**中国矿业大学**

**计算机科学与技术学院**

**2017级本科生课程报告**

课程名称 硬件课程设计

设计题目 出租车计价器(C语言)

开课学期 2020学年第一学期

报告时间 2019年12月20日

学生姓名 陆玺文

学 号 03170908

班 级 计科2017-06班

专 业 计算机科学与技术

任课教师 马海波

《硬件课程设计》课程报告评分表

开课学期: 2020学年第一学期

姓名: 陆玺文 学号：03170908 专业班级： 计科2017-06

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 毕业要求 | 课程教学目标 | 考查方式与考查点 | 占比 | 得分 |
| 1 | 1.3 | **目标1：**了解微机应用系统解决复杂工程问题的基本方法。掌握微机应用系统硬件电路设计及软件功能需求分析方法和模型。能够针对微机系统应用领域工程需求的系统要求，进行分析与设计。 | 中期检查与设计文档  掌握解决复杂工程问题的基本方法。微机应用系统软硬件设计相关的理论知识。 | 10% |  |
| 2 | 4.3 | **目标2：**能够针对硬件电路组成需求描述进行系统硬件设计，能够分析系统功能的软件需求，根据模块设计原则，综合考虑系统的算法模型和软硬件开发，进行合理的方案设计、编程实现、系统测试及对设计方案进行优化。 | 中期检查与设计文档  考核题目需求分析和功能分析；综合知识应用能力及设计方案的完整性；  考核软件编程及系统调试测试，设计方案进行优化。 | 30% |  |
| 3 | 9.1 | **目标3：**具备多学科背景知识，并制定项目计划，能够按照标准规范进行设计。能够在多学科背景下具备独立分析问题解决问题的能力。 | 中期检查与设计文档  考核独立分析问题解决问题的能力 | 10% |  |
| 4 | 10.3 | **目标4：**掌握设计报告撰写，通过成果演示、陈述发言的清晰表达、回答问题准确性等。 | 现场验收与答辩  考核编程实现的代码难度和复杂性、设计工作量等；考核设计成果、所涉及的问题答辩。验收设计报告的结构合理性、内容和图表的正确性。验收设计报告排版的规范性。 | 40% |  |
| 5 | 12.1 | **目标5：**对选题主动通过各种途径寻求解决方法（主动查阅资料、请教老师、同学讨论等）。通过各种资源平台的使用及教师意见的反馈，完成高质量的设计任务，有无创新意识。 | 现场验收与答辩  考核设计成果完整性；所涉及的设计课题的创新性。 | 10% |  |
| 总成绩 | | | | 100% |  |

任课教师：马海波

2019年12月20日

目录

[1.绪论 1](#_Toc27746213)

[1.1选题说明 1](#_Toc27746214)

[1.2设计任务及要求 1](#_Toc27746215)

[2.系统设计需求分析 1](#_Toc27746216)

[2.1实验系统及软件开发平台 1](#_Toc27746217)

[2.2系统功能需求分析 1](#_Toc27746218)

[2.3系统主要算法及分析 2](#_Toc27746219)

[2.4系统的组成及工作原理 3](#_Toc27746220)

[2.4.1 8255工作任务 3](#_Toc27746221)

[2.4.2 8254工作任务 3](#_Toc27746222)

[2.4.3 DAC0832工作任务 4](#_Toc27746223)

[2.4.4 8×8双色点阵 4](#_Toc27746224)

[2.4.5 LCD12864液晶屏 4](#_Toc27746225)

[2.4.6 8259工作任务 4](#_Toc27746226)

[2.4.7 AD0809工作任务 4](#_Toc27746227)

[3.系统概要设计 4](#_Toc27746228)

[3.1系统层次图 5](#_Toc27746229)

[3.2静态建模 5](#_Toc27746230)

[4.系统详细设计 6](#_Toc27746231)

[4.1空车牌翻动 6](#_Toc27746232)

[4.2汽车加减速 6](#_Toc27746233)

[4.3里程计量 7](#_Toc27746234)

[4.4价格计算 8](#_Toc27746235)

[4.5状态切换 8](#_Toc27746236)

[4.6界面显示与刷新 9](#_Toc27746237)

[4.8打印模块（喇叭发声模块） 10](#_Toc27746238)

[4.9中断接口 10](#_Toc27746239)

[4.10键值读取 11](#_Toc27746240)

[4.11录放音模块 11](#_Toc27746241)

[5.系统编码测试 12](#_Toc27746242)

[5.1单元测试（类测试） 12](#_Toc27746243)

[5.2集成测试（系统测试） 13](#_Toc27746244)

[6.设计结果分析及结论 14](#_Toc27746245)

[6.1计数初值偏离 14](#_Toc27746246)

[6.2按键灵敏度降低 14](#_Toc27746247)

[7.实验体会 14](#_Toc27746248)

[参考文献 14](#_Toc27746249)

[附录 15](#_Toc27746250)

# 1.绪论

1.1选题说明

如今，随着城市经济的发展，市民对于快速出行有了更高的需求，城市内的出租车数量也大大提高，计价器作为里程计量与金额计算仪器得到了广泛的运用。设计与实现一个计量精准，功能完善，界面美观的计价器具有极高的实用价值经济价值。

此外，计价器主功能为里程计算与界面显示，对于硬件的要求并不苛刻，在所选用的实验平台上可以完备地模拟出完整的计价器功能，能够更好地将所设计实现的模拟计价器与实物进行比对。

1.2设计任务及要求

利用实验仪器与平台，结合C语言编程，模拟出租车计价器的实现。利用0832控制直流电机运转，通过8253采集霍尔传感器的转速信号, 并在LCD屏显示公里数及价格，由8255模拟车辆的启、停，白天、夜间计价及清零功能。

在此基本功能基础上，加以创新实现新功能。

# 2.系统设计需求分析

2.1实验系统及软件开发平台

实验系统：TPC-ZK-II综合开放式微机原理及接口技术实验系统

软件平台：VC6.0

2.2系统功能需求分析

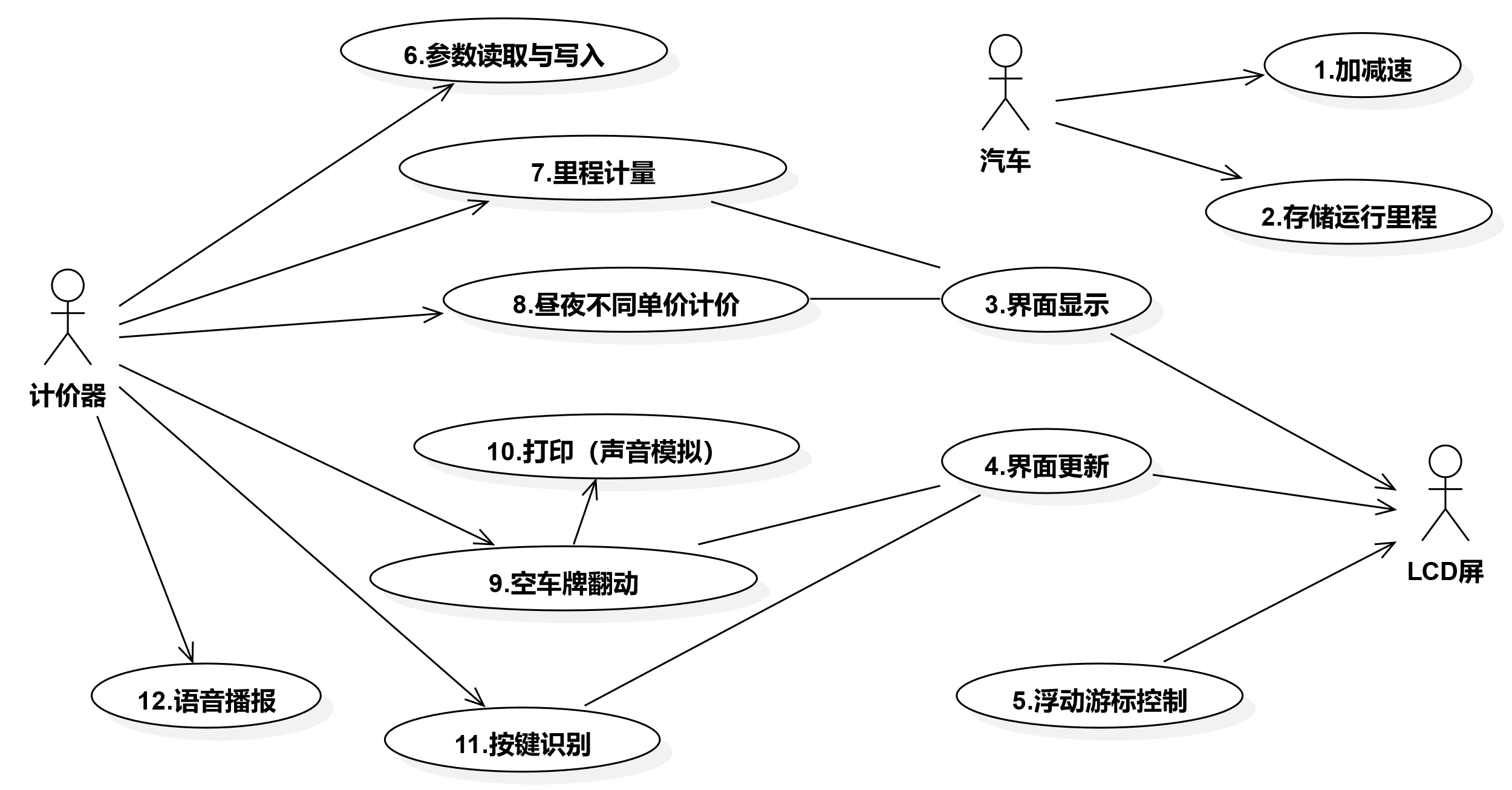


图 1 系统需求用例图

图 1展示了系统的**功能需求**。做为演示系统，应当能够模拟汽车的加减速，进而实现计价显示的数值观察。车辆也自然应当存储行驶里程记录，便于计价器的读取。计价器应当能够实现空车牌翻动、里程计量、价格计算、昼夜单价识别切换、打印凭条、识别按键、语音播报欢迎等功能。

此外，在计价器的工作运行过程中，应当满足界面显示美观，按键响应灵敏，界面切换更新顺畅，计价准确等**性能要求**。

2.3系统主要算法及分析

考虑到在计价器的运行过程中，按键操作是主要的交互操作，因而围绕按键识别与程序跳转设计系统主要的运行算法。



图 2 系统主算法流程图

如图 2所示，在系统中给计价器定义若干状态，程序中循环**查询键盘**，若有按键按下，则进入状态改变程序段，进行相应处理；若无按键按下，则根据目前的状态执行相应的界面更新。

考虑到按键动作与中断请求相似，引入**中断类接口[[1]](#footnote-1)**，使系统支持运用中断识别按键。

2.4系统的组成及工作原理

从图 1中系统需求中可以看出，需要使用的硬件有8255、8254、DAC0832、键盘、74LS273简单输出、LCD12864液晶屏、8×8点阵屏、AD0809、喇叭。

