MPOS-IOS-SDK

Programming Manual

V1.0

Fujian Morefun Electronic Technology Co., Ltd.

Context Table

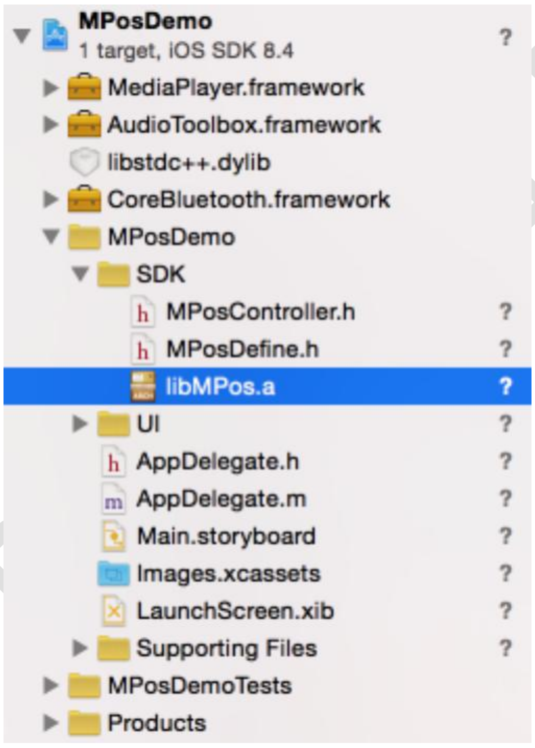
1. PROJECT SETTINGS

1.1. Import SDK file

SDK can be divided into two parts: library file and header file. First of all, you need to add .a and .h files to the project file in the SDK directory.

1.2. Linked Frameworks and library

CoreBluetooth.framework	Bluetooth framework support
Libstdc++.dylib	Standard C++ library dependencies



2. INSTRUCTIONS

- Initialization MPos Controller object
- Turn on & connect device
- Call API interface
- Gets the device interface to return the result data callback

PS; Because the mobile phone and the card reader are one question and one answer communication, the terminal can only receive one instruction at a time. Therefore, you need to send several consecutive instructions in the business. Please call on the last instruction callback function.

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

@interfaceMainTableViewController () <MPosDelegate>

{

}

@property (strong, nonatomic) MPosController *posCtrl;

@end


– (void)viewDidLoad

{

    [superviewDidLoad];

// Initialization MPosController object

self.posCtrl = [MPosControllersharedInstance];

self.posCtrl.delegate = self;


// Turn on BT device

    [self.posCtrl openBtDevice];

}

// Connected callback
```

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

{

        // Be sure to set the manufacturer ID number after the connection (default is

0, specific ID allocation, please contact us)

      [self.posCtrlsetFactoryCode: 0];

}

// Get Pos Information

-(void) getPosInfo

{

      [self.posCtrlreadPosInfo];

}

// Get Pos information callback

-(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 dev_model: (NSString *) model

{

      [selfalertMsg: [NSStringstringWithFormat: @"KSN: %@\n

Battery: %d\nAppVersion: %@\nDataVersion: %@\nCustom Info:\n%@\nDevice

Model: %@", ksn, battery, app_ver, data_ver, custom_info, model]];

}
```

3. INTERFACE FUNCTION SPECIFICATION

3.1. initBtDevice

Initialization bluetooth device

- Prototype

```
-(void) initBtDevice: (BOOL) isRepairConnect;
```

- Parameter

name	description
isRepairConnect	Bluetooth require a password pairing connection YES means require password NO mean direct connection

3.2. openBtDevice

Open bluetooth device

- Prototype

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

- Parameter

void

Note: Same effect as calling initBtDevice(NO);

3.3. scanBtDevice

Scan Bluetooth device

- Prototype

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

- Parameter

name	description
timeout	BT scan timeout time(Unit: ms)

- Delegate

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

name	description
btDevice	BT name (format: name,uuid)

3.4. stopBtDevice

Stop Bluetooth scan

- Function Prototype

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

- Parameter

void

- Related Delegate

```
-(void) didStopScanBtDevice;
```

3.5. connectBtDevice

Connect Bluetooth device

- Function Prototype

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

- Parameter

name	description
btDevice	BT device name(Format: name,uuid)
timeout	Connect timeout time

Remark: By this method, it can connect device without Bluetooth searching.

