ICS 35.100
L 79

# T/ITS

# 中国智能交通产业联盟标准

T/ITS 0033—2015

# 合作式智能运输系统 专用短程通信无线互联车载单元设备应用

**Cooperative ITS - Dedicated Short Range Communications (DSRC) - Wireless Interconnected On Board Unit Device and Application**

2015-11-23 发布　　2016-01-01 实施

中国智能交通产业联盟 发布

# 目　　次

# 前　言

本标准按照 GB/T 1.1—2009 给出的规则起草。

本标准由中国智能交通产业联盟提出并归口。

本标准于 2015 年 11 月首次发布，本次为首次发布。

本标准起草单位：深圳成谷科技有限公司、交通运输部公路科学研究院、北京市交通信息中心、北京万集科技股份有限公司、天津中兴智联科技有限公司、北京握奇智能科技有限公司、上海复旦微电子集团股份有限公司、北京聚利科技股份有限公司。

本标准主要起草人：于海、宋向辉、刘建峰、赵昱阳、马国松、段起志、段永刚、王小军、李东元。

# 引　言

根据中国智能交通系统发展要求，以及目前电子不停车收费车载单元发展情况，标准编制组在广泛调查研究，认真总结建设实践经验，参考国外先进标准，并广泛征求意见的基础上，制定本标准。

为规范各类DSRC车载单元通过蓝牙与手机连接的接口和设备要求，实现互联网智能交通应用。

本标准的主要内容是：无线互联车载单元的相关术语及定义，无线互联车载单元设备的系统架构、技术要求、应用接口和应用安全。

本标准为合作式智能运输系统专用短程通信无线互联车载单元系列标准中的设备应用部分，后续将围绕该标准制定相关业务和应用规范。

# 合作式智能运输系统 专用短程通信 无线互联车载单元设备应用

## 1 范围

本部分规定了采用合作式智能交通专用短程通信（DSRC）技术、支持蓝牙的车载单元的性能参数、数据接口、工作环境等设备要求。目前电子不停车收费的专用短程通信车载单元也是本规范所定义的车载单元的一种，应符合本标准规定。

本规范规定车载单元与移动终端之间通信接口的物理形式及通信数据帧格式，以及采用该技术的车载单元应达到的性能参数、安全机制、工作环境等设备要求。

本规范规定车载单元与移动终端通过蓝牙互通时，车载单元应提供的主流操作系统应用编程接口。应支持的主流操作操作系统为安卓和苹果 IOS 系统。

## 2 规范性引用文件

JR/T 0025-2005 中国金融集成电路（IC）卡规范

GB/T 20851.1-2007 电子收费 专用短程通信 第1部分：物理层

GB/T 20851.2-2007 电子收费 专用短程通信 第2部分：数据链路层

GB/T 20851.3-2007 电子收费 专用短程通信 第3部分：应用层

GB/T 20851.4-2007 电子收费 专用短程通信 第4部分：设备应用

GB/T 20851.5-2007 电子收费 专用短程通信 第5部分：物理层主要参数测试方法

ISO/IEC 14443 识别卡 非接触式集成电路卡

交通运输部2011年第13号公告 收费公路联网电子不停车收费技术要求

蓝牙核心规范版本4.0（Bluetooth Core Specification Version 4.0）

## 3 术语及定义、缩略语

### 3.1 术语及定义

GB/T 20851.1-2007、GB/T 20851.2-2007、GB/T 20851.3-2007、GB/T 20851.4-2007、《收费公路联网电子不停车收费技术要求》中确立的以及下列术语和定义适用于本文件。

### 3.2 缩略语

下列缩略语适用于本文件。

UUID：通用唯一识别码（Universally Unique Identifier）

DSRC：专用短程通信（Dedicated Short Range Communication）

ESAM：嵌入式安全访问模块（Embedded Secure Access Module）

ETC：电子收费（Electronic Toll Collection）

ICC：集成电路卡（Integrated Circuit Card）

ITS：智能运输系统（Intelligent Transportation Systems）

MMI：人机界面（Man-Machine Interface）

OBU：车载单元（On-Board Unit)

RF：射频（Radio Frequency）

BLE：蓝牙低功耗（Bluetooth Low Energy）

SE：非对称安全模块（Security Embedded Secure Access Module）

## 4 总体要求和设备分类

### 4.1 总体要求

车载单元应符合 GB/T 20851.1-2007、GB/T 20851.2-2007、GB/T 20851.3-2007 系列标准规定的 A 类上下行链路（ASK 调制方式，FM0 编码）的各项要求。

车载单元应支持通过蓝牙和移动终端如手机、PDA、车载导航仪、笔记本电脑、平板电脑等个人或车载终端互通，提供车辆身份认证、路侧单元消息推送等功能，从而和移动终端配合实现出行信息服务、一键救援、交通路况采集、通行卡充值消费等扩展应用。

### 4.2 车载单元物理接口

车载单元应通过内嵌蓝牙功能模块来具备蓝牙功能，实现蓝牙无线通信接口，实现与移动终端或车载终端的蓝牙无线通信功能，实现各类增值业务应用。

车载单元应支持 5.8G 电子收费 DSRC 接口，用于实现电子不停车收费功能。

车载单元应支持优先处理 5.8G 电子收费 DSRC 接口数据。

### 4.3 车载单元与移动终端物理层协议

车载单元的蓝牙物理层基本参数应符合蓝牙相关规范，蓝牙参数指标应满足下表，如表 1。

表 1　蓝牙物理参数要求

| 指标名称 | | 指标 |
|---|---|---|
| 通信特性 | 通信距离 | 符合蓝牙 Class 2 |
| | 接收灵敏度 | <-85dbm |
| | 发射功率 | <2.5dbm |
| | 通信频率 | 2.4G |
| 连接方式 | 接入方式 | 手机发起 |
| | 接入时间 | <4s |
| 接口定义 | UUID | 128bit /16bit |



