ICS 35.240.60
CCS L 67
备案号：103427-2023

DB11

北　京　市　地　方　标　准

DB11/T 1164.5—2023
代替 DB11/T 1164.5—2015

城市轨道交通自动售检票系统技术规范 第5部分：车票处理单元

Technical specifications of urban rail transit automatic fare collection system
—Part V:Ticket processing unit

2023－09－25 发布　　2024－01－01 实施

北京市市场监督管理局　发布

目　　次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则　第1部分：标准化文件的结构和起草规则》的规定起草。

本文件是DB11/T 1164《城市轨道交通自动售检票系统技术规范》的第5部分。DB11/T 1164分为以下9个部分：

——第1部分：系统结构及功能；

——第2部分：接口数据格式；

——第3部分：数据传输；

——第4部分：操作界面；

——第5部分：车票处理单元；

——第6部分：票卡；

——第7部分：终端；

——第8部分：检测；

——第9部分：技术指标体系。

本文件代替DB11/T 1164.5—2015《轨道交通联网收费系统技术要求 第5部分：车票处理单元》，与DB11/T 1164.5—2015相比，除编辑性修改外，主要技术变化如下：

——修改了范围（见1,2015版的1章节）；

——修改了规范性引用文件（见2,2015版的2章节）；

——增加了缩略语（见4）；

——修改了分体式车票处理单元外形尺寸（见6.1.2.2,2015版的5.1.2.2）；

——修改了嵌入式主机要求（见6.2,2015版的5.2）；

——修改了一体式车票处理单元电源适应性（见6.3.1,2015版的5.3.1）；

——修改了机械环境适应性（见6.5,2015版的5.5章节）；

——修改了性能要求（见7,2015版的6章节）；

——修改了应用接口要求（见8.7,2015版的7.7章节）；

——修改了附录A（见附录A，2015版的附录章节）。

本文件由北京市交通委员会提出并归口。

本文件由北京市交通委员会组织实施。

本文件起草单位：北京市轨道交通指挥中心。

本文件主要起草人：张莉、王照华、帅国莹、隋丽莉、梁材、张坤、张晓曦、周鳞真、靖立涛、戴国强、倪泽光、刘稳、韩鹏、尹宁、崔鹏鹏。

本文件历次版本发布情况为：

——本文件2015年首次发布为DB11/T 1164.5—2015。

本次为第一次修订。

城市轨道交通自动售检票系统技术规范
第5部分：车票处理单元

1 范围

本文件规定了城市轨道交通自动售检票系统车票处理单元的基本要求、性能要求和其他要求。

本文件适用于城市轨道交通自动售检票系统车票处理单元的设计、生产、检测与应用。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 6587 电子测量仪器通用规范

GB/T 9254 信息技术设备的无线电骚扰限值和测量方法

GB/T 17618 信息技术设备 抗扰度 限值和测量方法

GB/T 17626.2 电磁兼容 试验和测量技术 静电放电抗扰度试验

GB/T 17626.4 电磁兼容 试验和测量技术 电快速瞬变脉冲群抗扰度试验

GB/T 17626.8 电磁兼容 试验和测量技术 工频磁场抗扰度试验

JR/T 0025.3 中国金融集成电路（IC）卡规范 第3部分：与应用无关的IC卡与终端接口规范

JR/T 0025.5 中国金融集成电路（IC）卡规范 第5部分：借记/贷记应用卡片规范

3 术语和定义

下列术语和定义适用于本文件。

3.1

车票处理单元 ticket processing unit

具备车票交易处理整体功能的软、硬件综合体。

4 缩略语

API：应用程序接口（Application Programming Interface）

5 车票处理单元种类

车票处理单元分为一体式车票处理单元和分体式车票处理单元两种。一体式车票处理单元通常将控制板及天线集成在封闭外壳内；分体式车票处理单元包括控制板及天线两部分，通过馈线连接。

