

MPOS-IOS-SDK

接口说明

V2.5

福建魔方电子科技有限公司

目 录

1.	工程设置.....	3
1.1.	SDK 导入.....	3
1.2.	Framework 依赖.....	3
2.	SDK 使用说明.....	3
3.	接口函数说明.....	4
3.1.	打开蓝牙刷卡器.....	4
3.2.	搜索蓝牙设备.....	4
3.3.	停止蓝牙搜索.....	5
3.4.	连接一个蓝牙设备.....	6
3.5.	断开蓝牙连接.....	6
3.6.	获取 SDK 版本号.....	6
3.7.	查询 MPOS 连接状态.....	6
3.8.	设置数据接收超时.....	7
3.9.	数据接收超时(回调).....	7
3.10.	MPOS 交易.....	7
3.11.	取消刷卡操作.....	9
3.12.	计算 MAC.....	9
3.13.	输入密码.....	10
3.14.	下载 KEK.....	11
3.15.	下载主密钥.....	11
3.16.	下载工作密钥.....	12
3.17.	密钥选择.....	13
3.18.	设置 IC 卡公钥.....	13
3.19.	设置 AID 参数.....	14
3.20.	执行 IC 卡二次授权.....	14
3.21.	获取设备信息.....	15
3.22.	获取随机数.....	15
3.23.	蜂鸣器.....	16
3.24.	设置终端时间、厂商 ID.....	16
3.25.	获取终端时间.....	17
3.26.	更新 MPOS 固件.....	17
3.27.	数据加密.....	17
3.28.	自定义数据写入.....	18
3.29.	自定义数据读取.....	18
3.30.	读取设备 KSN 信息.....	19
4.	附录.....	19
4.1.	附录 A.....	19
	MFMFEU_READER_SESSION 消息取值及含义.....	19
4.2.	附录 B.....	20
	MFEU_READER_RESULT 消息取值及含义.....	20
4.3.	附录 C.....	22
	MFEU_TRADE_TYPE 取值定义.....	22

4. 4.	附录 D IC 卡公钥和 AID 格式说明.....	23
4. 5.	附录 E 读卡交易一般步骤.....	27

1. 工程设置

1. 1. SDK导入

SDK 主要分为库文件和头文件两个部分。首先需要把 SDK 目录下的 .a 和 .h 文件加入到工程文件中

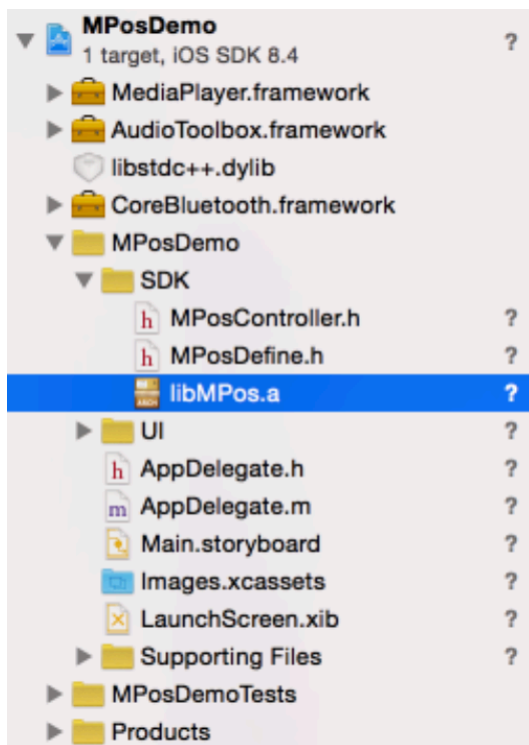
1. 2. Framework依赖

CoreBluetooth.framework

蓝牙 framework 层支持

Libstdc++.dylib

标准 C++库依赖



2. SDK使用说明

- 初始化MPosController对象
- 打开、连接设备
- 调用命令接口
- 获取终端接口返回结果数据回调

注：由于手机与刷卡器是一问一答的通讯方式，终端一次只能接收一条指令。因此业务中需要发送几条连续的指令，请在上一条指令回调函数时调用

```
#import "../SDK/MPosController.h"
```

```
@interface MainTableViewController () <MPosDelegate>
```

```
{
```

```
}
```

```
@property (strong, nonatomic) MPosController *posCtrl;
```

```
@end
```

```
- (void)viewDidLoad
```

```
{
    [super viewDidLoad];
    // 初始化MPosController对象
    self.posCtrl = [MPosController sharedInstance];
    self.posCtrl.delegate = self;

    [self.posCtrl openBtDevice];    // 打开蓝牙适配器
}
// 设备连接回调
- (void)didConnected:(NSString *)devName
{
    // 连接后务必设置厂商id号(默认为0,具体ID分配请与我们联系)
    [self.posCtrl setFactoryCode: 0];
}
// 获取设备信息
- (void) getPosInfo
{
    [self.posCtrl readPosInfo];
}
// 获取设备版本信息
- (void) didReadPosInfoResp:(NSString *)ksn status:
(MFEU_MSR_DEVSTAT)status battery: (MFEU_MSR_BATTERY)battery
app_ver: (NSString *)app_ver data_ver: (NSString *)data_ver
custom_info: (NSString *)custom_info
{
    [self alertMsg: [NSString stringWithFormat: @"KSN: %@\n 电池
等级: %d\n版本号: %@\n数据版本号: %@\n厂商自定义信息:\n%@",
        , ksn, battery, app_ver, data_ver, custom_info]];
}
}
```

