

V4.1

银联商务有限公司技术开发中心MIS室

版本控制信息

| 版本 | 日期 | 拟稿和修改人 | 说明 |
| --- | --- | --- | --- |
| 1.0.0 | 2014-08-26 | 王玉珏 | 创建 |
| 2.0.0 | 2014-10-30 | 王玉珏 | 修改交易编号  增加全民付48域组织说明 |
| 3.0.0 | 2014-12-03 | 王玉珏 | 增加对传统业务IC卡的支持 |
| 3.1.0 | 2015-01-16 | 王玉珏 | 增加对增值业务手机充值、信用卡还款IC卡的支持 |
| 3.2.0 | 2015-03-05 | 王玉珏 | 增加对外挂非接模块R50、R30的支持 |
| 3.3.0 | 2015-04-25 | 王玉珏 | 增加关闭读卡器函数、吞卡函数 |
| 3.3.0 | 2015-04-29 | 王玉珏 | 增加传统交易必填要素表 |
| 3.3.1 | 2016-01-19 | 王玉珏 | 增加了一体机显示接口3.12和3.13 |
| 3.4 | 2016-03-08 | 王玉珏 | 增加了检测卡位置的9号状态（在内置非接感应器上） |
| 4.0 | 2016-04-24 | 王玉珏 | 1 stddevice库支持内置（插在读卡器尾部）和外置（插在上位机）两种非接模块  2 废弃PbocProcessForCheck函数，只在读卡时进行一次卡交互，支持applepay  3 调整外部接口，增加设置入参接口UMS\_SetReq，UMS\_TransCard的第一个参数作废  4 新增外置非接模块标准接口  5 增加了单独使用外挂非接的工作模式  6 增加了对非接触式预授权的支持 |
| 4.1 | 2016-07-24 | 王玉珏 | 增加了小额免密免签功能 |

|  |
| --- |
| 本文档中的所有内容为中国银联商务有限公司专属所有。未经中国银联商务有限公司的明确书面许可，任何组织或个人不得以任何目的、任何形式及任何手段复制或传播本文档部分或全部内容。 |

## 文档说明

本文档向收银软件开发商描述了银联商务windows版本自助收银系统的接口和调用流程。此版本采用插卡器实现IC、磁条卡联机，采用外挂式电子现金模块（R50、R30）实现脱机闪付电子现金交易（目前只在传统应用内支持闪付类的消费和查余交易），采用内置、外置非接模块实现联机闪付（IC卡非接触式联机）。

## 函数调用流程

1. 调用初始化函数UMS\_Init(appType)，初始化应用类型根据所要进行的业务类型取值。
2. 调用设置入参函数UMS\_SetReq对请求参数进行设置。
3. 调用进卡函数UMS\_EnterCard()，系统根据配置设置读卡器为可插、挥卡状态。此函数同时尝试开启插卡器和外挂非接感应器，并返回设备开启情况。
4. 提示用户插、挥卡。
5. 调用检测卡函数UMS\_CheckCard (byte \*state\_out)，判断读卡器内或者非接感应器上是否有卡。如果有卡则关闭另一个设备。并以返回值返回采用了哪个设备。（此过程会阻滞约2秒钟时间）建议循环调用,直至检测卡超时时间到来。
6. 当检测到卡片插入或在非接模块感应区内之后，调用读卡函数UMS\_ReadCard (char \*cpData)，此时会返回给ERP用户卡号。此函数会产生阻滞。
7. 对于联机交易，读卡成功后，调用打开密码键盘函数UMS\_StartPin()。
8. 提示用户输入密码。
9. 调用取键值函数UMS\_GetOnePass (unsigned char \*key\_out)，根据具体的键值绘制密码输入画面。
10. 密码输入完毕后，调用取用户密码密文函数UMS\_GetPin ()。
11. 调用交易函数UMS\_TransCard (char \*strReq, char \*strResp)进行交易。如果该交易函数是无磁无密函数，则跳过3~10步。对于电子现金交易，跳过7~10步。
12. 调用关闭读卡器函数UMS\_EjectCard ()，弹出卡片（关闭非接模块），并提示用户拔卡、取走卡片。（若为磁条卡，也可以在第5步之后弹出卡片，防止持卡人忘记取卡。若为电子现金交易，请让持卡人在交易过程中保持卡片在非接感应器上，不可以中途移走卡片。）
13. 继续循环调用UMS\_CheckCard (byte \*state\_out)，判断持卡人是否真正拿走了卡片。
14. 如持卡人在时限内未拿走卡片，调用吞卡函数UMS\_CardSwallow()，将卡片没收。
15. 调用关闭读卡器函数UMS\_CardClose()。

