

单片机课程设计报告

目录

1.	设计的系统的目的、用途、功能.....	3.....
1.1.	硬件设计思想和电路原理图.....	3.....
1.2.	红外线发射接收模块.....	4.....
1.3.	棋盘显示模块.....	4.....
1.4.	单片机接口.....	5.....
1.5.	硬件单元的使用.....	5.....
2.	软件设计思想及软件流程.....	6.....
2.1.	主函数.....	6.....
2.2.	按键输入模块.....	7.....
2.3.	游戏算法模块.....	8.....
2.4.	显示模块.....	9.....
3.	详细说明软件功能.....	9.....
3.1.	显示模块.....	9.....
3.2.	按键模块.....	9.....
3.3.	黑白棋规则模块（ <code>check_chess</code> 函数）	9.....
3.4.	胜利模式（ <code>victory</code> 函数）	9.....
3.5.	主函数.....	10.....
3.6.	源程序.....	10.....
4.	系统测试过程及测试数据.....	22.....
4.1.	硬件测试：	22.....
4.2.	软件测试单独：	22.....
4.3.	系统与软件综合测试：	22.....
5.	测试数据：	22.....
6.	分析相应的指标参数.....	22.....
7.	所需的全部资源.....	23.....
8.	成员分工和工作情况.....	错误!未定义书签。.....

1. 设计的系统的目的、用途、功能

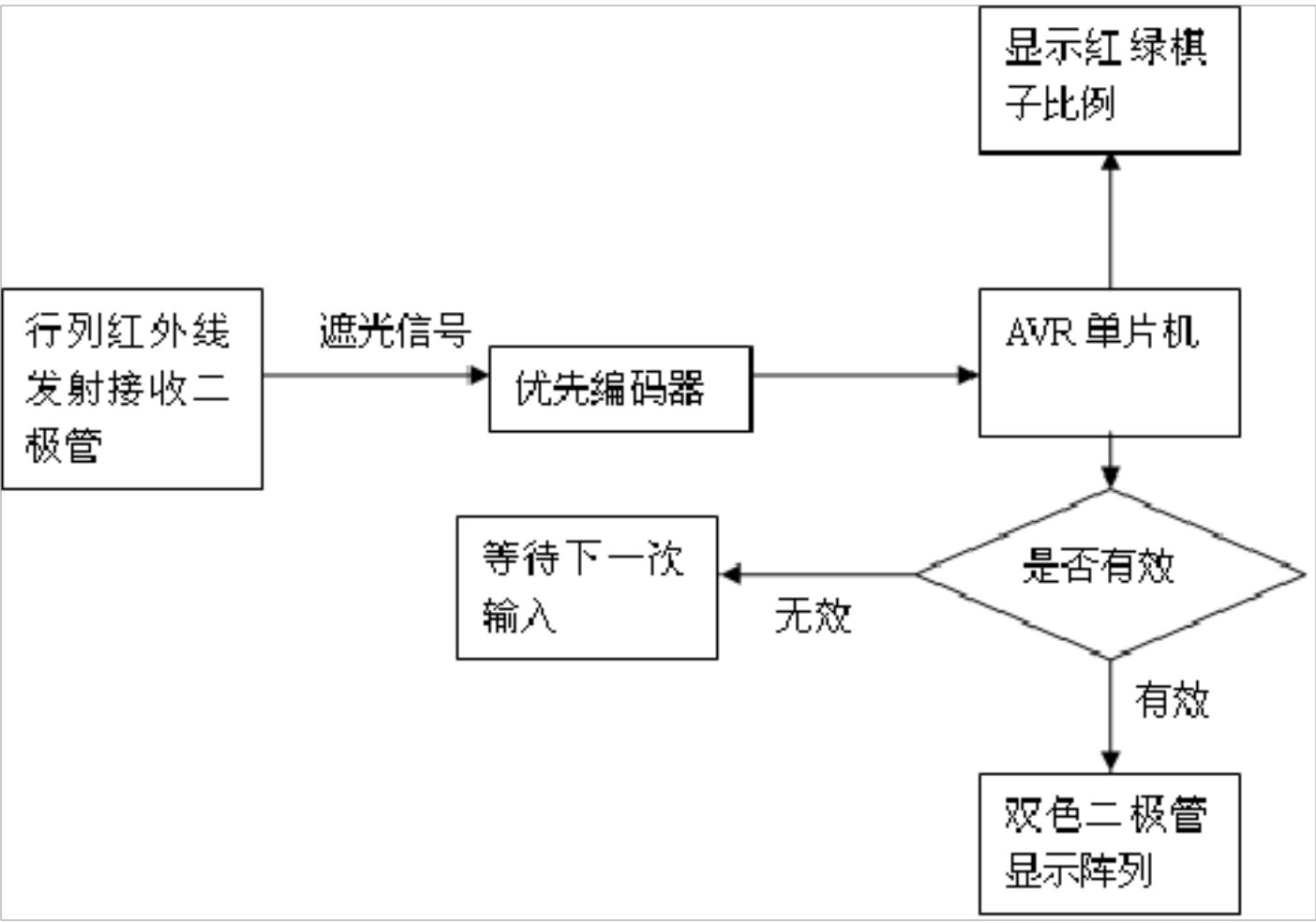
实现对黑白棋游戏的无子化操作，增加游戏的趣味性和方便性。学习实践单片机显示、按键等技术。