6 基本要求

6.1 外观结构

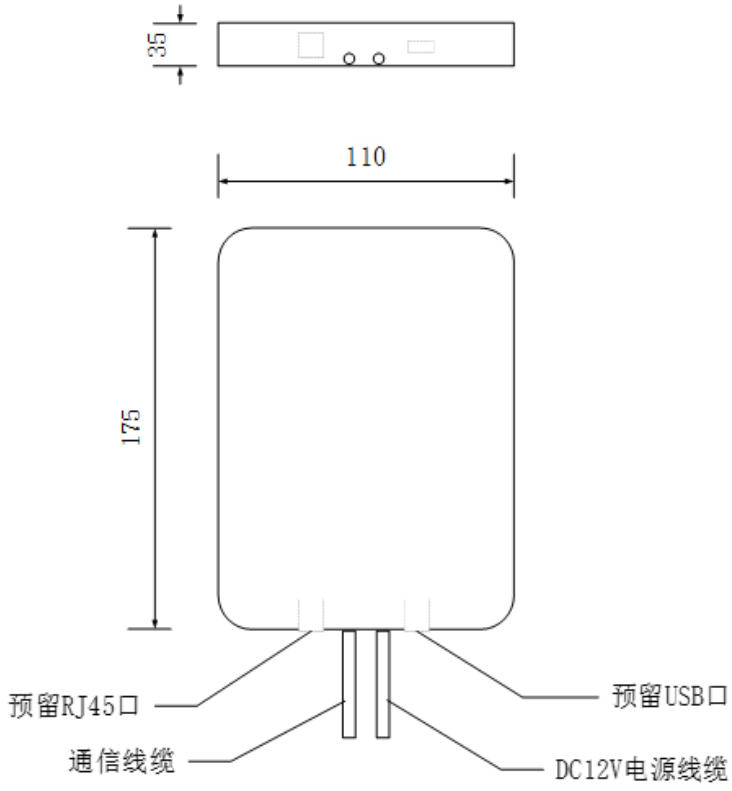
6.1.1 外观

车票处理单元表面应色泽均匀，面板与壳体封装严密。金属部分应无锈蚀和损伤，表面文字、图标、标志应清晰、牢固、完整。

6.1.2 外形尺寸

6.1.2.1 一体式车票处理单元外形尺寸

一体式车票处理单元不宜大于175 mm×110 mm×35 mm（长×宽×高），具体尺寸的要求参见图1。



单位为毫米

图1 一体式车票处理单元外形参考尺寸

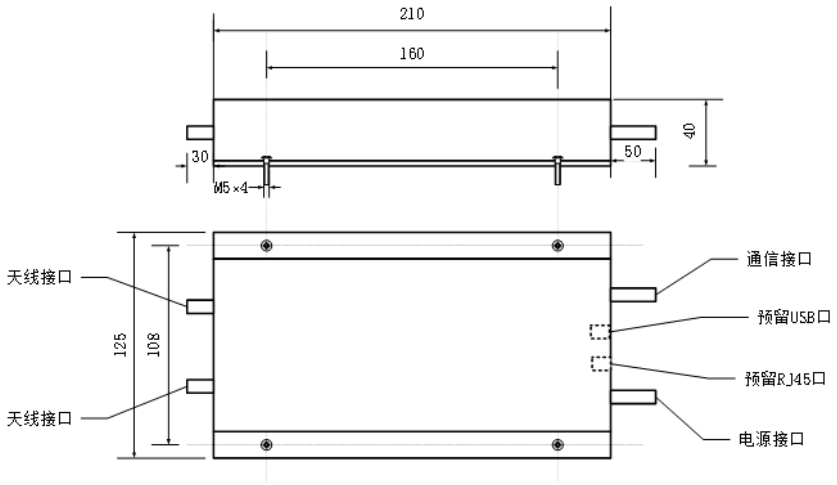
6.1.2.2 分体式车票处理单元外形尺寸

分体式车票处理单元要求参见表1。

表 1 分体式车票处理单元外形参考尺寸

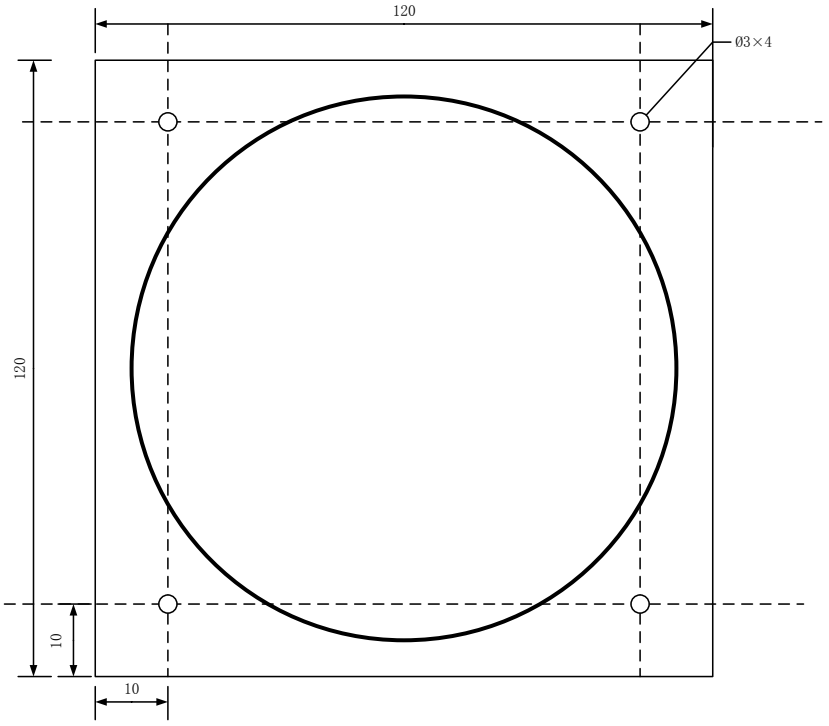
名称	尺寸(单位为毫米)			
	长	宽	高	直径
外壳尺寸	≤210	≤125	≤40	/
安装孔尺寸	160	108	/	Ø5
普通天线尺寸	120	120	20	/
普通天线安装尺寸	100	100	/	Ø3
小天线尺寸	75	45	20	/
小天线安装尺寸	68	38	/	Ø3

外壳的布局和参考尺寸见图 2，普通天线的安装尺寸参见图 3，小天线的安装尺寸参见图 4。



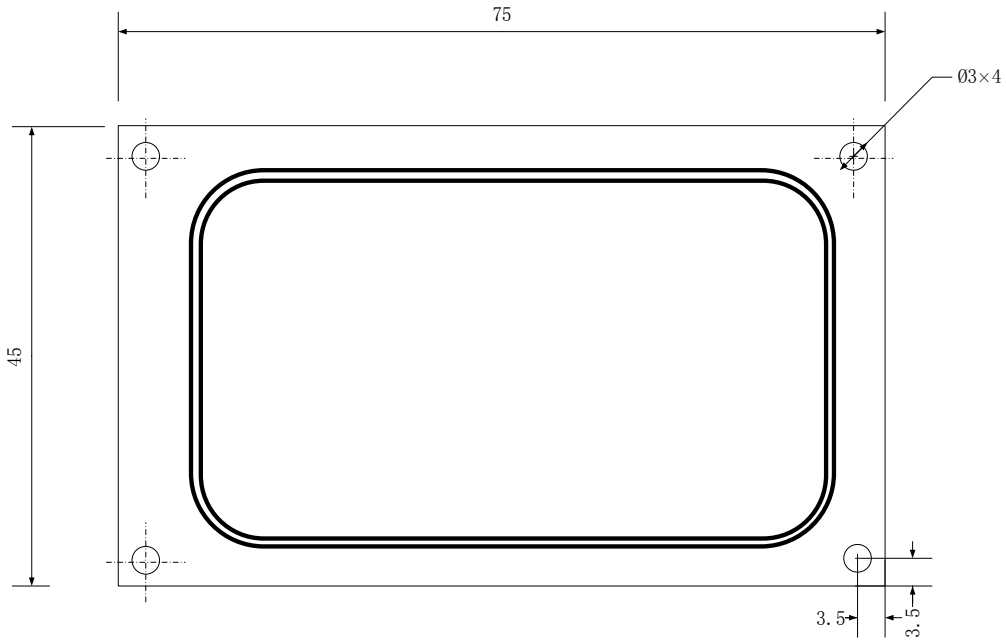
单位为毫米

图 2　车票处理单元外壳的参考布局和尺寸



单位为毫米

图 3　普通天线的参考尺寸



单位为毫米

图 4　小天线的参考尺寸

6.1.3　结构连接件

车票处理单元结构应牢固，有关联接导线、接插件及SAM卡插座应完好无损，天线接口采用带屏蔽的同轴电缆。

6.1.4　配置要求

车票处理单元应由天线、控制板、SAM卡、控制软件组成。控制板应包括微程序控制器、射频电路、接口单元、存储器和电源模块。

6.2　嵌入式主机要求

车票处理单元的嵌入式主机应满足如下要求：

a)　采用不低于 32 位嵌入式微处理器，工作频率应不低于 180 MHz；

b)　数据总线至少支持 16 位、32 位；

c)　外部地址总线不少于 16 位；

d)　具有复位和电源监控电路，具备看门狗功能；

e)　提供 3 线 RS232 或 4 线 RS422 数据通信接口，通信速率应不低于 115200 b/s。其中 D-Sub 9 针插头与终端设备连接，卡侬插头与车票处理单元连接，具体要求参见图 5；

