

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	1/52

## CRT-310（004）读卡器动态库接口函数说明(NET 版)

### 目 录

1. CRT 读卡器动态库文件说明	2
2. 动态库版本函数	2
3. 串口操作基本函数	3
4. CRT-310 专用函数	5
5. 磁卡函数	9
6. Mafare 1 函数	11
7. RF CPU 卡函数函数 (TYPE A/B)	20
8. IC 卡操作公用函数	22
9. CPU 卡函数 (T=0/T=1)	24
10. SIM 卡函数 (T=0/T=1)	28
11. SLE4428 函数	30
12. SLE4442 函数	33
13. 24Cxx 系列卡函数	36
14. AT88SC102 函数	38
15. AT88SC1604 函数	43
16. AT88SC1608 函数	44
17. AT45D041 函数	52

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	2/52

////////////////////CRT 读卡器动态库文件说明////////////////////////////////////

文件名称: CRT\_310.dll

内部版本: v20110729

功能简述: 针对 CRT 读卡器系列产品开发, 完成磁卡、射频卡和常规的几种接触式 IC 卡、CPU 卡的读写操作。

支持卡类型:

- 飞利浦 M1 射频卡
- SLE4428、SLE4442
- 24C01、24C02、24C04、24C08、24C16、24C32、24C64
- AT88SC102、AT88SC1604、AT45D041、AT88SC1608
- 接触式 CPU 卡 (T=0/T=1)
- 非接触式 CPU 卡 (TYPE=A/TYPE=B)
- SAM/SIM 卡 (T=0/T=1)
- 磁卡

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	3/52

//////////////////////////////////////动态库版本函数//////////////////////////////////////

获取内部版本信息。

**1. int APIENTRY GetSysVerion(HANDLE ComHandle, char \*strVerion);**

//参数:

// ComHandle: 串口句柄。

// strVerion: 版本信息字符串。

//返回:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

//////////////////////////////////////

获取读卡器命令执行失败(返回-1 时)的错误原因。

**2. int APIENTRY GetErrCode(int \*Errorcode);**

//参数:

// Errorcode: 最近一次执行命令失败的错误代码

错误代码请参考下表(红色为建议):

Errorcode	通讯协议 (非正常返回)	描述
-1		通讯失败==>请检查电源、通讯线、串口设置是否正常
-2		卡型错误
-101		串口打开错误或没有打开==>请执行 CommOpen 或 CommOpenWithBaut 来打开串口
-102		串口参数设置错==>如果通讯波特率不使用 9600bps, 建议用 CommOpenWithBaut 来打开串口
-200	0	未定义的命令参数
-202	1	命令不能执行(机型硬件限制)==>请确认所用机器型号是否正确, 具体看产品说明书中型号说明
-204	2	命令数据错误
-205	3	输入电源电压(低于 10.8V 或高于 14.5V)不在卡机工作范围内==>请确认电源为 DC12V
-206	4	卡机内有非标准长度的异常长度卡(短卡或长卡)
-207	5	当前卡机主电源处于掉电状态, 命令不能执行==>请接主电源为 DC12V

//返回:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	4/52

////////////////////////////////////串口操作基本函数////////////////////////////////////

打开串口（波特率为 9600bps）

**1. HANDLE WINAPI CommOpen(char \*Port);**

//参数:

//Port:串口号字符串

//例如:CommOpen("Com1");

//返回:

//串口文件句柄

//备注:必须先调用此函数，获得指定串口的串口文件句柄，才可调用其他函数。

// 可以同时打开多个串口，获得多个串口文件句柄，但不能多次打开同一个串口。

// 使用完毕后，必须调用 CommClose()关闭串口。

////////////////////////////////////

按指定的波特率打开串口

**2. HANDLE WINAPI CommOpenWithBaud(char \*Port, BYTE \_BaudOption);**

//参数:

//Port:串口号字符串

//\_BaudOption=

// 1=> 波特率为 1200

// 2=> 波特率为 2400

// 3=> 波特率为 4800

// 4=> 波特率为 9600

// 5=> 波特率为 19200

// 6=> 波特率为 38400

// 7=> 波特率为 57600

//例如:CommOpen("Com1",4);

//返回:

//串口文件句柄

//备注:必须先调用此函数，获得指定串口的串口文件句柄，才可调用其他函数。

// 可以同时打开多个串口，获得多个串口文件句柄，但不能多次打开同一个串口。

// 使用完毕后，必须调用 CommClose()关闭串口。

////////////////////////////////////

关闭指定串口

**3. int WINAPI CommClose(HANDLE ComHandle);**

//参数:

// ComHandle: 串口句柄。

//返回:

//0=成功

//备注:与 CommOpen()函数配套使用，并且在使用串口完毕后必须调用此函数关闭串口。



	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	6/52

////////////////////// CRT-310 卡机操作函数 //////////////////////////////////////

CRT310 复位读卡机。

1. int APIENTRY CRT310\_Reset(HANDLE ComHandle, BYTE \_Eject);;

//参数:

// ComHandle: 串口句柄。

// \_Eject:弹卡选择。 0=不弹卡 1=前端弹卡 2=后端弹卡

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////////

//读卡机序列号

2. int APIENTRY CRT310\_ReadSnr(HANDLE ComHandle, BYTE \_SNData[], BYTE \*\_dataLen)

//参数:

// ComHandle: 串口句柄。

// \_SNData: 卡机序列号。

// \_dataLen: 数据长度。

//返回值:

// 如果函数调用成功, 返回值为 0。数组\_SNData 中存放卡机序列号。

\_dataLen 中存放卡机序列号数据长度

// 如果函数调用失败, 返回值不为 0。

注: 卡机序列号数据包的长度 (0 ~ 16 byte)

卡序列号数据包: 以 A S C I I 码存贮:

如卡机的序列号的数据为 “CRT-310” 则卡机序列号数据包为:

“0x43,0x52,0x54,0x2D,0x33,0x31,0x30”

////////////////////////////////////

//写卡机序列号

3. int APIENTRY CRT310\_WriteSnr(HANDLE ComHandle, BYTE \_SNData[], BYTE \_dataLen)

//参数:

// ComHandle: 串口句柄。

// \_SNData: 写入的卡机序列号。

// \_dataLen: 写入的卡机序列号数据长度。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

注: 卡机序列号数据包的长度 (0 ~ 16 byte)

卡序列号数据包: 以 A S C I I 码存贮:

如卡机的序列号的数据为 “CRT-310” 则卡机序列号数据包为:



## SPECIFICATION

Model No.

