Unidad I

Sistemas numéricos

1.1 Sistemas numéricos (Binario, Octal, Decimal, Hexadecimal)

Los computadores manipulan y almacenan los datos usando interruptores electrónicos que están ENCENDIDOS o APAGADOS. Los computadores sólo pueden entender y usar datos que están en este formato binario, o sea, de dos estados. Los unos y los ceros se usan para representar los dos estados posibles de un componente electrónico de un computador. Se denominan dígitos binarios o bits. Los 1 representan el estado ENCENDIDO, y los 0 representan el estado APAGADO.

El Código americano normalizado para el intercambio de información (ASCII) es el código que se usa más a menudo para representar los datos alfanuméricos de un computador. ASCII usa dígitos binarios para representar los símbolos que se escriben con el teclado. Cuando los computadores envían estados de ENCENDIDO/APAGADO a través de una red, se usan ondas eléctricas, de luz o de radio para representar los unos y los ceros. Observe que cada carácter tiene un patrón exclusivo de ocho dígitos binarios asignados para representar al carácter.

Debido a que los computadores están diseñados para funcionar con los interruptores ENCENDIDO/APAGADO, los dígitos y los números binarios les resultan naturales. Los seres humanos usan el sistema numérico decimal, que es relativamente simple en comparación con las largas series de unos y ceros que usan los computadores. De modo que los números binarios del computador se deben convertir en números decimales.

A veces, los números binarios se deben convertir en números Hexadecimales (hex), lo que reduce una larga cadena de dígitos binarios a unos pocos caracteres hexadecimales. Esto hace que sea más fácil recordar y trabajar con los números.

Bits y Bytes

Un número binario 0 puede estar representado por 0 voltios de electricidad (0 = 0 voltios).

Un número binario 1 puede estar representado por +5 voltios de electricidad (1 = +5 voltios).

Los computadores están diseñados para usar agrupaciones de ocho bits. Esta agrupación de ocho bits se denomina byte. En un computador, un byte representa

una sola ubicación de almacenamiento direccionable. Estas ubicaciones de almacenamiento representan un valor o un solo carácter de datos como, por ejemplo, un código ASCII. La cantidad total de combinaciones de los ocho interruptores que se encienden y se apagan es de 256. El intervalo de valores de un byte es de 0 a 255. De modo que un byte es un concepto importante que se debe entender si uno trabaja con computadores y redes.

Unidades de Medida de Almacenamiento					
Medida	Simbología	Equivalencia	Equivalencia en bits	Equivalencia en Bytes	Ejemplos
Bit	b	0 ó 1	1	-	Encendido
Byte	B	8 bits	8	1	Un carácter
Kilobyte	KB	1024 B	8,192	1,024	Un block de notas
Megabyte	MB	1024 KB	8,388,608	1,048,576	Una foto de 2Mp
Gigabyte	GB	1024 MB	8,589,934,592	1,073,741,824	256 canciones de Mp3
Terabyte	TB	1024 GB	8,796,093,022,208	1,099,511,627,776	217 DVD
Petabyte	PB	1024 TB	9,007,199,254,740,992	1,125,899,906,842,624	20,971 BluRay de doble capa
Exabyte	EB	1024 PB	9,223,372,036,854,775,808	1,152,921,504,606,846,976	1 Datacenter
Zetabyte	ZB	1024 EB	9,444,732,965,739,290,427,392	1,180,591,620,717,411,303,424	100 Datacenters
Yottabyte	YB	1024 ZB	9,671,406,556,917,033,397,649,408	1,208,925,819,614,629,174,706,176	Un millón de Datacenters

2

1.2 Conversiones entre sistemas numéricos.

Los sistemas numéricos están compuestos por símbolos y por las normas utilizadas para interpretar estos símbolos. El sistema numérico que se usa más a menudo es el sistema numérico decimal, o de Base 10. El sistema numérico de Base 10 usa diez símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9. Estos símbolos se pueden combinar para representar todos los valores numéricos posibles.

