

计算机工程学院

**汇编语言与接口技术实践**

**报告书**

学年学期： **2022 - 2023 学年 第1学期**

所选题目： **题目0娱乐游戏机**

分组成员情况：

组 长 班级：计算2114 学号：202121331104 姓名： 庄佳强

组员1班级：计算2114 学号：202121331108 姓名： 范少军

组员2班级：计算2114 学号：202121331116 姓名： 广睿

成绩：

评语：

**0. 小组成员分工**

**组长：** 完成游戏主体部分，连接组员实现的个个子程序模块，参与测试实现效果，寻找BUG。

**组员1：**重点完善整体代码，实现小功能，添加诸多细节，撰写实验报告。

**组员2：**重点完成显示屏的编程，参与测试实现效果，寻找BUG。

**贡献度：组长： 40% 组员1： 30% 组员2：30%**

**1. 设计要求**

基于 51 开发板，利用独立键盘作为按键输入，将 3 个数码管显示不同字符，另外 4 个数码显示得分。通过按键启动，让数码管字符以不同速度变化。按键暂停后，3 个数码管显示的字符相同得最高分，根据显示字符给出不同分数。

（1）按键实现投币加分功能，每次游戏消耗一定分值，分值耗尽游戏结束。

（2）综合利用开发板的 LED、蜂鸣器、数码管等器件，自行设计游戏的开始和结束动画。（可结合跑马灯实验）

（3）游戏中得到不同分数奖励时，开发板显示不同的奖励动画（原理同上）。

**2. 设计思路**

**如何实现基础功能：**先设置好全局变量，调用不同地址存储不同内容，游戏开始前先进行液晶屏的显示，再进行开始检测轮询，直至按下开关K3进入游戏。先调用数码管改变程序，显示七个数码管，前三个表示分数，第四个表示投币数，后三个表示数码管变化字符。判断开关K2是否按下，实现投币数量的加一。再判断开关K1是否按下，暂停三个数码管的变化，并根据具体要求，获取得分在表示分数的数码管上显示。不断循环主程序，直至判断开关K4按下，进入死循环结束程序。

**自己增加的功能点：**

1. 通过液晶屏显示游戏的各种动画，并通过显示屏进行对游戏的开始和结束进行提示，在游戏中得到不同的奖励分数时，显示屏显示不同的奖励动画。
2. 实现一键将硬币数加到最大值，方便用户体验。