每一枚“棋子”就是一枚双色发光二极管，64 枚双色发光二极管排成 8×8 的阵列。每一枚二极管有三支引脚，引脚电平的高低决定了二极管显示的颜色，而有单片机控制双色发光二极管引脚的电平高低，实现棋盘上二极管显示不同颜色，以代表棋子。发光二极管亮度高，功耗低，寿命长，且选用双色发光二极管减少了焊接工作量，发光二极管的两种颜色红色和绿色的对比度也较大，使棋子醒目清楚。

1.1. 硬件设计思想和电路原理图

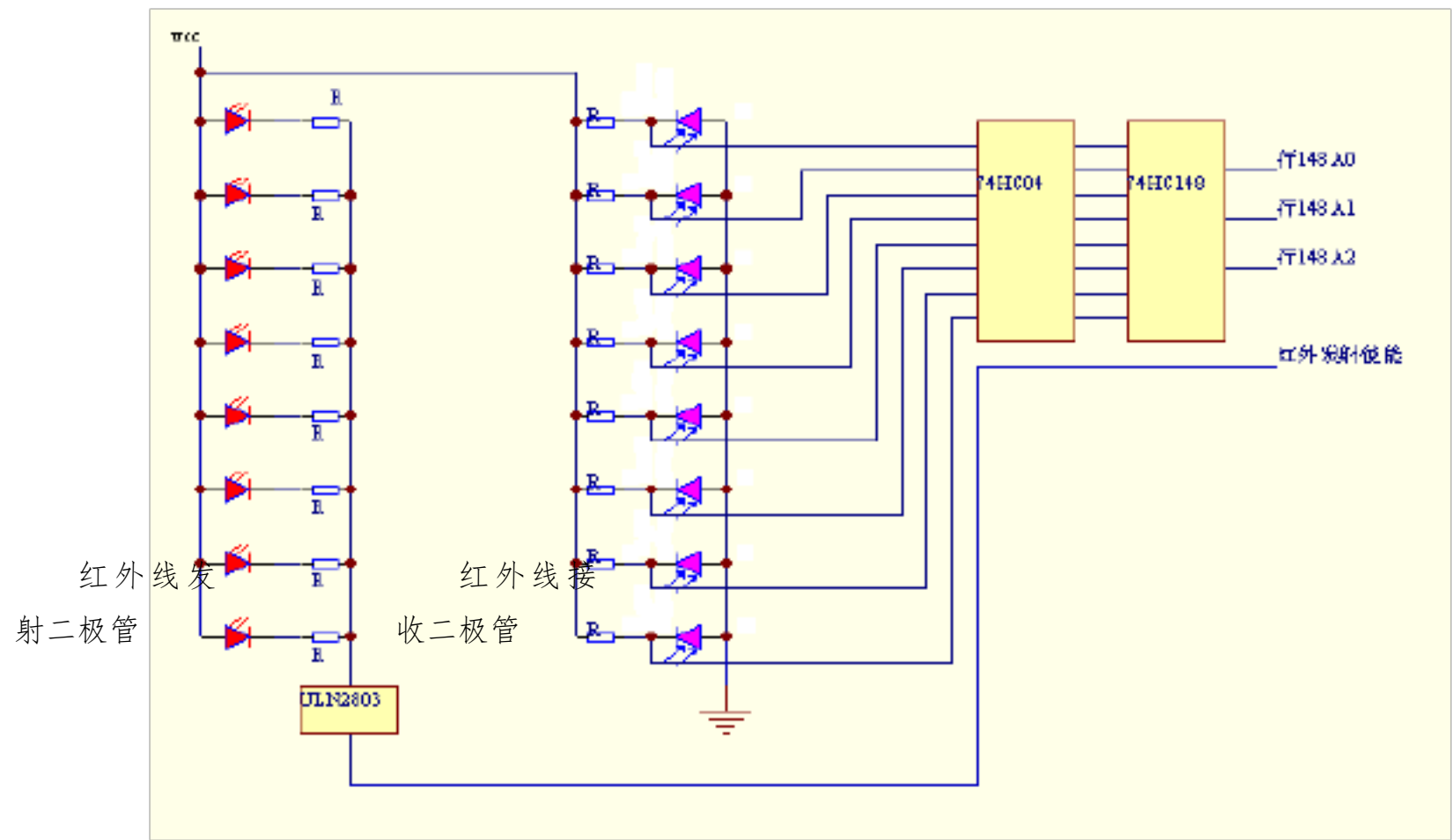
当阻断红外线发射二极管和接收二极管之间的光路时，接收管的电平会变化。用 16 对红外线发射接收二极管，8 对感受行信号，8 对感受列信号，行列发射接收二极管的光路的交点即一枚棋子的位置。当手指碰触到某个交点时，一组行列红外管的光路阻断，产生的信号通过优先编码器输入到单片机的 IO 口。