## 5　可无线扩展的车载单元系统构成

车载单元应支持 GB/T 20851 系列标准中电子收费车载单元的防拆特性、内置安全接入模块等功能。

车载单元应支持通过蓝牙无线通信实现与 5.8G、13.56M 等通信接口之间的数据无线转发功能，并应满足以下典型应用：

- 空中圈存：手机 APP 连接后台加密机，对车载单元做机具认证，并以车载单元为机具完成充值写卡的过程。
- 一键救援：手机 APP 一键求援，通过车载单元内置的安全访问模块，安全、保密、自动的提供车型、车牌、颜色等信息，并可通过通行卡支付有偿服务。
- 5.8G 信息推送：路侧天线通过 5.8G DSRC 向车载单元下发数据包，车载单元将该数据包通过蓝牙无线通信转发到手机，提示给车主。

支持扩展应用的车载单元系统构成如图 1 所示。

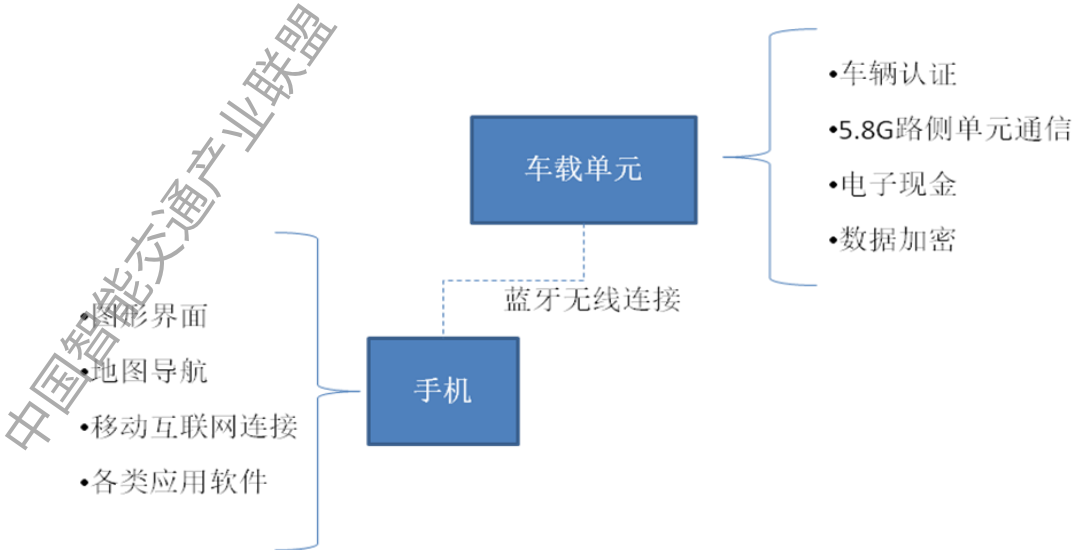


图 1　支持扩展应用的车载单元系统构成

如图 1，车载单元应能提供车辆认证、5.8G 路侧单元通信、电子现金、数据加密等基础功能。增值服务与应用 APP 由移动终端提供，移动终端包括个人手机、平板电脑、笔记本电脑、车载智能导航设备等。

用于 5.8G 信息推送的 DSRC 消息接口，应符合附录 5 定义。

## 6 移动智能终端蓝牙 API 接口

提供无线互联车载单元的实现方应提供基于蓝牙无线通信功能的 SDK 供第三方 APP 开发者使用。安卓（Android）平台和苹果（iOS）的 API 函数应符合附录 1 和附录 2 定义。

## 7 安全机制

在圈存应用中，圈存平台应对卡号做认证，确认是充值支付指定的卡号，才能启动写卡流程。

圈存平台应能认证充值用的车载单元合法性，认证功能应通过内置安全访问模块或其他相同安全级别的安全芯片实现。

圈存平台对车载单元的认证可通过读取车载单元内 ESAM 加密的车辆信息文件实现。圈存平台读取到车辆信息密文后，应通过加密机或 PSAM 模块解密该车辆信息文件，并应根据解密后的数据是否正确判断车载单元的合法性。

圈存平台对车载单元的认证可采用非对称证书模式。当采用非对称证书作为安全方案时，车载单元应内置非对称安全芯片(SE)，从而实现基于非对称密钥体系的签名、验签、加密、解密功能，进一步提高圈存等金融属性业务的安全性。

在采用非对称证书作为安全方案时，圈存平台应与 OBU 之间建立信任连接，建立连接的握手流程应遵循附录 3 中规定。在连接建立后，圈存平台与 OBU 之间的操作 IC 卡的 APDU 指令消息应采用附录 3 中的会话密钥加密。

在采用非对称证书作为安全方案时，圈存平台与 OBU 之间建立连接，操作 IC 卡的消息，应使用通道 A7，且发送消息的格式应符合附录 4 中规定。

在采用非对称证书作为安全方案时，OBU 应支持使用 3DES 或 SM4 算法对圈存中涉及 IC 卡操作的消息加解密。

# 附 录 A
（规范性附录）
Android 平台蓝牙 API 接口

## A.1 OBU初始化接口

说明：

蓝牙 OBU 初始化 API，功能包括对开发用户 ID，开发授权码进行校验；扫描对应的 OBU，并与该 OBU 建立蓝牙连接，设置接收蓝牙 OBU 命令的回调类和方法等。

类名：

BluetoothObuHandler

方法名：

public int initializeObu(Context context, String clientID, String clientKey,
BluetoothObuCallback callback)

