说明

该文档包含 F3-1300 读卡器 API 的说明。

F3-1300 开发包提供以下相关文件供开发使用：
F3API.h 头文件
F3API.lib F3API.DLL 的导入库
F3API.dll 32 位动态链接库。

API 参考

基本操作

F3_Connect

A6_Connect 在调用程序和读卡器间建立一个连接。

```
LONG
WINAPI
F3_Connect
    (
    IN  DWORD   dwPort,
    IN  DWORD   dwSpeed,
    IN  BYTE    bCRAddr,
    OUT LPREADERHANDLE lphReader
    );
```

参数

dwPort	COM 端口号。可用值: 1 ~ 256.
dwSpeed	波特率。可用值： 9600 19200 38400 57600
bCRAddr	卡机地址。可用值：0~15。
phReader	返回一个标识读卡器连接的句柄。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Disconnect

断开调用程序和读卡器间的连接。

```
LONG
WINAPI
F3_Disconnect
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Initialize

复位读卡器。

```
LONG
WINAPI
F3_Initialize
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bMode,
    IN  BOOL    fEnableCounter,
    OUT PSTR    pszRevBuff,
    IN OUT  PDWORD  pcbRevLength
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
bMode	复位模式。可用值： INIT_RETURN_TO_FRONT　复位并移动卡到出卡口位 INIT_CAPTURE_TO_BOX　复位并回收卡 INIT_WITHOUT_MOVEMENT　复位，不移动卡
fEnableCounter	是否启用回收卡计数功能。
pszRevBuff	指向返回的固件版本信息。
pcbRevLength	提供 pbVerBuff 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0，其它值为错误码。

F3_GetCRStatus

获取卡机的状态。

```
LONG
WINAPI
F3_GetCRStatus
    (
    IN  READERHANDLE hReader,
    OUT PCRSTATUS lpStatus
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

lpStatus　　指向返回的卡机状态。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_GetSenseDetail

获取传感器信息。

```
LONG
WINAPI
F3_GetSenserDetail
    (
    IN  READERHANDLE hReader,
    OUT BYTE    (&bStatus)[NUM_SENSORS]
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bStatus　　返回的传感器状态。依次为 S1~S10、KS1、KS2。
值为 0x31，表示有卡；值为 0x30，表示无卡。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MoveCard

移动卡片。

```
LONG
WINAPI
```

```
F3_MoveCard
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bMode
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bMode　　移动方式。可用值：
MM_RETURN_TO_FRONT　移动卡到前端持卡位
MM_RETURN_TO_IC_POS 移动卡到IC位
MM_RETURN_TO_RF_POS 移动卡到射频位
MM_CAPTURE_TO_BOX　回收卡
MM_EJECT_TO_FRONT　从前端弹出卡片

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_PermitInsertion

允许从前端进卡。

```
LONG
    WINAPI
    F3_PermitInsertion
    (
    __in    READERHANDLE hReader
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_DenieInsertion

禁止从前端进卡。

```
LONG
    WINAPI
    A6_DenieInsertion
    (
    __in    READERHANDLE hReader
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_DetectIccType

检测读卡器内的IC卡的类型。

```
LONG
WINAPI
F3_DetectIccType
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbCardType
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pbCardType　　返回 IC 卡的类型。可能的值：
ICCTYPE_UNKNOWN
ICCTYPE_T0_CPU
ICCTYPE_T1_CPU
ICCTYPE_SLE4442
ICCTYPE_SLE4428

ICCTYPE_AT24C01
ICCTYPE_AT24C02
ICCTYPE_AT24C04
ICCTYPE_AT24C08
ICCTYPE_AT24C16
ICCTYPE_AT24C32
ICCTYPE_AT24C64
ICCTYPE_AT24C128
ICCTYPE_AT24C256