**流程图：**

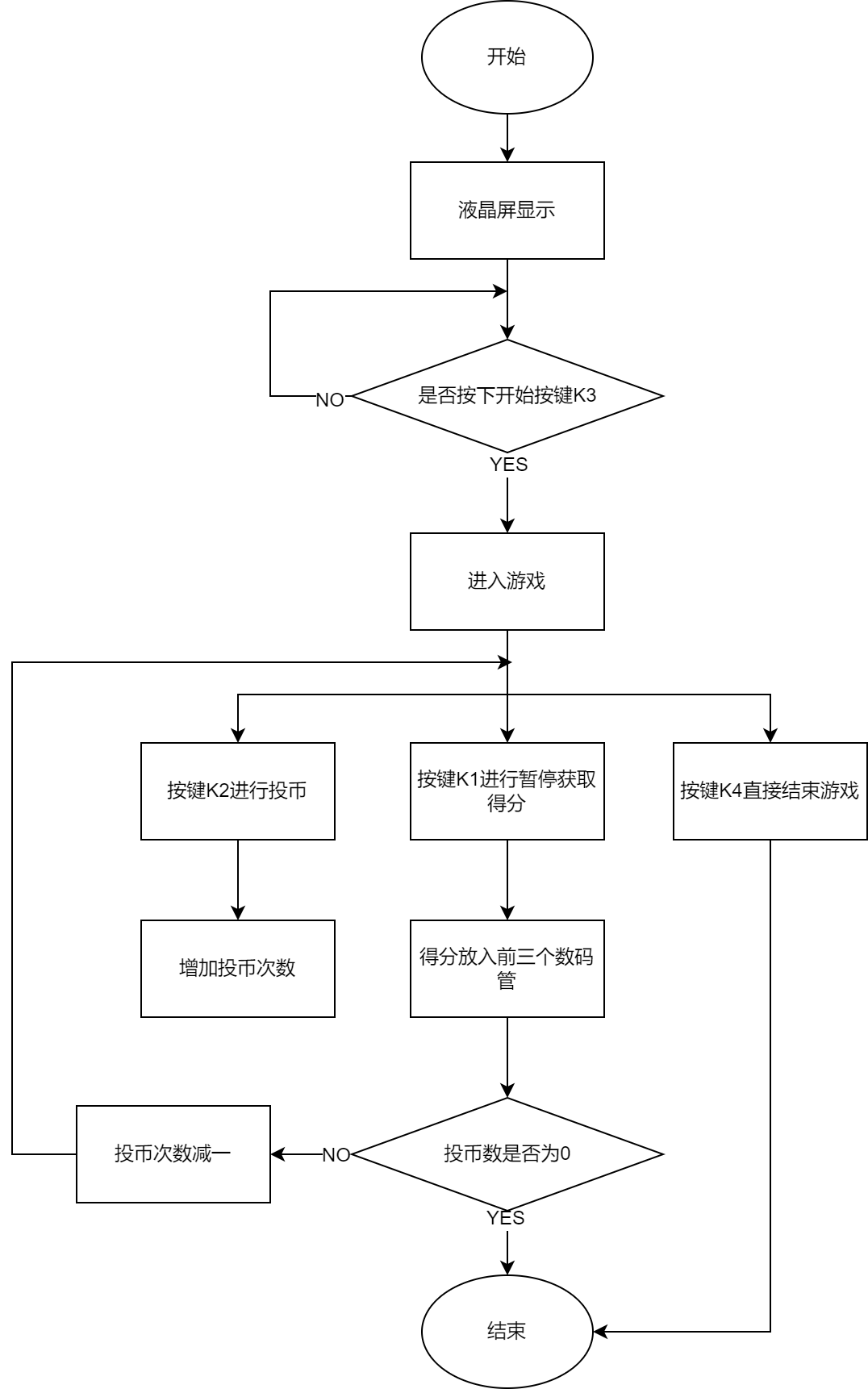
****

图1 总流程图

开始按钮对应流程图：

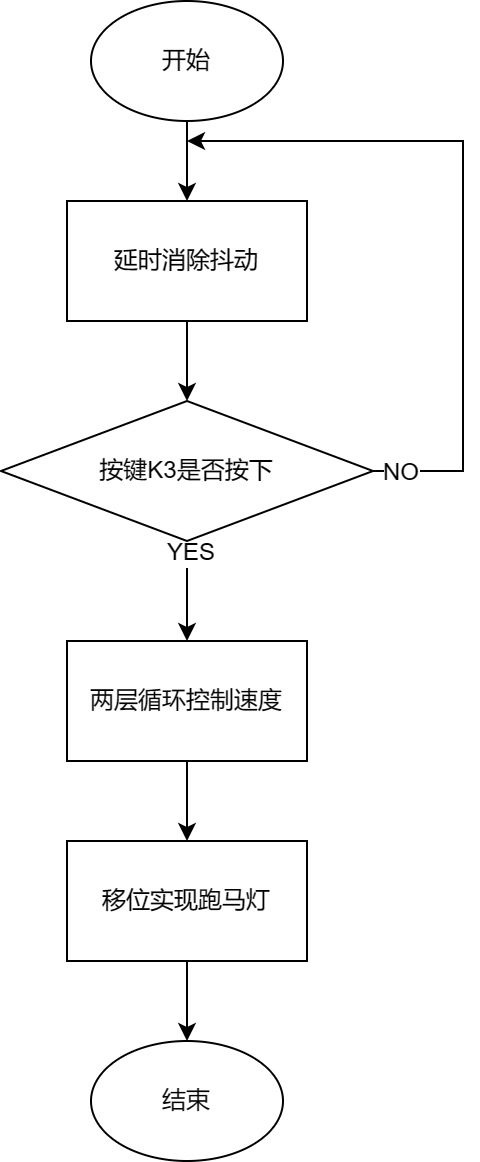


图2 开始按钮图

加币按钮对应流程图：

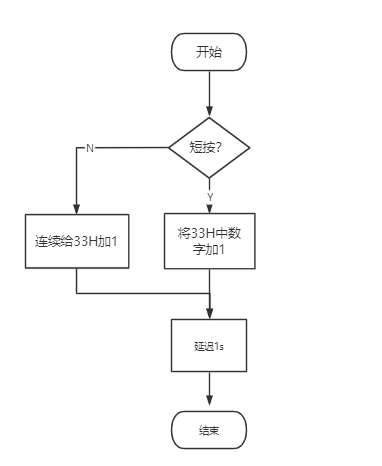


图3 加币按钮图

游玩按钮对应流程图：

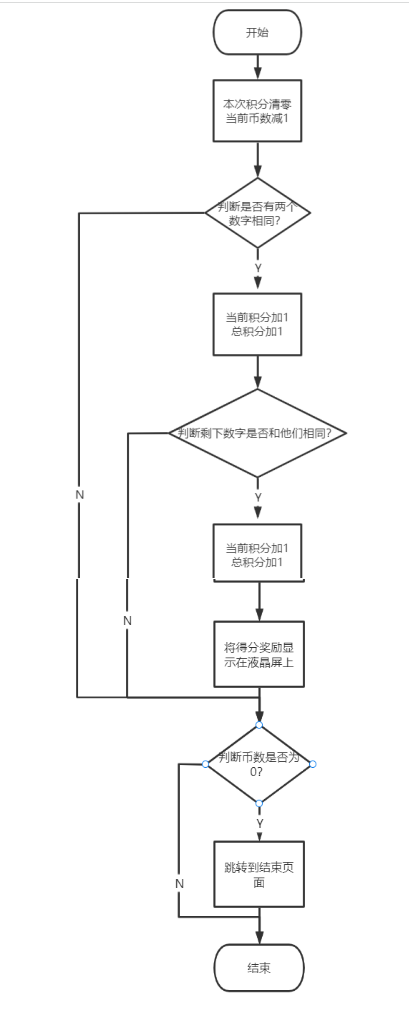


图4 游玩按钮图

结束按钮对应流程图：

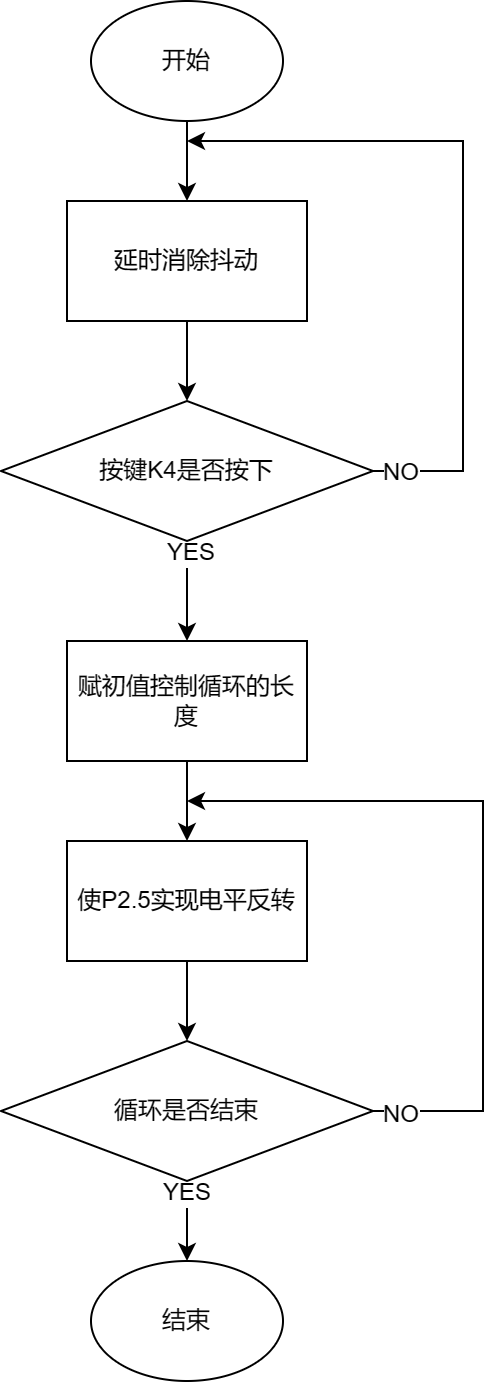