输入参数：

Context context：调用蓝牙 OBU 初始化 API 的实例上下文

String clientID：开发用户 ID

String clientKey：开发授权码

BluetoothObuCallback callback：接收蓝牙 OBU 命令的回调类

返回值：

int：返回值为 OBU 初始化结果，如下：

| | |
|---|---|
| BLUETOOTH_OBU_INIT_SUCCESS（0） | OBU 初始化成功 |
| BLUETOOTH_OBU_INIT_CLIENTID_ERROR（1） | OBU 开发用户 ID 错误 |
| BLUETOOTH_OBU_INIT_CLIENTKEY_ERROR（2） | OBU 开发授权码错误 |
| BLUETOOTH_OBU_INIT_OBUNAME_ERROR（3） | OBU 名称错误 |

## A.2 向OBU发送命令接口

说明：

向已经连接成功的 OBU 发送命令。

类名：

BluetoothObuHandler

方法名：

public void sendObuCmd(String channel, String command)

输入参数：

String channel：接收命令的 OBU 的通道号，如下：

OBU_INIT_CHANNEL（“A0”）：OBU 初始化通道

OBU_MMI_CHANNEL（“A1”）：OBU 人机交互通道

OBU_ICC_RESET_CHANNEL（“A2”）：IC 卡复位通道

OBU_ICC_CHANNEL（“A3”）：IC 卡通道
OBU_ESAM_CHANNEL（“A4”）：ESAM 通道
OBU_CHANNEL（“A5”）：OBU 通道
OBU_SE_CHANNEL（“A6”）：OBU 的加密芯片通道
CREDIT_LOAD_CHANNEL（“A7”）：OBU 的认证及圈存业务通道
OBU_58G_CHANNEL（“A8”）：OBU 的 5.8G 通道
null ：非通道指令

String command：需要发送的命令内容

返回值：

无

## A.3 接收OBU命令接口

说明：

本接口为回调类接口，用户实现该接口及对应方法，实现接收到的 OBU 命令处理。

接口名：

BluetoothObuCallback

方法名：

public void onReceiveObuCmd( String channel, String command)

输入参数：

String channel：接收命令的 OBU 的通道号，如下：
OBU_INIT_CHANNEL（“B0”）：OBU 初始化通道
OBU_MMI_CHANNEL（“B1”）：OBU 人机交互通道
OBU_ICC_RESET_CHANNEL（“B2”）：IC 卡复位通道
OBU_ICC_CHANNEL（“B3”）：IC 卡通道
OBU_ESAM_CHANNEL（“B4”）：ESAM 通道
OBU_CHANNEL（“B5”）：OBU 通道
OBU_SE_CHANNEL（“B6”）：OBU 的加密芯片通道
CREDIT_LOAD_CHANNEL（“B7”）：OBU 的认证及圈存业务通道
OBU_58G_CHANNEL（“B8”）：OBU 的 5.8G 通道
null：非通道指令

String command：收到的 OBU 发来的命令内容

返回值：

无

## A.4 扫描和连接OBU接口

说明：

扫描 OBU 设备，并与该 OBU 建立蓝牙连接。

类名：

BluetoothObuHandler

方法名：

public void startScan()

输入参数：

无

返回值：

无

### A.5 连接OBU接口

说明：

与 OBU 建立蓝牙连接。

类名：

BluetoothObuHandler

方法名：

public boolean connectObu(BluetoothDevice bluetoothDevice)

输入参数：

BluetoothDevice bluetoothDevice：蓝牙设备

返回值：

boolean：返回 OBU 蓝牙连接状态，如下：

TRUE：执行成功

FALSE：执行失败

### A.6 断开OBU蓝牙连接接口

说明：

断开 OBU 蓝牙连接，或者取消正在建立的 OBU 蓝牙连接。

类名：

BluetoothObuHandler

方法名：

public void disconnectObu()

输入参数：

无

返回值：

无

### A.7 判断OBU蓝牙连接是否连接接口

说明：

判断手机 APP 与 OBU 之间蓝牙是否已连接。

类名：

BluetoothObuHandler

方法名：

public boolean isObuConnected ()

输入参数：

无

返回值：

boolean：返回 OBU 蓝牙连接状态，如下：

TRUE：手机 APP 与 OBU 之间的蓝牙已连接

FALSE：手机 APP 与 OBU 之间的蓝牙未连接

### A.8 向OBU发送响蜂鸣器（Beep）命令接口

说明：

手机 APP 向 OBU 发送响蜂鸣器命令，该命令指示 OBU 进行蜂鸣器鸣叫操作。

类名：

BluetoothObuHandler

方法名：

public boolean sendObuBeep ()

输入参数：

无

返回值：

boolean：返回执行结果，如下：

TRUE：执行成功

FALSE：执行失败

### A.9 向OBU发送闪灯命令接口

说明：

手机 APP 向 OBU 发送闪灯命令，该命令指示 OBU 进行闪灯操作。

类名：

BluetoothObuHandler

方法名：

public boolean sendObuFlash (int obuFlash)

输入参数：

int light：取值如下：

OBU_FLASH_RED（1）：指示 OBU 红灯闪烁

OBU_FLASH_GREEN（2）：指示 OBU 绿灯闪烁

返回值：

boolean：返回执行结果，如下：

TRUE：执行成功

FALSE：执行失败

### A.10 扫描到的设备回调接口

说明：

本接口为回调类接口，用户实现该接口及对应方法，扫描到蓝牙设备会调用此接口。

类名：

BluetoothObuCallback

方法名：

public void onScanSuccess( BluetoothDevice device)

输入参数：

BluetoothDevice bluetoothDevice：蓝牙设备

返回值：

无

### A.11 停止扫描OBU接口

说明：

停止蓝牙扫描设备

类名：

BluetoothObuhandler

方法名：

public void stopScan()