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_DetectRfcType

检测读卡器内的射频卡的类型。

```
LONG
WINAPI
F3_DetectRfcType
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbCardType
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pbCardType　　返回射频卡的类型。可能的值：
RFCTYPE_UNKNOWN
RFCTYPE_MIFARE_S50
RFCTYPE_MIFARE_S70
RFCTYPE_MIFARE_UL
RFCTYPE_TYPEA_CPU
RFCTYPE_TYPEB_CPU

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

接触式 CPU 卡操作

F3_CpuActivate

CPU 卡激活（冷复位）。

```
LONG
WINAPI
F3_CpuActivate
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbProtocol,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength,
    IN OPTIONAL BYTE    bVCC = VCC_5V_EMV
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
pbProtocol	返回CPU卡的协议类型。可能的值： ICC_PROTOCOL_T0 ICC_PROTOCOL_T1
pbATRBuff	指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。
pcbATRLength	提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。
bVCC	激活 IC 卡时使用的电压。可用值： VCC_5V_EMV　　使用5V电压，并引用EMV标准 VCC_5V_ISO7816　　使用5V电压，并引用ISO7816标准 VCC_3V_ISO7816　　使用3V电压，并引用ISO7816标准

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_CpuDeactivate

CPU 卡释放。

```
LONG
WINAPI
F3_CpuDeactivate
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_CpuGetStatus

获取 CPU 卡状态。

```
LONG
WINAPI
F3_CpuGetStatus
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbStatus
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pbStatus　　返回 CPU 卡的状态。可能的值：

STATUS_DEACTIVATION	CPU卡未激活
STATUS_CLKFREQ_3_57	CPU卡已激活，工作频率为 3.57 MHz
STATUS_CLKFREQ_7_16	CPU卡已激活，工作频率为 7.16 MHz

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_CpuWarmReset

CPU 卡热复位。

```
LONG
WINAPI
F3_CpuWarmReset
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbProtocol,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
pbProtocol	返回CPU卡的协议类型。可能的值： ICC_PROTOCOL_T0 ICC_PROTOCOL_T1
pbATRBuff	指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。
pcbATRLength	提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_CpuTransmit

CPU 卡数据传输。

```
LONG
WINAPI
F3_CpuTransmit
```

```
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bProtocol,
    IN  PBYTE   pbSendBuff,
    IN  USHORT  cbSendLength,
    OUT PBYTE   pbRecvBuff,
    IN OUT  PDWORD  pcbRecvLength
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
bProtocol	CPU 卡的通信协议类型。可用值： ICC_PROTOCOL_T0　　T = 0 协议 ICC_PROTOCOL_T1　　T = 1 协议 ICC_PROTOCOL_AUTO　自动选择 T=0 或 T=1 协议
pbSendBuff	指向要写入到卡片的数据。不可以为 NULL。
cbSendLength	提供 pbSendBuff 参数的长度（字节数）。
pbRecvBuff	指向返回的数据。不可以为 NULL。
pcbRecvLength	提供 pbRecvBuff 参数的长度（字节数）并接收读卡器实际返回的长度。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

SAM 卡操作函数

F3_SamActivate

SAM 卡激活（冷复位）。

```
LONG
WINAPI
F3_SamActivate
    (
    IN  READERHANDLE hReader,
```

```
    OUT PBYTE   pbProtocol,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength,
    IN OPTIONAL BYTE    bVCC = VCC_5V_EMV
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
pbProtocol	返回SAM卡的协议类型。可能的值： ICC_PROTOCOL_T0 ICC_PROTOCOL_T1
pbATRBuff	指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。
pcbATRLength	提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。
bVCC	激活 IC 卡时使用的电压。可用值： VCC_5V_EMV 使用5V电压，并引用EMV标准 VCC_5V_ISO7816 使用5V电压，并引用ISO7816标准 VCC_3V_ISO7816 使用3V电压，并引用ISO7816标准

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_SamDeactivate

SAM 卡释放。

```
LONG
WINAPI
F3_SamDeactivate
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_SamGetStatus