单片机在本作品中的功能有三点：首先，接收处理红外线接收二极管产生的电平信号；其次，存储棋盘上棋子的状态，计算需要变色的二极管的坐标，并输出到双色发光二极管阵列；再次，计算双方比分，有七段数码管输出，判断游戏的胜负。



图表 1 工作流程

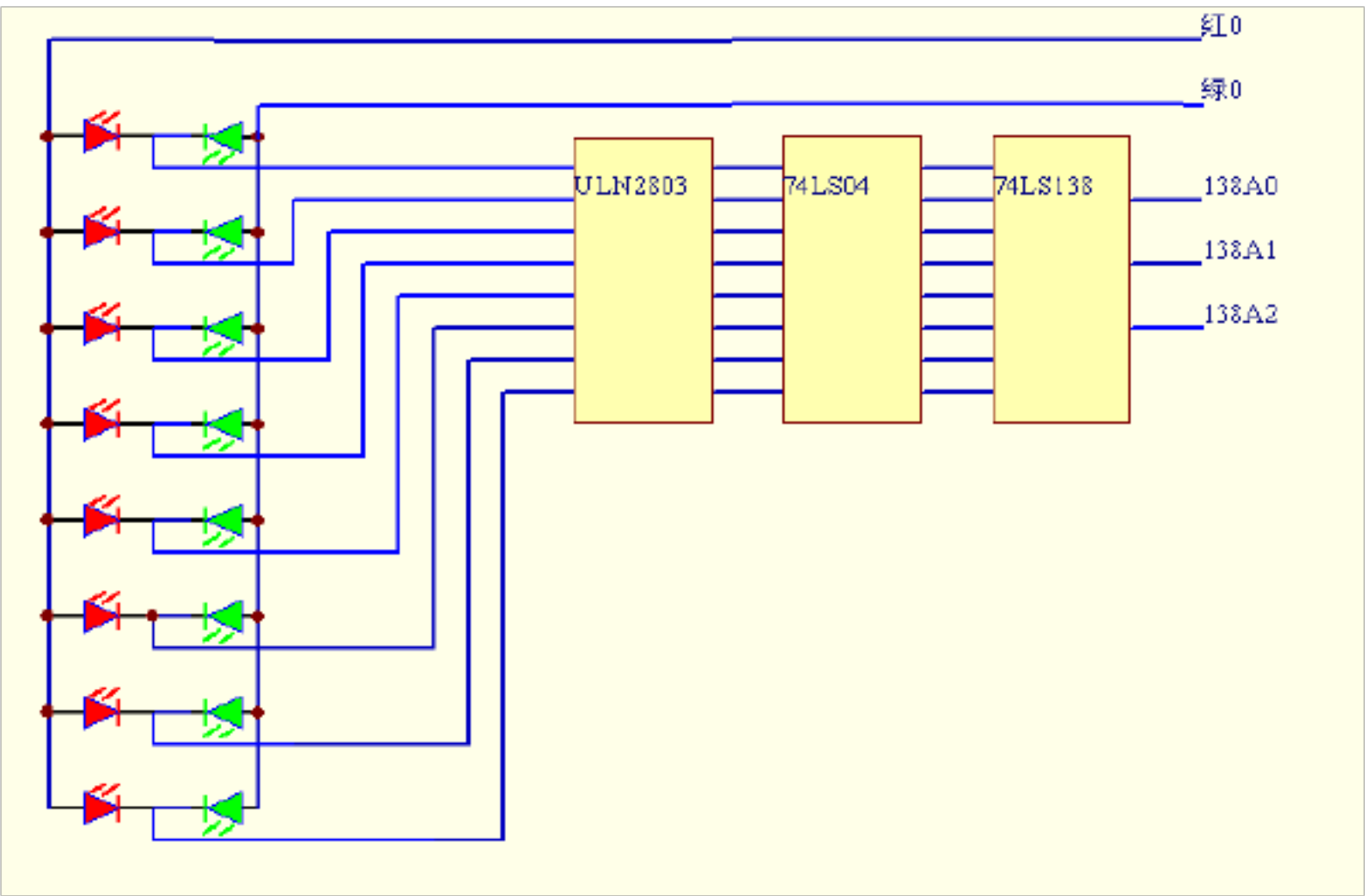
1.2. 红外线发射接收模块



图表 2 红外线发射接收模块

行列各有一组发射接收阵列，构成矩阵键盘。单片机 PB7 输出高电平时，非门 ULN2803 转变为低电平，红外线发射二极管开始工作，即使能红外线发射。红外线接收管电平变化的信号先通过非门 74HC04，然后通过 8-3 优先编码器 74HC148 输出到单片机。