## CRT-310 读卡器

Date \_\_\_\_\_

2011/07/29

Ver.

v20110729

Page

8/52

## 动态库说明

=0X49            卡机前端持卡位置有卡。

=0X4A            卡机内停卡位置有卡。

=0X4B 卡机内 IC 卡操作位置有卡，并且 IC 卡触点已下落。

=0X4C            卡机后端持卡位置有卡。

=0X4D            卡机后端不持卡位置有卡。

=0X4E            卡机内无卡。

```
// _frontSetting:      =0X49          卡机允许磁信号方式进卡, 只允许磁卡开闸门进卡。
```

=0X4A	卡机允许开关信号方式进卡，允许磁卡，IC 卡，M1 射频卡，双界面卡进卡。
-------	---------------------------------------

=0X4B	卡机允许磁信号方式进卡，允许纸磁卡，薄卡进卡。
-------	-------------------------

=0X4E      卡机禁止进卡。

```
//    _rearSetting:      =0X4A      卡机允许后端进卡, 允许磁卡, IC 卡, M1 射频卡, 双面卡进卡。
```

=0X4E            卡机禁止后端进卡。

//返回值:

```
// 如果函数调用成功, 返回值为 0。
```

```
// 如果函数调用失败，返回值不为 0。
```

////////////////////////////////////

### 读 CRT310 感应器状态

```
7.  int APIENTRY CRT310_SensorStatus (HANDLE ComHandle, BYTE *_PSS0, BYTE *_PSS1, BYTE *_PSS2, BYTE *_PSS3,
    BYTE *_PSS4, BYTE *_PSS5, BYTE *_CTSW, BYTE *_KSW)
```

//参数:

```
// ComHandle: 串口句柄。
```

\_PSS0—PSS5: 红外传感器状态: PSSx=0X30 表示此传感器位置上未探测到卡片;

PSSx=0X31 表示探测到有卡片。

\_CTSW: 闸门状态信息 CTSW=0X30 表示闸门已关闭;

CTSW=0X31 表示闸门已打开。

KSW : 开关进卡传感器状态 KSW=0X30 表示开关没有检测到卡片插入闸门信号;

KSW=0X31 表示开关检测到有卡片插入闸门。

```
//返回值:
```

```
// 如果函数调用成功, 返回值为 0。
```

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////////

CRT310 走卡位置设置。

**8. int APIENTRY CRT310 MovePosition(HANDLE ComHandle, BYTE Position)**

```
//参数:
```

```
// ComHandle: 串口句柄。
```

```
// Position:
```

=0x1 将卡重新走位到前端位置，不持卡。

=0x2 将卡重新走位到前端位置，并持卡。





	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	10/52

//////////////////////////////////// 磁卡函数 //////////////////////////////////////

读磁轨数据。

1. int APIENTRY MC\_ReadTrack(HANDLE ComHandle, BYTE \_mode, BYTE \_track, int \* \_BlockDataLen, BYTE \_BlockData[])

//参数:

// ComHandle: 串口句柄。

// \_mode: 数据模式。

读卡模式: 0x30 以 ASCII 码读卡数据  
0x31 以二进制码读卡数据

// \_track: 磁轨。

指定轨道号: 0x30 磁卡三轨都不读  
0x31 读磁卡一轨  
0x32 读磁卡二轨  
0x33 读磁卡三轨  
0x34 读磁卡一二轨  
0x35 读磁卡二三轨  
0x36 读磁卡一三轨  
0x37 读磁卡一二三轨

// \_BlockDataLen: 数据包长度。

// \_BlockData: 数据内容。

其中每轨数据包格式如下:

轨数据起始字+读卡状态字+卡轨道数据

轨道起始字: 0x1F

读卡状态字: 0x59 读该轨数据读正确, 卡轨道数据为该轨信息数据  
0x4E 读卡不正确, 卡轨道数据为错误信息  
0x4F 该轨道不读, 卡轨道数据为 0xE0;

错误信息:

0xE1 读该轨数据错误, 没有起始位 STX  
0xE2 读该轨数据错误, 没有结束位 ETX  
0xE3 读该轨数据错误, 位校验错误 VRC  
0xE4 读该轨数据错误, 字节校验位错误 LRC  
0xE5 读该轨数据错误, 该轨是空白信息磁道

\*二进制读卡传送的数据格式是:

一轨: b0,b1,b2,b3,b4,P

二轨,三轨: b0,b1,b2 b3,P

\*注意:

当设置以 ASCII 码读卡时将卡每一轨信息的别换成一个字节 ASCII 码上传读卡数据。

如: 一轨数据第一字节为: 0x03 (HEX)

上传数据时卡轨道数据包为: 0x33 (ASCII)

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	11/52

当设置成二进制读卡时将卡每一轨信息的每一字节数据按每 4 位转成一个字节以 ASCII 码形式上传数据。

如：一轨数据第一字节为：0x03 (HEX)

上传数据时卡轨道数据包为：0x30 0x3

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值为非 0。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	12/52

//////////////////////////////// Mafare 1 函数////////////////////////////////

检测是否是 M1 卡片。

1. int APIENTRY RF\_DetectCard(HANDLE ComHandle);

//参数:

// ComHandle: 串口句柄。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E) 寻卡不成功

P= 'E' (0X45) 卡机内无卡

P= 'W' (0X57) 卡不在允许操作的位置上。

////////////////////////////////

获取 M1 卡片的序列号。

2. int APIENTRY RF\_GetCardID(HANDLE ComHandle, BYTE \*\_CardIDLen, BYTE \_CardID[]);

//参数:

// ComHandle: 串口句柄。

// \_CardIDLen:序列号长度

// \_CardID[]:M1 卡片号码

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E) 获取卡序列号失败, 并返回空序列号 (0X00, 0X00, 0X00, 0X00)

P= 'E' (0X45) 卡机内无卡

////////////////////////////////

验证扇区的密码。

3. int APIENTRY RF\_LoadSecKey(HANDLE ComHandle, BYTE \_Sec, BYTE \_KEYType, BYTE \_KEYLen, BYTE \_KEY[]);

//参数:

// ComHandle: 串口句柄。

// \_Sec: 扇区号。

// \_KEYType: 密码类型选择。0=KEYA; 1=KEYB

// \_KEYLen: 密码长度固定为 6。

// \_KEY[]: 密码(6 个字节)。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= '0' (0X30) 寻不到射频卡

P= '3' (0X33) 密码错误

P= 'E' (0X45) 卡机内无卡



////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	15/52

写扇区指定块的数据。

7. `int APIENTRY RF_WriteBlock(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _BlockDataLen, BYTE _BlockData[]);`

//参数:

// ComHandle: 串口句柄。

// \_Sec: 扇区号。

// \_Block: 块号。

// \_BlockDataLen: 要写入的块数据包长度, 固定为 16

// \_BlockData[]: 数据块内容(16 字节)。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

- P= '0' (0X30) 寻不到 RF 卡
- P= '1' (0X31) 操作扇区号错 (不是验证密码后的扇区)
- P= '2' (0X32) 操作的卡序列号错
- P= '3' (0X33) 密码验证错
- P= '4' (0X34) 校验写入块数据错
- P= 'E' (0X45) 卡机内无卡
- P= 'W' (0X57) 卡不在允许操作的位置上。

注: 扇区号= 0x00 0x01 0x02 .....0x28

S50 卡片扇区号是 0x00 0x01 0x02 .....0x0F, S70 卡片扇区号是 0x00 0x01 0x02 .....0x28

块号= 0x00 0x01 0x02 .....0x0F

S50 卡片每个扇区有 4 个地块, 块号分别是 0x00 0x01 0x02 0x03,

S70 卡片第 0-31 扇区中每一扇区有 4 个块, 块号分别是 0x00 0x01 0x02 0x03,

第 32-39 扇区每一扇区有 16 个块, 块号分别是 0x00 0x01 0x02.....0x0F

注: 在进行块写操作操作时, S50, S70 第 0-31 扇区中每个扇区的第 3 块, S70 第 32-39 扇区中第 15 块是 KEYA、控制字、KEYB 的存储区域, 应注意每个扇区最后一块不要随意写

