****1.每个扇区的块0、块1、块2为数据块，可用于存贮数据。****

****数据块****可作两种应用：

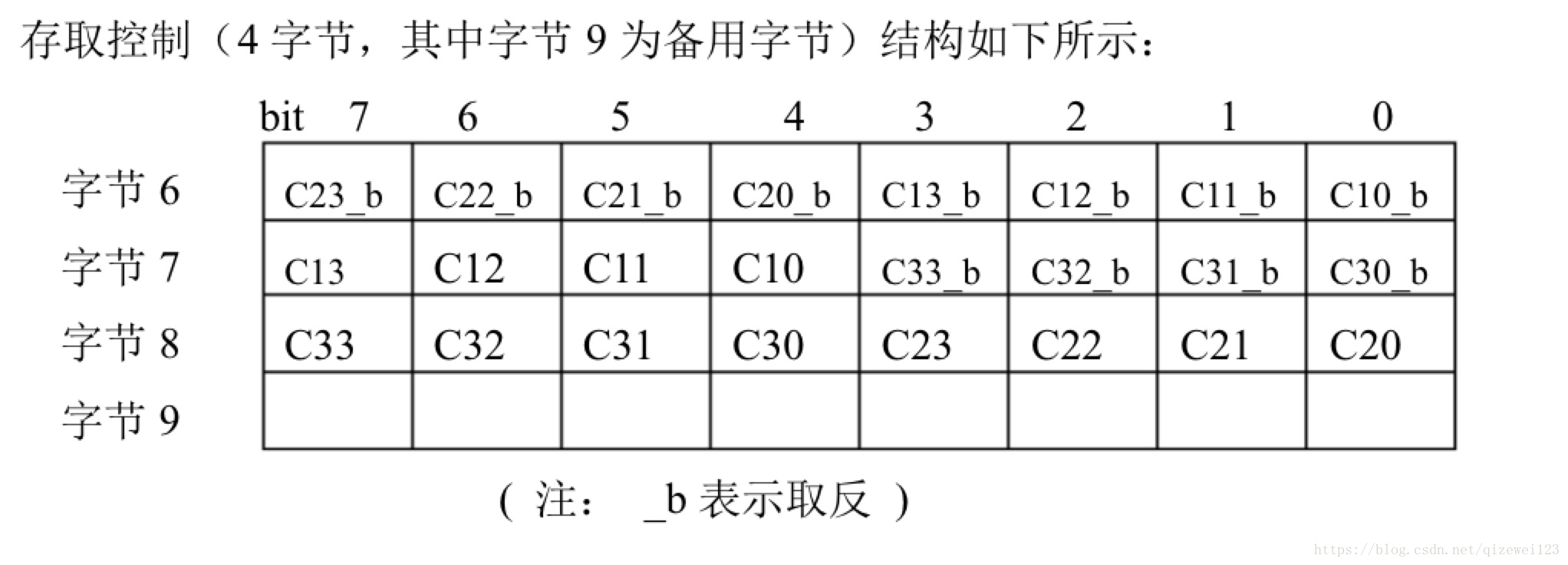
* 用作一般的数据保存，可以进行****读****、****写****操作。
* 用作数据值，可以进行****初始化值、加值、减值、读值****操作。

****2.每个扇区的块3为控制块，包括了密码A、存取控制、密码B。具体结构如下：****

A0 A1 A2 A3 A4 A5  FF 07 80 69  B0 B1 B2 B3 B4 B5

密码A（6字节） 存取控制（4字节） 密码B（6字节）

这个是每个扇区 最后一块的存储控制结构，一共4个字节，例如：（FF 07 80 69） 。 这些数字是16进制的数字，然后一个1byte（字节）=8bits，所以他们的结构如下：



我们知道如果要读取一个扇区的每一块的数据都需要通过KeyA或者KeyB的验证，而厂家生产出来的M1卡默认的控制位为（FF 07 80 69），默认的KeyA和KeyB为（FFFFFFFFFFFF）。但是这样是非常的不安全的，所以我们需要修改原始密码。原始密码的修改是非常的简单的，因为默认的控制位是（FF 07 80 69），对应的每一块的权限表达为：

首先按照16进制，把控制位解析出来：



现在我们就对照上面的表就能知道默认的控制位转换后是个什么数：

字节6——FF ： 1 1 1 1    1 1 1 1

字节7——07 ：  0 0 0 0    0 1 1 1

字节8——80 ：  1 0 0 0    0 0 0 0

字节9——69 ：（这个一般不用）

把数据填入到存取控制结构表如下图：



三个控制位以正和反两种形式存在于存取控制字节中，决定了该块的访问权限（如： 进行减值操作必须验证KEY A，进行加值操作必须验证KEY B，等等）。因为控制字是由部分控制位取反的来的，由上表C33\_b就可知道，所以我们要再取反一次获取原来的控制位，就能得出下面的控制位（\_b为取反，比如：C33为1取反后为C33\_b为0）：