Related Delegate

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

name	description
btDevice	BT device name(Format: name,uuid)

```
-(void) didConnectFail;
```

3.6. disconnectBtDevice

Disconnect Bluetooth connected device

- Prototype

```
-(void) disconnectBtDevice;
```

- Parameter

void

- Delegate

```
-(void) didDisconnected;
```

3.7. getVersion

Get the SDK version number

- Prototype

```
-(NSString *) getVersion;
```

- Parameter

void

3.8. getDeviceState

Get connection status

- Prototype

```
-(NSInteger) getDeviceState;
```

- Return:

value	description
0	No connect
1	Connected

3.9. setTimeout

Set receive timeout time

- Prototype:

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

- Parameter

name	description
timeout	Timeout time(Unit: ms)

3.10. didTimeout

Receive timeout callback, normally it is generated when the response data is not received within the timeout period after the instruction is issued.

- Prototype:

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

3.11. mPosTrade

MPOS card read the whole process, and ultimately returned to card transactions, data, etc.

- Prototype:

```
-(int) mPosTrade: (MFEU_READCARD_TYPE) cardType
     cardTimeout: (unsignedchar) cardTimeout
        tradeDes: (NSString *) tradeDes
        tradeAmt: (int) nAmt
       factoryId: (unsignedchar) 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: (unsignedchar) nPwdMaxLength
      pwdTimeout: (unsignedchar) nPwdTimeout
  enableFailback: (MFEU_FAILBACK) failback;
```

```
        flowNo: (NSString *)flowNo
       orderNo: (NSString *)orderNo
```

- Parameter

name	description
cardType	Support card type: MF_READ_TRACK read track data MF_IC_PRESENT Check if IC is in position MF_COMBINED read magnetic stripe card & IC card MF_READ_RFID read RFID card MF_READ_ALL read magnetic stripe card & IC &RFID card
cardTimeout	Card operation timeout time(Unit: ms)
tradeDes	Display text on MPOS
tradeAmt	Transaction amount(Unit: fen)
factoryId	Factory id
authAmt	Authorization amount(Unit: fen), default: 0
otherAmt	Other amount(Unit: fen), default: 0
transType	Transaction type, see about Appendix C
pbocFlow	PBOC process instructions
ecashTrade	Enable/disable support electronic cash
onlineTrade	Enable/disable force online identification
pinReq	PIN input display
pwdMaxLength	Maximum length of password (<=0x0C)
pwdTimeout	Input password timeout time(Unit: s)
enableFailback	Enable/disable allowed to downgrade
flowNo	Serial number(Maxlength: 6)
orderNo	Order number(Maxlegnth: 20)

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

NSDirtionary return desciption:

name	description
cardResp	Operation card method: 0: user cancel 1: read magnetic stripe card 2: read IC card 3: read RFID card 4: read timeout 5: read fail
maskedPAN	Card account number
expiryDate	Card period of validity
serviceCode	Card service code(magcard only)
track2Length	Card data length of Track 2
track3Length	Card data length of Track 3(magcard only)
track2Data	Card data of Track 2
track3Data	Card data of Track 3(magcard only)
randomNumber	Transaction random number
plainLength	Plaintext length of data55
data55	Transaction related data
enableFailback	Enable/disable allowed to downgrade
pwdLength	Password length
pinBlock	PIN Block
serialNum	Card serial number
KSN	SN(20) + PSAM(16)

MAC	MAC value
MAC_RAND	MAC random number

3.12. resetPos

The application actively cancels the cards operation

- Prototype

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

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN Unknown error MF_RESP_SUCC send successly MF_RESP_FAIL send fail

3.13. calcMac

The MAC of incoming messages is calculated based on the work key stored(MK/SK) in MPOS.