////////////////////////////////////

写扇区多块的数据。

8. `int APIENTRY RF_WriteMoreBlock(HANDLE ComHandle, BYTE _Sec, BYTE _StartBlock, BYTE _BlockNumber, BYTE _BlockDataLen, BYTE _BlockData[]);`

//参数:

// ComHandle: 串口句柄。

// \_Sec: 扇区号。

// \_StartBlock: 起始块号。

// \_BlockNumber: 块数。

// \_BlockDataLen: 写块数据包长度, 长度为 16\*\_BlockNumber

// \_BlockData[]: 数据块内容(16\*\_BlockNumber 字节)。

P= 'W' (0X57)      卡不在允许操作的位置上。



	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	17/52

//注意：在进行‘读值/增值/减值操作前，首先要使用该函数进行指定扇区内数据块的初始化，否则会产生错误！

初始化值：按 MIFARE 卡块数据格式进行写入 16 byter 数据，其格式如下：

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Value				/Value				Value				Adr	/Adr	Adr	/Adr

Value: 要初始化 4 byte 值（初始化函数初始值），注值的低字节在前，高字节在后

/Value: 要初始化 4 byte 值取反

Adr: 所要初始化值的块地址：Adr= 扇区号 X 4 + 块号

/Adr: 所要初始化值的块地址的取反

例：将第五扇区块 0 初始化为 10，所要写入 16 byte 扇区块数据为：(.net 用户请看例子源程序)

“ 0x0A, 0x00, 0x00, 0x00, 0xF5, 0xFF,0xFF ,0xFF , 0x0A, 0x00, 0x00, 0x00, 0x14, 0xEB, 0x14, 0xEB ”

```
int P;
BYTE value[4];
memset(value,0,4);
value[0]=0xA;
P=RF_InitValue(hCom,5,0,value);
```

注：在进行值操作时，S50，S70 第 0-31 扇区中每个扇区的第 3 块，S70 第 32-39 扇区中第 15 块是 KEYA、控制字、KEYB 的存储区域，是不能作值段数据存贮。初始化值，增值，减值，读值时应注意操作扇区块地址范围。

////////////////////////////////////

读值。

10. int APIENTRY RF\_ReadValue(HANDLE ComHandle, BYTE \_Sec, BYTE \_Block, BYTE \* \_DataLen, BYTE \_Data[]);

//参数:

```
// ComHandle: 串口句柄。
// _Sec: 扇区号。
// _Block: 块号。
// _DataLen: 读出的数据长度，固定为 4。
// _Data: 读出的值，4 个字节。
```

//返回值 P:

```
// 如果函数调用成功，返回值为 0。
// 如果函数调用失败，返回值不为 0。
P= '0' (0X30)      寻不到 RF 卡
P= '1' (0X31)      操作扇区号错（不是验证密码后的扇区）
P= '2' (0X32)      操作的卡序列号错
P= '3' (0X33)      密码验证错
P= '4' (0X34)      块数据格式错误（该块存贮数据没有写成值数据形式）
P= '5' (0X35)      增值溢出
P= 'E' (0X45)      卡机内无卡
P= 'W' (0X57)      卡不在允许操作的位置上。
```

注 1：在进行读值操作前，首先要使用初始化函数 RF\_InitValue，否则会产生错误！

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	18/52

注 2: 扇区号= 0x00 0x01 0x02 .....0x28

S50 卡片扇区号是 0x00 0x01 0x02 .....0x0F, S70 卡片扇区号是 0x00 0x01 0x02 .....0x28

块号= 0x00 0x01 0x02 .....0x0F

S50 卡片每个扇区有 4 个地块，块号分别是 0x00 0x01 0x02 0x03，

S70 卡片第 0-31 扇区中每一扇区有 4 个块，块号分别是 0x00 0x01 0x02 0x03，

第 32-39 扇区每一扇区有 16 个块, 块号分别是 0x00 0x01 0x02.....0x0F

注 3: 在进行值操作时, S50, S70 第 0-31 扇区中每个扇区的第 3 块, S70 第 32-39 扇区中第 15 块是 KEYA、控制字、KEYB 的存储区域, 是不能作值段数据存储。初始化值, 增值, 减值, 读值时应注意操作扇区地址范围。

////////////////////////////////////

增值。

```
11. int APIENTRY RF Increment(HANDLE ComHandle, BYTE _Sec, BYTE _Block, BYTE _DataLen, BYTE _Data[]);
```

//参数:

// ComHandle: 串口句柄。

// Sec: 扇区号。

// Block: 块号。

```
// DataLen: 要写入的数据长度, 固定为 4
```

// Data: 数值, 4 个字节。

//返回值 P:

```
// 如果函数调用成功, 返回值为 0。
```

// 如果函数调用失败, 返回值不为 0。

P= '0 '(0X30) 寻不到 RF 卡

P= '1' (0X31)      操作扇区号错 (不是验证密码后的扇区)

P= '2' (0X32)            操作的卡序列号错

P= '3' (0X33)      密码验证错

P= '4' (0X34)      块数据格式错误 (该块存贮数据没有写成值数据形式)

P= '5' (0X35)      增值溢出

P= 'E' (0X45)      卡机内无卡

P= 'W' (0X57)      卡不在允许操作的位置上。

注 1: 在进行读值操作前, 首先要使用初始化函数 RF\_InitValue, 否则会产生错误!

注 2: 扇区号= 0x00 0x01 0x02 .....0x28

S50 卡片扇区号是 0x00 0x01 0x02 .....0x0F, S70 卡片扇区号是 0x00 0x01 0x02 .....0x28

块号= 0x00 0x01 0x02 .....0x0F

S50 卡片每个扇区有 4 个地块, 块号分别是 0x00 0x01 0x02 0x03,

S70 卡片第 0-31 扇区中每一扇区有 4 个块，块号分别是 0x00 0x01 0x02 0x03，

第 32-39 扇区每一扇区有 16 个块, 块号分别是 0x00 0x01 0x02.....0x0F

注 3: 在进行值操作时, S50, S70 第 0-31 扇区中每个扇区的第 3 块, S70 第 32-39 扇区中第 15 块是 KEYA、控制字、KEYB 的存储区域, 是不能作值段数据存储。初始化值, 增值, 减值, 读值时应注意操作扇区块地址范围。

////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	19/52

减值。

12. int APIENTRY RF\_Decrement(HANDLE ComHandle, BYTE \_Sec, BYTE \_Block, BYTE \_DataLen, BYTE \_Data[]);

//参数:

// ComHandle: 串口句柄。  
// \_Sec: 扇区号。  
// \_Block: 块号。  
// \_DataLen: 要写入的数据长度, 固定为 4  
// \_Data: 数值, 4 个字节。

//返回值 P:

// 如果函数调用成功, 返回值为 0。  
// 如果函数调用失败, 返回值不为 0。  
P= '0' (0X30) 寻不到 RF 卡  
P= '1' (0X31) 操作扇区号错 (不是验证密码后的扇区)  
P= '2' (0X32) 操作的卡序列号错  
P= '3' (0X33) 密码验证错  
P= '4' (0X34) 块数据格式错误 (该块存贮数据没有写成值数据形式)  
P= '5' (0X35) 增值溢出  
P= 'E' (0X45) 卡机内无卡  
P= 'Y' (0X59) 操作成功  
P= 'W' (0X57) 卡不在允许操作的位置上。

注 1: 在进行读值操作前, 首先要使用初始化函数 RF\_InitValue, 否则会产生错误!

