El *Decimal* que es el más intuitivo y con el que lidiamos todos los días, El *Binario* que es el adecuado para poder expresar valores con instrumentos electrónicos y finalmente el *Hexadecimal* que nos permite expresar los valores de manera más compacta, es decir, utilizando una menor cantidad de posiciones. Por supuesto, existen tantos sistemas numéricos como se nos ocurra, por ejemplo el Sistema Numérico Octal, el cual tiene *ochos* símbolos: 0, 1, 2, 3, 4, 5, 6, 7.

Lo más importante, con independencia de qué sistema numérico utilicemos, es poder separar el concepto del *valor* de su *representación*. De hecho un mismo valor, por ejemplo el valor *ciento nueve* puede ser representado de tantas maneras como sistemas utilicemos: 109 (Decimal), 1101101 (Binario), 6D (Hexadecimal). Dentro de un sistema cada *número* representa un único *valor*. De esta

manera, también se establece una relación entre sus representaciones, es decir, podemos *convertir* un número de un sistema en otro.

Antes de analizar algunos métodos de conversión, la Figura 5.15 muestra cómo podemos poner en términos de una expresión aritmética (4) el valor de un número escrito en un sistema en base b (1). Por ejemplo el número 8743 escrito en base 10 lo podemos expresar como $((((8*10)+7)*10)+4)*10+3$, cuya evaluación es el valor *ochenta mil setecientos cuarenta y tres*. Otro ejemplo escrito en base 2 (Binario) $10011011 = (((((((((1 * 2) + 0) * 2) + 0) * 2) + 1) * 2) + 0) * 2) + 1) * 2 + 1$ cuya evaluación es el valor *ciento cincuenta y cinco*.

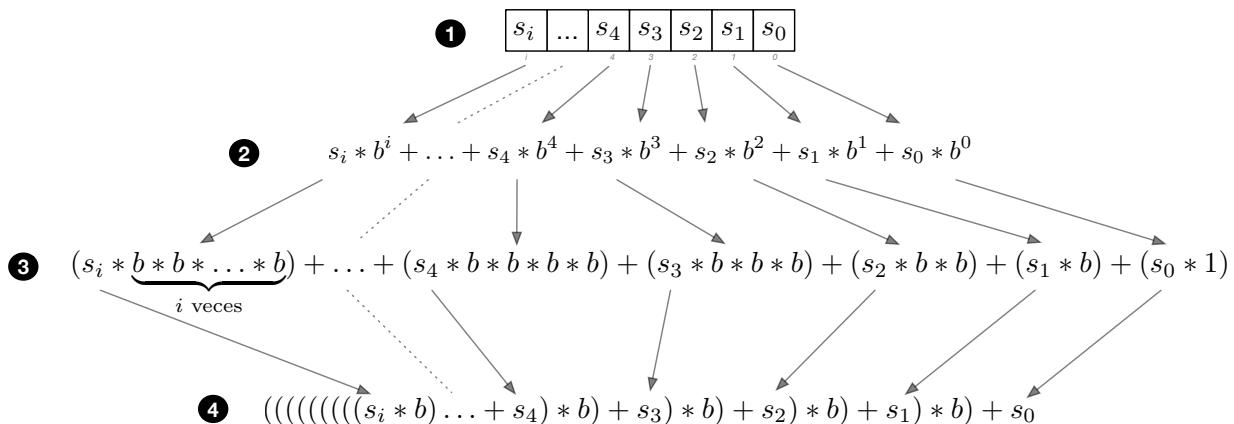


Figura 5.15: Composición anidada.

Comencemos por analizar la conversión entre los Sistemas Decimal y Binario. Una forma convertir un número binario en decimal, es calcular el valor que representa y luego, dado que estamos acostumbrados al Sistema Decimal, expresarlo en éste. Para ello podemos, o bien utilizar la expresión (2) de la Figura 5.15 recordando las potencias de dos: $1, 2, 4, 8, 16, 32, 64, \dots$ y luego sumar las multiplicaciones parciales; o bien utilizar la expresión anidada (4). En este último caso, debemos resolver desde adentro hacia afuera la expresión, es decir, comenzar por los símbolos más significativos (más a la izquierda) del número.

De la misma forma podemos convertir del Sistema Hexadecimal a Decimal, sólo que como la base es 16 debemos recordar sus bases en caso del resolverlo de la primera manera, o resolver la expresión con base 16 para la segunda forma. Por ejemplo, la expresión para convertir de hexadecimal a decimal el número $9AE0$, la expresión sería $((((9 * 16) + A) * 16) + E) * 16 + 0$, ahora con símbolos expresado en decimal $((((9 * 16) + 10) * 16) + 14) * 16 + 0 = 39648$.

Así como la expresión (4) de la Figura 5.15 nos permite *componer* un *valor numérico* a partir de sus bases mediante la multiplicación, podemos utilizar el mismo razonamiento para el proceso inverso. Esto es *descomponer* un valor en término de sus bases para un sistema numérico de base b , como se presenta en la Figura 5.16. Para ello utilizamos una secuencia de *divisiones* aplicando su definición de la Fórmula 4, el *cociente* por el *divisor* más el *resto* es igual al *dividendo*.

$$\begin{aligned}
 \text{valor} &= s_0 + (b * \text{cociente}) \\
 \text{valor} &= s_0 + (b * (s_1 + (b * \text{cociente}))) \\
 \text{valor} &= s_0 + (b * (s_1 + (b * (s_2 + (b * \text{cociente})))))) \\
 \text{valor} &= s_0 + (b * (s_1 + (b * (s_2 + \dots (b * (s_i + (b * 0))))))) \\
 \text{valor} &= s_0 + (b * (s_1 + (b * (s_2 + \dots (b * s_i)))))
 \end{aligned}$$

Figura 5.16: Descomposición anidada.

Con este método, podemos codificar un *valor* en un sistema numérico de base b . Como se muestra en la Figura 5.17, *ciento cincuenta y cinco* es codificado en el número binario correspondiente

10011011 (a) y en el número hexadecimal 9B (b). Cabe destacar que: i) Dado que es el sistema que utilizamos a diario, las operaciones las realizamos en el Sistema Decimal, es decir, expresamos el valor como 155 y la base como 2 y 16 respectivamente. ii) En caso en el que codificamos en el Sistema Binario, como TODOS su símbolos (0 y 1) son compartidos con el Decimal, no necesitamos reemplazar ninguno de ellos. En cambio, el Sistema Hexadecimal posee más símbolos que el decimal, es por ello que al realizar las operaciones en Sistema Decimal, luego debemos realizar los reemplazos correspondientes 10 → A, 11 → B, ..., 15 → F.

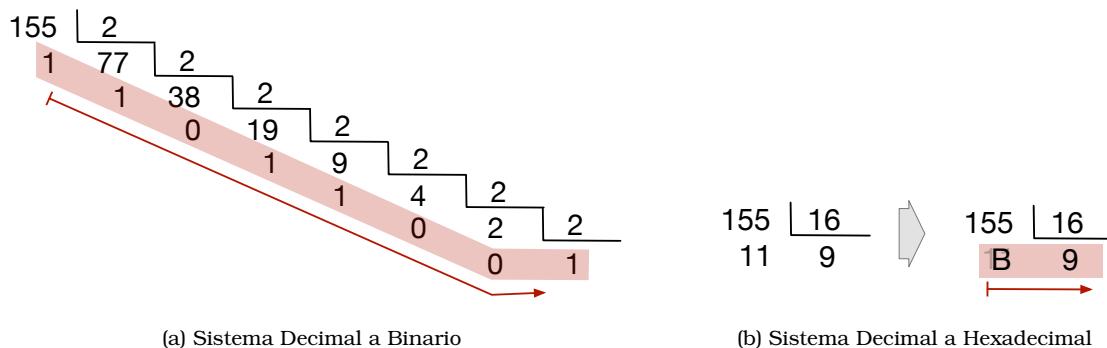


Figura 5.17: Conversión del número 155 del Sistema Decimal al Binario y al Hexadecimal.

