大连理工大学本科设计报告

题目：出租出计费系统设计

课程名称： 计算机原理实验

学 院： 信息与通信工程学院

专 业： 电子信息工程

班 级： 电信1801

学 号： 201883016

学生姓名： 杨题鸣

2020 年 12 月 28 日

1 设计要求

模拟出租车系统的功能，除了基本起价、里程计算、费用显示；以及开机音乐，费用清零，停止，直流电机转速可调，行驶方向显示。

具体要求如下：

1. 8253/8254对250kHz信号分频。

* TC0工作于方式3方波发生器模式，用于将250kHz信号分频到较低频率。
* TC1工作于方式2分频器模式，输出占空比可调的PWM信号。一方面用作中断触发输入，调用中断服务程序的执行；另一方面，经过非门作为直流电机的输入信号，通过占空比的调整改变直流电机的转速。
* TC2工作于方式3方波发生器模式，用于改变计数初值N产生特定频率的信号，实现音乐播放的功能。

2.六位共阴极LED数码管显示里程和费用。

六位数码管的高2位显示里程数（XX千米），低3位显示费用（显示XX.X元），两者之间的数码管显示“-”作为分隔。

3.系统初始状态的设定。

里程数00（km），低3位显示起步价（设置为10元）。2km内价格不变，超过2km，按每公里1.2元计价。

4. 出租车的速度大小同时由直流电机的转速和8\*8 LED点阵的闪烁频率表示。

5. 电位器控制出租车速度。通过ADC0809读取速度，对出租车里程、费用进行控制

6.键盘控制：

1. 8\*8 LED点阵显示‘大’,‘工’两个字

2. 键盘控制清零操作

2 设计分析及系统方案设计

程序由主程序和多个子程序组成，通过模块化的编程思路进行设计。

具体信息如下：

|  |  |  |
| --- | --- | --- |
| 模块名称 | 所用函数 | 实现功能 |
| 键盘模块 | Keyboard  Keyup  Cmpoffset  Delay | 用于键盘键入命令控制输入以用来完成命令输入1. 8\*8 LED点阵显示‘大’,‘工’两个字2. 键盘控制清零操作 |
| 音乐模块 | Clock  read\_state  PlayMusic  Sound | 根据sound\_flag判断是否播放音乐。当K6为高电平时，若sound\_flag对应的开关K5为高电平，则播放开机音乐；若sound\_flag对应的开关K5为低电平，则开机时静默。若K6为低电平，则直接开机。 |
| 里程费用模块 | int\_proc  AD\_transfer  read\_state  calculate\_m  calculate\_price  clear | 根据AD\_transfer读取得到的N值，控制TC1计数初值，从而改变产生中断的频率。8次中断为1km里程。calculate\_m和calculate\_price用于更新计算里程和费用。当read\_state读取到的clear\_flag有效时，通过clear函数重置里程和费用 |
| 直流电机模块 | AD\_transfer  machine | 根据AD\_transfer读取得到的N值，控制TC1计数初值（TC1同时控制中断和直流电机的转速），从而改变TC1输出的占空比，经过非门后接到直流电机，使直流电机获得不同的转速。 |
| 8\*8点阵模块 | int\_proc  disp88  disp0 | 在disp88中判断updown\_flag选择不同的显示图案，并调用disp0在8\*8LED点阵上显示相应图案。 |
| ６位数码管模块 | disp | 通过提取里程和费用的各个位的数字，调用相应的字符编码，通过对数码管的刷新完成对里程和费用的相应数字和字符的显示。 |

3 系统电路图

图示, 示意图

描述已自动生成

图1 系统电路图

4 外围接口模块硬件电路功能描述

1. 原件功能说明

|  |  |  |
| --- | --- | --- |
| 芯片/端口 | 基址值 | 功能 |
| 8253/8254 | 200H | 计数器0工作于方式3，对1MHz信号进行分频。计数器1工作于方式2，用于产生中断和控制直流电机。计数器2工作于方式3，对1MHz信号分频，播放音乐。 |
| 8255 | 210H | 并行接口芯片：模式0，PA、PB为输出，控制8\*8 LED点阵；pc口第四位->键盘的行；pc口高四位->键盘的列 |
| ADC0809 | 220H | 读取电位器的电压，将其转换为数字量，用于控制速度。 |
| 6位数码管 | 240H | 显示里程和费用 |