字节6——FF ：  0 0 0 0    0 0 0 0

字节7——07 ：   0 0 0 0   1 0 0 0

字节8——80 ：  1 0 0 0    0 0 0 0

字节9——69 ：（这个一般不用）

在****存取控制****中每个块都有相应的****三个控制位****,定义如下：

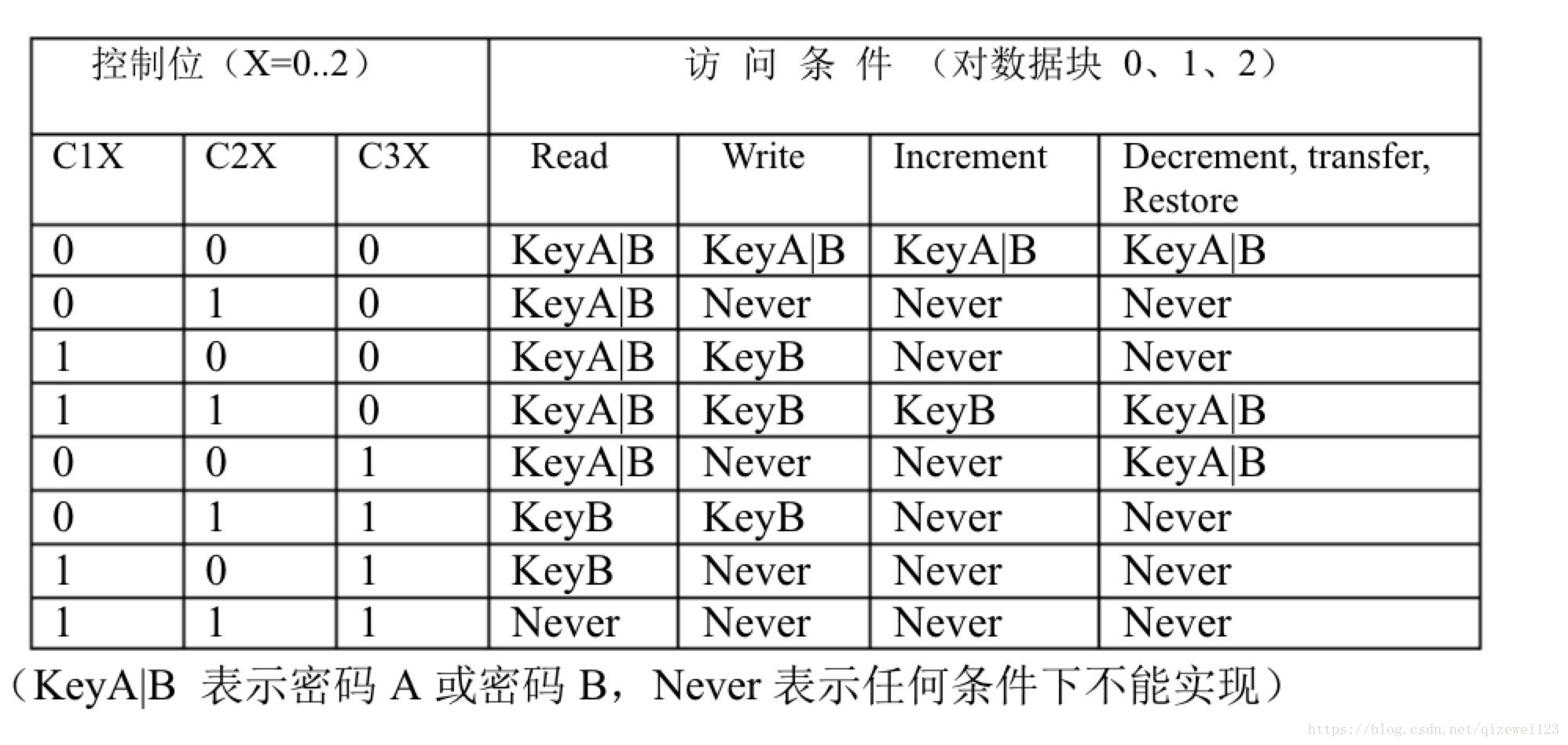
块0：  C10  C20  C30                                       0      0      0