- Prototype:

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

- Parameter

name	description
data	Data to be calculated (ASC format)
macAlg	MAC Algorithm. Values are as follows:

	MF_MACALG_UBC = 0x00, MF_MACALG_X99, MF_MACALG_EBC, MF_MACALG_ENCRYPT_UPAY = 0x0A, MF_MACALG_ENCRYPT_X99, MF_MACALG_ENCRYPT_X919, MF_MACALG_ENCRYPT_XOR

- Return:

value	description
-1	No connect
else	BT send data length

3.14. calcMac2

The MAC of incoming messages is calculated based on the work key stored(MK/SK) in MPOS.

- Prototype

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

- Parameter

name	description
data	Data to be calculated (ASC format) The content will be converted to hex format For example, data=@"123456", the actual content of the terminal is 0x12 0x34 0x56
macAlg	MAC Algorithm. Values are as follows: MF_MACALG_UBC = 0x00, MF_MACALG_X99, MF_MACALG_EBC, MF_MACALG_ENCRYPT_UPAY = 0x0A, MF_MACALG_ENCRYPT_X99, MF_MACALG_ENCRYPT_X919, MF_MACALG_ENCRYPT_XOR

- Return:

value	description
-1	No connect
else	BT send data length

● Delegate

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

name	description
mac	Calculated MAC (ASC format)
text	When the MAC returns the value of BCD compression format, it is converted to the visible ASC format
randNumber	Calculated MAC random number(ASC format)
randNumstr	When the MAC random number returns the value of BCD compression format, it is converted to the visible ASC format

3.15. InputPin

Enter the password and return the encrypted PINBLOCK.

● Prototype:

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

● Parameter

name	description
maxlen	Maximum length of password
timeout	Opertation timeout time

pan	Card account number

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description

type	Press key type, Values are as follows:
	MF_KEYTYPE_OK = 0x00, //!< Press OK
	MF_KEYTYPE_CANCEL = 0x06, //!< Press Cancel

len	Password length
text	Encrypted pinblock (ASC format)

3.16. loadKek

Download KEK to MPOS device

- Prototype:

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

- Parameter

name	description
kek	KEK(40 bits) value(BCD format). Format: key(32bits)+check value(8bits)

len	KEK type, Value are as follows:
	MF_LEN_SINGLE = 0x01, //!< Haploid length
	MF_LEN_DOUBLE = 0x02, //!< Double length

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN Unknown error MF_RESP_SUCC Successly MF_RESP_FAIL Fail

3.17. Download MasterKey

Download Master Key to MPOS device

- Prototype:

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

- Parameter

name	description
masterKey	Master key(40 bits) value(BCD format). Format: key(32bits)+check value(8bits)

method	Encrypt method, Value are as follows:
	MF_ENCRYPT_KEK = 0x00, //!< KEK encrypt
	MF_ENCRYPT_MASTERKEY = 0x01, //!<
	MF_ENCRYPT_PLAINTEXT = 0x02, //!< Plain text

index	Key index, define MF_KEY_IND_0

len	KEK type, Value are as follows:
	MF_LEN_SINGLE = 0x01, //!< Haploid length
	MF_LEN_DOUBLE = 0x02, //!< Double length

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

-(void) didLoadMasterKeyResp: (MFEU_MSR_RESP)resp;

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN Unknown error MF_RESP_SUCC Successly MF_RESP_FAIL Fail

3.18. loadWorkKey

Download Work Key to MPOS device

- Prototype:

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

- Parameter

name	description
pin	PIN key(40 bits) value(BCD format). Format: key(32bits)+check value(8bits)
mac	MAC key(40 bits) value(BCD format). Format: key(32bits)+check value(8bits)
track	Track key(40 bits) value(BCD format). Format: key(32bits)+check value(8bits)
index	Key index, define MF_KEY_IND_0

If the pin/mac/track key data length is only 16 bits, it must be filled to 32 bits (direct copy), plus 8 bit kvc. If NULL, please fill any other valuable key data.