3. 接口函数说明

3.1. 打开蓝牙刷卡器

函数原型:

```
-(void) openBtDevice;
```

说明: 设置为蓝牙驱动

参数说明: 空

3.2. 搜索蓝牙设备

函数原型:

```
-(void) scanBtDevice:(NSInteger)timeout;
```

说明:

设置为蓝牙驱动, 并开始搜索附近的蓝牙设备

参数说明:

参数名	取值含义
timeout	蓝牙搜索超时时间

相关 Delegate:

```
-(void) didFoundBtDevice:(NSString *)btDevice;
```

参数说明:

参数名	取值含义
btDevice	蓝牙信息 (格式: name,uuid)

备注:

为了提高蓝牙的兼容性, 我们采用广播方式获取蓝牙设备名称。因此建议直接采用我们的 SDK 接口进行蓝牙设备搜索操作, 同时完全兼容正常的搜索方式。

如果贵公司的 APP 是自己实现蓝牙搜索功能, 则需要在蓝牙搜索广播中加入代码判断一下, 具体代码如下:

```
-(void)addPeripheral:(CBPeripheral*)peripheral
advertisementData:(NSDictionary*)advertisementData
RSSI:(NSNumber*)RSSI
{
    .....
    if (advertisementData) {
        NSLog(@"advertisementData: %@", advertisementData);
        NSData *data = [advertisementData objectForKey:
@"kCBAAdvDataManufacturerData"];
        if (data != nil) {
            //02264d46 2d394539 44414531 32
            // 魔方设备, 广播kCBAAdvDataManufactureerData,以0x0226开
            unsigned char *dbyte = (unsigned char *)[data
bytes];
            NSString *name;
            if (dbyte[0] == 0x02 && dbyte[1] == 0x26) {
                name = [[NSString alloc] initWithBytes: dbyte + 2
length: [data length]-2 encoding::NSUTF8StringEncoding];
                pi.localName = name;
                pi.name = name;
            }
        }
        .....
    }
    .....
}
```

3.3. 停止蓝牙搜索

函数原型:

```
-(void) stopScan;
```

说明: 停止当前蓝牙设备搜索操作

参数说明: 空

相关 Delegate:

-(void) didStopScanBtDevice;

3.4. 连接一个蓝牙设备

函数原型:

-(void) connectBtDevice:(NSString *)btDevice;

参数说明:

参数名	取值含义
btDevice	蓝牙信息 (格式: name,uuid)

-(void) connectBtDevice:(NSString *)btDevice
connectTimeout:(NSInteger) timeout;

参数说明:

参数名	取值含义
btDevice	蓝牙信息 (格式: name,uuid)
timeout	蓝牙连接超时时间

注: 这个方法不需要进行蓝牙搜索即可连接设备

相关 Delegate:

-(void) didConnected:(NSString *)devName;

参数说明:

参数名	取值含义
devName	蓝牙信息 (格式: name,uuid)

-(void) didConnectFail;

说明: 蓝牙设备连接失败(超时)

3.5. 断开蓝牙连接

函数原型:

-(void) disconnectBtDevice;

参数说明: 空

相关 Delegate:

-(void) didDisconnected;

3.6. 获取SDK版本号

函数原型:

-(NSString *) getVersion;

说明: 获取 SDK 版本号

参数说明: 空

3.7. 查询MPOS连接状态

函数原型:

-(NSInteger) getDeviceState;

说明:

查询 MPOS 是否已连接上。

返回值说明：

驱动类型	取值含义
音频	-1 设备未插入, 0 设备未连接, 1 设备已连接
蓝牙	0 设备未连接, 1 设备已连接

3.8. 设置数据接收超时

函数原型：

```
-(void) setTimeout: (NSInteger) timeout;
```

说明：

设置数据接收超时时间

参数说明：

参数名	取值含义
timeout	超时时间, 单位: 秒

3.9. 数据接收超时(回调)

函数原型：

```
-(void) didTimeout;
```

说明：

一般在发出指令后, 在超时时间之内未接收到应答数据时, 产生该回调

3.10. MPOS交易

● 指令请求函数原型：

```
-(int) mPosTrade: (MFEU_READCARD_TYPE) cardType
    cardTimeout: (unsigned char) cardTimeout
    tradeDes: (NSString *) tradeDes
    tradeAmt: (int) nAmt
    factoryId: (unsigned char) cFactory
    authAmt: (int) nAuthAmt
    otherAmt: (int) nOtherAmt
    tradeType: (MFEU_TRADE_TYPE) tradeType
    pbocFlow: (MFEU_PBOC_FLOW) pboc
    ecashTrade: (MFEU_ECASH_TRADE) ecash
    onlineTrade: (MFEU_IC_ONLINE) online
    pinReq: (MFEU_PINREQ) pinreq
    pwdMaxLength: (unsigned char) nPwdMaxLength
    pwdTimeout: (unsigned char) nPwdTimeout
    enableFailback: (MFEU_FAILBACK) failback;
    flowNo: (NSString *)flowNo
    orderNo: (NSString *)orderNo
```

