中国矿业大学

计算机科学与技术学院

2018 级本科生课程报告

课程名称 硬件课程设计

设计题目 电梯设计

开课学期 2020-2021 学年第一学期

报告时间 2021 年 1 月 8 日

学生姓名 郑晓雪

学　　号 09183180

班　　级 计算机科学与技术 18-2 班

专　　业 计算机科学与技术

任课教师 王凯

《硬件课程设计》课程报告评分表

开课学期: 2020-2021 学年第一学期 姓名: 郑晓雪　学号：09183180　专业班级：计科 18-2

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：

年　月　日

目录

1. 绪论

生活在继续，科技在发展，电梯也在进步。电梯的材质由黑白到彩色，在操纵控制方面更是步步出新—手柄开关操纵、按钮控制、信号控制、人机对话等等，多台电梯还出现了并联控制，智能群控。

建筑业的迅速发展以及高层建筑的不断涌现，电梯则成为建筑内提供上下交通运输的工具，而它的发展空间也在不断扩大。为使发展迅速的电梯产业得以稳定发展、人们的日常生活得以正常进行，电梯应具有安全、可靠、高效、环保、方便等控制优点。

本实验设计题目是电梯设计，基于 TPC-ZK 微机接口实验系统，实现了对实际电梯运行的模拟。在实验过程中，我们可以深入了解电梯的内部逻辑，锻炼解决问题的能力，为进一步完善电梯的功能、提高运行效率、满足乘客需求进行深入的研究。

1.1 问题提出

电梯设计需要实现什么功能？在分析设计中提出以下几点问题？

问题一：如何实现 4*4 键盘输入楼层号和操作指令？

问题二：如何实现步进电机转动、延时和停止？

问题三：如何实现步进电机正反转？

问题四：如何实现 LCD 显示汉字？

问题五：如何实现超载检测？以及超载报警？

问题六：如何实现电梯近距离优先原则进行运行？

1.2 设计任务与要求

利用 8255 控制 4*4 键盘与 LCD 显示屏，利用 8255 控制步进电机正、反转，完成一个 5 层电梯上下方向控制，LCD 屏显示上行、下行、开门、关门、楼层好显示。要求电梯运行顺畅，能够判断目标楼层与中间楼层，控制近距离优先原则进行运行。本实验采用 C 语言。

2. 系统设计需求分析

2.1 系统组成原理及开发平台

从**错误!未找到引用源。**中系统需求中可以看出，需要使用的硬件有 8255、ADC0809、

DAC0832、4*4 键盘、LCD12864 液晶屏、直流信号、蜂鸣器。

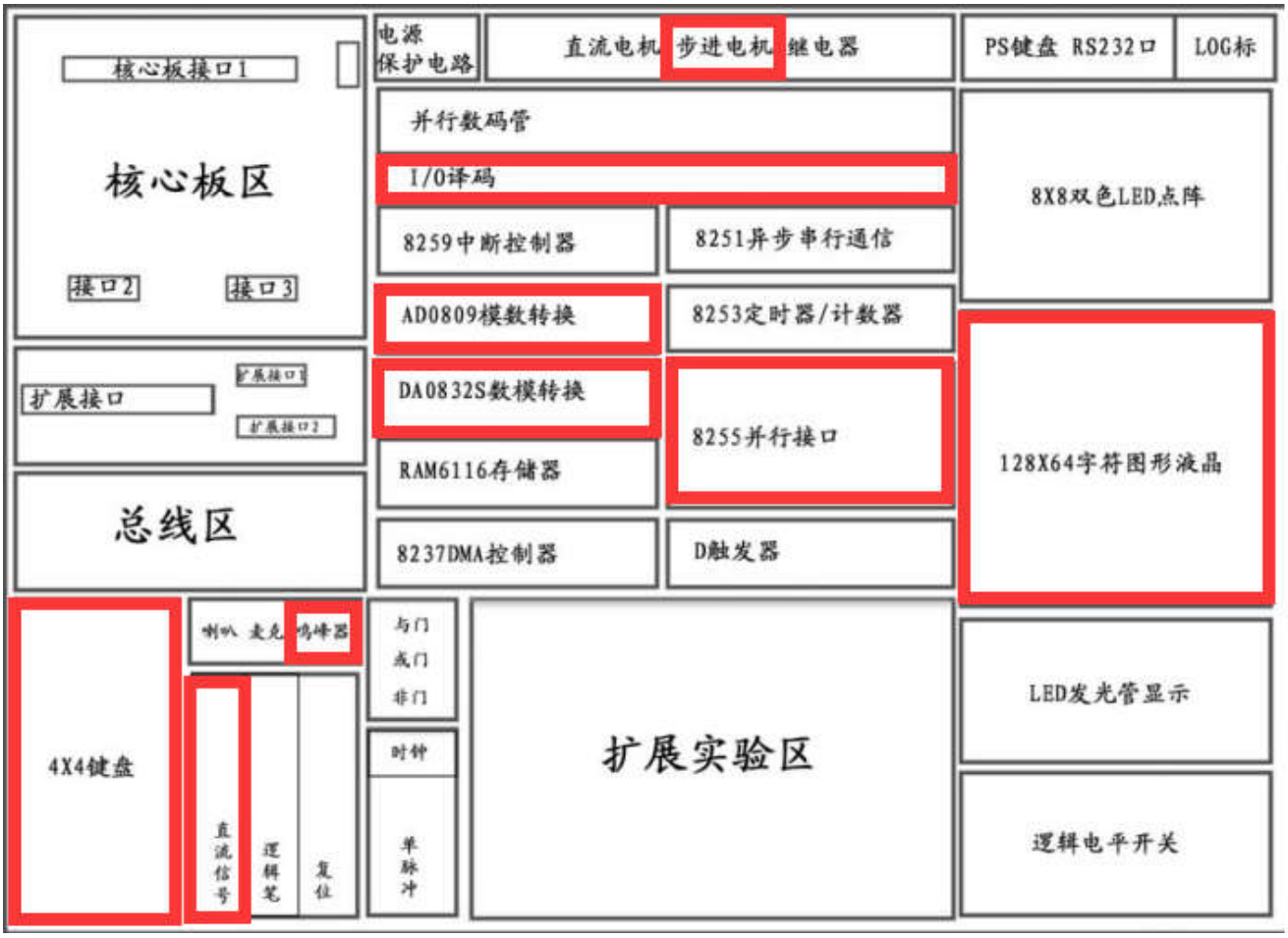


图2-1　TCP-ZK 实验系统结构图

2.1.1 8255

8255 是系统的主控芯片，选用控制字 0x80，使 A 口 B 口输出。当系统采用查询键盘的方式进行工作时，先选用控制字为 0x81，C 口高 4 位输出低 4 位输入，后选用控制字 0x88，C 口高 4 位输入低 4 位输出。B 口的低 3 位负责控制 LCD12864 液晶的控制引脚，B 口高四位负责控制步进电机的 BA、BB、BC、BD，通过对每相线圈中的电流的顺序切换来使电机作步进式旋转。

2.1.2 4*4 键盘

按键扫描的具体思想是：首先从第一行到第四行依次判断，是否有某一行有键按下，当确定了行之后，利用移位操作，来寻找该行中按下的键所在列，至此，已经可以确定按键位置。这里需要注意的是——消抖，防止电路抖动，造成不期望看到的现象。

先选用控制字为 0x81，C 口高 4 位输出低 4 位输入，C 口高 4 位负责行线的拉低，检测低四位，若低四位有 0，则读行扫描值，后选用控制字 0x88，C 口高 4 位输入低 4 位输出，C 口低 4 位负责列线的拉低，若高四位有 0，则读列扫描值，确定键盘号。

2.1.3 DAC0832

DAC0832 在系统中，负责在超载报警过程中，输出电压信号到蜂鸣器，控制蜂鸣器启停。

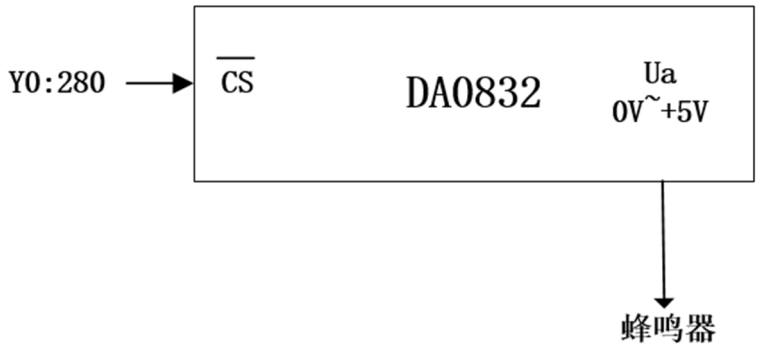


图2-2　DAC0832 连线图

2.1.4 LCD12864 液晶屏

LCD12864 液晶屏在系统中，主要负责界面的显示与更新，显示电梯运行楼层变化的过程，还有超载提示信息 。

LCD12864 所有的数据读写都是先送地址，然后进行读写。每个字中的 2 个字节自动结合查找字模并显示字符

地址与屏幕显示对应关系如下：

表2.1　汉字显示坐标

	X 坐标							
Line1	80H	81H	82H	83H	84H	85H	86H	87H
Line2	90H	91H	92H	93H	94H	95H	96H	97H
Line3	88H	89H	8AH	8BH	8CH	8DH	8EH	8FH
Line4	98H	99H	9AH	9BH	9CH	9DH	9EH	9FH

表2.2　LCD 的控制线功能

RS (CS)	H: Data L: Instruction Code
R/W (SID)	H: Read L: Write
E (SCLK)	Enable Signal

2.1.5 步进电机

电梯上升时，步进电机正转，每变化一个楼层，执行一个 00110011——10011001 这样的循环，到达目标楼层时，写入 11111111 停止电机。下降时，改变通电顺序使步进电机反转。

步进电机**驱动原理**是通过对每相线圈中的电流的顺序切换来使步进电机作步进式旋转。首先向 φ1 线圈－φ2 线圈输入驱动电流，接着 φ2－φ3，φ3－φ4，φ4－φ1，又返回到 φ1－φ2，按这种顺序切换，电机轴按顺时针方向旋转。反之，逆时针方向旋转。