El sistema numérico decimal se basa en potencias de 10. Cada posición de columna de un valor, pasando de derecha a izquierda, se multiplica por el número 10, que es el número de base, elevado a una potencia, que es el exponente. La potencia a la que se eleva ese 10 depende de su posición a la izquierda de la coma decimal. Cuando un número decimal se lee de derecha a izquierda, el primer número o el número que se ubica más a la derecha representa 100 (1), mientras que la segunda posición representa 101 (10 x 1= 10) La tercera posición representa 102 (10 x 10 =100). La séptima posición a la izquierda representa 106 (10 x 10 x 10 x 10 x 10 x 10 =1.000.000). Esto siempre funciona, sin importar la cantidad de columnas que tenga el número.

Ejemplo:

2134 = (2x10 potencia 3) + (1x10 potencia 2) + (3x10 potencia 1) + (4x10 potencia 0)

Hay un 4 en la posición correspondiente a las unidades, un 3 en la posición de las decenas, un 1 en la posición de las centenas y un 2 en la posición de los miles. Este ejemplo parece obvio cuando se usa el sistema numérico decimal. Es importante saber exactamente cómo funciona el sistema decimal, ya que este conocimiento permite entender los otros dos sistemas numéricos, el sistema numérico de Base 2 y el sistema numérico hexadecimal de Base 16. Estos sistemas usan los mismos métodos que el sistema decimal.

Sistema Numérico de Base 2

Los computadores reconocen y procesan datos utilizando el sistema numérico binario, o de Base 2. El sistema numérico binario usa sólo dos símbolos, 0 y 1, en lugar de los diez símbolos que se utilizan en el sistema numérico decimal. La posición, o el lugar, que ocupa cada dígito de derecha a izquierda en el sistema numérico binario representa 2, el número de base, elevado a una potencia o exponente, comenzando desde 0. Estos valores posicionales son, de derecha a izquierda, 2 potencia 0, 2 potencia 1, 2 potencia 2, 2 potencia 3, 2 potencia 4, 2 potencia 5, 2 potencia 6 y 2 potencia 7, o sea, 1, 2, 4, 8, 16, 32, 64 y 128, respectivamente.

Ejemplo:

101102 = (1 x 2 potencia 4 = 16) + (0 x 2 potencia 3 = 0) + (1 x 2 potencia 2 = 4) + (1 x 2potencia1= 2)+(0 x 2 potencia 0 = 0) = 22 (16 + 0 + 4 + 2 + 0)

Al leer el número binario (101102) de izquierda a derecha, se nota que hay un 1 en la posición del 16, un 0 en la posición del 8, un 1 en la posición del 4, un 1 en la posición del 2 y un 0 en la posición del 1, que sumados dan el número decimal 22.

Sistema Numérico de Base 8

El inconveniente de la codificación binaria es que la representación de algunos números resulta muy larga. Por este motivo se utilizan otros sistemas de numeración que resulten más cómodos de escribir: el sistema octal y el sistema hexadecimal. Afortunadamente, resulta muy fácil convertir un número binario a octal o a hexadecimal.

En el sistema octal, usa ocho dígitos diferentes: 0, 1, 2, 3, 4, 5, 6 y 7.Cada posición de columna de un valor, pasando de derecha a izquierda, se multiplica por el número 8, que es el número de base, elevado a una potencia, que es el exponente. Cada dígito tiene, naturalmente, un valor distinto dependiendo del lugar que ocupen. El valor de cada una de las posiciones viene determinado por las potencias de base 8.

Ejemplo:

El número octal 2738 = 2*8 potencia 2 + 7*8 potencia 1 + 3*8 potencia 0 = 2*64 + 7*8 + 3*1 = 187

Sistema Numérico de Base 16 (Hexadecimal)

El sistema hexadecimal usa dieciséis símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F. Se utilizan los caracteres A, B, C, D, E y F representando las cantidades decimales 10, 11, 12, 13, 14 y 15 respectivamente, porque no hay dígitos mayores que 9 en el sistema decimal. Cada dígito tiene, naturalmente, un valor distinto dependiendo del lugar que ocupen. El valor de cada una de las posiciones viene determinado por las potencias de base 16.

