

第九课 宠物追球



一. 本节要点

- ◆ 学习 IIC 通讯在两块控制板上的运用
- ◆ 利用自己的小车实现追球的功能
- ◆ 器件准备：miniQ 小车，Micro USB 线，小车顶板，红外发射及接收管，洞洞板。

二. 制作步骤

STEP 1：制作小球（红外发射端）

学习了这么多的关于小车上器件的电路和编程知识，不自己做点好玩的应用实在是说不过去，鉴于巡线和壁障之类的在以前我们都学习过了，现在大家一起来试着做一辆可以追着你的球运动的小车吧。

既然是 DIY 智能玩具，那就要充分发挥我们的动手能力吧。作为自己的小宠物，当然

不能是个小球就去追，这样就没了宠物的归属感。所以，我们先抛弃利用壁障传感器来作为物体探测的想法吧。既然要做，就不怕困难一起连球也做好了吧。这里的球是由一圈圈红外发射管组成的，如下图：



图 1 全向红外球图片

当然 ,由于器材限制 ,我们就自己发挥下聪明才智去动手做一个属于自己风格的小球吧。首先我们要明确做这个球的目的 ,这样才能在制作中了解哪些是重点哪些是可以一笔带过的东西。

做成球形的目的是可以滚动，但是一般情况下，由于小车传感器对感应距离的限制，如果小球滚得太远则会超过检测范围，所以最好在球上加个绳子，这样就不用跟着小球到处跑了，弄到左后什么床底下，洗手间，桌子下到处捡球。其实这些都是我做的这个球的借口，下图为一个比较不好看的实物：

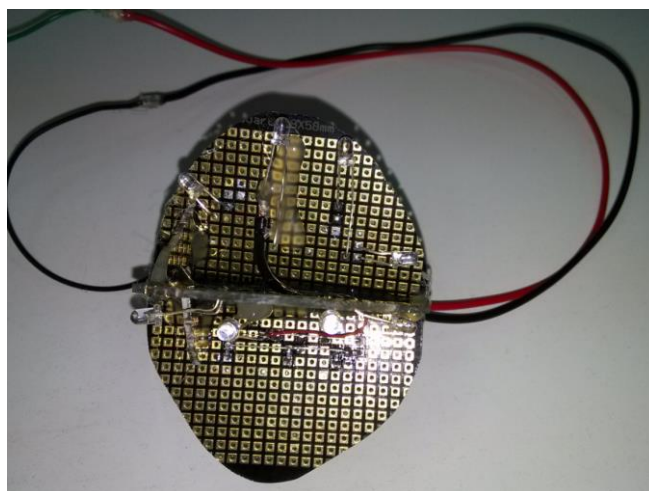


图 2 自己做的简便的红外球

小球做的是不太好看，不过大家可以自己发挥聪明才智设计自己喜欢的造型，原理就是让他能朝各个方向发出红外光给小车检测就可以了。即使不是各个方向，自己玩的时候先做个一个或多个方向的也可以。

电路方面，其实就是简单的红外发射电路，只是并联了多个而已：

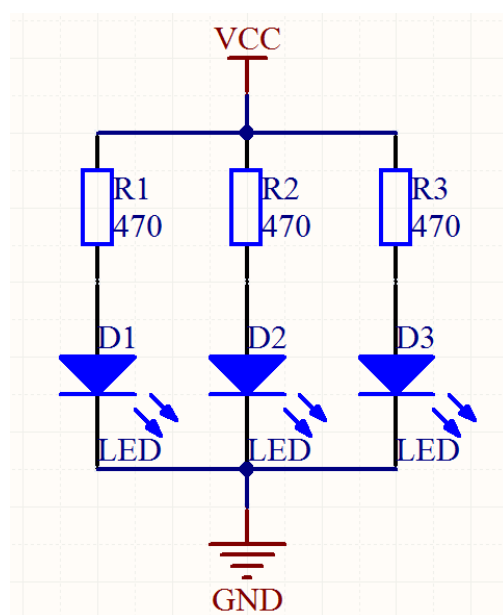


图 3 红外发射管电路图（3 组）

上图，R1，R2，R3 都是阻值为 470 欧的电阻，D1，D2，D3 都是红外发射管，VCC 是电源正极，GND 是电源负极。如果这张图看起来有问题，可以看看下面这张：