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN Unknown error MF_RESP_SUCC Successly MF_RESP_FAIL Fail

3.19. dukptLoadKey

Inject DUKPT Key to MPOS

- Prototype:

```
-(int) dukptLoadKey: (MFEU_DUKPT_ALG)alg
          withIndex: (MFEU_KEY_INDEX)index
            withKey: (NSString *)key
            withKsn: (NSString*)ksn;
```

- Parameter

name	description
alg	MFEU_DUKPT_ALG MF_DUKPT_IPEK IPEK plain text MF_DUKPT_BDK BDK plain text MF_DUKPT_IPEK_ENC_KEK IPEK cipher text(KEK decrypt) MF_DUKPT_BHK_ENC_KEK BDK cipher text(KEK decrypt) MF_DUKPT_IPEK_ENC_MAK IPEK cipher text(Master key decrypt) MF_DUKPT_BHK_ENC_MAK BDK cipher text(Master key decrypt)
index	Key index(MF_KEY_INDEX0 ~ MF_KEY_INDEX7)

key	Key index(30 bits)
ksn	KSN(20 bits)

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

```
-(void) didDukptLoadKeyResp: (MFEU_MSR_RESP)resp
                   withKvc: (NSString *)kvc;
```

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN Unknown error MF_RESP_SUCC Successly MF_RESP_FAIL Fail
kvc	Key value check

3.20. dukptGetKey

Get DUKPT Key

- Prototype:

```
-(int) dukptGetKey: (MFEU_KEY_INDEX)index
           withType: (MFEU_DUKPT_TYPE)type;
```

- Parameter

name	description
index	Key index(MF_KEY_INDEX0 ~ MF_KEY_INDEX7)
type	Key type(MFEU_DUKPT_TYPE) MF_DUKPT_DES_KEY_PIN PIN Key MF_DUKPT_DES_KEY_MAC1 MAC Request Key MF_DUKPT_DES_KEY_MAC2 MAC Response Key MF_DUKPT_DES_KEY_DATA1 DATA Request Key MF_DUKPT_DES_KEY_DATA2 DATA Response Key MF_DUKPT_DES_KEY_PEK PEK PEK

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

```
-(void) didDukptGetKeyResp: (NSString *)key
                  withKsn: (NSString *)ksn;
```

name	description
key	Specified key value
ksn	Current KSN

3.21. dukptGenKey

Generate DUKPT Key(Increase KSN add 1)

- Prototype:

```
-(int) dukptGenKey: (MFEU_KEY_INDEX)index;
```

- Parameter

name	description
index	Key index(MF_KEY_INDEX0 ~ MF_KEY_INDEX7)

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

```
-(void) didDukptGenKeyResp: (NSString *)key;
```

name	description
key	Specified key value

3.22. dukptDes

Calculate DES value of the data

- Prototype:

```
-(int) dukptDes: (MFEU_DUKPT_TYPE)type
```

```
       withOper: (MFEU_DUKPT_OPER)oper
     withMethod: (MFEU_DUKPT_METHOD)method
       withData: (NSString*)data;
```

- Parameter

name	description
type	Key type(MFEU_DUKPT_TYPE) MF_DUKPT_DES_KEY_PIN PIN Key MF_DUKPT_DES_KEY_MAC1 MAC Request Key MF_DUKPT_DES_KEY_MAC2 MAC Response Key MF_DUKPT_DES_KEY_DATA1 DATA Request Key MF_DUKPT_DES_KEY_DATA2 DATA Response Key MF_DUKPT_DES_KEY_PEK PEK PEK
oper	Encrypt or Decrypt(MFEU_DUKPT_OPER) MF_DUKPT_ENCRYPT MF_DUKPT_DECRYPT
method	Calulate algorithm(MFEU_DUKPT_METHOD) MF_DUKPT_ECB ECB MF_DUKPT_CBC CBC
data	Participating data

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

```
-(void) didDukptGetKeyResp: (NSString *)key
                   withKsn: (NSString *)ksn;
```

name	description
key	Specified key value
ksn	Current KSN

3.23. setKeyIndex

Set key download index, default index==0.