1. 原件引脚图

图片包含 表格

描述已自动生成

图 8254/8253 硬件电路

图示, 示意图

描述已自动生成

图3 8255硬件电路

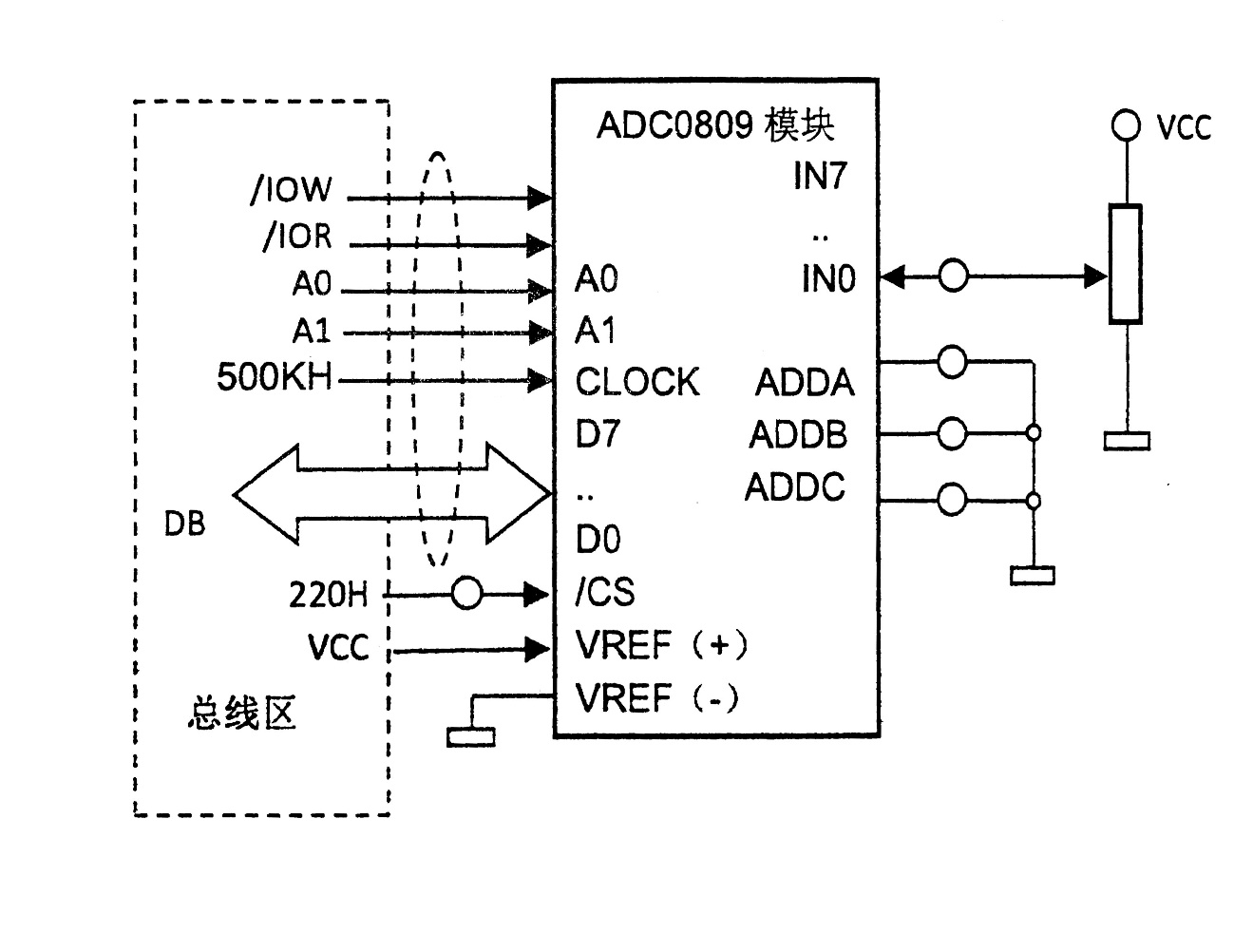


图4 ADC0809硬件电路

5 主程序中主要变量说明

|  |  |  |  |
| --- | --- | --- | --- |
| 变量名称 | | 变量类型 | 功能 |
| ledcode\_1 | | db | 6为数码管显示字符的编码，不带小数点 |
| ledcode\_2 | | db | 6为数码管显示字符的编码，带小数点 |
| ledcode\_3 | | db | 6为数码管显示”-”字符的编码 |
| count\_km | | db | 里程（km） |
| price | | dw | 价格（角） |
| N | | db | 用于赋值到TC1的计数初值 |
| clear\_flag | | db | 清零标志。1：重置里程价格。0：不重置。 |
| km\_flag | | db | 里程更新标志。1：对里程+1，并更新费用。 |
| updown\_flag | | db | 8\*8 点阵显示标志。1：上箭头。0：下箭头 |
| led3 | | db | 8\*8 点阵上箭头的编码。 |
| led4 | | db | 8\*8 点阵下箭头的编码。 |
| count | | db | disp函数中间变量，刷新显示8\*8点阵计数 |
| bz | | db | disp函数中间变量 |
| ShowProgress\_flag | | db | 更新进度显示标志。1：更新。0：不更新 |
| progress\_0 | | db | 进度显示函数用到的初始值，为00000001b。 |
| sound\_flag | | db | 音乐播放标志。1：播放音乐。0：不播放。 |
| mus\_freq | | dw | 存储音乐每个字符的频率。 |
| mus\_time | | dw | 存储音乐每个字符的时间。 |
| Get\_key | db | | 用于存放按键按下的按钮大小以确定按下的元素 |

6 系统软件中各个子程序的功能描述

|  |  |  |  |
| --- | --- | --- | --- |
| 子程序名称 | 入口参数 | 出口参数 | 功能描述 |
| KEYBOARD | 无 | 无 | 用于调用4\*4键盘的主函数 |
| CMPOFFSET | 无 | Get\_key | 用于最后求得键值 |
| READKEY | 无 | Ah，al | 用于扫描键盘，查看按下的键都有哪些 |
| KEYUP | 无 | 无 | 判断按键是否已经释放 |
| int\_proc | 中断服务子程序 | 中断服务子程序 | 中断服务程序。保护现场，之后进行状态读取和8\*8 LED点阵显示，以及１km内进度更新和里程费用更新的判断和执行，并根据AD转换结果对TC1计数初值重新赋值（最小值为2）。最后发中断结束EOI命令，恢复现场，开中断并返回。 |