La relación entre la cantidad de símbolos que disponen diferentes sistemas posicionales puede ayudarnos a realizar conversiones de manera más simple. Esto sucede cuando una cantidad es potencia de la otra. Para ayudar a comprender la estrategia, observemos la Tabla 5.4 que establece la correspondencia entre los números de los Sistemas Decimal, Binario, Hexadecimal y Octal.

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 5.4: Correspondencia de símbolos en Sistemas Decimal, Binario, Hexadecimal y Octal

Prestemos atención a las columnas del Sistema Binario en contraste con el Hexadecimal y nuevamente del Binario con el Octal. En ambas situaciones 16 y 8 son potencia de 2 respectivamente, ya que $2^4 = 16$ y $2^3 = 8$. Esta relación nos indica que necesitamos *exactamente* 4 posiciones en el Sistema Binario para una en el Hexadecimal y 3 para el Octal. En términos de la Fórmula 1 de representación de sistema numéricos, el Sistema Binario con 4 posiciones puede representar 16 números, cantidad que puede representar el Sistema Hexadecimal con tan solo una posición. Lo mismo sucede con el Sistema Octal, pero con 3 posiciones.

Teniendo en cuenta esta correspondencia *exacta* entre posiciones, podemos convertir entre dichos sistemas de manera directa posición a posición como se muestra en la Figura 5.18. Entre el Sistema Hexadecimal y Binario la relación es $4 \leftrightarrow 1$, esto es, a cada símbolo hexadecimal le corresponde cuatro posiciones en su representación binaria (a). En cuanto al Octal la relación es $3 \leftrightarrow 1$ (b).

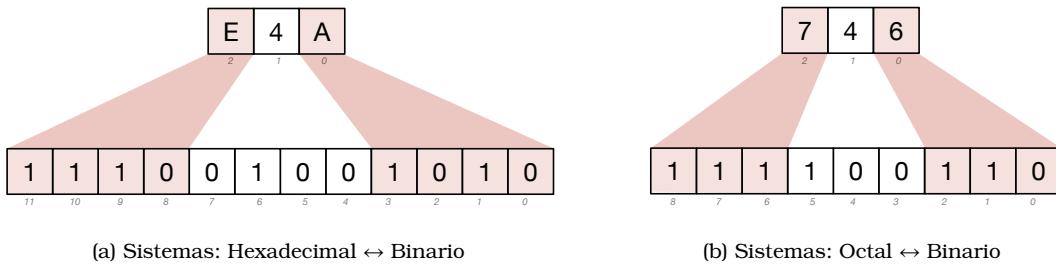


Figura 5.18: Conversión de sistemas numéricos con bases relacionadas por potencia.

5.2. Representaciones Finitas

En los sistemas numéricos la cantidad de números es *infinita*, sin embargo en la aplicación que haremos de ellos, tenemos necesariamente el límite que imponen los recursos finitos con los que contamos para su uso. Es decir, la cantidad de posiciones que disponemos para expresar un número es finita. Los recursos a los que referimos pueden ser, desde la cantidad de cuadrados que disponemos en un hoja para escribir los números hasta la cantidad de *bits* que dispone una computadora para almacenar información. Este límite implica una cota en la cantidad de números que podemos expresar, recordemos que podemos calcular esta cantidad con la Fórmula 1

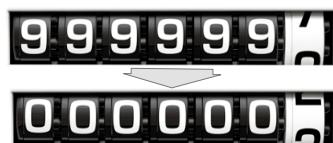


Figura 5.19: Ejemplo de representación finita de números: Odómetro.

Un ejemplo de representación finita de números que evidencia su límite son los *odómetros*, frecuentemente encontrados en cualquier tipo de vehículos terrestres. Estos dispositivos cuentan la cantidad de kilómetros recorridos desde el comienzo de la vida útil del vehículo, una vez que llega a su máximo número posible, comienzo nuevamente desde 0, como se muestra en la Figura 5.19.

Otros aspectos a tener en cuenta, además de las limitaciones en cantidad de números a representar, son las operaciones aritméticas en presencia de estos límites. Por ejemplo, en el Sistema Decimal con 3 posiciones, qué sucede si intentamos realizar la siguiente operación: $500 + 500$. El valor resultante no lo podemos expresar, esta situación la denominaremos como *fuerza de rango* u *overflow*. Otro aspecto importante a tener en cuenta, cuando consideramos la representación de valores enteros, es cómo anotamos los valores negativos, ya que si utilizamos una posición para distinguirlos, podemos afectar más aún la capacidad de representación.

Dado nuestro interés en analizar la representación de la información en sistemas electrónicos, en particular en las computadoras, exploraremos la representación finita de *números enteros y racionales*. Para ello, inicialmente consideraremos dichas representaciones en Sistema Decimal para una comprensión de los conceptos más intuitiva y luego en Sistema Binario. Este último con principal énfasis dado nuestro objeto de estudio.

5.3. Representaciones finitas de números enteros

Hasta aquí ya hemos visto los sistemas posicionales con diferentes bases, en particular Sistema Decimal y Binario. Ahora exploraremos ambos sistemas numéricos para la representación de valores enteros, es decir, tanto positivos como negativos, pero con una cantidad fija de posiciones.

Signo y Magnitud

Comencemos con un ejemplo simple, supongamos que disponemos de 3 posiciones. Contando con esta capacidad, con el Sistema Decimal podemos conformar 1000 números, desde 000 hasta el 999. Si sólo necesitáramos expresar valores positivos, podríamos representar 1000 valores, del *cero* al *novecientos noventa y nueve*. Ahora bien, si queremos expresar tanto valores positivos como

negativos, debemos analizar diferentes opciones. La primera y más simple, que además es la que utilizamos a diario es utilizar *una posición para el signo*. Con esta forma de representación, denominada *signo y magnitud* utilizamos la posición más significativa para representar el signo: *positivo* (+) o *negativo* (-).

Aplicando la representación de *signo y magnitud* para 3 posiciones, los números que podemos conformar son -00... -99 y +00... +99. En total podemos expresar 199 valores, del *noventa y nueve negativo* al *noventa y nueve positivo*; teniendo en cuenta que tenemos dos representaciones para el *valor cero*: -0 y +0.

Como podemos apreciar, la representación de números enteros con *signo y magnitud*, si bien muy fácil de leer e interpretar en la vida cotidiana, no es eficiente en cuanto a la cantidad de valores que podemos capturar. En el caso anterior, con 3 posiciones, de 1000 números posibles, podemos representar 199 valores. Esto es muy importante teniendo en cuenta nuestro propósito, donde los recursos disponibles deben ser utilizados de manera eficiente.

Representaciones con complemento

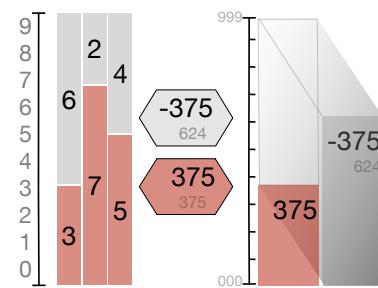
Para atacar esta situación y aprovechar al máximo la relación entre los números expresables y los valores representados, analicemos las representaciones por *complemento*. Comencemos por repasar la noción de *complemento*. Esta refiera a *lo que falta para llegar al todo*, como es en Teoría de Conjuntos, el complemento de un conjunto A son todos los elementos del universo que no están en A , es decir su diferencia.