f)　提供 USB2.0 及以上或 RS232 等多种主流接口，接口通信速率不低于 10 Mb/s；

g)　预留 10 Mbps/100 Mbps 自适应 RJ45 网络接口（可选）；

h)　支持通过 USB 接口连接移动式存储器；

i） 具备实时时钟，支持实时时钟的电池寿命应大于 5 年；

j） 支持在线编程，包括在系统编程和在应用编程。

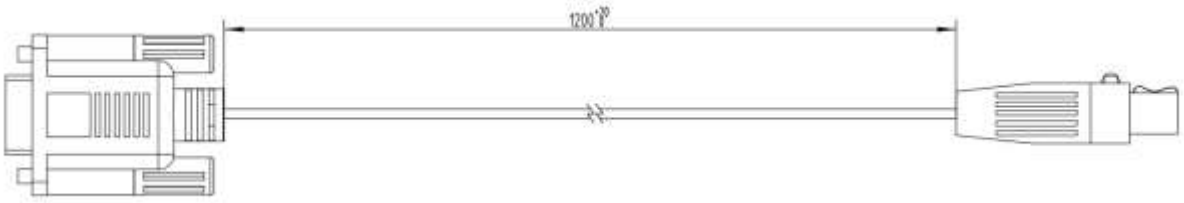


图 5 RS232 和 RS422 接口线缆的接线要求

6.3 电源适应性

6.3.1 一体式车票处理单元

一体式车票处理单元应采用直流电源供电，能在（12±0.5）V、（3±0.3）A条件下正常工作。

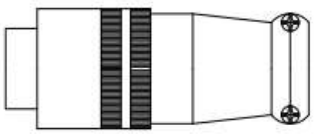
6.3.2 分体式车票处理单元

分体式车票处理单元电源应满足下列要求：

a） 采用直流电源供电，能在（12±0.5）V、（3±0.3）A 条件下正常工作；

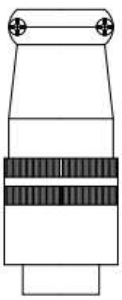
b） 采用 3 芯航空插头（公口）形式的电源；

c） 接口与终端设备相连接，电源接口三视图见图 6、图 7 和图 8。



右视图

图 6 电源接口右视图



上视图

图 7 电源接口上视图

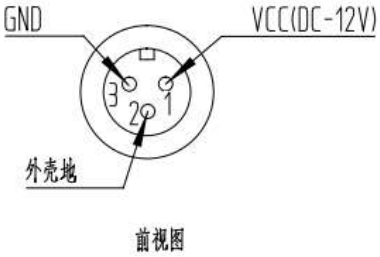


图 8 电源接口前视图

电源接口线的连接关系说明参见表 2。

表 2 电源接口线的连接说明

连接关系	引脚说明
3 芯航空插头的 1 脚	VCC(DC-12 V)
3 芯航空插头的 2 脚	外壳地
3 芯航空插头的 3 脚	GND
注：电源接口线可根据外接设备提供的 12 V 电源供电接口定制。	

6.4 电磁兼容性

6.4.1 无线电干扰极限值

车票处理单元的无线电干扰极限值应满足GB/T 9254规定的A级干扰极限值要求。

6.4.2 电磁敏感度

车票处理单元电磁敏感度满足下列要求：

a) 电磁敏感度应满足 GB/T 17618 规定的试验要求；

b) 静电放电敏感度应满足 GB/T 17626.2 规定的试验要求；

c) 辐射敏感度应满足 GB/T 17626.4 规定的试验要求；

d) 辐射干扰应满足 GB/T 17626.4 规定的试验要求；

e) 工作状态磁场干扰应满足 GB/T 17626.8 规定的试验要求；

f) 静电放电抗扰度应满足 GB/T 17626.2 规定的试验要求。

6.5 机械环境适应性

车票处理单元运输包装件跌落实验应满足GB/T 6587流通条件的要求，试验后性能特性应符合第7章性能要求的规定。

运输包装件跌落适应性参见表3。

表 3　运输包装件跌落适应性

包装件质量 kg	跌落高度（单位：mm）
≤15	1000
15～30（含 30）	800
30～40（含 40）	600
40～45（含 45）	500
45～50（含 50）	400
>50	300

6.6　环境适应性

6.6.1　工作环境温度及湿度

车票处理单元工作环境温度及湿度应满足下列要求：

a)　工作环境温度应为 0 ℃～50 ℃；

b)　工作环境相对湿度应为 20%～90%（非凝露）。

6.6.2　存储温度及湿度

车票处理单元存储温度及湿度应满足下列要求：

a)　存储温度：-20 ℃～60 ℃；

b)　存储相对湿度：10%～90%（非凝露）。

6.7　可靠性

车票处理单元的可靠性要求见表4。

表 4　可靠性要求

单位为次

名称	平均无故障次数（MCBF）	总使用次数（C）	总故障次数（F）
车票处理单元	≥100000	产生交易次数	全部关联故障
注：平均无故障次数（MCBF）=总使用次数（C）/总故障次数（F）。			

7　性能要求

车票处理单元性能参见表5。

表 5　车票处理单元性能要求

性能名称	性能要求
车票处理单元与天线的距离	应支持最长 2000 mm
车票处理单元 IC 卡读写距离（卡片至天线的距离）	40 mm～80 mm
普通天线的有效读写范围（以天线中心点为原点，车票平面与天线平面之间的角度不大于 45°）	X：±100 mm
	Y：±60 mm
	Z：0 mm～100 mm
小天线的读写范围（车票平面与天线平面平行）	40 mm～60 mm
车票处理单元的普通天线和小天线最小距离	50 mm，且不产生相互干扰和影响
存储指标	非易失性存储器空间不小于 256 MB； 易失性存储器空间不小于 64 MB； 访问程序运行存储器的时间周期不大于 7 ns； 访问程序存储器的最大时间周期不大于 90 ns。
设备的峰值功率	≤2 W
车票处理单元读写时间	单程票进出站处理时间：≤200 ms； 一卡通卡进出站处理时间：≤300 ms。
在有效的感应距离内，天线表面磁场强最大值	＜7.5 A/m
5 cm 处电磁场强度最小值	＞1.5 A/m
工作频率	13.56 MHz±7 kHz
车票处理单元与车票之间的通信速率	＞106 kb/s。