| clock | 无 | 无 | 对8254的TC0，TC1和TC2进行初始化。 |
| AD\_transfer | 无 | N | 对电位器上的电压进行AD转换，并将读取结果放缩到预定的范围内（0~12）。 |
| read\_state | 无 | clear\_flag  updown\_flag  sound\_flag | 通过8255读取系统控制信息（即K3、K4、K5三个开关的电平信息），并根据其电平高低对相应标志（clear\_flag、updown\_flag、sound\_flag）赋值。 |
| calculate\_m | count\_km | count\_km | 计算里程count\_km，8个方波信号后，里程数增加1km。里程最大值设定为99km，到达99km后里程将重置为0。 |
| calculate\_price | count\_km | price | 根据里程count\_km计算费用price。计算方式为2km以内10元（100角），超过2km之后12角/千米。由于显示价格最低位为角，故计算价格时单位为角。 |
| machine | N | 无 | 进行直流电机控制。对AD\_transfer读取的N（0~12）设定最小值为2（即2~12），并将N作为TC1的计数初值，以调节TC1输出信号的占空比，从而控制直流电机的转速。硬件上需要将TC1的输出接非门后控制直流电机。 |
| disp | count\_km  price | 无 | 控制六位数码管进行显示，高两位显示时需要通过状态来控制。需要注意，在显示下一个字符前，要将当前显示清零，否则产生鬼影。硬件电路采用CPU模式，接上片选端即可。 |
| delay | 无 | 无 | 延时函数，辅助数码管显示的正常运行。 |
| PlayMusic | sound\_flag | 无 | 根据sound\_flag判断是否播放音乐。首先对TC1赋值255，使直流电机模拟齿轮转动之后遍历mus\_freq和mus\_time得到单个音符的频率和时间，并调用sound函数演奏单个音符。当频率为-1时，播放结束。 |
| sound | mus\_freq  mus\_time | 无 | 用于演奏一个字符。计算分频数N的公式为：输入频率f0 =音符频率fs \*分频数N。通过上式计算得到N，赋值给TC2，并设置一段时间的延时。每个音符的频率由mus\_freq和mus\_time给定。 |
| clear | 无 | 无 | 将里程和费用重置。里程重置为0km，费用重置为100角。 |
| disp88 | updown\_flag  led3  led4 | 无 | 控制8\*8 LED点阵显示。通过判断updown\_flag显示向上箭头或向下箭头。 |
| disp0 | led3  led4 | 无 | 根据给定的编码，刷新并显示向上箭头或向下箭头（由updown\_flag决定）。 |