De manera general, lo que proponen estas representaciones es utilizar un *rango* de números para representar los números positivos y otro para los negativos. De manera intuitiva y como lo muestra la Figura 5.20 (a), en la representación por complemento utilizamos la primera mitad de los números que podemos construir para representar valores positivos (cuánto llevamos recorrido) y, la segunda mitad, para representar los valores negativos (cuánto nos falta).

En la Figura 5.20 (b), se muestra la representación del valor *trecientos setenta y cinco* en su versión positiva y negativa. La versión positiva es representada por el número 375 como naturalmente lo escribimos. Para la representación negativa, complementamos cada símbolo (gráfico a la izquierda), obteniendo el número 624. En la misma figura, en el gráfico de la derecha podemos observar visualmente cómo una representación es complemento de la otra.



(a) Intuición de la representación por complemento.



(b) Complemento a la base -1.

Figura 5.20: Representaciones por complemento.

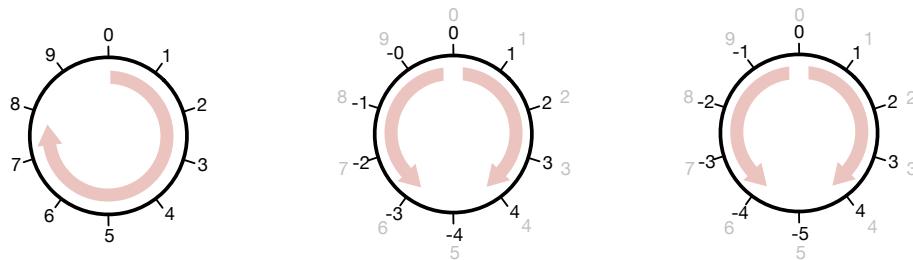
Complemento a la Base -1

La representación que utilizamos en la Figura 5.20(b) se denomina *complemento a la base -1*. La Fórmula 5 permite calcular la representación de cualquier valor negativo con esta representación, siempre y cuando el tamaño sea suficiente para poder hacerlo. Retomando el ejemplo de la figura, utilizando el sistema decimal (base 10) y con tamaño 3, el cálculo nos queda $-375 = (10^3 - 1) - 375 = 999 - 375 = 624$.

Fórmula 5: Complemento a la base -1

$$-v = (\text{base}^{\text{tamaño}} - 1) - v$$

De manera mnemotécnica, para expresar un valor negativo, restamos su representación positiva al *máximo número* que podemos construir con el *tamaño* disponible. De allí el nombre de la representación dado que este *máximo número* es uno menos que la base^{tamaño}.



(a) Representación Positivos

(b) Complemento a la Base -1

(c) Complemento a la base

Figura 5.21: Valor de un dígito decimal en diferentes representaciones

La Figura 5.21 (a,b) muestra la diferencia entre los valores representados con un dígito decimal con la representaciones positiva y complemento a la base -1. Como podemos observar, esta representación de valores negativos es más eficiente que la de signo y magnitud. Con la Fórmula 8 podemos calcular el rango de representación, por ejemplo con tamaño 3 en base decimal, podemos representar los valores que van desde el *cuatrocientos noventa y nueve* negativo, hasta dicho valor positivo: [-499, 499].

Fórmula 6: Rango de representación con complemento a la base -1

$$\left[-\frac{\text{base}^{\text{tamaño}}}{2} - 1 , +\frac{\text{base}^{\text{tamaño}}}{2} - 1 \right]$$

Si prestamos atención, podemos distinguir si el número representa un valor positivo o negativo teniendo en cuenta el símbolo más significativo. Si dicho símbolo, utilizando el Sistema Decimal, es *mayor o igual* que 5, entonces sabemos que representa un valor negativo. Invirtiendo el proceso de complementación identificamos cuál es. Por ejemplo, el número 5023 con Sistema Decimal y tamaño 4, representa el valor *cuatro mil novecientos setenta y seis negativo*; ya que $9999 - 5023 = 4976$.

Continuando con este mismo escenario, ¿qué valor representa el número 9999? Como comienza con un número mayor o igual que 5, 9 en este caso, sabemos que es negativo. Si realizamos el mismo cálculo que en el caso anterior para averiguar qué valor representa: $9999 - 9999 = 0$, obtenemos que es el *cero negativo*. Una de las *desventajas* de esta representación es que tenemos *dos números para el mismo valor*, dado que el valor *cero* es independiente del su signo: $0 = -0$.

Complemento a la Base

Para salvar la desventaja anterior, en vez de calcular el complemento a la base menos uno, y como lo define su nombre, la representación con *complemento a la base* (CB) expresa los valores negativos según la Fórmula 7. Como puede observarse, el concepto es el mismo que en la representación anterior, pero en complemento a la base, en vez, de tener una doble representación del *cero*, disponemos de un valor negativo más. Podemos calcular el rango de representación de esta representación con la Fórmula 8.

Fórmula 7: Complemento a la base

$$-v = (\text{base}^{\text{tamaño}}) - v$$

La Figura 5.21 muestra la comparación para un dígito en Sistema Decimal de los valores expresados en las tres representaciones abordadas. Para transformar un número negativo en esta representación, restamos a la cantidad de representación $\text{base}^{\text{tamaño}}$ el valor expresado en número positivo. Retomando el ejemplo de la representación anterior, para expresar el *trescientos setenta y cinco negativo*, calculamos $-375 = (10^3) - 375 = 1000 - 375 = 625$. Ahora, si nos hacemos la misma

pregunta con tamaño 4 en sistema decimal, ¿qué valor representa el número 9999?, realizando el cálculo $10000 - 9999 = 1$, el valor es el **uno negativo**.

Fórmula 8: Rango de representación con complemento a la base

$$\left[-\frac{\text{base}^{\text{tamaño}}}{2}, +\frac{\text{base}^{\text{tamaño}}}{2} - 1 \right] \quad \text{o} \quad \left[-\text{base}^{\text{tamaño}-1}, +\text{base}^{\text{tamaño}-1} - 1 \right]$$

Además de la eficiencia en la representación de valores, la representación con complemento a la base posee otra característica importante en relación al cálculo de operaciones, en particular para *restar* un valor a otro. Comencemos por un ejemplo simple con tamaño 2, supongamos que deseamos restar *veintiocho menos dieciséis*, en vez de realizar el procedimiento de la Figura 5.8, podemos realizar la *suma* entre 28 y la representación negativa de *dieciséis*:84. Si sumamos $28 + 84 = 112$, evidentemente *ciento doce* no es el resultado que esperábamos, pero como nuestro tamaño es 2, dado que tenemos una representación finita, el resultado de dicha operación en la representación finita es 12 (*doce*). Resultado correcto para la operación de *veintiocho menos dieciséis*.

Si analizamos la situación anterior de manera general restando a un valor v_1 un valor v_2 tenemos:

$$v_1 + (-v_2) \underset{\text{por def. CB}}{=} v_1 + ((\text{base}^{\text{tamaño}}) - v_2) \underset{\text{por repres. finita}}{=} v_1 - v_2$$

Como lo abordaremos más adelante, esta característica de la representación es de gran importancia para simplificar los circuitos de cómputo. No obstante debemos tener en cuenta las situaciones donde el resultado *está fuera de rango*. Por ejemplo, continuando con tamaño 2 en sistema decimal, si sumamos dos valores positivos *cuarenta y tres más dieciséis*: $43 + 16 = 59$, lo cual no es correcto, ya que el número 59, como lo analizamos antes, representa el valor *cuarenta y uno negativo*. Lo mismo sucede si sumamos dos valores negativos, por ejemplo *cuarenta y uno negativo más treinta y seis negativo*: $59 + 64 = 123$. Nuevamente, dado el tamaño 2, el resultado es 23 pero no es correcto, ya que representa el valor *veintitrés* en contraste con el valor correcto que es *setenta y siete negativo*. Si evaluamos los valores correctos de las operaciones anteriores *cincuenta y nueve y setenta y siete negativo* en el rango de representación de CB con tamaño 2, podremos contrastar que ambos **no son representables**.