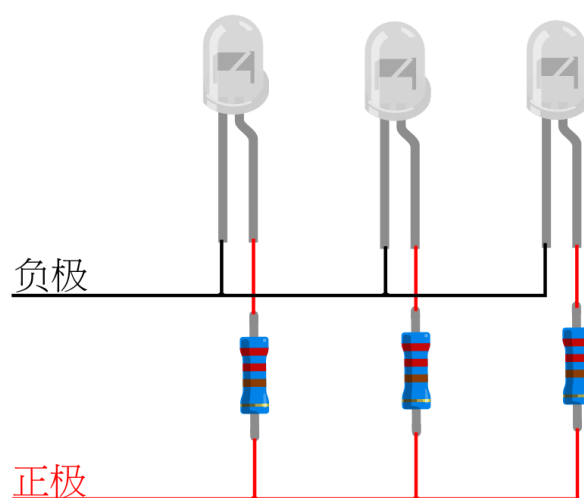


图 4 红外发射示意图（3 组）

电源一般使用 4-6V 都可以 ,电阻看实际想要达到的检测范围 ,建议使用 470 欧左右的 ,这样 ,再把红外发射管负极接在电源负极上就可以了。

记住电路 ,只要焊接起来就可以了 ,找一块[洞洞板](#) ,把元器件焊接在板子上 ,事先注意焊接位置和洞洞板的形状 ,方便后面设计成自己喜欢的形状。

STEP 2 : 设计小车上层板 (红外接收端)

下面就是设计小车上层板上的电路了。小车上层板使用的是和 Leonardo 一样的控制芯片 ,接口也都是相同的 ,而且还外加了很多的焊接孔 ,很方便我们的改造。先上一下板子焊接完成后的图吧:

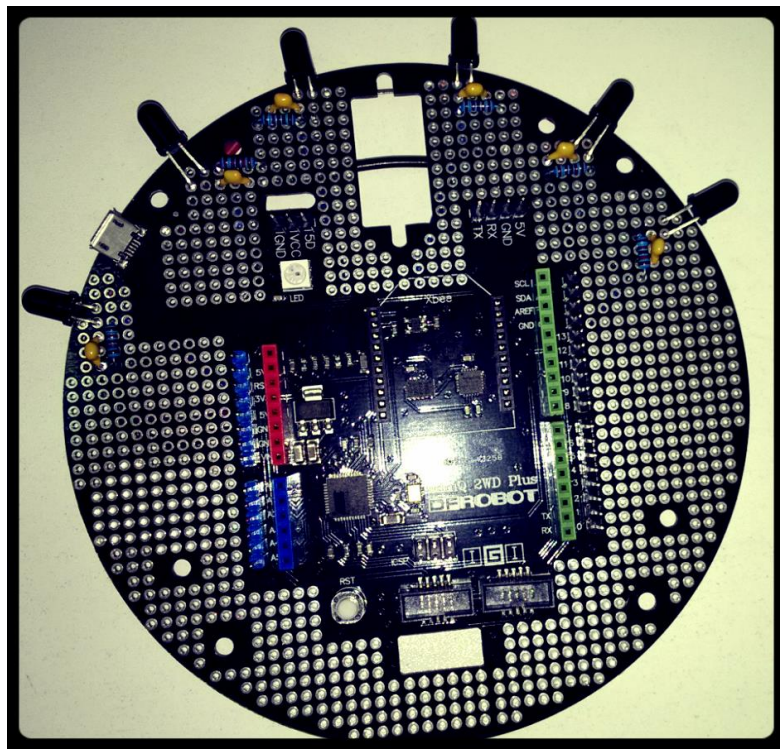


图 5 增加了红外检测器件的小车上层板正面图

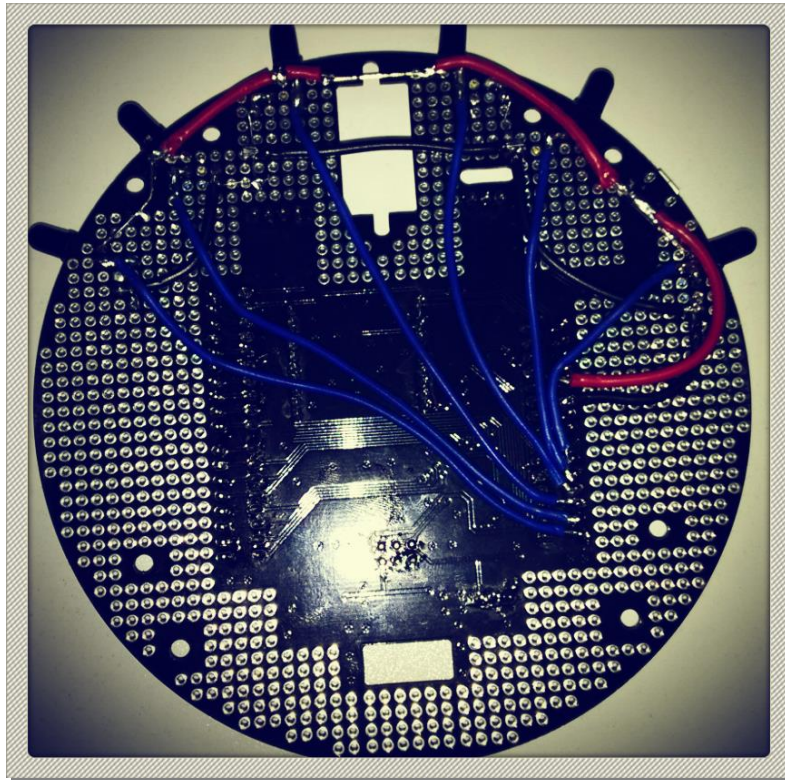


图 6 增加了红外检测器件的小车上层板背面图

如果上面提到的焊接红外发射管是焊接的热身，这个更像是焊接技术的测验，当然，我们不需要把它当成是测试，但是要注意焊接时细心，因为这里的引脚距离比较近，初次焊接时还是需要仔细，注意焊接完要检查是否漏焊，短路等。

这个是焊接的电路图：

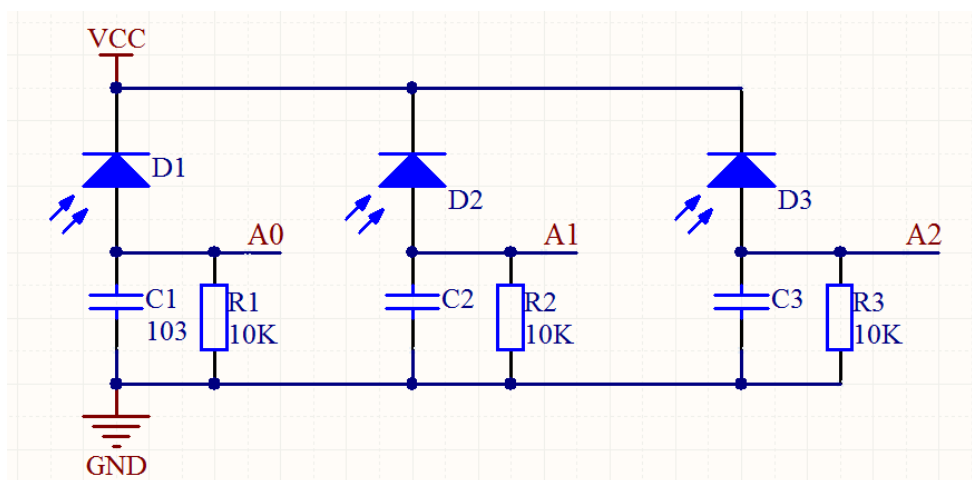


图 7 上层红外接收电路图

D1, D2, D3 分别为红外接收二极管, A0, A1, A2 为上层板的模拟口 A0, A1, A2.

