El formato "Big Endian" y el "Little Endian"

Estas palabras se usan para referirse a las dos formas en que se pueden guardar los números que ocupan más de un byte.

Vamos a verlo usando como ejemplo el número 5. Si lo expresamos en binario con 4 bits, su equivalencia es 0101. Si lo hacemos con 8 bits, sería 00000101. Pero si usamos más de un byte para representarlo, por ejemplo porque se tratara de un número "entero corto", usando la nomenclatura del lenguaje C, tenemos que el número 5 equivaldría a 16 dígitos binarios: 00000000 00000101.

Por tanto, ahora este número 5 queda representado como 2 bytes. El de la izquierda es el byte más significativo (MSB) y el de la derecha es el bit menos significativo (LSB). Estos nombres vienen de que un cambio en el byte de la derecha hace que el número cambie poco (por ejemplo, 00000000 00000100 = 4 en decimal), mientras que un cambio en el byte de la izquierda provoca cambios mucho más grandes (00000001 00000101 = 261 en decimal).

Pues bien, según la arquitectura del ordenador puede ocurrir que esos dos bytes se guarden en el orden que hemos visto: 00000000 00000101 (primero el MSB y luego el LSB) o al contrario (00000101 00000000). Lo primero, que a priori es lo que podría parecer más natural, es lo que se hace en procesadores como los de la familia PowerPc (usados en muchos ordenadores Apple, por ejemplo). Lo segundo, aunque parezca más enrevesado, es lo que se hace en la mayoría de procesadores de Intel, como los usados en los ordenadores PC.

El primer formato (MSB LSB) es lo que se conoce como Big Endian, porque eso de que el extremo más grande aparece en primer lugar. El segundo formato (LSB MSB) es lo que se conoce como Little Endian, porque se almacena primero el dato más pequeño.

¿Y eso en qué afecta a un programador? En que muchas veces deberemos saber con qué plataforma se ha creado un fichero de datos, para poderlo interpretar correctamente.

Por ejemplo, si leemos un fichero que contiene un dato "entero corto", de 2 bytes, formado por la secuencia 00000101 00000000, y ese fichero se ha creado con un ordenador basado en un Pentium u otro procesador de Intel (Little Endian), podemos suponer que ese número es un 5. Por el contrario, si el dato se ha creado desde un equipo Apple clásico, la secuencia 00000101 00000000 estaría en formato Big Endian, luego equivaldría al número 1280.

En general, en caso de que un fichero contenga datos numéricos de más de un byte, deberíamos saber si están almacenados en formato Little Endian o Big Endian, e intercambiar el orden de los bytes después de leer, si fuera necesario.

Como primera curiosidad, existen arquitecturas que permiten escoger la "endianness" que se prefiere usar (como IA64, MIPS y ARM) y que reciben el nombre de "bi-endian". En estos sistemas, normalmente este cambio se puede hacer por software (al arrancar el equipo,por ejemplo), pero en algún caso se ha de realizar por hardware (como podría ser cambiando un jumper en la placa base).

Como segunda curiosidad, el nombre de "Big Endian" y de "Little Endian" se tomó irónicamente de "Los viajes de Gulliver", en que aparece una discusión sobre si un huevo hervido debería empezar a comerse abriéndolo por su extremo pequeño o por su extremo grande.