说明：

MPOS 读卡全过程, 最终返回卡片交易数据等

参数说明：

参数名	取值含义
-----	------

cardType	支持卡类型 MF_READ_TRACK 读取磁道信息 MF_IC_PRESENT 检查 IC 是否在位 MF_COMBINED 读磁条+IC MF_READ_RFID 读取 RFID MF_READ_ALL 读磁条+IC+RFID
cardTimeout	刷、插、挥卡超时时间(单位: 秒)
tradeDes	终端显示内容
tradeAmt	交易金额, 以分为单位
factoryId	厂商 ID, 默认为 0
authAmt	授权金额, 以分为单位
otherAmt	其他金额, 以分为单位
transType	交易类型, 详见附录 C
pbocFlow	PBOC 流程指示
ecashTrade	是否支持电子现金
onlineTrade	是否强制联机标识
pinReq	PIN 输入指示
pwdMaxLength	密码的最大长度 <=0x0C
pwdTimeout	密码超时时间(1-255, 单位: 秒)
enableFailback	是否允许降级
flowNo	流水号(6 位)
orderNo	订单号(最大 20 位)

● 结果返回 Delegate 原型:

```
-(void)didMPosTradeResult:(NSDictionary *)dicResult;
```

NSDictionary 返回域值说明:

Key 名称	结果说明
cardResp	读卡方式 0: 用户取消, 1: 磁条卡交易 2: IC 卡交易, 3: 非接交易 4: 刷卡超时, 5: 读卡失败
maskedPAN	主账号
expiryDate	有效期
serviceCode	服务代码
track2Length	二磁长度
track3Length	三磁长度
track2Data	二磁数据
track3Data	三磁数据
randomNumber	随机数
plainLength	加密前长度
data55	交易相关数据
enableFailback	是否降级
pwdLength	密码长度
pinBlock	密码加密后的值

serialNum	卡号序列号
KSN	20 位序列号+16 位 psam 号
MAC	MAC
MAC_RAND	MAC 随机数

3. 11. 取消刷卡操作

- 指令请求函数原型:

```
-(NSInteger) resetPos;
```

说明:

应用上位端主动取消刷卡操作

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

```
-(void) didResetPosResp: (MFEU_MSR_RESP) resp;
```

返回值说明:

参数名	定义说明
resp	结果返回码。取值如下: MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败

3. 12. 计算MAC

- 指令请求函数原型 1:

```
-(NSInteger) calcMac: (NSString *)data  
                  macAlg: (MFEU_MAC_MFEU_ALG) macAlg;
```

说明:

根据保存在 MPOS 的工作密钥, 计算传入的报文的 MAC

参数说明:

参数名	取值含义
data	待计算的报文(ASC 格式)
macAlg	mac 算法. 取值如下: MF_MACALG_UB UBC 算法 MF_MACALG_X99 X99/X919 MF_MACALG_EBC EBC 算法

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 指令请求函数原型 2:

```
-(NSInteger) calcMac2: (NSString *)data  
                  macAlg: (MFEU_MAC_MFEU_ALG) macAlg;
```

说明:

根据保存在 MPOS 的工作密钥, 计算传入的报文的 MAC

参数说明:

参数名	取值含义
data	待计算的报文(ASC 格式), 内容将转换为 hex 格式

	如 data="@123456", 实际给终端的内容是 0x12 0x34 0x56
macAlg	mac 算法, 取值如下: MF_MACALG_UB UBC算法 MF_MACALG_X99 X99/X919 MF_MACALG_EBC EBC 算法

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回函数原型:

```
-(void) didCalcMacResp: (NSString *)mac
                string: (NSString *)text
        randomNumber: (NSString *)randNumber
        randomNumstr: (NSString *)randNumstr;
```

参数说明:

成员变量	取值含义
mac	计算出来的 mac (ASC 格式)
text	当 mac 返回值为 BCD 压缩格式时, 转成可见的 ASC 格式
randNumber	计算出来的 mac 随机数 (ASC 格式)
randNumstr	当 mac 随机数为 BCD 压缩格式时, 转成可见的 ASC 格式

3.13. 输入密码

- 指令请求函数原型:

```
-(NSInteger) inputPin: (NSInteger) maxlen
                timeout: (NSInteger)timeout
                maskedPAN: (NSString *)pan;
```

说明:

输入密码, 返回加密后的 PINBLOCK

参数说明:

参数名	取值含义
maxlen	输入密码的最大长度
timeout	超时时间, s 为单位
pszPan	主账号

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

```
-(void) didInputPinResp: (MFEU_MSR_KEYTYPE) type
                pwdLength: (NSInteger) len
                pwdText: (NSString *)text;
```

参数说明:

成员变量	取值含义
type	返回按键类型。取值如下: MF_KEYTYPE_OK: 用户按确认键

	MF_KEYTYPE_CANCEL: 用户按取消键
len	输入密码的长度
text	加密后的 pinblock (ASC 格式)

3.14. 下载KEK

- 指令请求函数原型:

```
-(NSInteger)loadKek: (NSString *)kek
                keyLength: (MFEU_KEY_LENGTH)len;
```

说明:

下载 KEK 到终端

参数说明:

参数名	取值含义
kek	长度为40个字符串, 最后8位是检验值 (BCD格式)
len	Kek 类型. 取值如下: MF_KEY_SINGLE, 单倍长 MF_KEY_DOUBLE, 双倍长

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

```
-(void) didLoadKekResp: (MFEU_MSR_RESP)resp;
```

返回值说明:

参数名	定义说明
resp	结果返回码。取值如下: MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败

3.15. 下载主密钥

- 指令请求函数原型:

```
-(NSInteger)loadMainKey: (NSString *)mainKey
                encryptMethod: (MFEU_ENCRYPT_METHOD)method
                keyIndex: (MFEU_KEY_INDEX)index
                keyLength: (MFEU_KEY_LENGTH)len;
```