实际在 A0—A6 都接有相同的电路，剩下的大家可以按照上面的一样接另外 3 个至剩下的模拟口。

这个和红外发射的接线方法差不多，电源正极和负极都接在一起就可以了，信号脚各自接在不同的模拟脚。

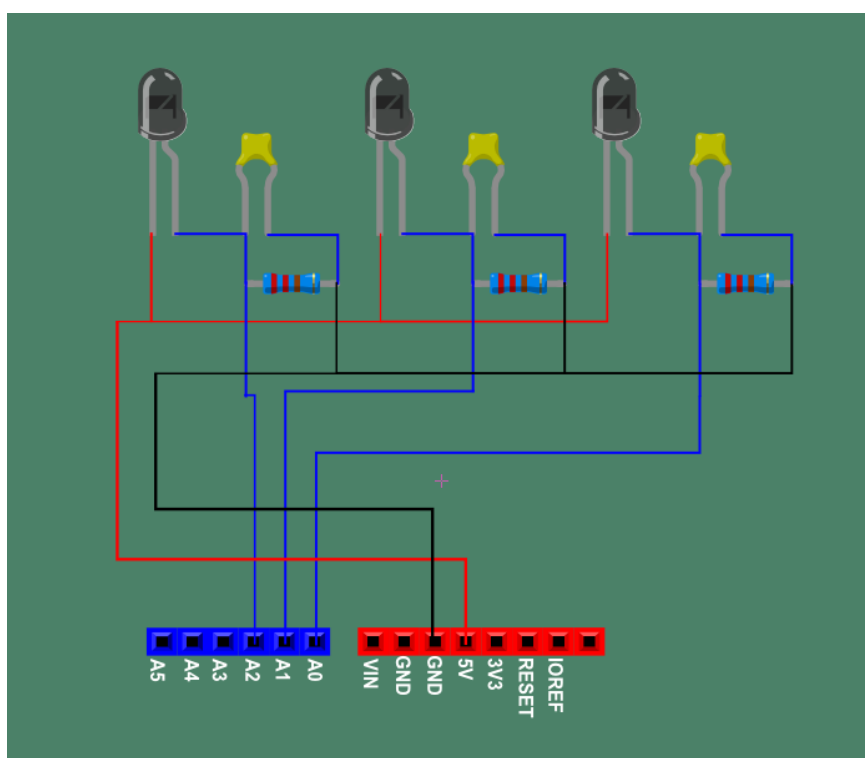


图 8 接收电路示意图

好了，按照电路图焊接好电路后线写个程序验证下做的上层板电路是不是成功的吧！

STEP 3：给上层板下载测试代码

程序如下：

```
void setup() {  
    Serial.begin(9600);    //串口设置波特率为 9600  
}  
void loop() {  
    for(int i=0;i<6;i++)    {  
        int x=analogRead(i);    //读取数字 i 脚电压值，i 值为 0~5  
        Serial.print(x);        //打印获得的电压值  
        Serial.print(" ");      //空格，以区分各个值  
    }  
    Serial.println();          //打印换行  
    delay(500);                //延时 500ms  
}
```

