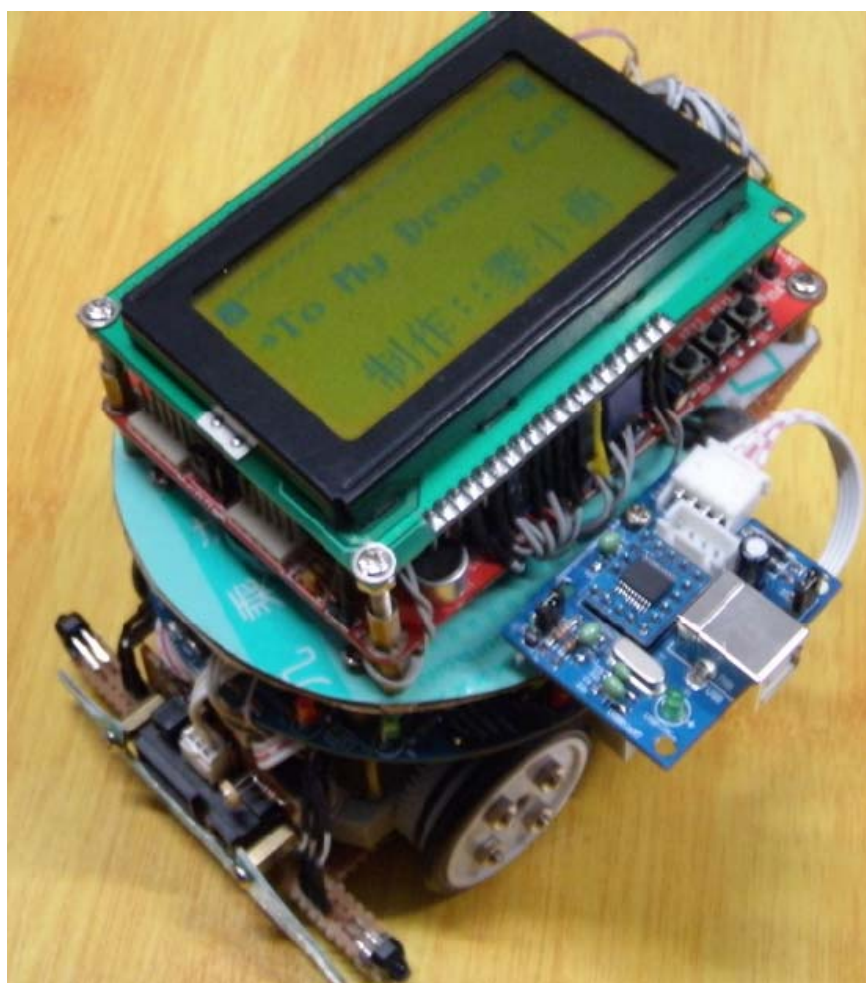


圆梦小车也“双核”

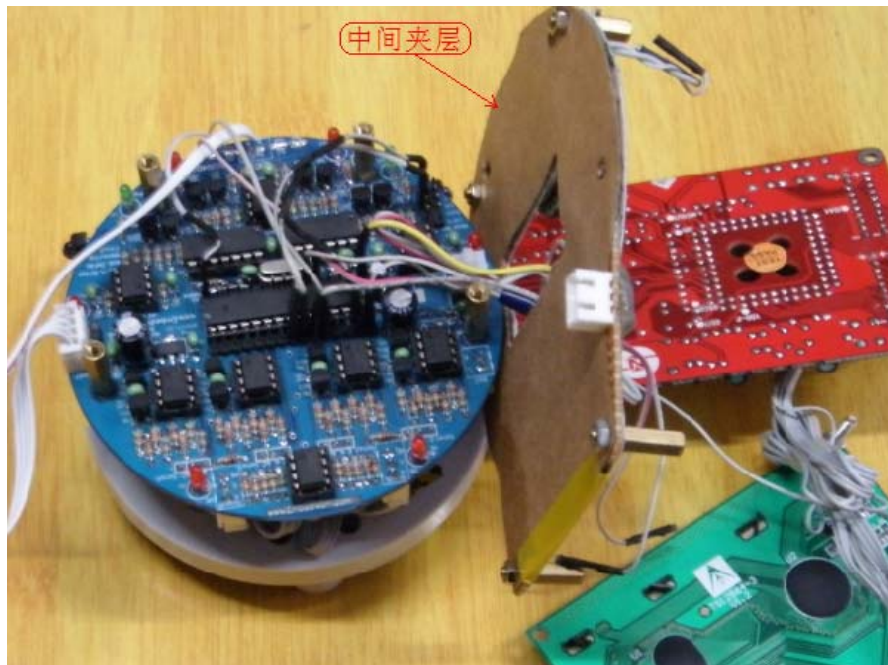
——双单片机控制圆梦车

在本篇中我们要做的“双核”小车，分别使用的 MCU 分别为 51 的 STC12LE5410AD 单片机和凌阳的 SPCE061A 单片机。那么在我们动手之前首先要解决一些装配上的细节问题，然后比较一下这两“核”的区别，再分配给他们各自相应的任务！首先让我们还是先来看看成型后小车酷姿吧！如图：