1.3. 棋盘显示模块



图表 3 棋盘显示模块

8 排上图所示结构构成棋盘。单片机输出的控制双色二极管变色的信号通过一个 3-8 译码器 74LS138 译码再通过两个非门对双色发光二极管的颜色。利用 ULN2803 灌电流大的特

点，避免芯片的损坏。

1. 4. 单片机接口

十位使能	PB0	1	40	PA0	红0	红g
让棋1, 悔棋0	PB1	2	39	PA1	红1	红f
按键中断	PB2	3	38	PA2	红2	红a
红绿棋子状使能	PB3	4	37	PA3	红3	红b
138使能	PB4	5	36	PA4	红4	红c
138A0	PB5	6	35	PA5	红5	红e
138A1	PB6	7	34	PA6	红6	红d
138A2	PB7	8	33	PA7	红7	红子状态
个位使能	RESET	9	32	AREF		
红外发射使能	Vcc	10	31	GND		
	GND	11	30	AVcc		
	XTAL2	12	29	PC7	绿7	绿子状态
	XTAL1	13	28	PC6	绿6	绿d
行148Ys有效输入1	PD0	14	27	PC5	绿5	绿e
行148A0	PD1	15	26	PC4	绿4	绿c
行148A1	PD2	16	25	PC3	绿3	绿b
行148A2	PD3	17	24	PC2	绿2	绿a
列148A0	PD4	18	23	PC1	绿1	绿f
	PD5	19	22	PC0	绿0	绿g
列148A1	PD6	20	21	PD7	列148A2	

