第六十三节： 指针“化整为零”和“化零为整”的“灵活”应用。

【63.1　化整为零的“灵活”应用。】

上一节讲“化整为零”的例子，指针是跟数组的首地址（下标是 0）“绑定”的，这样，很多初学者就误以为指针跟数组“绑定”时，只能跟数组的“首地址”关联。其实，指针是可以跟数组的任何一个成员的地址“绑定”（只要不超过数组的长度导致越界），它不仅仅局限于首地址，指针的这个特征就是本节标题所说的“灵活”。请看下面这个例子：

有 3 个变量，分别是单字节 unsigned char a，双字节 unsigned int b，四字节 unsigned long c，它们加起来一共有 7 个字节，要把这 7 个字节放到一个 7 字节容量的数组里。除了用传统的“移位法”，还有一种更加便捷的“指针法”，代码如下：

```
unsigned char a=0x01;
unsigned int b=0x0203;
unsigned long c=0x04050607;

unsigned char Gu8BufferABC[7]; //存放 3 个不同长度变量的数组

unsigned char *pu8;   //引入的 unsigned char 类型指针
unsigned int *pu16;   //引入的 unsigned int 类型指针
unsigned long *pu32;  //引入的 unsigned long 类型指针

pu8=&Gu8BufferABC[0]; //指针跟数组的第 0 个位置“绑定”起来。
*pu8=a; //把 a 的 1 个字节放在数组第 0 个位置。

pu16=(unsigned int *)&Gu8BufferABC[1]; //指针跟数组的第 1 个位置“绑定”起来。
*pu16=b;            //把 b 的 2 个字节放在数组第 1、2 这两个位置。

pu32=(unsigned long *)&Gu8BufferABC[3]; //指针跟数组的第 3 个位置“绑定”起来。
*pu32=c;            //把 c 的 4 个字节放在数组第 3、4、5、6 这四个位置。
```

【63.2　化零为整的“灵活”应用。】

刚才讲的是“化整为零”，现在讲的是“化零为整”。刚才讲的是“分解”，现在讲的是“合成”。请看下面这个例子：

有一个容量为 7 字节数组，第 0 字节存放的是 unsigned char d 变量，第 1、2 字节存放的是 unsigned int e 变量，第 3、4、5、6 字节存放的是 unsigned long f 变量，现在要从数组中“零散”的字节里提取并且合成为“完整”的 3 个变量。代码如下：

```
unsigned char Gu8BufferDEF[7]={0x01,0x02,0x03,0x04,0x05,0x06,0x07}; //注意大小端的问题

unsigned char d;
```

```
    unsigned int e;
    unsigned long f;

    unsigned char *pu8;   //引入的 unsigned char 类型指针
    unsigned int *pu16;   //引入的 unsigned int 类型指针
    unsigned long *pu32;  //引入的 unsigned long 类型指针

    pu8=&Gu8BufferDEF[0]; //指针跟数组的第 0 个位置“绑定”起来。
    d=*pu8;           //从数组第 0 位置提取单字节完整的 d 变量。

    pu16=(unsigned int *)&Gu8BufferDEF[1]; //指针跟数组的第 1 个位置“绑定”起来。
    e=*pu16;         //从数组第 1,2 位置提取双字节完整的 e 变量。

    pu32=(unsigned long *)&Gu8BufferDEF[3]; //指针跟数组的第 3 个位置“绑定”起来。
    f=*pu32;         //从数组第 3,4,5,6 位置提取四字节完整的 f 变量。
```