获取 SAM 卡状态。

```
LONG
WINAPI
F3_SamGetStatus
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbStatus,
    OUT PBYTE   pbSAMNumber
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
pbStatus	返回 SAM 卡的状态。可能的值： STATUS_DEACTIVATION　　CPU卡未激活 STATUS_CLKFREQ_3_57　　CPU卡已激活，工作频率为 3.57 MHz STATUS_CLKFREQ_7_16　　CPU卡已激活，工作频率为 7.16 MHz
pbSAMNumber	返回当前的 SAM 卡号。可能的值: 1, 2, 3, …。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_SamWarmReset

SAM 卡热复位。

```
LONG
WINAPI
F3_SamWarmReset
    (
    IN  READERHANDLE hReader,
```

```
    OUT PBYTE   pbProtocol,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
pbProtocol	返回 SAM 卡的协议类型。可能的值： ICC_PROTOCOL_T0 ICC_PROTOCOL_T1
pbATRBuff	指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。
pcbATRLength	提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_SamTransmit

SAM 卡数据传输。

```
LONG
WINAPI
F3_SamTransmit
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bProtocol,
    IN  PBYTE   pbSendBuff,
    IN  USHORT  cbSendLength,
    OUT PBYTE   pbRecvBuff,
    IN OUT  PDWORD  pcbRecvLength
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
bProtocol	SAM 卡的通信协议类型。可用值： ICC_PROTOCOL_T0　　T = 0 协议

	ICC_PROTOCOL_T1　　T = 1 协议 ICC_PROTOCOL_AUTO　自动选择 T=0 或 T=1 协议
pbSendBuff	指向要写入到卡片的数据。不可以为 NULL。
cbSendLength	提供 pbSendBuff 参数的长度（字节数）。
pbRecvBuff	指向返回的数据。不可以为 NULL。
pcbRecvLength	提供 pbRecvBuff 参数的长度（字节数）并接收读卡器实际返回的长度。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_SamWarmReset

SAM 卡热复位。

```
LONG
WINAPI
F3_SamSelect
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSAMNumber
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
bSAMNumber	SAM 卡号。可用值：1，2，3，…

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

SLE4442 卡操作

F3_Sle4442Activate

激活 SLE4442 卡。

```
LONG
WINAPI
F3_Sle4442Activate
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pbATRBuff　　指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。

pcbATRLength　　提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442Deactivate

释放 SLE4442 卡。

```
LONG
WINAPI
F3_Sle4442Deactivate
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442GetStatus

获取 SLE4442 卡状态。

```
LONG
WINAPI
F3_Sle4442GetStatus
    (
    IN  READERHANDLE hReader,
    OUT PBOOL   pfActivated
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pfActivated　　返回值为 TRUE，表示卡已激活；返回值为 FALSE，表示卡未激活。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442ReadMainMemory

读主存储区。

```
LONG
WINAPI
F3_Sle4442ReadMainMemory
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bStartAddress,
    IN  BYTE    bBytesToRead,
    OUT PBYTE   pbBuffer,
```

```
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bStartAddress　　要操作的地址。

bBytesToRead　　要读取的字节数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442UpdateMainMemory

更新主存储区。

```
LONG
WINAPI
F3_Sle4442UpdateMainMemory
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bStartAddress,
    IN  BYTE    nBytesToWrite,
    IN  PBYTE   pbBuffer
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bStartAddress　　要操作的地址。

bBytesToWrite　　要写的字节数。

pbBuffer　　指向要写入到卡片的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442ReadProtectionMemory

读保护存储区。

```
LONG
WINAPI
F3_Sle4442ReadProtectionMemory
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bStartAddress,
    IN  BYTE    bBytesToRead,
    OUT PBYTE   pbBuffer,
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 A6_Connect 返回的句柄值。

bStartAddress　　要操作的地址。

bBytesToRead　　要读取的字节数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442WriteProtectionMemory