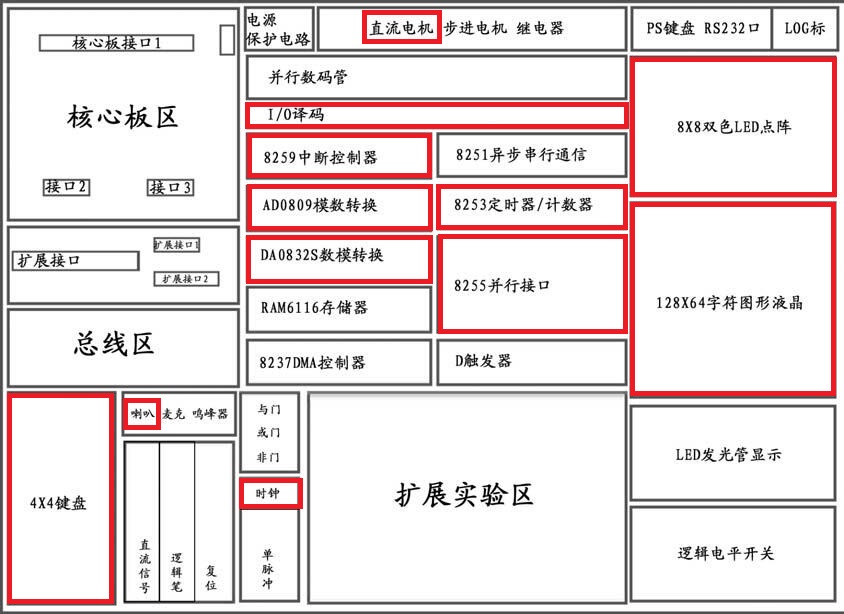


图 3 TPC-ZK实验系统使用部分示意图

2.4.1 8255工作任务

8255是系统的主控芯片，选用方式字0x81，使A口B口输出，C口高4位输出第四位输入。当系统采用查询键盘的方式进行工作时，C口高4位负责行线的拉低，第四位读取列线，进而识别按下的按键。B口的低3位负责控制LCD12864液晶的控制引脚，B口最高位B7通过控制8253的Gate门控信号，使系统发出“嗒嗒嗒”声模拟打印操作。

2.4.2 8254工作任务

8254在系统中，一是负责接收电机所产生的霍尔传感器信号，进行里程计数；二是输出特定频率波，模拟打印声“嗒嗒嗒”；三是可以当做定时器，进行界面的定时刷新。

2.4.3 DAC0832工作任务

DAC0832在系统中，一是负责模拟汽车的档位控制，调节电机转速；二是在录放音过程中，输出不同频率的信号。

2.4.4 8×8双色点阵

点阵屏在系统中，模拟空车牌，通过亮（灭）对应相应的空车牌抬起（翻下）。

2.4.5 LCD12864液晶屏

液晶屏在系统中，主要负责界面的显示与更新，同时增添游标显示（关闭）使得在进一步拓展设置参数时可以有所提示。

2.4.6 8259工作任务

TPC-ZK-II实验平台在USB核心板上包含主、从两片8259，在计价器系统中，考虑到端口数量及C语言不易注册中断向量问题，选用扩展8259芯片，结合查询中断查询字的方法，将8255查询键盘方式进行升级改进。同时使用一行按键，使得系统可以通过列线跳变自动得知按键键值。

(a)查询式下选用按键示意图 (b)中断方式下选用键盘示意图

图 4 查询与中断方式下键盘示意图

2.4.7 AD0809工作任务

AD0809在系统中，主要负责在录放音模块中采集语音，并转换为数字量进行存储，供以后进行调用播放。

# 3.系统概要设计

采用面向对象的软件设计方法，使用MVC架构进行程序编写，每一芯片为一个模型（Model），实现对业务逻辑的处理；液晶屏与键盘代表视图（View），分别用于显示数据和接受用户请求与输入；出租车和计价器类为控制器（Controller），连接模型和视图，调用不同的模型处理用户请求，选择不同视图显示系统的信息。

3.1系统层次图

图 5 计价器系统层次图

3.2静态建模

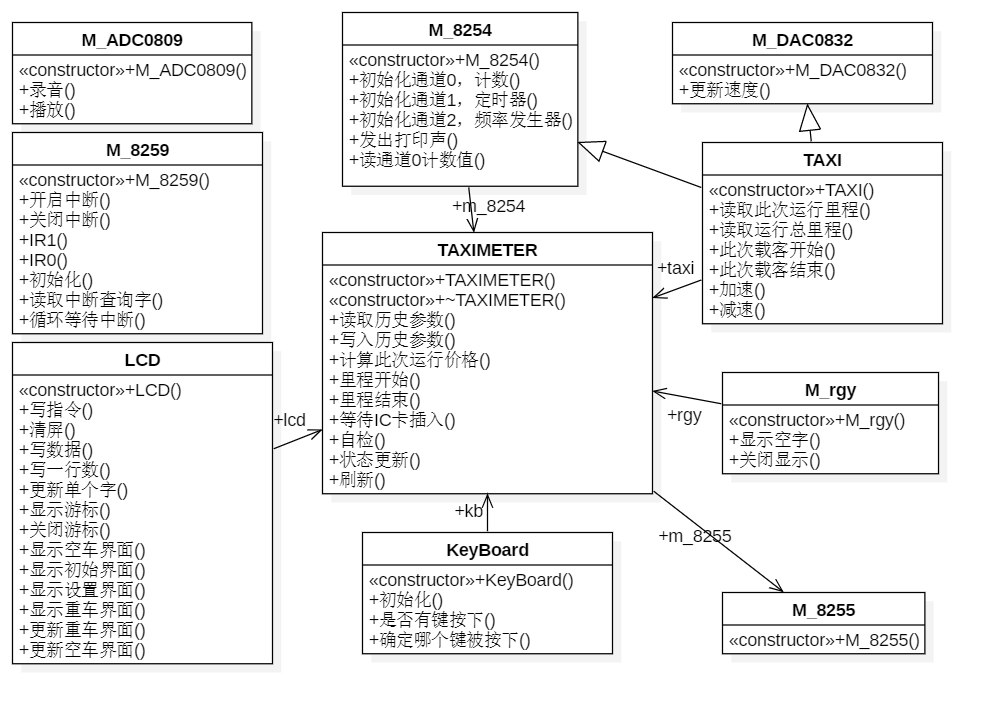


图 6 计价器系统类图（隐藏属性值）

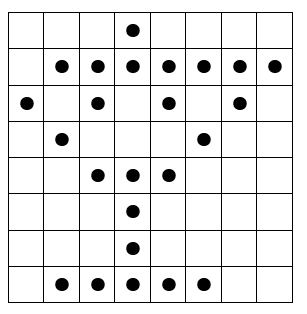
图 6中，设计了程序的整体框架，主要以各个芯片为一类，另外设计TAXI（车辆）类，封装加速减速操作，以及负责与8254交互读取当前里程的里程数。TAXIMETER（计价器）类，为程序主控制类，负责程序整体运行流程的控制，同时接收键盘键值，执行相应的状态转换，并负责界面的更新。

# 4.系统详细设计

在架构设计完成的基础上，针对整个系统的各个模块，进行详细的设计并实现，最终整合各模块，完成系统的整体功能。

4.1空车牌翻动

空车牌的翻上（翻下），在现实计价器功能中代表了单次运营服务计价的开始（结束）。在微机实验箱上，选用8×8点阵屏显示“空”字，模拟空车牌的亮起。同时在查询方式下设定按键“B”的按下事件代表空车牌的翻动，进行点阵屏相应的亮灭操作。

(a)点阵屏空字示意图 (b)点阵屏接线示意图

图 7 点阵屏显示与接线示意图

在具体的编程实现过程中，由图 7可以看出，首先针对点阵屏的特点，对所要进行显示的“空”字编码，然后逐行输出特定码值，动态刷新显示各行，利用视觉暂留实现“空”字的显示。整个点阵屏工作在总线模式下，数据区与总线进行交互。同时，为了提高显示效果，在每刷新一行后，延时一小段时间（10ms），从而点阵屏更加稳定。

4.2汽车加减速

在系统中，使用直流电机的转动模拟汽车的运行，转速即为运行时速。DAC0832控制电机的转速，接受一个0~255内的数值，输出相应模拟量控制电机。在此基础上TAXI类继承DAC0832，守护速度属性值，同时提供加减速操作。

DAC0832片选端口外接Y0:280，0 ~ +5v向直流电机输出。

TAXI类代表汽车动作与状态，并维护未启动(state:0)，运行(state:1)，静止等待(state:10)，三个状态。

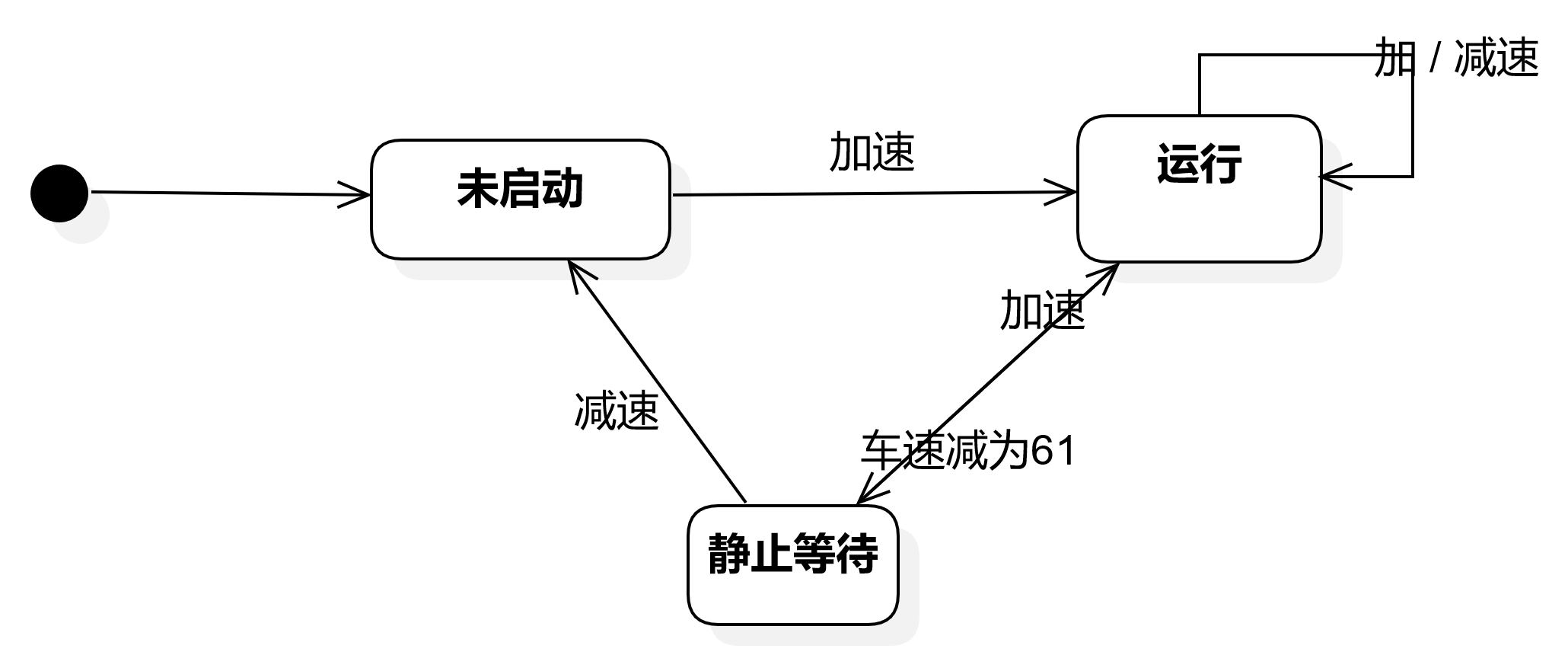


图 8 车辆（TAXI类）状态图

如图 8所示，当汽车速度达到254时，不再响应加速动作；当速度不大于70时，再接受减速操作时，速度固定为61，同时转为静止等待状态；速度为61时，再次响应减速操作则认为汽车已经熄火，进入未启动状态。