【63.3　例程练习和分析。】

现在编一个练习程序。

```
/*---C 语言学习区域的开始。-------------------------------------------*/

    unsigned char a=0x01;
    unsigned int b=0x0203;
    unsigned long c=0x04050607;
    unsigned char Gu8BufferABC[7]; //存放 3 个不同长度变量的数组

    unsigned char Gu8BufferDEF[7]={0x01,0x02,0x03,0x04,0x05,0x06,0x07}; //注意大小端的问题
    unsigned char d;
    unsigned int e;
    unsigned long f;

    unsigned char *pu8;   //引入的 unsigned char 类型指针
    unsigned int *pu16;   //引入的 unsigned int 类型指针
    unsigned long *pu32;  //引入的 unsigned long 类型指针


void main() //主函数
{
    //第 1 类例子：化整为零。
    pu8=&Gu8BufferABC[0]; //指针跟数组的第 0 个位置“绑定”起来。
    *pu8=a; //把 a 的 1 个字节放在数组第 0 个位置。
```

```
    pu16=(unsigned int *)&Gu8BufferABC[1]; //指针跟数组的第 1 个位置“绑定”起来。
    *pu16=b;             //把 b 的 2 个字节放在数组第 1、2 这两个位置。

    pu32=(unsigned long *)&Gu8BufferABC[3]; //指针跟数组的第 3 个位置“绑定”起来。
    *pu32=c;             //把 c 的 4 个字节放在数组第 3、4、5、6 这四个位置。

    //第 2 类例子：化零为整。
    pu8=&Gu8BufferDEF[0]; //指针跟数组的第 0 个位置“绑定”起来。
    d=*pu8;            //从数组第 0 位置提取单字节完整的 d 变量。

    pu16=(unsigned int *)&Gu8BufferDEF[1]; //指针跟数组的第 1 个位置“绑定”起来。
    e=*pu16;          //从数组第 1,2 位置提取双字节完整的 e 变量。

    pu32=(unsigned long *)&Gu8BufferDEF[3]; //指针跟数组的第 3 个位置“绑定”起来。
    f=*pu32;          //从数组第 3,4,5,6 位置提取四字节完整的 f 变量。


    View(Gu8BufferABC[0]);  //把第 1 个数 Gu8BufferABC[0]发送到电脑端的串口助手软件上观察。
    View(Gu8BufferABC[1]);  //把第 2 个数 Gu8BufferABC[1]发送到电脑端的串口助手软件上观察。
    View(Gu8BufferABC[2]);  //把第 3 个数 Gu8BufferABC[2]发送到电脑端的串口助手软件上观察。
    View(Gu8BufferABC[3]);  //把第 4 个数 Gu8BufferABC[3]发送到电脑端的串口助手软件上观察。
    View(Gu8BufferABC[4]);  //把第 5 个数 Gu8BufferABC[4]发送到电脑端的串口助手软件上观察。
    View(Gu8BufferABC[5]);  //把第 6 个数 Gu8BufferABC[5]发送到电脑端的串口助手软件上观察。
    View(Gu8BufferABC[6]);  //把第 7 个数 Gu8BufferABC[6]发送到电脑端的串口助手软件上观察。

    View(d);               //把第 8 个数 d 发送到电脑端的串口助手软件上观察。
    View(e);               //把第 9 个数 e 发送到电脑端的串口助手软件上观察。
    View(f);               //把第 10 个数 f 发送到电脑端的串口助手软件上观察。

    while(1)
    {
    }
}
/*---C 语言学习区域的结束。-----------------------------------------------*/
```

在电脑串口助手软件上观察到的程序执行现象如下：

```
1 个数
十进制:1
十六进制:1
二进制:1

第 2 个数
```

十进制:2
十六进制:2
二进制:10

第 3 个数
十进制:3
十六进制:3
二进制:11

第 4 个数
十进制:4
十六进制:4
二进制:100

第 5 个数
十进制:5
十六进制:5
二进制:101

第 6 个数
十进制:6
十六进制:6
二进制:110

第 7 个数
十进制:7
十六进制:7
二进制:111

第 8 个数
十进制:1
十六进制:1
二进制:1

第 9 个数
十进制:515
十六进制:203
二进制:1000000011

第:个数（这里是第 10 个数。本模块程序只支持显示第 1 到第 9 个,所以这里没有显示“10”）
十进制:67438087
十六进制:4050607
二进制:1000000010100000011000000111

分析：

Gu8BufferABC[0]为 0x01。

Gu8BufferABC[1]为 0x02。

Gu8BufferABC[2]为 0x03。

Gu8BufferABC[3]为 0x04。

Gu8BufferABC[4]为 0x05。

Gu8BufferABC[5]为 0x06。

Gu8BufferABC[6]为 0x07。

d 为 0x01。

e 为 0x0203。

f 为 0x04050607。

【63.4　如何在单片机上练习本章节 C 语言程序？】

直接复制前面章节中第十一节的模板程序，练习代码时只需要更改“C 语言学习区域”的代码就可以了，其它部分的代码不要动。编译后，把程序下载进带串口的 51 学习板，通过电脑端的串口助手软件就可以观察到不同的变量数值，详细方法请看第十一节内容。