STEP 4：串口查看数据

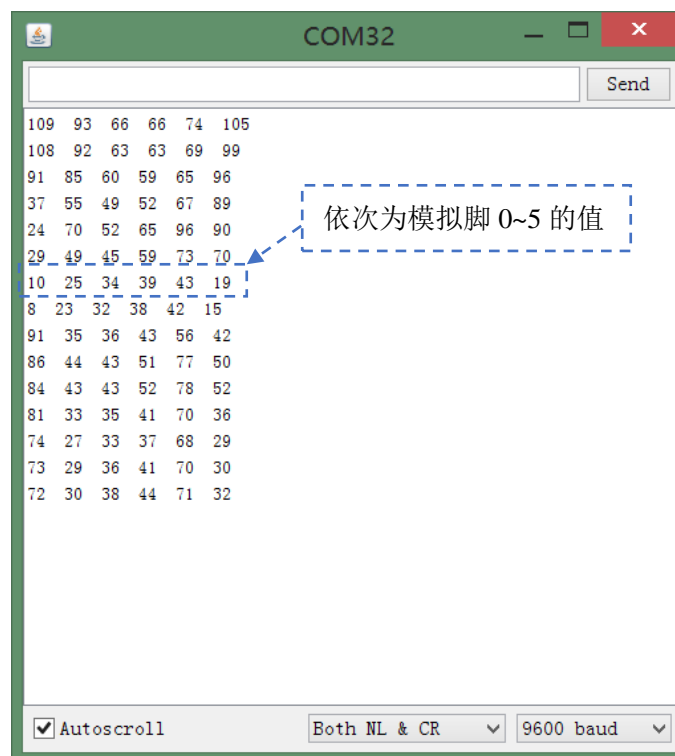


图 9 没有红外球时获得的数据

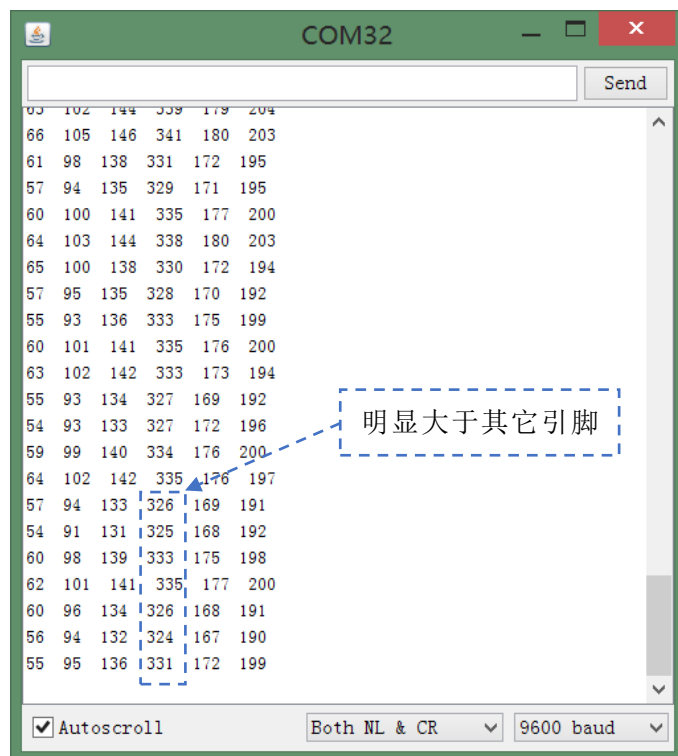


图 10 有小球在前方是的读数

STEP 5：给上下板下载代码

本次需要下载两次程序，分别为小车上层控制板的程序和 miniQ 车身的程序。miniQ 车身程序为 Bottom.ino，上层板程序为 Top.ino，下载完毕，就可以拿着你的小球玩耍啦。

三. 程序解析

上层板部分 (Top.ino)

(1) 命令发送子函数

```
void Send_Command(char command,int num) {  
    //发送数据给地址为 2 的器件  
    Wire.beginTransmission(Address_Bottom);  
    Wire.write(command);        //发送数据  
    Wire.write(num);            //发送数据  
    Wire.endTransmission();     // 停止发送  
    Serial.println(num); //串口打印，调试时用到的  
}
```

(2) 找红外值最大的引脚，即小球方向

```
for(int i=0;i<6;i++) {          //找出最大值及其号码  
    if(Distance[i]>Dist_Max){  
        Dist_Max=Distance[i];  
        Num_Max=i;  
    }  
    Serial.print(Distance[i]);  
    Serial.print(" ");  
}
```

(3) 发送命令，并限制最小值不小于 100

```
if(Dist_Max>100) {              //根据最大值所在号码发出不同命令  
    //如果号码为 0 或 1，则发送 L100  
    if(Num_Max<2) {  
        Send_Command(cmd[2],100);  
    }else if(Num_Max==4 || Num_Max==5) {  
        Send_Command(cmd[3],100);  
    } //如果号码为 4 或 5 则发送 R100  
    else if(Num_Max==2 || Num_Max==3) {  
        if(Dist_Max>800) {      //如果距离很近  
            Send_Command(cmd[4],100);  
            Serial.println(cmd[4]);  
            delay(200); //发送 S100，即 Stop  
        }else{  
            Send_Command(cmd[0],100);  
            Serial.println(cmd[0]); //如果距离没有到达足够近，则发送 F100  
        }  
    }  
}else{  
    Send_Command(cmd[4],80);
```

```

        //如果接受到的值都小于 100，则表示没有求，小车发送 s80
        Serial.println(cmd[2]);
    }