4.3里程计量

在系统中，TAXI类负责维护自身的里程计量，包含单次里程与总运营里程[[2]](#footnote-2)。具体里程计量由8254芯片的计数功能实现。编写M\_8254类封装8254的功能，其中通道0工作在方式0下，采用倒计数的方式，计数初值为16位二进制的最大数65536。同时提供读数方法，及时锁存端口计数值，读出数值取补并返回TAXI，得到的即为相应时间段内通过的脉冲数量。当读出计数值大于设定阈值（60000）时，命令8254自行重装计数初值。



图 9 TAXI类里程计量流程图

TAXI类向外提供里程数读取方法，内部调用8254读数方法得到一个计数返回值，这里维护一个上次计数值（temp），将两者差值除以比例因子（factor）即得到里程数。

如图 9所示，考虑到8254计数时自行重装计数初值的问题，每次读数所得值与temp进行比较，若小于temp则说明8254在时间间隔内完成了一次计数初值的重装。此时的里程计算不再使用差值，而直接使用所读到的计数初值进行计算[[3]](#footnote-3)。

4.4价格计算

计价器主类TAXIMETER从车辆类TAXI中，读取到当前里程数后，负责对此次服务价格的计算。为了实现昼夜不同单价计量，在每次服务开始时，系统会根据当前时间确定所要使用的单价，并维护当前运行单价（Per\_Price\_Now）这一属性值。计算价格时，采用分段计价法，并将结果处理成精度为小数点后一位的值返回。



上式即为计价器中价格计算函数，其中*M*为总价格Money，*PS*为起步价，*PN*为单次运行单价，由属性Per\_Price\_Now定义。*m*即为从TAXI类中所读取得到的里程数。

计价器的参数，在对象生成时自动从历史数据中读取，程序系统目前采用直接赋值的方式进行实现，此处可以进一步拓展从文件或存储器中读取历史参数。

4.5状态切换

计价器工作在不同情况下时，接受操作所执行响应也应有所区别。借鉴自动机的思想，定义计价器的如下几种工作状态，并设定状态之间的转移规则。

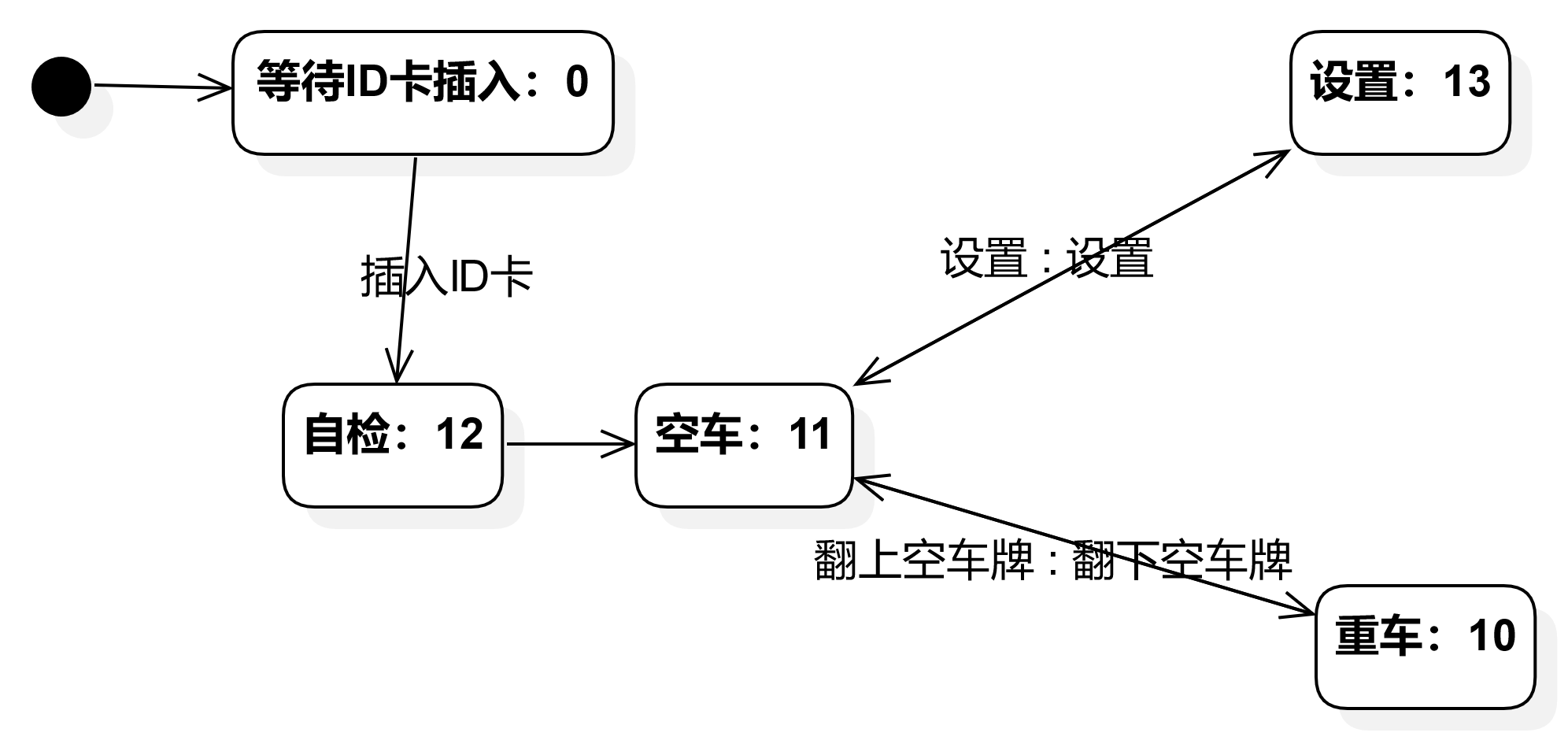
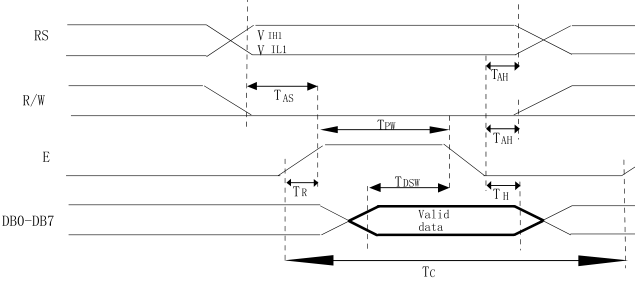


图 10 计价器状态图

定义计价器的工作状态有，等待ID卡（state:0），自检（state:12），空车（state:11），设置（state:13），重车（state:10）。所有功能按键中，加速、减速按键不受影响，在任何时候都能得到响应，并执行相应的动作。其余功能键（插卡、设置、启停）在计价器的不同状态有不同的功能受限。如图 10所示，等待插卡时，只接受插卡动作；空车时，所有按键正常响应，并转入相应状态；设置状态时，只响应设置操作回到空车界面；重车状态时，只接受停止操作，单次服务停止并回到空车界面。针对重车状态的启停，单独在计价器类中再设旅程开始与旅程结束函数，将重车状态切换时各项功能组合在一起。

4.6界面显示与刷新

液晶屏是系统的主要显示途径，界面设计以美观稳定为主要考虑因素。

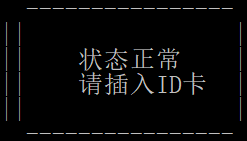
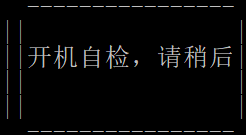
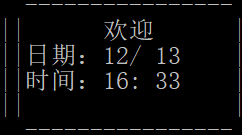
(a)显示屏接线示意图 (b)LCD写资料到模块时序图

图 11 显示屏接线与写资料时序示意图

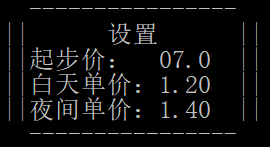
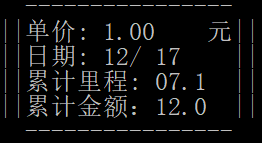
图 11中展示了液晶屏的连线示意图，以及其向写资料到模块时的时序图[6]，可以看到在E端口的一个高电平区间，指令数据得以写入。因而稳定时间段，E端口保持低电平，在需要写入指令或数据时，给高电平，再回到稳定的低电平。

**界面显示**

为了与计价器的5种状态相匹配，液晶屏的主显示界面也设计了对应的5种界面，如图 12所示。其中，图时钟信息的显示，由C语言获取系统时钟后传入参数，并将数字转为对应的显示字符编码，进行实时时钟日期的显示。

(a)等待插入ID卡 (b)自检 (c)空车

(d)设置 (e)重车

图 12 液晶屏界面显示设计

**界面刷新**

计价器的界面中，有不少信息是动态变化的，如时钟信息，里程信息，需要不断对其进行刷新。考虑到显示效果的稳定性，故选择针对界面进行最小更新，而不是整屏更新。当系统工作在查询方式下时，每次查询的最后，若当次无操作发生，则根据此时的计价器状态，进行界面的特定更新。

在里程更新时，传入参数为保留至小数点后一位的里程数和服务费，同样需将数字转为对应的显示字符编码，在特定位置予以输出。

4.8打印模块（喇叭发声模块）

单次运行结束时，即空车牌翻下时，发出“嗒嗒嗒”声模拟系统自动执行打印操作。其中，8254通道2工作在方式3发出特定频率声音，8255的PB7端口作为8254通道2的门控信号，使音频有所间断。其中，8254工作在方式2时，当门控信号为低电平时，输出始终为高电平，为了使门控信号与输出信号一致，将输出端外接逻辑非门。整体线路连接，见图 13。



图 13 8254连线示意图

发声函数封装在M\_8254类中，当单次运营服务结束时，受到计价器类中旅程结束函数（Journey\_End()）函数的调用，完成打印声模拟。

4.9中断接口

程序使用查询方式扫描键盘执行任务效率较低，此外计数初值的重装与阈值设定相关有不小的误差。引入中断操作改进系统，可以省去键盘键值判断且能够实现计数初值的中断重装，大大提高系统的准确度，灵敏度。



图 14 8259改进程序接线示意图

由实验指导书[3]知，实验箱在PC/AT机中提供了主/从两片8259，但开放的中断源端口仅两个，不能满足需求，故选用扩展8259进行中断接口的实现。在C语言编程环境下，使用查询中断查询字的方式来进行中断程序跳转控制。

图 14显示了使用中断改进程序的示意图，其中最高级中断响应赋给计数初值的重置，进而当8254计数结束时即刻进行初值重装；第1级中断赋给ID卡插入键，在系统初始状态时接受操作；第2、3级中断为车辆加减速操作，不受计价器系统状态的影响；第4级中断赋给设置键；第5级中断赋给空车牌翻动操作（单脉冲实现）；第6级中断赋给8254控制下的定时器（由通道1实现），实现程序的定时界面刷新。