输入参数：

无

返回值：

无

### A.12 蓝牙断开回调

说明：

发送指令让 OBU 断开蓝牙连接，或者取消一个正在建立的 OBU 蓝牙连接。

类名：

BluetoothObuCallback

方法名：

Public void onDisconnect()

输入参数：

无

返回值：

无

# 附 录 B
（规范性附录）
iOS平台蓝牙API接口

## B.1 OBU初始化接口

说明：

OBU 初始化 API，功能包括对开发用户 ID，开发授权码进行校验；扫描对应的 OBU，并与该 OBU 建立蓝牙连接，设置接收 OBU 命令的回调类和方法等。

类名：

BluetoothObuHandler

方法名：

-(id) initObu: (NSString*)clientID ClientKEY:(NSString*)clientKey Delegate:(id< BluetoothObuHandlerDelegate >)delegate

输入参数：

NSString * clientID，开发用户 ID

NSString * clientKey，开发授权码

id< BluetoothObuDelegate>delegate：接收 OBU 蓝牙命令的回调类

返回值：

(id)：返回值蓝牙 OBU 初始化结果，如下：

| | |
|---|---|
| BLUETOOTH_OBU_INIT_SUCCESS（0） | OBU 初始化成功 |
| BLUETOOTH_OBU_INIT_CLIENTID_ERROR（1） | OBU 开发用户 ID 错误 |
| BLUETOOTH_OBU_INIT_CLIENTKEY_ERROR（2） | OBU 开发授权码错误 |
| BLUETOOTH_OBU_INIT_OBUNAME_ERROR（3） | OBU 名称错误 |

## B.2 向OBU发送命令接口

说明：

向已经连接成功的蓝牙 OBU 发送命令。

类名：

BluetoothObuHandler

方法名：

-(void)sendObuCmd: (NSString*)channel WithCommand: (NSData*)command

输入参数：

NSString* channel：接收命令的 OBU 的通道号，如下：

OBU_INIT_CHANNEL（“A0”）：OBU 初始化通道

OBU_MMI_CHANNEL（“A1”）：OBU 人机交互通道

OBU_ICC_RESET_CHANNEL（“A2”）：IC 卡复位通道

OBU_ICC_CHANNEL（“A3”）：IC 卡通道

OBU_ESAM_CHANNEL（“A4”）：ESAM 通道

OBU_CHANNEL（“A5”）：OBU 通道

OBU_SE_CHANNEL（“A6”）：OBU 的加密芯片通道

CREDIT_LOAD_CHANNEL（“A7”）：OBU 的认证及圈存业务通道
OBU_58G_CHANNEL（“A8”）：OBU 的 5.8G 通道
OBU_ESAM_RESET_CHANNEL（“A9”）：ESAM 复位通道
null : 非通道指令

NSData* command：需要发送的命令内容

返回值：

BOOL：发送命令执行结果

YES：执行成功

NO：执行失败

### B.3 接收OBU命令接口

说明：

本接口为回调类接口，用户实现该接口的对应方法，以实现蓝牙 OBU 命令的处理。

接口名：

BluetoothObuDelegate

方法名：

-(void)onReceiveObuChannel: (NSString*)channel WithData: (NSData*)receiveData

输入参数：

NSString* channel：接收命令的 OBU 的通道号，如下：

OBU_INIT_CHANNEL（“B0”）：OBU 初始化通道
OBU_MMI_CHANNEL（“B1”）：OBU 人机交互通道
OBU_ICC_RESET_CHANNEL（“B2”）：IC 卡复位通道
OBU_ICC_CHANNEL（“B3”）：IC 卡通道
OBU_ESAM_CHANNEL（“B4”）：ESAM 通道
OBU_CHANNEL（“B5”）：OBU 通道
OBU_SE_CHANNEL（“B6”）：OBU 的加密芯片通道
CREDIT_LOAD_CHANNEL（“B7”）：OBU 的认证及圈存业务通道
OBU_58G_CHANNEL（“B8”）：OBU 的 5.8G 通道
OBU_ESAM_RESET_CHANNEL（“B9”）：ESAM 复位通道
null：非通道指令

