费率设置卡参数说明

2007-1-18

根据客户的需求，要对 256 类卡进行不同的权限设置，以至可以实现不同的卡类可按不同的收费标准进行收费。现对 M1 卡片结构制定如下：

一、设置卡类型存储位置及说明

第 0 扇区

块号	读写条件	序号	地址	长度	内　容	说　明
0 块	KEYA 读	1	0--15	16	卡片物理序列号等信息	系统保留
1 块（设置卡类型及钱包号）	KEYA 读	1	0-3	4	“LYCS”或“CKSJ”或“LYCJ”	“LYCS”约定为费率设置卡 “CKSJ”约定为常开设置卡 “LYCJ”约定为采集卡
		2	4	1	使用的淋浴钱包号	活动钱包号 1----8（是费率设置卡时，该数据有效）
		3	5	1	计费器模式	FF 为大浴室卡，为公寓卡
		4	6--14	9	保留	可全为 FF
		5	15	1	校验字节 BCC	地址（0---15）内数据字节之累加和
2 块（主工作密钥及淋浴钱包密钥）		1	0—7	8	主工作密钥	访问卡片（淋浴钱包扇区除外）所需要的密钥（明文），仅费率设置卡才有此数据.
		2	8—15	8	小钱包工作密钥	访问淋浴钱包所需要的密钥（明文）仅费率设置卡才有此数据.

二、费率参数定义

费率参数从第 1 扇区开始存放，每扇区使用块 0，1，2。可根据所需要设置的卡类权限数量选择 S50 卡或 S70 卡片做为设置卡.

1、费率设置定义			
序号	长度	内　容	说　明
1	32	权限位图	每个 bit 代表一个权限(1 为启用，0 为禁用) 第 0 字节 BIT7 表示第 256 类卡的权限 第 31 字节的 BIT0 表示 1 类卡的权限 以此类推.

2	2	数据区长度	代表数据区的长度， 高位在前，低位在后
3	不定长	c 费率数据区(变长)	按照前边 32 个字节的位图，组织的具体的费率

2、费率数据区定义(变长)			
序号	长度	内 容	说 明
1	1	权限 1 费率个数(1 字节)，n1	当前权限下允许多少个费率，最多 5 个
2	n1*3	权限 1 具体费率	(费率个数*3 字节)
3	1	权限 2 费率个数(1 字节)，n2	当前权限下允许多少个费率，最多 5 个
4	n2*3	权限 2 具体费率	(费率个数*3 字节)
5	1	权限 3 费率个数(1 字节)，n3	当前权限下允许多少个费率，最多 5 个
6	n3*3	权限 3 具体费率	(费率个数*3 字节)
7	1	权限 x 费率个数(1 字节)，nx	当前权限下允许多少个费率，最多 5 个
8	nx*3	权限 x 具体费率	(费率个数*3 字节)
9	1	校验和	所有数据区的累加和

三、费率数据说明

序号	地址	长度	格式	内容说明
1	0	1	HEX	时间阶梯 T⊿（分钟），表示从第 T⊿分钟开始按此费率扣费
2	1	1	HEX	扣费的单位时间，以秒为单位
3	2	1	HEX	扣费的单位金额，以分为单位
4	共 3 字节			

四、卡片访问密码说明

设置卡访问密码为 KEYA= “物理卡号 4 字节” + BCC (BCC 为物理卡号的累加和(两字节))

KEYA 的产生方法如下.

```
/*
//Srn 为卡片序列号.
//KeyA 为卡片的密码 A.

void GenerateKeyA(byte Srn[4],unsigned char KeyA[6])
{
    unsigned char i;
    unsigned int BCC;
```

```
    BCC = 0;
    for(i=0;i<4;i++)
    BCC += Srn[i];

    memcpy(KeyA,Srn,4);
    KeyA[4] =(unsigned char)(BCC>>8);
    KeyA[5]  =(unsigned char)BCC ;
    return;
}
```

数据采集卡结构

采集卡卡片结构制定如下：

序号	扇区	块号	地址	长度	格式	内容说明
1	0	1	0 - 3	4	ASCII 码	约定字符 “LYCJ”
2	1	0	0-7	8	HEX	计费器 1 数据
3		0	8-15	8	HEX	计费器 2 数据
4		1	0-7	8	HEX	计费器 3 数据
5		1	8-15	8	HEX	计费器 4 数据
6		2	0-7	8	HEX	计费器 5 数据
7		2	8-15	8	HEX	计费器 6 数据
3	2-15					计费器 7-90 数据