Para el caso de complemento a la base con sistema de numeración decimal, las posibles situaciones para detectar fuera de rango en la suma (y resta) entre dos valores son las siguientes:

- positivo vs. negativo: cuando sumo un valor positivo con uno negativo, o vice versa, el resultado seguro es representable.
- positivo vs. positivo: **no** debería tener *acarreo final* y el resultado **no** debería tener el símbolo más significativo mayor o igual que 5, ya que si sumo dos valores positivos no puedo obtener uno negativo.
- negativo vs. negativo: debería tener *acarreo final* y el resultado debería tener el símbolo más significativo mayor o igual que 5, ya que si sumo dos valores negativos no puedo obtener uno positivo.

Dada su eficiencia y la simplificación que ofrece para operar entre los valores representados (suma y resta), el *Complemento a la Base* es la forma más utilizada para representar valores negativos en dispositivos. Como analizaremos en detalle en el Capítulo ??, esta representación es la más simple de utilizar para implementar circuitos electrónicos que realicen operaciones aritméticas. Para introducir su utilización, analicemos la representación de valores negativos con complemento a la base para el Sistema Binario, dado que este sistema numérico es el utilizado para representar información dispositivos electrónicos, las computadoras en particular.

Representación finita de Enteros con Complemento a la Base y Sistema de Numeración Binario

Como lo mencionamos anteriormente el Sistema Binario es un sistema numérico posicional con base 2, esto lo hace el más adecuado para utilizar en los dispositivos electrónicos, ya que concuerda con los dos estados posibles en los que puede encontrarse su *combustible*, la *electricidad*.

La Figura 5.22 (a) muestra la representación de valores positivos del Sistema Binario con 3 posiciones (tamaño tres), comenzando desde el 000 que representa el valor *cero* hasta el 111 que

representa el valor *siete*. Como puede calcularse, la cantidad de elementos que podemos representar es $2^3 = 8$, ocho valores. Ahora cuando deseamos representar valores positivos y negativos, es decir, valores enteros, debemos elegir alguna de las posibles representaciones que abordamos anteriormente; en nuestro caso por *complemento a la base*.

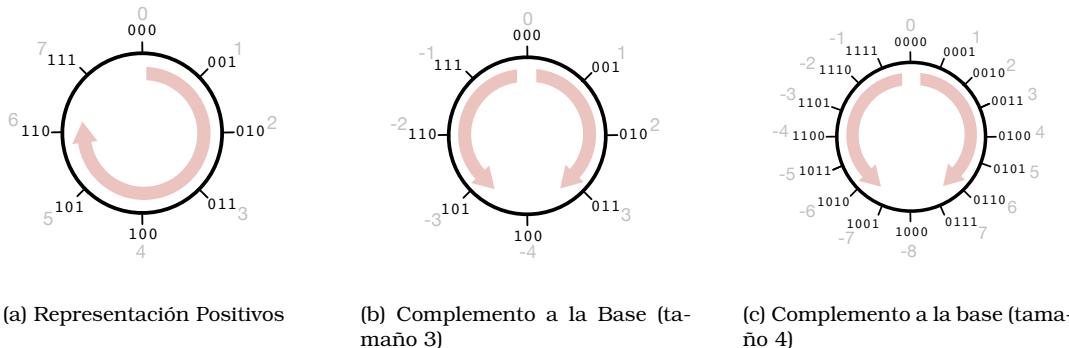


Figura 5.22: Sistema Binario y Complemento a la Base

Si bien el concepto y razonamiento sobre el complemento es siempre el mismo, con el Sistema Binario el proceso de *complementación* es mucho más simple, *cambiamos ceros por unos*. Realizando esta acción lo que obtenemos como resultado es el *complemento a la base -1* dado que es lo mismo que al máximo número (que en binario es la secuencia de unos del tamaño disponible) restarle el que deseamos complementar. Por ejemplo, si queremos complementar 0110101011, cambiando ceros por unos, o restando dicho número a 1111111111 obtendremos 1001010100.

Finalmente como la representación que necesitamos en Complemento a la Base, al proceso anterior *debemos sumarle uno*. Continuando con el ejemplo el complemento a la base de 0110101011 sería 1001010101. Una regla mnemotécnica que combina los dos pasos, *complementación + 1*, es la siguiente: *procesar el número desde la derecha hacia la izquierda sin hacer cambios hasta encontrar el primer 1, a partir de allí (exclusive), cambiar 0 por 1*.

Como se muestra en la Figura 5.22(b y c), otra simplificación que ofrece esta base con respecto a los valores negativos representados por complementación, es que los números negativos comienzan (símbolo más significativo) con 1 mientras que los positivos comienzan con 0.

Resumiendo, la combinación de Sistema Binario y Complemento a la Base es la más adecuada para representar valores enteros en dispositivos electrónicos ya que:

- Solo necesitamos dos símbolos 0 y 1 que aplican a los dos posibles estados de la electricidad.
- Es eficiente en cuanto al rango de representación: **no** tenemos doble cero (positivo y negativo).
- Es simple determinar el signo, coincide con el símbolo más significativo.
- El proceso de suma o resta es único, siempre se suma.

Estos dos últimos puntos tomarán relevancia cuando analicemos los circuitos y procesos que llevan adelante estas operaciones. En este sentido algo que debemos estudiar es cómo podemos averiguar cuándo el resultado la suma (o resta) está *fuera de rango*. Nuevamente analicemos los casos:

- positivo vs. negativo: cuando sumo un valor positivo con uno negativo, o vice versa, el resultado es representable. Esto es independiente de con qué base lo representamos.
- positivo vs. positivo: **no** debería tener *acarreo final* y el resultado **no** debería tener el símbolo más significativo igual a 1, ya que si sumo dos valores positivos no puedo obtener uno negativo. Esta situación podemos detectarla verificando que *los últimos dos acarreos sean 0*.
- negativo vs. negativo: debería tener *acarreo final* y el resultado debería tener el símbolo más significativo igual a 1, ya que si sumo dos valores negativos no puedo obtener uno positivo. De manera análoga a la situación anterior, para el que símbolo más significativo sea 1, dado que son dos números negativos, debo tener un acarreo hacia esa posición. Por lo tanto, cuando los dos números son negativos y los *últimos dos acarreos son 1* no tengo *fuera de rango*.

En resumen, en las operaciones de suma/resta, cuando los *últimos dos acarreos son iguales*, el resultado **no está fuera de rango**.

Extensión de signo

Como vimos anteriormente la multiplicación de valores enteros puede obtenerse mediante una serie de sumas. Analicemos este procedimiento para valores enteros en presencia de tamaños finitos y representación con Sistema Binario con complemento a la base.

El primer inconveniente que notamos es con el tamaño, el resultado de la multiplicación puede tener un tamaño muy superior (hasta el doble) que sus operandos. Por ejemplo, si tenemos tamaño 10, el rango de representación es $[-2^9, 2^9 - 1]$, es decir $[-512, 511]$; el mayor valor como producto es $-512 * -512 = 262144$, cuya representación binaria es 01000000000000000000 para el cual necesitamos tamaño 20, esto es, **el doble** del tamaño de sus operandos.