驱动方式为二相激磁方式，各线圈通电顺序如下表。

顺序 / 相	Φ1	Φ2	Φ3	Φ4
0	1	1	0	0
1	0	1	1	0
2	0	0	1	1
3	1	0	0	1

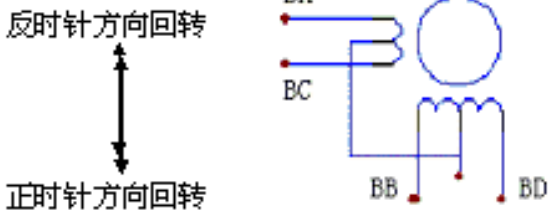


图2-3　步进电机转动原理图

2.1.6 AD0809

AD0809 在系统中，主要负责在超载模块中采集重量，此处使用直流信号（0V-5V），并转换为数字量进行存储，供超载检测。

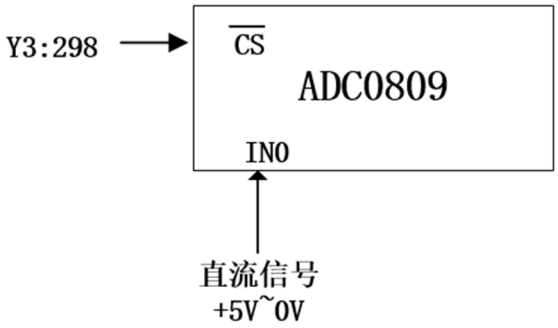


图2-4　ADC0809 连线图

2.1.7 开发平台

操作系统： Windows 7 64 位

集成开发环境：DEV C++

开发平台： TPC-ZK-II 综合开放式微机原理及接口技术实验系统

TPC-ZK 教学实验系统是高等院校理工科类各专业微机接口实验常用的系统，TPC-ZK 实验系统上配置了 USB 接口模块，直接与主机（PC）的 USB 接口连接，形成了一整套完整的 USB 接口的微机接口实验系统。该系统由一块 USB 总线接口模块、TPC-ZK 实验系统及集成环境软件组成。具有高速 USB 下的通信能力，即插即用。接口集成电路丰富，包括：可编程定时器/计数器（8254）、可编程并行接口（8255）、数/模转换器（DAC0832）、模/数转换器（ADC0809）等。外围电路包括：逻辑电平开关、LED 显示、七段数码管显示、8*8 双色发光二极管点阵及驱动电路、键盘显示控制电路等。实验程序可以使 8086 汇编和 C 语言编程实验，可以对汇编程序和 C 语言程序进行调试。并且实验台自备电源，具有电源短路保护确保系统安全。使用 USB 接口与 PC 机相连，省却了打开主机箱安装接口卡的麻烦。

2.1.8 系统设计电路原理图

表2.3 系统设计电路接线：

8255/CS	接	I/O 地址译码/Y1(288H---28FH)
8255/JP8(PC7-PC0)	接	4*4 键盘行 3---列 0
8255/JP6(PA7-PA0)	接	128x64 液晶显示屏 D7-D0
8255/JP7(PB2-PB0)	接	LCD 的 E，RW，D/I
8255/JP7(PB7-PB4)	接	步进电机 BA,BB,BC,BD
0809/CS	接	I/O 地址译码/Y3(298H---29FH)
0809/IN0	接	直流信号+5V~0V
DAC0832/CS	接	I/O 地址译码/Y0(280H---287H)
DAC0832/Ua	接	蜂鸣器