写保护存储区。

```
LONG
```

```
WINAPI
F3_Sle4442WriteProtectionMemory
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bStartAddress,
    IN  BYTE    bBytesToWrite,
    IN  PBYTE   pbBuffer
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bStartAddress　　要操作的地址。

bBytesToWrite　　要写的字节数。

pbBuffer　　指向要写入到卡片的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442ReadSecurityMemory

读安全存储区。

```
LONG
WINAPI
F3_Sle4442ReadSecurityMemory
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bStartAddress,
    IN  BYTE    bBytesToRead,
    OUT PBYTE   pbBuffer,
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 A6_Connect 返回的句柄值。

bStartAddress　　要操作的地址。

bBytesToRead　　要读取的字节数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442VerifyPSC

校验安全码。

```
LONG
WINAPI
F3_Sle4442VerifyPSC
    (
    IN  READERHANDLE hReader,
    IN  BYTE    (&bPSCBytes)[3]
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bPSCBytes　　安全码字节数组。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442UpdatePSC

更新安全码。

```
LONG
WINAPI
F3_Sle4442UpdatePSC
```

```
    (
    IN  READERHANDLE hReader,
    IN  BYTE    (&bPSCBytes)[3]
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bPSCBytes　　安全码字节数组。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442WriteErrorCounter

写密码错误计数器。

```
LONG
WINAPI
F3_Sle4442WriteErrorCounter
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bValue
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bValue　　错误计数值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

SLE4428 卡操作

F3_Sle4428Activate

激活 SLE4428 卡。

```
LONG
WINAPI
F3_Sle4428Activate
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pbATRBuff　　指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。

pcbATRLength　　提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4428Deactivate

释放 SLE4428 卡。

```
LONG
WINAPI
F3_Sle4442Deactivate
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4442GetStatus

获取 SLE4428 卡状态。

```
LONG
WINAPI
F3_Sle4428GetStatus
    (
    IN  READERHANDLE hReader,
    OUT PBOOL   pfActivated
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

pfActivated　　返回值为 TRUE，表示卡已激活；返回值为 FALSE，表示卡未激活。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

A6_Sle4428ReadMainMemory

读主存储区。

```
LONG
WINAPI
F3_Sle4428ReadMainMemory
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToRead,
    OUT PBYTE   pbBuffer,
```

```
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

wStartAddress　　要操作的地址。

bBytesToRead　　要读取的字节数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4428ReadProtectionBits

读保护位。

```
LONG
WINAPI
F3_Sle4428ReadProtectionBits
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToRead,
    OUT PBYTE   pbBuffer,
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

wStartAddress　　要操作的地址。

bBytesToRead　　要读取的字节数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

A6_Sle4428WriteWithoutPB

不带保护位写卡。

```
LONG
WINAPI
F3_Sle4428WriteWithoutPB
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToWrite,
    IN  PBYTE   pbBuffer
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

wStartAddress　　要操作的地址。

bBytesToWrite　　要写的字节数。

pbBuffer　　指向要写入到卡片的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4428WriteWithPB

带写保护位写卡。

```
LONG
```

```
WINAPI
F3_Sle4428WriteWithPB
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToWrite,
    IN  PBYTE   pbBuffer
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
wAddress	要操作的地址。
bBytesToWrite	要写的字节数。
pbBuffer	指向要写入到卡片的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4428WritePBWithDataComparison

带数据校验写保护位。

```
LONG
WINAPI
F3_Sle4428WriteWithDataComparison
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToWrite,
    IN  PBYTE   pbBuffer
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
wAddress	要操作的地址。

bBytesToWrite　　要写的字节数。

pbBuffer　　指向要写入到卡片的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_Sle4428VerifyPSC

校验安全码。

```
LONG
WINAPI
F3_Sle4428VerifyPSC
    (
    IN  READERHANDLE hReader,
    IN  BYTE    (&bPSCBytes)[2]
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bPSCBytes　　安全码字节数组。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。
更新密码。