Ahora bien, salvada la aclaración respecto del tamaño, si multiplicamos dos valores positivos, podremos observar que el resultado es correcto. Ahora la pregunta es qué sucede si operamos con valores negativos (expresados con complemento a la base). Supongamos que deseamos multiplicar el valor *tres negativo* por *tres negativo*, es decir $101 * 101$ expresado en números binarios con complemento a la base. En un primer intento, que se muestra en la Figura 5.23(a), siguiendo el mismo procedimiento, podemos observar que el resultado es *incorrecto*. Este resultado, *veinticinco positivo* no es el esperado, sino que es el producto de considerar a 101 como *círculo*, es decir, como la representación de un número positivo: $5 * 5 = 25$. Esto sucede porque, como lo mencionamos anteriormente, el resultado de la multiplicación necesita para operar, el doble de tamaño. Por lo tanto, para expresar el valor *tres negativo* debemos **extender el signo** hasta completar el tamaño de operación, tal como se muestra en la Figura 5.23(b).

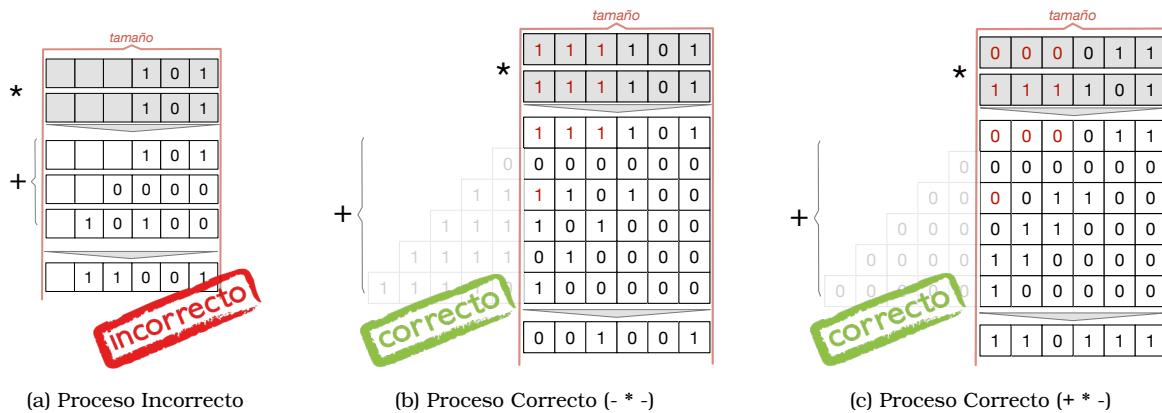


Figura 5.23: Multiplicación con Sistema Binario y Complemento a la Base (extensión de signo)

Teniendo en cuenta la extensión de signo, la multiplicación entre valores enteros representados con el Sistema Binario y Complemento a la Base funciona correctamente para cualquier combinación de signo $+ * +$, $- * -$, $+ * -$ y $- * +$, conservando correctamente el signo del resultado de dicha operación como se muestra en la Figura 5.23(c) para el caso $+ * -$.

Ejercicio 3: Multiplicación de valores enteros con Sistema Binario y Complemento a la Base

Realizar la multiplicación entre *tres negativo* por *tres positivo*.

5.4. Representación de Racionales

Frecuentemente necesitamos expresar valores que no necesariamente son enteros, por ejemplo la mitad de *siete*, o simplemente un valor que contiene una parte que no es entera, como el valor de π . Más allá de las diferentes clasificaciones de los estos valores: *Racionales*, *Irracionales*, *Algebraicos*, *Trascendentes*, etc.; lo que nos ocupa es cómo podemos representar esos valores y particularmente, cuál es forma de representación de los mismos utilizada en los dispositivos electrónicos.

Al igual que para valores enteros, comenzemos por analizar la representación de valores racionales utilizando un sistema posicional con base 10, es decir, nuestra forma habitual de representarlo. Por ejemplo, observando la Figura 5.24, supongamos que deseamos expresar el valor de *uno y medio* (manzanas), *cuatro sextos* (pizza restante), el valor de la hipotenusa para los valores *tres* y *cinco* (proyección de sombra de un árbol). Para ello utilizamos el siguiente número $1,5$, $0,66666\dots$ y $5,83$

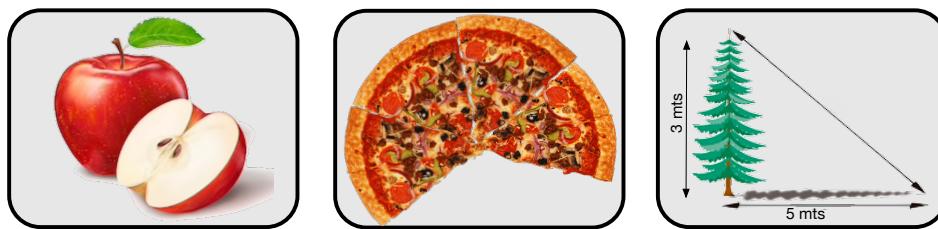


Figura 5.24: Ejemplos de valores racionales

respectivamente. Esto es, separamos la *parte entera* de la **parte decimal** con un símbolo, frecuentemente un *punto* “.” o una *coma* “,”. Ahora bien, qué es más, 0,5 o 0,05?, es decir, cómo se interpreta los símbolos que de la *parte decimal*.

Continuando con el mismo razonamiento de los sistemas posicionales, y tomando como ejemplo 436,572, en la Figura 5.25 puede observarse que la *parte decimal* no es más que la continuación de las posiciones pero con *exponente negativo de manera decreciente*; luego del punto (parte decimal) los exponentes son: $-1, -2, -3, -4, \dots$. Finalmente, al igual que para la parte entera, lo que representa es la suma del símbolo en esa posición elevada al exponente correspondiente.

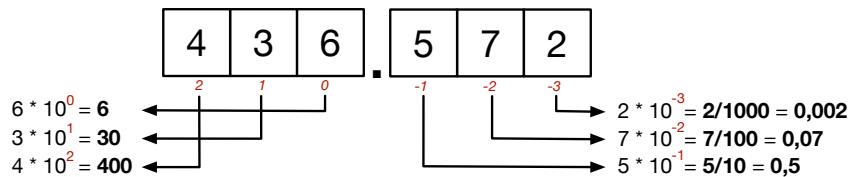


Figura 5.25: Ejemplo de valor racional en sistema posicional con base

Con este cálculo, en la medida que nos alejamos hacia la derecha del punto (*parte decimal*), estamos representando valores cada vez más “pequeños”. En la medida que los números crecen de tamaño, ya sea en la parte entera o decimal, necesitamos una forma más compacta de expresarlos. Por ejemplo, la distancia a la galaxia más próxima es de 2365200000000000000 kmts., o el peso aproximado de electrón es de 0,000000000911. Con el objetivo de mejorar su lectura haciendo más compacta su escritura, se propuso la denominada *notación científica*. Esta forma de escribir números racionales es muy frecuente en cualquier dispositivos cuando expresamos valores muy grandes o muy pequeños.

La notación científica consta de un **número** multiplicado por una **base** (10) elevada a un **exponente**. De esta manera, el número 2365200000000000000 se escribe como $2,3652 \times 10^{19}$ y 9,11 $\times 10^{-11}$ para el valor del peso aproximado de un electrón. Como podemos deducir de los ejemplos, en notación científica, el exponente nos indica la posición del primer símbolo (la parte entera), y a partir de él, van en forma decreciente hacia la derecha.



Figura 5.26: Ejemplo de Notación Científica

Si bien esta notación es la más utilizada a la hora de representar valores (extremos), no siempre su aplicación es una solución abreviada. Imagine que necesitamos expresar la suma de los dos valores anteriores, ¿cómo queda expresado?