****

图5 结束按钮图

液晶屏显示：

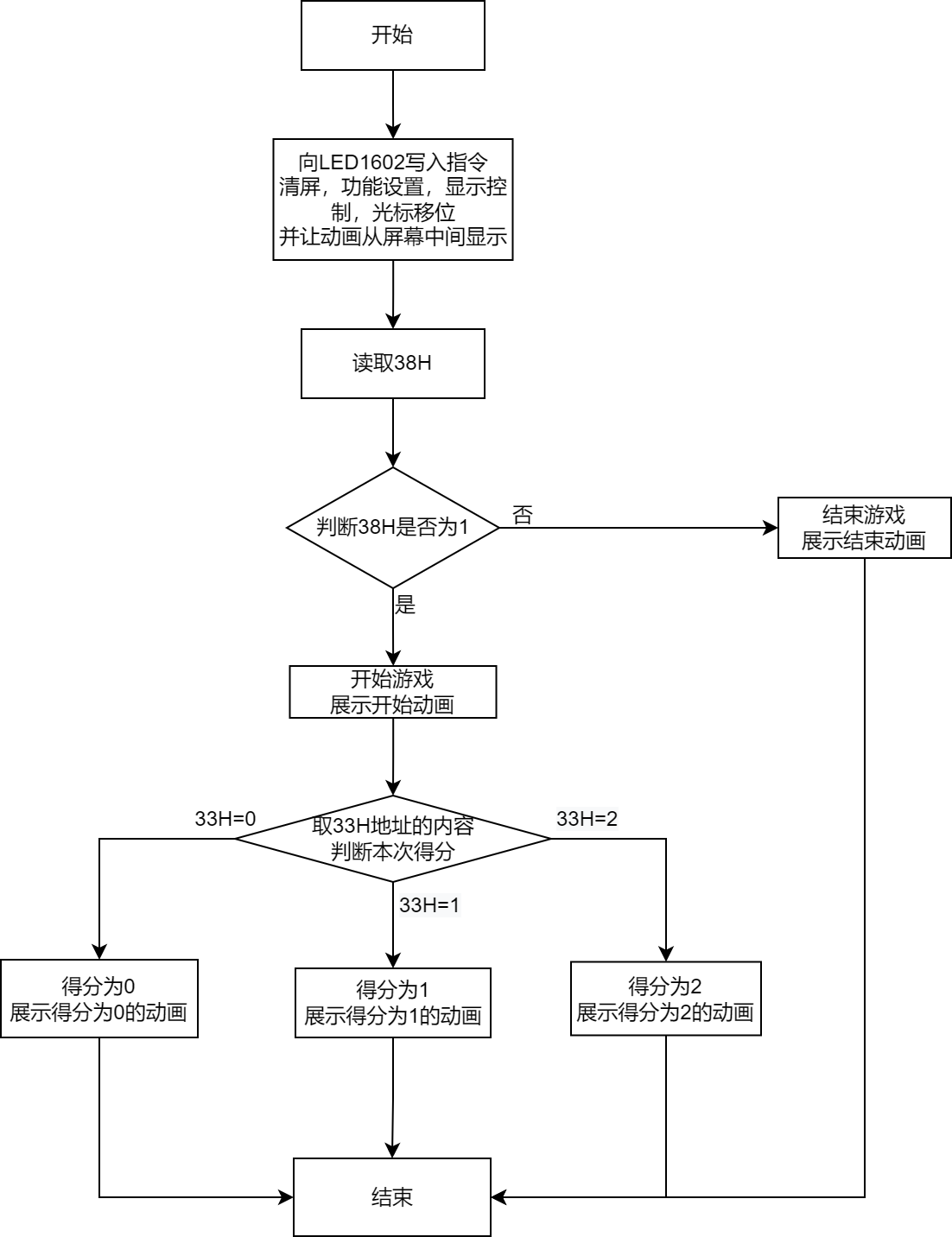
****

图6 液晶屏显示图

数码管数字的变化：

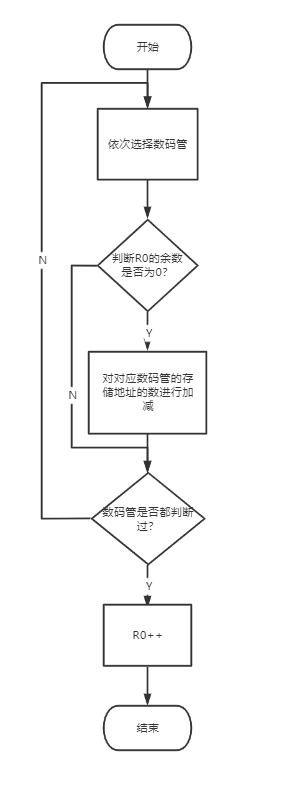
****

图7 数码管变化图

**相关电路图：**

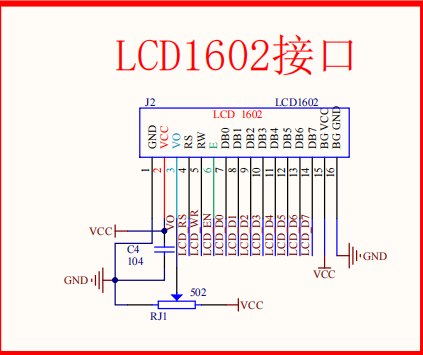


图8 LCD1602接口电路图

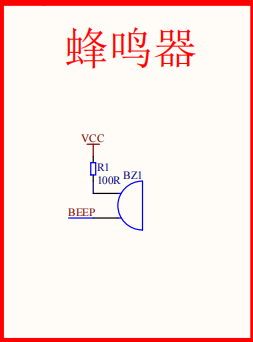


图9 蜂鸣器电路图

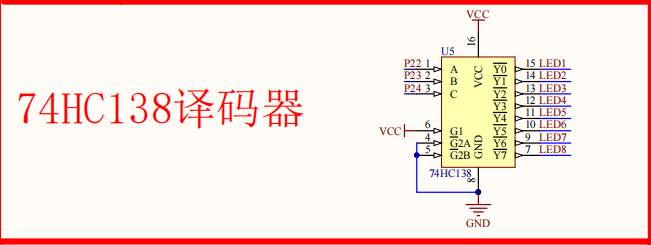