图表 4 单片机接口

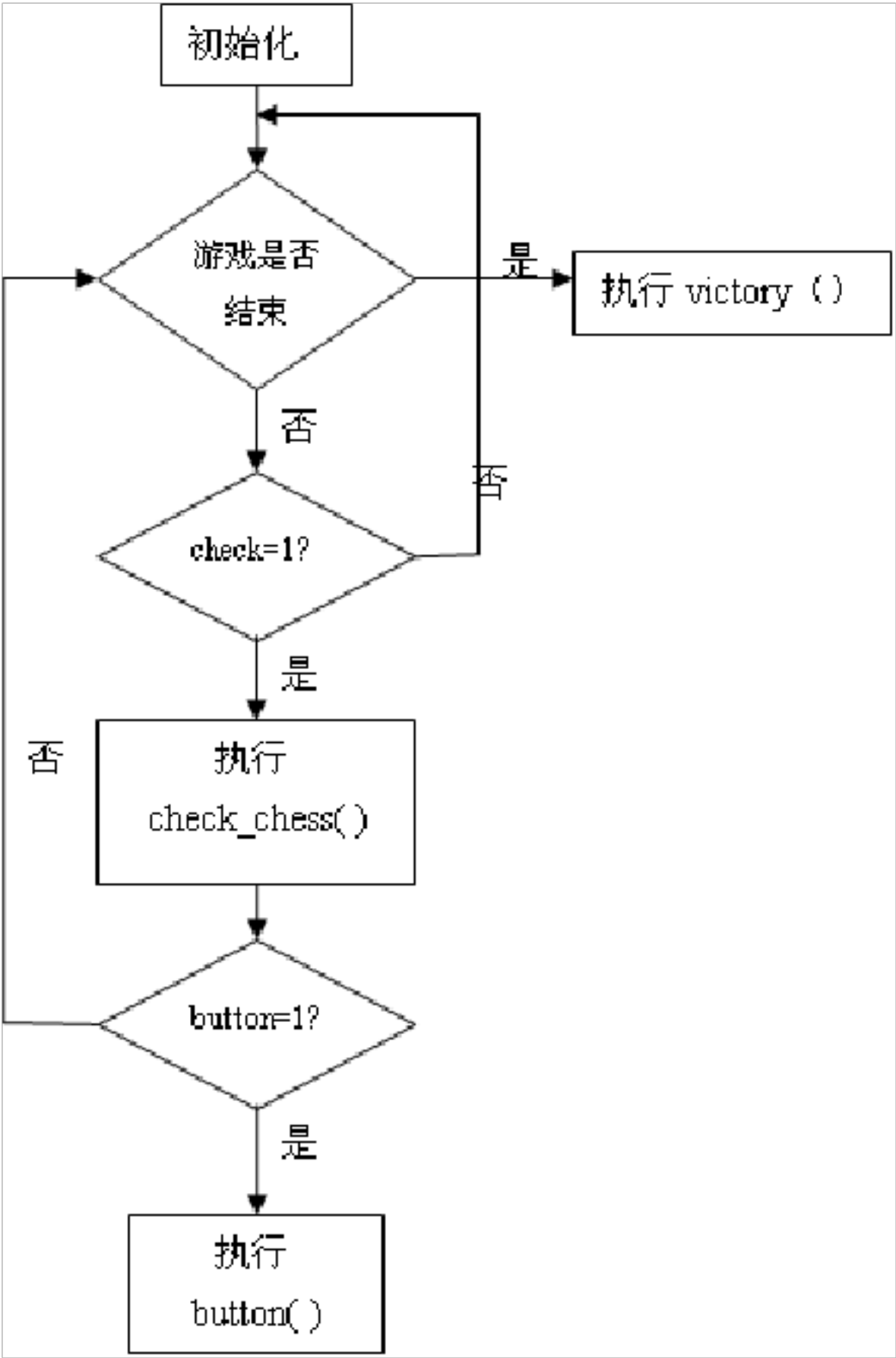
1. 5. 硬件单元的使用

每一枚“棋子”就是一枚双色发光二极管，64 枚双色发光二极管排成 8×8 的阵列。每一枚二极管有三支引脚，引脚电平的高低决定了二极管显示的颜色，而有单片机控制双色发光二极管引脚的电平高低，实现棋盘上二极管显示不同颜色，以代表棋子。

有两组各八对红外线发射接收二极管，每只二极管前面加一只准直小管以保证任意两对红外线发射接收管的信号不相互干扰。使用时用手指遮挡一组行列红外发射接收二极管光路的交点（即对应于棋盘上的一枚双色二极管），把由这些红外发射接收二极管组成的虚拟按键触发产生一个落子点的位置坐标信息，再由单片机识别判断。还设计了“让棋”“悔棋”按键使游戏更加完善。两个七段数码管负责显示双方成绩，双方各有一枚代表自己颜色的发光二极管，自己所对应的二极管亮时代表落子权在自己手中。当某一方胜利时，棋盘上会闪动胜方棋子所对应的颜色。