4.10键值读取

在查询式工作方式下，系统通过拉低键盘行线，读取列线来判断所按下的键值。利用C语言的优势，读取列线时可以直接得知是那一列被按下（值非1），再通过逐行拉低即可确认键值[7]。详细处理流程见图 15。



图 15 键值确认流程图

键值确认程序封装在键盘类（KeyBoard类）中，并对外提供调用接口检查是否有按键按下，以及具体是哪一个按键。在系统工作过程中，由计价器主类调用键盘接口，并对接状态转移模块，实现控制。

4.11录放音模块

AD0809外接扬声器以固定时间间隔采集语音信号，并存储至文件中。当计价器执行到相应状态时，读取文件，并交DAC0832外接输出相应语音。录音与播放皆封装在M\_ADC0809类中。实际设计中，通过一小段延时给0809足够时间转换，代替查询EOC。



图 16 AD0809接线示意图

AD0809的接线情况如所示，参考资料后选用的采样频率为6000Hz[5]。

# 5.系统编码测试

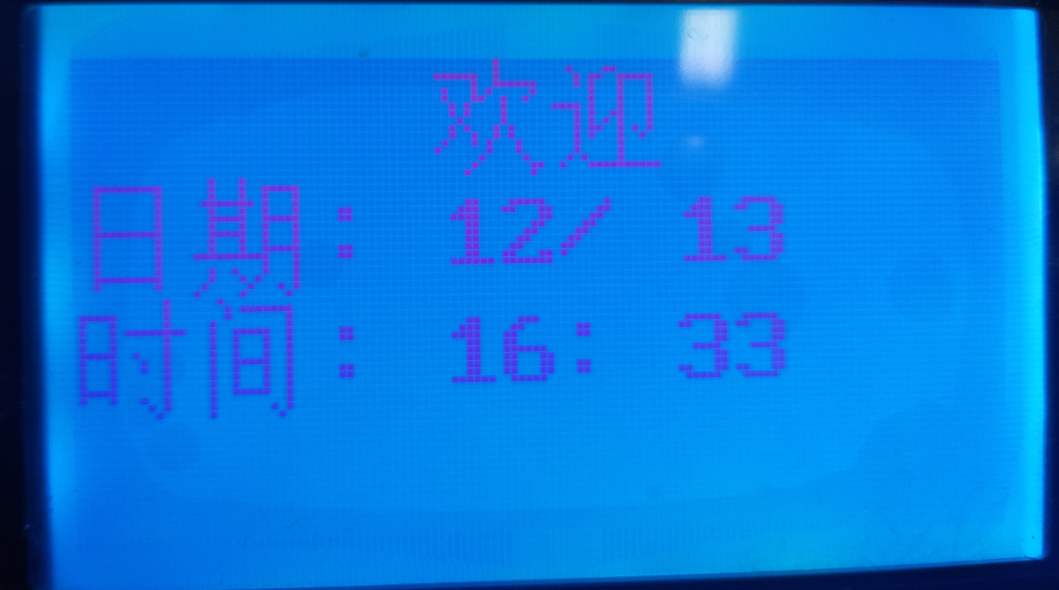
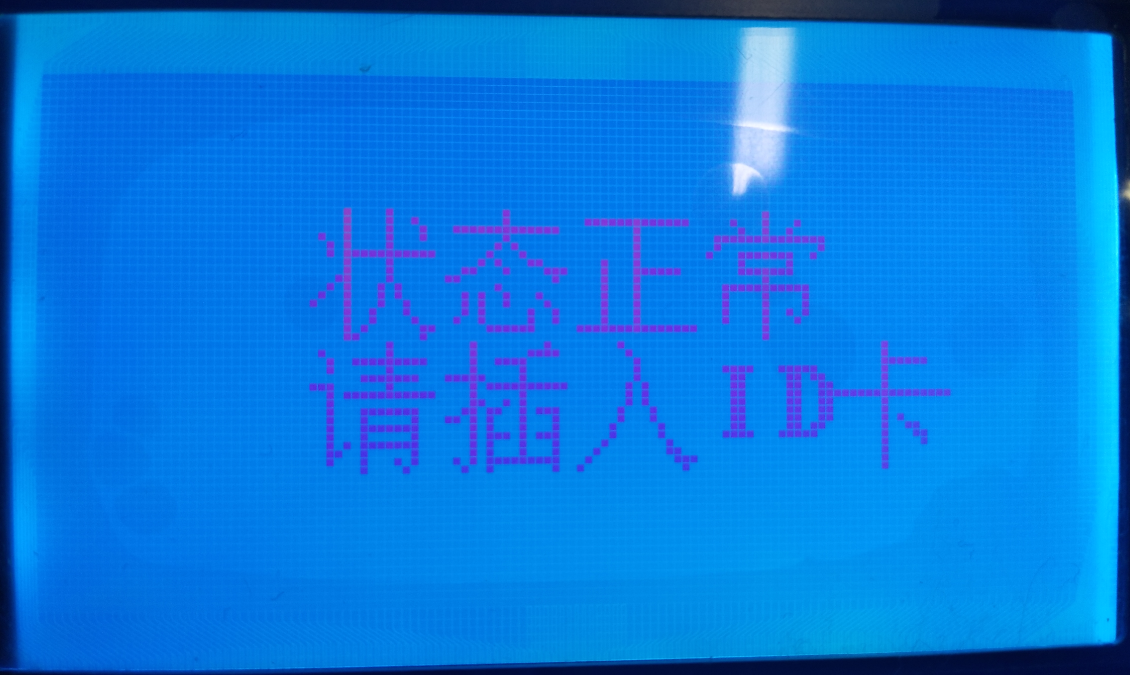
5.1单元测试（类测试）

逐一编写各个模块类，并进行测试。在模块类的关键方法中嵌入控制台输出，进而即使没有硬件动作也能得知C程序的执行时序。各类模块详见附录，单元测试用例如表格 1所示。

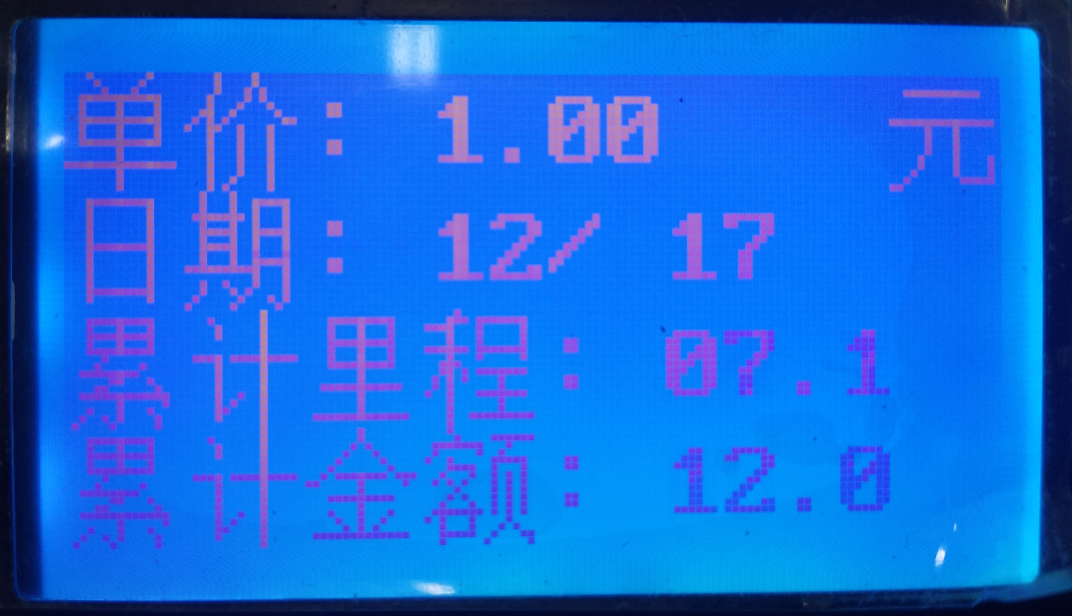
表格 1 单元测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 测试类 | 测试内容 | 测试数据或方法 | 期望结果 | 测试结果 |
| 1 | DAC0832类 | 电机速度更新 | 0 | 电机停转 | ✓ |
| 2 | 200 | 电机高速转动 | ✓ |
| 3 | M\_rgy类（8\*8点阵类） | 空字显示 | 循环调用Print\_Kong() | 空字稳定显示 | 略带闪动， 总体可以✓ |
| 4 | 关闭显示 | Print\_Null() | 停止显示 | ✓ |
| 5 | M\_8254类 | 计数 | 拨动电机 | 可以记录每转 | ✓ |
| 6 | 喇叭声 | Buzzer\_Call() | 发出“嗒嗒嗒” | ✓ |
| 7 | M\_8255类 | 初始化 | 生成对象 | 构造函数正常 | ✓ |
| 8 | KeyBoard类 | 检测按键按下 | 循环Is\_Press() | 识别按键按下 | ✓ |
| 9 | 键值读取 | Read() | 识别键值 | ✓ |
| 10 | LCD类 | 显示 | 等待ID卡界面 | 正常显示 | ✓ |
| 11 | 自检界面 | 正常显示 | ✓ |
| 12 | 空车界面 | 时钟获取正常 | ✓ |
| 13 | 重车界面 | 正常显示 | ✓ |
| 14 | 设置界面 | 正常显示 | ✓ |
| 15 | 更新 | 更新空车界面 | 更新正确 | ✓ |
| 16 | 更新重车界面 | 更新正确 | ✓ |
| 17 | TAXI类（车辆类） | 加速 | Start\_or\_SpeedUp() | 电机转速加快 | ✓ |
| 18 | 减速 | Stop\_or\_SpeedDown() | 电机转速减慢 | ✓ |
| 19 | 读取里程数 | Read\_Meter\_Thistime() | 读取正常 | ✓ |
| 20 | M\_8259类 | 中断查询 | Wait\_Interrupt() | 识别中断请求 | ✓ |
| 21 | M\_ADC0809类（录放音类） | 录音 | Record() | 录音生成文件 | ✓ |
| 22 | 放音 | Play() | 播放录音 | ✓ |

LCD类为系统主要可视化界面类，在单元测试中的测试截图如图 17所示。



(a)等待插入ID卡界面 (b)空车界面



(c)重车界面 (d)设置界面

图 17 液晶屏界面显示实物图

5.2集成测试（系统测试）

在各模块实现的基础上，编写TAXIMETER类（计价器主类），整合各模块，并测试系统，整个系统测试用例见表格 2。

表格 2 系统测试用例（状态转移测试）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 计价器状态 | 测试操作 | 期望结果 | 测试结果 |
| 1 | 等待插入ID | ID键 | 转入自检状态 | ✓ |
| 2 | 加速、减速 | 正常响应 | ✓ |
| 3 | 设置、启停 | 不响应 | ✓ |
| 4 | 自检 | 任意按键 | 不响应 | ✓ |
| 5 | 等待3秒 | 转入空车状态 | ✓ |
| 6 | 空车 | 加速、减速 | 正常响应 | ✓ |
| 7 | 设置 | 转入设置界面 | ✓ |
| 8 | ID键 | 不响应 | ✓ |
| 9 | 启停 | 转入重车，点阵灭 | ✓ |
| 10 | 等待 | 自动刷新时钟、点阵 | ✓ |
| 11 | 重车 | 加速、减速 | 正常响应 | ✓ |
| 12 | 设置、ID | 不响应 | ✓ |
| 13 | 启停 | 转入空车，点阵亮喇叭响 | ✓ |
| 14 | 等待 | 刷新里程与价格 | ✓ |
| 15 | 设置 | 加速、减速 | 正常响应 | ✓ |
| 16 | 设置 | 转入空车 | ✓ |
| 17 | 其他按键 | 不响应 | ✓ |
| 18 | 等待 | 刷新点阵 | ✓ |