NSData* command：收到的 OBU 发来的命令内容

返回值：

无

### B.4 扫描和连接OBU接口

说明：

扫描 OBU 设备，并与该 OBU 建立蓝牙连接。

类名：

BluetoothObuHandler

方法名：

-(void)scanObuPeripherals:(int)timeout

输入参数：

(int)timeout：扫描的时间

返回值：

无

### B.5 连接OBU接口

说明：

与 OBU 建立蓝牙连接。

类名：

BluetoothObuHandler

方法名：

-(void)connectObu:(CBPeripheral*)obuPeripheral

输入参数：

(CBPeripheral*)obuPeripheral ：需要连接的蓝牙设备

返回值：

无

### B.6 断开OBU蓝牙连接接口：

说明：

手机端主动断开 OBU 蓝牙连接，或者取消一个正在建立的 OBU 蓝牙连接。

类名：

BluetoothObuHandler

方法名：

-(void) disconnectObu

输入参数：

无

返回值：

无

### B.7 判断OBU蓝牙是否连接接口：

说明：

判断手机 APP 与 OBU 之间蓝牙是否已连接。

类名：

BluetoothObuHandler

方法名：

-(BOOL)isObuConnected

输入参数：

无

返回值：

BOOL：OBU 蓝牙连接状态返回值，如下：

YES：手机 APP 与 OBU 之间的蓝牙已连接

NO：手机 APP 与 OBU 之间的蓝牙未连接

### B.8 向OBU发送响蜂鸣器（Beep）命令接口：

说明：

手机 APP 向 OBU 发送响蜂鸣器命令，该命令指示 OBU 进行蜂鸣器鸣叫操作。

类名：

BluetoothObuHandler

方法名：

-(void)sendObuBeep

输入参数：

无

返回值：

无

### B.9 向OBU发送闪灯命令接口：

说明：

手机 APP 向 OBU 发送闪灯命令，该命令指示 OBU 进行闪灯操作。

类名：

BluetoothObuHandler

方法名：

-(void)sendObuFlash:(NSString*)obuFlash withFlashRandomNumber:(NSInteger)random

输入参数：

NSString* obuFlash：

OBU_FLASH_RED（1）： 指示 OBU 红灯闪烁

OBU_FLASH_GREEN（2）：指示 OBU 绿灯闪烁

(NSInteger)random：闪灯次数

返回值：

无

### B.10 扫描到的设备回调接口：

说明：

本接口为回调类接口，用户实现该接口的对应方法，以实现接收到设备蓝牙的处理。

类名：

BluetoothObuHandlerDelegate

方法名：

-(void)onReceiveObuPeripheral:(CBPeripheral*)obuPeripheral

输入参数：

obuPeripheral：扫描到的蓝牙设备

返回值：

无

### B.11 停止扫描OBU接口：

说明：

停止扫描 OBU 设备。

类名：

BluetoothObuHandler

方法名：

-(void)stopScanObu

输入参数：

无

返回值：

无

### B.12 蓝牙断开回调

说明：

OBU 为保护自身的电量，超时主动断开蓝牙后，回调的接口

类名：

BluetoothObuHandlerDelegate

方法名：

-(void)onDisconnected;

输入参数：

无

返回值：

无

# 附 录 C
（规范性附录）
基于非对称证书的双向握手流程

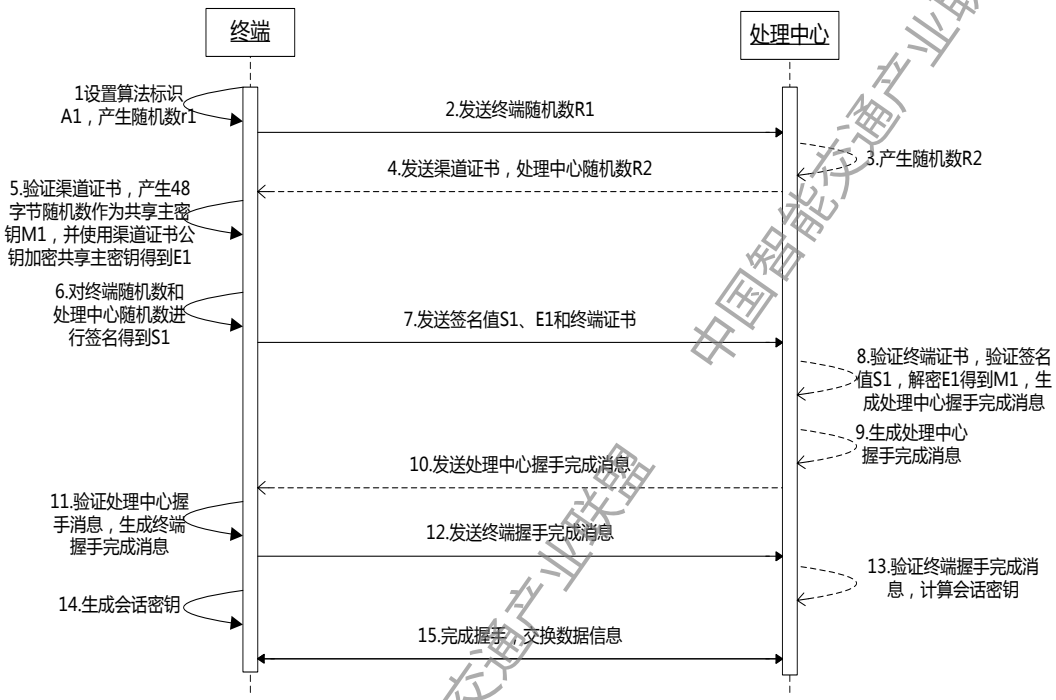


图 C.1　握手协议消息序列图

**握手协议工作步骤**

1） 终端产生随机数 R1；
2） 终端将随机数 R1 发送到处理中心，启动握手协议；
3） 处理中心产生随机数 R2；
4） 处理中心发送随机数和处理中心的渠道证书；
5） 终端使用终端中预置的平台证书验证收到的渠道证书，如果验证不通过，则发送出错消息，结束链接；否则，终端产生 48 字节随机数作为共享主密钥 M1，并且使用处理中心的渠道证书中的公钥采用非对称算法对 M1 加密得到 E1；
6） R1 和 R2 连接后得到 R3，终端先对 R3 进行摘要算法得到 H1，然后使用终端私钥对 H1 进行签名运算得到 S1；
7） 终端将 S1、E1 和终端证书发送到处理中心；
8） 处理中心使用平台证书验证终端证书合法性，若终端证书验证不通过，则发送错误消息，结束链接；如果终端证书验证通过，则使用终端证书验证 S1。若 S1 验证不通过，则发送错误消息，结束链接。否则，从 E1 中解密得到共享主密钥 M1；
9） 处理中心对渠道证书进行摘要运算得到 H2，对终端证书进行摘要运算得到 H3。将 R1、R2、H2、H3、S1、E1 连接后得到 T1（T1=R1||R2||H2||H3||S1||E1）；然后对 T1 进行摘要运算得到 H4；将 ASCII 码“SERVER”和 H4 连接后得到 D1；使用 M1 前 16 个字节对 D1 进行 HMAC 运算得到 F1；
10） 处理中心发送握手验证完成消息 F1 到终端；

11）终端验证接收到的处理中心发来的 F1，若验证不成功，则发送错误消息，结束链接；否则，发送终端握手验证消息F2到处理中心;F2运算与F1运算方法一样,只需要将F1运算时的ASCII码“SERVER”改为ASCII码“CLIENT”；

12）终端发送握手验证完成消息 F2 到处理中心；

13）处理中心使用同样的计算方法验证接收到的 F2 消息。验证失败，则发送错误消息，结束链接；

14）上述握手过程成功后，双方使用如下方法计算会话密钥：

X = HMAC(M1, key_label||r1||r2) .....