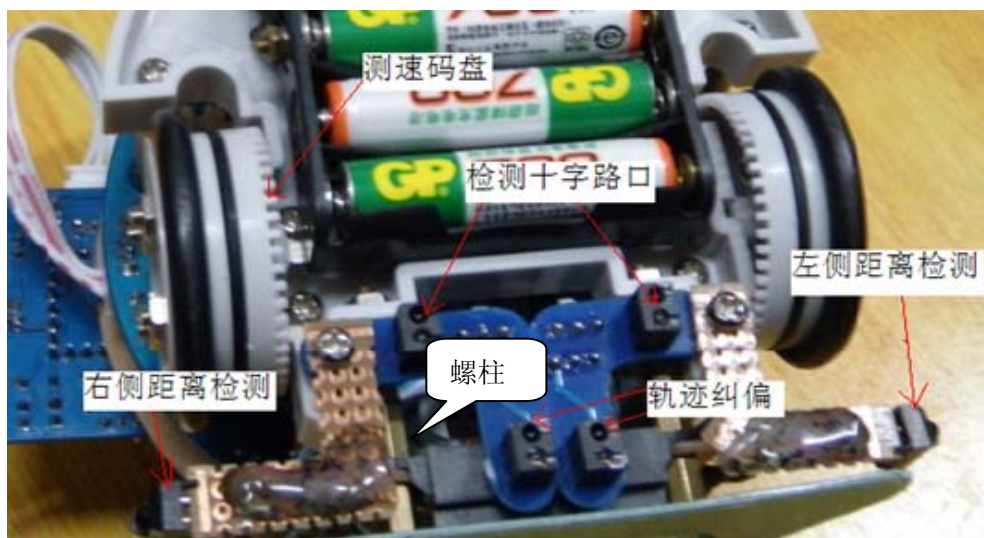


一、装配细节

首先是圆梦车和 61 板的螺孔不是太配套, 我们可以在两板中间加一个中间空心的夹层, 如下图。这样圆梦车控制板上的电源和通信等引线就可以从中间的通孔引到凌阳板上。



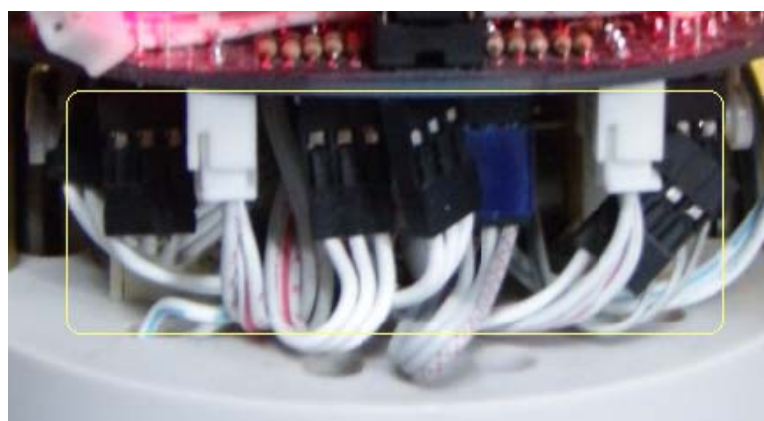
其次就是七个红外发射接收对管的安装 (两只用于纠偏, 两只检测十字路口, 两只检测左右距离, 一只用于弥补 SHARP 的盲区), 其中六只红外对管安装位置如下图:



第七只红外对管安装位置如下图所示, 其中需要留意的就是上图中安装的螺柱。由于用来安装红外管的直角板的引出线妨碍了 SHARP 测距传感器的安装, 在此我们用螺柱把测距传感器顶到前面进行安装。如果觉得有点不美观, 可以剪一个小挡板安装在前面, 这样还有利于避免下面四只红外管受环境光的影响。如下图所示:



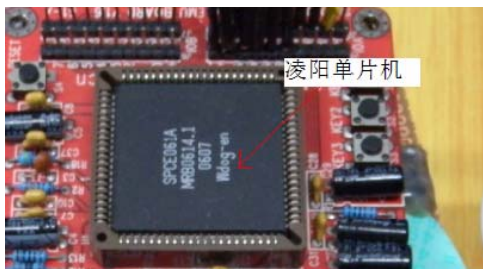
上图中光敏电阻是为了在迷宫行走中使用的，是用来检测光线强度的，占用 AD 通道的 P1.7 口。此时你可能要问：在控制板当中只有四路红外管的测量电路，而我们在小车上装了七个红外管，怎么分配他们呢？不用担心，因为这七只红外管是不需要同时使用的，我们可以把所有的插线引到插座下方，在完成不同功能时换插不同位置的管子。如下图所示，是不是很方便！



装配基本完成了，下面就是要来看看这双“核”的区别了，只有知道了他们各自的优缺点，才能更好的配合工作嘛！

二、双“核”比较

我们要做的“双核”小车，分别使用的 MCU 分别为 51 系列的 STC12LE5410AD 单片机和凌阳的 SPCE061A 单片机（如下图）。



我在卖圆梦小车之前，也有过想用凌阳 61 板完成圆梦所有功能的想法，也做了一些尝试，但最后我还是选择了 51 机，而把凌阳机作为上位机。其主要原因有：

①：要想让圆梦车动起来，首先就必须解决 H 桥驱动的问题，而 SPCE061A 单片机只有两路 PWM，我们可以用其对 H 桥进行电平控制。但是问题来了，他的两路 PWM 分别是占用和时钟 A 和时钟 B 的，这样在进行小车左右轮测速时就没有时钟可以用了。而 51 的 STC12LE5410AD 单片机有四路 PCA，只要让其两路工作于 PWM 模式，这完全可以满足使用的需求。

②：SPCE061A 单片机的两路 PWM 的占空比只能从 1/16 调整到 16/16，这在粗糙控制车速时还能应付过去。但如果你想尝试 PID 等高级控制时，这样的调节范围似乎就有点显得太小了。就拿全压为 3.3V 算，控制电平调节一次输出电平就有 0.2V 以上的变化。但 51 机的 PCA 工作于 PWM 模式时，占空比可以从 1/256 到 256/256。只要你的编程能力可以的话，用这样可调宽度去进行 PID 的话，应该会满足学习需求了。

