[1. Little-Endian Big-Endian 2](#_Toc464813504)

[1-1. 指的是CPU结构的事情，不是操作系统或者文件 2](#_Toc464813505)

[1-2. 具体内容 2](#_Toc464813506)

[1-3. 数字的高位地位表示和在文件中的位置 2](#_Toc464813507)

# Little-Endian Big-Endian

## 指的是CPU结构的事情，不是操作系统或者文件

所以Little Endian还是Big Endian 的全称应该是CPU是Little Endian还是Big Endian。

## 具体内容

http://blog.163.com/huang\_bp/blog/static/123119837201081611131341

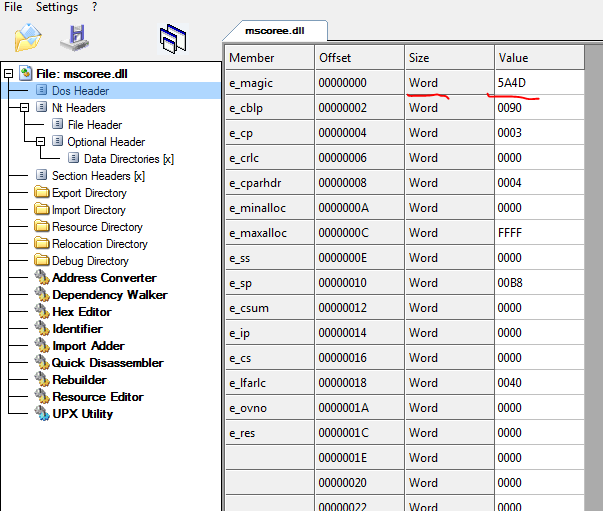
计算机的所有内存以字节数组的方式进行编址。

当一个逻辑上长于一个字节的整形数据放置在内存中时（比如16位，32位，和64位的整数），需要考虑这些字节的存储顺序。Little Endian: 将字节的逻辑顺序与物理顺序一致，即将逻辑上较低的字节放置在物理上较低的字节上，比如Intel x86系列；

Big Endian字节的逻辑顺序与物理顺序相反，即将逻辑上较低的字节放置在物理上较高的字节上，比如Motorola的PowerPC以及Sun Sparc。还有一些平台同时支持两种方案，由开发者决定使用哪一种。

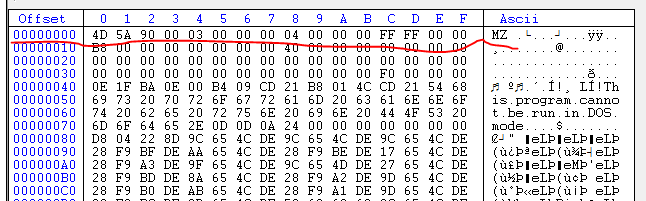
## 数字的高位地位表示和在文件中的位置

一般我们读一个数的时候，如一个16位的Word 值5A4D（代表字符MZ），都是左边的是逻辑高位，右边的是逻辑地位，5A4D的完整表示应该是0x5A4D



当我们看以HEX的形式看一个文件时，最左边是地位地址，右边是高位地址

从00000000 到0000000F如下：



所以0x5A4D 在文件中就显示成了4D 5A。

所以在文件中要读一个数字，不能简单把几个byte连接起来。首先要知道这个数字的长度，byte，word ，int， int64， 然后找到最地位和最高位的byte，然后然后逻辑高位到逻辑地位拼接出来。