AT24Cxx 卡操作

F3_I2cActivate

激活内存卡。

LONG

```
WINAPI
F3_I2cActivate
    (
    IN  READERHANDLE hReader,
    IN  BYTE     bCardType
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。
bCardType	卡片类型。可用值：

I2CTYPE_UNKNOWN	自动选择
I2CTYPE_24C01	AT24C01
I2CTYPE_24C02	AT24C02
I2CTYPE_24C04	AT24C04
I2CTYPE_24C08	AT24C05
I2CTYPE_24C16	AT24C16
I2CTYPE_24C32	AT24C32
I2CTYPE_24C64	AT24C64
I2CTYPE_24C128	AT24C128
I2CTYPE_24C256	AT24C254

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_I2cDeactivate

释放内存卡。

```
LONG
WINAPI
F3_I2cDeactivate
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_I2cGetStatus

获取内存卡状态。

```
LONG
WINAPI
F3_I2cGetStatus
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbActivatedCard
    );
```

参数

hReader	引用 F3_Connect 返回的句柄值。	
pbActivatedCard	返回已激活的内存卡。可能的值：	
	I2CTYPE_UNKNOWN	无内存卡被激活
	I2CTYPE_24C01	AT24C01
	I2CTYPE_24C02	AT24C02
	I2CTYPE_24C04	AT24C04
	I2CTYPE_24C08	AT24C05
	I2CTYPE_24C16	AT24C16
	I2CTYPE_24C32	AT24C32
	I2CTYPE_24C64	AT24C64
	I2CTYPE_24C128	AT24C128
	I2CTYPE_24C256	AT24C254

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_I2cReadMemory

读存储区。

```
LONG
```

```
WINAPI
F3_I2cReadMemory
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToRead,
    OUT PBYTE   pbBuffer,
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

wStartAddress　　要操作的地址。

bBytesToRead　　要读取的字节数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_I2cWriteMemory

写存储区。

```
LONG
WINAPI
F3_I2cWriteMemory
    (
    IN  READERHANDLE hReader,
    IN  WORD    wStartAddress,
    IN  BYTE    bBytesToWrite,
    IN  PBYTE   pbBuffer
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

wStartAddress　要操作的地址。

bBytesToWrite　要写入的字节数。

pbBuffer　指向要写入的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

射频卡操作

F3_RfcActivate

激活射频卡。

```
LONG
WINAPI
F3_RfcActivate
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbATRBuff,
    IN OUT  PDWORD  pcbATRLength,
    IN  BYTE    bFirstProtocol = RFC_PROTOCOL_NONE,
    IN  BYTE    bSecondProtocol = RFC_PROTOCOL_NONE
    );
```

参数

hReader　引用 F3_Connect 返回的句柄值。

pbATRBuff　指向返回的复位信息。如果不为 NULL，pcbATRLength 也不可以为 NULL。

pcbATRLength　提供 pbATRBuff 参数的长度（字节数）并接收读卡器实际返回的长度。

bFirstProtocol　首选协议。可用值：
RFC_PROTOCOL_NONE
RFC_PROTOCOL_TYPE_A
RFC_PROTOCOL_TYPE_B

bSecondProtocol 次选协议。可用值：

RFC_PROTOCOL_NONE
RFC_PROTOCOL_TYPE_A
RFC_PROTOCOL_TYPE_B

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_RfcDeactivate

释放射频卡。

```
LONG
WINAPI
F3_RfcDeactivate
    (
    IN  READERHANDLE hReader
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_RfcGetStatus

获取射频卡状态。

```
LONG
WINAPI
F3_RfcGetStatus
    (
    IN  READERHANDLE hReader,
    OUT PBYTE   pbActivatedCard
    );
```

参数

hReader 引用 F3_Connect 返回的句柄值。

pbActivatedCard 返回已激活的射频卡类型。可能的值：
RFCTYPE_UNKNOWN 无射频卡被激活
RFCTYPE_MIFARE_S50
RFCTYPE_MIFARE_S70
RFCTYPE_MIFARE_UL
RFCTYPE_TYPEA_CPU
RFCTYPE_TYPEB_CPU

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

Mifare 卡操作

F3_MfVerifyPassword

认证扇区。