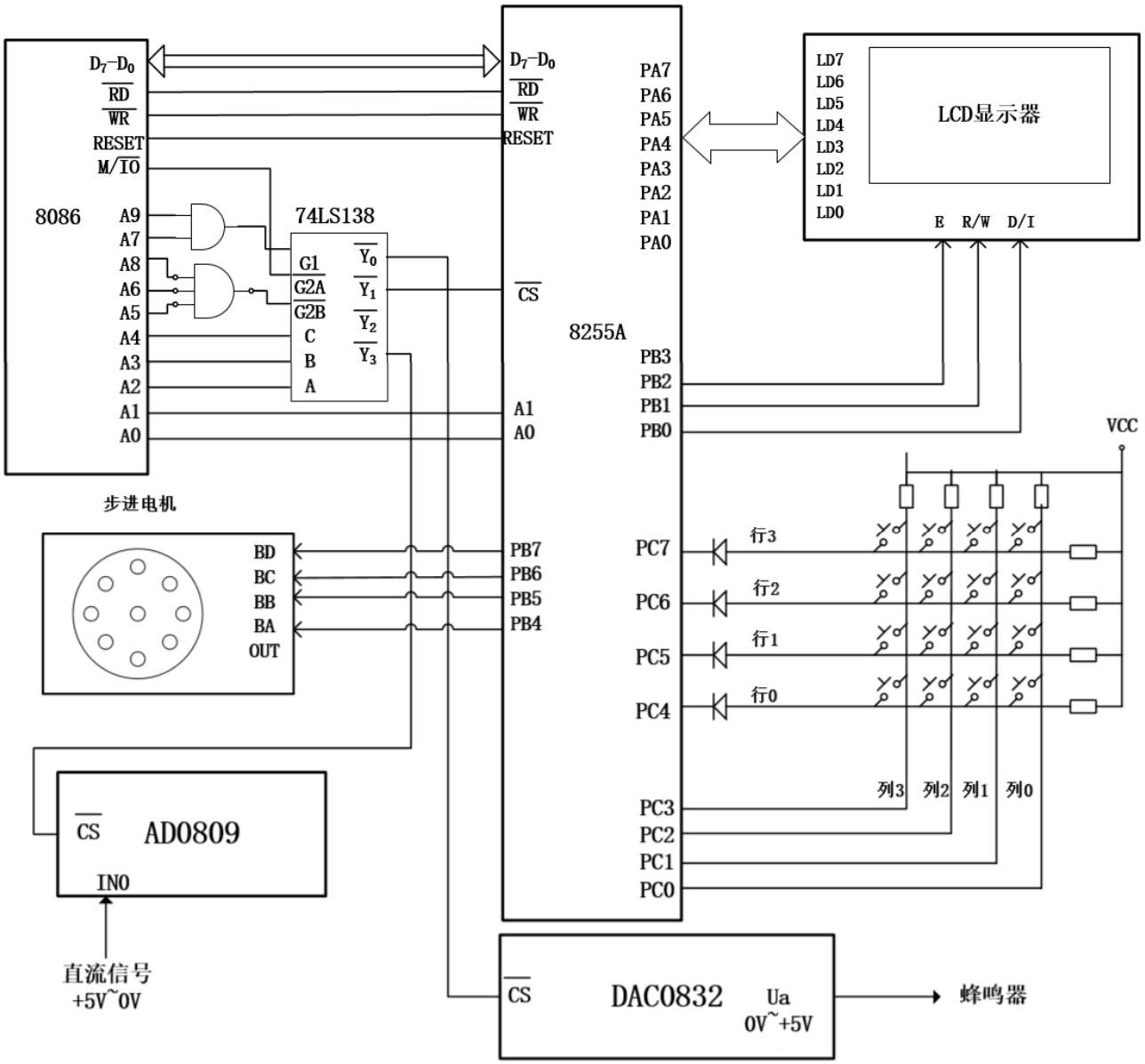


图2-5 系统设计电路原理图

2.2 系统工作业务流程图

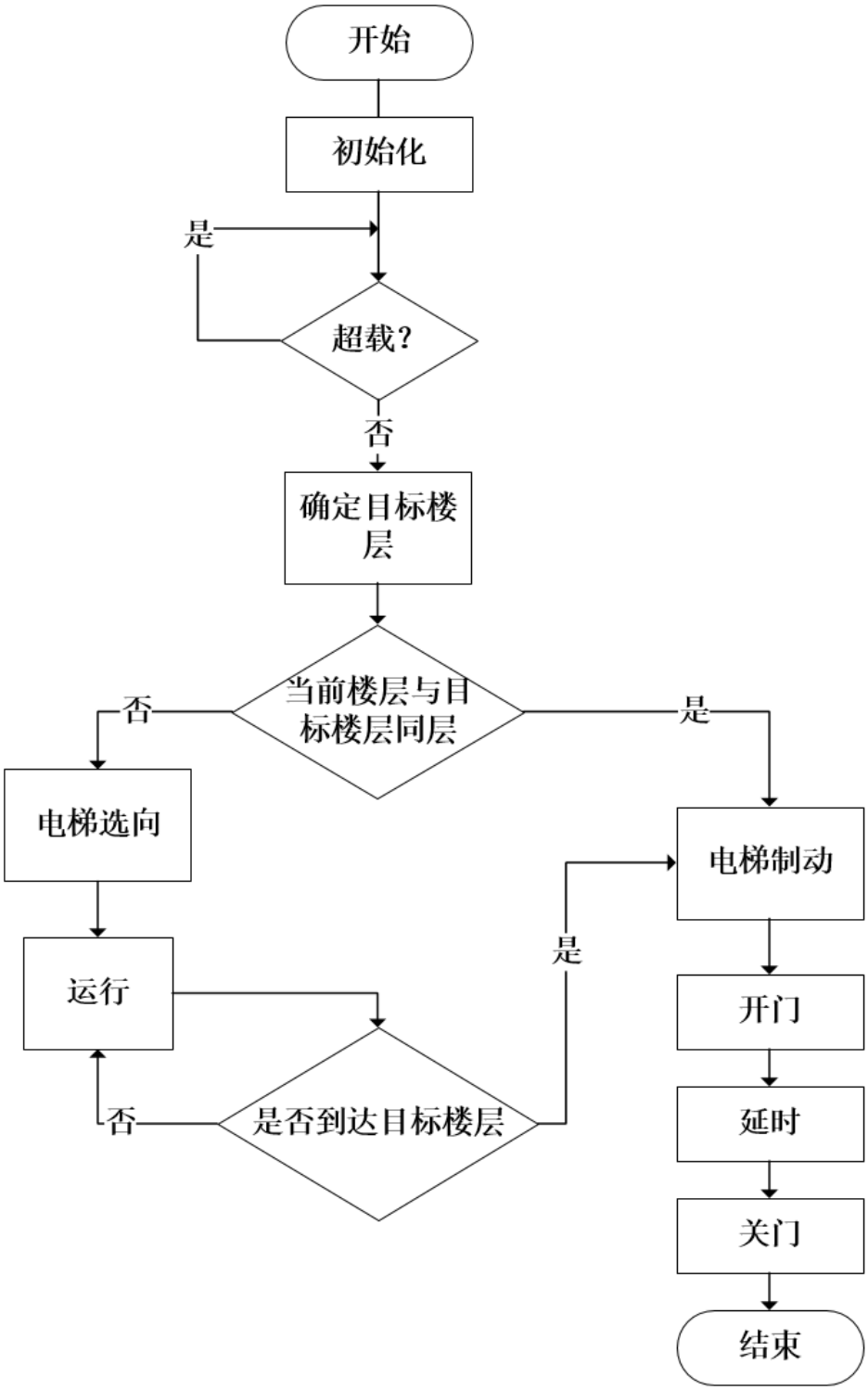


图2-6 系统工作业务流程图

2.3 系统数据流程分析

2.3.1 系统设计 DFD 图

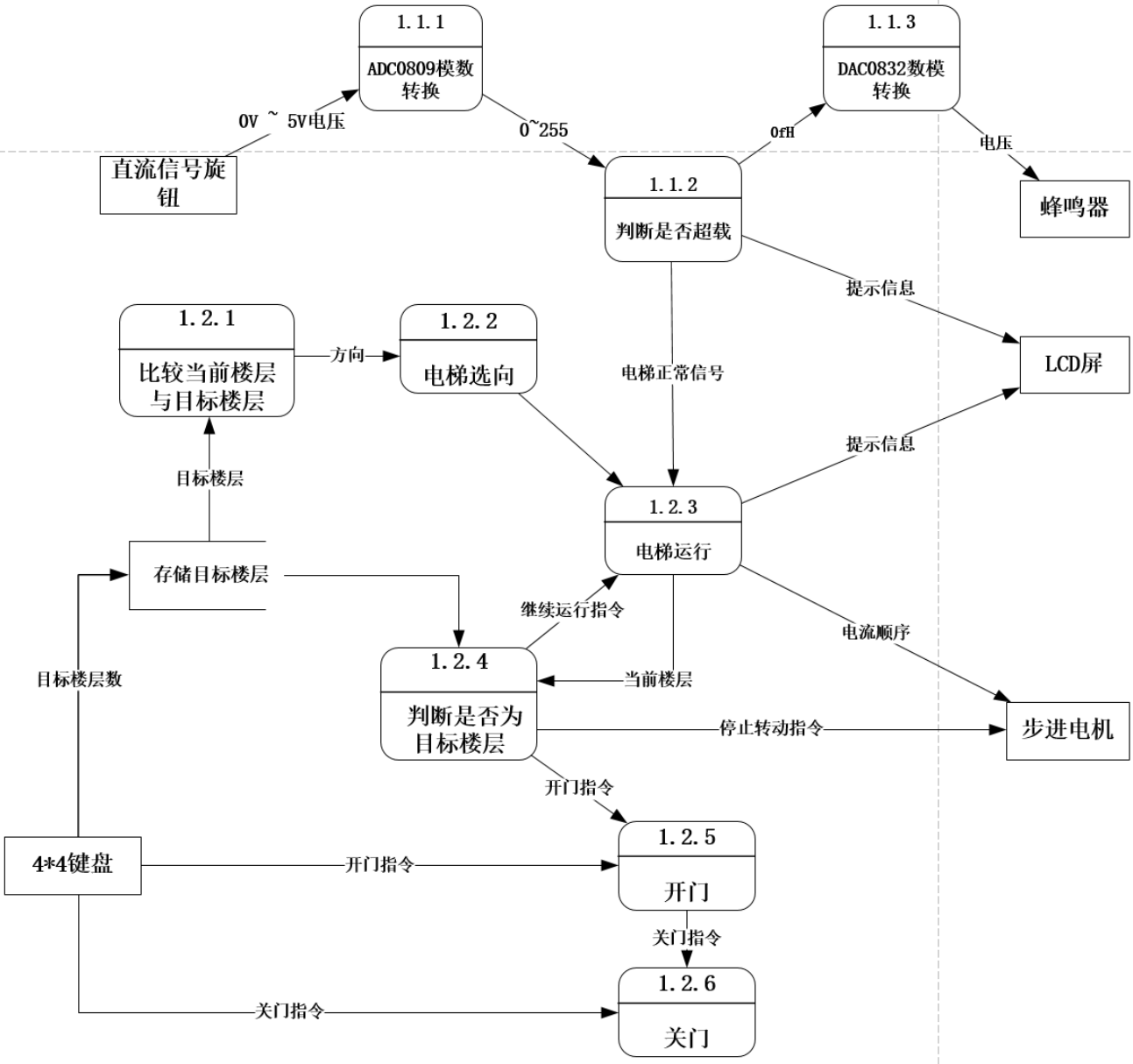


图2-7 系统设计 DFD 图

3. 系统的总体设计

3.1 系统设计目标

电梯系统实现五层电梯的运行，实现电梯上下行、开门、关门、信息显示、超载检测的功能。

(1) 电梯运行前，电梯系统进行超载检测，若电梯超载，响一声“滴”，并在屏幕上提示乘客减少乘客数；否则电梯正常运行。

(2) 电梯系统能够判断目标楼层与中间楼层，若目标楼层和当前楼层同层，电梯制动，电梯开门关门，否则电梯保持运行。

(3) 电梯系统延时控制近距离优先原则进行运行，

3.2 系统功能层次图

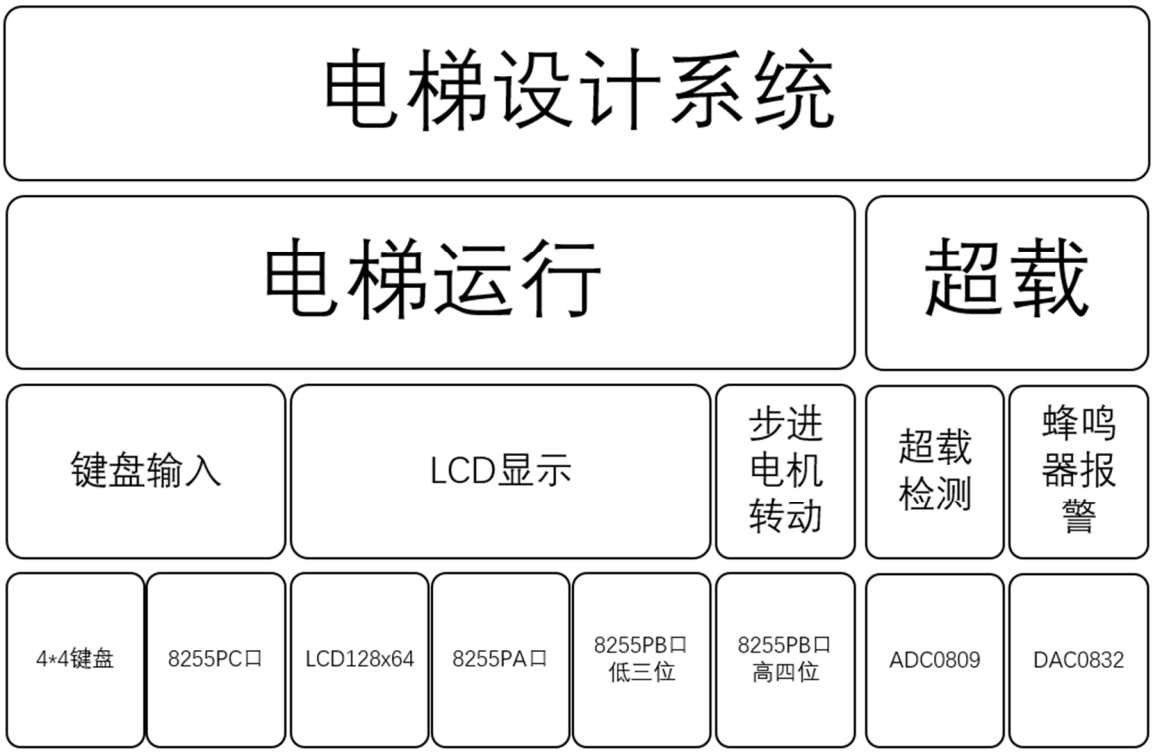


图3-1 系统功能层次图

3.2.1 系统功能描述

1. 起始楼层为 1，按下 2 至 5 中任意楼层 X 电梯上升，LCD 显示电梯上楼的过程，每一层

楼延时 2 秒，同时步进电机顺时针转动。到达目标楼层后步进电机停止下来，LCD 显示“到达楼层：X”，此信息显示 1.4 秒。之后显示开门信息，开门 5 秒之后关门。

2.起始楼层为 1，先按下楼层 X，再在经过 X-1 楼层之前按下 X-1。LCD 显示电梯上楼的过程，每一层楼延时 2 秒，同时步进电机顺时针转动。电梯会先在 X-1 楼层停下，步进电机停止转动，显示到达信息后并开门关门。关门后继续前往 X 楼层，步进电机继续转动。到达 X 楼层后步进电机停止转动，LCD 显示到达信息和开关门信息。

3. 起始楼层为 1，先按下楼层 X，再在经过 X-1 楼层之后按下 X-1。LCD 显示电梯上楼的过程，每一层楼延时 2 秒，同时步进电机顺时针转动。电梯会先在 X 楼层停下，步进电机停止转动，显示到达信息后并开门关门。关门后继续前往 X-1 楼层，改变电梯运行方向，步进电机反向转动。到达 X-1 楼层后步进电机停止转动，LCD 显示到达信息和开关门信息。

4.在上述过程中，扫描到键盘按下按键 0，进行楼层初始化，电梯初始化为 1 楼，LCD 显示“当前楼层：1”，电梯停止转动。

5.在电梯运行时扫描到按下按键 6 开门是不处理的，只有在电梯停止运行的时候才会处理开门和关门的按键。

3.2.2 系统算法设计

1.设置全局变量 door 判断当前状态电梯门的开关状态，now 当前楼层。这两个全局变量在各个函数中皆有使用，通过查验实时状态来进行相关控制。主函数的主要参数 dir 用来判断电梯当前是否运行和运行的方向，参数实时状态分别传入对应的电梯上升和电梯下降楼层。