说明:

下载主密钥到终端

参数说明:

参数名	取值含义
mainKey	40 个字符串, 最后 8 位是检验值 (BCD 格式)
method	加密方式. 取值如下: MF_ENCRYPT_KEY: 采用 KEK 加密 MF_ENCRYPT_MAINKEY: 采用原主密钥加密 MF_ENCRYPT_PLAINTEXT: 明文

	MF_ENCRYPT_C_FJPOSTAR: 国通新驿模式
index	主密钥索引。取值如下: MF_KEY_IND_0 ~ MF_KEY_IND_9, 分别对应索引 0~索引 9
len	主密钥类型。取值如下: MF_KEY_SINGLE, 单倍长 MF_KEY_DOUBLE, 双倍长

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

-(void) didLoadMainKeyResp: (MFEU_MSR_RESP) resp;

参数说明:

参数名	定义说明
resp	结果返回类型: MFEU_MSR_RESP。取值如下: MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败

3.16. 下载工作密钥

- 指令请求函数原型:

-(NSInteger) loadWorkKey: (NSString *)workKey
macKey: (NSString *)mac
trackKey: (NSString *)track
keyIndex: (MFEU_KEY_INDEX) index;

说明:

下载工作密钥到终端

参数说明:

参数名	取值含义
pin	pin 密钥数据, ASC 格式, 40 位=32(key)+8(kvc)
mac	mac 密钥数据, ASC 格式, 40 位=32(key)+8(kvc)
track	磁道密钥数据, ASC 格式, 40 位=32(key)+8(kvc)
index	工作密钥索引。取值如下: MF_KEY_IND_0 ~ MF_KEY_IND_9, 分别对应索引 0~索引 9

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

备注:

如果 pin/mac/磁道密钥数据长度只有 16 位时, 必须填补成 32 位(直接复制即可), 加上 8 位 kvc。如果为空, 任意填补其他有值的密钥数据

- 结果返回 Delegate 原型:

-(void) didLoadWorkKeyResp: (MFEU_MSR_RESP) resp;

参数说明:

参数名	定义说明
resp	结果返回类型: MFEU_MSR_RESP。取值如下:

	MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败
--	---

3.17. 密钥选择

- 指令请求函数原型:

-(NSInteger) setKeyIndex: (MFEU_KEY_INDEX)index;

说明:

选择进行加密的密钥。出厂默认为 0

参数说明:

参数名	取值含义
index	密钥索引。取值如下: MF_KEY_IND_0 ~ MF_KEY_IND_9, 分别对应索引 0~索引 9

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

-(void) didSetKeyIndexResp: (MFEU_MSR_RESP)resp;

返回值说明:

参数名	定义说明
resp	结果返回类型: MFEU_MSR_RESP。取值如下: MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败

3.18. 设置IC卡公钥

- 指令请求函数原型:

-(void) setIcKey: (NSArray *)dataArray;

说明:

设置 IC 卡公钥

参数说明:

参数名	取值含义
dataArray	公钥信息(ASC 格式)

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

-(void) didSetICKeyResp: (NSInteger)index
totalCount: (NSInteger)total;

参数说明:

参数名	取值含义
index	当前进度值
total	总公钥数

3.19. 设置AID参数

- 指令请求函数原型:

-(void) setIcAid: (NSArray *)dataArray;

说明:

设置 IC 卡公钥

参数说明:

参数名	取值含义
dataArray	AID 参数(ASC 格式)

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

-(void) didSetAidResp: (NSInteger)index
totalCount: (NSInteger)total;

参数说明:

参数名	取值含义
index	当前进度值
total	总公钥数

3.20. 执行IC卡二次授权

- 指令请求函数原型:

-(NSInteger) icDealOnline: (NSString *)data
onlineResult: (MFEU_ONLINE_RESULT)result;

说明:

执行二次授权, 即 IC 卡联机之后的后续处理

参数说明:

参数名	取值含义
data	8583 应答报文的 55 域值(ASC 格式)
result	是否联机成功, 取值如下: MF_ONLINE_SUCC, 联机成功 MF_ONLINE_FAIL, 联机未成功

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

-(void) didIcDealOnlineResp: (MFEU_MSR_REAUTH_RESP) resp;

参数说明:

参数名	MFEU_READER_REAUTH_RESP
resp	结果返回码。取值如下: MF_RESP_REAUTH_UNKNOWN, 未知错误 MF_RESP_REAUTH_ACCEPT, 交易授受 MF_RESP_REAUTH_REJECT, 二次授权交易拒绝 MF_RESP_REAUTH_FAIL, 交易失败

3. 21. 获取设备信息

- 指令请求函数原型:

–(NSInteger) readPosInfo;

- 指令扩展函数原型:

–(NSInteger) readPosInfoEx;

说明:

获取设备机身号, 电量状态等信息

扩展指令接口, 支持接收大于 15 位长度的设备序列号

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

```
–(void) didReadPosInfoResp:(NSString *)ksn
    status: (MFEU_MSR_DEVSTAT)status
    battery: (MFEU_MSR_BATTERY)battery
    app_ver: (NSString *)app_ver
    data_ver: (NSString *)data_ver
    custom_info: (NSString *)custom_info;
```

参数说明:

参数名	取值含义
ksn	机身号
status	MPOS 状态。取值如下: MF_DEVSTAT_DEFAULT, 默认初始状态 MF_DEVSTAT_WKEYIN, 工作密钥已灌装 MF_DEVSTAT_MKEYIN, 主密钥已灌装 MF_DEVSTAT_KEKMOD, KEK 已修改
battery	设备电池电量状态, 取值如下: MF_BATTERY_CAPACITY_UNKOWN, 未知状态 MF_BATTERY_CAPACITY_CRITICAL, 临界 MF_BATTERY_CAPACITY_LOW, 低压 MF_BATTERY_CAPACITY_NORMAL, 正常 MF_BATTERY_CAPACITY_HIGH, 电量充足 MF_BATTERY_CAPACITY_FULL, 满电 注: 当该值小于等于 MF_BATTERY_CAPACITY_LOW 时, 表示设备低电, 请提示用户充电
app_ver	应用版本号
data_ver	数据版本号
custom_info	厂商自定义信息, 用于存储特定数据

3. 22. 获取随机数

- 指令请求函数原型:

–(NSInteger) getRandomNum;

说明:

获取随机数

返回值说明:

正常返回蓝牙发送字节数，-1 表示：蓝牙未准备好

- 结果返回函数原型：

-(void) didGetRandNumResp: (NSString *)randNum;

参数说明：

成员变量	取值含义
randNum	MPOS 生成的随机数(ASC 格式)

3. 23. 蜂鸣器

- 指令请求函数原型：

-(NSInteger) beep: (NSInteger)times
 freq: (NSInteger)freq
 duration: (NSInteger)duration
 step: (NSInteger) step;

说明：

触发 MPOS 蜂鸣器

参数说明：

参数名	取值含义
times	蜂鸣次数
freq	蜂鸣频率, 单位 hz
duration	每次蜂鸣时间, 单位 ms
step	次蜂鸣时间间隔, 单位 ms

返回值说明：

正常返回蓝牙发送字节数，-1 表示：蓝牙未准备好

- 结果返回 Delegate 原型：

-(void) didBeepResp;

3. 24. 设置终端时间、厂商ID

- 指令请求函数原型：

-(NSInteger) setDatetime: (NSString *)datetime
 factoryId: (NSInteger)factoryid;

说明：

设置 MPOS 的时间。需要确保 MPOS 的时间为当前时间

参数说明：

参数名	取值含义
datetime	时间，YYYYMMDDHHMMSS 格式, 长度 14 位
cFactory	厂商 I D，默认为 0，特殊要求请与我们联系

- 设置厂商 ID 函数原型：

-(NSInteger)setFactoryCode: (NSInteger)fCode;

说明：

设置厂商 ID，并写入当前时间

参数说明：

参数名	取值含义
-----	------

fCode	厂商 I D，初始化默认为 0, 即返回的数据不加密 具体 ID 分配，请与我们联系
-------	---

- 结果返回 Delegate 原型:

-(void) didSetDatetimeResp;

3. 25. 获取终端时间

- 指令请求函数原型:

-(NSInteger) getDatetime;

说明:

获取 MPOS 的当前时间

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

- 结果返回函数原型:

-(void) didGetDatetimeResp: (NSString *)datetime;

参数说明:

参数名	取值含义
datetime	时间, YYYYMMDDHHMMSS 格式, 长度 14 位

3. 26. 更新MPOS固件

- 函数原型:

-(NSInteger) updatePos: (NSString *)upgradeFilename;

说明:

指定文件名进行固件更新

参数说明:

参数名	取值含义
upgradeFilename	指定的文件名, 确保文件位于当前应用目录上即可

返回值说明:

正常返回大于 0, -1 表示: 蓝牙未准备好

- 结果返回 Delegate 原型:

-(void) didUpgradeResp: (NSInteger) pos
size: (NSInteger) length;

说明: 升级下载进度

参数名	取值含义
pos	当前进度值
length	总文件大小

-(void) didUpgradeFinish;

说明: 升级完成

3. 27. 数据加密

- 指令请求函数原型:

-(NSInteger) dataEncode: (NSString *)data
encryptKey: (NSString *)key;

说明:

用主密钥解密密钥密文后得到密钥明文, 用密钥明文对输入数据采用密钥加密

参数说明:

参数名	取值含义
data	要加密的数据(ASC 格式)
key	密钥密文使用主密钥加密(ASC格式)

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

● 结果返回 Delegate 原型:

-(void) didDataEncodeResp: (NSString *)data;

返回值说明:

参数名	取值含义
data	加密数据(ASC 格式)

3.28. 自定义数据写入

● 指令请求函数原型:

-(NSInteger) dataWrite: (NSString *)data;

说明:

在指定位置上写入用户自定义数据 (注: 自定义数据区大小为 1K 字节)

参数说明:

参数名	取值含义
data	要写入的数据

返回值说明:

正常返回蓝牙发送字节数, -1 表示: 蓝牙未准备好

● 结果返回 Delegate 原型:

-(void) didDataWriteResp: (MFEU_MSR_RESP) resp;

返回值说明:

参数名	定义说明
resp	结果返回码。取值如下: MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败

3.29. 自定义数据读取

● 指令请求函数原型:

-(NSInteger) dataRead;

说明:

用主密钥解密密钥密文后得到密钥明文, 用密钥明文对输入数据采用密钥加密

参数说明: 空

返回值说明：

正常返回蓝牙发送字节数，-1 表示：蓝牙未准备好

- 结果返回 Delegate 原型：

```
-(void) didDataReadResp: (MFEU_MSR_RESP)resp  
        dataRead: (NSString *)data;
```