```
LONG
WINAPI
F3_MfVerifyPassword
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BOOL    fWithKeyA,
    IN  BYTE    (&bKeyBytes)[6]
    );
```

参数

hReader 引用 F3_Connect 返回的句柄值。

bSectorNumber 要认证的扇区号。

fWithKeyA 值为TRUE，验证KEY-A；值为FALSE，验证KEY-B。

bKeyBytes 密钥字节数组。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfUpdatePassword

更新扇区密码。

```
LONG
WINAPI
F3_MfUpdatePassword
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    (&bKeyBytes)[6]
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bKeyBytes　　新的密钥字节数组。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfLoadPassword

从EEPROM加载密钥并校验扇区。

```
LONG
WINAPI
F3_MfLoadPassword
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BOOL    fWithKeyA
```

```
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要认证的扇区号。

fWithKeyA　　值为TRUE，验证KEY-A；值为FALSE，验证KEY-B。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfDownloadPassword

下载密码到EEPROM。

```
LONG
WINAPI
F3_MfDownloadPassword
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BOOL    fWithKeyA,
    IN  BYTE    (&bKeyBytes)[6]
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

fWithKeyA　　值为TRUE，密鈕类型为KEY-A；值为FALSE，密鈕类型为KEY-B。

bKeyBytes　　密钥字节数组。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfReadSector

读扇区块数据。

```
LONG
WINAPI
F3_MfReadSector
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    bStartBlockNumber,
    IN  BYTE    bBlocksToRead,
    OUT PBYTE   pbBuffer,
    IN OUT  PDWORD  pcbLength
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bStartBlockNumber　　起始块号。

bBlocksToRead　　要读取的块数。

pbBuffer　　指向返回的数据。不可以为 NULL。

pcbLength　　提供 pbBuffer 参数的长度（字节数）并接收读卡器实际返回的长度。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfWriteSector

写扇区块数据。

```
LONG
WINAPI
F3_MfWriteSector
    (
```

```
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    bStartBlockNumber,
    IN  BYTE    bBlocksToRead,
    OUT PBYTE   pbBuffer
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bStartBlockNumber　　起始块号。

bBlocksToRead　　要写的块数。

pbBuffer　　指向要写入的数据。不可以为 NULL。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfInitializeValue

初始化值块。

```
LONG
WINAPI
F3_MfInitializeValue
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    bBlockNumber,
    IN  UINT32  iValue
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bBlockNumber　　要操作的块号。

iValue　　要初始化的值块的值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfReadValue

读值块。

```
LONG
WINAPI
F3_MfReadValue
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    bBlockNumber,
    OUT UINT32  *piValue
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bBlockNumber　　要操作的块号。

piValue　　返回值块的值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfIncrementValue

增值。

```
LONG
WINAPI
F3_MfIncrementValue
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    bBlockNumber,
    IN  UINT32  iValue
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bBlockNumber　　要操作的块号。

iValue　　要增加到值块的值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

F3_MfDecrementValue

减值。

```
LONG
WINAPI
F3_MfDecrementValue
    (
    IN  READERHANDLE hReader,
    IN  BYTE    bSectorNumber,
    IN  BYTE    bBlockNumber,
    IN  UINT32  iValue
    );
```

参数

hReader　　引用 F3_Connect 返回的句柄值。

bSectorNumber　　要操作的扇区号。

bBlockNumber　　要操作的块号。

iValue　　要从值块中减掉的值。

返回值

如果函数调用成功，返回值为 0, 其它值为错误码。

结构/类

CRSTATUS

磁道信息结构。