③：其次就是小车的车轮脉冲计数了，因为精确的脉冲，可以为我们以后为小车实行走固定距离控制提供很大的方便。而凌阳机在外部中断时只支持下降沿触发，也就是说小车的车轮转一圈，凌阳机只能检测到 50 个脉冲。而在这里 51 机的优势又来了，他的 PCA 模式支持上升沿和下降沿触发，因此对于 51 机来说的话，他能捕捉到 100 个脉冲。这可以大大的提高对小车的控制精度！

从分析来看，小车的各种反馈信号还是用 51 机更合适了。当然了，凌阳机也不是无用武之地，在处理语音方面的问题时，他可是很能干的啊。所以凌阳机更适合进行人机的对话工作。

三、任务分配

51 单片机任务：

- ①：小车 H 桥驱动电平控制，运行方向控制
- ②：小车速度检测，码盘脉冲计数。

③：红外管信号检测、sharp 距离检测、光线强度检测及左右电机电流检测

④：接受上位机命令及检查信息反馈

⑤：各种子功能实现

凌阳单片机任务：

①：液晶屏驱动，检测信息动态显示

②：语音播放及语音识别

③：发出相应控制命令

④：KEY 键扫描

两机各自的任务分配完成了，我们还需要对这“核”的 IO 端口作一个合理的分配，分配好各端口任务后最好把他们记载在一个文档中，以便以后在编程时查找使用。

四、IO 端口分配

STC12LE5410AD

端口	实现功能
P1.0~P1.3	红外对管信号检测（AD 转换）
P1.4~P1.5	左右电机电流检测（AD 转换）
P1.6	SHARP 测距传感器距离检测（AD 转换）
P1.7	光线强度检测（AD 转换）
P2.0~P2.2	左电机运行控制
P2.4~P2.6	右电机运行控制
P3.0~P3.1	UART 通信接口
P3.3	红外管启用信号
P3.4	指示灯
P3.5~P3.7	左右马达测速和脉冲计数

SPCE061A

端口	实现功能
IOA0~IOA2	键盘按键
IOA3	RS 端(液晶显示)

IOA4	RW 端(液晶显示)
IOA5	E 端(液晶显示)
IOA6	PSB 端(液晶显示)
IOA7	RST 端(液晶显示)
IOA8~IOA15	数据口(液晶显示)
IOB7,IOB10	UART 通信接口

好了，下面只要双机协同合作就行了。既然要合作，那么双机就要交互相流，这就用到通信了。

五、双机交流

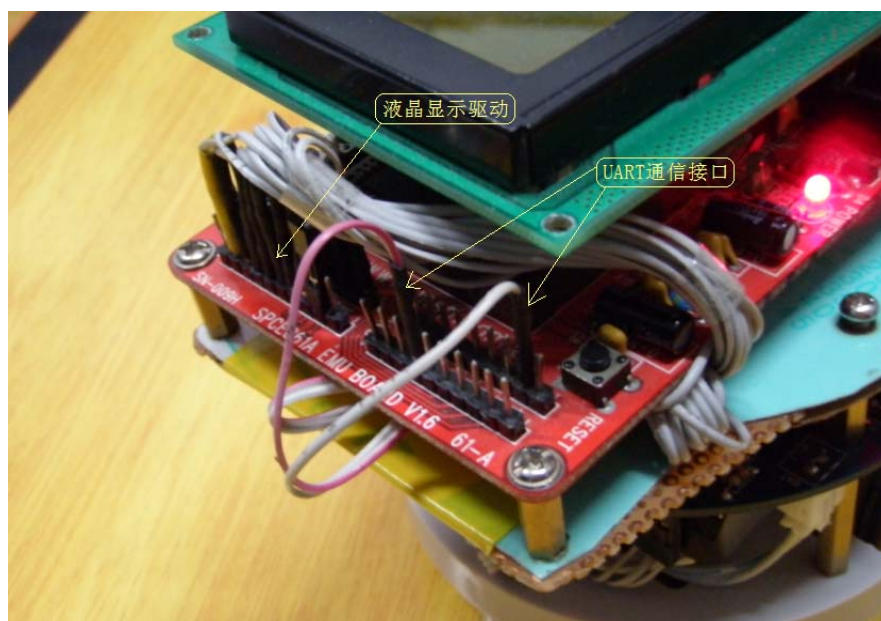
51 机和凌阳机交流采用 UART 通信，由于距离不远，波特率较低，一般不会出错，在这里我们可以指定一个简单的通信协议来验证是否数据正常。

基本通讯格式：

标准 UART 格式 —— 19200 8 N 1

帧格式：开始标志(1 字节) 地址(1 字节) 数据 1 字节) 校验和(1 字节)

至于接线，我想大家都会明白的，两条线就搞定了。就是 51 机的 RXD 口接凌阳机的 TXD 口，51 机的 TXD 接凌阳机的 RXD，相互交叉就行了。51 机的通信口分别在 P3.0 和 P3.1 口，凌阳机的通信口分别在 IOB7 和 IOB10 口，如图所示：



我们可以在两个单片机中分别定义两个缓冲区，分别为待发送数据信息缓冲区和接受数据缓冲区。至于各自的通信程序可类似于下图：

```

072 //*****
073 //功能：UART通信中断服务函数
074 //时间：2008-11-19
075 //编辑：秦小勇
076 //*****
077 void UART_Interrupt(void) interrupt 4
078 {
079     if(TI==1)
080     {
081         TI=0;
082         switch(To_State)
083         {
084             case 1: SBUF=0xff;To_State=2;break;
085             case 2: SBUF=To_Num+1;To_State=3;break;
086             case 3: SBUF=To_Code[To_Num];To_State=4;break;//发送数据
087             case 4: SBUF=(To_Num+1+To_Code[To_Num])*0xff;
088                     To_State=1;To_Num++;
089                     if(To_Num>39) To_Num=0;break;
090             default:break;
091         }
092         Work_Display = 1;
093     }
094     else
095     {
096         RI=0;
097         if(SBUF==0xff) //判断开始标志
098             Rx_State=1;
099         switch(Rx_State)//接受数据
100         {
101             case 1: Rx_State=2;break;
102             case 2: Rx_Dizhi=SBUF;
103                     if(Rx_Dizhi>10) Rx_State=0;
104                     else Rx_State=3;break;
105             case 3: Rx_Data=SBUF;Rx_State=4;break;
106             case 4: if(((Rx_Dizhi+Rx_Data)^0xff)==SBUF) Rx_Code[Rx_Dizhi-1]=Rx_Data;
107                     Rx_State=0;
108                     break;
109         }
110     }
111 }