整体系统演示视频如下：<https://v.youku.com/v_show/id_XNDQ3NTg3ODQwMA==.html?spm=a2h3j.8428770.3416059.1>

# 6.设计结果分析及结论

在系统的运行测试过程中，有不少与设计或实际相左的结果，其中有一些在设计阶段已经估计到了，还有一些经过分析之后，也得到了结论。

6.1计数初值偏离

当计价器转入重车状态开始计价时，若此时计数器已经开始了工作，且电机已经开始了转动，则初值不是0。而程序中转入重车状态后第一次计量里程时，保留的上次计数值temp为0，这里会有一个由初值造成的偏移误差，范围在(0~65535)。

6.2按键灵敏度降低

系统工作在查询式下时，设定消颤时长为50ms，且当程序工作在其他步骤时，可能遗漏查询响应。实际测试，当手指按键的时长适当提高时，有助于识别。

# 7.实验体会

《硬件课程设计》**收获颇丰**：一是对于上学期所学理论课程有了非常全面的复习；二是对于完整系统软件的设计实现能力有了很大增长；三是结合同期所学《软件工程》，在设计与分析系统，撰写报告陈述思路方面得到了很好的实践；四是查阅资料，尤其是阅读硬件芯片手册的能力也有了很大的提升。

受限于课程的时间，回顾整个48学时，觉得仍然有一小部分的**遗憾** ，没有能够去实际的检验利用中断实现8254自动重装计数初值；没能够实现将录音模块嵌入主系统中，实现语音播报；时钟获取也可以考虑利用实验仪上PC机自动获取；设想的日志模块尚未实现……

不过总体上，仍然十分难忘整个过程。

# 参考文献

1. ST7920GB中文字型码表. (2000年4月3日).
2. TPC-2003A通用32位微机实验系统学生用实验指导书. (2004年10月).
3. TPC-ZK系列USB教师实验指导书. (2017年9月25日). 检索来源: 原创力文档: https://max.book118.com/html/2017/0925/134943357.shtm
4. 陈楠. (2014年12月15日). TPC-ZK-II实验指导书.
5. 接口技术综合性实验报告-数字录音机. (2016年9月3日). 检索来源: 百度文库: https://wenku.baidu.com/view/fc8137da4b73f242326c5feb
6. 深圳亚斌显示科技有限公司. (无日期). 中文字库液晶显示模块使用手册. 深圳.
7. 周荷琴, & 冯焕清. (2014). 微型计算机原理与接口技术 (第 5 版 版本). 合肥: 中国科学技术大学出版社.

# 附录

代码 1 M\_rgy类（8\*8点阵类）

|  |  |
| --- | --- |
| 1 | #define port\_HCS 0x2B8 |
| 2 | #define port\_RCS 0x2B0 |
| **3** | **struct** M\_rgy **{** |
| 4 | **int** port\_h**;** |
| 5 | **int** port\_r**;** |
| 6 | **int** Sleep\_time**;** |
| 7 | M\_rgy**():**port\_h**(**port\_HCS**),**port\_r**(**port\_RCS**),**Sleep\_time**(**10**)** **{}** |
| 8 | **void** Print\_Kong**()** **{** |
| 9 | byte Lie**[**8**]=** **{**0x04**,**0xef**,**0x55**,**0x22**,**0x1c**,**0x08**,**0x08**,**0x3e**};** |
| 10 | byte h**=**0x01**;** |
| 11 | **for(int** i**=**0**;** i**<**8**;** **++**i**)** **{** |
| 12 | PortWriteByte**(**port\_r**,**Lie**[**i**]);** |
| 13 | PortWriteByte**(**port\_h**,**h**);** |
| 14 | h**=**h**<<**1**;** |
| 15 | Sleep**(**Sleep\_time**);** |
| 16 | **}** |
| 17 | **}** |
| 18 | **void** Print\_Null**()** **{** |
| 19 | PortWriteByte**(**port\_h**,**0x00**);** |
| 20 | **}** |
| **21** | **};** |

代码 2 M\_8254类

|  |  |
| --- | --- |
|  | #define port\_8254 0x298 |
|  | **struct** M\_8254 **{** |
|  | **int** port\_ctr**;** |
|  | **int** result**;** |
|  | M\_8254**():**port\_ctr**(**port\_8254**+**3**)** **{** |
|  | Init\_Gate0**();** |
|  | Init\_Gate2**();** |
|  | result**=**0**;** |
|  | **}** |
|  | **void** Init\_Gate0**()** **{** *//通道0，此段计数初值可供之后中断调用* |
|  | PortWriteByte**(**port\_ctr**,**0x30**);***//控制字：00 11 000 0通道0，二进制计数* |
|  | PortWriteByte**(**port\_ctr**-**3**,**0x00**);** |
|  | PortWriteByte**(**port\_ctr**-**3**,**0x00**);** |
|  | cout**<<"8254 init, Gate 0 chosed"<<**endl**;** |
|  | **}** |
|  | **void** Init\_Gate1**()** **{** *//通道1，此段硬延时产生一个刷新屏幕的频率* |
|  | PortWriteByte**(**port\_ctr**,**0x77**);***//控制字：01 11 011 1通道1，10进制计数* |
|  | PortWriteByte**(**port\_ctr**-**2**,**0x00**);** |
|  | PortWriteByte**(**port\_ctr**-**2**,**0x00**);** |
|  | cout**<<"8254 init, Gate 1 chosed"<<**endl**;** |
|  | **}** |
|  | **void** Init\_Gate2**()** **{** *//通道2，用于给蜂鸣器输出打印声（未完全）* |
|  | PortWriteByte**(**port\_ctr**,**0xB7**);***//控制字：10 11 011 1通道2，方式3，BCD计数* |
|  | PortWriteByte**(**port\_ctr**-**1**,**0xe9**);** |
|  | PortWriteByte**(**port\_ctr**-**1**,**0xf1**);** |
|  | cout**<<"8254 init, Gate 2 chosed"<<**endl**;** |
|  | PortWriteByte**(**0x289**,**0x00**);***//默认为不发声；* |
|  | **}** |
|  | **void** Buzzer\_Call**()** **{** |
|  | *//8255 B7连Gate2，out连非门再输出（因为8254工作在方式3时，Gate为0默认为不计数输出恒高）* |
|  | cout**<<"Buzzer\_Call init"<<**endl**;** |
|  | **for(int** i**=**0**;** i**<**8**;** **++**i**)** **{** |
|  | PortWriteByte**(**0x289**,**0x80**);** |
|  | Sleep**(**150**);***//对应发声时间* |
|  | cout**<<"end:"<<**endl**;** |
|  | PortWriteByte**(**0x289**,**0x00**);** |
|  | Sleep**(**30**);***//对应的是沉默时间* |
|  | **}** |
|  | cout**<<"Buzzer\_Call ended"<<**endl**;** |
|  | **}** |
|  | **int** read**()** **{** |
|  | PortWriteByte**(**port\_ctr**,**0x00**);***//控制字：00 00 000 0通道0锁存* |
|  | byte L**;** |
|  | byte H**;** |
|  | PortReadByte**(**port\_ctr**-**3**,&**L**);** |
|  | PortReadByte**(**port\_ctr**-**3**,&**H**);** |
|  | result**=**65536**-(**256**\***H**+**L**);** |
|  | cout**<<"8254计数补值为："<<**result**<<**endl**;** |
|  | **if(**result**>**60000**)**Init\_Gate0**();** |
|  | **return** result**;** |
|  | **}** |
|  | **};** |

代码 3 M\_8255类

|  |  |
| --- | --- |
|  | #define port\_8255 0x288 |
|  | **struct** M\_8255 **{** |
|  | **int** port**;** |
|  | M\_8255**():**port**(**port\_8255**)** **{** |
|  | PortWriteByte**(**port**+**3**,**0x81**);** |
|  | **}** |
|  | **};** |

代码 4 KeyBoard类

|  |  |
| --- | --- |
|  | #define port\_8255\_C 0x28A |
|  | **struct** KeyBoard **{** |
|  | */\*\*键盘工作流程* |
|  | *\*1. 当行线都拉低时，正常列线应该全为1* |
|  | *\*2. 若出现非1，说明改行按键被按下了，改为逐行拉低去检测是哪一行出现非1* |
|  | *\*3. 继续拉低该行，继续去检测是哪一列非1，在c中，哪一列可以直接计算得到* |
|  | *\*\*/* |
|  | **int** port\_c**;** |
|  | **int** result**;** |
|  | **int** row**;** |
|  | **int** col**;** |
|  | **int** ret**;***//返回按键数值* |
|  | byte condition**;** |
|  | KeyBoard**():**port\_c**(**port\_8255**+**2**)** **{** |
|  | init**();** |
|  | cout**<<"keyboard started."<<**endl**;** |
|  | **}** |
|  | **void** init**()** **{** |
|  | PortWriteByte**(**port\_c**+**1**,**0x81**);***//控制字：1 00 0 0 00 1* |
|  | **}** |
|  | **bool** IsPRES**()** **{** |
|  | PortWriteByte**(**port\_c**,**0x00**);***//拉低所有行* |
|  | PortReadByte**(**port\_c**,&**condition**);** |
|  | **if((int)**condition**%**16**==**15**)return** **false;** |
|  | **else** **{** |
|  | *//先引入一步“消颤”，根据实验，* |
|  | *//没有这一步会在一次按下中，多次识别* |
|  | Sleep**(**50**);** |
|  | PortReadByte**(**port\_c**,&**condition**);** |
|  | **if((int)**condition**%**16**==**15**)** **{** |
|  | **return** **false;** |
|  | **}** |
|  | *//change();* |
|  | *//cout<<"PRES!"<<endl;* |
|  | *//read();* |
|  | **return** **true;** |
|  | **}** |
|  | **}** |
|  | **void** read**()** **{** |
|  | **int** a**=**15**-**condition**%**15**;** |
|  | col**=**0**;** |
|  | **while(**a**!=**0**)** **{** |
|  | a**/=**2**;** |
|  | col**++;** |
|  | **}** |
|  | *//逐行拉低，读取非1时的该行，同时去计算是哪一列被按下了* |
|  | byte line**=**0xEF**;** |
|  | row**=**0**;** |
|  | **do** **{** |
|  | PortWriteByte**(**port\_c**,**line**);** |
|  | PortReadByte**(**port\_c**,&**condition**);** |
|  | **++**row**;** |
|  | line**=**line**<<**1**;** |
|  | **}** **while((int)**condition**%**16**==**15**);** |
|  |  |
|  | **if(**col**==**1**)** **{** |
|  | **if(**row**==**1**)**ret**=**1**;** |
|  | **else** **if(**row**==**2**)**ret**=**2**;** |
|  | **else** **if(**row**==**3**)**ret**=**3**;** |
|  | **}** **else** **if(**col**==**2**)** **{** |
|  | **if(**row**==**1**)**ret**=**4**;** |
|  | **else** **if(**row**==**2**)**ret**=**5**;** |
|  | **else** **if(**row**==**3**)**ret**=**6**;** |
|  | **}** |
|  | cout**<<"the board is "<<**row**<<" "<<**col**<<", ret is "<<**ret**<<**endl**;** |
|  | **}** |
|  | **int** Pressed\_Button**()** **{** |
|  | *//这个函数作为查询式的一个接口函数，始终等待按键按下，一旦按下返回内置的键值* |
|  | **while(!**IsPRES**());** |
|  | read**();** |
|  | cout**<<"Button "<<**ret**<<" is Pressed."<<**endl**;** |
|  | **return** ret**;** |
|  | **}** |
|  | **};** |