## 函数说明

所有函数都封装在umsapi.dll里。使用前请先加载此动态库。也可以采用头文件加lib库的形式静态加载。发布文件请仅仅放置于c:\umsips文件夹目录下。

### 初始化

int UMS\_Init(int appType);

功能描述：初始化密码键盘、检查冲正、检查签到结算、下公钥、下公参、检查设备能力、脱机交易上送、TC上送。

参数：apptype:应用类型 1:传统银行卡应用 2:全民付应用 3:预付卡 4:pos通。不同应用支持的交易类型，见本章3.2节。

返回：

0：成功

其他情况：

|  |  |
| --- | --- |
| -101：appType非法 | -102：加载主配置main.ini失败 |
| -103：初始化密码键盘失败 | -104：银联卡模块初始化失败 |
| -105：增值模块初始化失败 | -106：预付卡模块初始化失败 |
| -107：暂不支持的appType | -108：向EMV内核下载IC卡参数失败 |

### 设置入参

int \_\_stdcall UMS\_SetReq(char\* cpInReq)

参数：

char\* strReq：传入参数，格式如下：

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名称** | **字段长度（BYTE）** | | **字段含义** |
| **strCounterId** | 8 | 款台号 | |
| **strOperId** | 8 | 操作员号 | |
| **strTransType** | 2 | 交易编号  **传统类交易**：00:消费 01：撤销 02:退货03:查余  04:结算\* 05:签到\* 08：预授权 09：预授权撤销  10：预授权完成 15：非接参数下载\* 16：新增Bin表下载\*  17：非接免密黑名单下载\*  **全民付交易**：01:手机充值 02:信用卡还款手续费查询\*  03:信用卡还款缴费 04：卡卡转账  05：公共事业查询\* 06：公共事业缴费  **预付卡**：00 消费  **POS通**：01：消费\* 05：消费C扫B\* 06：C扫B结果查询\* | |
| **strAmount** | 12 | 金额 | |
| **strOldTrace** | 6 | 原流水号 | |
| **strOldDate** | 8 | 原交易日期YYYYMMDD | |
| **strOldRef** | 12 | 原系统参考号 | |
| **strOldAuth** | 6 | 原授权号 | |
| **strOldBatch** | 6 | 原批次号 | |
| **strMemo** | 1024 | 增值交易48域附加信息（采用第4章所述格式传入） | |
| **strLrc** | 3 | 3个校验字符 | |

请求中带\*的为无磁无密交易。

### 进卡

int \_\_stdcall UMS\_EnterCard()

功能描述：允许读卡器进卡，调用成功后，提示用户插卡、放置卡片。

参数：无。

返回：

0:读卡设备启动成功，电子现金模块无法打开；

1:电子现金模块启动成功，读卡设备无法打开；

2:电子现金模块、读卡设备均启动成功；

-201~-210:失败。

### 检测卡

int \_\_stdcall UMS\_CheckCard(byte \*state\_out)

功能描述：返回读卡器内卡片位置，可循环调用，建议1秒为周期。

参数：state\_out:卡片信息码

|  |  |  |
| --- | --- | --- |
| 位置码 | 含义 | 说明 |
| 0x34 | 卡在插卡器卡口位置 | 可提示用户插、拔卡 |
| 0x35 | 未检测到卡 | 可提示用户插卡、挥卡 |
| 0x37 | 插卡器内部有卡 | 可以读卡 |
| 0x38 | 卡在电子现金感应器上 | 可以读卡 |
| 0x39 | 卡在非接联机感应器上 | 可以读卡 |