（M1取其前16个字节）

其中key_label为3字节ASCII码“KEY”，HMAC算法参见B.2。令X1X2…X20分别为X的第1个至第20字节，则加密密钥SKey为：SKey = X1X2…X16，MAC密钥MKey为：MKey = X5X6…X20；

15）握手过程结束。

HMAC 算法

HMAC 算法依照 FIPS 规范，使用摘要算法生成 HMAC。

**表 C.1　HMAC 算法参数说明**

| | |
|---|---|
| ipad | 填充字节串，内容为：8 位字节 0x36 重复 64 次 |
| opad | 填充字节串，内容为：8 位字节 0x5c 重复 64 次 |
| text | 所输入的需要计算 MAC 的数据，不包括填充字节串 |
| K | MAC 密钥 |
| t | 所得 MAC 的字节长度 |
| Hash 安全哈希算法 | 使用密钥 K 对 M 进行加密 |
| $D_K(C)$ | 见 FIPS 180-3 |

算法步骤

使用如下公式计算输入数据 text 的 MAC 值：

$MAC(text)_t = HMAC(K, text)_t = Hash((K_0 \oplus opad) || Hash((K_0 \oplus ipad) || text))$

具体描述如下：

1. 若 K = 64，令 $K_0 = K$。跳转到步骤 4；
2. 若 K > 64，令 $K_0 = Hash(K)$。跳转到步骤 4；
3. 若 K < 64，则在 K 末尾补字节 0x00 产生 64 字节 $K_0$；
4. $K_0$与 ipad 异或产生 64 字节字符串：$K_0 \oplus ipad$；
5. 将 text 追加到步骤 4 得到的字符串 $K_0 \oplus ipad$ 末尾：$(K_0 \oplus ipad)||text$；
6. 对步骤 5 得到的字符串进行哈希得到：$Hash((K_0 \oplus ipad)||text)$；
7. $K_0$与 opad 异或：$K_0 \oplus opad$；
8. 将步骤 6 产生的结果追加到步骤 7 的结果末尾：

$(K_0 \oplus opad) || Hash((K_0 \oplus ipad) || text)$

9. 对步骤 8 的结果做哈希得到：

$Hash((K_0 \oplus opad) || Hash((K_0 \oplus ipad) || text))$。

步骤 9 得到的哈希值作为 MAC 值

# 附 录 D
# （规范性附录）
# 圈存接口

本接口为 OBU 与 APP 之间的应用层接口，APP 调用圈存 API 时，需要按照本接口规定的数据帧格式传递数据。

## D.1 数据帧格式

数据帧格式：

| TYPE | LEN | DATA | BCC |
|---|---|---|---|

数据域说明见表 D.1。

**表 D.1 数据域说明**

| 字段 | 长度 | 描 述 |
|---|---|---|
| TYPE | 1 | 控制字，具体定义见附录 D.2 |
| LEN | 2 | 数据域长度 |
| DATA | n | 数据域，具体定义见附录 D.3 |
| BCC | 1 | 异或校验值，校验字段包含 CTL，LEN 和 DATA |

## D.2 控制字定义

APP 端发送到 OBU 的数据帧控制字定义见表 D.2。

**表 D.2 APP 端发送到 OBU 的数据帧控制字定义**

| 控制字 | 功能说明 |
|---|---|
| 0x60 | 充值申请 |
| 0x61 | 认证步骤 1 |
| 0x62 | 认证步骤 2 |
| 0x63 | 使用加密指令操作 IC 卡 |

OBU 发送到 APP 的数据类型定义见表 D.3。

**表 D.3 OBU 发送到 APP 的数据类型定义**

| 控制字 | 功能说明 |
|---|---|
| 0x06 | 充值申请应答 |
| 0x16 | 认证步骤 1 应答 |
| 0x26 | 认证步骤 2 应答 |
| 0x36 | IC 卡加密操作应答 |

## D.3 充值申请

报文字段说明见表 D.4。

**表 D.4 充值申请报文字段说明**

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| Data | 数据域 | 字节 | 10 | 预留，默认全 0 |

返回值说明见表 D.5。

**表 D.5 充值申请报文字段说明**

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| Code | 响应码 | 字节 | 1 | 0x00：表示成功<br>其他：表示失败 |
| AlgorithmID | 算法标识<br>A1 | 字节 | 1 | 值：默认 0x01，算法采用 RSA |
| Random1 | 终端随机数<br>R1 | 字节 | 32 | 32 字节随机数 |

## D.4 认证步骤 1

报文字段说明见表 D.6。

**表 D.6 认证步骤 1 报文字段说明**

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| CertificateNum | 渠道证书编号 | 字节 | 1 | 渠道证书的编号，默认为 0x01 |
| Random2 | 处理中心随机数<br>R2 | 字节 | 32 | 32 字节随机数 |
| PIN | 个人识别码 | 字节 | 6 | 验证 PIN 通过后才能使用终端私钥 |

返回值说明见表 D.7。

**表 D.7 认证步骤 1 返回值说明**

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| Code | 响应码 | 字节 | 1 | 0x00：表示成功<br>其他：表示失败 |
| SerialNum | 设备序列号 | 字节 | 8 | 设备序列号，使用压缩 BCD 码表示 |
| MasterKey | 共享主密钥<br>E1 | 字符 | 128 | 使用渠道证书公钥对共享主密钥 M1 进行加密得到 E1 |
| SignRandom | 随机数签名<br>S1 | 字节 | 128 | 将 R1、R2 连接得到 R3，然后后进行摘要算法得到 H1，最后使用终端私钥对 H1 进行签名运算得到 S1 |