代码 5 LCD类

|  |  |
| --- | --- |
|  | #include"time.h" |
|  | #define port\_74LS273 0x290 | |
|  | **struct** LCD**:public** M\_8255 **{** |
|  | **int** port\_data**;** |
|  | **int** port\_ctr**;** |
|  | **int** Addr\_start**;** |
|  | **int** Youbiao\_isON**;** |
|  | LCD**():**M\_8255**(),**port\_data**(**port\_74LS273**),**port\_ctr**(**port\_8255**+**1**),**Addr\_start**(**128**)** **,**Youbiao\_isON**(**0**)** **{};** |
|  | **void** cmd\_setup**()** **{** |
|  | PortWriteByte**(**port\_ctr**,**0x00**);** |
|  | PortWriteByte**(**port\_ctr**,**0x04**);** |
|  | PortWriteByte**(**port\_ctr**,**0x01**);** |
|  | **}** |
|  | **void** Clear\_All**()** **{** |
|  | PortWriteByte**(**port\_data**,**0x01**);** |
|  | cmd\_setup**();** |
|  | **}** |
|  | **void** data\_setup**()** **{** |
|  | PortWriteByte**(**port\_ctr**,**0x01**);** |
|  | PortWriteByte**(**port\_ctr**,**0x05**);** |
|  | PortWriteByte**(**port\_ctr**,**0x01**);** |
|  | **}** |
|  | **void** Display\_Line**(int** n**,int** data**[**8**])** **{** |
|  | **switch(**n**)** **{** |
|  | **case** 1**:** |
|  | Addr\_start**=**128**;** |
|  | **break;** |
|  | **case** 2**:** |
|  | Addr\_start**=**144**;** |
|  | **break;** |
|  | **case** 3**:** |
|  | Addr\_start**=**136**;** |
|  | **break;** |
|  | **default:** |
|  | Addr\_start**=**152**;** |
|  | **}** |
|  | **for(int** i**=**0**;** i**<**8**;** i**++)** **{** |
|  | PortWriteByte**(**port\_data**,**Addr\_start**);** |
|  | cmd\_setup**();** |
|  | PortWriteByte**(**port\_data**,**data**[**i**]>>**8**);** |
|  | data\_setup**();** |
|  | PortWriteByte**(**port\_data**,**data**[**i**]%**256**);** |
|  | data\_setup**();** |
|  | Addr\_start**++;** |
|  | **}** |
|  | **}** |
|  | **void** Display\_Update\_Car\_Empty**()** **{** |
|  | **struct** tm **\***local**;** |
|  | time\_t t**=**time**(**NULL**);** |
|  | local**=**localtime**(&**t**);** |
|  | cout**<<"Date: "<<**local**->**tm\_mon**<<" / "<<**local**->**tm\_mday**<<**endl**;** |
|  | cout**<<"Time: "<<**local**->**tm\_hour**<<" : "<<**local**->**tm\_min**<<**endl**;** |
|  |  |
|  | **int** B\_mon**,**B\_mday**,**B\_hour**,**B\_min**;** |
|  | B\_mon**=**0x3030**+((**local**->**tm\_mon**+**1**)/**10**)\***256**+(**local**->**tm\_mon**+**1**)%**10**;** |
|  | B\_mday**=**0x3030**+(**local**->**tm\_mday**/**10**)\***256**+**local**->**tm\_mday**%**10**;** |
|  | B\_hour**=**0x3030**+(**local**->**tm\_hour**/**10**)\***256**+**local**->**tm\_hour**%**10**;** |
|  | B\_min**=**0x3030**+(**local**->**tm\_min**/**10**)\***256**+**local**->**tm\_min**%**10**;** |
|  | Display\_Update\_ch**(**2**,**3**,**B\_mon**);** |
|  | Display\_Update\_ch**(**2**,**5**,**B\_mday**);** |
|  | Display\_Update\_ch**(**3**,**3**,**B\_hour**);** |
|  | Display\_Update\_ch**(**3**,**5**,**B\_min**);** |
|  | **}** |
|  | **void** Display\_Update\_ch**(int** n**,int** r**,int** data**)** **{** |
|  | **switch(**n**)** **{** |
|  | **case** 1**:** |
|  | Addr\_start**=**128**;** |
|  | **break;** |
|  | **case** 2**:** |
|  | Addr\_start**=**144**;** |
|  | **break;** |
|  | **case** 3**:** |
|  | Addr\_start**=**136**;** |
|  | **break;** |
|  | **default:** |
|  | Addr\_start**=**152**;** |
|  | **}** |
|  | PortWriteByte**(**port\_data**,**Addr\_start**+**r**);** |
|  | cmd\_setup**();** |
|  | PortWriteByte**(**port\_data**,**data**>>**8**);** |
|  | data\_setup**();** |
|  | PortWriteByte**(**port\_data**,**data**%**256**);** |
|  | data\_setup**();** |
|  | **}** |
|  | **void** Display\_Youbiao\_ON**()** **{***//显示游标* |
|  | PortWriteByte**(**port\_data**,**0x0F**);** |
|  | cmd\_setup**();** |
|  | **}** |
|  | **void** Display\_Youbiao\_OFF**()** **{** |
|  | PortWriteByte**(**port\_data**,**0x0C**);** |
|  | cmd\_setup**();** |
|  | **}** |
|  | **void** Display\_Self\_Check**()** **{** |
|  | *//* |
|  | *//开机自检，请稍候* |
|  | *//* |
|  | *//* |
|  | Clear\_All**();** |
|  | **int** Data\_Self\_Check**[**8**]=** **{**0xBFAA**,**0xBBFA**,**0xD7D4**,**0xBCEC**,**0xA3AC**,**0xC7EB**,**0xC9D4**,**0xBAF2**};** |
|  | Display\_Line**(**2**,**Data\_Self\_Check**);** |
|  | **}** |
|  | **void** Display\_Wait\_ICcard**()** **{** |
|  | *//* |
|  | *//\_\_ \_\_ 状态正常 \_\_ \_\_* |
|  | *//\_\_ \_\_ 请插入IC 卡 \_\_* |
|  | *//* |
|  | Clear\_All**();** |
|  | **int** Data\_Condition\_OK**[**8**]=** **{**0xA1A0**,**0xA1A0**,**0xD7B4**,**0xCCAC**,**0xD5FD**,**0xB3A3**,**0xA1A0**,**0xA1A0**};** |
|  | **int** Data\_Wait\_ICcard**[**8**]=** **{**0xA1A0**,**0xA1A0**,**0xC7EB**,**0xB2E5**,**0xC8EB**,**0x4944**,**0xBFA8**,**0xA1A0**};** |
|  | Display\_Line**(**2**,**Data\_Condition\_OK**);** |
|  | Display\_Line**(**3**,**Data\_Wait\_ICcard**);** |
|  | **}** |
|  | **void** Display\_Setup**()** **{** |
|  | *//\_\_ \_\_ \_\_ 设 置 \_\_ \_\_ \_\_* |
|  | *//起 步 价 :\_ \_\_ 07 .0 \_\_* |
|  | *//白 天 单 价 :\_ 1. 20 \_\_* |
|  | *//夜 间 单 价 :\_ 1. 40 \_\_* |
|  | **int** Data\_Setup**[**8**]=** **{**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xC9E8**,**0xD6C3**,**0xA1A0**,**0xA1A0**,**0xA1A0**};** |
|  | **int** Data\_Starting\_Fare**[**8**]=** **{**0xC6F0**,**0xB2BD**,**0xBCDB**,**0x3A00**,**0xA1A0**,**0x3037**,**0x2E30**,**0xA1A0**};** |
|  | **int** Data\_PerPrice\_Day**[**8**]=** **{**0xB0D7**,**0xCCEC**,**0xB5A5**,**0xBCDB**,**0x3A00**,**0x312E**,**0x3230**,**0xA1A0**};** |
|  | **int** Data\_PerPrice\_Night**[**8**]=** **{**0xD2B9**,**0xBCE4**,**0xB5A5**,**0xBCDB**,**0x3A00**,**0x312E**,**0x3430**,**0xA1A0**};** |
|  | Display\_Line**(**1**,**Data\_Setup**);** |
|  | Display\_Line**(**2**,**Data\_Starting\_Fare**);** |
|  | Display\_Line**(**3**,**Data\_PerPrice\_Day**);** |
|  | Display\_Line**(**4**,**Data\_PerPrice\_Night**);** |
|  | **}** |
|  | **void** Display\_Car\_Empty**()** **{** |
|  | *//\_\_ \_\_ \_\_ 欢 迎 \_\_ \_\_ \_\_* |
|  | *//日 期 :\_ 12 /\_ 09 \_\_ \_\_* |
|  | *//时 间 :\_ 15 :\_ 30 \_\_ \_\_* |
|  | *//* |
|  | **int** Data\_Welcome**[**8**]=** **{**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xBBB6**,**0xD3AD**,**0xA1A0**,**0xA1A0**,**0xA1A0**};** |
|  | **int** Data\_Date**[**8**]=** **{**0xC8D5**,**0xC6DA**,**0x3A00**,**0x3132**,**0x2F00**,**0x3039**,**0xA1A0**,**0xA1A0**};** |
|  | **int** Data\_Time**[**8**]=** **{**0xCAB1**,**0xBCE4**,**0x3A00**,**0x3135**,**0x3A00**,**0x3330**,**0xA1A0**,**0xA1A0**};** |
|  | **int** Blank**[**8**]=** **{**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xA1A0**,**0xA1A0**};** |
|  | Display\_Line**(**1**,**Data\_Welcome**);** |
|  | Display\_Line**(**2**,**Data\_Date**);** |
|  | Display\_Line**(**3**,**Data\_Time**);** |
|  | Display\_Line**(**4**,**Blank**);** |
|  |  |
|  | Display\_Update\_Car\_Empty**();** |
|  | **}** |
|  | **void** Display\_Update\_Car\_Full**(double** Mileage\_Thistime**,double** Price\_Thistime**)** **{** |
|  | **int** Meter\_0**,**Meter\_1**;** |
|  | Meter\_0**=**0x2E30**+(int)(**Mileage\_Thistime**\***10**)%**10**;** |
|  | Meter\_1**=**0x3030**+(int)**Mileage\_Thistime**/**10**\***256**+((int)**Mileage\_Thistime**\***10**%**100**/**10**);** |
|  | Display\_Update\_ch**(**3**,**6**,**Meter\_0**);** |
|  | Display\_Update\_ch**(**3**,**5**,**Meter\_1**);** |
|  |  |
|  | **int** Money\_0**,**Money\_1**;** |
|  | Money\_0**=**0x2E30**+(int)(**Price\_Thistime**\***10**)%**10**;** |
|  | Money\_1**=**0x3030**+(int)**Price\_Thistime**/**10**\***256**+((int)**Price\_Thistime**\***10**%**100**/**10**);** |
|  | Display\_Update\_ch**(**4**,**6**,**Money\_0**);** |
|  | Display\_Update\_ch**(**4**,**5**,**Money\_1**);** |
|  | **}** |
|  | **void** Display\_Car\_Full**(double** Per\_Price**)** **{** |
|  | *//单价 :\_ 1. 20 \_\_ \_\_ 元 如：7.0* |
|  | *//时间 :\_ 00 :\_ 23 \_\_ \_\_ 如：1:23 //行驶时间暂时去掉，优先实现完备系统，刚需为里程与计价* |
|  | *//累计 里 程 :\_ \_\_ .\_ \_\_* |
|  | *//累计 金 额 :\_ \_\_ .\_ \_\_* |
|  | **int** Data\_Per\_Price**[**8**]=** **{**0xB5A5**,**0xBCDB**,**0x3A00**,**0x312E**,**0x3230**,**0xA1A0**,**0xA1A0**,**0xD4AA**};** |
|  | *//int Data\_Total\_Time[8]= {0xCAB1,0xBCE4,0x3A00,0x3030,0x3A00,0x3031,0xA1A0,0xA1A0};* |
|  | **int** Data\_Total\_Meter**[**8**]=** **{**0xC0DB**,**0xBCC6**,**0xC0EF**,**0xB3CC**,**0x3A00**,**0x3030**,**0x2E30**,**0xA1A0**};** |
|  | **int** Data\_Total\_Money**[**8**]=** **{**0xC0DB**,**0xBCC6**,**0xBDF0**,**0xB6EE**,**0x3A00**,**0x3037**,**0x2E30**,**0xA1A0**};** |
|  | Display\_Line**(**1**,**Data\_Per\_Price**);** |
|  | *//Display\_Line(2,Data\_Total\_Time);* |
|  | Display\_Line**(**3**,**Data\_Total\_Meter**);** |
|  | Display\_Line**(**4**,**Data\_Total\_Money**);** |
|  | **int** Per\_p\_1**=**0x302E**+(int)**Per\_Price**\***256**;** |
|  | **int** Per\_p\_0**=**0x3030**+(int)**10**\*(**Per\_Price**-(int)**Per\_Price**)\***256**;** |
|  | Display\_Update\_ch**(**1**,**3**,**Per\_p\_1**);** |
|  | Display\_Update\_ch**(**1**,**4**,**Per\_p\_0**);** |
|  | **}** |
|  | **};** |