返回：

0:成功;

-211~-220:失败。

### 读卡

int \_\_stdcall UMS\_ReadCard(char\* cpData);

功能描述：读卡号、选择扣款账户。传统应用返回屏蔽（除前六后四位之外，其他用\*代替）后的卡号，闪付类交易返回全卡号，全民付应用返回全卡号。

出参: cpData 卡号存储指针

返回:

0:磁条卡读取成功；

1:芯片卡读取成功；

2:非接卡读取成功，不免密；

3:非接卡读取成功，免密；

-221~-230:失败;

### 弹卡

int \_\_stdcall UMS\_EjectCard()

功能描述：把卡弹出到读卡器口。IC卡交易，请在交易结束后，再弹卡。

参数：无

返回：

0:成功；

-231~-240:失败

### 关闭读卡器

int \_\_stdcall UMS\_CardClose()

功能描述：关闭设备。当内部有卡时，卡片将从卡口弹出。

参数：无

返回：

0 成功 其他 失败

### 吞卡

int \_\_stdcall UMS\_CardSwallow()

功能描述：当卡片停留在卡口或者在内部时，从读卡器后部将卡片弹出。

参数：无

返回：

0 成功 其他 失败

### 开启密码键盘

int \_\_stdcall UMS\_StartPin()

若为电子现金账户扣款，无需次步骤。

返回：

0 成功 其他 失败

### 获得键值

int \_\_stdcall UMS\_GetOnePass(byte\* key\_out)

若为电子现金账户扣款，无需次步骤。

|  |  |
| --- | --- |
| 键值 | 含义 |
| 0x02 | 输入超时 |
| 0x08 | 退格 |
| 0x2A | 一个按键 |
| 0x1B | 用户取消交易 |
| 0x0D | 输入确认 |
| 0xFF | 用户暂时没有输入 |

返回：

0 成功 其他 失败

### 获取pin密文

int \_\_stdcall UMS\_GetPin()

若为闪付类交易，无需次步骤。

返回：

0 成功 其他 失败

### 交易函数

int \_\_stdcall UMS\_TransCard(char\* strReq,char\* strResp)

功能描述：自助终端交易函数。

char\* strResp：返回参数，格式如下:

|  |  |  |
| --- | --- | --- |
| **字段名称** | **字段长度（BYTE）** | **字段含义** |
| **strRespCode** | 2 | 应答码 |
| **strRespInfo** | 40 | 应答码说明信息（汉字） |
| **strCardNo** | 20 | 交易卡号 |
| **strAmount** | 12 | 金额 |
| **strTrace** | 6 | 终端流水号（凭证号） |
| **strBatch** | 6 | 批次号 |
| **strTransDate** | 4 | 交易日期MMDD |
| **strTransTime** | 6 | 交易时间hhmmss |
| **strRef** | 12 | 系统参考号（中心流水号） |
| **strAuth** | 6 | 授权号 |
| **strMId** | 15 | 商户号 |
| **strTId** | 8 | 终端号 |
| **strMemo** | 1024 | 48域附加信息（采用第4章所述格式传出） |
| **strLrc** | 3 | 3个校验字符 |