```

51 机 UART 服务程序

```

//=====
//功能：UART中断服务程序
//时间：2008-11-19
//编辑：秦小勇
//=====
void IRQ7(void) __attribute__((ISR));
void IRQ7(void)
{
    unsigned int Num,Data;
    Num = *P_UART_Command2;
    if((Num&0x80)!=0)
    {
        Data=*P_UART_Data;
        if(Data==0xff) //判断开始标志
            Rx_State=1;
        switch(Rx_State)//接受数据
        {
            case 1: Rx_State=2;break;
            case 2: Rx_Dizhi=Data;
                    if(Rx_Dizhi>40) Rx_State=0;
                    else Rx_State=3;break;
            case 3: Rx_Data=Data;Rx_State=4;break;
            case 4: if(((Rx_Dizhi+Rx_Data)^0xff)==Data) Rx_Code[Rx_Dizhi-1]=Rx_Data;
                    Rx_State=0;
                    break;
            default:break;
        }
    }
    else
    {
        switch(To_State)
        {
            case 1: *P_UART_Data=0xff;To_State=2;break;

```

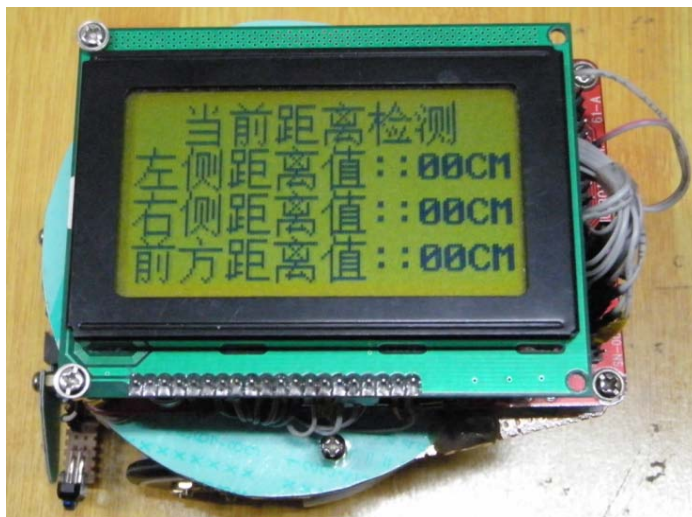
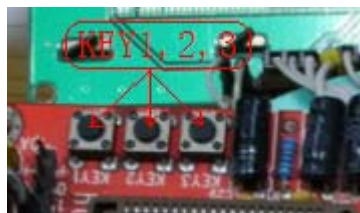
凌阳机 UART 服务程序

小车最后是要通过人机对话来实现多个功能，人机界面的构建还是很有必要的。双“核”之间交流的问题解决了，人与“核”之间的交流靠什么呢？那就用到人机对话了。

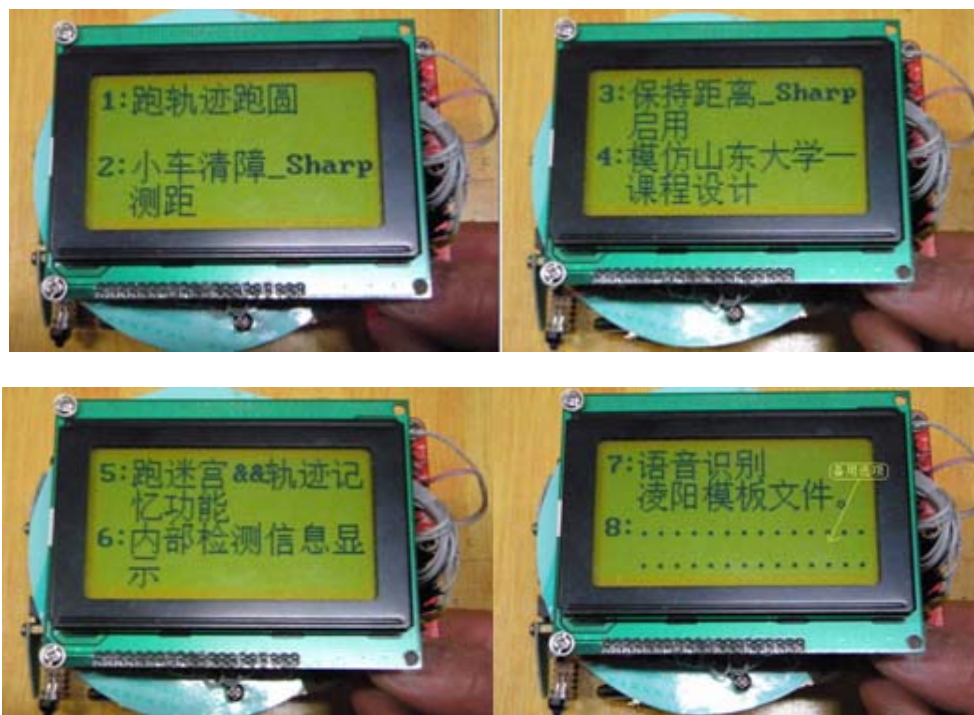
现在 51 机引脚已经所剩无几了，而凌阳机还有相当丰富的接口，我们可以再上面加一个液晶显示屏。这样，51 机检测的各种信息就可以显示了，此时我们离智能型的小车又迈进了一步！