8　其他要求

8.1　车票处理单元识别要求

应识别符合JR/T 0025.5规定的非接触式IC卡。

8.2　防冲突要求

在多张实体车票同时处于同一个车票处理单元的操作区域内时，车票处理单元应不进行读写。

8.3　掉电保护要求

外部电源掉电时，不应破坏或改变车票处理单元的内存数据。电源恢复时，应恢复到掉电前的内存数据。

8.4　车票处理完整性要求

车票在读写过程中离开读写范围且再次进入时，车票处理单元应继续处理并保证写入信息的完整性。

8.5 安全模块接口要求

安全模块接口满足下列要求：

a) 车票处理单元应具有与安全模块连接的硬件接口；

b) 车票处理单元应至少提供 8 个 SAM 插槽，操作符合 JR/T 0025.3 规定的车票。车票处理单元通过 2 个专门的控制器来实现对卡片的访问，系统软件 API 库支持对这些卡片的访问（T=0/1 通信协议，符合 JR/T 0025.3 标准）；

c) 车票处理单元应可同时独立访问装载的 SAM 卡，并在不同的工作频率下同时独立工作；

d) 安全模块应具有加密和反破译功能，且物理上相对独立的硬件加密模块；

e) 安全模块 SAM 插槽的物理特性、逻辑接口和通信协议应满足 JR/T 0025.3 的要求；

f) 与安全模块 SAM 卡的通信应支持独立的 PPS 设置和高速通信（不低于 312 kb/s）。

8.6 应用功能要求

车票处理单元应用功能满足下列要求：

a) 应满足车票安全保密处理要求及业务处理流程要求,具备支持符合规范卡片的能力；

b) 应内部集成业务处理，包括车票读写处理、安全保密认证、业务处理、参数处理、交易数据生成、设备运行等功能。

c)

8.7 应用接口要求

应用接口满足下列要求：

a) 内部软件应包括内核软件及应用程序两部分,二者均保存在车票处理单元控制板上的非易失闪存中，应用程序对外提供 API；

b) 内核软件应提供任务运行服务、内存管理以及定时事件管理等功能，提供给应用程序的功能函数将编译成为系统 API 接口库，应用开发人员使用该 API 进行应用程序开发。当车票处理单元在线连接时，内核软件应支持通过串行口进行远程固件下载；

c) 应提供 API 接口函数来支持外部应用的编程,这些 API 接口函数将为低层硬件接口提供一种方便而清晰的接口，外部应用通过这些接口函数应能访问车票处理单元内的各种部件，API 接口参见附录 A；

d) 应通过硬件和软件保护相结合的方式,确保车票处理单元内的各种部件硬件不被外部应用程序直接访问，外部应用程序只有通过 API 接口函数才能访问这些硬件资源。车票处理单元应提供描述清晰且符合本标准的 API 接口函数文档,以便外部应用编程人员通过此 API 接口函数文档进行程序开发。

8.8 与终端设备的工作界面要求

8.8.1 与自动检票机接口界面

与自动检票机间的接口界面满足下列要求：

a) USB 驱动程序及预留 RJ45 或 RS232/RS422 接口：车票处理单元应提供不同运行环境（操作系统）平台下的 USB 及 RJ45 或 RS232/RS422 驱动程序，车票处理单元 USB、RJ45 或 RS232/RS422 接口的驱动程序安装于自动检票机 ECU；

b) 程序配置参数接口：闸机主程序应根据有关规范向车票处理单元提供配置参数。配置参数包括：票价表、运行模式、运行时间、黑名单、产品参数等；

c) 通信控制命令接口：闸机主程序应通过发送串行通信控制命令数据包的方式控制车票处理单元的运作、车票处理单元软件更新、获取交易数据、获取车票处理单元的运行情况。

与自动检票机间的接口界面参见图 9。

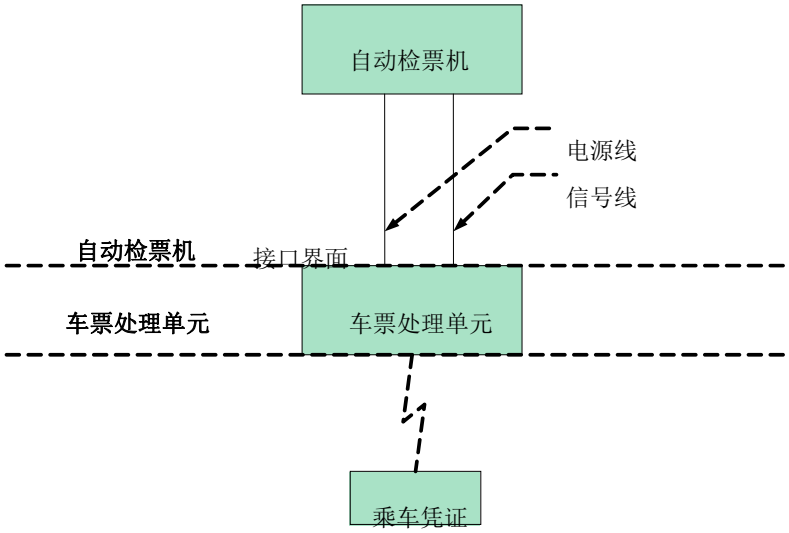


图 9 车票处理单元与自动检票机接口界面示意图

8.8.2 与自动售票机接口界面

与自动售票机接口界面满足下列要求：

a) USB 驱动程序及预留 RJ45 或 RS232/RS422 接口：车票处理单元应提供不同运行环境（操作系统）平台下的 USB 及 RJ45 或 RS232/RS422 驱动程序，车票处理单元 USB、RJ45 或 RS232/RS422 接口的驱动程序安装于自动售票机 ECU；

b) 程序配置参数接口：自动售票机主程序应根据有关规范向车票处理单元提供配置参数。配置参数包括：票价表、运行模式、运行时间、黑名单、产品参数等；

c) 通信控制命令接口：自动售票机主程序应通过 API 函数调用的方式控制车票处理单元的运作、车票处理单元软件更新、获取交易数据与获得车票处理单元的运行情况。

与自动售票机接口界面参见图 10。

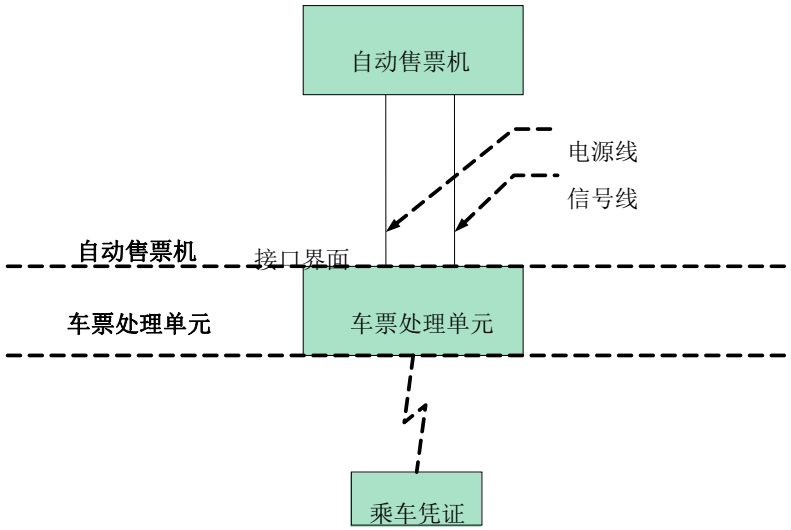


图 10 车票处理单元与自动售票机接口界面示意图

8.8.3 车票处理单元与半自动售票机接口界面

与半自动售票机间的接口界面满足下列要求：

a) USB 驱动程序及预留 RJ45 或 RS232/RS422 接口：车票处理单元应提供不同运行环境（操作系统）平台下的 USB 及 RJ45 或 RS232/RS422 驱动程序，车票处理单元 USB、RJ45 或 RS232/RS422 接口的驱动程序安装于半自动售票机 ECU；