2.利用 8255 芯片的 C 接口控制键盘的行和列，键盘设置的按键值对应为下表

表3.1　4*4 键盘按键对应功能表

按键	对应功能
0	楼层初始化
1	楼层 1
2	楼层 2
3	楼层 3
4	楼层 4
5	楼层 5
6	开门
7	关门

设置循环函数扫描键盘，对键盘的行和列分别判断后读入，如果按键是楼层键就向上层函数返回按下的楼层键；如果按键是功能键就转入对应的功能函数里面，返回-1。

3.设置一个电梯上升和下降的函数显示楼层变化的一个过程，同时在函数里面控制电机。

电梯上升就控制步进电机顺时针转动，传入一个布尔型变量；电梯下降就控制步进电机逆时针转动，传入与上升时相反的布尔型变量。

电梯的转动另外设置在一个接收布尔型变量的无返回值函数里，对初始定义二进制的自变量左移右移控制以及补上溢出位数来保持自变量的一个循环使用，从而对应步进电机的的四个相位控制转动。

4.LCD 显示主要分为到达楼层显示函数和楼层变化过程中显示函数。其中到达楼层显示函数中在显示完达到楼层之后显示开门，在到达楼层显示函数末尾调动开门函数；开门一段时间之后进行关门，也要建立开门函数对关门函数的一个控制。楼层变化过程显示又分为电梯上升和电梯下降。LCD 的基本操作步骤先下入控制字然后对 LCD 写入命令，再清除缓存，对 LCD 写入数据，再读入数据。

5.延时采用的软延时，通过 Sleep 函数传入参数(单位毫秒)对程序进行延时。

3.2.3 系统输入设计

（1） 程序输入控制字到 8255；
（2） 键盘输入目标楼层号和开门、关门指令到 8255 的 PC 口；
（3） 直流信号+5V~0V 输入到 0809/IN0；
（4） 程序输入数值 0x0f 到 0832/Ua。

3.2.4 系统输出设计

（1） 8255PA 口连接 LCD 的数据总线，在 LCD 上显示汉字；
（2） 利用 8255PB 口的低三位输出控制 IO，RW，E；
（3） 利用 8255PB 口输出脉冲序列到步进电机 BA,BB,BC,BD。

3.2.5 系统拓展设计

电梯超载，当系统检测到电梯超载，蜂鸣器“滴”的响一声，同时 LCD 屏上显示提示信息。

利用 ADC0809 将电压模拟量转换为二进制数，若检测到数值则在 LCD 屏上显示“电梯超载！请减少乘客数量”。旋转直流信号旋钮，将电压值调到最小，电梯恢复正常。

利用 DAC0832 将输入的数值 0FH 转换为电压，启动蜂鸣器，在经过延时后，再将输入的数值 00H 为电压 0V，关闭蜂鸣器。

4. 系统的详细设计

4.1 键盘扫描模块设计

4.1.1 程序流程图或算法流程图

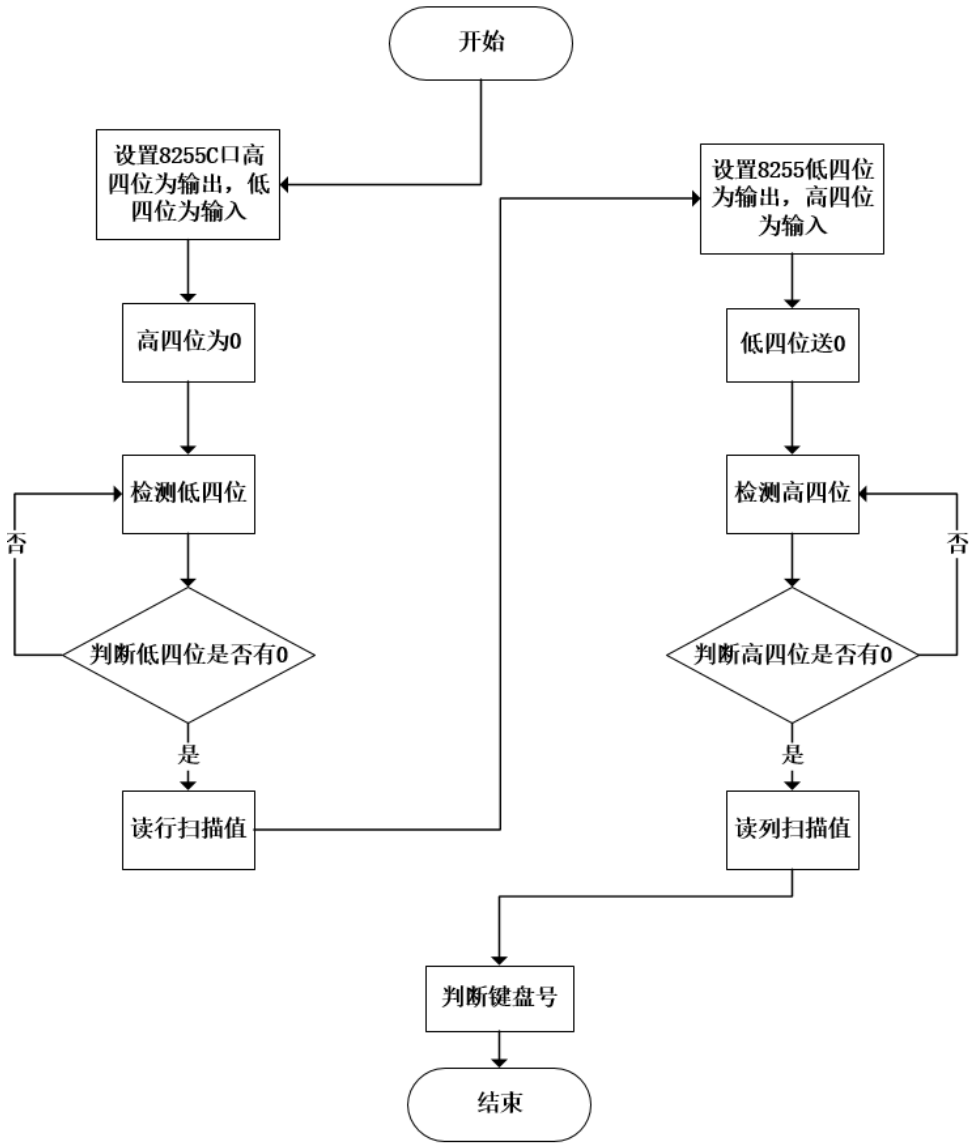


图4-1 键盘扫描模块程序流程图

4.1.2 系统功能描述

（1）当有键按下时进入键盘扫描模块，先修改 8255 控制字使 C 口高四位为输出，低四位

为输入，向 C 口高四位送 0，判断低四位是否有 0，若无，则重新扫描，若有，则读取行扫描值；接着修改 8255 控制字使 C 口低四位为输出，高四位为输入，向 C 口低四位送 0，判断高四位是否有 0，若无，则重新扫描，若有，则读取列扫描值。

（2）将记录的按键与键盘扫描码表对比，直至找到按键对应的扫描码，若找不到则重新扫描。根据键盘扫描码的位置进行表转换，找到对应的键盘记录表的位置，存入键盘的记录号。

4.2 超载检测模块设计

4.2.1 程序流程图或算法流程图

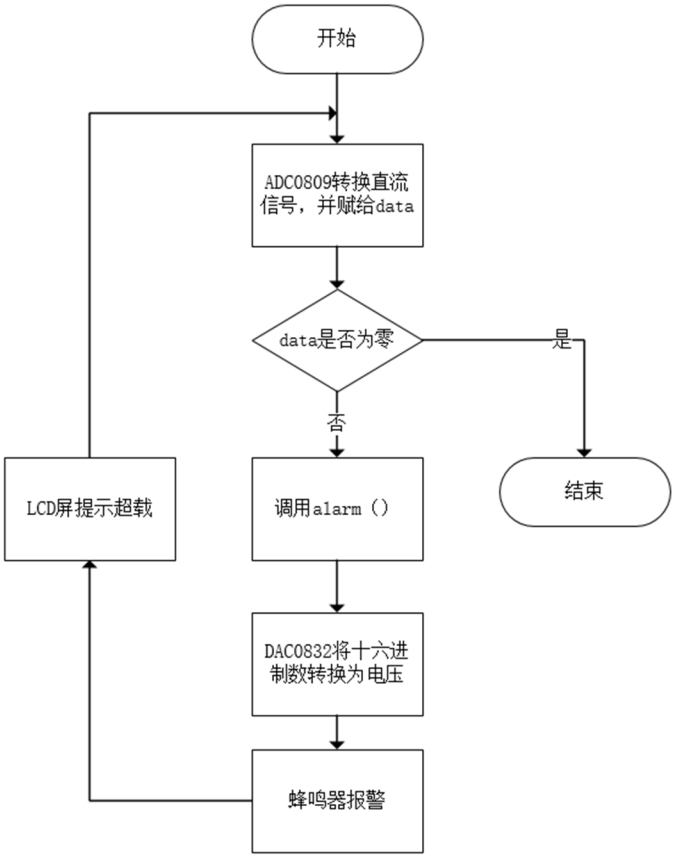


图4-2 超载检测模块程序流程图

4.2.2 系统功能描述

（1）电梯运行前进行超载检测，使用 ADC0809 芯片将直流信号转换成二进制数字量，向 0809 送任意值虚写启动，延时一段时间后，读入数值 data，若 data 为零，则不超载，电梯正常运行，否则调用 alarm（）使蜂鸣器报警。