块1：  C11  C21  C31                                       0      0      0

块2：  C12  C22  C32       对应上表数据可得——      0      0      0

块3：  C13  C23  C33                                       0      0      1

块0，1，2控制位权限参照下面权限表可知：



块0控制位为：0 0 0       权限为： 通过A或者B密码认证后可读，可写，可进行加值和减值操作。

块1控制位为：0 0 0       权限为： 通过A或者B密码认证后可读，可写，可进行加值和减值操作。

块2控制位为：0 0 0       权限为： 通过A或者B密码认证后可读，可写，可进行加值和减值操作。

块3和其他的权限参照表是不一样的，块3对应的权限表为下表：



块3控制位为： 0 0 1      权限为：A密码不可读，验证A或者B密码后可改写A密码。验证A或者B密码后，可读可改写存取控制 。

验证A密码或者B密码后，可读可改写B密码。

这个就是M1卡出厂默认控制位各块的权限，那么如果我们要改写厂家的默认控制位自定义自己的控制位要怎么做呢？

有两个方法：

1.通过上面的步骤，根据自己的需求，逆推出控制位。（当然这个比较难，下面我会讲解）

2.通过下载一些转换工具进行转换。（相对简单，但是知其然不知其所以然）

先说一下简单的吧，毕竟开发都是以效率为先的。

方法2：

我们先去下载一个转换工具：<https://www.xzking.com/html/53519.html>

解压后打开界面，如下图：



        根据自己的需求，对应0,1,2块权限对照表获取对应块的控制位，进行修改数据块。然后根据块3权限对照表获取对应块的控制位，进行修改密码块。接下来就是点击生成控制字，就可以得到你想要的控制字，把它替换默认的控制字就可以了。这个就是使用工具来直接生成控制字的方法，非常的简单迅速。满足了开发需求了，那我们就可以去深究一下这个软件到底是怎么根据我们的权限选择进行逆推我们的控制字。

方法1：

         首先根据我们的需求，对应块0,1,2权限表获取我们想要的权限的控制位，比如：

块0控制位为：1 1 0       权限为： 验证A密码或者B密码可读，验证B密码可写，验证B密码可进行加值操作，验证A密码或者B密                                                        码可进行减值操作。

块1控制位为：1 0 0       权限为： 验证A密码或者B密码可读，验证B密码可写，不可进行加值和减值操作。

块2控制位为：1 0 1       权限为： 验证A密码可读，不可写，不可进行加值和减值操作。

        接着根据我们的需求，对应块3权限表获取我们想要的权限的控制位，比如：

块3控制位为:  0 0 1        权限为： A密码不可读，验证A密码或者B密码可以改写A密码。验证A密码或者B密码可读可改写控制                                                            字。验证A密码或者B密码可以读可以改写B密码。

这样我们自定义的权限的各块控制位就知道是什么值了，同时因为我们知道在****存取控制****中每个块都有相应的****三个控制位，****排列一下：

块0：  C10  C20  C30      1      1      0

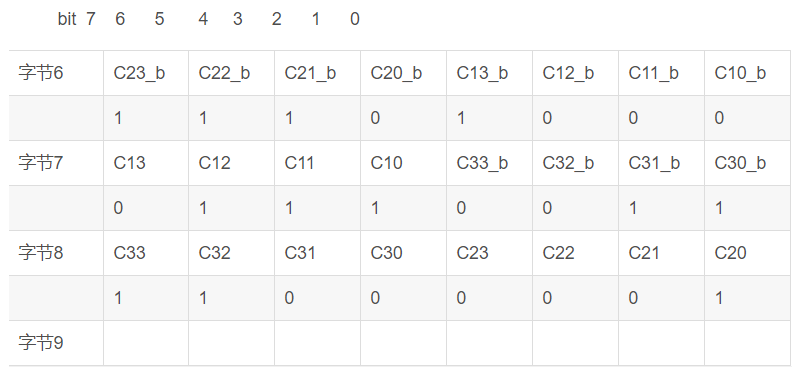
块1：  C11  C21  C31      1      0      0

块2：  C12  C22  C32       对应控制位表数据可得——  1      0      1

块3：  C13  C23  C33    0      0      1

根据存取控制字节对应填入下表后（\_b为取反，比如：C33为1取反后为C33\_b为0）

存取控制（4字节，其中字节9为备用字节）结构如下所示：





再把备用的字节9（默认69）连接起来就可以得到我们想要的控制字：E8 73 C1 69。让我们检验一下我们转换的是不是正确的，在方法2的工具下输入我们各块的权限对应的控制位，再点击生成控制字，结果如下：



完全正确！！这样我们就可以随心所欲的去设置自己想要的效果了，原理也是知道了。接下来就是把自己自定义的控制字写入M1卡了就行了，大大提高了M1卡的保密性。本篇涉及到了M1卡的基本原理和复制卡的基本原理.