b) 程序配置参数接口：半自动售票机主程序应根据有关规范向车票处理单元提供配置参数。配置参数包括：票价表、运行模式、运行时间、黑名单、产品参数等；

c) 通信控制命令接口：半自动售票机主程序应通过 API 函数调用的方式控制车票处理单元的运作、车票处理单元软件更新、获取交易数据与获得车票处理单元的运行情况。

与半自动售票机间的接口界面参见图 11。

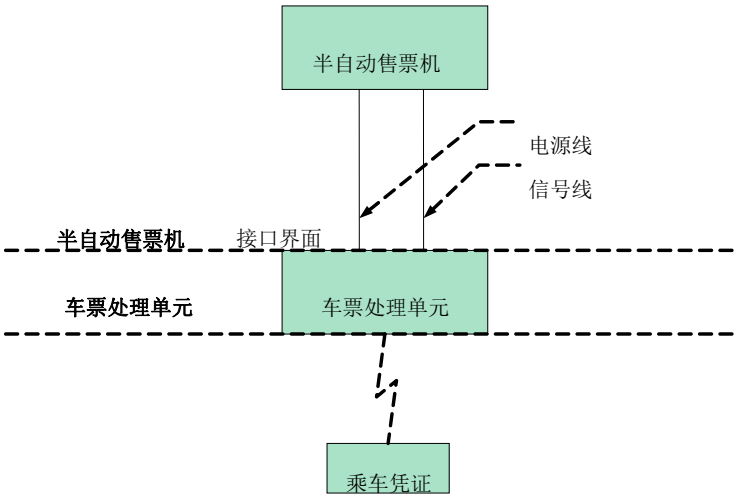


图 11 车票处理单元与半自动售票机接口界面示意图

附 录 A
（资料性）
API 接口示例

A.1 蜂鸣器API

A.1.1 Beep Key Open

Beep Key Open 包括：

a) 原型：S16_t BeepKeyOpen(void)；

b) 描述：打开 BeepKey（蜂鸣器）设备；

c) 参数：无。

A.1.2 Beep Key Close

Beep Key Close 包括：

a) 原型：S16_t BeepKeyClose(void)；

b) 描述：关闭 BeepKey（蜂鸣器）设备；

c) 参数：无。

A.1.3 Beep

Beep 包括：

a) 原型：S16_t Beep(U32_t BeepLev, U32_t Delay50Ms)；

b) 描述：蜂鸣器开始鸣叫；

c) 参数：

1) U32_t BeepLev： Beep 音调（0 --- 6） 其中 6 蜂鸣声音最响；

2) U32_t Delay50Ms： 蜂鸣的声音长度。

A.1.4 函数返回值

无特殊说明时，函数返回0为成功，其他为失败。

A.2 LED灯API

LedLight 包括：

a) 原型：S16_t LedLight(U32_t index, U32_t mode)；

b) 描述：控制 LED 灯；

c) 参数：

1) U32_t index：LED 灯的索引位置，从 1 开始；

2) U32_t mode： 灯状态 0 表示熄灭，1 表示亮。

A.3 以太网API

A.3.1 TCPIP_CommOpen

TCPIP CommOpen 包括：

a) 原型：S32_t TCPIP_CommOpen(U8_t * pPortDescriptor, U8_t *pOpenParams, S32_t dwPortAttr, S32_t nTimeout, S32_t *pErrCode)；

b) 描述：建立 TCP 连接；

c) 参数：

1) U8_t *pPortDescriptor： Linux 网络设备描述符字符串（"/dev/eth0","/dev/eth1"）；

2) U8_t *pOpenParams:IP 地址和端口号字符串具体格式 IP:PORT 比如"192.167.1.22:2000" 如果作为服务器，IP 地址可以忽略；

3) S32_t dwPortAttr：

工作模式：

0 表示本地服务器端口；

1 表示连接远程服务器的客户机；

2 表示连接本地服务器的客户机；

4) S32_t nTimeout： 连接超时时间澹(单位毫秒)；

5) S32_t *pErrCode：出错返回码，参考本标准部分的“函数返回码”。

d) 返回值：网络句柄

网络句柄包括：

1) =0 出错句柄 无效；

2) >0 正确句柄 有效。

A.3.2 TCPIP_CommAccept

TCPIP_CommAccept 包括：

a) 原型：S32_t TCPIP_CommAccept(S32_t hPort)；

b) 描述：服务器监控 TCP/IP 端口的 TCP/IP 连接；

c) 参数：

S32_t hPort： 服务器端口号；

d) 返回值：

网络句柄：

0 表示出错句柄 无效；

大于 0 表示正确句柄 有效。

A.3.3 TCPIP_CommRead

TCPIP_CommRead 包括：

a) 原型：S32_t TCPIP_CommRead(S32_t hPort,U8_t *pBuffer,S32_t nBytesToRead)；
b) 描述：向已打开 TCP/IP 端口读数据；
c) 参数：
   1) S32_t hPort：端口句柄；
   2) U8_t *pBuffer：接收数据所放数据指针；
   3) S32_t nBytesToRead：希望接收的数据个数；
d) 返回值：实际从 PORT 端口读的数据个数。

A.3.4 TCPIP_CommWrite

TCPIP_CommWrite 包括：

a) 原型：Int TCPIP_CommWrite(S32_t hPort,U8_t *pBuffer, S32_t nBytesToWrite)；
b) 描述：向已打开 TCP/IP 端口写入数据；
c) 参数：
   1) S32_t hPort： 端口句柄；
   2) U8_t *pBuffer： 发送数据所放数据指针；
   3) S32_t nBytesToWrite： 希望发送的数据个数；
d) 返回值：实际向 PORT 端口发送的数据个数。