```
typedef struct _CRSTATUS
{
    BYTE    bLaneStatus;
    BYTE    bCardBoxStatus;
    BOOL    fCaptureBoxFull;
} CRSTATUS, *PCRSTATUS;
```

成员

bLaneStatus　　通道状态。可用值：

LS_NO_CARD_IN	机内无卡
LS_CARD_AT_GATE_POS	卡在出卡口位
LS_CARD_IN	机内无卡

bCardBoxStatus　卡箱状态。可用值：

CBS_EMPTY	卡箱空
CBS_INSUFFICIENT	卡箱卡少
CBS_ENOUGH	卡箱卡足

fCaptureBoxFull 值为TRUE，表示回收箱满；值为FALSE，表示回收箱未满。

错误码描述

F3_S_SUCCESS	操作成功。
F3_E_PORT_UNAVAILABLE	指定的COM端口不存在，或者被其它程序占用。
F3_E_DEV_NOT_RECOGNIZED	未检测到设备。可能的原因： 1 指定的COM端口不正确 2 指定的波特率不正确 3 指定的卡机地址不正确 4 串口线有问题 (注：仅在调用 F3_Connect 函数时才有可能返回该错误。)
F3_E_COMM_ERROR	通信错误。可能的原因： 1 通信过程中接收的字符在通信协议上未定义。 2 返回的响应消息的包头、包尾或BCC不正确。 3 返回的响应数据的长度与通信协义上定义的不一致。
F3_E_COMM_TIMEOUT	通信超时。
F3_E_UNKNOWN_ERROR	检测到一个内部错误，但原因不明。
F3_E_MESSAGE_TOO_LONG	命令消息或接收的响应消息的长度超过了1024个字符。
F3_E_NO_MEMORY	没有足够的内存来完成当前的操作。
F3_E_BUFFER_TOO_SMALL	接收返回数据的缓冲区太小。
F3_E_INVALID_HANDLE	提供的句柄无效。
F3_E_UNDEFINED_COMMAND	未定义的命令。
F3_E_INVALID_PARAMETER	提供的一个或多个参数无效或者为NULL值。
F3_E_DISABLED_COMMAND	命令不能在当前的状态下执行。
F3_E_UNSUPPORTED_COMMAND	不支持的命令。
F3_E_CONTACT_NO_RELEASE	IC触点未释放。
F3_E_CARD_JAMMED	卡片堵塞。

F3_E_SENSOR_ABNORMALITY	传感器异常。
F3_E_TOO_LONG_CARD	插入到卡机内的卡片过长。
F3_E_TOO_SHORT_CARD	插入到卡机内的卡片过短。
F3_E_CARD_WITHDRAWN	回收卡时卡片被拿走。
F3_E_IC_SOLENOID_ERROR	IC电磁线圈错误。
F3_E_CANT_MOVED_TO_IC_POS	不能移动卡到IC触点位。
F3_E_CARD_POSITION_CHANGE	卡片位置被人为改变。
F3_E_COUNTER_OVERFLOW	回收卡计数器溢出。
F3_E_MOTOR_ABNORMALITY	马达异常。
F3_E_POWER_SHORT	IC卡供电电源短路。
F3_E_ICC_ACTIVATION_ERROR	IC卡激活错误。
F3_E_ICC_NOT_ACTIVATED	IC卡未激活。
F3_E_UNSUPPORTED_ICC	不支持的IC卡。
F3_E_ICC_RECEPTION_ERROR	从IC卡接收数据时出错。
F3_E_ICC_COMM_TIMEOUT	IC卡通信超时。
F3_E_MISMATCH_EMV	CPU/SAM卡不符合EMV2000规范。
F3_E_CARD_BOX_EMPTY	卡箱空。
F3_E_CAPTURE_BOX_FULL	回收箱满。
F3_E_WAITING_FOR_RESET	等待复位。
F3_E_COMMAND_FAILURE	命令执行失败。
F3_E_DISAGREEMENT_OF_VC	校验卡时提供的校验码不正确。
F3_E_CARD_LOCKED	卡片已被锁。
F3_E_ADDRESS_OVERFLOW	操作地址溢出。
F3_E_LENGTH_OVERFLOW	操作长度溢出。