六、人机界面

双“核”之间的交流是靠 UART 通信，而人机之间的交流就要通过键盘和显示器等设备。键盘我们就使用 61 板上自带的三个按键（如右图），这对于小车来说已经够用了。由于凌阳单片机还具有强大的 DSP 功能，我们还可以利用语音进行信息交流，这同时也增加了小车的娱乐性。如果有条件的话，在加一个液晶显示器就跟完美了，通过液晶屏可以很仔细的现实小车的各种检测信息，下图是我构建的一个进行距离检测显示的画面。这样在调试时，通过按键切换不同的画面，我们可以很方便的了解单片机内部的各种检测信息了。



小车最后是要完成多个功能的，具体是怎样进行功能切换的呢？当然了，在这里我们还是通过液晶屏和键盘来进行切换的（如下图构建的几个画面）。在液晶屏处理程序中通过返回一个值，然后把一个对应的子程序代号通过串口发送到 51 机，让其 PC 指针转向对应的子程序段，从而实现功能的切换。



那么 51 机又是怎样进行功能切换的呢？我们可以在 51 机中加入一个模式选择的函数（如下图）。同时为了子程序能够重复运行，在进行功能退出时还需将在该子程序中使用的变量进行清零。

```

091 //*****
092 //功能：模式选择
093 //时间：2008-11-24
094 //编辑：秦小勇
095 //*****
096 void Mode_Choise(void)
097 {
098     switch(Go_Code[4])
099     {
100         case 0x01:Run_Trace();break;//跑轨迹
101         case 0x02:Clear_Object();break;//小车清障
102         case 0x03:Keep_Distance();break;//保持距离
103         case 0x04:Run_Trace_ShanDong();break;//跑轨迹（模仿山东大学一课程设计）
104         case 0x05:Run_Mi_Gong();break;//跑迷宫
105         case 0x06:Motor_Clear();//清马达函数
106             Clear_All_Data();//数据清零函数
107             break;
108         case 0x07:MotorP_Clear();break;//清马达脉冲函数
109         default:break;
110     }
111 }

```

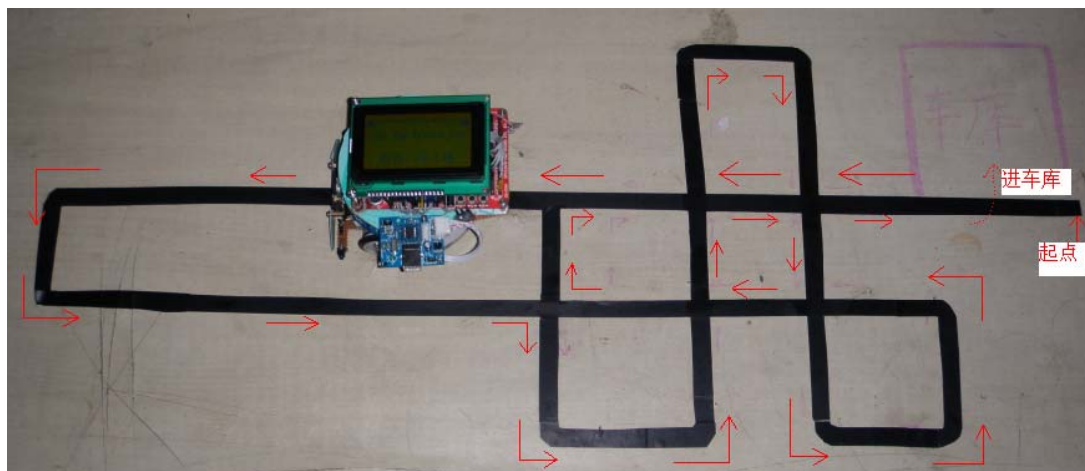
好了，现在准备工作已经基本做完了，而我也做了一些功能的尝试，在此来和大家一起分享。我们还是先去看看把！^_^

七、功能展示

功能①：模仿三东大学的一个课程设计

此功能是我在网上无意间看到的一个网页，然后模仿实现该功能的，小车的主要任务

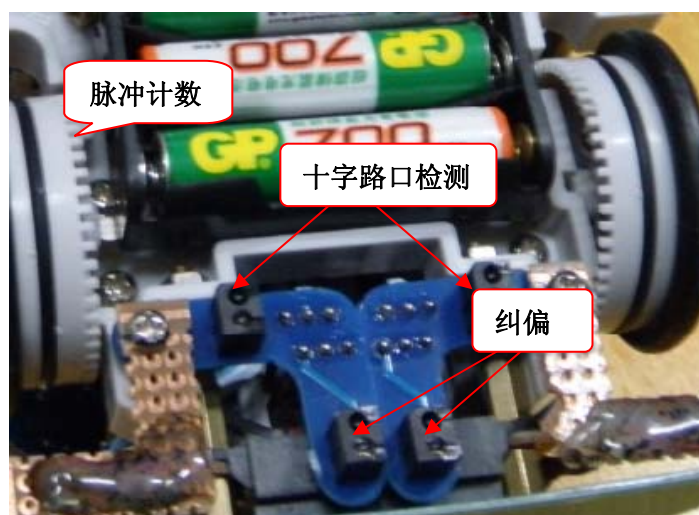
是从起点出发，沿着规定的路线走完，然后倒车回车库！很简单，关键就是小车在十字路口转弯时要控制好！行走路线如下图所示：