A.3.5 TCPIP_CommControl

TCPIP_CommControl 包括：

a) 原型：
   S32_t TCPIP_CommControl(S32_t hPort, S32_t nCmd, void *pBuffer, S32_t nDataLength)；
b) 描述：向已打开 TCP/IP 端口通过命令发送控制信息；
c) 参数：
   1) S32_t hPort： 端口句柄；
   2) S32_t nCmd：
      发送命令字节具体规划如下：
      1 表示获取上次 TCP/IP 端口操作失败的错误码（适合服务器和客户器）；
      2 表示获取向端口发送的信息数据（适合服务器和客户器）；
      3 表示设置通信超时间（单位 ms）（适合服务器和客户器）；
      4 表示获取通信超时间；
      5 表示清除发送缓冲区数据；
      6 表示清除接收缓冲区数据；
      7 表示设置最大可连接客户机的个数；
      8 表示获取最大可连接客户机的个数；
      9 表示获取当前已经连接的客户机的个数；

10 表示获取当前服务器所能连接客户机器最大个数能力；

其他值表示为预留；

3) void *pBuffer： 输入数据指针；

4) S32_t nDataLength：输入数据长度：

当 nCmd =1 时候，pBuffer：错误码，nDataLength：固定 4 个字节；

当 nCmd =2 时候，pBuffer：输入信息指针，nDataLength：输入信息长度；

当 nCmd =3、4 时候，pBuffer：输入超时信息结构指针，nDataLength：输入超时信息结构长度；

当 nCmd =5、6 时候，pBuffer：NULL，nDataLength：NULL；

当 nCmd =7、8、9、10 时候，pBuffer：个数，nDataLength：固定 4 个字节。

A.3.6 TCPIP_CommClose

TCPIP_CommClose 包括：

a) 原型：S32_t TCPIP_CommClose(S32_t hPort)；

b) 描述：关闭已打开端口并释放端口资源；

c) 参数：

S32_t hPort：端口句柄；

d) 返回值：

1) 0 表示关闭成功；

2) 小于 0 表示关闭失败。

A.3.7 函数返回码

无特殊说明时，函数返回0为成功，其他为失败。

A.4 串口API

A.4.1 Serial_CommOpen

Serial_CommOpen 包括；

a) 原型：S32_t Serial_CommOpen(U8_t *pPortDescriptor, U8_t *pOpenParams, S32_t dwPortAttr, S32_t nTimeout, S32_t *pErrCode)；

b) 描述：打开串口函数；

c) 参数：

1) U8_t *pPortDescriptor： Linux 串口设备描述符字符串；

串口 0："/dev/ttyS0"；

串口 1："/dev/ttyS1"；

串口 2："/dev/ttyS2"；

串口 3："/dev/ttyS3"；

串口 4："/dev/ttyS4"；

2) U8_t *pOpenParams：串口参数字符串具体格式如："9600，n，8，1"；

3) S32_t dwPortAttr：工作模式；

=0 本地服务器端口；

=1 连接远程服务器的客户机；

=2 连接本地服务器的客户机；

4) S32_t nTimeout： 连接超时时间(单位毫秒)；

5) S32_t *pErrCode： 出错返回码，参考本标准部分的“函数返回码”；

d) 返回值：网络句柄

1) =0 出错句柄 无效；

2) >0 正确句柄 有效。

A.4.2 Serial_CommRead

Serial_CommRead 包括：

a) 原型：S32_t Serial_CommRead(S32_t hPort,U8_t *pBuffer,S32_t nBytesToRead)；

b) 描述：向已打开 TCP/IP 端口读数据；

c) 参数：

1) S32_t hPort ：端口句柄；

2) U8_t *pBuffer：接收数据所放数据指针；

3) S32_t nBytesToRead： 希望接收的数据个数；

d) 返回值：实际从 PORT 端口读的数据个数。

A.4.3 Serial_CommWrite

Serial_CommWrite 包括：

a) 原型：S32_t Serial_CommWrite(S32_t hPort,U8_t *pBuffer, S32_t nBytesToWrite)；

b) 描述：向已打开 TCP/IP 端口写入数据；

c) 参数：

1) S32_t hPort：端口句柄；

2) U8_t *pBuffer：发送数据所放数据指针；

3) S32_t nBytesToWrite： 希望发送的数据个数；

d) 返回值：实际向 PORT 端口发送的数据个数。

A.4.4 Serial_CommControl

Serial_CommControl 包括：

a) 原型：S32_t Serial_CommControl(S32_t hPort,S32_t nCmd, void *pBuffer, S32_t nDataLength;

b) 描述：向已打开串口端口通过命令发送控制信息；

c) 参数：

S32_t hPort： 端口句柄；

S32_t nCmd： 发送命令字节具体规划如下：

=1 获取上次 TCP/IP 端口操作失败的错误码(适合服务器和客户器)；

=2 获取向端口发送的信息数据（适合服务器和客户器）；

=3 设置通信超时间（单位 ms）（适合服务器和客户器）；

=4 获取通信超时间；

=5 清除发送缓冲区数据；

=6 清除接收缓冲区数据；

=7 设置最大可连接客户机的个数；

=8 获取最大可连接客户机的个数；

=9 获取当前已经连接的客户机的个数；

=10 获取当前服务器所能连接客户机器最大个数能力；

=其他保留以后使用；

void *pBuffer： 输入数据指针；

S32_t nDataLength： 输入数据长度：

当 nCmd =1 时：pBuffer：错误码；nDataLength：固定 4 个字节；

当 nCmd = 2 时：pBuffer：输入信息指针；nDataLength：输入信息长度；

当 nCmd =3、4 时：pBuffer：输入超时信息结构指针；nDataLength：输入超时信息结构长度；

当 nCmd =5、6 时：pBuffer：NULL；nDataLength：NULL；

当 nCmd =7、8、9、10 时：pBuffer：个数；nDataLength：固定 4 个字节。

A.4.5 Serial_CommClose

Serial_CommClose 包括：

a) 原型：S32_t Serial_CommClose(S32_t hPort)；

b) 描述：关闭已打开端口并释放端口资源；

c) 参数：

S32_t hPort：端口句柄。

A.4.6 函数返回码

无特殊说明时，函数返回0为成功，其他为失败。

A.5 射频驱动程序API

A.5.1 射频模块操作函数

A.5.1.1 RFIDMoudleOpen

RFIDModuleOpen 包括：

a) 原型：S16_t RFIDMoudleOpen(U32_t bIndex)；

b) 描述：打开射频模块功能；

c) 参数：

U32_t bIndex：选择射频模块号,在这里固定为 1。

A.5.1.2 RFIDMoudleClose

RFIDMoudleClose 包括：

a) 原型：S16_t RFIDMoudleClose(U32_t bIndex)