（2）使用 DAC0832 将 0x0f 转换成电压模拟量，首先写入 0x00 清零，然后进行 0x0f 转

换，蜂鸣器有电压输入，开始持续地发出“滴”的声音，最后再次写入 0x00 将蜂鸣器关闭。

4.2.3 运行界面截图

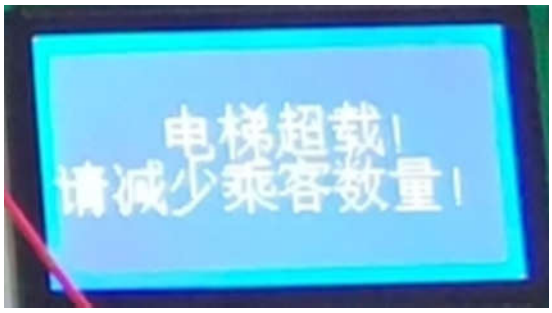


图4-3　超载检测运行ＬＣＤ显示界面

4.3　楼层选择模块设计

4.3.1 程序流程图或算法流程图

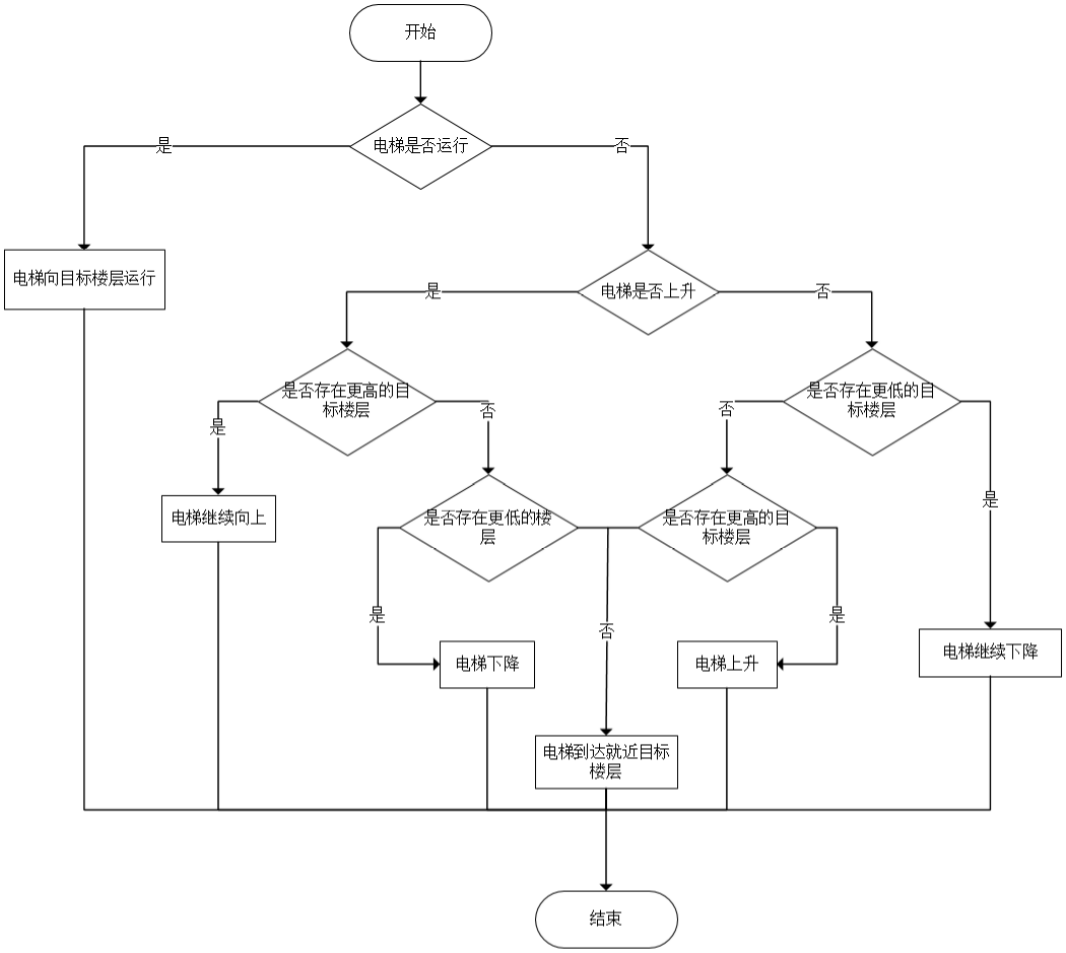


图4-4　楼层选择模块程序流程图

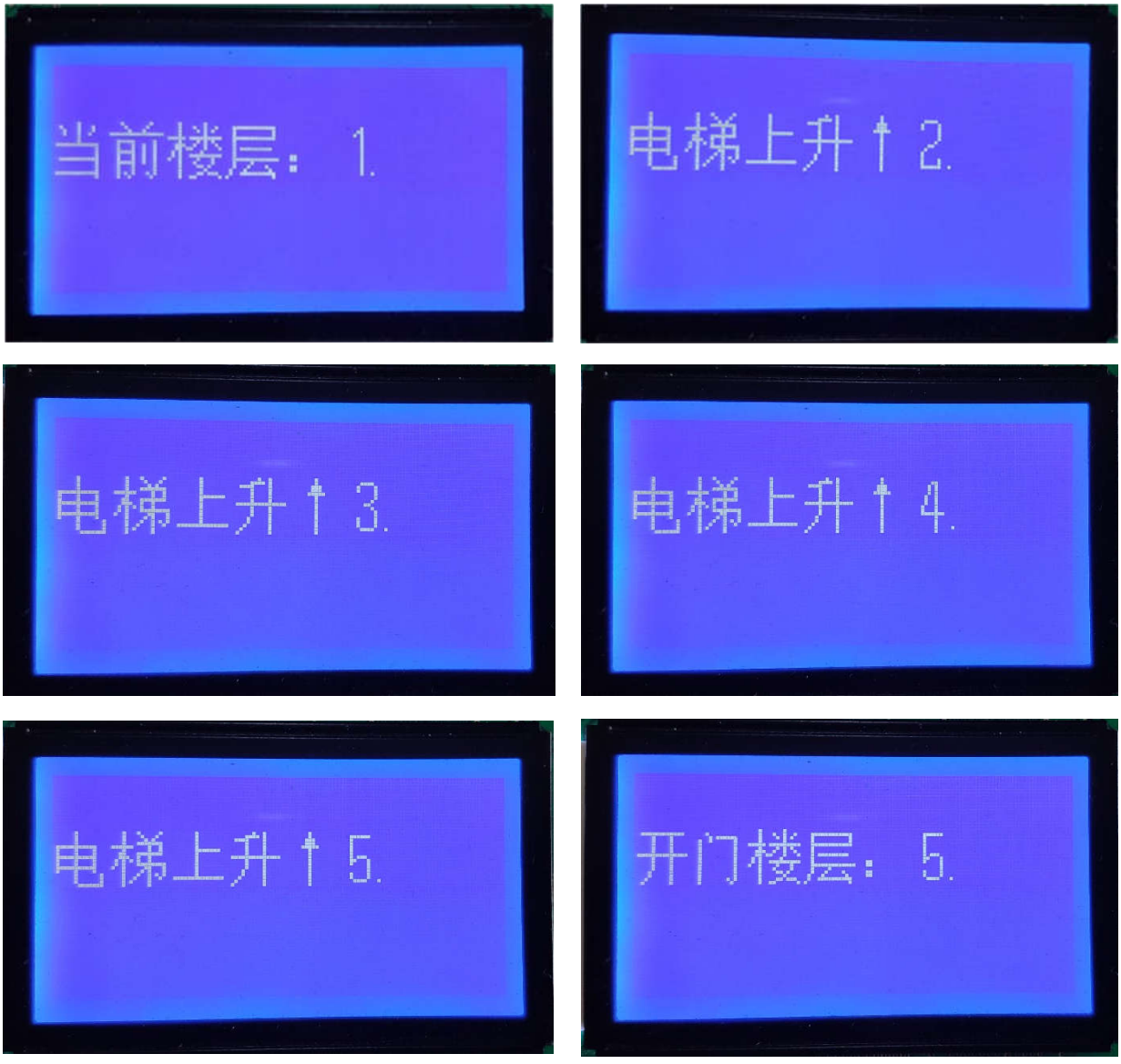
4.3.2 系统功能描述

先是设置两个基本函数获取目标楼层中最大楼层和最小楼层。再在主函数中判断电梯的运行情况，如果当前电梯还没有目标楼层，再按下目标楼层之后就开始向目标楼层运动，与当前楼层相减就可以比较出是上升还是下降。

如果电梯当前在运行中，每经过一楼就判断这一楼是否在目标楼层里，这样可以避免走多余的路径。如果达到目标楼层后，判断在当前运行方向时候存在最大/最小的目标楼层，有就继续运行，没有就判断运行的反方向是否存在最小/最大的目标楼层，有就开始电梯反方向运行，若没有那则是电梯没有目标楼层了，电梯保持静止。

4.3.3 运行界面截图

电梯从一楼到五楼：



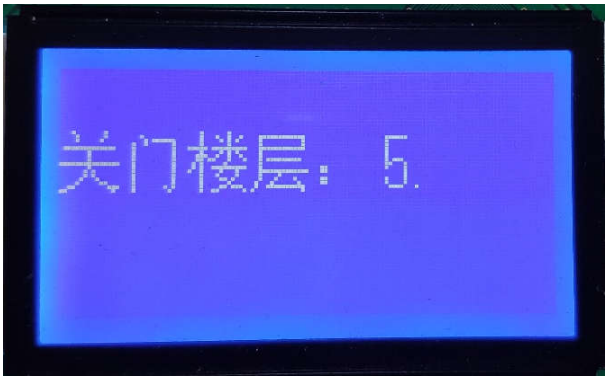
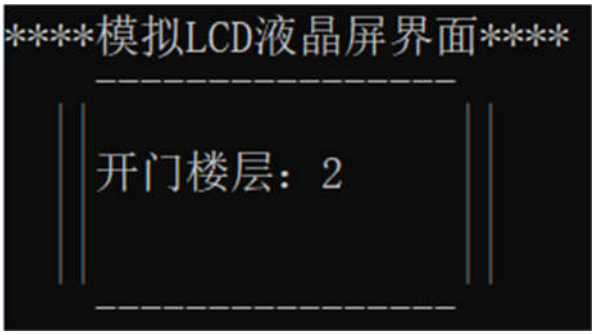
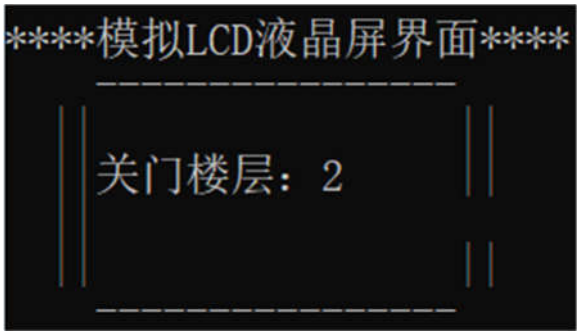
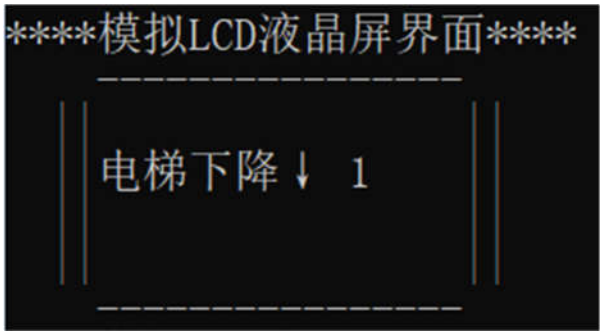