```

miniQ 小车程序部分 (下层 Bottom.ino)

(1) 主程序部分

```

void loop(){
    Motor();//电机状态控制
    Color();//RGB 灯颜色控制
}

```

(2) 颜色控制子函数

```

void Color(){          //对比接收到的命令值，发出不同颜色
    switch (Cmd_Str) {
        case 'F':
            strip.setPixelColor(0, 0xbb0000);
            strip.show();
            break;
        case 'B':
            strip.setPixelColor(0, 0x0000bb);
            strip.show();
            break;
        case 'L':
            strip.setPixelColor(0, 0x00bb00);
            strip.show();
            break;
        case 'R':
            strip.setPixelColor(0, 0x550055);
            strip.show();
            break;
        case 'S':
            strip.setPixelColor(0, 0x005555);
            strip.show();
            break;
        default:
            break;
    }
}

```

(3) 电机控制子函数

```

void Motor(){          //根据接收的命令和参数，电机执行不同动作
    switch (Cmd_Str) {
        case 'F': Forward(Cmd_Num,Cmd_Num);      break;
        case 'B': Back(Cmd_Num,Cmd_Num);         break;
        case 'L': Turn_Left(Cmd_Num,Cmd_Num);    break;
        case 'R': Turn_Right(Cmd_Num,Cmd_Num);   break;
        case 'S': Stop();                          break;
        default: break;
    }
}

```

(4) 命令接收

```
void receiveEvent(int howMany){    //中断接收程序
    while(Wire.available()>1){ // loop through all but the last
        Cmd_Str = Wire.read(); // receive byte as a character
        Serial.print(Cmd_Str);    // print the character
    }
    Serial.print(" ");
    Cmd_Num = Wire.read();    // receive byte as an integer
    Serial.println(Cmd_Num);    // print the integer
}
```

四. 程序解析

小车程序分为两个部分，总体说来就是两个主控器之间的相互通讯，一个负责采集红外球的位置，一个负责控制电机运动。这里还是要消除一个疑问：为什么不直接用一个板子完成？这个因为 miniQ 上器件很多，基本占用了所有的控制引脚，所以很难完成这种多传感器的检测任务。还有一点就是学习下 IIC 在双机通讯中的应用，小车上 IIC 的器件只局限于电子罗盘，只有自己学习了双机通讯相关的知识，才可以真正了解 IIC 等总线的意义。

先说说小车本身在这个整体中所担任的角色

1. 既然有了电机，就不可避免需要控制电机转动实现各种运动；
2. 本身器件也可以被用来指示小车当前状态，用于在调试时检查是否程序执行的问题，如利用小车上的全彩 LED 灯指示红外球的方向。
3. 为上层板供电（通过接线）。关于上层控制板：检测红外球位置，处理后发送命令给 miniQ 控制 miniQ 的运动。由上可以看出，miniQ 的运动状态和 LED 的颜色显示完全由上层板发送的命令控制，自己则作为一个很简单的执行机器人，而上层，则只是负责采集数据，判断小车该去哪里，把执行交给下层。两个的分工明确，这样检查错误也和后面的和其它模块的拼接也就很方便了。