Ejemplo:

El valor del número hexadecimal 1A3F = 1*16 potencia 3 + A*16 potencia 2 + 3*16 potencia 1 + F*16 potencia 0

1*4096 + 10*256 + 3*16 + 15*1 = 6719

1A3F16 = 671910

1.3 Operaciones básicas (Suma, Resta, Multiplicación, División)

Suma de números binarios

La tabla de sumar para números binarios es la siguiente:

+	0	1
0	0	1
1	1	10

Las posibles combinaciones al sumar dos bits son:

0 + 0 = 0

0 + 1 = 1
1 + 0 = 1
1 + 1 = 10

Note que al sumar 1 + 1 es 102, es decir, llevamos 1 a la siguiente posición de la izquierda (acarreo). Esto es equivalente, en el sistema decimal a sumar 9 + 1, que da 10: cero en la posición que estamos sumando y un 1 de acarreo a la siguiente posición.

Ejemplo

Acarreo			1					
	1	0	0	1	1	0	0	0
+	0	0	0	1	0	1	0	1
Resultado	**1**	**0**	**1**	**0**	**1**	**1**	**0**	**1**

Se puede convertir la operación binaria en una operación decimal, resolver la decimal, y después transformar el resultado en un (número) binario. Operamos como en el sistema decimal: comenzamos a sumar desde la derecha, en nuestro ejemplo, 1 + 1 = 10, entonces escribimos 0 en la fila del resultado y llevamos 1 (este "1" se llama acarreo o arrastre). A continuación se suma el acarreo a la siguiente columna: 1 + 0 + 0 = 1, y seguimos hasta terminar todas la columnas (exactamente como en decimal).

Resta de números binarios

El algoritmo de la resta en sistema binario es el mismo que en el sistema decimal. Pero conviene repasar la operación de restar en decimal para comprender la operación binaria, que es más sencilla. Los términos que intervienen en la resta se llaman minuendo, sustraendo y diferencia.

Las restas básicas 0 - 0, 1 - 0 y 1 - 1 son evidentes:

0 - 0 = 0

1 - 0 = 1

1 - 1 = 0

0 - 1 = **1** (se transforma en 10 - 1 = 1) (en sistema decimal equivale a 2 - 1 = 1)

La resta 0 - 1 se resuelve, igual que en el sistema decimal, tomando una unidad prestada de la posición siguiente: 0 - 1 = **1** y *me llevo* 1, lo que equivale a decir en el sistema decimal, 2 - 1 = 1.

En decimal, por ejemplo tienes 100-19, obviamente a 0 no le puedes quitar 9, así que debemos tomar prestado 1 para volverlo un 10 (en decimal la base es 10), y así si 10-9=1.

En binarios pasa lo mismo, no le puedes quitar 1 a 0, debes de tomar 1 prestado al de un lado, pero cuidado aquí viene lo complicado tu numero no se va a volver 10, recuerda que en binario la base es 2 y por lo tanto se volverá 2 en binario, y ahora sí a 2 le quitas 1, 2-1=1, y continuas restando pero recuerda que llevas 1, porque pediste prestado.

Ejemplo para que le entiendas mejor, vamos a restar 201 - 67, ya sabemos que es 134, vamos a hacerlo en binario :

```
  1 1 0 0 1 0 0 1.....................201
- 0 1 0 0 0 0 1 1.....................67
```

Tomamos los dos últimos números, 1-1 es igual a 0, y no llevamos nada (no pedimos prestado)

```
  1 1 0 0 1 0 0 1
- 0 1 0 0 0 0 1 1
-----------------------
                    0
```

Ahora la siguiente columna 0-1, ya dijimos que no se puede, así que va a tomar 1 prestado al de la columna del lado izquierdo, se que vas a decir "es un cero, no nos puede prestar 1", lo que pasa es que ese cero le pide a su vez al de lado, y así hasta que encuentres un 1, pero no te fijes en eso, vamos a seguir restando y no nos vamos a preocupar por eso ahora, entonces ahora nos prestaron 1 (no importa quién) y tenemos un 1 0 (este numero es 2 en binario no 10 en decimal, no te vayas a confundir), entonces en binario tienes 10-1, que en decimal es 2-1=1, y llevamos 1 (porque pedimos 1 prestado)

```
  1 1 0 0 1 0 0 1 arriba
- 0 1 0 0 0 0 1 1 abajo
-----------------------
                  1 0
```