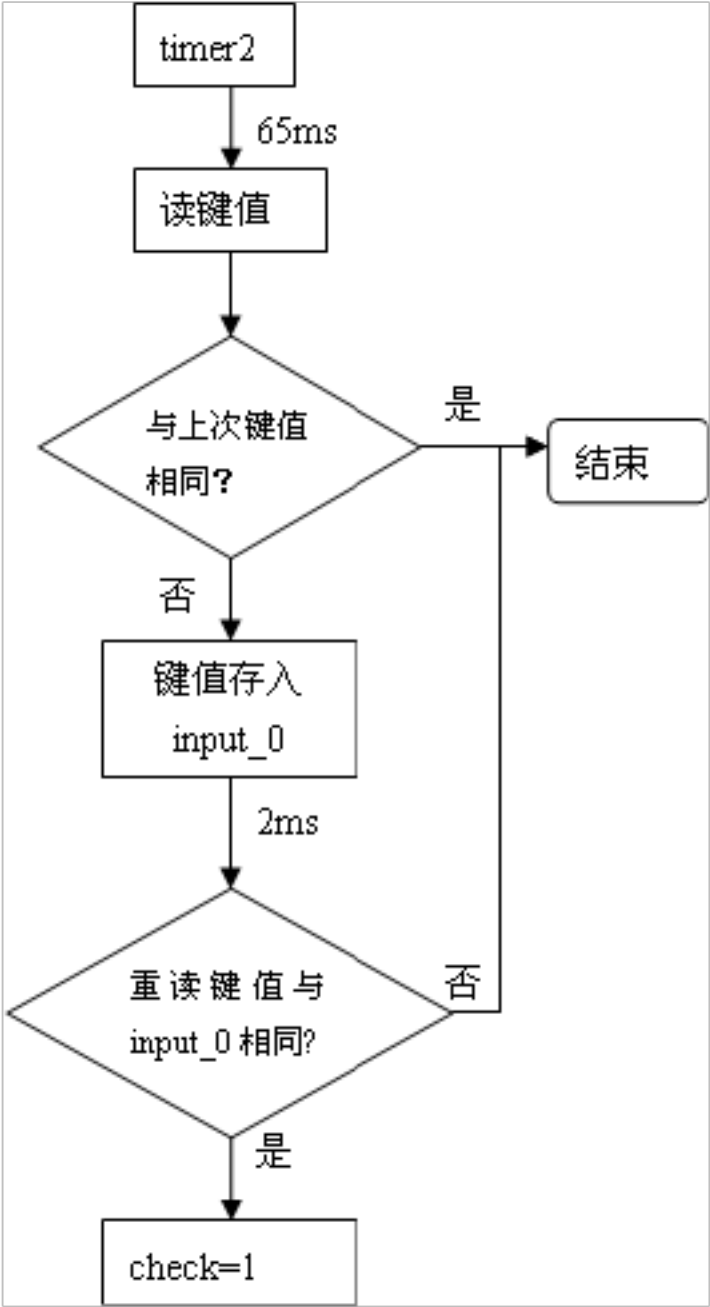
2. 软件设计思想及软件流程

2.1. 主函数

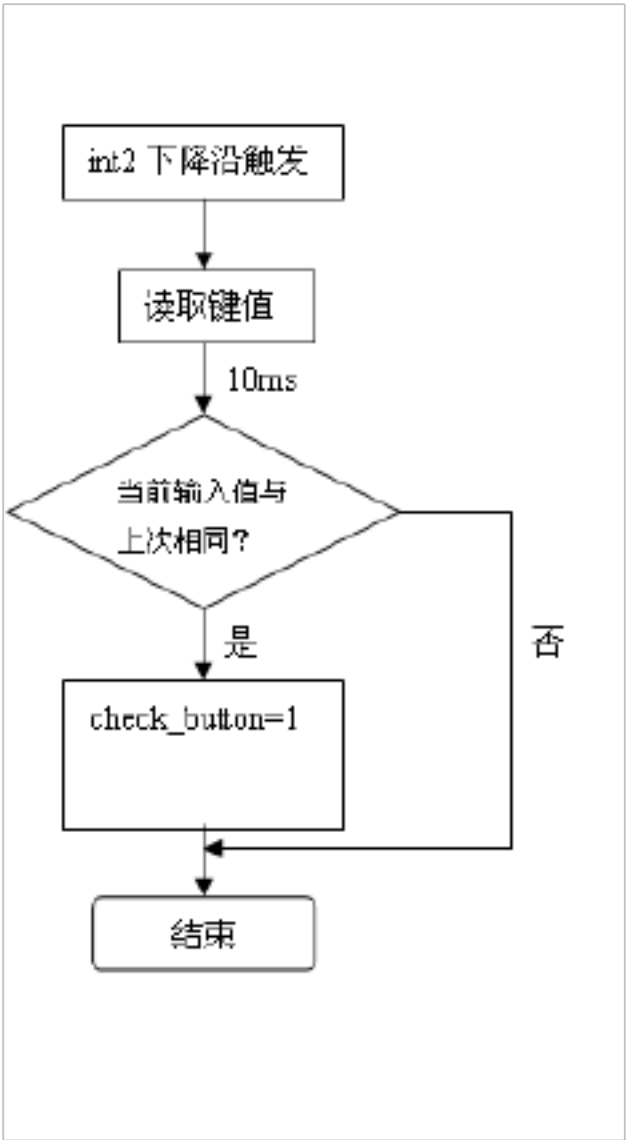


图表 1 主函数

2.2. 按键输入模块

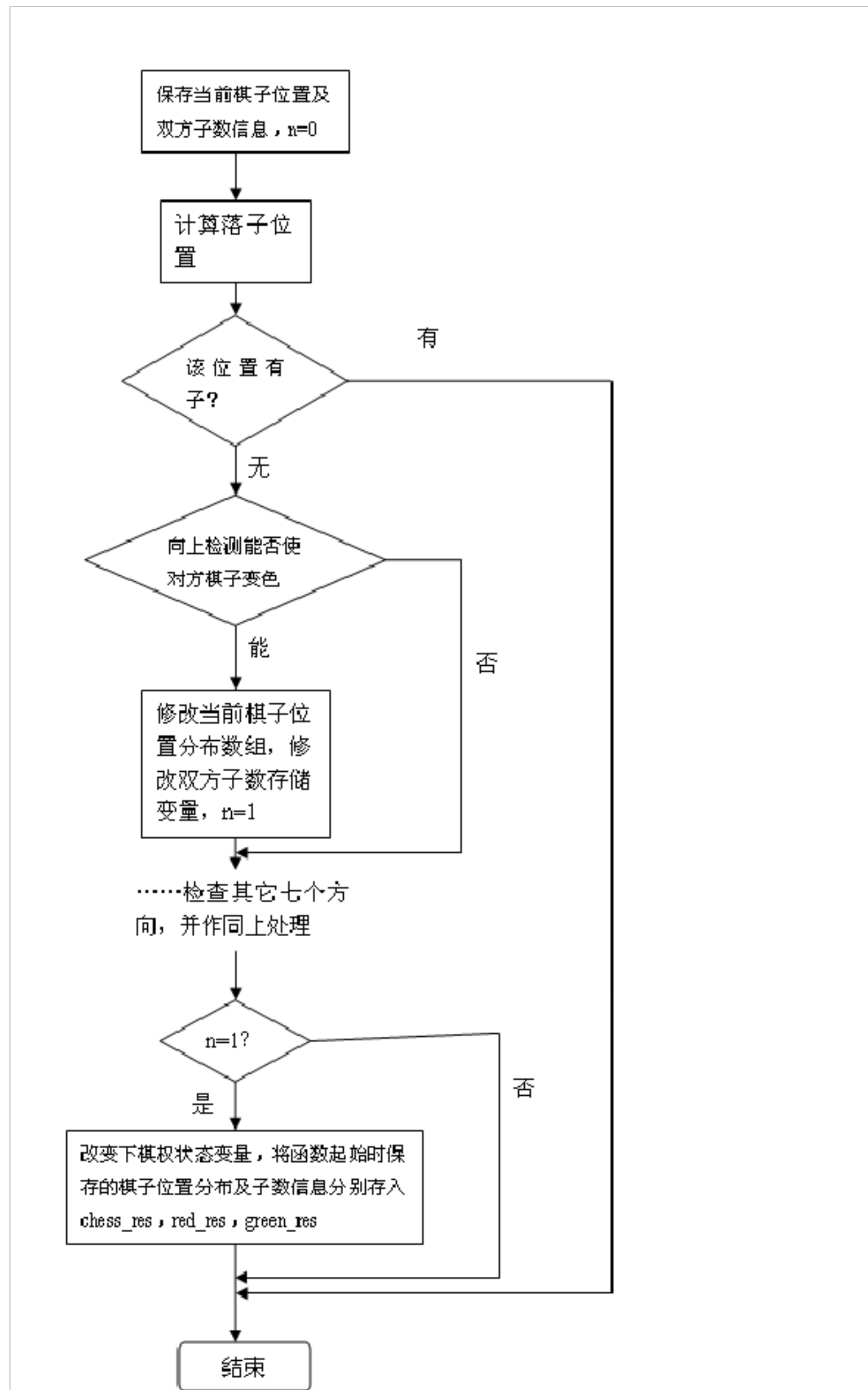


图表 2 棋子按键

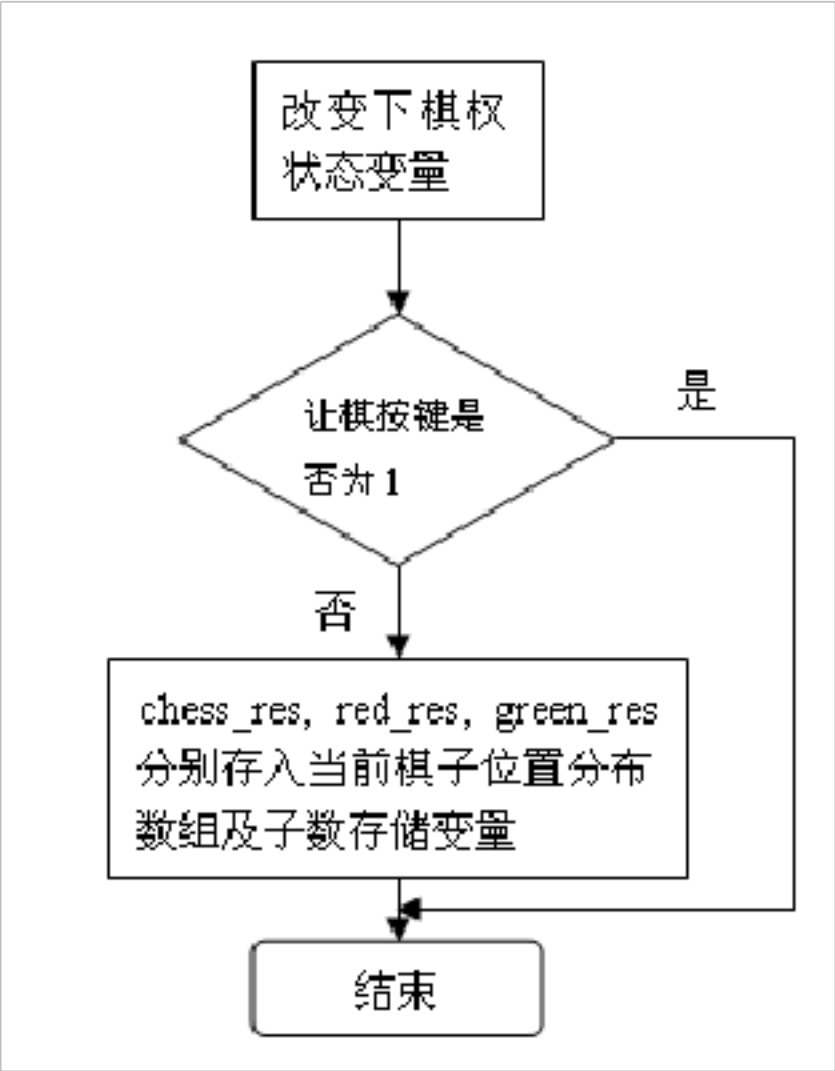


图表 3 悔棋让棋按键

2.3. 游戏算法模块



图表 4 `check_chess` 函数



图表 5 button 函数

2. 4. 显示模块

定时器 timer0 每隔 1ms 触发一次中断，扫描相应 LED和七段数码管。

3. 详细说明软件功能

3. 1. 显示模块

利用定时器 0 比较中断，每隔 1ms 改变各个端口的输出，从而实现棋盘和记分板的扫描。

3. 2. 按键模块

分为红外按键（落子按键）和悔棋让棋按键两部分：

红外按键：利用定时器 2 的溢出中断，每隔 65ms 扫描一次红外按键阵列。若有输入，则使能其比较中断，2ms 后在比较中断中比较两次输入的键值，实现消抖处理。

悔棋让棋按键：悔棋让棋按键接在了 PB2端口，一旦有键按下，便会触发 INT2下降沿中断。在 INT2的中断函数中，将 check_button 置 1。主函数中，检查 chess_button 的值，若为 1，则执行 button 函数。button 函数中，实现按键消抖，和判断具体是悔棋还是让棋的功能并加以执行。若 PB1值为 1，则执行让棋操作，值为 0 执行悔棋操作。

3. 3. 黑白棋规则模块（check_chess 函数）

通过检查当前落子位置周围的八个方位上能否使对方棋子变色。能变色，则允许落子，实现棋子存储数组 chess 和比分记录变量 red, green 的相应改变。

3. 4. 胜利模式（victory 函数）

获胜方棋子闪动

然后判断棋局是否结束。没结束则一直检测是否落子和有悔棋让棋按钮按下。若有落子，执行 **check_chess** 函数。若有悔棋让棋按钮按下，则执行 **button** 函数。棋局结束后，执行 **victory** 函数。