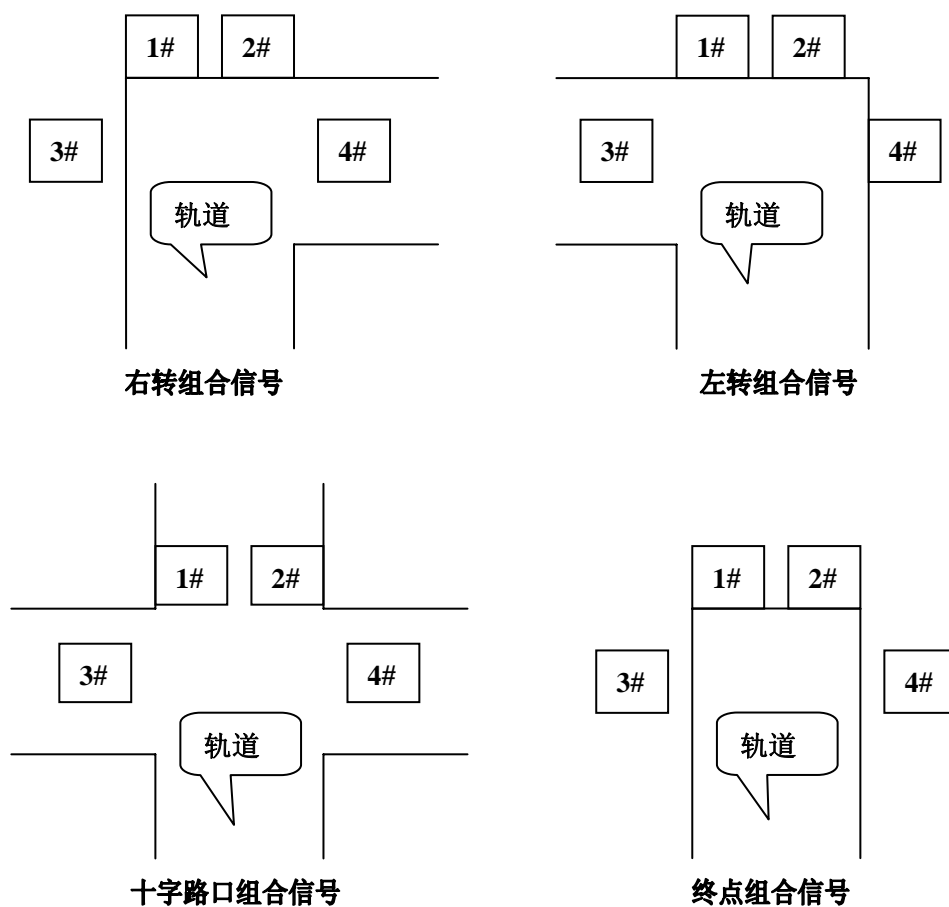
首先为了了解小车内部的具体检测信息，我们需构建一个信息显示界面（如下图所示），主要是用来了解四只红外管的轨迹检测情况（为 1 表示检测到黑线轨迹）。



传感器需求：在完成此功能时只启用了四个红外对管，两个用于纠偏，两个用于对十字路口的判断及计数等，还有就是对小车车轮脉冲计数控制回车库了。分布位置如下图所示：



控制过程是这样的：首先需要完成小车在行驶过程中的纠偏功能，我们可以让小车先学着跑直线，具体怎样控制好走直线，我想大家都是很清楚的。接下来就是小车行驶轨道过程中对反馈组合信号的处理，具体的控制策略如下图所示：



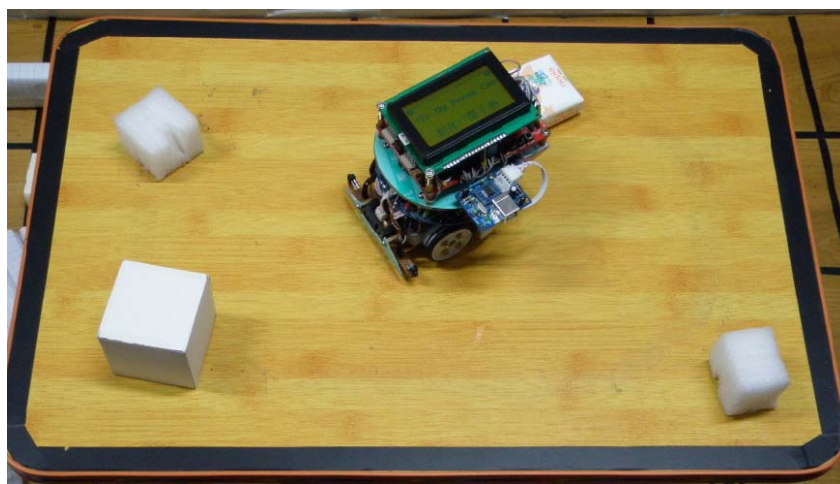
下面就是小车在十字路口的控制策略了，由图中可以知道，小车在不同的十字路口有不同的行驶方向，仅仅靠反馈的组合信号小车是无法识别的，这就需要在程序上给小

车一个相应的信号。因此我们就需要对小车在行驶过程中对所经过的十字路口进行计数了，这只要插入一段计数小程序就行了。受小车传感器能力有限，小车到达终点的回车库动作就简单处理了，到达终点后给小车在左右轮分别控制其行驶给定的脉冲就可以了。