代码 6 DAC0832类

|  |  |
| --- | --- |
|  | #define port\_DAC0832 0x280 |
|  | **struct** M\_DAC0832 **{** |
|  | **int** port\_ctr**;** |
|  | **int** speed**;** |
|  | M\_DAC0832**():**port\_ctr**(**port\_DAC0832**),**speed**(**0**)** **{** |
|  | PortWriteByte**(**port\_ctr**,**speed**);** |
|  | cout**<<"M\_0832 init, Speed\_0 = 0"<<**endl**;** |
|  | **}** |
|  | **void** Speed\_Update**(int** sp**)** **{** |
|  | PortWriteByte**(**port\_ctr**,**sp**);** |
|  | cout**<<"Speed update, Speed\_now = "<<**speed**<<**endl**;** |
|  | **}** |
|  | **};** |

代码 7 TAXI类（车辆类）

|  |  |
| --- | --- |
|  | **struct** TAXI**:public** M\_DAC0832**,**M\_8254 **{** |
|  | *//TAXI 类继承M\_DAC0832芯片,M\_8254芯片,在此类中完成里程数计算* |
|  | *//继承speed与加减速，同时自身带有数据成员state状态* |
|  | *//state:0-未启动，1-运行，10-静止等待（运行中，车速0）* |
|  | **int** state**;** |
|  | **double** Meter\_Total**;** |
|  | **double** Meter\_Thistime**;** |
|  | **double** temp**;** |
|  | **double** factor**;** |
|  | TAXI**():**M\_DAC0832**(),**state**(**0**),**Meter\_Thistime**(**0**),**Meter\_Total**(**0**),**temp**(**0**),**factor**(**1440**)** **{};** |
|  | **double** Read\_Meter\_Thistime**()** **{** |
|  | **int** Now**=this->**read**();** |
|  | *//8254计数是倒计数，这里在装入下一个计数值时会出现较大误差* |
|  | *//在出租车速度不快时没有问题，当速度达到五档时有很大误差。* |
|  | **if(**Now**>=**temp**)** **{** |
|  | Meter\_Thistime**+=(double)(**Now**-**temp**)/**factor**;** |
|  | temp**=**Now**;** |
|  | **}** **else** **{** |
|  | Meter\_Thistime**+=(double)(**Now**)/**factor**;** |
|  | temp**=**Now**;** |
|  | **}** |
|  | **return** Meter\_Thistime**;** |
|  | **}** |
|  | **double** Read\_Meter\_Total**()** **{** |
|  | **return** Meter\_Total**+**Meter\_Thistime**;** |
|  | **}** |
|  | **void** Thistime\_Start**()** **{** |
|  | Meter\_Thistime**=**0**;** |
|  | temp**=**0**;** |
|  | **}** |
|  | **int** Thistime\_End**()** **{** |
|  | Meter\_Total**+=**Read\_Meter\_Thistime**();** |
|  | cout**<<"本次运行结束，总里程为："<<**Meter\_Thistime**<<**endl**;** |
|  | **return** Meter\_Thistime**;** |
|  | **}** |
|  | **void** Start\_or\_SpeedUp**()** **{** |
|  | **if(**state**==**0**)** **{** |
|  | state**=**1**;***//汽车启动* |
|  | speed**=**90**;** |
|  | **this->**Speed\_Update**(**speed**);** |
|  | **return;** |
|  | **}** |
|  | **if(**speed**<**245**)**speed**+=**10**;** |
|  | **else** speed**=**254**;** |
|  | **this->**Speed\_Update**(**speed**);** |
|  | **if(**state**==**10**)** **{** |
|  | state**=**1**;** |
|  | cout**<<"Taxi continue runing"<<**endl**;** |
|  | **}** |
|  | **}** |
|  | **void** Stop\_or\_SpeedDown**()** **{** |
|  | **if(**state**==**10**)** **{** |
|  | state**=**0**;** |
|  | cout**<<"Taxi stop"<<**endl**;** |
|  | **return;** |
|  | **}** |
|  | **if(**speed**>**70**)**speed**-=**10**;** |
|  | **else** speed**=**61**;** |
|  | Speed\_Update**(**speed**);** |
|  | **if(**speed**==**61**)** **{** |
|  | state**=**10**;** |
|  | cout**<<"Taxi wait"<<**endl**;** |
|  | **}** |
|  | **}** |
|  | **};** |

代码 8 TAXIMETER类（计价器主类）

|  |  |
| --- | --- |
|  | **struct** TAXIMETER **{** |
|  | **int** state**;** |
|  | */\*\*state 代表计价器的状态* |
|  | *\*0 IC卡未插* |
|  | *\*10 重车状态* |
|  | *\*11 空车状态* |
|  | *\*12 自检状态* |
|  | *\*13 设置状态* |
|  | *\*14 计价暂停状态* |
|  | *\*\*/* |
|  | **double** Per\_Price\_Day**;** |
|  | **double** Per\_Price\_Night**;** |
|  | **double** Per\_Price\_Now**;** |
|  |  |
|  | **double** Starting\_Price**;** |
|  | **int** Total\_Times**;***//累积载客次数* |
|  | **int** Total\_Price**;** |
|  | *//单次营运中的一些参数* |
|  | **int** Money\_Thistime**;** |
|  | **int** Time\_Start**;** |
|  | **int** Time\_End**;** |
|  |  |
|  | LCD lcd**;** |
|  | M\_8255 m\_8255**;** |
|  | M\_8254 m\_8254**;** |
|  | KeyBoard kb**;** |
|  | TAXI taxi**;** |
|  | M\_rgy rgy**;** |
|  |  |
|  | TAXIMETER**():**state**(**0**)** **{** |
|  | History\_Read**();** |
|  | *//Wait\_IC();* |
|  | **};** |
|  | **~**TAXIMETER**()** **{** |
|  | *//析构函数中可以加“写硬盘”* |
|  | **}** |
|  | **void** History\_Read**()** **{** |
|  | *//此步可以使用存储器进行断电存储* |
|  | Per\_Price\_Day**=**1.2**;** |
|  | Per\_Price\_Night**=**1.4**;** |
|  | taxi**.**Meter\_Total**=**123**;** |
|  | Total\_Price**=**374**;** |
|  | Total\_Times**=**12**;** |
|  | Starting\_Price**=**7**;** |
|  | cout**<<"Read Log OK, "<<**Per\_Price\_Day**<<" "<<**Per\_Price\_Night**<<" "** |
|  | **<<**taxi**.**Meter\_Total**<<" "<<**Total\_Price**<<" "<<**Total\_Times**<<" "<<**endl**;** |
|  | **}** |
|  | **void** History\_Write**()** **{}***//写入参数，可以使用存储器扩充* |
|  | **double** Caculate\_Money\_Thistime**(double** meter**)** **{** |
|  | **double** M**;** |
|  | **if(**meter**<**3**)**M**=**7**;** |
|  | **else** **{** |
|  | M**=**Per\_Price\_Now**\*(**meter**-**3**)+**7**;** |
|  | M**\*=**10**;** |
|  | M**=(int)(**M**+**0.5**);** |
|  | M**=(double)**M**/**10**;***//精确到小数点后一位* |
|  | **}** |
|  | cout**<<"Meter: "<<**meter**<<",Money: "<<**M**<<**endl**;** |
|  | **return** M**;** |
|  | **}** |
|  | **void** Journey\_Start**()** **{** |
|  | state**=**10**;** |
|  | **struct** tm **\***local**;** |
|  | time\_t t**=**time**(**NULL**);** |
|  | local**=**localtime**(&**t**);** |
|  | **if(**local**->**tm\_hour**>**18**||**local**->**tm\_hour**<**6**)**Per\_Price\_Now**=**Per\_Price\_Night**;** |
|  | **else** Per\_Price\_Now**=**Per\_Price\_Day**;** |
|  | rgy**.**Print\_Null**();** |
|  | taxi**.**Thistime\_Start**();** |
|  | lcd**.**Display\_Car\_Full**(**Per\_Price\_Now**);** |
|  | **}** |
|  | **void** Journey\_End**()** **{** |
|  | state**=**11**;** |
|  | rgy**.**Print\_Kong**();** |
|  | taxi**.**Thistime\_End**();** |
|  | m\_8254**.**Buzzer\_Call**();** |
|  | Sleep**(**1000**);** |
|  | lcd**.**Display\_Car\_Empty**();** |
|  | **}** |
|  | **void** State\_Change**(int** ret**)** **{** |
|  | *//这段函数根据键盘操作执行状态之间的跳转代码，类似于事务中心的转移* |
|  | **switch(**state**)** **{** |
|  | **case** 11**:** **{** |
|  | *//空车* |
|  | **if(**ret**==**6**)**taxi**.**Start\_or\_SpeedUp**();** |
|  | **else** **if(**ret**==**3**)**taxi**.**Stop\_or\_SpeedDown**();** |
|  | **else** **if(**ret**==**1**)** **{** |
|  | Journey\_Start**();** |
|  | **}** **else** **if(**ret**==**2**)** **{** |
|  | state**=**13**;** |
|  | lcd**.**Display\_Setup**();** |
|  | **}** **else** **return;** |
|  | **break;** |
|  | **}** |
|  | **case** 10**:** **{** |
|  | *//重车状态，仅可以加减速* |
|  | **if(**ret**==**6**)**taxi**.**Start\_or\_SpeedUp**();** |
|  | **else** **if(**ret**==**3**)**taxi**.**Stop\_or\_SpeedDown**();** |
|  | **else** **if(**ret**==**1**)** **{** |
|  | Journey\_End**();** |
|  | **}** |
|  | **return;** |
|  | **}** |
|  | **case** 13**:** **{** |
|  | *//设置状态，仅可以设置或者加减速* |
|  | **if(**ret**==**6**)**taxi**.**Start\_or\_SpeedUp**();** |
|  | **else** **if(**ret**==**3**)**taxi**.**Stop\_or\_SpeedDown**();** |
|  | **else** **if(**ret**==**2**)** **{** |
|  | **if(**lcd**.**Youbiao\_isON**==**0**)** **{** |
|  | lcd**.**Display\_Youbiao\_ON**();** |
|  | lcd**.**Youbiao\_isON**=**1**;** |
|  | **}** **else** **if(**lcd**.**Youbiao\_isON**==**1**)** **{** |
|  | lcd**.**Display\_Youbiao\_OFF**();** |
|  | lcd**.**Youbiao\_isON**=**2**;** |
|  | **}** **else** **if(**lcd**.**Youbiao\_isON**==**2**)** **{** |
|  | lcd**.**Display\_Car\_Empty**();** |
|  | lcd**.**Youbiao\_isON**=**0**;** |
|  | state**=**11**;** |
|  | **}** |
|  | **}** |
|  | *//else if(ret==5);* |
|  | **else** **return;** |
|  | **}** |
|  | **default:** |
|  | **;** |
|  | **}** |
|  | **}** |
|  | **void** Wait\_IC**()** **{** |
|  | lcd**.**Display\_Wait\_ICcard**();** |
|  | *//接下来查询等待键盘插入IC（按键模拟）* |
|  | **while(**kb**.**Pressed\_Button**()!=**4**);** |
|  | state**=**12**;** |
|  | Self\_Check**();** |
|  | **}** |
|  | **void** Self\_Check**()** **{** |
|  | lcd**.**Display\_Self\_Check**();** |
|  | Sleep**(**1500**);** |
|  | state**=**11**;** |
|  | lcd**.**Display\_Car\_Empty**();** |
|  | Update**();** |
|  | *//等待空车牌被翻下，或是按键被按下* |
|  | **}** |
|  | **void** Update**()** **{** |
|  | **while(**1**)** **{** |
|  | **if(**kb**.**IsPRES**())** **{** |
|  | kb**.**read**();** |
|  | State\_Change**(**kb**.**ret**);** |
|  | **}** **else** **{** |
|  | **switch(**state**)** **{** |
|  | **case** 11**:** |
|  | lcd**.**Display\_Update\_Car\_Empty**();** |
|  | rgy**.**Print\_Kong**();** |
|  | **break;** |
|  | **case** 13**:***//此处本为设置，可以加一些反白的操作* |
|  | rgy**.**Print\_Kong**();** |
|  | **break;** |
|  | **case** 10**:** |
|  | lcd**.**Display\_Update\_Car\_Full**(**taxi**.**Read\_Meter\_Thistime**(),**Caculate\_Money\_Thistime**(**taxi**.**Meter\_Thistime**));** |
|  | **break;** |
|  | **default:** |
|  | **;** |
|  | **}** |
|  | **}** |
|  | **}** |
|  | **}** |
|  | **};** |