注 2: 扇区号= 0x00 0x01 0x02 .....0x28

S50 卡片扇区号是 0x00 0x01 0x02 .....0x0F, S70 卡片扇区号是 0x00 0x01 0x02 .....0x28


块号= 0x00 0x01 0x02 .....0x0F

S50 卡片每个扇区有 4 个地块, 块号分别是 0x00 0x01 0x02 0x03,

S70 卡片第 0-31 扇区中每一扇区有 4 个块, 块号分别是 0x00 0x01 0x02 0x03,

第 32-39 扇区每一扇区有 16 个块, 块号分别是 0x00 0x01 0x02.....0x0F

注 3: 在进行值操作时, S50, S70 第 0-31 扇区中每个扇区的第 3 块, S70 第 32-39 扇区中第 15 块是 KEYA、控制字、KEYB 的存储区域, 是不能作值段数据存贮。初始化值, 增值, 减值, 读值时应注意操作扇区块地址范围。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	20/52

## ////////////////////RF CPU 卡函数 (TYPE=A / TYPE=B) //////////////////////

注： 1>建议先自动测卡型，得到卡类型（type A 或 type B）后才进行其它命令

2>必须先寻卡成功后，才能执行 APDU 命令，

获取 CPU 卡片的序列号(只有 TYPE A 才有)。

```
1. int APIENTRY CPU_GetRfCardID(HANDLE ComHandle, BYTE *_CardIDLen, BYTE _CardID[]);
```

//参数:

// ComHandle: 串口句柄。

// \_CardIDLen:序列号长度，固定为 4

// \_CardID[]:M1 卡片号码(4 个字节)。

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

P= 'N' (0X4E) 获取卡序列号失败，并返回空序列号 (0X00, 0X00, 0X00, 0X00)

P= 'E' (0X45) 卡机内无卡

////////////////////////////////////

寻 RF CPU 卡(卡片复位)

```
2. int APIENTRY CPU_SelRfCard(HANDLE ComHandle,BYTE _CPUType,BYTE _exData[], int *_exdataLen);
```

//参数:

// ComHandle: 串口句柄。

// \_CPUType: CPU 卡类型

=0x0 TYPE A

=0x1 TYPE B

// \_exData: 返回数据。

// \_exdataLen: 返回数据的长度。

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

= 'N' (0X4E) 操作不成功

= 'E' (0X45) 卡机无卡

////////////////////////////////////

CPU 卡 C-APDU 命令。

```
3. int APIENTRY CPU_SendRfAPDU(HANDLE ComHandle, int _dataLen, BYTE _APDUData[], BYTE _exData[], int *_exdataLen);
```

//参数:

// ComHandle: 串口句柄。

// \_dataLen: C-APDU 命令长度。

// \_APDUData: CUP 卡 C-APDU 命令。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	21/52

```
// _exData: 返回数据。
// _exdataLen: 返回数据的长度。
//返回值:
// 如果函数调用成功, 返回值为 0。
// 如果函数调用失败, 返回值不为 0。
//   = 'N' (0X4E)  操作不成功
//   = 'E' (0X45)  卡机无卡
/////////////////////////////////////////////////////////////////
释放 RF CPU 卡
4.  int APIENTRY CPU_DESelRfCard(HANDLE ComHandle);
//参数:
//   ComHandle: 串口句柄。
//返回值:
// 如果函数调用成功, 返回值为 0。
// 如果函数调用失败, 返回值不为 0。
```

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	22/52

////////////////////////////////// IC 卡操作的公用函数 //////////////////////////////////////

应用程序在操作接触式 IC 卡之前，需调用此函数给卡片上电。

**1. int APIENTRY CRT\_IC\_CardOpen(HANDLE ComHandle);**

//参数:

// ComHandle: 串口句柄。

//返回值:

// 如果函数调用成功，返回值为 0，

// 如果函数调用失败，返回值不为 0。

////////////////////////////////////

应用程序在操作完接触式 IC 后，在用户拔出卡片之前，需调用此函数以给卡片下电。

**2. int APIENTRY CRT\_IC\_CloseCard(HANDLE ComHandle);**

//参数:

// ComHandle: 串口句柄。

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

////////////////////////////////////

自动测试读卡机中的卡片类型。

**3. int APIENTRY CRT\_R\_DetectCard(HANDLE ComHandle,BYTE \*\_CardType,BYTE \*\_CardInfor)**

//参数:

// ComHandle: 串口句柄。

// CardType: 卡片类型。

// CardInfor: 卡片信息。

CardType	CardInfor	说明
'N'	'0'	卡机内无卡
	'1'	未知卡类型
	'2'	卡不在允许操作的位置上
'0'	'0'	卡为非接触式射频卡 S50
	'1'	卡为非接触式射频卡 S70
	'2'	卡为非接触式射频卡 UL
	'4'	卡为非接触式射频卡 TYPE A CPU
	'5'	卡为非接触式射频卡 TYPE B CPU
'1'	'0'	卡为 T=0 接触式 CPU 卡
	'1'	卡为 T=1 接触式 CPU 卡
'2'	'0'	卡为 24C01 卡
	'1'	卡为 24C02 卡
	'2'	卡为 24C04 卡

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	23/52

	‘3’	卡为 24C08 卡
	‘4’	卡为 24C16 卡
	‘5’	卡为 24C32 卡
	‘6’	卡为 24C64 卡
‘3’	‘0’	卡为 SL4442 卡
	‘1’	卡为 SL4428 卡
‘4’	‘0’	卡为 AT88S102 卡
	‘1’	卡为 AT88S1604 卡
	‘2’	卡为 AT45D041 卡
	‘3’	卡为 AT88S1608 卡

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	24/52

//////////////////////////////// CPU 卡函数 (T=0/ T=1) //////////////////////////////////

CPU 卡复位。

1. **int APIENTRY CPU\_ColdReset(HANDLE ComHandle, BYTE \_CPUMode, BYTE \*\_CPUType, BYTE \_exData[], int \*\_exdataLen);**

//参数:

// ComHandle: 串口句柄。

// **\_CPUMode: 复位模式,**

=0x0 EMV 模式(复位命令包中如无 Mode 参数仍然按 EMV 复位模式来操作)

=0x1 ISO7816 模式

// \_CPUType: 返回的 CPU 卡类型

=0x0 T=0

=0x1 T=1

// \_exData: 返回数据。

// \_exdataLen: 返回数据的长度。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

= 'N' (0X4E) 操作不成功

= 'E' (0X45) 卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

举例: (.net 用户请看.net 例子源程序)

int rc;

BYTE exData[300]; //预设长度为 300, 实际返回的有效长度依&lenASK 中的值为准。

int lenASK;

BYTE \_CPUType;

rc=CPU\_Reset(hCom, 0, &\_CPUType, exData, &lenASK);

//复位成功

//rc=0

//&lenASK 保存返回数据长度

//exData 数组保存复位后返回的数据

CString str,t;

for(int i=0; i<lenASK; i++)

{

t.Format("%02x ", exData[i]);

str += t;

t="";

}

t.Format("CPU 卡类型为 T%d \n:", \_CPUType);





## SPECIFICATION

Model No.