图10 译码器电路图

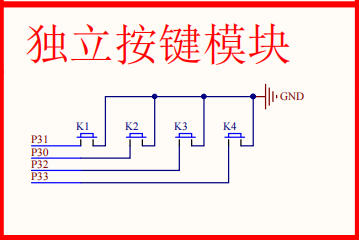


图11 独立按键模块电路图

**3. 实现效果**

**操作说明手册：**按键K1用于暂停三个变化得字符，三个字符全部相同为2分，两个相同为1分，无相同为0分。按键K2用于投币加分，短按加1，长按可以增快加1的速度，投币次数表示可以进行的游戏次数，投币数为0时自动结束游戏。按键K3用于开始游戏。按键K4用于结束游戏。开始和结束动画由跑马灯和液晶屏构成。游戏得分在最前面的三个数码管和液晶屏中显示。

**实物效果图：**

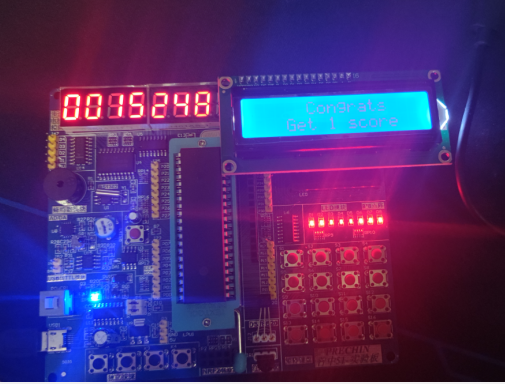


图12 实验效果图

**4. 总结**

**实验困难：**

1. 实现长按投币功能时，循环次数不能放在寄存器R5、R6、R7中，因为在调用延时程序时会改变这三个寄存器的值。
2. 在实现主体部分的数码管显示不同数字更新的时候是最难的。一开始是把更新融入到显示程序中，但这样根本没法正常的显示。后来想到先更新在显示，把更新的程序单独写到一个子程序中，更新完存储到字节地址后在显示出来，从而解决问题。
3. 实现数字的随机性也是一个巨大的难题。通过讨论后得出先设置一个1递进的数，然后把它除3,2,1 ，当余数为0,时，对应的数码管加1，从而实现不错的伪随机随机性。