Para la siguiente columna tenemos 0 - 0, pero recuerda que tomamos 1 prestado así que en realidad tenemos 0 - 1 (le sumamos el 1 al de abajo), de nuevo

tenemos que pedir prestado y entonces tenemos en binaria 1 0 -1 que en decimal es 2-1=1, y de nuevo llevamos 1

```
  1 1 0 0 1 0 0 1
- 0 1 0 0 0 0 1 1
-----------------------
            1 1 0
```

Continuamos con 1 - 0 , pero como llevamos 1 tenemos ahora 1 - 1, esto si lo podemos resolver 1 - 1 = 1 (en binario y decimal).

```
  1 1 0 0 1 0 0 1
- 0 1 0 0 0 0 1 1
-----------------------
          0 1 1 0
```

Lo demás es muy fácil:
0 - 0=0
0 - 0=0
1 - 1=0
1 - 0=1

```
  1 1 0 0 1 0 0 1
- 0 1 0 0 0 0 1 1
-----------------------
  1 0 0 0 0 1 1 0
```

que en decimal es 134.

Es lo mismo que la resta en decimal, pides prestado y llevas, nada más debes de ser cuidadoso y recordar que tu base es 2.

"En este mundo solo existen 10 tipos de personas, las que saben binario y las que no" =)

PRODUCTO DE NÚMEROS BINARIOS

La tabla de multiplicar para números binarios es la siguiente:

·	0	1

0	0	0
1	0	1

El algoritmo del producto en binario es igual que en números decimales; aunque se lleva a cabo con más sencillez, ya que el 0 multiplicado por cualquier número da 0, y el 1 es el elemento neutro del producto.

Por ejemplo, multipliquemos 10110 por 1001:

10110 X 1001

10110

00000

00000

10110

11000110

División de números binarios

La división en binario es similar al decimal; la única diferencia es que a la hora de hacer las restas, dentro de la división, éstas deben ser realizadas en binario.

Ejemplo

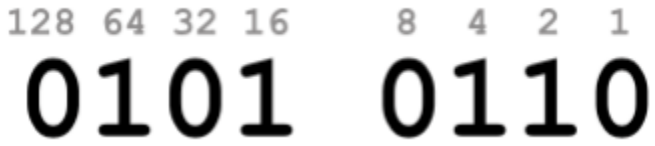
Dividir 100010010 (274) entre 1101 (13):

```
  100010010 |1101
-0000               010101
  10001
  -1101
    01000
   - 0000
     10000
     - 1101
       00011
```

- 0000
01110
- 1101
00001

1.4 Algoritmos de Booth para la multiplicación y división en binario.

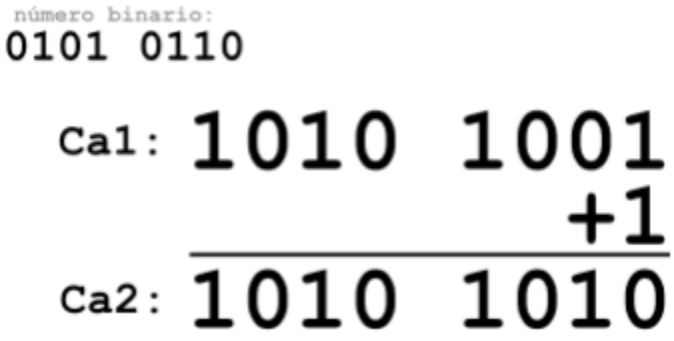
El algoritmo de Booth es un método rápido y sencillo para obtener el producto de dos números binarios con signo en notación complemento a dos.
Debemos saber que un número binario está formado por bits de ceros y unos, y que se puede traducir a decimal fácilmente de la siguiente forma:



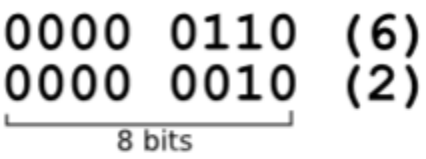
Sabiendo que la posición de cada bit es 2^n (elevado a n) y partimos de n=0 de derecha a izquierda, sólo queda realizar la suma total de multiplicar por dicho bit, en este caso:

(0·2^7+1·2^6+0·2^5+1·2^4+0·2^3+1·2^2+1·2^1+0·2^0 = 86).