- Prototype:

```
-(NSInteger) setKeyIndex: (MFEU_KEY_INDEX)index;
```

- Parameter

name	description
index	Key download index, default 0. Values are as follows: MF_KEY_IND_0 ~ MF_KEY_IND_9

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN Unknown error MF_RESP_SUCC Successly MF_RESP_FAIL Fail

3.24. setIcKey

Set IC card public key

- Prototype:

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

- Parameter

name	description
dataArray	IC card public key, every item is ASC format.

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
index	Current item index
total	Total item index

3.25. setIcAid

Set IC card AID parameter

● Prototype:

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

● Parameter

name	description
dataArray	IC card AID parameter, every item is ASC format.

● Return:

value	description
-1	No connect
else	BT send data length

● Delegate

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

name	description
index	Current item index
total	Total item index

3.26. icDealOnline

Perform authorizations, followed by subsequent processing of IC cards online.

● Prototype:

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

- Parameter

name	description
data	Request data(ASC format)

result	Whether to succeed online or not, take as follows:
	MF_ONLINE_NO = 0x00, //!< Not Forced Online
	MF_ONLINE_YES = 0x01, //!< Forced Online

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

-(void) didIcDealOnlineResp: (MFEU_MSR_REAUTH_RESP) resp;

name	description

resp	Return code, Values are as follows:
	MF_RESP_REAUTH_UNKNOWN = 0x00, //!> Unknown error
	MF_RESP_REAUTH_ACCEPT = 0x01, //!>Transaction acceptable
	MF_RESP_REAUTH_GACAAC = 0x02, //!>Transaction reject
	MF_RESP_REAUTH_ONLINE = 0x03, //!> Online
	MF_RESP_REAUTH_REJECT = 0x04, //!> Authorized transaction rejection
	MF_RESP_REAUTH_FAIL = 0xFF, //!>Transaction failure

3.27. ICPoweron / ICPoweroff

IC card module power on / off

- Prototype:

```
-(NSInteger) ICPoweron;
-(NSInteger) ICPoweroff;
```

● Return:

value	description
-1	No connect
else	BT send data length

● Delegate

-(void) didCardPowerResp: (MFEU_MSR_RESP)resp;

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWNUnknown error MF_RESP_SUCCSuccessly MF_RESP_FAILFail

3.28. ICExchange

IC card module exchange data

● Prototype:

-(NSInteger) ICExchange: (NSString *)icdata;

● Parameter

name	description
icdata	IC APDU command

● Return:

value	description
-1	No connect
else	BT send data length

● Delegate

-(void) didCardExchangeResp: (NSString *)str;

name	description
str	APDU response data

3.29. readPosInfoEx

Get device serial number, battery status and other information

- Prototype:

```
-(NSInteger) readPosInfoEx;
```

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

```
-(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
                             dev_model: (NSString *)model;
```

name	description
ksn	KSN

status	Device key download status,Values are as follows:
	MF_DEVSTAT_DEFAULT = 0xFF, //!>default status
	MF_DEVSTAT_WKEYIN = 0x00, //!>WorkKey Downloaded
	MF_DEVSTAT_MKEYIN = 0x01, //!> MasterKey Downloaded
	MF_DEVSTAT_KEKMOD = 0x02, //!>KEK updated

battery	Battery status, Values are as follows: BATTERY_CAPACITY_UNKOWN = 0, //!> Unknown BATTERY_CAPACITY_CRITICAL, //!> Critical BATTERY_CAPACITY_LOW, //!> Low power BATTERY_CAPACITY_NORMAL, //!> Normal BATTERY_CAPACITY_HIGH, //!> Enough BATTERY_CAPACITY_FULL, //!> Full

