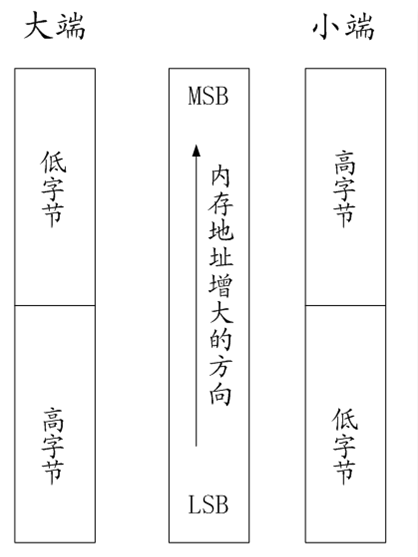
## 一、字节序的概念

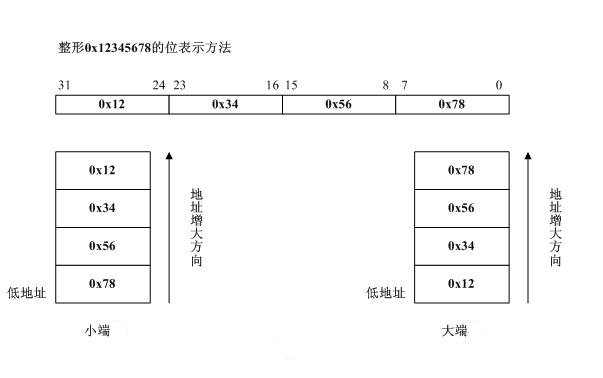
字节序是指多字节数据的存储顺序，在设计计算机系统的时候，有两种处理内存中数据的方法：大端格式、小端格式。

小端格式(Little-Endian)：将低位字节数据存储在低地址。

大端格式(Big-Endian)：将高位字节数据存储在低地址。



举个简单的例子，对于整形 0x12345678，它在大端格式和小端格式的系统中，分别如下图所示的方式存放：



下面例子为确定主机的字节序：

    unsigned int a = 0x12345678;

    unsigned char \*p = (unsigned char \*)&a;     //只取一个字节

    if(0x12==\*p)

    {

        printf("Big-Endian\n");

    }

    else if(0x78==\*p)

    {

        printf("Little-Endian\n");

    }

**网络协议指定了通讯字节序：大端**

## 二、字节序转换函数介绍

以下接口所需头文件：#include <arpa/inet.h>

uint32\_t htonl(uint32\_t hostint32);

功能：将 32 位主机字节序数据转换成网络字节序数据

参数：hostint32：需要转换的 32 位主机字节序数据，uint32\_t 为 32 为无符号整型

返回值：成功，返回网络字节序的值

uint16\_t htons(uint16\_t hostint16);

功能：将 16 位主机字节序数据转换成网络字节序数据

参数：hostint16：需要转换的 16 位主机字节序数据，uint16\_t，unsigned short int

返回值：成功，返回网络字节序的值

uint32\_t ntohl(uint32\_t netint32);

功能：将 32 位网络字节序数据转换成主机字节序数据

参数：netint32：待转换的 32 位网络字节序数据，uint32\_t，unsigned int

返回值：成功：返回主机字节序的值

uint16\_t ntohs(uint16\_t netint16);

功能：将 16 位网络字节序数据转换成主机字节序数据

参数：netint16：待转换的 16 位网络字节序数据，uint16\_t，unsigned short int

返回值：成功：返回主机字节序的

    int a = 0x01020304;

    short int b = 0x0102;

    printf("htonl(0x%08x)=0x%08x\n", a, htonl(a));

    printf("htons(0x%04x)=0x%04x\n", b, htons(b));

**三、地址转换函数**

在网络上传输地址需要使用32位int的ip地址，所以我们的ip地址需要继续转换

以下接口所需头文件：#include <arpa/inet.h>

int inet\_pton(int family, const char \*strptr, void \*addrptr);

功能：

将点分十进制数串转换成 32 位无符号整数

参数：

family：协议族( AF\_INET、AF\_INET6、PF\_PACKET 等 )，常用 AF\_INET

strptr：点分十进制数串

addrptr：32 位无符号整数的地址

返回值：

成功返回 1 、 失败返回其它

const char \*inet\_ntop( int family, const void \*addrptr,   char \*strptr, size\_t len );

功能：

将 32 位无符号整数转换成点分十进制数串

参数：

family：协议族( AF\_INET、AF\_INET6、PF\_PACKET 等 )，常用 AF\_INET

addrptr：32 位无符号整数

strptr：点分十进制数串

len：strptr 缓存区长度

len 的宏定义

#define INET\_ADDRSTRLEN   16  // for ipv4

#define INET6\_ADDRSTRLEN  46  // for ipv6

返回值：

成功：则返回字符串的首地址

失败：返回 NULL

测试示例：

    char ip\_str[]="172.20.223.75";

    unsigned int ip\_uint = 0;

    // 将ip地址字符串转成4个字节的int，每个字节保存一个ip段

    inet\_pton(AF\_INET,ip\_str,&ip\_uint);

printf("in\_uint = %d\n",ip\_uint);

    // 根据字节获取ip段

    unsigned char \*ip\_p = (unsigned char \*)&ip\_uint;

printf("in\_uint = %d,%d,%d,%d\n",\*ip\_p,\*(ip\_p+1),\*(ip\_p+2),\*(ip\_p+3));      // 输出 172,20,223,75

    char ip\_s[16];

    // 将int字节转成ip字符串

    inet\_ntop(AF\_INET, &ip\_uint, ip\_s, 16);

    printf("ip\_str = %s\n",ip\_s);