## CRT-310 读卡器

Date \_\_\_\_\_

2011/07/29

Ver.

v20110729

Page

25/52

## 动态库说明

```
m_list1.AddString (t);
```

```
t.Format("冷复位时返回数据长度为%d \n 如下所示:", lenASK);
```

```
m_list1.AddString (t);
```

```
m_list1.AddString (str); //字符串 str 中的数据即为实际返回的 16 进制字符串数据
```

```
m_list1.AddString ("\n");
```

////////////////////////////////////

CPU 卡热复位。

**2. int APIENTRY CPU\_WarmReset(HANDLE ComHandle, BYTE \*\_CPUType, BYTE \_exData[], int \*\_exdataLen);**

//参数:

```
// ComHandle: 串口句柄。
```

```
// _CPUType: 返回的 CPU 卡类型
```

$$=0 \times 0 \quad T=0$$
$$=0 \times 1 \quad T=1$$

```
// _exData: 返回数据。
```

```
// _exdataLen: 返回数据的长度。
```

//返回值:

```
// 如果函数调用成功, 返回值为 0。
```

// 如果函数调用失败, 返回值不为 0。

= 'N' (0X4E)    操作不成功

= 'E' (0X45)    卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

举例：请参考 CPU 复位命令函数

////////////////////////////////////

CPU(T=0)卡 C-APDU 命令。

```
3.  int APIENTRY CPU_T0_C_APDU(HANDLE ComHandle, int _dataLen, BYTE _APDUData[], BYTE _exData[], int
*_exdataLen);
```

//参数:

```
// ComHandle: 串口句柄。
```

```
//      dataLen: C-APDU 命令长度。
```

```
// _APDUData: CUP 卡 C-APDU 命令。
```

// exData: 返回数据。

```
// _exdataLen: 返回数据的长度。
```

//返回值:

```
// 如果函数调用成功, 返回值为 0。
```

// 如果函数调用失败, 返回值不为 0。

= 'N' (0X4E)    操作不成功

= 'E' (0X45)      卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。



	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	27/52

- = ‘E’ (0X45) 卡机无卡
- = ‘W’ (0X57) 卡不在允许操作的位置上。

举例：请参考 CPU (T=0) 中 C-APDU 命令函数

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	28/52

//////////////////////////////////// SIM 卡函数 (T=0/ T=1) //////////////////////////////////////

SIM 卡复位。

1. **int APIENTRY SIM\_Reset(HANDLE ComHandle, BYTE \_VOLTAGE, BYTE \_SIMNo, BYTE \*\_SIMTYPE, BYTE \_exData[], int \*\_exdataLen);**

//参数:

// ComHandle: 串口句柄。

// \_VOLTAGE

= 0x2E 对工作电压是 1.8 V 的 SIM 卡进行复位操作

= 0x2F 对工作电压是 3.0 V 的 SIM 卡进行复位操作

= 0x30 对工作电压是 5.0 V 的 SIM 卡进行复位操作

// SIMNo: SIM 卡座号码。

SIM 卡座号=0x30 操作 SIM 卡 1

=0x31 操作 SIM 卡 2

=0x32 操作 SIM 卡 3

=0x33 操作 SIM 卡 4

=0x34 操作 SIM 卡 5

=0x35 操作 SIM 卡 6

=0x36 操作 SIM 卡 7

=0x37 操作 SIM 卡 8

// \_SIMTYPE: 返回的 SIM 卡类型

=0x0 T=0

=0x0 T=1

// \_exData: 返回数据。

// \_exdataLen: 返回数据的长度。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////////

2. SIM 卡(T=0)C-APDU 命令。

**int APIENTRY SIM\_T0\_C\_APDU(HANDLE ComHandle, BYTE SIMNo, int \_dataLen, BYTE \_APDUData[], BYTE \_exData[], int \*\_exdataLen);**

//参数:

// ComHandle: 串口句柄。

// SIMNo: SIM 卡座号码。

SIM 卡座号=0x30 操作 SIM 卡 1

=0x31 操作 SIM 卡 2

=0x32 操作 SIM 卡 3

=0x33 操作 SIM 卡 4

// 如果函数调用失败, 返回值不为 0。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	30/52

//////////////////////////////// SLE4428 函数////////////////////////////////

SEL4428 复位。

**1. int APIENTRY SLE4428\_Reset(HANDLE ComHandle)**

//参数:

// ComHandle: 串口句柄。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

= 'E' (0X45) 卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

////////////////////////////////

此函数从卡片内的 \_Address 地址起读取 \_dataLen 字节长的数据, 并存储在 \_BlockData 指针内返回给应用程序。

**2. int APIENTRY SLE4428\_Read(HANDLE ComHandle, int \_Address, BYTE \_dataLen, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。

// \_Address

// 指定要读取的数据在卡片存储区内的起始地址。

// \_dataLen

// 指定要读取的数据长度。SLE4428 存储区的长度为 1024 字节。

// \_BlockData

// 字符型指针, 指向从卡片存储区内读到的数据并返回给应用程序。

//返回值:

// 如果函数调用成功, 返回值为 0, 且 \_bReadData 的内容为从卡片读取到的数据。

// 如果函数调用失败, 返回值不为 0。

= 'E' (0X45) 卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

////////////////////////////////

此函数将 \_BlockData 指向的数据写入到卡片内 \_Address 地址, 共 \_dataLen 个字节。为了保证写入的正确性, 应用程序应确保 \_BlockData 指针指向的内容及长度。

**3. int APIENTRY SLE4428\_Write(HANDLE ComHandle, int \_Address, BYTE \_dataLen, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。

// \_Address

// 指定卡片存储区内被重写的起始地址。

// \_dataLen

// 指定要写入的数据长度。SLE4428 存储区的长度为 1024 字节。

// \_BlockData

// 字符型指针。应用程序在调用此函数前, 将要写入到卡片内的数据存于 \_BlockData 指针内, 该内容会覆盖卡片内 \_Address

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	31/52

地址起的\_dataLen 个字节。

//返回值:

// 如果函数调用成功, 返回值为 0。  
// 如果函数调用失败, 返回值不为 0。  
= 'E' (0X45) 卡机无卡  
= 'W' (0X57) 卡不在允许操作的位置上。

//

此函数读取 SLE4428 卡片的保护区内容。0x01 表示相应数据区的内容未被保护, 只要密码校验正确即可改写; 0x00 表示相应数据区的内容已被保护, 不能再被改写。

**4. int APIENTRY SLE4428\_ReadP(HANDLE ComHandle, int \_Address, BYTE \_dataLen, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。  
// \_Address  
// 指定要读取的保护区的起始地址。  
// \_dataLen  
// 指定要读取的保护区的长度。SLE4428 保护区的长度为 1024 字节。  
// \_BlockData  
// 字符型指针, 指向从卡片内读到的保护区内容并返回给应用程序。