b) 描述：关闭射频模块功能

c) 参数：

U32_t bIndex：选择射频模块号,在这里固定为 1

A.5.1.3 RFIDInit

RFIDInit 包适：

a) 原型：S16_t RFIDInit(U32_t bIndex)；

b) 描述：初始化射频模块功能；

c) 参数：

U32_t bIndex：选择射频模块号(JC5620 固定为 1)。

A.5.1.4 SelectRFIDSlot

SelectRFIDSlot 包括：

a) 原型：S16_t SelectRFIDSlot(U32_t sLot)；

b) 描述：打开射频模块功能；

c) 参数：

U32_t sLot ： 选择射频模块号 0 或者 1。

A.5.2 TypeA 卡(Mifare One)相关函数

A.5.2.1 MifareGetSNR

MifareGetSNR 包括：

a) 原型：S16_t MifareGetSNR(U32_t mode, U8_t *bLen, U8_t *pSNR)；

b) 描述：寻卡并读出 UID+SNK+ATQA；

c) 参数：

U32_t mode：

=0 寻 IDLE 状态的卡；

=1 寻 HALT 状态的卡；

U8_t *bLen：信息长度

U8_t *pSNR： 包括(bLen-1)个字节 UID，一个字节为 SNK 码+2 个字节的 ATQA。

A.5.2.2 mif_Authen

mif_Authen 包括：

a) 原型：S16_t mif_Authen(U8_t cKeyab,U8_t cSecotrNo, U8_t *pKey,U8_t *pSNR)；

b) 描述：对 S50/S70 卡进行密钥认证；

c) 参数：

U8_t cKeyab；

=1 A 密钥；

=0 B 密钥；

U8_t cSecotrNo：扇区号；

U8_t *pKey：密钥；

U8_t *pSNR：卡唯一号。

A.5.2.3 mif_Read

mif_Read 包括：

a) 原型：S16_t mif_Read(U8_t cBlockNo,U8_t *pRdData)；

b) 描述：对 S50/S70 卡进行读卡操作；

c) 参数：

U8_t cBlockNo：块号；

U8_t *pRdData：读出的数据。

A.5.2.4 mif_Write

mif_Write 包括：

a) 原型：S16_t mif_Write(U8_t cBlockNo,U8_t *pWrData)；

b) 描述：对 S50/S70 卡进行写操作；

c) 参数：

U8_t cBlockNo：块号；

U8_t *pWrData：写入的数据。

A.5.2.5 mif_Change

mif_Change 包括：

a) 原型：S16_t mif_Change(U32_t cSubCommand, U32_t cBlockNo,U8_t *pValue);
b) 描述：对 S50/S70 卡进行密钥加减操作；
c) 参数：
U32_t cSubCommand：
=0xC0 减值操作；
=0xC1 加值操作；
U32_t cBlockNo：块号；
U8_t *pValue：值块。

A.5.2.6 mif_transfer

mif_transfer 包括：
a) 原型：S16_t mif_transfer(U32_t cBlockNo);
b) 描述：对 S50/S70 卡进行密钥传输操作；
c) 参数：
U32_t cBlockNo：块号。

A.5.2.7 mif_Halt

mif_Halt 包括：
a) 原型：void mif_Halt(void);
b) 描述：对卡进行暂停操作；
c) 参数：无；
d) 返回值：无。

A.5.2.8 Mifare_inc

Mifare_inc 包括：
a) 原型：S16_t mifare_inc(U8_t block , S32_t value);
b) 描述：对 TypeA 进行加值操作；
c) 参数：
U8_t block： 块号(0 -63)；
S32_t value：值域(小在前)。

A.5.2.9 Mifare_dec

Mmifare_dec 包括：
a) 原型：S16_t mifare_dec(U8_t block , long value);
b) 描述：对 TypeA 进行减值操作;

c) 参数：

U8_t block： 块号(0 -63)；

long value： 值域 (小在前).

A.5.2.10 Mifare_restor

Mifare_restor 包括：

a) 原型：S16_t mifare_restor(U8_t dest_block , U8_t src_block)；

b) 描述：对 TypeA 数据块进行移动操作；

c) 参数：

U8_t dest_block：目标块；

U8_t src_block：源块；

源块和目标块都必应满足 Mifare 值域数据格式标准。

A.5.3 TypeA卡(非接触CPU卡)操作函数

A.5.3.1 MifareGetSNR

MifareGetSNR 包括：

a) 原型：S16_t MifareGetSNR(U32_t mode, U8_t *bLen, U8_t *pSNR)；

b) 描述：寻卡并读出 UID+SNK+ATQA；

c) 参数：

U32_t mode：

=0 寻 IDLE 状态的卡；

=1 寻 HALT 状态的卡；

U8_t *bLen：信息长度

U8_t *pSNR： 包括(bLen-1)个字节 UID，一个字节为 SNK 码+2 个字节的 ATQA。

A.5.3.2 TypeA_RATS

TypeA_RATS 包括：

a) 原型：S16_t TypeA_RATS(U32_t cid,U8_t *resp)；

b) 描述：卡复位；

c) 参数：

U32_t cid：CID 码；

U8_t *resp：卡复位信息,第一个字节是复位信息的长度。

A.5.3.3 RF_APDU

RF_APDU 包括：

a) 原型：S16_t RF_APDU(U32_t cid,U8_t *send,U32_t len,APDU_RET *pStuApduResp)；

b) 描述：对 TypeA 的 CPU 卡进行操作；

c) 参数：

U32_t cid：多卡操作 cid 码；

U8_t *send：发送数据指令；

U32_t len：发送数据长度；

APDU_RET *pStuApduResp：接收数据结构指针；

APDU_RET 结构定义如下：

typedef struct{

U32_t Len;//接收数据长度；

U8_t inf[200]; //接收数据存放指针；

U16_t SW;//接收数据状态字；

}APDU_RET。

A.5.4 TypeA卡(非接触DesFire卡)操作函数

A.5.4.1 MifareGetSNR

MifareGetSNR 包括；

a) 原型：S16_t MifareGetSNR(U32_t mode, U8_t *bLen, U8_t *pSNR)；

b) 描述：寻卡并读出 UID+SNK+ATQA；

c) 参数：

U32_t mode：

=0 寻 IDLE 状态的卡；

=1 寻 HALT 状态的卡；

U8_t *bLen：

U8_t *pSNR： 包括(bLen-1)个字节 UID，一个字节为 SNK 码+2 个字节的 ATQA。

A.5.4.2 DesFire_APDU

DesFire_APDU 包括：

a) 原型：S16_t DesFire_APDU(DES_EXCHANGE *pStuApduResp)；

b) 描述：对 TypeA 的 DesFire 卡进行操作；

c) 参数：

DES_EXCHANGE *pStuApduResp：发送接收数据结构指针；

DES_EXCHANGE 结构定义如下：

typedef struct{

U8_t cSeLen; //发送数据长度；

U8_t cSendBuff[100];//发送数据缓冲区指针；

U8_t cReLen; //接收数据长度；

U8_t cReBuff[100];//接收数据缓冲区指针；

U8_t cTimeOut; //接收数据超时单位毫秒；

}DES_EXCHANGE。

A.5.5 TypeB卡操作函数

A.5.5.1 PiccREQB

PiccREQB 包括：

a) 原型： unsigned char PiccREQB(unsigned char afi,
unsigned char param,
unsigned char *atqb)；