说明：

参数名	取值含义
resp	返回结果(类型: MFEU_MSR_RESP) MF_RESP_UNKNOWN, 未知错误 MF_RESP_SUCC, 成功 MF_RESP_FAIL, 失败 如果失败, 就不返回后续字段内容
sData	读取数据

3.30. 读取设备KSN信息

- 指令请求函数原型：

```
-(NSInteger) getKsn;
```

参数说明：空

返回值说明：

正常返回蓝牙发送字节数，-1 表示：蓝牙未准备好

- 结果返回 Delegate 原型：

```
-(void) didGetKsnResp: (NSString *)serial  
        PSAM: (NSString *)psamNo;
```

说明：

参数名	取值含义
serial	序列号, 最长 20 位
psamNo	PSAM 卡号, 最后 16 位

4. 附录

4.1. 附录A

MFMEU_READER_SESSION消息取值及含义

MF_SESSION_UNKNOWN	未知状态
MF_SESSION_SCAN_START	蓝牙搜索开始
MF_SESSION_SCAN_STOP	蓝牙搜索结束
MF_SESSION_CONN_FAIL	连接失败
MF_SESSION_CONN_VALID	连接合法设备
MF_SESSION_CONN_INVALID	连接非法设备
MF_SESSION_DISCONNECT	断开连接
MF_SESSION_AUDIO_CONNECT	音频设备接入
MF_SESSION_AUDIO_DISCONNECT	音频设备拔除
MF_SESSION_KEK_DOWNLOAD	KEK下载
MF_SESSION_MKEY_DOWNLOAD	主密钥下载

MF_SESSION_WKEY_DOWNLOAD	工作密钥下载
MF_SESSION_SELECT_PIN	密钥选择
MF_SESSION_INPUT_PIN	密码输入
MF_SESSION_CALC_MAC	MAC计算
MF_SESSION_OPEN_CARD	开启读卡器
MF_SESSION_READ_CARD	读磁条卡
MF_SESSION_SET_ICKEY	设置IC公钥
MF_SESSION_SET_AID	设置AID
MF_SESSION_SET_DATA	设置交易数据
MF_SESSION_START_EMV	开始IC交易流程
MF_SESSION_IC_REAUTH	IC二次授权
MF_SESSION_END_EMV	结束IC交易流程
MF_SESSION_GET_DEVINFO	读取设备信息
MF_SESSION_GET_RANDOM	获取随机数
MF_SESSION_BEEP	蜂鸣器
MF_SESSION_SET_DATETIME	设置时间日期
MF_SESSION_GET_DATETIME	获取时间日期
MF_SESSION_CLOSE_DEVICE	关闭设备
MF_SESSION_UPGRADE	升级应用/固件
MF_SESSION_DATA_ENCODE	数据加密
MF_SESSION_DATA_WRITE	自定义数据写入
MF_SESSION_DATA_READ	自定义数据读取
MF_SESSION_ICDATA	IC卡数据返回
MF_SESSION_EMVDATA_EX	55 域以及 IC 卡数据返回
MF_SESSION_EMVDATA_EX2	取 IC 卡数据(从 TAG 中获取)
MF_SESSION_JF_MAGCARD	读磁条卡(即富)
MF_SESSION_JF_ICDATA	取 IC 交易数据(即富)
MF_SESSION_JF_KSN	读序列号(即富)
MF_SESSION_GET_DEVINFO_EX	读取设备信息(扩展)

4. 2. 附录B

MFEU_READER_RESULT消息取值及含义

返回数据包字段出错	
MF_RET_FAIL	未知错误
MF_RET_FAIL_STX	STX字段出错
MF_RET_FAIL_LEN	LEN字段出错
MF_RET_FAIL_PATH	PATH字段出错
MF_RET_FAIL_TYPE	TYPE字段出错
MF_RET_FAIL_ID	ID不一致
MF_RET_FAIL_ETX	ETX字段出错
MF_RET_FAIL_LRC	LRC字段出错
响应码错误返回	
MF_RET_FAIL_CMD	指令码不支持

MF_RET_FAIL_PARAM	参数错误
MF_RET_FAIL_LENGTH	可变数据域长度错误
MF_RET_FAIL_FORMAT	帧格式错误
MF_RET_FAIL_GETLRC	LRC错误
MF_RET_FAIL_OTHER	其他错误
MF_RET_FAIL_TIMEOUT	超时
MF_RET_FAIL_STATUS	返回当前状态
正常接收（合法数据）	
MF_RET_KEK_DOWNLOAD	KEK下载
MF_RET_MKEY_DOWNLOAD	主密钥下载
MF_RET_WKEY_DOWNLOAD	工作密钥下载
MF_RET_SELECT_KEY	密钥选择
MF_RET_INPUT_PIN	密码输入
MF_RET_CALC_MAC	MAC计算
MF_RET_OPEN_CARD	开启读卡器
MF_RET_READ_CARD	读磁条卡
MF_RET_SET_ICKEY	设置IC公钥
MF_RET_SET_AID	设置AID
MF_RET_SET_DATA	设置交易数据
MF_RET_START_EMV	开始IC交易流程
MF_RET_IC_REAUTH	IC二次授权
MF_RET_END_EMV	结束IC交易流程
MF_RET_GET_DEVINFO	读取设备信息
MF_RET_GET_RANDOM	获取随机数
MF_RET_BEEP	蜂鸣器
MF_RET_SET_DATETIME	设置时间日期
MF_RET_GET_DATETIME	获取时间日期
MF_RET_CLOSE_DEVICE	关闭设备
MF_RET_UPGRADE	正在升级应用/固件
MF_RET_UPGRADE_FINISH	完成升级应用/固件
MF_RET_DATA_ENCODE	数据加密
MF_RET_DATA_WRITE	自定义数据写入
MF_RET_DATA_READ	自定义数据读取
MF_RET_ICDATA	获取IC卡数据
MF_RET_JF_MAGCARD	读磁条卡(即富)
MF_RET_JF_ICDATA	取IC卡交易数据(即富)
MF_RET_JF_KSN	取KSN号(即富)
MF_RET_GET_DEVINFO_EX	读取设备信息（扩展）
MF_RET_TIMEOUT	蓝牙接收超时
MF_RET_USER_CANCEL	用户取消