Dado un valor v en Sistema Decimal expresado como $e.d$ (parte entera . parte decimal), para expresarlo con Notación Científica, asumimos que es $e.d \times 10^0$, es decir el primer símbolo a la izquierda

del punto se corresponde con el índice 0, 10^0 dado que estamos con el Sistema Decimal. Luego comenzamos a “mover” el punto hasta lograr posicionar el *primer símbolo distinto de 0* inmediatamente a la izquierda del punto. Si para lograrlo el punto lo mueve hacia la **izquierda**, vamos **incrementando** el exponente, si lo hacemos hacia la **derecha** lo vamos **decrementando**. Con este proceso, los 0 que quedan en los extremos se quitan ya que pasan a ser *no significativos*, obteniendo una versión abreviada del número. Las siguientes ecuaciones 5.1 muestra el procedimiento aplicado a los números de ejemplo anteriores.

$$\begin{aligned}
 2365200000000000000000 &= 2365200000000000000000 * 10^0 & 0,00000000911 &= 0,00000000911 * 10^0 \\
 &= 236520000000000000000,0 * 10^1 & &= 0,00000000911 * 10^{-1} \\
 &= 2365200000000000,0 * 10^2 & &= 0,0000000911 * 10^{-2} \\
 &= 2365200000000000,0 * 10^3 & &= 0,000000911 * 10^{-3} \\
 &= \dots & &= \dots \\
 &= 23652,0 * 10^{15} & &= 0,0911 * 10^{-9} \\
 &= 2365,2 * 10^{16} & &= 0,911 * 10^{-10} \\
 &= 236,52 * 10^{17} & &= 9,11 * 10^{-11} \\
 &= 23,652 * 10^{18} \\
 &= 2,3652 * 10^{19}
 \end{aligned} \tag{5.1}$$

Además de una representación más compacta, la Notación Científica ofrece ciertas ventajas a la hora de realizar operaciones aritméticas, sobre todo a la hora de multiplicar y dividir. A continuación detallamos el proceso para cada una de las operaciones abordadas:

- **Suma y Resta:** para estas operaciones, el primer paso es *alinear los exponentes*, es decir, modificamos corriendo el punto con su correspondiente modificación en el exponente hasta que éste concuerde con el exponente el otro operando. Luego realizamos la operación como si fuese un número entero. Finalmente *normalizamos* el resultado, es decir, volvemos a correr, si fuese necesario el punto hasta que quede justo a la derecha del primer símbolo del número.
- **Multiplicación y División:** en estas operaciones aritméticas es donde esta notación muestra sus ventajas, para realizar una *multiplicación*, simplemente *multiplico* ambos números (sin importar la alineación de sus exponentes), luego el exponente del resultado es la *suma de los exponentes* de cada operando. Finalmente *normalizo* el número. En caso de la *división* el procedimiento es análogo, *dividiendo los números*, pero esta vez el exponente del resultado es la *resta de los exponentes* (dividendo menos divisor).

En resumen la Notación Científica presenta una forma eficiente en tamaño para representar valores racionales. Su idea es contar con un número *normalizado* donde su símbolo más significativo (como parte entera) se encuentra en la posición de base 10 que indica el exponente. Con ello, minimizamos la cantidad de símbolos 0 significativos obteniendo un escritura abreviada del mismo. Además, utilizando propiedades algebraicas, podemos realizar ciertas operaciones de manera más simple. A continuación abordaremos esta idea aplicada para la representación de valores racionales en dispositivos electrónicos, objeto de nuestro estudio.

Representación finita de Racionales con Sistema Binario

De manera similar que para la representación de valores enteros en dispositivos electrónicos, la base que utilizaremos será *dos*, es decir, el Sistema Binario. Antes de detallar la representación con *punto flotante* que es la que utilizan los dispositivos, analicemos algunas alternativas para la representación de valores racionales. Nuevamente, el mayor obstáculo que afrontamos es la *finitud*, y por cierto escasa, del tamaño que disponemos para expresarlos.

Punto Fijo

Comencemos por analizar una representación de *punto fijo*, como una extensión de la representación que utilizamos para representar valores enteros. La idea de punto fijo radica en dividir nuestro espacio de representación en *parte entera* y *parte decimal*. Como se muestra en la Figura 5.27, el número 01011,01010 representa el valor *once con tres mil ciento veinticinco milésimas*

(11,3125 en Sistema Decimal). Aquí estamos utilizando un tamaño total de diez posiciones, donde destinamos cinco para la parte entera y cinco para la decimal.

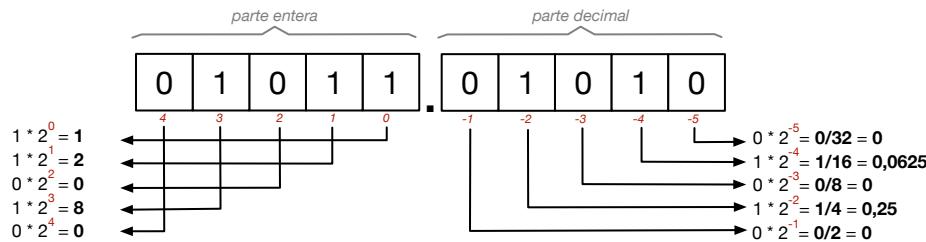


Figura 5.27: Ejemplo de representación binaria con punto fijo.

Otro punto a tener en cuenta es la representación de negativos dentro de los racionales. Si combinamos la idea anterior con Complemento a la Base (el utilizado para la representación de enteros), por ejemplo el rango de representación para 10 posiciones (5 parte entera + 5 parte decimal) sería desde el número 10000,11111 (-16,96875 en decimal) al 01111,11111 (15,96875 en decimal). Esta situación nos hace reflexionar algunas desventajas de esta forma de representación. Los diferentes escenarios y usos de valores racionales pueden cambiar, en ocasiones necesitamos mayor representación entera y otras mayor precisión decimal. Al trabajar con punto fijo escoger de manera predeterminada los tamaños para la parte entera y decimal va en detrimento de ello.

Errores de representación

Antes de continuar con la representación de Punto Flotante, exploremos el siguiente caso de representación, supongamos que deseamos expresar el número 0,30 (con representación decimal). La siguiente tabla muestra la representación para diferentes tamaños de la parte de decimal, y el error de representación correspondiente.

Tamaño Decimal	Número Binario	Conversión a Decimal	Error de representación
1	0.0	0.5	0.2
2	0.01	0.25	0.05
3	0.010	0.25	0.05
4	0.0101	0.3125	0.0125
5	0.01010	0.3125	0.0125
6	0.010011	0.296875	0.003125
7	0.0100110	0.296875	0.003125
8	0.01001101	0.30078125	0.00078125

Tabla 5.5: Errores de representación de valores racionales con Sistema Binario

Esta situación es intrínseca de la representación de valores que no se pueden expresar *exactamente* con el Sistema de Numeración Binario. Si bien no abordaremos este tema en profundidad, es necesario saber que existe y que el máximo error (diferencia entre el valor que deseamos y el representado) depende de la cantidad de posiciones destinadas a la parte decimal, en particular y como puede observarse en la Tabla 5.5, el error **nunca** supera el valor de $2^{-\text{tamaño parte decimal}}$.