3. 6. 源程序

```
#include <iom16v.h>
#include <macros.h>
flash unsigned char disbuffer[10]={0x01,0x67,0x12,0x22,0x64,0x28,0x08,0x63,0x00,0x20};
unsigned char chess[2][8]={ {0,0,0,0x10,0x08,0,0,0},{0,0,0,0x08,0x10,0,0,0}};
//red=0,green=1
unsigned char red=2,green=2,state=1,check=0,vic=0,count=0,count_timer0=9,check_button=0;
unsigned char chess_res[2][8]={ {0,0,0,0x10,0x08,0,0,0},{0,0,0,0x08,0x10,0,0,0}};
unsigned char chess_p[2][8]={ {0,0,0,0x10,0x08,0,0,0},{0,0,0,0x08,0x10,0,0,0}};
unsigned char red_res=2,green_res=2;
unsigned char input=0x00,input_0,input_jtd=0x20;
/*****
延时函数，实现延时 1ms
*****/

void delay_ms(char n)
{
    char i;
    while(n--)
    {
        i=400;
        while(--i)
        {
            NOP();
            NOP();
            NOP();
            NOP();
            NOP();
        }
    }
}

/*****
各个 I/O 端口的初始化
*****/

void port_init(void)
{
    PORTA = 0x00;
```

```

DDRA = 0xFF;
PORTB = 0x06;