## D.5 认证步骤 2

报文字段说明见表 D.8。

表 D.8　认证步骤 2 报文字段说明

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| ServerHMAC | HMAC<br>F1 | 字节 | 20 | 服务器握手验证信息 |

返回值说明见表 D.9。

表 D.9　认证步骤 2 返回值说明

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| Code | 响应码 | 字节 | 1 | 0x00：表示成功<br>其他：表示失败 |
| ClientHMAC | HMAC<br>F2 | 字节 | 20 | 终端握手验证信息 |

## D.6　IC卡操作

报文字段说明见表 D.10。

表 D.10　IC 卡操作报文字段说明

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| Instructions | 指令集 | 字节 | n | 加密后的指令集,参考 PBOC 数据,采用 TLV 格式（见附录 D.7） |

返回值说明见表 D.11。

表 D.11　IC 卡操作返回值说明

| 字段名 | 别名 | 类型 | 长度 | 说 明 |
|---|---|---|---|---|
| Code | 响应码 | 字节 | 1 | 0x00：表示成功<br>其他：表示失败 |
| InstructionResps | 指令集应答 | 字节 | n | 加密后的指令集应答，参考 PBOC 数据，采用 TLV 格式（见附录 D.7） |

## D.7　TLV（Tag，Length，Value）格式说明

**TAG**：固定一个字节。嵌套表示，0x80 表示 cmd，其对应的 value 为用户卡指令 TPDU 的合集。0x81 表示 resp，其对应的 value 为用户卡响应的合集，tag 对应每条 TPDU 指令的 tag。

Tpdu 指令的 tag 的低 4 位从 0x01 增长，表示用户卡 TPDU 指令和回复的序号，表示执行和回复顺序，高 4 位具有特殊含义，具体见表 D.12。

表 D.12　TLV（Tag，Length，Value）格式说明

| bit | 说 明 |
|---|---|
| 7 | 1：不返回执行结果；0：返回 |
| 6 | 1：执行失败时继续执行下一条指令；0：执行失败时不继续 |
| 5 | 保留 |
| 4 | 保留 |
| 3 | 指令和执行结果的序号 |
| 2 | |
| 1 | |
| 0 | |

**LEN**：变长表示。当需表示的长度小于 0x80 时，占一个字节，直接表示长度。 当需表示的长度大于 0x80 时，变长表示，用 0x80+n 来表示后续 n 个字节代表长度。

示例：

1、表示 0x77，LEN=0x77

2、表示 0x88，LEN=0x81 0x88

3、表示 0x0156，LEN=0x82 0x01 0x56

**VAULE**：值域，长度由 LEN 指定。

示例：

指令 0x80 + LENc + 0x01 lenc1 tpdu1 + 0x02 lenc2 tpdu2 + ... ...

应答 0x81 + LENr + 0x01 len1 resp1 + 0x02 len2 resp2 + ... ...

其中 LENc 和 LENr 指后续所有字节的长度。lenc1、lenc2 是 tpdu1、tpdu2 的长度，lenr1、lenr2 是 resp1、resp2 的长度，resp1、resp2 是 tpdu1、tpdu2 的返回数据。