Punto Flotante

La representación de valores racionales mediante la idea de Punto Flotantes es similar a la de Notación Científica. Tenemos una parte para expresar el valor de manera relativa que la denominamos **mantisa**, una parte para definir el **exponente** que ubica los símbolos de la mantisa con la correspondiente base y finalmente una posición para determinar el **signo**. La Figura 5.28 refleja una distribución de los recursos para cada una de las partes mencionadas. A partir de esta primera impresión surgen algunas preguntas: ¿cuánto espacio destinamos a la mantisa y qué mejoramos al incrementarla? ¿cuánto espacio destinamos al exponente y qué mejoramos al incrementarlo? Dado que podemos necesitar exponentes negativos ¿cómo los representamos?

Para comenzar a analizar estas preguntas comencemos con un ejemplo de configuración para tamaño diez, como lo propone la Figura 5.28. Una posición, por convención la primera, la destinaremos al signo. En coherencia con la representaciones vista para enteros, el símbolo 1 representará

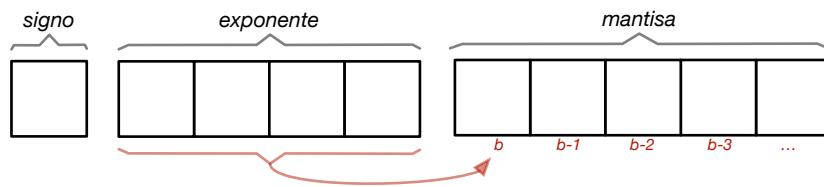


Figura 5.28: Estructura de representación con Punto Flotante.

el signo –, es decir, que es un valor **negativo**, y 0, que el valor es **positivo**. Otra convención que asumiremos es que, si bien podría ser de otra manera, el *exponente indica la base de la primera posición de la mantisa* y a partir de ella, hacia la derecha se va decrementando. Finalizando con una posible configuración de ejemplo, de las nueve restantes posiciones, destinamos cinco posiciones a la mantisa y cuatro al exponente.

Supongamos que deseamos representar el valor *dos y medio negativo*, la siguiente tabla muestra diferentes formas de expresarlo, con la configuración y convenciones que acordamos hasta este momento:

Signo	Mantisa	Exponente
1	101000	0001
1	010100	0010
1	001010	0011
1	000101	0100

Tabla 5.6: Diferentes representaciones de 3,5 con Punto Flotante (1+5+4)

No es difícil imaginar que en la medida que aumentamos el tamaño de la mantisa, para este mismo valor vamos a tener una gran cantidad de formas diferentes para representarlo. Para atacar este inconveniente, el de tener más de una representación para un mismo valor, vamos a acordar por convención que el mismo esté **normalizado**. Consideraremos un número *normalizado cuando su primer símbolo más significativos se encuentra en la primera posición* (contando de izquierda a derecha). Con esta definición, la representación correcta para el valor es *dos y medio negativo* la primera de la tabla, es decir, 11010000001.

De la observación de la Tabla 5.6 podemos deducir un *procedimiento de normalización*, para lograr posicionar el primer símbolo más significativo en la primera posición, en la medida que movemos la *mantisa hacia la derecha incrementamos el exponente* y si lo hacemos hacia la *izquierda decrementamos*.

Continuando con la misma configuración, representemos el valor *un cuarto* (0,25 en decimal). Para expresarlo de manera correcta (normalizado) necesitamos representar el valor del exponente negativo, en particular -2, dado que es la base para el primer símbolo más significativo ($1 \cdot 2^{-2} = 0,25$). De esta situación surge la pregunta ¿ cómo representamos los valores de los exponentes ?. Una primera opción sería utilizar la misma representación que propusimos para valores enteros, es decir, Complemento a la Base. Sin embargo, dadas las operaciones más frecuentes de normalización y la identificación del máximo exponente con *unos*, la representación del valor del exponente es con *exceso o polarizante*.

La idea de representación de con *exceso o polarizante* es la de partir el espacio (rango) en dos partes a partir de un valor, de allí el nombre de polarizante, ya que a partir de él, hacia la *izquierda se decrementan* los valores hasta llegar a valor mínimo y hacia la *derecha se incrementan* hasta llegar al valor máximo.

Las Figuras 5.29(a,b) exhiben la diferencia de representación entre Complemento a la Base y Exceso para tamaño 4. Una configuración que lleva esta última representación por exceso, es determinar qué valor es el *polarizante o exceso*. Por convención, y dado que estamos con base binaria se utiliza $2^{\text{tamaño}-1} - 1$. De allí que el número 0111 (7 en decimal) representa el valor *cero* y desde allí en la medida que incrementamos el número vamos incrementando el valor representado hasta llegar al valor máximo *ocho* en este caso, expresado por el número 1111. De manera dual, en la medida que decrementamos el número 0111 vamos obteniendo valores más chicos hasta llegar al mínimo *siete negativo* expresado por el número 0000. Esta coherencia con su objetivo convierte la *representación por exceso* a la más adecuada y utilizada para expresar el valor del exponente en la representación de valores racionales con *Punto Flotante*.

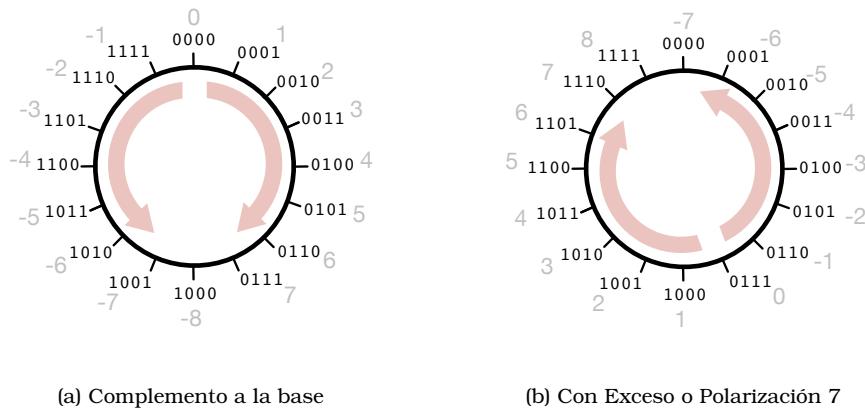


Figura 5.29: Diferencia de representaciones Complemento a la Base vs. Exceso (tamaño 4)

Para calcular el valor del exponente dada su representación, calculamos su valor binario y le restamos el polarizante. Por ejemplo, para la configuración de tamaño 4 y polarizante 7, si deseamos conocer que valor de exponente representa el número 1010, lo convertimos a decimal 9 y le restamos el polarizante, en este caso $9 - 7 = 2$. Esto es, el número 1010 como exponente, significa que la base del primer símbolo significativos de la mantisa (más a la izquierda) es **dos**. Si el número del exponente fuese 0011, 3 en decimal, le restamos el polarizante $3 - 7 = -4$, es decir, la base del primer símbolo sería **cuatro negativo**.

Analicemos ahora el rango de valores que disponemos con esta representación. Para nuestro ejemplo 1,4,5 (tamaño de signo, exponente y mantisa respectivamente), el valor más pequeño que podemos representar es 1111111111, es decir, signo negativo y exponente 8, con ello 111111 corresponde al número decimal 496 ($256 + 128 + 64 + 32 + 16$), y el mayor rango sería $[-496, 496]$. Si contrastamos este rango con el de punto fijo, aún resignando a tres las posiciones para la parte decimal la diferencia es importante, y crece exponencialmente en la medida que disponemos de mayores recursos (tamaño).