4.3. 附录C

MFEU_TRADE_TYPE取值定义

类型	MFEU_TRADE_TYPE
说明	<p>MF_FUNC_BALANC 查询</p> <p>MF_FUNC_SALE 消费</p> <p>//预授权类</p> <p>MF_FUNC_PREAUTH 预授权</p> <p>MF_FUNC_AUTHSALE 预授权完成请求</p> <p>MF_FUNC_AUTHSALEOFF 预授权完成通知</p> <p>MF_FUNC_AUTHSETTLE 预授权结算</p> <p>MF_FUNC_ADDTO_PREAUTH 追加预授权</p> <p>//退货类</p> <p>MF_FUNC_REFUND 退货</p> <p>//撤销类</p> <p>MF_FUNC_VOID_SALE 消费撤销</p> <p>MF_FUNC_VOID_AUTHSALE 预授权完成撤销</p> <p>MF_FUNC_VOID_AUTHSETTLE 结算撤销</p> <p>MF_FUNC_VOID_PREAUTH 预授权撤销</p> <p>MF_FUNC_VOID_REFUND 撤销退货</p> <p>//离线类</p> <p>MF_FUNC_OFFLINE 离线结算</p> <p>MF_FUNC_ADJUST 结算调整</p> <p>//电子钱包类</p> <p>MF_FUNC_EP_LOAD EP圈存</p> <p>MF_FUNC_EP_PURCHASE EP消费</p> <p>MF_FUNC_CASH_EP_LOAD 现金充值圈存</p> <p>MF_FUNC_NOT_BIND_EP_LOAD 非指定帐户圈存</p> <p>//分期类</p> <p>MF_FUNC_INSTALLMENT 分期付款</p> <p>MF_FUNC_VOID_INSTALLMENT 撤销分期</p> <p>//积分类</p> <p>MF_FUNC_BONUS_IIS_SALE 发卡行积分消费</p> <p>MF_FUNC_VOID_BONUS_IIS_SALE 撤销发卡行积分消费</p> <p>MF_FUNC_BONUS_ALLIANCE 联盟积分消费</p> <p>MF_FUNC_VOID_BONUS_ALLIANCE 撤销联盟积分消费</p> <p>MF_FUNC_ALLIANCE_BALANCE 联盟积分查询</p> <p>MF_FUNC_ALLIANCE_REFUND 联盟积分退货</p> <p>MF_FUNC_INTEGRALSIGNIN 收银员积分签到</p> <p>//电子现金类</p> <p>MF_FUNC_QPBOC 快速消费</p> <p>MF_FUNC_EC_PURCHASE 电子现金消费</p> <p>MF_FUNC_EC_LOAD 电子现金指定账户圈存</p> <p>MF_FUNC_EC_LOAD_CASH 电子现金圈存现金</p> <p>MF_FUNC_EC_NOT_BIND_OUT 电子现金非指定账户圈存转出</p>

MF_FUNC_EC_NOT_BIND_IN	电子现金非指定账户转入
MF_FUNC_EC_VOID_LOAD_CASH	电子现金圈存现金撤销
MF_FUNC_EC_REFUND	电子现金脱机退货
MF_FUNC_EC_BALANCE	电子现金余额查询
//无卡类	
MF_FUNC_APPOINTMENT_SALE	无卡预约消费
MF_FUNC_VOID_APPOINTMENT_SALE	撤销无卡预约消费
//磁条充值类	
MF_FUNC_MAG_LOAD_CASH	磁条预付费卡现金充值
MF_FUNC_MAG_LOAD_ACCOUNT	磁条预付费卡账户充值
//手机芯片类	
MF_FUNC_PHONE_SALE	手机芯片消费
MF_FUNC_VOID_PHONE_SALE	撤销手机芯片消费
MF_FUNC_REFUND_PHONE_SALE	手机芯片退货
MF_FUNC_PHONE_PREAUTH	手机芯片预授权
MF_FUNC_VOID_PHONE_PREAUTH	撤销手机芯片预授权
MF_FUNC_PHONE_AUTHSALE	手机芯片预授权完成
MF_FUNC_PHONE_AUTHSALEOFF	手机芯片完成通知
MF_FUNC_VOID_PHONE_AUTHSALE	撤销手机完成请求
MF_FUNC_PHONE_BALANCE	手机芯片余额查询
MF_FUNC_ORDER_SALE	订购消费
MF_FUNC_VOID_ORDER_SALE	订购消费撤销
MF_FUNC_ORDER_PREAUTH	订购预授权
MF_FUNC_VOID_ORDER_PREAUTH	订购预授权撤销
MF_FUNC_ORDER_AUTHSALE	订购预授权完成
MF_FUNC_VOID_ORDER_AUTHSALE	订购预授权完成撤销
MF_FUNC_ORDER_AUTHSALEOFF	订购预授权完成通知
MF_FUNC_ORDER_REFUND	订购退货
//其他	
MF_FUNC_EMV_SCRIPT	EMV脚本结果通知
MF_FUNC_EMV_REFUND	EMV脱机退货
MF_FUNC_PBOC_LOG	读PBOC日志
MF_FUNC_LOAD_LOG	读圈存日志
MF_FUNC_REVERSAL	冲正
MF_FUNC_TC	
MF_FUNC_SETTLE	结算