//返回值:

// 如果函数调用成功, 返回值为 0, 且\_bPReadData 的内容为从卡片读取到的字符串。  
// 如果函数调用失败, 返回值不为 0。  
= 'E' (0X45) 卡机无卡  
= 'W' (0X57) 卡不在允许操作的位置上。

//

此函数将 \_BlockData 指向的字符串写入到卡片内 \_Address 地址, 共 \_dataLen 个字节, 同时写相应的保护位。

**5. int APIENTRY SLE4428\_WriteP(HANDLE ComHandle, int \_Address, BYTE \_dataLen, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。  
// \_Address  
// 指定卡片存储区内被重写的起始地址。  
// \_dataLen  
// 指定要写入的数据长度。SLE4428 存储区受保护字节范围为字节地址 0 至 1023, 共 1024 字节。  
// \_BlockData  
// 字符型指针。应用程序在调用此函数前, 将要写入到卡片内的数据存于 \_BlockData 指针内, 该串的内容会覆盖卡片内 \_Address 地址起的 \_dataLen 个字节。

//返回值:

// 如果函数调用成功, 返回值为 0。  
// 如果函数调用失败, 返回值不为 0。





	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	33/52

//////////////////////////////// SLE4442 函数////////////////////////////////

SEL4442 复位。

**1. int APIENTRY SLE4442\_Reset(HANDLE ComHandle);**

//参数:

// ComHandle: 串口句柄。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////

SEL4442 读数据。

**2. int APIENTRY SLE4442\_Read(HANDLE ComHandle, BYTE \_Address, BYTE \_dataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

// \_Address: 地址。

// \_dataLen: 数据长度。

// \_BlockData: 数据。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////

SEL4442 读保护数据。

**3. int APIENTRY SLE4442\_ReadP(HANDLE ComHandle, BYTE \_BlockDataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

// \_BlockDataLen: 要读出的数据长度, 固定为 32

// \_BlockData: 数据, 32 个字节。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////

SEL4442 读安全数据。

**4. int APIENTRY SLE4442\_ReadS(HANDLE ComHandle, BYTE \_BlockDataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

// \_BlockDataLen: 要读出的数据长度, 固定为 4

// \_BlockData: 数据, 4 个字节。

//返回值:

// 如果函数调用成功, 返回值为 0。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	34/52

// 如果函数调用失败，返回值不为 0。

//

SEL4442 验证密码。

**5. int APIENTRY SLE4442\_VerifyPWD(HANDLE ComHandle, BYTE \_PWDDataLen, BYTE \_PWDData[]);**

//参数:

// ComHandle: 串口句柄。

// \_PWDDataLen: 要写入的密码长度，固定为 3

// \_PWDData: 密码，3 个字节。

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

//

SEL4442 写数据。

**6. int APIENTRY SLE4442\_Write(HANDLE ComHandle, BYTE \_Address, BYTE \_dataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

// \_Address: 地址。

// \_dataLen: 数据长度。

// \_BlockData: 数据。

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

//

SEL4442 写保护数据。

**7. int APIENTRY SLE4442\_WriteP(HANDLE ComHandle, BYTE \_Address, BYTE \_dataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

// \_dataLen: 要写入的写保护数据长度

// \_BlockData: 数据。

//返回值:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

//

SEL4442 更改密码。

**8. int APIENTRY SLE4442\_WritePWD(HANDLE ComHandle, BYTE \_PWDDataLen, BYTE \_PWDData[]);**

//参数:

// ComHandle: 串口句柄。

// \_PWDDataLen: 密码长度，固定为 3。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	35/52

```
//  _PWData: 密码，3 个字节。  
  
//返回值:  
  
//  如果函数调用成功，返回值为 0。  
  
//  如果函数调用失败，返回值不为 0。
```

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	36/52

////////////////////////////////////24Cxx 系列卡函数////////////////////////////////////

检测 24Cxx 卡类型。

1. **int APIENTRY IC\_DetectCard(HANDLE ComHandle, BYTE \*\_CardType);**

//参数:

// ComHandle: 串口句柄。

// \_CardType: 卡类型。

- =0 设置卡为 24C01 128BYTE ADR=0x0000—0x007F
- =1 设置卡为 24C02 256BYTE ADR=0x0000—0x00FF
- =2 设置卡为 24C04 512BYTE ADR=0x0000—0x01FF
- =3 设置卡为 24C08 1K BYTE ADR=0x0000—0x03FF
- =4 设置卡为 24C16 2K BYTE ADR=0x0000—0x07FF
- =5 设置卡为 24C32 4K BYTE ADR=0x0000—0x0FFF
- =6 设置卡为 24C64 8K BYTE ADR=0x0000—0x1FFF

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= ‘N’ (0X4E) 失败

P = ‘E’ (0X45) 卡机无卡

P = ‘W’ (0X57) 卡不在允许操作的位置上。

////////////////////////////////////

读指定地址数据。

2. **int APIENTRY IC\_ReadBlock(HANDLE ComHandle, BYTE \_CardType, int \_Address, BYTE \_dataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

// \_Address: 起始地址。

// \_dataLen: 数据长度。

// \_BlockData: 数据。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

////////////////////////////////////

写指定地址数据。

3. **int APIENTRY IC\_WriteBlock(HANDLE ComHandle, BYTE \_CardType, int \_Address, BYTE \_dataLen, BYTE \_BlockData[]);**

//参数:

// ComHandle: 串口句柄。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	37/52

```
// _Address: 起始地址。
// _dataLen: 数据长度。
// _BlockData: 数据。

//返回值:
// 如果函数调用成功，返回值为 0。
// 如果函数调用失败，返回值不为 0。
```

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	38/52

//////////////////////////////////// AT88SC102 函数////////////////////////////////////

复位。

1.    **int APIENTRY AT88SC102\_Reset(HANDLE ComHandle);**

//参数:

//     ComHandle: 串口句柄。

//返回值:P

//     如果函数调用成功, 返回值为 0。

//     如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E)    失败

P = 'E' (0X45)    卡机无卡

P = 'W' (0X57)    卡不在允许操作的位置上。

////////////////////////////////////

验证主密码。

2.    **int APIENTRY AT88SC102\_VerifyPWD(HANDLE ComHandle, BYTE \_PWDataLen, BYTE \_PWData[]);**

//参数:

//     ComHandle: 串口句柄。

//     \_PWDataLen: 密码长度, 固定为 2。

//     \_PWData: 主密码, 2 个字节。

//返回值 P:

//     如果函数调用成功, 返回值为 0。

//     如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E)    失败

P= 'E' (0X45)    卡机无卡

P= 'F' (0X46)    卡已报废 (密码验证失败超过允许次数后卡锁死报废)

P= 'W' (0X57)    卡不在允许操作的位置上。

在安全级别 1 模式下验证主密码后所有单元均可读出。

在安全级别 2 模式下验证主密码成功除密码存贮单元读不出外, 其余单元均可读出。

////////////////////////////////////

更改密码。

3.    **int APIENTRY AT88SC102\_WritePWD(HANDLE ComHandle, BYTE \_PWIndex, BYTE \_PWData[]);**