具体的做法就是这样了，赶紧打开你的 Kill 开发环境去试试吧。（注：此功能可放在 51 机中实现）

功能②：小车清障

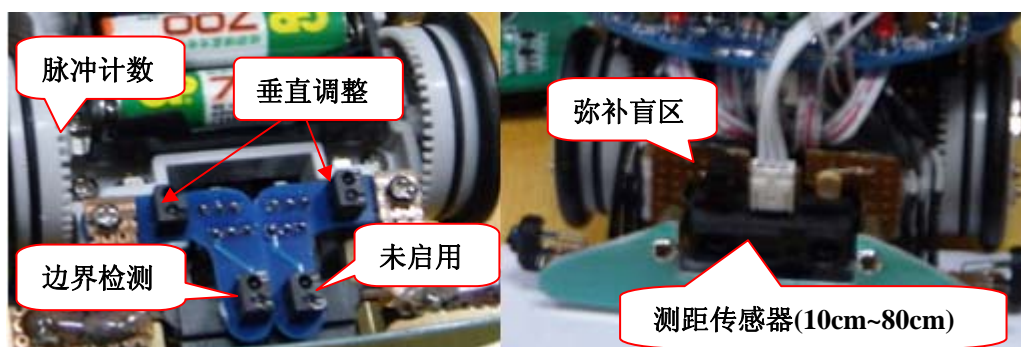
在此功能中，小车通过寻找桌面上的障碍物，然后将其清理出界。不过由于只使用了一个 sharp 的测距传感器，只能用来检查前方是否有障碍物，并不能完成对障碍物进行瞄准。所以在小车寻找障碍物时，在程序上稍微要讲究一些^_^!!在此功能演示中，只是小车基本功能的演示，目的是和大家一起分享。图中小车清障的桌面和地面是有一定高度落差的，如果在平面上进行清障的话，就要小车能有判断障碍物是在黑线内还是在黑线外的能力，有一点挑战性，希望哪位高手能实现之。



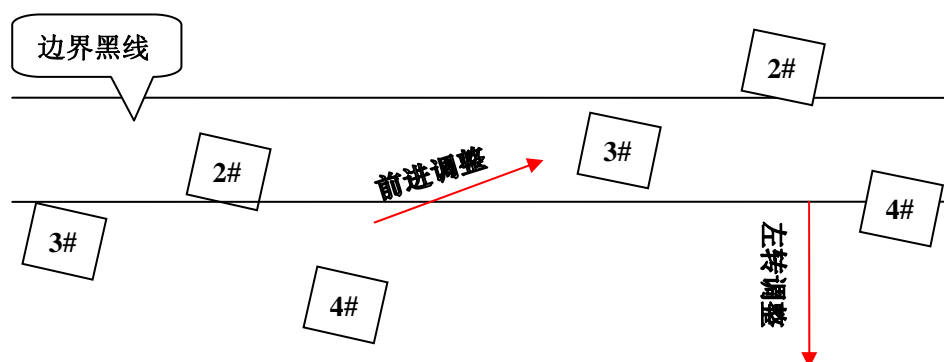
同样为了了解小车内部的具体检测信息，我们还是需要构建一个信息显示界面，如下图所示，以便及时了解小车的信息。

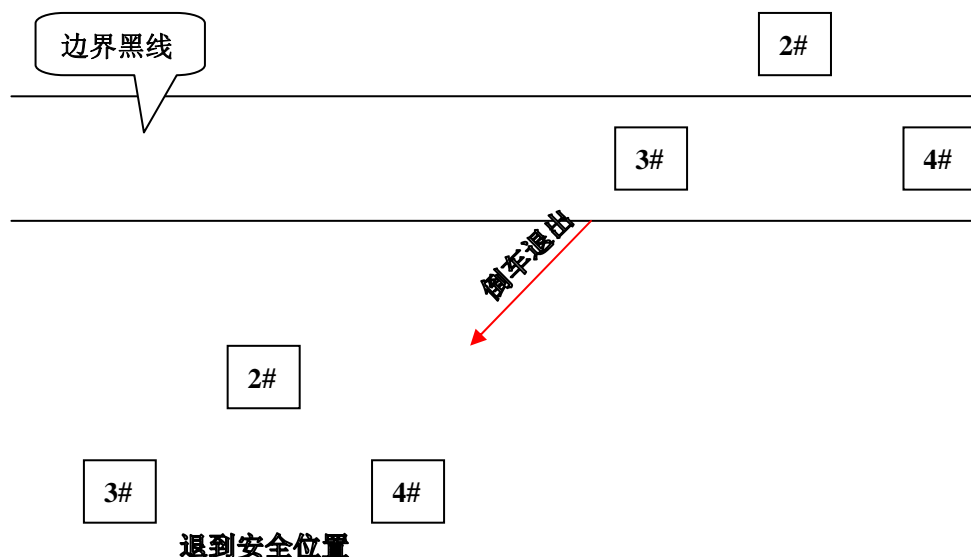


传感器需求：在完成此功能时同样启用了四个红外对管，一个用于检测边界，一个用于弥补 SHARP 测距传感器的盲区(0~10CM)，两个用于调整小车与边界垂直；一个 SHARP 的测距传感器（10CM~80CM）和脉冲计数了。分布位置如下图所示：

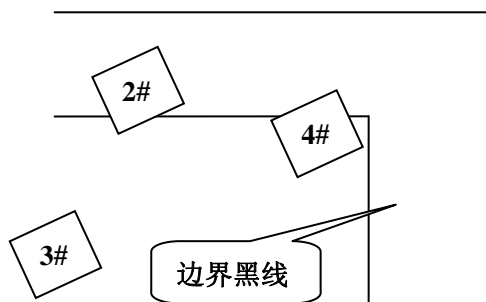


在程序上完成此功能不是太复杂的，大家可以尝试着去试试。关键就是小车在边界时要处理好，以便能让小车安全退出。具体的说就是：当小车的边界检测红外管或垂直调整红外管检测到信号时，小车要进行位置调整，直到垂直调整的两只红外管都在边界线上（此时小车与边界线垂直），然后让小车倒退指定的脉冲数退出边界线，再进行寻找障碍物。如下图所示：





为什么要进行小车垂直位置调整，然后倒车退出？主要是为了小车在角落边界处理时出界，如下图所示。如果在这种情况下直接进行倒车，小车会跑出边界线，其主要原因是因为后面没有安装检测管。如果想避免这种繁琐的处理方法，我们何不以将前面的 2# 红外管移装到小车后面，用来检测后方的边界，有兴趣者不妨试试效果！