4.4. 附录D IC卡公钥和AID格式说明

当读取指定公钥时，返回内容参考如下标签列表内容

认证中心公钥类参数采用TLV（tag+length+value）格式表示，具体取值及含义为：

表1 认证中心公钥参数

参数名称	参数属性	参数长度 (byte)	参数 tag 值	参数含义	参数下载时间	参数适应场合
RID	b	5	9F06	与认证中心公钥索引一起标识认证中心的公钥	安装或调整时	交易应用
认证中心公钥索引	b	1	9F22	与 RID 一起标识认证中心的公钥	安装或调整时	交易应用
认证中心公钥有效期	n8	8	DF05	认证中心规定的有效期限	安装或调整时	交易应用
认证中心公钥哈希算法标识	b	1	DF06	标识用于在数字签名方案中产生哈希结果的哈希算法	安装或调整时	交易应用
认证中心公钥算法标识	b	1	DF07	标识使用在认证中心公钥上的数字签名算法	安装或调整时	交易应用
认证中心公钥模	b	变长, 最大为 248	DF02	公钥模值	安装或调整时	交易应用
认证中心公钥指数	b	1 或 3	DF04	公钥指数	安装或调整时	交易应用
认证中心公钥校验值	b	变长	DF03	验证认证中心公钥用	安装或调整时	交易应用
注: 认证中心公钥校验值的计算内容为RID+认证中心公钥索引+认证中心公钥模+认证中心公钥指数; 认证中心公钥校验值的计算方法为SHA-1。						

当读取指定 AID 时, 返回内容参考如下标签列表内容

表2 IC 卡 AID 参数列表

参数名称	参数属性	参数长度 (byte)	参数 tag 值	参数含义	参数下载时间	参数适用场合
AID	b	5-16	9F06	终端支持的借/贷记应用列表, 如 ISO/IEC 7816-5 所述, 指明应用	安装或调整时	交易应用
应用选择指示符 (ASI)	b	1	DF01	指示应用选择时终端上的 AID 与卡片中的 AID 是完全匹配 (长度和内容都必须一样), 还是部分匹配 (卡片 AID 的前面部分与终端 AID 相同, 长度可以更长)。终端支持的应用列表中的每个 AID 仅有一个应用选择指示符。	安装或调整时	交易应用
应用版本号	b	2	9F09	支付系统给应用分配的版本号	安装或调整时	交易应用

参数名称	参数属性	参数长度 (byte)	参数 tag 值	参数含义	参数下载时间	参数适用场合
TAC—缺省	b	5	DF11	标识如果交易可以联机完成但终端没有联机交易能力时，拒绝交易的收单行条件	安装或调整时	交易应用
TAC—联机	b	5	DF12	标识联机交易的收单行条件	安装或调整时	交易应用
TAC—拒绝	b	5	DF13	标识不作联机尝试即拒绝交易的收单行条件	安装或调整时	交易应用
终端最低限额	b	4	9F1B	IC 卡消费时终端允许的最低脱机限额	安装或调整时	交易应用
偏置随机选择的 阈值	b	4	DF15	在终端风险管理中用于随机交易选择的值	安装或调整时	交易应用
偏置随机选择的 最大目标百分数	cn (包含两位有效数字)	1	DF16	用于偏置随机选择的最大目标百分数	安装或调整时	交易应用
随机选择的目标 百分数	cn (包含两位有效数字)	1	DF17	用于随机选择的目标百分数	安装或调整时	交易应用
缺省 DDOL	b	变长	DF14	卡片中无 DDOL 时用于构造内部认证命令的 DDOL	安装或调整时	交易应用
终端联机 PIN 支持能力	b	1	DF18	指示终端在每个 AID 的要求下是否支持联机 PIN 的输入。	安装或调整时	交易应用 当值为 00000001 时表示支持联机 PIN。当值为 00000000 时表示不支持联机 PIN。
终端电子现金交易限额	cn	6	9F7B	终端使用此数据元（如果存在的话）判断一个交易的处理方式，当授权金额小于该限额时允许电子现金交易，否则设置终端行为代码并根据判断确认交易方式（小额支付参数）。	安装或调整时	交易应用

参数名称	参数属性	参数长度 (byte)	参数 tag 值	参数含义	参数下载时间	参数适用场合
非接触读写器脱机最低限额	cn	6	DF19	在 AID 联合中，用来指示读写器中非接触交易的最低限额	安装或调整时	交易应用
非接触读写器交易限额	cn	6	DF20	如果非接触交易的金额大于或等于此数值，则交易终止。允许在其他界面尝试此交易	安装或调整时	交易应用
读写器持卡人验证方法（CVM）所需限制	cn	6	DF21	如果非接触交易超过此值，读写器要求一个持卡人验证方法（CVM）。	安装或调整时	交易应用

4.5. 附录E 读卡交易一般步骤