//参数:

//     ComHandle: 串口句柄。

//     \_PWIndex: 密码类型选择。0=主密码; 1=擦除密码一; 2=擦除密码二

//     \_PWData: 密码。

区号 =0    修改控制区密码       密码数据为 2 byte

      =1    修改应用区一密码    密码数据为 6 byte

      =2    修改应用区二密码    密码数据为 4 byte

////////////////////





	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	41/52

// \_BlockData: 数据。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E) 失败

P= 'E' (0X45) 卡机无卡

P= 'W' (0X57) 卡不在允许操作的位置上。

//

个人化操作, 使卡进入安全级别模式 2。

**9. int APIENTRY AT88SC102\_InitSecurity2(HANDLE ComHandle, BYTE \_CtrlMode);**

//参数:

// ComHandle: 串口句柄。

// \_CtrlMode: 操作模式。

\_CtrlMode =0x30 使卡模拟进入安全级别模式 2, 可供测试,

\_CtrlMode =0x31 使模拟进入安全级别模式 2 的卡恢复到安全级别模式 1。

\_CtrlMode =0x32 使卡完全进入安全级别模式 2 , 一旦将卡操作成安全模式 2,

将无法再恢复到安模式 1。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E) 失败

P= 'E' (0X45) 卡机无卡

P = 'W' (0X57) 卡不在允许操作的位置上。

进入安全模式 2 前一定设定好应用区一, 二的密码, 应用区一第一字节 (0x16) 应用区二第一字节 (0x5C) 不能轻易修改, 是控制这些区单元的读写使能。

进入安全模式 2 后要写这些应用区, 卡进行擦除操作时是对这些应用区整块擦除, 应注意写入新数据前应先读出保存, 以防数据丢失。同时这些应用区受熔丝计数器的控制。使熔丝计数器有效则写入 128 次后不能再写入。使其无效后则写入次数为卡的最大有效操作数 (100, 000 次)。

//

二区擦除计数器操作字 EC2 设置成无效操作。

**10. int APIENTRY AT88SC102\_DisableEC2(HANDLE ComHandle);**

//参数:

// ComHandle: 串口句柄。


//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E) 失败

P= 'E' (0X45) 卡机无卡

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	42/52

P = 'W' (0X57) 卡不在允许操作的位置上。

要使卡在安全模式 2 下应用区二擦除次数不受限则要在进入个人化操作前执行此操作。

否则在卡设置完成模式 2 后，卡默认应用区二在模式 2 下擦除次数受限有效(只能擦除 128 次)。再要取消应用二区擦写不受限，则无法取消二区擦除受限次数（128 次）。同样设置成卡在模式 2 下擦写次数不受限后不能再设置成擦写受限。同时用户也要对 EC2 操作状态保存，卡在模式 2 下应用时要擦写应用区二时（验证应用区二的擦除密码）应注意相应的参数。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	43/52

//////////////////////////////////// AT88SC1604 函数////////////////////////////////////

复位。

1.    **int APIENTRY AT88SC1604\_Reset(HANDLE ComHandle)**

//参数:

//     ComHandle: 串口句柄。

//返回值 P:

//     如果函数调用成功, 返回值为 0。

//     如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E)    失败

P= 'E' (0X45)    卡机无卡

P= 'W' (0X57)    卡不在允许操作的位置上。

////////////////////////////////////

验证密码。

2.    **int APIENTRY AT88SC1604\_VerifyPWD(HANDLE ComHandle, BYTE \_PWIndex,BYTE \_PWDData[])**

//参数:

//     ComHandle: 串口句柄。

//     PWIndex: 密码类型索引。

其中密码类型号: = 0 验证主密码

                  = 1 验证应用一区密码

                  = 2 验证应用一区擦除密码

                  = 3 验证应用二区密码

                  = 4 验证应用二区擦除密码

                  = 5 验证应用三区密码

                  = 6 验证应用三区擦除密码

                  = 7 验证应用四区密码

                  = 8 验证应用四区擦除密码

//     \_PWDData: 密码。

//返回值 P:

//     如果函数调用成功, 返回值为 0。

//     如果函数调用失败, 返回值不为 0。

P= 'N' (0X4E)    失败

P= 'E' (0X45)    卡机无卡

P= 'W' (0X57)    卡不在允许操作的位置上。

P= 'F' (0X46)    卡已报废或应用块报废



	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	45/52

// 如果函数调用成功，返回值为 0。

// \_BlockData: 返回的数据

// 如果函数调用失败，返回值不为 0。

P= 'N' (0X4E) 失败

P= 'E' (0X45) 卡机无卡

P= 'W' (0X57) 卡不在允许操作的位置上。

//

写指定地址的数据。

5. **int APIENTRY AT88SC1604\_Write(HANDLE ComHandle, BYTE \_Index, int \_Address, BYTE \_dataLen, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。

// \_Index: 数据区选择。

区号:     = 0       一区( 0x020 --- 0x21A )  
          = 1       二区( 0x21B --- 0x420 )  
          = 2       三区( 0x421 ---- 0x621 )  
          = 3       四区(0x622 ---- 0x7F5 )  
          = 4       其它区(除一, 二, 三, 区以外的区域)

// \_Address: 起始地址。

// \_dataLen: 数据长度。

// \_BlockData: 返回的数据

//返回值 P:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

P= 'N' (0X4E) 失败

P= 'E' (0X45) 卡机无卡

P= 'W' (0X57) 卡不在允许操作的位置上。

//

擦除指定地址的数据。

6. **int APIENTRY AT88SC1604\_Clear(HANDLE ComHandle, BYTE \_Index, int \_Address, BYTE \_dataLen)**

//参数:

// ComHandle: 串口句柄。

// \_Index: 数据区选择。

区号:     = 0       一区( 0x020 --- 0x21A )  
          = 1       二区( 0x21B --- 0x420 )  
          = 2       三区( 0x421 ---- 0x621 )  
          = 3       四区(0x622 ---- 0x7F5 )  
          = 4       其它区(除一, 二, 三, 区以外的区域)

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	46/52

```
//  _Address: 起始地址。
//  _dataLen: 数据长度。
//返回值 P:
//    如果函数调用成功, 返回值为 0。
//    如果函数调用失败, 返回值不为 0。
//    P= 'N' (0X4E)   失败
//    P= 'E' (0X45)   卡机无卡
//    P= 'W' (0X57)   卡不在允许操作的位置上。

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
个人化操作 (卡进入安全模式 2)。

7.  int APIENTRY AT88SC1604_Personalization(HANDLE ComHandle, BYTE _data);
//参数:
//    ComHandle: 串口句柄。
//    _data: 操作号 = 0x30  软个人化操作 (模拟个人化操作使卡进入安全模式 2, 供测试)
//           = 0x31  退出软个人化操作
//           = 0x32  完全个人化操作不可再恢复
//返回值 P:
//    如果函数调用成功, 返回值为 0。
//    如果函数调用失败, 返回值不为 0。
//    P= 'N' (0X4E)   失败
//    P= 'E' (0X45)   卡机无卡
//    P= 'W' (0X57)   卡不在允许操作的位置上。
```

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	47/52

//////////////////////////////////// AT88SC1608 函数////////////////////////////////////