图4-5　LCD 显示电梯从一楼到五楼运行状态

五楼出发下降，先按下一楼，再快速按下二楼：这部分实验，我拍摄的照片不够清晰，打印效果不佳，所以用代码模拟了 LCD 界面。







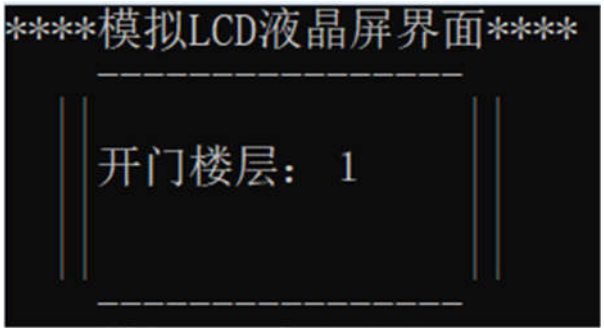


图4-6　LCD 显示电梯就近选择运行状态

4.4 显示模块设计

4.4.1 程序流程图或算法流程图

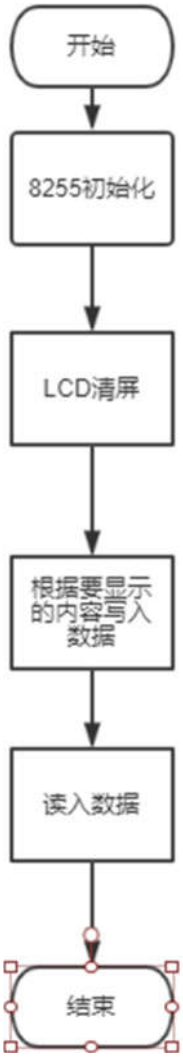


图4-7 LCD 显示模块程序流程图

4.4.2 系统功能描述

将电梯运行中需要显示内容的每个汉字和对应楼层号都设置为各个特殊值。开始时先对 8255 初始化，使 A 口和 B 口都为输出，对 LCD 液晶屏进行清屏。接着，根据存入的特殊值，判断 LCD 液晶屏要显示的内容，并把特殊值对应的显示值的表存入缓存中。接着读入缓存数据，实现 LCD 液晶屏的按需求显示。

4.5 步进电机模块设计

4.5.1 程序流程图或算法流程图

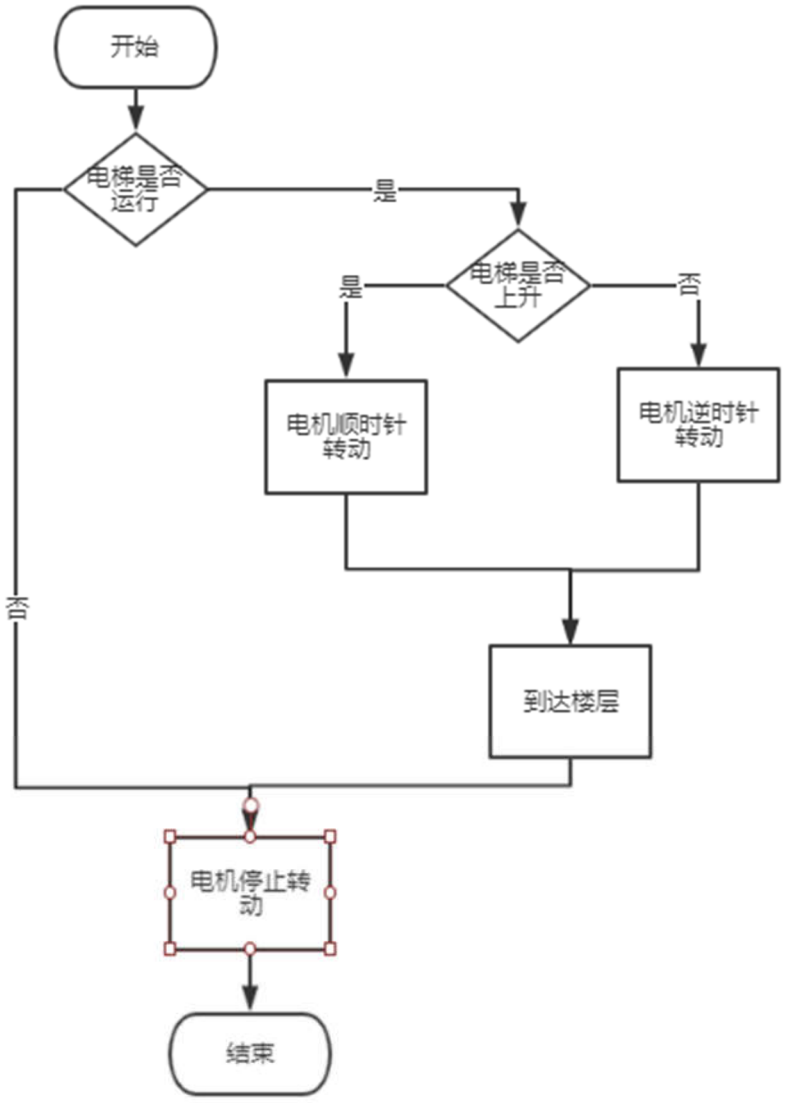


图4-8 步进电机模块程序流程图

4.5.2 系统功能描述

PB4 接步进电机的 BA，PB5 接 BB，PB6 接 BC，PB7 接 BD。定义自变量 buff 初始值为 00110011，当电机顺时针转动的时候，buff 中的每一位都进行左移，如果溢出的那位为 1 则在最右边重新补进一个 1，从而构成一个 00110011——10011001 这样的循环。逆时针转动就进行右移。再把循环的每一步数据输入步进电机接口，即可达到步进电机正反转功能。

再单独另外设置一个步进电机停止的函数，写入 11111111 即可停止电机。

4.4.3 运行界面截图

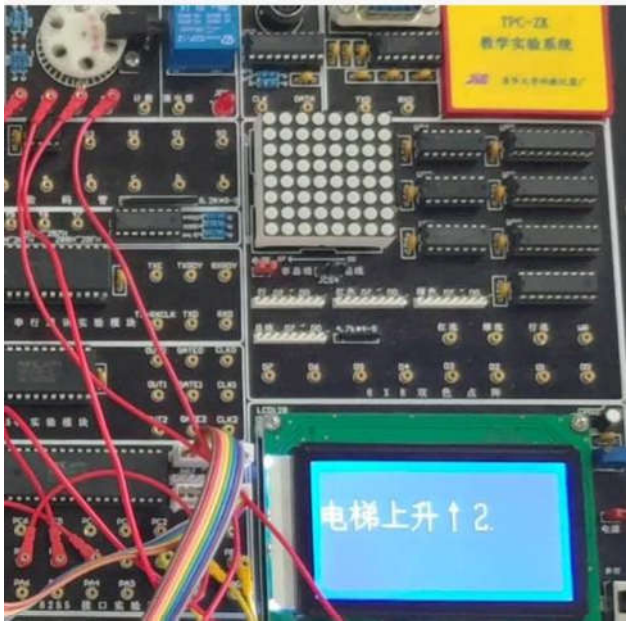


图4-9　电梯上升步进电机正转

5.　系统测试

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**错误!未找到引用源。**，单元测试用例如**错误!未找到引用源。**所示。

表5.1　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	alarm()类（**错误!未找到引用源。**）	打开蜂鸣器	0x00	DAC 清零	✓
2			0x0f	蜂鸣器开始响“滴”	✓
3		关闭蜂鸣器	0x00	蜂鸣器停止响声	✓
3	**错误!未找到引用源。**	ADC0809 模数转换	PortReadByte(0x298,&data)	随着直流信号的变化，data 的数值也不断变化	✓
4		超载显示	overweight[14], loseweight[16]	超载提示信息显示	✓

序号	测试类	测试内容	测试数据或方法	期望结果	测试结果
7	错误!未找到引用源。	初始化	initPortAndArray()	正常	✓
8	错误!未找到引用源。	检测按键按下	pressKey()	识别按键按下	✓
9		键值读取	getKey(int data)	识别键值	✓
10	错误!未找到引用源。	显示	当前楼层界面	正常显示	✓
11			超载界面	正常显示	✓
12			运行界面	楼层数获取正常	✓
13			开门界面	正常显示	✓
14			关门界面	正常显示	✓
17	步进电机类	正反转	zhuan(false)	电机正转	✓
			zhuan(true)	电机反转	✓
18		停止转动	stop_dianji()	电机停转	✓