b) 描述：对 TypeB 卡进行寻卡操作；

c) 参数：

unsigned char afi:0 所有的卡都响应,<>0 仅对应 AFI 的卡响应；

unsigned char param:

b7-b4:保留；

b3: 1—唤醒 halt 状态的卡；0—唤醒 IDLE 状态的卡；

b2-b0: 0 - 1 time Slot；

1 - 2 timeSlot；

2 - 4 timeSlot；

3 - 8 time Slot；

4 - 16 time Slot；

unsigned char *atqb: 响应的数据。

A.5.5.2 PiccAttrib

PiccAttrib 包括：

a) 原型： unsigned char PiccAttrib(unsigned char *uid,
unsigned char cid,
unsigned char *inf,
unsigned char inf_len,
unsigned char *ata)；

b) 描述： 对 TypeB 卡发送相应的 APDU 指令数据；

c) 参数：

unsigned char *uid:卡的 PUPI（4bytes）；

unsigned char cid: 0-14;

unsigned char *inf: 发送的指令数据信息;

unsigned char inf_len: 发送的数据长度;

unsigned char *ata: 返回的应答数据。

A.5.6 函数返回码

无特殊说明时，函数返回0为成功，其他为失败。

A.6 SIM卡驱动程序API

A.6.1 OpenSimMoudle

OpenSimMoudle 包括：

a) 原型：S16_t OpenSimMoudle(U32_t ucPlusinID);

b) 描述：打开 SIM 卡模块;

c) 参数：

U32_t ucPlusinID：卡槽(1 <= ucPiusinID <= 4)。

A.6.2 IccSimReset

IccSimReset 包括：

a) 原型：

S16_t IccSimReset(U32_t ucPlusinID,U32_t baud,

U32_t ucVoltage, U8_t *rLen, U8_t *ATR);

b) 描述：读取复位应答

c) 参数：

U32_t ucPlusinID：卡槽(1 <= ucPiusinID <= 4);

U32_t baud ：波特率(9600,19200,38400,115200);

U32_t ucVoltage：电压(1：1.8v;2：3.3V; 3：5V);

U8_t *rLen：复位应答数据长度;

U8_t *ATR：读取复位应答数据指针。

A.6.3 SimdSendAPDUT0

SimdSendAPDUT0 包括：

a) 原型：

S16_t SimdSendAPDUT0(U32_t Slot,ISO7816_ADPU_SEND *ApduSend,

ISO7816_ADPU_RESPONSE *ApduRecv);

b) 描述：向 SIM 卡发送指令和数据;

c) 参数：

U32_t Slot：输入的卡槽(1,2,3,4)

ISO7816_ADPU_SEND *ApduSend：输入命令参数；

ISO7816_ADPU_RESPONSE *ApduRecv：接收的参数结果；

ISO7816_ADPU_SEND 定义格式如下：

```
typedef struct{
U8_t CLA;
U8_t S;
U8_t P1;
U8_t P2;
U8_t LC;
U8_t DATA[240];
U8_t LEN;
}ISO7816_ADPU_SEND;
```

ISO7816_ADPU_RESPONSE 定义格式如下：

```
typedef struct{
U8_t LEN;
U8_t DATA[240];
U8_t SW1;
U8_t SW2;
}ISO7816_ADPU_RESPONSE。
```

A.6.4 IccCpuDetect

IccCpuDetect 包括：

a) 原型：S16_t IccCpuDetect(void)；

b) 描述：检测 CPU 卡；

c) 参数：无。

A.6.5 CloseSimModule

CloseSimModule 包括：

a) 原型：S16_t CloseSimModule(U32_t ucPlusinID)；

b) 描述：关闭 SIM 卡模块；

c) 参数：

U32_t ucPlusinID：卡槽。

A.6.6 GetSimVer

GetSimVer 包括：

a) 原型：S16_t GetSimVer(U32_t ucPlusinID, U8_t *DevMsg, U8_t *SoVer)；

b) 描述：获得 SIM 功能驱动和库函数版本信息；

c) 参数：

U32_t ucPlusinID：卡槽；

U8_t *DevMsg：内核驱动版本号；

U8_t *SoVer：动态库驱动版本。

A.6.7 函数返回码

无特殊说明时，函数返回0为成功，其他为失败。

A.7 时钟/铁电存储器/E2PROM存储器驱动程序API

A.7.1 FeromRtcOpen

FeromRtcOpen 包括：

a) 原型：S16_t FeromRtcOpen(void)；

b) 描述：打开时钟,FeRom,E2PROM 模块；

c) 参数：无。

A.7.2 FeromRtcClose

FeromRtcClose 包括：

a) 原型：S16_t FeromRtcClose(void)；

b) 描述：关闭时钟,FeRom,E2PROM 模块；

c) 参数：无。

A.7.3 GetDateTime

GetDateTime 包括：

a) 原型：S16_t GetDateTime(U8_t *Time)；

b) 描述：读系统时间(YYYYMMDDHHMMSS)；

c) 参数：

U8_t *Time：返回时间参数(YYYYMMDDHHMMSS)。

A.7.4 SetDateTime

SetDateTime 包括：

a) 原型：S16_t SetDateTime(U8_t *Time)；

b) 描述：设置系统时间(YYYYMMDDHHMMSS)；

c) 参数：

U8_t *Time：返回时间参数(YYYYMMDDHHMMSS)。

A.7.5 SyncTimeFromPOS

SyncTimeFromPOS 包括：

a) 原型：S16_t SyncTimeFromPOS(void)；

b) 描述：同步 Linux 系统时间；

c) 参数：无。

A.7.6 FeSysWrite

FeSysWrite 包括：

a) 原型：S16_t FeSysWrite(U32_t lFeAddr,U32_t lrLength, U8_t *pRtr)；

b) 描述：写铁电存储器数据；

c) 参数：

U32_t lFeAddr：地址 0x0000 -- 0x2000；

U32_t lrLength：写入数据长度；

U8_t *pRtr：写入数据指针。

A.7.7 FeSysRead

FeSysRead 包括：

a) 原型：S16_t FeSysRead(U32_t lFeAddr,U32_t lrLength, U8_t *pRtr)；

b) 描述：读铁电存储器数据；

c) 参数：

U32_t lFeAddr：地址 0x0000 -- 0x2000；

U32_t lrLength：读入数据长度；

U8_t *pRtr：读入数据指针。

A.7.8 函数返回码

无特殊说明时，函数返回0为成功，其他为失败。