//复位

1. int APIENTRY AT88SC1608\_Reset(HANDLE ComHandle);

//参数:

// ComHandle: 串口句柄。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

= 'N' (0X4E) 失败

= 'E' (0X45) 卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

////////////////////////////////////

//验证密码

2. int APIENTRY AT88SC1608\_VerifyPWD(HANDLE ComHandle, BYTE \_PWIndex, BYTE \_PWDData[],BYTE \_PWDDataLen);

//参数:

// ComHandle: 串口句柄。

// PWIndex: 密码类型索引。

= 0 验证应用一 区读密码

= 1 验证应用二 区读密码

= 2 验证应用三 区读密码

= 3 验证应用四 区读密码

= 4 验证应用五 区读密码

= 5 验证应用六 区读密码

= 6 验证应用七 区读密码

= 7 验证应用八 区读密码

= 8 验证应用一 区写密码

= 9 验证应用二 区写密码

=10 验证应用三 区写密码

=11 验证应用四 区写密码

=12 验证应用五 区写密码

= 13 验证应用六 区写密码

= 14 验证应用七 区写密码/验证主密码

= 15 验证应用八 区写密码

// \_PWDData: 密码, 3 个字节。

// \_PWDDataLen: 密码长度, 固定为 3。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	48/52

// 如果函数调用失败，返回值不为 0。

= ‘N’ (0X4E) 验证密码失败

= ‘F’ (0X46) 卡已报废或应用块报废

= ‘E’ (0X45) 卡机无卡

= ‘W’ (0X57) 卡不在允许操作的位置上。

注意：每一个区密码只有允许验证 8 次，8 次校验错误后卡锁死，就表明卡这个块区不能读或写。

//

//读数据

3. int APIENTRY AT88SC1608\_Read(HANDLE ComHandle, BYTE \_Index, BYTE \_Address, BYTE \_dataLen, BYTE \_BlockData[]);

//参数:

// ComHandle: 串口句柄。

// \_Index: 区号。

- = 0 应用一区 (len=0x01—0x80)
- = 1 应用二区 (len=0x01—0x80)
- = 2 应用三区 (len=0x01—0x80)
- = 3 应用四区 (len=0x01—0x80)
- = 4 应用五区 (len=0x01—0x80)
- = 5 应用六区 (len=0x01—0x80)
- = 6 应用七区 (len=0x01—0x80)
- = 7 应用八区 (len=0x01—0x80)
- = 8 设置区 (len=0x01—0x80)

// \_Address: 起始地址。操作地址范围: 0x00---0xFF (除设置区外)

// dataLen: 数据长度。操作长度范围: 0x01----0xFF (除设置区外)

// BlockData: 数据。

要对应用区进行读时请校验该区读密码正确后才能进行读，否则读的数据无效，设置区数据只有密码区域（0x40---0x7F）是受密码保护，只能校验正确后才能正确读出

//返回值 P:

// 如果函数调用成功，返回值为 0。

// 如果函数调用失败，返回值不为 0。

= ‘N’ (0X4E) 失败

= ‘E’ (0X45) 卡机无卡

= ‘W’ (0X57) 卡不在允许操作的位置上。

//

//写数据

4. int APIENTRY AT88SC1608\_Write(HANDLE ComHandle, BYTE \_Index, BYTE \_Address, BYTE \_dataLen, BYTE \_BlockData[]);

//参数:



////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	50/52

//设置熔丝状态

6. int APIENTRY AT88SC1608\_SetFUSE(HANDLE ComHandle);

//参数:

// ComHandle: 串口句柄。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

= 'N' (0X4E) 失败

= 'E' (0X45) 卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

//

//初始化验证区

7. int APIENTRY AT88SC1608\_InitAuth(HANDLE ComHandle, BYTE \_Qlen, BYTE \_Q[]);

//参数:

// ComHandle: 串口句柄。

// \_Qlen 随机数长度, 固定为 8。

// \_Q[]: 8 byte 随机数 Q0、Q1、Q2、Q3、Q4、Q5、Q6、Q7

初始化认证区是先读取卡中的 Nc, Ci, 通过 F1 或 F2 算法, 计算出 Gc=F1(Ks, Nc) 得到的随机数 Q0~Q7, 送入 AT88SC1608 卡中, 完成进行初始化认证区。

//返回值 P:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值不为 0。

= 'N' (0X4E) 失败

= 'E' (0X45) 卡机无卡

= 'W' (0X57) 卡不在允许操作的位置上。

//

//校验验证区

8. int APIENTRY AT88SC1608\_VerifyAuth(HANDLE ComHandle, BYTE \_Qlen, BYTE \_Q[]);

//参数:

// ComHandle: 串口句柄。

// \_Qlen 随机数长度, 固定为 8。

// \_Q[]: 8 byte 随机数 Q0、Q1、Q2、Q3、Q4、Q5、Q6、Q7

校验认证区是在进行初始化认证区操作后, 按 F2 算法完成 Q1=F2(Gc, Ci, Q0), 分别生成的 Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7 送入 AT88SC1608 卡中由卡中来完成校验认证区, 进行此验证。

注: Nc: 识别码, 通常用作卡的唯一标识——卡号。个人化前定义。

Ci: 密文, 个人化前可写一随机数, 认证卡时使用, 每次认证会被自动改写。

Gc: 密钥, 64 位的保密种子, 由 Nc 通过 F1 公式推算出来, 在个人化前, 写入卡

中。个人化后不可访问, 认证时作为该卡的 F2 公式的参数。(详细用法参见认证协议)

////////////////////////////////////

	SPECIFICATION	Model No.	CRT-310 读卡器
		Date	2011/07/29
	动态库说明	Ver.	v20110729
		Page	52/52

//////////////////////////////////// AT45D041 函数////////////////////////////////////

复位。

1. **int APIENTRY AT45D041\_Reset(HANDLE ComHandle)**

//参数:

// ComHandle: 串口句柄。

//返回值:

// 如果函数调用成功, 返回值为 0。

// 如果函数调用失败, 返回值为-1。

////////////////////////////////////

读指定地址页的数据。

2. **int APIENTRY AT45D041\_Read(HANDLE ComHandle, int \_Address, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。

// \_Address: 页地址。页地址: 0x0000---0x07FF

// \_BlockData: 读取的数据

//返回值 P:

// 如果函数调用成功, 返回值为 0, 且\_BlockData 的内容为从卡片读取到的字符串。

// 如果函数调用失败, 返回值不为 0。

= ‘E’ (0X45) 卡机无卡

= ‘W’ (0X57) 卡不在允许操作的位置上。

////////////////////////////////////

写指定地址页的数据。

3. **int APIENTRY AT45D041\_Write(HANDLE ComHandle, int \_Address, BYTE \_BlockData[])**

//参数:

// ComHandle: 串口句柄。

// \_Address: 页地址。页地址: 0x0000---0x07FF

// \_BlockData: 写入的数据 \_BlockData 的内容为要写入卡片的字符串。

//返回值 P:

// 如果函数调用成功, 返回值为 0

// 如果函数调用失败, 返回值不为 0。

= ‘E’ (0X45) 卡机无卡

= ‘W’ (0X57) 卡不在允许操作的位置上。