先说下上层小车的算法结构吧！

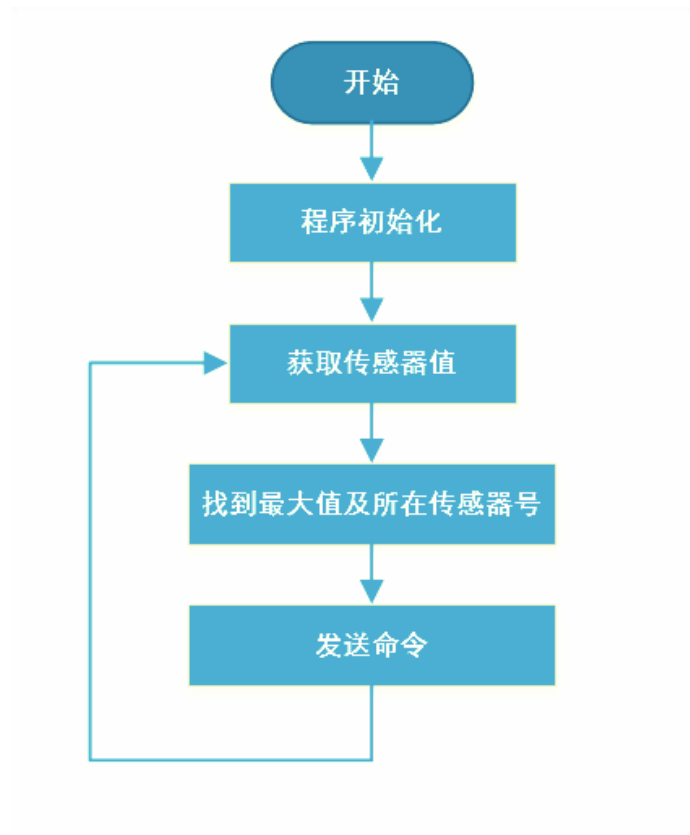


图 11 上层控制板程序流程图

由上图,小车只是循环的采集数据 → 分析数据 → 得到红外球位置 → 发送命令的工作,当然由于这些已经足够完成小车目前的需求,以后大家可以自己加上其它的命令,控制小车进行运动。

比如数据采集更改为: 发送测量命令 → 底部小车发眼 → 上层采集数据 → 综合分析数据 → 发出命令。这些都是很容易就实现的。

下面再看看底层的 miniQ 小车的运动流程图吧！

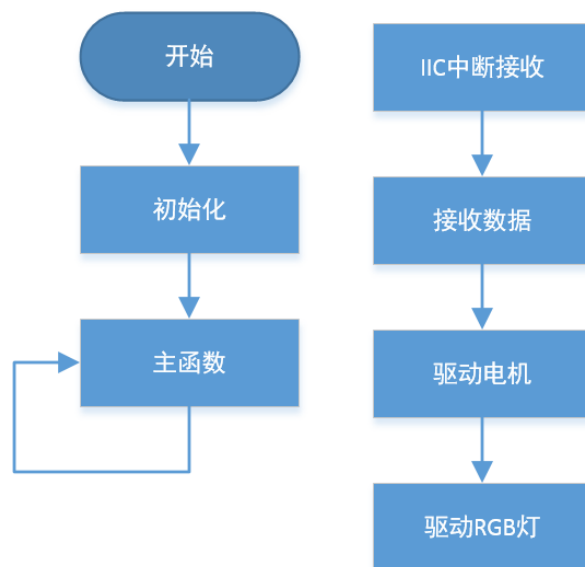


图 12 下层控制板程序流程图

由于 IIC 通讯为中断式，所以小车控制电机的参数在会由上层板的命令来更新。小车只要按照这些参数来执行电机或 LED 灯的色彩选择就可以了。

还要简单提到的一点就是通讯协议，通讯协议又称通信规程，是指通信双方对数据传送控制的一种约定。约定中包括很多项，如 IIC 中的起始位，通讯速度，停止位等，但是这里所说的是在 IIC 基础上的通讯协议，就是上层板发送的数据下层板如何区分的问题。样例中给出的是一个很简单的例子，只是用 F，B，L，R 等字母配上数字完成了对下层板的控制，如上层板发送 F100，下层板接收到后线判断字母是什么命令，再提取 100 来判断改用什么速度完成这个指令（程序中是指前进，PWM 值为 100）。这当然是自己定下的一个小的协议，上下两层都按照这个标准执行，即使是以后更换其它的控制器的，只要也是按照这个规定来执行的，就可以直接用在这个上面。比如现在上层板跟换为其它程序，但只要发送的格式是一样的，就任然可以控制下层小车。这也进一步体现了多机通讯在程序上的便利。

好了，基本上需要解释的都在上面了，欢迎补充和发聃建议：

Holiday.Jin@DFRobot.com