返回：该函数只返回0

注意：对于无磁无密的交易，初始化成功后，直接调用交易函数

应答码的具体中文说明，在程序包的rsp.ini中

#### 传统交易必填要素表

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 交易名称 | 交易号 | 款台号 | 操作员号 | 金额 | 原流水号 | 原交易日期 | 原系统参考号 | 原授权号 | 原批次号 | Memo | LRC |
| 00消费 | √ | ○ | ○ | √ | □ | □ | □ | □ | □ | □ | ○ |
| 01撤销 | √ | ○ | ○ | √ | √ | □ | □ | □ | □ | □ | ○ |
| 02退货 | √ | ○ | ○ | √ | □ | √ | √ | □ | □ | □ | ○ |
| 03查余 | √ | ○ | ○ | □ | □ | □ | □ | □ | □ | □ | ○ |
| 04结算 | √ | ○ | √ | □ | □ | □ | □ | □ | □ | □ | ○ |
| 05签到 | √ | □ | □ | □ | □ | □ | □ | □ | □ | □ | ○ |
| 08预授权 | √ | ○ | ○ | √ | □ | □ | □ | □ | □ | □ | ○ |
| 09预授权撤销 | √ | ○ | ○ | √ | □ | √ | □ | √ | □ | □ | ○ |
| 10预授权完成 | √ | ○ | ○ | √ | □ | √ | □ | √ | □ | □ | ○ |

○：可随意填的字段，该字段会被打印或被校验

□：采用空格填充，此字段在该交易中无意义

√：必须填入实际值的字段

#### 对于信用卡还款、卡卡转账交易的流程说明

首先，初始化。然后获得信用卡或者转入卡的卡号。可以通过3.3~3.7函数来实现。然后再次读取转出卡。启动密码键盘，输入密码，调用交易函数。最后，弹卡，并判断卡片是否被取走。如未取走，吞卡。

### 清除屏幕提示

int \_\_stdcall UMS\_ClearScreen()

功能描述：对于一体机，在显示提示信息内容之前，清除屏幕原先的内容

参数：

无

返回：

0:成功；

其他:失败

### 显示提示信息

int \_\_stdcall UMS\_ShowData(int iInLow, char\* cpInStr)

功能描述：对于一体机，按照指定的位置（行数），显示提示信息内容

参数：

int iInLow：行数（位置）

char\* cpInStr：提示信息内容

返回：

0:成功；

其他:失败

## 全民付交易48域组织说明

### 数据格式

ERP传入48域数据在入参字段strMemo中，采用通常意义上的TLV（tag-length-value）格式，即每个子域由tag标签(T)，子域取值的长度(L)和子域取值(V)构成。

tag标签的属性为bit，由16进制表示，占1～2个字节长度。例如，“9F33”为一个占用两个字节的tag标签。而“95”为一个占用一个字节的tag标签。若tag标签的第一个字节（注：字节排序方向为从左往右数，第一个字节即为最左边的字节。bit排序规则同理。）的后四个bit为“1111”，则说明该tag占两个字节，例如“9F33”；否则占一个字节，例如“95”。

子域长度（即L本身）的属性也为bit，占1～3个字节长度。具体编码规则如下：

a) 当L字段最左边字节的最左bit位（即bit8）为0，表示该L字段占一个字节，它的后续7个bit位（即bit7～bit1）表示子域取值的长度，采用二进制数表示子域取值长度的十进制数。例如，某个域取值占3个字节，那么其子域取值长度表示为“00000011”。所以，若子域取值的长度在1～127字节之间，那么该L字段本身仅占一个字节。

b) 当L字段最左边字节的最左bit位（即bit8）为1，表示该L字段不止占一个字节，那么它到底占几个字节由该最左字节的后续7个bit位（即bit7～bit1）的十进制取值表示。例如，若最左字节为10000010，表示L字段除该字节外，后面还有两个字节。其后续字节的十进制取值表示子域取值的长度。例如，若L字段为“1000 0001 1111 1111”，表示该子域取值占255个字节。所以，若子域取值的长度在127～255字节之间，那么该L字段本身需占两个字节。

### 数据限制

本程序中使用的TLV数据除符合上述TLV规则以外，还符合下面的条件约束：

1：所有tag都是2个字节的。

2：L的长度不超过3字节。即一个子域最大数据长度为FFFF（65535）字节。

3：TLV数据段起始指针为strMemo+4。前面的四个字节为从strMemo+4开始的TLV数据总长度，采用AscII码标示（左对齐，右补空格）。

### 相关函数列表

为了方便厂商打TLV数据，在umsapi.dll中提供了一些工具函数供打TLV数据使用。

#### 初始化TLV

int \_\_stdcall Tlv\_Init( byte\* bpInBuf,

int iInBufLen)

功能说明:

初始化TLV指针和总长度。在每一次打TLV数据前，都需要调用此函数，初始化。

只有初始化成功后，方可调用Tlv\_AddTag。否则会引起异常。

参数:

byte\* bpInBuf：外部分配的存储TLV数据的空间首地址

int iInBufLen：外部分配的存储TLV数据空间的大小

返回:

0：成功 -1：外部分配空间不可以小于4字节

#### 增加一个tag

int \_\_stdcall Tlv\_AddTag( char\* cpInTagNameAsc,

char\* cpInData,

int iInDataLen)

功能描述:

打一个tag

参数:

char\* cpInTagNameAsc： AscII码标示的tag名

char\* cpInData：数据首地址

int iInDataLen：数据长度

例子:

要在sbuf中打tag为"9F36"的数据"123456"

byte sbuf[512]

Tlv\_Init(sbuf,512)

Tlv\_AddTag("9F36","123456",6);

返回:

大于0 成功,当前TLV的总长度

-1 外部越界 -2 内部越界 -3 tag重复 -4 总长度超过可表示范围

#### 解出指定tag的数据

int \_\_stdcall Tlv\_GetTag(char\* cpInTlvData,

char\* cpInTagAsc,char\* cpOutData);

功能描述:

解出指定tag的数据

参数:

char\* cpInTlvData 长度+TLV数据（前4个字节为AscII码表示的后面TLV数据的总长度）

char\* cpInTagAsc 采用AscII码表示的tag

char\* cpOutData 取得的数据

返回:

大于0 成功,所获得的数据长度

-1 未找到所要tag

例子:

iRet = Tlv\_AddTag("9F36","123456",6);

iRet = Tlv\_AddTag("9F37","11",2);

//此时数据为"\x31\x34\x20\x20\x9F\x36\x06\x31\x32\x33\x34\x35\x36\x9F\x37\x02\x31\x31"

Tlv\_GetTag((char\*)sbuf,"9F37",sbuf2);//此时sbuf2为"11"

Tlv\_GetTag((char\*)sbuf,"9F36",sbuf2);//此时sbuf2为"123456"

#### 将TLV数据转换成可见AscII字符

int \_\_stdcall Tlv\_GetAscData( const char\* cpInDataTLV,

char\* cpOutDataAsc)

参数:

const char\* cpInDataTLV:传入TLV数据指针

char\* cpOutDataAsc:返回AscII字符数据指针

返回:

该函数只返回0

例子:

char BufTlv[MEMOLEN];

char BufAsc[MEMOLEN\*2+1];

iRet= Tlv\_Init(BufTlv,MEMOLEN);

iRet = Tlv\_AddTag("9F36","123",3);

Tlv\_GetAscData(BufTlv,BufAsc); //此时BufAsc里面为"362020209F3603313233"

注意:

此函数执行前无需外部清cpOutDataAsc

### 增值交易必须传入的tag

|  |  |  |
| --- | --- | --- |
| **交易类型** | **必须传入的tag** | **需要获得的返回数据** |
| **手机充值** | 手机号码（0F12）、运营商类别（0F13） |  |
| **信用卡还款查询** | 信用卡卡号（0F14）、手机号码（0F12） | 手续费（0F15）、提示信息（0F17） |
| **信用卡还款缴费** | 信用卡卡号（0F14）、手机号码（0F12） |  |
| **卡卡转账** | 转入账户号码（0F16） |  |
| **公共事业查询** | 地区码（0F1A）、出账机构代码（0F1B）、条码号（或者账单号、户号）（0F1C）、出账机构特殊需要的域（0F19）、手机号码（可选）（0F12） | 总金额（0F0D）、账期（0F18）、出账机构特殊需要的域（0F19） |
| **公共事业缴费** | 无 | 订单号（0F1E） |
| **广西交警罚没款账单查询** | 处罚书号（0F00）、处罚书类型（0F01）、交警校验码（0F02） | 操作结果信息（0F03）、当事人（0F04）、执行单位编码（0F05）、执行单位名称（0F06）、缴费状态（0F07）、处罚日期（0F08）、车牌号码（0F09）、处理机关名称（0F0A）、罚款金额（0F0B）、滞纳金（0F0C）、总金额（0F0D）、查询新一代流水号（0F0E） |
| **广西交警罚没款缴费** | 查询新一代流水号（0F0E） | 操作结果信息（0F03）、业务流水号（0F10）、校验码（0F11）、销账新一代流水号（0F0E） |
| **POS通消费B扫C** | **串码（0F1F）** |  |
| **POS通消费C扫B** |  | 支付二维码（0F20）、商户账单号 （0F21）、账单时间（0F22） |
| **C扫B结果查询** | 商户账单号 （0F21）、账单时间（0F22） | 交易状态（0F23） |