	Note: when the value is less than or equal to MF_BATTERY_CAPACITY_LOW, indicating the battery of device is low, please prompt the user to charge
app_ver	Application version
data_ver	Data version
custom_info	Custom information
model	Device model

3.30. getRandomNum

Get a random number from device

- Prototype:

```
-(NSInteger) getRandomNum;
```

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
randNum	Generated random number(ASC format)

3.31. beep

Trigger the MPOS buzzer

- Prototype:

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

- Parameter

name	description
times	Beep times
freq	Beep frequency(Unit: hz)
duration	Beep duration time(Unit: ms)
step	Beep interval(Unit: ms)

- Return

value	description
-1	No connect
else	BT send data length

- Delegate

```
-(void) didBeepResp;
```

3.32. setDatetime

Set the POS time, make sure MPOS's time is the current time

- Prototype:

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

- Parameter

name	description
datetime	Time, YYYYMMDDHHMMSS format, length 14 bits
factoryid	Factory ID code, default 0. If you have any special requirement, please contact us

- Delegate

```
-(void) didSetDatetimeResp;
```

3.33. setFactoryCode

Set factory code of the MPOS, and set the current time

- Prototype:

```
-(NSInteger)setFactoryCode: (NSInteger)fCode;
```

● Parameter

name	description
factoryid	Factory ID code, default 0. If you have any special requirement, please contact us

● Return:

value	description
-1	No connect
else	BT send data length

● Delegate

```
-(void) didSetDatetimeResp;
```

3.34. getDatetime

Get the MPOS current time

● Prototype:

```
-(NSInteger) getDatetime;
```

● Return:

value	description
-1	No connect
else	BT send data length

● Delegate

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

name	description
datetime	Time, Format: YYYYMMDDHHMMSS(14 bits)

3.35. updatePos

Firmware updates

● Prototype:

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

- Parameter

name	description
upgradeFilename	Specify the name of the upgrade file (including the path), make sure the file is readable

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
pos	Current postion
length	Total size

```
-(void) didUpgradeFinish;
```

Upgrade Complete callback

3.36. dataWriteEx

- Prototype:

```
-(NSInteger) dataWriteEx: (NSString *)data
                   start: (NSInteger)pos;
```

- Parameter

name	description
data	Write data(Maximum size 1K bytes)
start	Data write location (0-1023)

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
resp	Return code, Values are as follows: MF_RESP_UNKNOWN //!> Unknown error MF_RESP_SUCC //!> Successly MF_RESP_FAIL //!> Fail

3.37. dataReadEx

- Prototype:

```
-(NSInteger) dataReadEx: (NSInteger) start
                 length: (NSInteger)size;
```

- Parameter

name	description
start	Data write location (0-1023)
size	Read data size

- Return:

value	description
-1	No connect
else	BT send data length

- Delegate

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

name	description
resp	Return code, Values are as follows:

	MF_RESP_UNKNOWN //!> Unknown error MF_RESP_SUCC //!> Successly MF_RESP_FAIL //!> Fail
data	Read result, If resp == MF_RESP_SUCC, the read result is returned

4. APPENDIX

4.1. Appendix A

MFMFEU_READER_SESSION, Values are as follows:

name	description
MF_SESSION_UNKNOWN	Unknown
MF_SESSION_SCAN_START	Bluetooth start scan
MF_SESSION_SCAN_STOP	Bluetooth stop scan
MF_SESSION_CONN_FAIL	Connect failure
MF_SESSION_CONN_VALID	Connect vaild equipment
MF_SESSION_CONN_INVALID	Connect invalid equipment
MF_SESSION_DISCONNECT	Disconnect
MF_SESSION_KEK_DOWNLOAD	KEK download
MF_SESSION_MKEY_DOWNLOAD	MasterKey download
MF_SESSION_WKEY_DOWNLOAD	WorkKey download
MF_SESSION_SELECT_PIN	Key select
MF_SESSION_INPUT_PIN	Input PIN
MF_SESSION_CALC_MAC	MAC Calculate
MF_SESSION_SET_ICKEY	Set IC public key
MF_SESSION_SET_AID	Set AID parameter
MF_SESSION_SET_DATA	PBOC EMV trade
MF_SESSION_START_EMV	Start IC trade