7 主程序程序流程图

8 程序清单

;--------------------------------------------------------------------

; 带键盘出租车系统设计

;---------------------------------------------------------------------

.model small

.386

data segment

;---------------------设置各端口地址---------------------------

;8254基址值 --> 计时器

i8254\_0 equ 200H

;8255端口地址 --> 中间接口

i8255\_a equ 210h ;8255A口，初始化PA为输出

i8255\_b equ 211h ;8255B口，初始化PB为输入

i8255\_c equ 212h ;8255C口

i8255\_k equ 213h ;8255控制口

AD\_IO equ 220h

led\_io equ 240h ;LED灯地址

;---------------------设置led编码-------------------------------

ledcode\_1 db 3fh,06h,5bh,4fh,66h,6dh,7dh,07h,7fh,67h ;不带. --> 指小数点

ledcode\_2 db 0bfh,86h,0dbh,0cfh,0e6h,0edh,0fdh,87h,0ffh,0e7h ;带. --> 指小数点

ledcode\_3 db 40h ;字符"-"

;--------------------设置变量信息--------------------------------

count\_km db 00h ;路程（km）

price dw 100 ;起步价定为100角（10元）

N db 0

clear\_flag db 0 ;为1则表示清零

km\_flag db 0 ;为7则表示已达到1km，此时更新km和price，并将km\_flag清零

;8\*8点阵

updown\_flag db 0

led3 db 8ch,0cch,6ch,3fh,3fh,6ch,0cch,8ch

led4 db 0c3h,0c3h,0c3h,0ffh,0ffh,0c3h,0c3h,0c3h

count dw 00h

bz dw ?

ShowProgress\_flag db 0

progress\_0 db 00000010b

;音乐模块

sound\_flag db 0

mus\_freq dw 262,294,330,262

dw 262,294,330,262

dw 330,349,392,330

dw 349,392,392,440

dw 392,349,330,262

dw 392,440,392,349

dw 330,262,294,196

dw 262,294,196,262,-1

mus\_time dw 10 dup(25),50,25,25,50,4 dup(12),25,25

dw 4 dup(12),4 dup(25),50,25,25,50

GET\_KEY DB 00H

counttt db 00h

data ends

ssg segment stack

dw 100 dup(0)

ssg ends

code segment

assume cs:code,ds:data,ss:ssg

main proc far

mov ax, data

mov ds, ax

;--------------------各端口初始化----------------

cli ;关中断进行初始化

call clock ;初始化8245

;--------------------初始化8255--------------------

mov dx,i8255\_k

mov al,88h ;8255控制字

out dx,al ;8255为模式0，PA、PB为输出，PC高位为输入，低位为输出

;--------------设置386EX主8259中断屏蔽字-------------

in al,21h

;开放IR3、IR5中断，IR3是里程清零，IR5是正常计数740的产生的中断

and al,11010111b

out 21h,al

;---------------------设置中断向量表----------------------

;设置IR5

push ds

mov ax, 0

mov ds, ax ;ds指向00000H，为中断向量表的地址范围的起始地址

;中断向量表的IP值

lea ax, cs:int\_proc1 ;ax为中断程序入口地址的偏移地址IP

mov si, 35h ;类型码35h

add si, si ;类型码\*4得到中断向量

add si, si ;找到中断向量表中终端类型码35h对应的位置的起始地址

mov ds:[si], ax ;将中断服务程序的IP放到中断类型码的对应位置

;中断向量表的CS值

push cs

pop ax ;ax <-- 中断服务程序的段地址cs

mov ds:[si+2], ax ;将中断服务程序的cs放到中断类型码的对应位置

pop ds

;---------------------音乐模块---------------------

call PlayMusic

sti ;开中断

;---------------------等待中断----------------------

LLL:

call machine ;重复刷新直流电机

CALL KEYBOARD

call read\_state ;读取状态

;判断清零标志，clear\_flag不为0则初始化count\_km和price

cmp clear\_flag, 00010000b

jne next12

call clear

next12:

cli ;关中断执行disp

call disp ;重复刷新LED显示

sti ;执行disp结束，开中断

jmp LLL

main endp

;---------------------中断时钟设置-----------------

;子函数功能：对TC0、TC1、TC2初始化

clock proc

;计数器 0 的 CW 和 N

mov dx, i8254\_0

add dx, 3 ;指向控制口

mov al, 00110110b ;计数器0，方式3（方波方式），读写先低后高，二进制计数

out dx, al

mov dx, i8254\_0 ;指向计数器0

mov ax, 1000 ;赋初值N

out dx, al

mov al, ah

out dx, al

;计数器 1 的 CW 和 N

mov dx, i8254\_0 ;向8254写控制字

add dx, 3

mov al, 01010100b ;计数器1，方式2（分频器），只写低

out dx, al

mov dx, i8254\_0

add dx, 1 ;指向计数器1

mov al, 2 ;在N属于[0,12]下，使N初始化为2

out dx, al

;计数器 2 的 CW

mov al, 10110110b ;8254初始化TC2，读写16位，方式3方波模式

mov dx,i8254\_0

add dx, 3

out dx,al

ret

clock endp

;---------------------开机音乐播放-----------------

PlayMusic proc

push ax

push dx

push si

push di

;使得直流电机模拟齿轮转动

mov al, 255

mov dx, i8254\_0

add dx, 1

out dx, al

call read\_state

cmp sound\_flag, 01000000b

jne next\_test\_music

;输出音乐

;获得频率和时间的首地址

lea si, mus\_freq

lea di, mus\_time

play:

mov dx, [si]

cmp dx, -1 ;频率为-1时结束播放

je end\_play

call sound ;sound函数用于演奏一个字符

add si, 2 ;指向下一个数据

add di, 2

call read\_state ;允许在歌曲中间打断歌曲，下次歌曲依旧从头播放

cmp sound\_flag, 01000000b

jne sound\_exit

jmp play

end\_play:

mov ax, 2

mov dx, i8254\_0

add dx, 2

out dx, al ;先送低8位到计数器，dx是8253芯片通道2的端口地址

mov al, ah

out dx, al ;后送高8位计数器

next\_test\_music:

pop di

pop si

pop dx

pop ax

ret

PlayMusic endp

;-----------------AD转换并计算速度函数---------------

AD\_transfer proc

mov dx, AD\_IO ;ADC启动转换地址

out dx, al ;启动ADC

mov dx, AD\_IO ;获取ADC EOC状态地址

add dx, 2

L1:

in al, dx

test al, 01h ;测试EOC是否为1

jz L1

mov dx, AD\_IO ;读数据地址

inc dx

in al, dx

mov ah, 0

mov cl, 21

div cl ;获得0-12档，2档最快，12档停止

mov N, al ;对于设定N最小值的操作，在N被使用时进行

ret

AD\_transfer endp

;---------------------进度显示子函数-------------------

progress proc

push ax

mov al, progress\_0 ;读取progress\_0到al

dec al ;得到要显示的进度

mov dx,i8255\_c

out dx, al ;将al输出到下C口

rol progress\_0, 1 ;progress\_0循环左移，为下一次做准备

cmp progress\_0, 00010000b ;1从第四位左移到高四位，则将progress\_0还原为初始值

jb next\_progress

mov progress\_0, 00000001b

next\_progress:

pop ax

ret

progress endp

;----------------中断服务子程序IS5---------------

int\_proc1 proc far

push ax

push dx

call AD\_transfer ;读取电位器对应的N

call disp

call disp88 ;调用点阵显示函数

inc counttt

cmp counttt,10

jne continue\_int\_proc1

mov counttt,0

add km\_flag, 1

cmp km\_flag, 4

jne no\_update ;km\_flag为8时，表示已经运行1KM，需更新count\_km和price

;更新km和price

call calculate\_km

call calculate\_price

mov km\_flag, 0 ;更新完毕，对km\_flag清0

no\_update:

cmp ShowProgress\_flag,10000000B

jne continue\_int\_proc1

call progress ;更新km进度

continue\_int\_proc1:

;对计数器1重新赋初值

mov dx, i8254\_0

add dx, 1 ;指向计数器1

cmp N, 2

jae next\_int\_proc1

mov N, 2 ;设置最快速度，高低电平至少各0.5秒

next\_int\_proc1:

mov al, N ;赋值新的N值

out dx, al

;发送中断结束命令 EOI

mov al,20h

out 20h,al ;（向386EX主8259）

pop dx

pop ax

sti ;开中断

iret

int\_proc1 endp

;----------------演奏音符函数---------------

;sound函数：演奏一个音符

;入口参数：si - 要演奏的音符的频率的地址

; di - 要演奏的音符的音长的地址

sound proc

push ax

push dx

push cx

;{dx, ax}作为被除数， {dx, ax}是TC2的输入频率

;基于一个公式：输入频率250kHz = 分频数ax（TC2的N）\* 声音频率（分频后的频率）

;call read\_state ;允许在歌曲中间打断歌曲，下次歌曲依旧从头播放

;cmp sound\_flag, 01000000b

;jne sound\_exit

mov dx,0003h ;TC2输入频率为250kHz

mov ax,0d090h

;[si]作为除数

div word ptr [si] ;计算分频值,赋给ax。[si]中存放声音的频率值。

;将ax作为TC2的N，输出到TC2端口

mov dx, i8254\_0

add dx, 2

out dx, al ;先送低8位到计数器，dx是8253芯片通道2的端口地址

mov al, ah

out dx, al ;后送高8位计数器

mov dx, [di] ;保持[di]时长

wait11:

mov cx, 14000

delay\_sound:

nop

loop delay\_sound

dec dx

jnz wait11

pop cx

pop dx

pop ax

sound\_exit:

ret

sound endp

;-------------------6位数码管显示函数------------------

disp proc

push bx

push dx

;km显示

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

lea bx, ledcode\_1

mov al, count\_km

mov ah, 0

mov cl, 10 ;截取十位

div cl

mov ah, 0

div cl

mov al, ah

mov ah, 0

add bx, ax

mov al, [bx] ;取得要显示的数据

mov dx, led\_io ;获取显示的地址

out dx, al ;输出数据

mov al, 100000b ;指明要点亮的位为第一位

inc dx ;地址加1

out dx, al

call delay

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

lea bx, ledcode\_1

mov al, count\_km

mov ah, 0

mov cl, 10 ;截取个位

div cl

mov al, ah

mov ah, 0

add bx, ax

mov al, [bx]

mov dx, led\_io

out dx, al

mov al, 010000b

inc dx

out dx, al

call delay

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

;显示"-"字符

mov al, ledcode\_3

mov dx, led\_io

out dx, al

mov al, 001000b

inc dx

out dx, al

call delay

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

;显示价格

lea bx, ledcode\_1

mov ax, price

mov cl, 10 ;截取百位

div cl

mov ah, 0

div cl

mov ah, 0

div cl

mov al, ah

mov ah, 0

add bx, ax

mov al, [bx]

mov dx, led\_io

out dx, al

mov al, 000100b

inc dx

out dx, al

call delay

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

lea bx, ledcode\_2 ;带小数点

mov ax, price

mov cl, 10 ;截取十位

div cl

mov ah, 0

div cl

mov al, ah

mov ah, 0

add bx, ax

mov al, [bx]

mov dx, led\_io

out dx, al

mov al, 000010b

inc dx

out dx, al

call delay

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

lea bx, ledcode\_1

mov ax, price

mov cl, 10 ;截取个位

div cl

mov al, ah

mov ah, 0

add bx, ax

mov al, [bx]