### 要素Tag列表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **Tag** | **最大长度** | **说明** | **来源** |
| **处罚书号** | 0F00 | 40 |  | 广西财政非税（交警罚没）终端接口V1.4.1.doc |
| **处罚书类型** | 0F01 | 1 | 1=简易处罚决定书，2=行政处罚决定书 | 同上 |
| **交警校验码** | 0F02 | 1 | 校验码为一位字符，如果决定书上还没有的，就置为字符”0”。 | 同上 |
| **操作结果信息** | 0F03 | 30 | 保留使用 | 同上 |
| **当事人** | 0F04 | 10 |  | 同上 |
| **执行单位编码** | 0F05 | 10 |  | 同上 |
| **执行单位名称** | 0F06 | 30 |  | 同上 |
| **缴费状态** | 0F07 | 1 | 1(已缴)0(未缴) | 同上 |
| **处罚日期** | 0F08 | 20 |  | 同上 |
| **车牌号码** | 0F09 | 20 |  | 同上 |
| **处理机关名称** | 0F0A | 20 |  | 同上 |
| **罚款金额** | 0F0B | 20 |  | 同上 |
| **滞纳金** | 0F0C | 20 |  | 同上 |
| **总金额** | 0F0D | 20 |  | 同上 |
| **新一代流水号** | 0F0E | 12 | 银商新一代流水号 | 同上 |
| **透传标志** | 0F0F | 1 | 0代表透传，无需校验消费信息（非新一代平台），1表示验证缴费（新一代） | 同上 |
| **业务流水号** | 0F10 | 30 |  | 同上 |
| **校验码** | 0F11 | 30 | 缴费确认校验码，来自第三方。仅用于显示。 | 同上 |
| **手机号码** | 0F12 | 20 | 左对齐，右补空格 |  |
| **运营商类别** | 0F13 | 2位固定长度 | 01—移动；  02—联通；  03—电信； |  |
| **信用卡卡号** | 0F14 | 20 | 左对齐，右补空格 |  |
| **手续费** | 0F15 | 12 | 右对齐，左补空格，分为单位 |  |
| **转入账户卡号** | 0F16 | 20 | 左对齐，右补空格 |  |
| **终端提示信息** | 0F17 | 100 | 左对齐，右补空格 |  |
| **账期** | 0F18 | 6 |  | 便民终端接口-公用事业缴费部分V1.7.doc |
| **出账机构特殊需要的域** | 0F19 | 100 | 缴费的时候原样上送 | 同上 |
| **地区码** | 0F1A | 4 | 所属地区的电话区号，不满四位的左补0 |  |
| **出账机构代码** | 0F1B | 15 | 参考《CP便民缴费渠道及缴费项目明细表.xls》 |  |
| **条码号（或者账单号、户号）** | 0F1C | 50 | 请求上送 |  |
| **支付日期** | 0F1D | 8 | YYYYMMDD |  |
| **订单号** | 0F1E | 20 |  |  |
| **串码** | 0F1F | 32 |  |  |
| **支付二维码** | 0F20 | 200 | 内容为URL，由终端转换成二维码显示 |  |
| **商户账单号** | 0F21 | 64 | YYMMDDHHMIssSSS+11位随机数 |  |
| **账单时间** | 0F22 | 10 | YYYY-MM-DD |  |
| **交易状态** | 0F23 | 20 |  |  |