MF_SESSION_IC_REAUTH	IC reauthorize
MF_SESSION_END_EMV	End IC trade
MF_SESSION_GET_DEVINFO_EX	Read POS infomation
MF_SESSION_GET_RANDNUM	Get random number
MF_SESSION_BEEP	Beep
MF_SESSION_SET_DATETIME	Set the MPOS date&time
MF_SESSION_GET_DATETIME	Get the MPOS date&time
MF_SESSION_UPGRADE	Fireware updates
MF_SESSION_DATA_WRITE	Data is written to MPOS
MF_SESSION_DATA_READ	Read data from MPOS

4.2. Appendix B

MFEU_READER_RESULT, Values are as follows:

Return failure:	
MF_RET_FAIL	Unknown failure
MF_RET_FAIL_STX	Field STX parsing error
MF_RET_FAIL_LEN	Field LEN parsing error
MF_RET_FAIL_PATH	Field PATH parsing error
MF_RET_FAIL_TYPE	Field TYPE parsing error
MF_RET_FAIL_ID	Field ID dissimilarity
MF_RET_FAIL_ETX	Field ETX parsing error
MF_RET_FAIL_LRC	Field LRC parsing error
MF_RET_FAIL_CMD	Command not support
MF_RET_FAIL_PARAM	Parameter failure
MF_RET_FAIL_LENGTH	Data length error
MF_RET_FAIL_FORMAT	Frame format error
MF_RET_FAIL_GETLRC	LRC error
MF_RET_FAIL_OTHER	Other error
MF_RET_FAIL_TIMEOUT	Timeout error

MF_RET_FAIL_STATUS	Status error
Normal return:	
MF_RET_KEK_DOWNLOAD	KEK download
MF_RET_MKEY_DOWNLOAD	MasterKey download
MF_RET_WKEY_DOWNLOAD	WorkKey download
MF_RET_SELECT_KEY	Key select
MF_RET_INPUT_PIN	Input PIN
MF_RET_CALC_MAC	MAC Calculate
MF_RET_SET_ICKEY	Set IC public key
MF_RET_SET_AID	Set AID parameter
MF_RET_SET_DATA	PBOC EMV trade
MF_RET_START_EMV	Start IC trade
MF_RET_IC_REAUTH	IC reauthorize
MF_RET_END_EMV	Stop IC trade
MF_RET_GET_DEVINFO_EX	Get device infomation
MF_RET_GET_RANDNUM	Get a random number from MPOS
MF_RET_BEEP	Bepp
MF_RET_SET_DATETIME	Set the MPOS date&time
MF_RET_GET_DATETIME	Get the MPOS date&time
MF_RET_UPGRADE	Fireware updates
MF_RET_UPGRADE_FINISH	Fireware update finish.
MF_RET_DATA_WRITE	Data is written to MPOS
MF_RET_DATA_READ	Read data from MPOS
MF_RET_TIMEOUT	Recvice timeout
MF_RET_USER_CANCEL	User cancel