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所示。

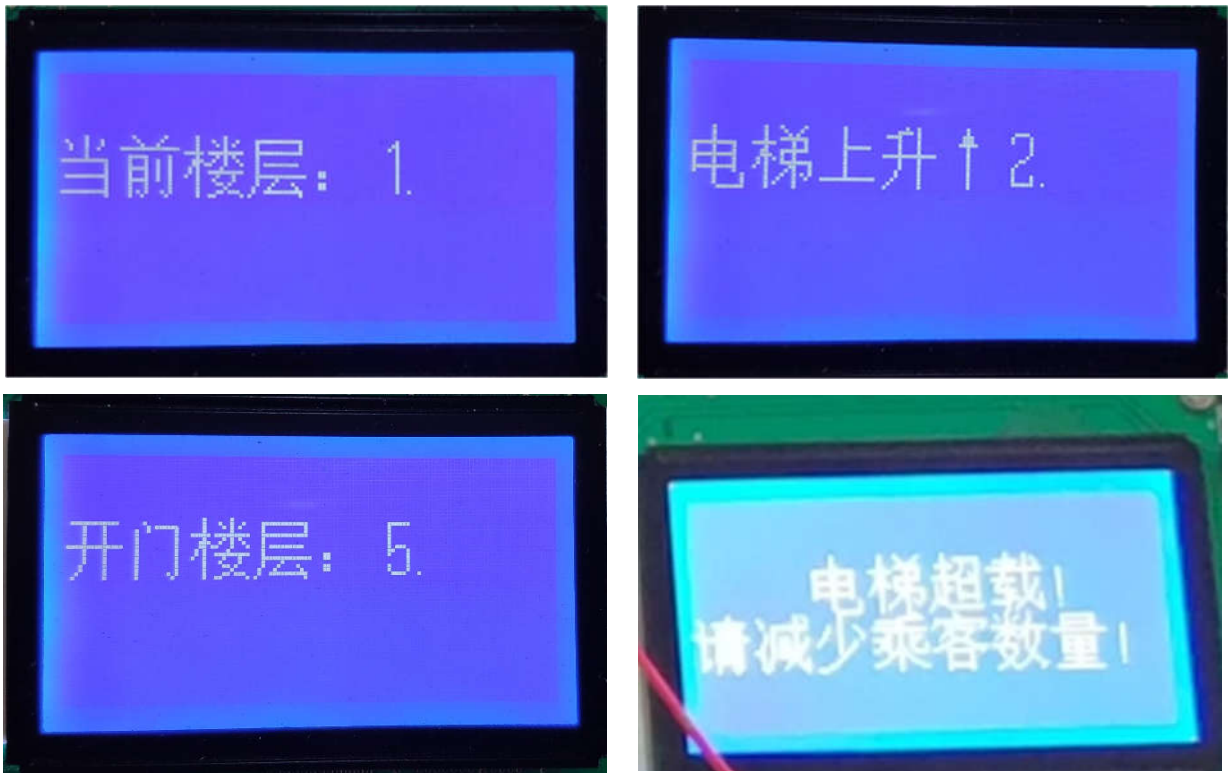


图5-1　液晶屏界面显示实物图

5.2 集成测试（系统测试）

在各模块实现的基础上，编写**错误!未找到引用源。**，整合各模块，并测试系统，整个系统测试用例见表 4。

表5.2　系统测试用例（状态转移测试）

序号	电梯状态	测试操作	期望结果	测试结果
1	初始状态	等待	LCD显示当前楼层号	✓
2	超载检测	任意按键	不响应	✓
3		旋转直流信号旋钮（非零）	LCD显示超载提示信息	✓
4			蜂鸣器报警	✓
5		旋转直流信号旋钮（为零）	转入等待状态	✓
6				
7	等待	按键选择楼层号	正常响应	✓
8		等待	转入运行状态	✓
9	运行	任意按键	不响应	✓
10		等待	LCD 显示楼层变化	✓
11		等待	步进电机正转（或反转）	✓
12		等待	开关门	✓

6.　系统设计结果及结论

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。

结果分析：设计出的电梯能够较为流畅地完成运行，自动进行就近目标楼层的判断，先向最近目标楼层运行，再一次到达所有目标楼层。电梯运行过程中步进电机进行正反转动，电梯停下则步进电机停下。每到一个目标楼层后会自动开门，开门秒之后自动关门，或此时用户有再一次开门需求，程序也可以响应。

但是在程序运行过程中，使用较多的软延时，在延时的时候较难接收到键盘的按下，给程序流畅度造成了一定的影响，以及在按下第一个按键后，要快速按下另一个按键才能接收到，才能进行就近楼层判断，系统设计时考虑的不周全。

7.　设计体会

通过《硬件课程设计》这门课，我**收获颇丰：**一是对上学期所学微机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。

通过这次课程设计，发现自己的在微机原理和 C 语言方面的很多不足，自己原先的知识有很多漏洞，对以前所学过的知识理解得不够深刻，掌握得不够牢固，通过这次课程设计之后，

把以前所学过的知识重新温故了一遍。从拿到题目到具体设计，从理论到实践，在这段时间里，虽然遭遇了一些挫折，但是收获了很多，不仅巩固了以前所学过的知识，而且学到了很多在书本上所没有学到过的知识。经过亲自上手的实践，了解到了自己的实践经验还是比较缺乏，理论联系实际的能力还急需提高。

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] 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.

[6] 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

```
/*C 口插键盘

 B 口低四位 LCD 屏控制  一位连灯光

 B 口高四位 电机控制

 A 口 LCD 数据写入 */
#include <stdio.h>
#include <conio.h>
#include "ApiExusb.h"
#pragma comment(lib, "ApiExusb.lib")

#define Max(a,b) ((a>b)?a:b)
#define Min(a,b) ((a<b)?a:b)

int keyword[8]={119, 123, 125, 126, 183, 187,189,190};//0  1 2 3 4 5 开门 关门
char lcd2[16]={0xb5, 0xb1, 0xc7, 0xb0, 0xc2, 0xa5, 0xb2, 0xe3, 0xa3, 0xba};//当前楼层=