DDRB = 0xF9;
PORTC = 0x00; //m103 output only
DDRC = 0xFF;
PORTD = 0x20;
DDRD = 0x00;
}

```

```

/*****

```

0 的初始化，利用其比较中断，每隔 1ms 改变各个端口的输出，从而实现棋盘和记分板的扫描。

```

*****/

```

```

void timer0_init(void)
{
    TCCR0 = 0x00; //stop
    TCNT0 = 0x00; //set count
    OCR0 = 0x3F; //set compare
    TCCR0 = 0x0B; //start timer
}

```

```

#pragma interrupt_handler timer0_comp_isr:20

```

```

void timer0_comp_isr(void)
{
    //compare occurred TCNT0=OCR0

    if(count_timer0==9)
    {
        PORTB=0X06;
        PORTC=disbuffer[green%10]|(state<<7);
        PORTA=disbuffer[red%10]|(~state<<7);
        PORTB=0x86;
    }
    else if(count_timer0==8)
    {
        PORTB=0X06;
        PORTC=disbuffer[green/10]|(state<<7);
        PORTA=disbuffer[red/10]|(~state<<7);
        PORTB=0x07;
    }
    else
    {
        PORTB=0X06;
        PORTA=0x00;
    }
}

```

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：<https://d.book118.com/637056031142006155>