代码 9 M\_ADC0809类（录放音类）

|  |  |
| --- | --- |
|  | #define port\_ADC0809 0x2A0 |
|  | #define port\_DAC0832 0x280 |
|  | **using** **namespace** std**;** |
|  | **struct** M\_ADC0809 **{** |
|  | BYTE **\***Data\_Point**;***//数据区指针* |
|  | **int** port\_change**;** |
|  | M\_ADC0809**():**port\_change**(**port\_ADC0809**+**3**)** **{** *//以IN3口转换为例* |
|  | Data\_Point **=** **(**BYTE **\*)**malloc**(**6000**);** */\*分配空间用于存放录音数据\*/* |
|  | **if(!**Data\_Point**)** **{** |
|  | printf**("No memory!\7");** |
|  | exit**(**0**);** |
|  | **}** |
|  | **}** |
|  | **void** Record**()** **{** |
|  | BYTE data**;** |
|  | printf**("Press any key to record!\n");** */\*录音提示\*/* |
|  | getch**();** |
|  | printf**("录音中……\n");** */\*录音提示\*/* |
|  | fstream file**;** |
|  | file**.**open**("SoundOut.txt",**ios**::**out**);** |
|  | **for(int** i**=**0**;** i**<**30000**;** i**++)** **{** |
|  | */\*启动A/D,采集6000个数据放在开辟的内存空间中\*/* |
|  | PortWriteByte**(**port\_change**,**0**);***//特色的软启动* |
|  | *//delay();* |
|  | PortReadByte**(**port\_change**,&**data**);** |
|  | *//省略检测EOC，等待1ms后便直接读取转换后的数值* |
|  | **\*(**Data\_Point**+**i**)** **=** data**;** |
|  | file**<<**data**<<**endl**;** |
|  | **}** |
|  | file**.**close**();** |
|  | printf**("录音已结束。\n");** */\*录音提示\*/* |
|  | **}** |
|  | **void** Play**()** **{** |
|  | printf**("Press any key to playing!\n");** */\*放音提示\*/* |
|  | getch**();** |
|  | BYTE data**;** |
|  | fstream file**;** |
|  | file**.**open**("SoundOut.txt",**ios**::**in**);** |
|  | **for(int** i**=**0**;** i**<**30000**;** i**++)** **{** |
|  | */\*将ii中的6000个从D/A输出\*/* |
|  | data **=** **\*(**Data\_Point**+**i**);** |
|  | PortWriteByte**(**port\_DAC0832**,**data**);** |
|  | **}** |
|  | file**.**close**();** |
|  | printf**("Playing end!\n");** |
|  | **}** |
|  | **void** delay**()** **{** |
|  | byte d**;** |
|  | **do** **{** |
|  | PortReadByte**(**0x28a**,&**d**);** |
|  | **}** **while(**d**&**1**!=**0**);** |
|  | **do** **{** |
|  | PortReadByte**(**0x28a**,&**d**);** |
|  | **}** **while(**d**&**1**!=**1**);** |
|  | **}** |
|  | **};** |

代码 10 M\_8259类

|  |  |
| --- | --- |
|  | #define port\_8259 0x2A8 |
|  | **struct** M\_8259 **{** |
|  | **int** port\_odd**;** |
|  | **int** port\_even**;** |
|  | M\_8259**():**port\_odd**(**port\_8259**+**1**),**port\_even**(**port\_8259**)** **{** |
|  | Init**();** |
|  | InterruptMask\_Open**();** |
|  | Wait\_Interrupt**();** |
|  | **}** |
|  | **void** InterruptMask\_Open**()** **{** |
|  | PortWriteByte**(**port\_odd**,**0x00**);***//写入OCW1开中断* |
|  | **}** |
|  | **void** Interrupt\_End**()** **{** |
|  | *//写入OCW2为00100000来结束刚才服务过的中断* |
|  | PortWriteByte**(**port\_even**,**0x20**);** |
|  | Wait\_Interrupt**();** |
|  | **}** |
|  | **void** IR0**()** **{** |
|  | cout**<<"中断0"<<**endl**;** |
|  |  |
|  | Interrupt\_End**();** |
|  | **}** |
|  | **void** IR1**()** **{** |
|  | cout**<<"中断1"<<**endl**;** |
|  | Interrupt\_End**();** |
|  | **}** |
|  | **void** Init**()** **{** |
|  | PortWriteByte**(**port\_even**,**0x13**);***//ICW1* |
|  | PortWriteByte**(**port\_odd**,**0xB0**);***//ICW2* |
|  | PortWriteByte**(**port\_odd**,**0x03**);***//ICW4,自动结束中断方式* |
|  | **}** |
|  | **int** Read\_ISR**()** **{** |
|  | byte data**;** |
|  | PortWriteByte**(**port\_even**,**0x0B**);** |
|  | PortReadByte**(**port\_even**,&**data**);** |
|  | **return** data**;** |
|  | **}** |
|  | **void** Wait\_Interrupt**()** **{** |
|  | cout**<<"Wait Interrupt...."<<**endl**;** |
|  | **while(**Read\_ICheck**()==-**1**)** **{** |
|  | Sleep**(**20**);** |
|  | **}** |
|  | **}** |
|  | **int** Read\_IRR**()** **{** |
|  | byte data**;** |
|  | PortWriteByte**(**port\_even**,**0x0A**);** |
|  | PortReadByte**(**port\_even**,&**data**);** |
|  | **return** data**;** |
|  | **}** |
|  | **int** Read\_ICheck**()** **{** |
|  | *//读取中断查询字，最高位表示有无中断，低三位显示是哪一级中断* |
|  | byte data**;** |
|  | PortWriteByte**(**port\_even**,**0x0C**);** |
|  | PortReadByte**(**port\_even**,&**data**);** |
|  | *//cout<<(int)data<<endl;* |
|  | **if((int)**data**/**128**==**1**)** **{** |
|  | **switch((int)**data**%**8**)** **{** |
|  | **case** 0**:** |
|  | IR0**();** |
|  | **break;** |
|  | **case** 1**:** |
|  | IR1**();** |
|  | **default:** |
|  | **;** |
|  | **}** |
|  | **return** **(int)**data**%**8**;** |
|  | **}** **else** **return** **-**1**;** |
|  | **}** |
|  | **int** Read\_IMR**()** **{** |
|  | byte data**;** |
|  | PortReadByte**(**port\_odd**,&**data**);** |
|  | **return** data**;** |
|  | **}** |
|  | **};** |

代码 11 主文件代码

|  |  |
| --- | --- |
|  | #include <conio.h> |
|  | #include <stdio.h> |
|  | #include "ApiExUsb.h" |
|  | #pragma comment(lib,"ApiExUsb.lib") |
|  | #include<iostream> |
|  | **using** **namespace** std**;** |
|  |  |
|  | #include"TAXIMETER.h" |
|  | **int** main**(){** |
|  | Cleanup**();** |
|  | **if(**Startup**()){** |
|  | cout**<<"right"<<**endl**;** |
|  | **}** |
|  | TAXIMETER tm**;** |
|  | tm**.**Wait\_IC**();** |
|  | Cleanup**();** |
|  | **return** 0**;** |
|  | **}** |

1. 程序中实现了8259中断，并提供了中断接口，但整体演示系统仍使用8255C端口查询键盘的方式。 [↑](#footnote-ref-1)
2. 系统中实现的总运营里程仅包含载客里程总数，不包含空车状态下的行驶里程。 [↑](#footnote-ref-2)
3. 这种方式进行的循环装数再计数在理论上会有一定的计数误差。车速越快，读数间隔越长，误差越大。 [↑](#footnote-ref-3)