```

```
19. char lcdup[10]={0xb5, 0xe7, 0xcc, 0xdd, 0xc9, 0xcf, 0xc9, 0xfd, 0xa1, 0xfc};//电梯上升（箭头）
20. char lcddown[10]={0xb5, 0xe7, 0xcc, 0xdd, 0xcf, 0xc2, 0xbd, 0xb5, 0xa1, 0xfd};//电梯下降（箭头）
21. 
22. char lcdNum[10]={0xa2, 0xb1, 0xa2, 0xb2, 0xa2, 0xb3, 0xa2, 0xb4, 0xa2, 0xb5};//12345
23. 
24. char overweight[14] = {0xa1,0xa0,0xa1,0xa0,0xb5, 0xe7, 0xcc, 0xdd, 0xb3, 0xac, 0xd4, 0xd8, 0xa3, 0xa1};//电梯超载!
25. char loseweight[16] = {0xc7,0xeb,0xbc,0xf5,0xc9,0xd9,0xb3,0xcb,0xbf,0xcd,0xca,0xfd,0xc1,0xbf,0xa3,0xa1};//请减少乘客数量!
26. 
27. char open[4]={0xbf,0xaa,0xc3,0xc5};//开门
28. char close[4]={0xb9,0xd8,0xc3,0xc5};//关门
29. 
30. char clearlcd[16] = {0xa1,0xa0,0xa1,0xa0,0xa1,0xa0,0xa1,0xa0,0xa1,0xa0,0xa1,0xa0,0xa1,0xa0,0xa1,0xa0};
31. 
32. int port_8255a, port_8255b, port_8255c, port_8255ctl;
33. 
34. bool vis[6];
35. int door=0;
36. int now = 1;
37. 
38. // 获取当前数组中的最大值，如若没有就是-1
39. int getMax()
40. {
41.     int res = -1;
42.     for(int i=0; i<6; i++) {
43.         if(vis[i])
44.             res = Max(i, res);
45.     }
46.     return res;
47. }
48. 
49. // 获取当前数组中的最小值，如若没有就是-1
50. int getMin()
51. {
52.     int res = 6;
53.     for(int i=0; i<6; i++) {
54.         if(vis[i])
55.             res = Min(i, res);
```

```
    }
    if(i == res)
        return -1;
    return res;
}

// 初始化端口
void initPortAndArray()
{
    port_8255a = 0x288;
    port_8255b = 0x289;
    port_8255c = 0x28a;
    port_8255ctl = 0x28B;
    for(int i=0; i<6; i++)
        vis[i] = 0;
}

// 获取按键的键值
int getKey(int data)
{
    for(int i=0; i<8; i++) {
        if(keyword[i] == data) {
            return i;
        }
    }
    return -1;
}
// lcd128 写命令
void cmdsetup()
{
    PortWriteByte(port_8255b,0x00);//00000000B     ;PB1 置 0,pB0 置 0 （LCD I 端=0，W 端＝0）
    Sleep(1);
    PortWriteByte(port_8255b,0x04);//00000100B     ;PB2 置 1 （LCD E 端＝1）
    Sleep(1);
    PortWriteByte(port_8255b,0x00);//00000000B     ;PB2 置 0,（LCD E 端置 0）
    Sleep(1);
}
// lcd128 写数据
void datasetup()
{
    PortWriteByte(port_8255b,0x01);//00000001B PB1 置 0，PB0=1 （LCD I 端=1）
    Sleep(1);
```

```
98.     PortWriteByte(port_8255b,0x05);//00000101B PB2 置 1 （LCD E 端＝1）
99.     Sleep(1);
100.     PortWriteByte(port_8255b,0x01);//00000001B PB2 置 0,（LCD E 端＝0）
101.     Sleep(1);
102. }
103.
104. // lcd 清缓存 LCD 清除
105. void clear()
106. {
107.     PortWriteByte(port_8255a,0x0c);//设置 CLEAR 命令
108.     cmdsetup();//启动 LCD 执行命令
109. }
110.
111. void clearLcd()
112. {
113.     PortWriteByte(port_8255ctl, 0x80);
114.     clear();
115.     PortWriteByte(port_8255a,0x90);
116.     cmdsetup();
117.     Sleep(10);
118.     for(int i=0;i<16;i++)
119.     {
120.         PortWriteByte(port_8255a,clearlcd[i]);
121.         datasetup();
122.     };
123.
124.     PortWriteByte(0x288,0x88);
125.        cmdsetup();
126.        Sleep(10);
127.        for (i=0;i<16;i++)
128.           {
129.             PortWriteByte(0x288,clearlcd[i]);
130.             datasetup();
131.            };
132.
133. }
134.
135. //超载报警，蜂鸣器响延时的时间后关闭
136. void alarm()
137. {
138.     PortWriteByte(0x280,0x00);
139.     Sleep(10);
```

```
140.     PortWriteByte(0x280,0x0F);
141.     Sleep(1000);
142.     PortReadByte(0x280,0x00);
143. }
144.
145. //超载检测
146. void pressure()
147. {
148.     byte data;
149.     while(!kbhit())
150.     {
151.         PortWriteByte(0x298,0x00);//虚写启动
152.         Sleep(70);
153.         PortReadByte(0x298,&data);//读入数据
154.         printf("%d\n",data);
155.         if(data==0) break;
156.         else alarm();
157.
158.         PortWriteByte(port_8255ctl, 0x80);//控制字 1000 0000  A 口输出
159.         clear();//LCD 清除
160.         PortWriteByte(port_8255a,0x90);
161.         cmdsetup();//设定 DDRAM 地址命令
162.         Sleep(10);
163.         for(int i=0;i<14;i++)
164.         {
165.             PortWriteByte(port_8255a,overweight[i]);
166.             datasetup();
167.         };
168.         puts("电梯超载!");
169.         PortWriteByte(port_8255a,0x88);
170.         cmdsetup();//设定 DDRAM 地址命令
171.         Sleep(10);
172.         for (i=0;i<16;i++)
173.         {
174.             PortWriteByte(port_8255a,loseweight[i]);
175.             datasetup();
176.         };
177.         puts("请减少乘客数量！ ");
178.         Sleep(1000);
179.
180.     }
181. }
```

```
182.
183.  //关门
184. void closed()
185. {
186.     door=0;
187.     PortWriteByte(port_8255ctl, 0x80);
188.     clear();
189.     PortWriteByte(port_8255a,0x90);//PA 口 90H=1001 0000H 第二行
190.     cmdsetup();
191.     Sleep(10);
192.     for(int i=0;i<4;i++)
193.     {
194.         PortWriteByte(port_8255a,close[i]);//close[i]关门的 LCD 字符编码
195.         datasetup();
196.     };
197.     puts("关门");//puts()函数只能输出字符串，不能输出数值或进行格式变换
198.     Sleep(1000);//puts()函数用来向标准输出设备（屏幕）输出字符串并换行
199. }
200. //开门
201. void opend()
202. {
203.     door=1;
204.     PortWriteByte(port_8255ctl, 0x80);
205.     clear();
206.     PortWriteByte(port_8255a,0x90);
207.     cmdsetup();
208.     Sleep(10);
209.     for(int i=0;i<4;i++)
210.     {
211.         PortWriteByte(port_8255a,open[i]);
212.         datasetup();
213.     };
214.     puts("开门");
215.     Sleep(1000);
216. }
217.
218.
219. //显示器 显示当前楼层 如果为-1 则 显示失败 否则 显示成功
220. void printFloor(int now)
221. {
222.     PortWriteByte(port_8255ctl, 0x80);
223.     clear();
```

```
224.     PortWriteByte(port_8255a,0x90);
225.     cmdsetup();
226.     Sleep(10);
227.     for (int i=0;i<10;i++){
228.         PortWriteByte(port_8255a,lcd2[i]);//当前楼层=
229.         datasetup();
230.     };
231.     byte c1 = lcdNum[2* (now - 1)];
232.     byte c2 = lcdNum[2* now - 1];
233.     PortWriteByte(port_8255a,c1);//每个字中的 2 个字节自动结合 查找字模并显示字符
234.     datasetup();
235.     PortWriteByte(port_8255a,c2);
236.     datasetup();
237.     Sleep(10);
238.     printf("当前楼层 %d\n",now);
239. }
240.
241. // 返回按键值  如果为-1 则读取失败 否则成功返回按的键盘
242. int pressKey()
243. {
244.     if(!PortWriteByte(port_8255ctl, 0x81)) { //c 0-3 in   4-7 out
245.         puts("控制字没有写成功");
246.         return -1;
247.     }
248.     byte data;
249.     if(!PortWriteByte(port_8255c, 0x0F)) {
250.         printf("写 C 口数据失败\n");
251.         return -1;
252.     }
253.     if(!PortReadByte(port_8255c, &data)) {
254.         printf("读取数据失败\n");
255.         return -1;
256.     }
257.     if(data == 0x0F) {
258.           return -1;
259.     }
260.     byte col = data;//列
261.     if(!PortWriteByte(port_8255ctl, 0x88)) { //c 0-3 out   4-7 in
262.         puts("控制字没有写成功");
263.         return -1;
264.     }
265.     PortWriteByte(port_8255c, 0xf0);//PC 口高位是行
```

```
    byte row;//行
    PortReadByte(port_8255c, &row);
    data = row + col;
    int c = getKey(data);
    if(c == 6)
    {
    door=1;
    puts("按下开门");
    opend();
    return -1;
    }
    if(c == 7)
    {
    door=0;
    puts("按下关门");
    closed();
    return -1;
     }
    /*if(c == 0 ) {
    puts("楼层初始化");
    int now = 1;
    printFloor(now);
    return -1;
    } */
    if(c == -1||c == 0) {
        return -1;
    }
    printf("pressKey value:%d\n", c);
    return c;
}

//步进电机正反转
int buf = 0x33;//0011 0011H
void zhuan(bool flag)
{
    int cnt = 40;
    while(cnt--) {
        Sleep(20);
        if(flag) //右
           buf = ((buf&1)<<7)|(buf>>1);
        else //左
           buf = ((buf&128)>>7)|(buf<<1);
```

```
308.         PortWriteByte(port_8255b, buf);
309.     }
310. }
311. 
312. // 电机停止转动
313. void stop_dianji()
314. {
315.     PortWriteByte(port_8255b,0xff);
316. }
317. 
318. // 显示器输出从 from 到 to 的上升过程
319. void up_print(int from, int to)
320. {
321.     PortWriteByte(port_8255ctl, 0x80);
322.     int step = to - from;
323.     for(int i=0; i<step; i++) {
324.         clear();
325.         PortWriteByte(port_8255a, 0x90);
326.         cmdsetup();
327.         Sleep(10);
328.         for (int j=0; j<10; j++){
329.             PortWriteByte(port_8255a, lcdup[j]);
330.             datasetup();
331.         };
332.         int nxt = from+i+1;
333.         byte c1 = lcdNum[2* (nxt - 1)];
334.         byte c2 = lcdNum[2* nxt - 1];
335.         PortWriteByte(port_8255a,c1);
336.         datasetup();
337.         PortWriteByte(port_8255a,c2);
338.         datasetup();
339.         PortWriteByte(port_8255a,0x88);
340.         cmdsetup();
341.         Sleep(10);
342.         printf("当前楼层 %d\n", nxt);
343.         zhuan(false);
344.         Sleep(1000);
345.     }
346.     return ;
347. }
348. 
349. //从 from 到 to 的下降过程
```

```
350. void down_print(int from, int to)
351. {
352.     PortWriteByte(port_8255ctl, 0x80);
353.     int step = -(to - from);
354.     for(int i=0; i<step; i++){
355.         clear();
356.         PortWriteByte(port_8255a, 0x90);
357.         cmdsetup();
358.         Sleep(10);
359.         for (int j=0; j<10; j++){
360.             PortWriteByte(port_8255a, lcddown[j]);
361.             datasetup();
362.         };
363.         int nxt = from-i-1;
364.         byte c1 = lcdNum[2* (nxt - 1)];
365.         byte c2 = lcdNum[2* nxt - 1];
366.         PortWriteByte(port_8255a,c1);
367.         datasetup();
368.         PortWriteByte(port_8255a,c2);
369.         datasetup();
370.         PortWriteByte(port_8255a,0x88);
371.         cmdsetup();
372.         Sleep(10);
373.         printf("当前楼层 %d\n", nxt);
374.         zhuan(true);
375.         Sleep(1000);
376.     }
377.     return ;
378. }
379.
380. void main()
381. {
382.     if(!Startup()) {
383.         puts("启动失败，请检查设备情况");
384.         return ;
385.     }
386.     initPortAndArray();
387.
388.     now=1;//初始楼层
389.     printFloor(now);
390.     // 判断方向上升还是下降 为0 表示不动 为1表示上升  为-1表示下降
391.     int dir = 0;
```

```
392.     // 判断电梯是否在运行，刚开始没运行
393.     bool running = false;
394.     while(true)
395.     {
396.
397.         Sleep(1000);
398.         pressure();
399.         clearLcd();
400.         for(;;) {
401.
402.         // 定义按下的键值
403.         int nxt = pressKey();
404.         // 当前电梯没有运行下的情况
405.         if(running == false) {
406.             if(nxt == -1 || now == nxt) {
407.                 continue;
408.             }
409.             else {
410.                 vis[nxt] = 1;
411.                 dir = ((now > nxt) ? -1 : 1);
412.                 running = true;
413.                 if(dir > 0) {
414.                     up_print(now, now+1);
415.                     now++;
416.                 }
417.                 else if(dir < 0){
418.                     down_print(now, now-1);
419.                     now--;
420.                 }
421.             }
422.         } // 电梯当前正在运行过程中
423.         else {
424.             // 当前已经到目标了
425.             if(vis[now] == 1) {
426.                 vis[now] = 0;
427.                 stop_dianji();
428.                 printFloor(now);
429.                 printf("已经到达楼层%d 了\n", now);
430.                 opend();
431.                 Sleep(2000);
432.                 closed();
433.                 break;//添加
```

```
434.             }
435.             // 当前按键 可达
436.             if(nxt != -1 && nxt != now) {
437.                 vis[nxt] = 1;
438.             }
439.             // 电梯向上
440.             if(dir > 0) {
441.                 int mx = getMax();
442.                 // 当前没有需要响应的楼层了
443.                 if(mx == -1) {
444.                     running = false;
445.                     dir = 0;
446.                     stop_dianji();
447.                 }// 当前有比当前更高的楼层  因此继续往上
448.                 else if(mx != -1 && mx > now) {
449.                     running = true;
450.                     dir = 1;
451.                 }// 有比当前低的楼层
452.                 else if(mx != -1 && mx < now) {
453.                     running = true;
454.                     dir = -1;
455.                 }
456.             } //电梯向下
457.             else if(dir < 0) {
458.                 int mn = getMin();
459.                 // 当前没有需要响应的楼层了
460.                 if(mn == -1) {
461.                     running = false;
462.                     dir = 0;
463.                     stop_dianji();
464.                 } // 当前有比当前更小的楼层  因此继续往下
465.                 else if(mn != -1 && mn < now) {
466.                     running = true;
467.                     dir = -1;
468.                 } // 当前没有比当前更小的楼层了   但有比当前楼层高的楼层需要到达  因此往上走
469.                 else if(mn != -1 && mn > now) {
470.                     running = true;
471.                     dir = 1;
472.                 }
473.             }
474.             // 电梯正在上升
475.             if(dir > 0) {
```

```
476.                 up_print(now, now+1);
477.                 now++;
478.             } //电梯正在下降
479.             else if(dir < 0) {
480.                 down_print(now, now-1);
481.                 now--;
482.             }
483.         }
484.     }
485. }
486.     Cleanup();
487.     return ;
488. }
```