一、数据说明

计费器 N 数据：存储某台计费器累计的消费总额，格式如下：

4 字节计费器终端 ID 号+4 字节累计消费总额（低字节在前）

二、使用方法

1、 管理软件按上述结构发行采集卡：先改写密钥，然后在 0 扇区 1 块写入约定数据表明用途，最后将 1-15 扇区数据全部写成 0。

2、 采集卡在计费器上采集数据：计费器检测到是采集卡后，在卡片的第 1-15 扇区的数据存储区内搜索未存储的区域（8 个字节为全 0），找到后将计费器的终端 ID 号及累计消费总额写入采集卡，若没有搜索到未存储的区域，显示空间已满。

3、 采集卡在管理软件上录入数据：管理软件登陆卡片后搜索 1-15 扇区的数据区域，读出全部非空数据存储区域的数据，保存好后再将 1-15 扇区数据全部写成 0，这样采集卡就可以按步骤 2 再次到计费器上采集数据了。

采集卡接口:

int __stdcall InitNewCollectionCard(int *err_code, LPProcessCallBack lpCallBack)

函数说明: 发行采集卡

输入参数: err_code: 获取错误码

lpCallBack: 回调函数地址

回调函数原型 typedef void (CALLBACK * LPProcessCallBack)(int step);

返回值: 返回 0 表示成功, 返回-1 表示错误

int __stdcall RefineWaterCard(int *err_code, LPProcessCallBack lpFunc)

函数说明: 回收采集卡

输入参数: err_code: 获取错误码

lpCallBack: 回调函数地址

回调函数原型: typedef void (CALLBACK * LPProcessCallBack)(int step);

返回值: 返回 0 表示成功, 返回-1 表示调用失败

费率设置卡接口:

int __stdcall InitNewFeeRateCard(int *err_code, LPProcessCallBack lpCallBack)

函数说明: 发行费率设置卡

输入参数: err_code: 获取错误码

lpCallBack: 回调函数地址 ;

回调函数原型: typedef void (CALLBACK * LPProcessCallBack)(int step);

返回值: 返回 0 表示成功, 返回-1 表示调用失败

int __stdcall RefineWaterCard(int *err_code, LPProcessCallBack lpFunc)

函数说明: 回收费率设置卡

输入参数: err_code: 获取错误码

lpCallBack: 回调函数地址

回调函数原型: typedef void (CALLBACK * LPProcessCallBack)(int step);

返回值: 返回 0 表示成功, 返回-1 表示调用失败

int __stdcall PublishFeeRateCard(FEE_RATE_CARD_INFO *fee_rate_card_info)

函数说明: 发行费率设置卡

输入参数: fee_rate_card_info: 费率设置结构体

```
#pragma pack(1)
/*****************************************************/
// 费率结构
/*****************************************************/
typedef struct
{
    BYTE time_ladder[1];            // 时间阶梯
    BYTE deduct_time[1];            // 扣费的单位时间，以秒为单位
    BYTE deduct_fee[1];             // 扣费的单位金额，以分为单位
```

```
}FEE_RATE;

/*******************************************************/
// 当前权限的费率个数
/*******************************************************/
typedef struct
{
    short right_flag;                   // 权限标志, 1 打开, 0 关闭
    short right_num;                    // 传入当前权限下的费率个数
    FEE_RATE fee_rate[5];               // 最大设置为 5 个, 其实有一些没有用到
}FEE_RIGHT_NO;

/*********************************************************/
// 费率卡结构说明
/*********************************************************/
typedef struct
{
    BYTE water_card_flag[5];            // 水控卡标志
    char packet_num;                    // 水控钱包号
    BYTE main_work_key[9];              // 主工作密钥(明文)
    BYTE packet_work_key[9];            // 小钱包工作密钥(明文)
    BYTE data_length[3];                // 数据长度
    BYTE check_crc[2];                  // 效验字节
    FEE_RIGHT_NO fee_right_num[256];// 设置费率的个数
}FEE_RATE_CARD_INFO;
#pragma pack()

int __stdcall CollectionCardReceiveData(int *err_code, BYTE shop_id[360], BYTE
shop_card_total_sum[360])
函数说明: 对采集卡中数据进行采集
输入参数: err_code: 错误返回码
          shop_id:  设备 ID 号
          shop_card_total_sum: 商户采集总额
返回值: 0 表示正常返回, 否则表示错误返回
```