4.3. Appendix C

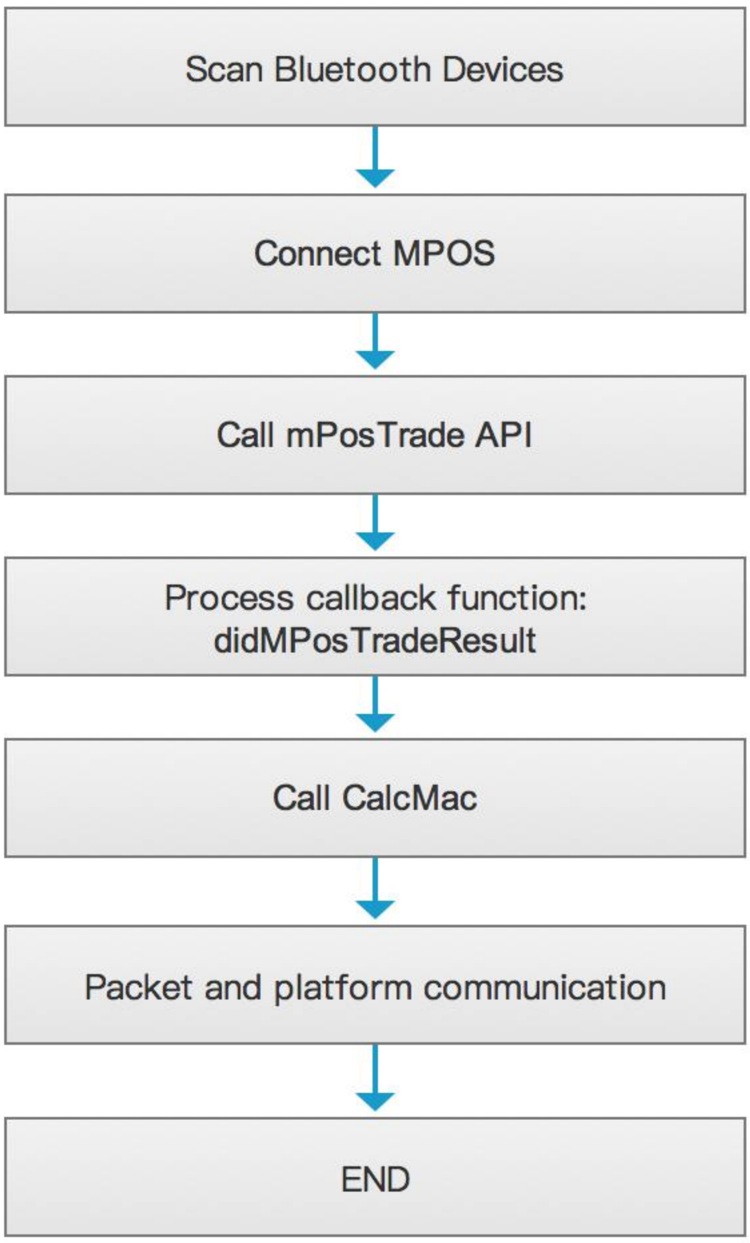
MFEU_TRADE_TYPE, Values are as follows:

name	description

MF_FUNC_BALANCE	Balance query
MF_FUNC_SALE	Sale trade
MF_FUNC_PREAUTH	Pre-authorization
MF_FUNC_AUTHSALE	Request pre-authorization finish
MF_FUNC_AUTHSALEOFF	Notify pre-authorization finish
MF_FUNC_AUTHSETTLE	Settle pre-authorization
MF_FUNC_ADDTO_PREAUTH	Append pre-authorization
MF_FUNC_REFUND	Refund
MF_FUNC_VOID_SALE	Undo sale trade
MF_FUNC_VOID_AUTHSALE	Undo pre-authorization finish
MF_FUNC_VOID_AUTHSETTLE	Undo settle pre-authorization
MF_FUNC_VOID_PREAUTH	Undo pre-authorization
MF_FUNC_VOID_REFUND	Undo refund
MF_FUNC_OFFLINE	Offline Settlement
MF_FUNC_ADJUST	Settlement adjustment
MF_FUNC_SETTLE	Settlement

4.4. Appendix D

`MPOS General operating procedure:`


Scan Bluetooth Devices
Connect MPOS
Call mPosTrade API
Process callback function: didMPosTradeResult
Call CalcMac
Packet and platform communication
END