# 附 录 E
（资料性附录）
圈存认证全流程

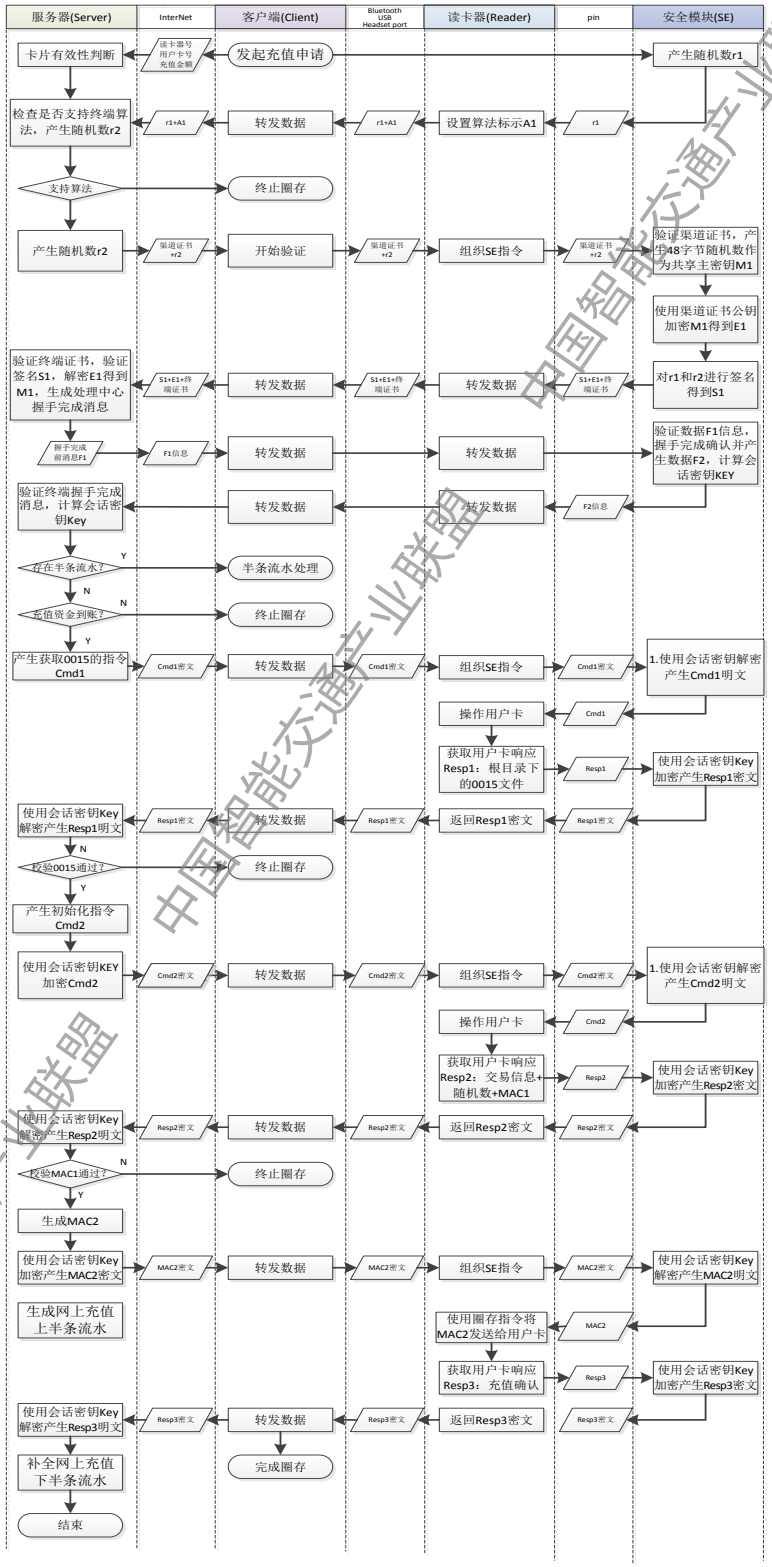


图 E.1　圈存认证全流程

# 附 录 F
# （资料性附录）
# 5.8G DSRC 信息推送接口

车载单元应支持路侧单元通过 5.8G DSRC 推送信息。该信息应采用 TRANSFER.rq 消息发送，并通过 transferChannel 标识目的设备为移动终端。

车载单元收到信息推送消息后，应通过蓝牙无线连接转发给手机 APP 处理。

车载单元对信息推送 TRANSFER.rq 的响应可不等待手机回复，以保证 DSRC 交互的效率，响应报文可固定返回成功。

本接口中定义的 container 编号、channelID 编号，需对 GB/T 20851 系列标准提出修订要求。

## F.1 TRANSFER.rq消息结构定义

表F.1 TRANSFER.rq消息结构定义

| 数据长度（字节数） | 字段 | 二进制位说明 B7 B0 | 数据描述 |
|---|---|---|---|
| 1 | 帧起始标识 | 0111 1110 | 0x7E |
| 4 | MAC 地址 | 32 位二进制数 | OBU 专用 MAC 地址 |
| 1 | MAC 控制域 | 0100 0000 | 下行链路、存在 LPDU、命令 |
| 1 | LLC 控制域 | 0111 0111 | 类型 3 |
| 1 | 段字头 | 1xxx x001 | 无分段。xxxx：PDU 号码，取值 00102。不得设定到 00002 或 00012. 此处可填 0x91 |
| 1 | Action-Request | 0000<br>0<br>1<br>0<br>1<br>需要重画，不明白具体含义 | Action.request |
| | accessCredentials | | 0 无需认证 |
| | actionParamter | | 1 有参数 |
| | IID | | 无 IID |
| | mode | | 1 需应答 |
| 1 | DID | 0000 0001 | 1 号目录位 ETC 应用目录 |
| 1 | ActionType | 0000 0011 | TransferChannel ActionType=3 |
| **1** | **ActionParameter** | **xxxxxxxx** | **Container=0x??待定** |
| **1** | **channelID** | **0010 0000** | **路径识别通道 0x20** |
| **1** | **InfoLength** | **xxxxxxxx** | **推送信息的长度** |
| **InfoLength** | **Info** | 32 位二进制数 | **推送信息内容** |
| 2 | FCS | 32 位二进制数 | 帧校验 |
| 1 | 帧结束标识 | 0111 1110 | 7E |

## F.2 TRANSFER.rs消息结构定义

表F.2 TRANSFER.rs消息结构定义

| 数据长度（字节数） | 字段 | 二进制位说明 B7 B0 | 描述 |
|---|---|---|---|
| 1 | 帧起始标识 | 0111 1110 | 7E |
| 4 | MAC 地址 | ———— | OBU 专用 MAC 地址 |
| 1 | MAC 控制域 | 1110 0000 | 上行链路、存在 LPDU、响应 |
| 1 | LLC 控制域 | 1111 0111 | 类型 3，第 8 位与 TransferChannel.rq 交替 |
| 1 | 状态子域 | 0000 0000 | 类型 3 操作的响应信息域 |
| 1 | 段字头 | 1xxx x001 | 无分段。xxxx: PDU 号码，取值 $0010_2$。不得设定到 $0000_2$ 或 $0001_2$. 此处可填 0x91 |
| 1 | Action-Response | 0001 1000 | Action.response,<br>存在 responseParameter |
| 1 | DID | 0010 0000 | 0x20, 信息推送 DID |
| 1 | ChannelRs | 0011 0011 | **Container=0x??，待定** |
| 1 | channelID | 0010 0000 | 路径识别通道 0x20 |
| 1 | ReturnStatus | —— | 处理状态 |
| 2 | FCS | ———— | 帧校验 |
| 1 | 帧结束标识 | 0111 1110 | 0x 7E |

中国智能交通产业联盟标准

**合作式智能运输系统 专用短程通信 无线互联车载单元设备应用**

T/ITS 0033-2015

北京市海淀区西土城路 8 号（100088）

中国智能交通产业联盟印刷

网址：http://www.c-its.org

2015 年 11 月第一版　2015 年 11 月第一次印刷