mov dx, led\_io

out dx, al

mov al, 000001b

inc dx

out dx, al

call delay

;清除鬼影

mov al, 0

mov dx, led\_io

out dx, al

mov al, 000000b

inc dx

out dx, al

call delay

pop dx

pop bx

ret

disp endp

;-----------------里程计算函数------------------

;子函数功能：该程序在IR5被响应时调用，表示740个clk计数完毕，对应公里数加1

calculate\_km proc

push ax

mov al, 1

add count\_km, al

cmp count\_km, 99

jb no\_clear

mov count\_km, 0

mov price, 100

no\_clear:

pop ax

ret

calculate\_km endp

;-----------------价格计算函数------------------

calculate\_price proc

mov al, count\_km

cmp al, 2 ;2km为起始距离

ja bigger

mov price, 100 ;起始价为10

jmp exit1

bigger:

sub al, 2

mov dl, 12

mul dl

add ax, 100

mov price, ax ;存储结果，用于显示

exit1:

ret

calculate\_price endp

;----------------状态读取并显示函数---------------

read\_state proc

AAAA:

mov al,GET\_KEY

cmp al,8

jne BBBB

mov clear\_flag, 00010000b ;读取clear\_flag状态，为D4位

BBBB:

mov al,GET\_KEY

cmp al,9

jne CCCC

mov updown\_flag, 00100000b ;读取updown\_flag状态，为D5位

CCCC:

mov al,GET\_KEY

cmp al,10

jne DDDD

mov sound\_flag, 01000000b ;读取sound\_flag状态，为D4位

DDDD:

mov al,GET\_KEY

cmp al,11

jne AAAA0

mov ShowProgress\_flag, 10000000b ;读取sound\_flag状态，为D4位

AAAA0:

mov al,GET\_KEY

cmp al,12

jne BBBB0

mov clear\_flag, 00000000b ;读取clear\_flag状态，为D4位

BBBB0:

mov al,GET\_KEY

cmp al,13

jne CCCC0

mov updown\_flag, 00000000b ;读取updown\_flag状态，为D5位

CCCC0:

mov al,GET\_KEY

cmp al,14

jne DDDD0

mov sound\_flag, 00000000b ;读取sound\_flag状态，为D4位

DDDD0:

mov al,GET\_KEY

cmp al,15

jne enddddd

mov ShowProgress\_flag, 00000000b ;读取sound\_flag状态，为D4位

enddddd:

ret

read\_state endp

;-----------------里程清零函数------------------

;子函数功能：当clear\_flag为1时被调用，清零count\_km

clear proc

mov count\_km, 0

mov price, 100

ret

clear endp

;-------------------直流电机函数------------------

;原理，通过方式2的分频，导致占空比不同，N越大，占空比越小（需经过非门）

machine proc

mov al, N

mov ah, 0

cmp al, 2

jae next\_machine

mov al, 2 ;设置最快速度，高低电平至少各0.5秒

next\_machine:

mov dx, i8254\_0

add dx, 1 ;指向TC1

out dx, al ;al为2~12，2最快，12停止

ret

machine endp

;---------------8\*8点阵显示---------------

disp88 proc

push cx

lea di, led3 ;设di为字形码缓冲区指针（指向首地址）, LED3为上升箭头

call read\_state

cmp updown\_flag, 00100000b

je next\_disp88

lea di, led4 ;设di为字形码缓冲区指针（指向首地址）, LED3为下降箭头

next\_disp88:

mov cx, 0005h ;点阵扫描次数

call disp0 ;调用扫描子程序

pop cx

ret

disp88 endp

;--------------点阵子函数--------------

disp0 proc

push bx

push ax

push dx

loop5:

mov count, 0000h ;计数器初始=0

mov bh, 01h ;产生最高位扫描码

mov al, 0 ;扫描码消失（熄灭）

mov dx, i8255\_b ;关闭扫描信号

out dx, al ;LED灯熄灭

lll0:

mov byte ptr bz,bh

push di

add di,count

mov bl,byte ptr[di]

pop di

mov al,bl

mov dx,i8255\_a ;8255a控制行

out dx,al

mov al,byte ptr bz

mov dx,i8255\_b ;8255b控制列，显示bz列

out dx,al

inc count

push cx

mov cx,20h ;延迟一段时间

delay0:

loop delay0

pop cx

mov al,0

mov dx, i8255\_b ;关闭扫描信号

out dx,al

mov bh,byte ptr bz

shl bh,1 ;bh指向下一位

jnz lll0 ;8列都扫描完结束

loop loop5

pop dx

pop ax

pop bx

ret

disp0 endp

;------------------键盘------------------

KEYBOARD PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

START1:

CALL READKEY ;读取键盘(AH，AL作为出口)

MOV BH,AH ；AH=行

MOV BL,AL ；AL=列

CALL DELAYKEY ；延迟

CALL KEYUP ；等待释放后再处理

MOV AX,BX

CALL CMPOFFSET ；求得(get\_key的值)

CALL DELAYKEY

POP DX

POP CX

POP BX

POP AX

RET

KEYBOARD ENDP

READKEY PROC

LOOP1:

MOV AH,11111110B

SCAN:

MOV AL,AH

MOV DX,I8255\_C ；输出一行线为低

OUT DX,AL

MOV DX,I8255\_C

IN AL,DX ；输出列状态

OR AL,00001111B ；高四位为列

push CX ；交换高四位，低四位的位子

MOV CL,4

ROR AL,CL

pop cx

CMP AL,0FFH ；次行有按键按下吗？

JNE EXITKEY

ROL AH,1 ；没有检查下一行

CMP AH,11101111B

JZ LOOP1

JMP SCAN

EXITKEY:

RET

READKEY ENDP

DELAYKEY PROC ；延迟

PUSH CX

MOV CX,8000H

X11: LOOP X11

POP CX

RET

DELAYKEY ENDP

KEYUP PROC

NOUP:

MOV AL,AH

MOV DX,I8255\_C ；输出行线

OUT DX,AL

MOV DX,I8255\_C

IN AL,DX ；已读行线

OR AL,00001111B

push CX ；交换高四位，低四位的位子

MOV CL,4

ROR AL,CL

pop cx

CMP AL,0FFH

JE EXIT

JMP NOUP

EXIT:

RET

KEYUP ENDP

CMPOFFSET PROC

NOT AH ;行值取反

NOT AL ；列值取反

MOV BH,00H

MOV BL,00H

KK1:

SHR AH,1

JC NEXT11

ADD BH,4H ；一行4列

JMP KK1

NEXT11:

SHR AL,1

JC NEXT22

ADD BL,1 ；列号加1

JMP NEXT11

NEXT22:

ADD BH,BL ；求得键值对

MOV AL,BH

MOV GET\_KEY,AL ；付值给get\_key

CMPEXIT:

RET

CMPOFFSET ENDP

;------------------延时函数------------------

delay proc

push cx

mov cx, 0FFH

X1:

loop X1

pop cx

ret

delay endp

code ends

end main

9 系统调试运行结果说明、分析所出现得问题，设计体会与建议

软硬件环境:window7

设计语言:汇编ASM

调试过程中遇到的主要问题和解决方法：

1.问题：8255的利用率开始不够高

解决办法：用上键盘。高四位作为列输出，低四位作为行输入。使得8255的利用率达到了100%

2.问题：使用键盘后，led数码管出现闪烁熄灭的状态

解决办法：把数码管加入到中断程序里面，实现跟着中断的节奏显示，由于人眼的识别速度跟不上实际的闪烁效果，就形成了常亮状态。

3. 鬼影现象的解决。

一开始的时候，6位数码管显示由鬼影的现象。具体体现为这一位显示的字符会叠加很淡的下一位显示字符。

通过检查，产生鬼影的原因是在切换显示位置时，没有对显示进行清除，导致在代码执行的显示下一字符代码的时间内，待显示变量仍为上一字符的编码值，导致串位，造成鬼影。

在显示字符的结束后，清除编码值，鬼影消失。

10.结论及设计体会

由于时间紧迫，无法实验非常完美的代码编程，代码当中还存在许多的问题，没有时间解决。尽管实验就要结束了，但是这一次计算机原理给我留下的回忆确实终身的回忆，总之，路漫漫其修远兮，吾将上下而求索