Una pregunta que surge naturalmente ante esta situación es ¿dónde está la magia?. La respuesta es que no es mágica esta forma de representación, sino que, a diferencia de la representación de punto fijo, su **precisión no es homogénea**. Es decir, en la medida que trabajamos con exponentes más altos perdemos precisión a cambio de mayor representación. Por ejemplo, con la configuración 1,4,5, pensemos qué pasaría si a 0111111110 (480 en decimal) deseamos sumarle **cuatro**; pues al normalizar los exponentes para poder realizar la suma perdemos el valor significativo de **cuatro**. Esto es, cuando estamos trabajando con valores grandes perdemos la posibilidad de operar con valores pequeños. La Figura 5.30 expresa esta situación de manera gráfica, como se puede observar en ella, a medida que el exponente crece (color naranja) los valores representables (líneas de la regla) se vuelve más dispersos. Además la figura permite comprender la relación del tamaño de la exponente y mantisa con respecto a los valores representados, a **mayor tamaño de exponente** ganamos **mayor rango de representación**, a **mayor tamaño de mantisa** incrementamos la ventana de trabajo, es decir, la **precisión**.



Figura 5.30: Representación con Punto Flotante.

Si bien la situación descrita es una desventaja, en general, nunca sucede en su utilización. Cuando se trabaja con valores muy grandes, los pequeños valores son despreciables y vice versa. En lo ejemplo vistos, si trabajamos con distancias planetarias, poco importa un par de metros, o cuando trabajamos en un escenario de valores pequeños como peso de electrones, no se opera con valores astronómicos. Con lo cual, la representación de Punto Flotante es la más adecuada y

utilizada para expresar valores racionales.

Estándar IEEE 754 para representación de punto flotante

Si bien acordamos que la representación con Punto Flotante es la más adecuada, como vimos anteriormente para un mismo tamaño podemos tener muchas variantes, escogiendo diferentes porciones para la mantisa y exponente, ya que el tamaño del signo no varía. Esto significa que dado un número, por ejemplo 010111011 si no sabemos de antemano, cuál es el tamaño del exponente y en consecuencia el de la mantisa, y la configuración del exceso, que también podría variar, sólo podríamos asegurar que el número anterior se corresponde con un valor positivo. Es por ello que en 1985, y con sucesivas revisiones, el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) estableció dos estándares para la representación con punto flotante de valores racionales [2].

Dado su aplicación a la aritmética computacional (base 2), propone tres configuraciones básicas para 32 posiciones (Simple), 64 posiciones (Doble) y 128 posiciones (cuádruple). La siguiente Tabla 5.7 enumera la distribución para cada una de ellas.

Configuración	Tamaño total	Signo	Exponente	Mantisa	Exceso o Polarizante
Simple	32	1	8	23	127
Doble	64	1	11	52	1023
Cuádruple	128	1	15	112	16383

Tabla 5.7: Configuraciones para las representaciones básicas de valores racionales - IEEE754

Además de la configuración respecto de la distribución de tamaño entre exponente y mantisa, este estándar propone una serie de convenciones que abordaremos a continuación. Comencemos por analizar las situaciones donde el resultado de la operación, por ejemplo suma o multiplicación, excede el rango de representación. Un abordaje particular para estos casos que plantean estas representaciones (simple, doble y cuádruple) es la posibilidad de expresar el concepto de *infinito*. Esto es, cuando el número resultante de una operación no se puede representar porque excede el máximo valor representable, se expresa como *infinito positivo* y, cuando excede al valor mínimo, se expresa como *infinito negativo*. Ahora bien, ¿cuál sería el resultado de restarle infinito a infinito?, como podemos razonar su resultado no se puede precisar, por ello estas representaciones contemplan el valor indefinido: *no es un número*, **NaN** por su nombre en inglés **Not a Number**.

Algunas de las operaciones que pueden producir como resultado *NaN* son las siguientes:

- Infinito+ - Infinito+
- Infinito+ + Infinito- ó Infinito- + Infinito+
- 0 divido 0
- cualquier operación con un operando *NaN*

Para poder expresar estos valores particulares, se destina un valor particular del exponente y la mantisa, que además en combinación con el signo nos permite distinguir entre los dos infinitos, positivo y negativo, y el valor *NaN*. Si el *exponente son todos unos* y la *mantisa todos ceros*, estamos en presencia del valor *infinito*, cuyo signo dependerá de la correspondiente posición. Si el *exponente son todos unos* y la *el símbolo más significativo de la mantisa en 1 y el resto en 0* entonces estamos en presencia de *NaN*.

Para finalizar, dado que se asume que la mantisa se encuentra *normalizada*, es decir, siempre debería contener un 1 en la posición más significativa, y con el objetivo de ganar más precisión, las representaciones IEEE considera que el valor del exponente corresponde a la *primera posición oculta* de la mantisa. Con ello *genera una posición extra* de representación en la mantisa, aumentando su precisión. Por ejemplo, si la mantisa está completada con todos 0 y el valor del exponente, una vez decodificado (quitándole el exceso) es *dos*, el valor representado es el *cuatro*, ya que se asume que el símbolo más significativo y oculto es 1 y el resto de mantisa es 0, esto es $2^2 = 4$. Nuevamente, de la situación anterior surge la pregunta ¿cómo expresamos el *cero*? ya que para ello necesitamos que toda la mantisa esté en 0, incluida la posición oculta. Para expresar el valor *cero*, al igual que para los *infinitos* y *NaN*, estas representaciones utilizan una configuración de excepción. Se destina el *exponente con todos ceros* y *mantisa con todos ceros* para expresar el valor *cero*, que por cierto, puede estar en su forma positiva o negativa.

Como ejemplo, la siguiente Tabla 5.8 detalla la configuración y el correspondiente valor para algunos valores racionales.

Valor	Hexadecimal	Signo	Exponente	Mantisa
+0	00000000	0	00000000	00000000000000000000000000000000
-0	80000000	1	00000000	00000000000000000000000000000000
1	3F800000	0	01111111	00000000000000000000000000000000
2	40000000	0	10000000	00000000000000000000000000000000
157.25	431D4000	0	10000110	00111010100000000000000000000000
-2049.3125	C5001500	1	10001010	000000000001010100000000
+Infinito	7F800000	0	11111111	00000000000000000000000000000000
-Infinito	FF800000	1	11111111	00000000000000000000000000000000
NaN	7FC00000	0	11111111	10000000000000000000000000000000
Máximo + ($3,40282347e + 38$)	7F7FFFFF	0	11111110	11111111111111111111111111111111
Mínimo + ($1,17549435e - 38$)	00800000	0	00000001	00000000000000000000000000000000

Tabla 5.8: Ejemplos de valores racionales representados con IEEE754 (Simple)

Como ejemplo de representación y decodificación, la Figura 5.31 muestra la configuración detallada del número *dos mil cuarenta y nueve con tres mil ciento veinticinco milésimas* (sexto valor de la tabla anterior).

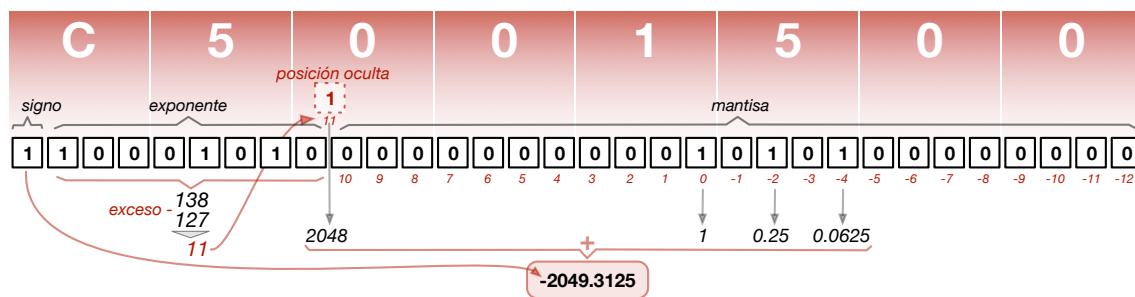


Figura 5.31: Ejemplo de representación y decodificación con IEEE Simple.