También debemos saber que el complemento a uno de un número binario es cambiar sus ceros por unos, y sus unos por ceros (complementar): (010010 -> ca1:101101) y que el complemento a dos de un número binario es el resultado de sumar 1 al complemento a uno de dicho número binario:

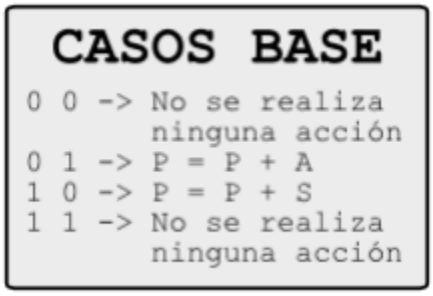


Realizar una suma con dos números binarios es tarea fácil, pero la multiplicación resulta algo más complicada. Con el algoritmo de Booth, resulta mucho más sencillo de implementar. Partimos del ejemplo de la multiplicación 6·2=12:

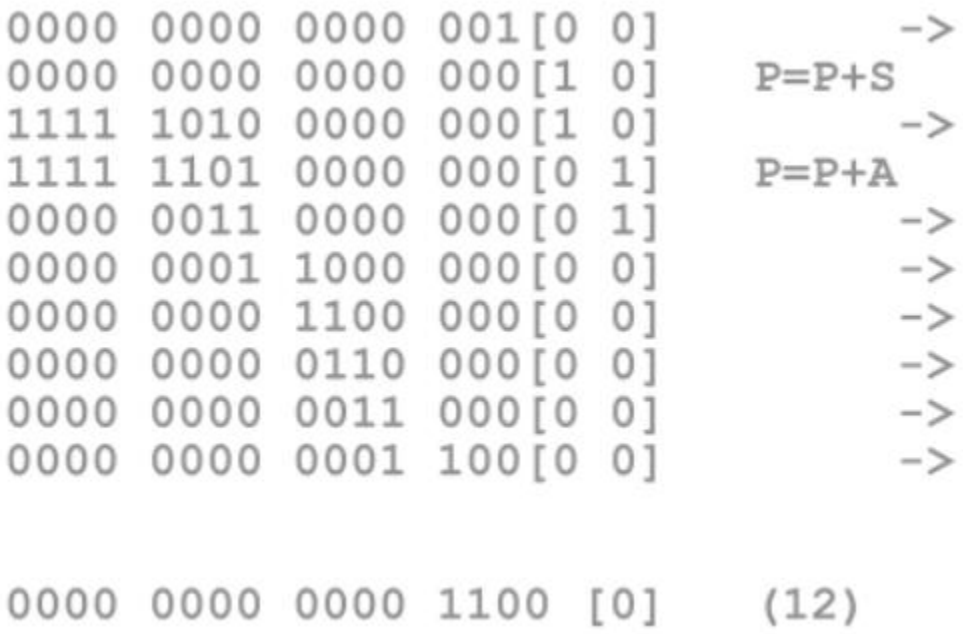


Como se puede ver en la imagen superior, partiendo de los números binarios de la multiplicación 6·2 (multiplicando y multiplicador) creamos tres nuevos números binarios del doble de tamaño (16 en el ejemplo): A, S y P.

Partiendo del número P (producto) comenzamos a comparar los últimos 2 bits de la derecha, siguiendo los casos base del recuadro:



Se realizará esta comparación 8 veces en este ejemplo (número de bits de los operandos) y al final de cada comparación, realizamos un desplazamiento de un bit hacia la derecha, manteniendo el último bit de la izquierda, y descartando el último bit del lado contrario. Si hacemos una traza paso a paso nos quedarían los siguientes resultados:



Finalmente obtenemos el número en binario resultante (12 en este ejemplo), descartando el bit extra que hemos añadido al principio del procedimiento y que se encuentra en el extremo a la derecha.

1.5 Aplicación de los sistemas numéricos en la computación.

Existe una cantidad infinita de sistemas numéricos, sin embargo, para unacomputadora, únicamente existen 4, que son el Binario (con base 2), el octal (conbase 8), el decimal (base 10) y hexadecimal (base 16). Detallaremos el uso decada uno de ellos por la computadora.