**实验收获：**

通过实验更加了解蜂鸣器的原理，长按和短按的区分和液晶屏的显示方法，学习到了更多的汇编知识。并通过小组成员的互相讨论，完善整个游戏的实现功能，提高了团结协作的能力。

1. **代码**

RS EQU P2.6

RW EQU P2.5

E EQU P2.7

LCD EQU P0

BF EQU P0.7

ORG 0000H

MOV DPTR,#TABLE

MOV R0,#0 //设置一个全局累加变量，用于除法取余

MOV 34H,#9 //初始化数码管内容

MOV 33H,#1 //币的存储位置，开始币为1

MOV 38H,#1 //游戏开始检测存储字节地址，为1时表示游戏未开始 AJMP MAIN //

AJMP MAIN

ORG 0100H

MAIN:

ACALL MAIN1 //游戏开始前进行液晶屏的显示

ACALL MAIN0 //进行开始检测轮询

START:

ACALL Change //调用数码管改变程序

JNB p3.0,AddCoinContinue

ACALL D125ms

JNB p3.1,Scores

JNB P3.3,EndContinue //结束

AJMP START

Scores:

ACALL D10ms

JNB p3.1,Scoress

AJMP START

Scoress:

MOV 28H,#0 //当前分数位减1

DEC 33H //币数减1

MOV A,34H

CLR C

SUBB A,35H

JZ SUBB37 //第一个数字和第二个数字相同时跳转

MOV A,34H

CLR C

SUBB A,36H

JZ SUBB36 //第一个数字和第三个数字相同时跳转

MOV A,35H

CLR C //进位清零

SUBB A,36H

JZ SUBB35 //第二个数字和第三个数字相同时跳转

AJMP detect //都没有跳转到检测子程序

AddCoinContinue: //因为target out of range 所以利用中转程序中转跳转

AJMP AddCoin

continue: //因为target out of range 所以利用中转程序中转跳转

AJMP START

EndContinue: //因为target out of range 所以利用中转程序中转跳转

AJMP END1

SUBB37:

INC 28H //当前积分数加1

INC 29H //总积分处加1

MOV A,34H

CLR C

SUBB A,36H //判断三个是否相同

JZ all //为零调转到再加一的子程序

AJMP assignment //否则跳转到分配子程序

SUBB36:

INC 28H

INC 29H

MOV A,34H

CLR C

SUBB A,35H //判断三个是否相同

JZ all

AJMP assignment

SUBB35:

INC 28H

INC 29H

MOV A,35H

CLR C

SUBB A,34H

JZ all

AJMP assignment

all: //再加一

INC 28H //当前积分加1

INC 29H //总积分加1

assignment: //分配

MOV A,29H

MOV B,#100 //分离百位

DIV AB

MOV 30H,A

MOV A,B

MOV B,#10 //分离十位和个位

DIV AB

MOV 31H,A

MOV A,B

MOV 32H,A

detect:

ACALL DisPlay

JNB p3.1,detect //循环检测按键是否松开，让用户可以清楚看见自己的按下数字

ACALL MAIN1 //显示本次得分奖励

MOV 28H,#0 //当前积分清空

MOV A,33H

CJNE A,#0,continue //没有币直接结束

AJMP END2

AddCoin:

MOV 20H,#25

ACALL D10ms //延时子程序消除抖动

JNB P3.0,Addcoin0

AJMP START

AddCoin0:

DEC 20H

JB P3.0, AddCoin1

ACALL D10ms

MOV R5,20H

DJNZ R5, AddCoin0

CJNE R5, #00H, AddCoin1

XUNHUAN:

INC 33H

MOV A, 33H

CJNE A, #16, XUNHUAN1

DEC 33H

XUNHUAN1:

ACALL D10ms

ACALL D10ms

ACALL D10ms

JNB P3.0, XUNHUAN

AJMP START

AddCoin1:

INC 33H //币数加1

MOV A, 33H

CJNE A, #16, AddCoin2

DEC 33H

AddCoin2:

AJMP START

MAIN0:

MOV p0,#0

JNB P3.2, START1 //开始

AJMP MAIN0

START1:

ACALL MAIN1 //显示得分

ACALL DD10ms //延时子程序消除抖动

MOV R7, #08

MOV P2, #0FEH

JNB P3.2, START2 //K3控制开始

AJMP MAIN0

START2:

MOV R6, #200

START3:

MOV R5, #200 //两层循环控制跑马灯的速度

START4:

DJNZ R5, START4

DJNZ R6, START3

MOV A, P2

RL A //移位实现跑马灯

MOV P2, A

DJNZ R7, START2

ACALL MAIN1

AJMP START

END1: MOV 33H,#0

ACALL MAIN1

LCALL DD10ms //延时子程序消除抖动

JNB P3.3, END2 //再次判断K4是否按下

AJMP START

END2:

MOV P2, #18H

MOV R7, #07 //赋初值控制循环的长度

END3:

MOV P0, #00H

MOV A, P2

SUBB A, #04H

MOV P2, A

DJNZ R7, END3 //使P2.5实现电平反转

AJMP END2 //使程序不断进行死循环

Change: //数字改变

PUSH PSW //现场保护

PUSH Acc

MOV R2,#0 //初始化R2为0

Change1:

MOV A,R2

CLR C //清除进位，防止SUBB时出错

SUBB A,#0

JZ NUM0 //A为零时表示当前在对于P2口,跳转到对应口

CLR C

SUBB A,#1

JZ NUM1

CLR C

SUBB A,#1

JZ NUM2

INC R0

POP Acc

POP PSW

RET

NUM0:

MOV A,R0 //把全局变量赋值到A

MOV B,#3 //除数赋值到B

DIV AB //相除

MOV A,B //把余数赋值到A

JZ NUM00 //如果被3整除 跳转到加一子程序

AJMP NUM000 //否则跳转到跳转下一个的子程序

NUM00: //加一

INC 36H //第三个数码管显示的数字加一

MOV A,36H

CJNE A,#10,NUM000 //如果不到10就跳转到跳转下一个子程序

MOV 36H,#0 //归零

NUM000://跳转下一个

MOV A,R2

ADD A,#1 //转移到下一个数码管数字字节

MOV R2,A

AJMP change1

NUM1:

INC 35H //

MOV A,35H

CJNE A,#10,NUM111

MOV 35H,#0

NUM111:

MOV A,R2

ADD A,#1

MOV R2,A

AJMP Change1

NUM2: //同理

MOV A,R0

MOV B,#2 //除2为0时跳转

DIV AB

MOV A,B

JNZ NUM22

AJMP NUM222

NUM22:

DEC 34H

MOV A,34H

CJNE A,#-1,NUM222

MOV 34H,#9

NUM222:

MOV A,R2

ADD A,#1

MOV R2,A

AJMP Change1

DELAY: //数码管显示延迟

MOV R5,#20

DEL0: MOV R6,#1

DEL1: MOV R7,#10

DEL2: DJNZ R7,DEL2

DJNZ R6,DEL1

DJNZ R5,DEL0

RET

DisPlay: //动态显示数码管子程序

PUSH PSW

PUSH Acc

MOV R2,#4 //初始化R2为0

MOV R1,#36H //R1初始化存储地址

MOV R3,#07H

STARTS:

MOV A,R2

MOV P2,A

ADD A,#4

MOV R2,A

MOV A,@R1

DEC R1

MOVC A,@A+DPTR

MOV p0,A

LCALL DELAY

DJNZ R3,STARTS

POP Acc

POP PSW

RET

D500ms: //延时500ms 使用3次循环

PUSH PSW

PUSH Acc

MOV 40H,#100

D12:

MOV 41H,#5

D3:

MOV 42H,#100

D4:

DJNZ 42H,D4

DJNZ 41H,D3 //((100\*2+4)\*5+3500)\*100约500`000ms=0.5s

ACAll Display //在延迟中反复刷新显示，防止数码管的显示消失

DJNZ 40H,D12

POP Acc

POP PSW

RET

D125ms: //延时125ms 使用3次循环

PUSH PSW

PUSH Acc

MOV 40H,#25

DD12:

MOV 41H,#5

DD3:

MOV 42H,#100

DD4:

DJNZ 42H,DD4

DJNZ 41H,DD3 //((100\*2+4)\*5+3500)\*25约125000ms=0.125s

ACAll Display //在延迟中反复刷新显示，防止数码管的显示消失

DJNZ 40H,DD12

POP Acc

POP PSW

RET

D10ms: //延时10ms

PUSH PSW

PUSH Acc

MOV 40H,#5

D0: MOV 41H,#4

D2:

MOV 42H,#250 //250\*2\*4\*5=10000纳秒=10ms

D1:

DJNZ 42H,D1

DJNZ 41H,D2

LCALL DisPlay //区别在此循环时在进入游戏之后，需要显示数码管内容

DJNZ 40H,D0

POP Acc

POP PSW

RET

DD10ms: //延时10ms

PUSH PSW

PUSH Acc

MOV 40H,#5

DD0: MOV 41H,#4

DD2:

MOV 42H,#250 //250\*2\*4\*5=10000纳秒=10ms

DD1:

DJNZ 42H,DD1

DJNZ 41H,DD2

DJNZ 40H,DD0

POP Acc

POP PSW

RET

MAIN1: PUSH PSW //LED液晶显示器

PUSH Acc

PUSH DPH

PUSH DPL

MOV 00H,#0

MOV A,#01H //O1H为清屏操作命令(命令1)

ACALL W\_CMD //向LCD1602写入命令01H

MOV A,#38H //38H为功能设置命令(命令6)

ACALL W\_CMD //向LCD1602写入命令38H

MOV A,#0CH //0CH为显示控制指令(命令4)

ACALL W\_CMD //向LCD1602写入命令0CH

MOV A,#14H //14H为光标移位指令(命令5)

ACALL W\_CMD //向LCD1602写入命令14H

MOV A,#83H //83H(80H+03H)显示位置设置(命令8)，第一行从03H开始

ACALL W\_CMD //向LCD1602写入命令83H

MOV R0,0 //寄存器R0为显示字符个数的计数器

// MOV 34H,#1

MOV A,38H

CJNE A,#1,GG //

MOV 38H,#0

MOV DPTR,#TAB1

SJMP LOOP1

GG: MOV A,33H

CJNE A,#0,W1

MOV DPTR,#TAB2

SJMP LOOP1

W1: //MOV 30H,#2

MOV A,28H //取本次加的分

CJNE A,#0,SORE1

MOV DPTR,#TAB3

SJMP LOOP1

SORE1: CJNE A,#1,SORE2

MOV DPTR,#TAB4

SJMP LOOP1

SORE2: MOV DPTR,#TAB5

SJMP LOOP1

LOOP1: MOV A,R0

MOVC A,@A+DPTR //显示字符的ASCII写入A

ACALL W\_DATA //显示字符的ASCII写入LCD1602

INC R0 //字符计数器

CJNE R0,#10,LOOP1 //第一行显示10个字符是否发送完毕

MOV A,#0C0H //C0H(80H+40H)显示扫描设置(命令8)，第2行从40H

ACALL W\_CMD

LOOP2: MOV A,R0

MOVC A,@A+DPTR //将显示字符的ASCII码送入LCD1602

ACALL W\_DATA

INC R0

CJNE R0,#27,LOOP2 //判断第二行16个字符是否发送完毕

LOOP: POP DPL

POP DPH

POP Acc

POP PSW

RET

WAIT: MOV LCD,#0FFH //检测1602是否处于忙状态

CLR RS //将RS请0

SETB RW //将RW置1

CLR E //产生上升沿信号，使读数据指令生效

NOP

SETB E

JB BF,WAIT //检测忙标志BF，BF=1则循环等待

RET

W\_CMD: ACALL WAIT //写入命令子程序

MOV LCD,A

CLR RS //将RS,RW清0

CLR RW

SETB E //令E段产生下降沿,命令写入LCD1602

NOP

CLR E

RET

W\_DATA: ACALL WAIT //写入数据子程序

MOV LCD,A

SETB RS //将RW置1

CLR RW //将RW置0

SETB E //令E端产生下降沿,使数据写入LCD1602

NOP

CLR E

RET

TAB1: DB'Welcome to'

DB' Game starts '

TAB2: DB' Game Over'

DB' You Win '

TAB3: DB' Regret '

DB' No score '

TAB4: DB' Congrats'

DB' Get 1 score '

TAB5: DB' Well done'

DB' Get 2 score '

TABLE:

DB 3FH, 06H, 5BH, 4FH, 66H, 6DH //0 1 2 3 4 5

DB 7DH, 07H, 7FH, 6FH //6 7 8 9 A b

DB 77H, 7cH, 39H, 5eH, 79H, 71H

END