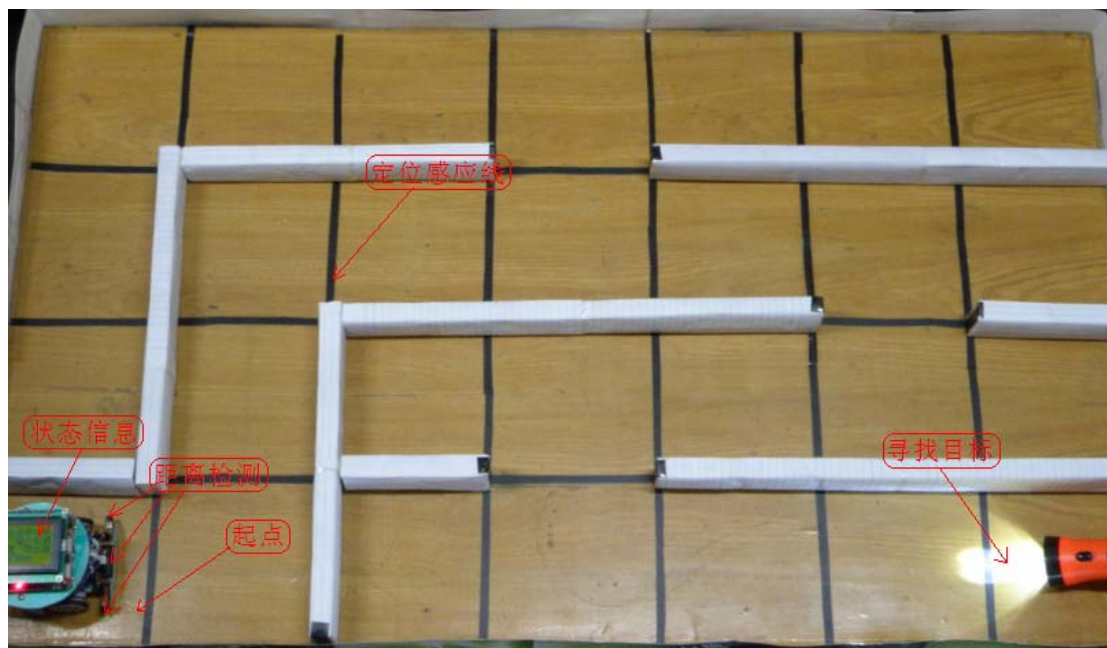


还有就是检测小车和障碍物之间距离的问题了，SHARP 测距传感器的有效距离在 9CM~80CM 间，有一个 9CM 的盲区。在这里我们对红外管信号的 AD 转换来进行盲区的弥补。在程序中可以这样处理：当红外管的检测距离在 8CM 内时就屏蔽 SHARP 检测的数据值（即认为 SHARP 进入盲区），当红外管的检测距离大于 8CM 时就启用 SHARP 检测的距离值。这样就完成了 0CM~80CM 间距离的检测。

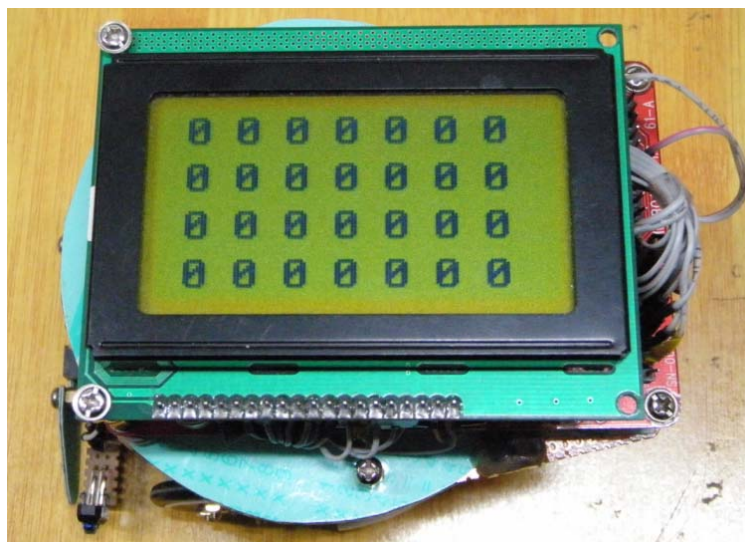
功能③：走迷宫_寻找光源

在此功能中，小车从规定起点出发，通过在迷宫中寻找光源(目标)，找到目标后进行语音提示并停车。在行驶过程当中小车要不能碰到墙壁，同时为了小车能够较快的找到目标，要小车不走回头路，这就需要小车具有轨迹记忆功能，图中的“轨迹感应线”就是为了小

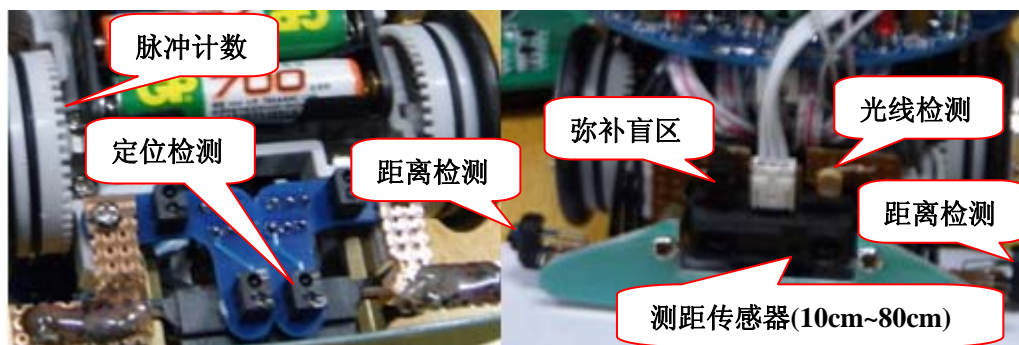
车能够进行轨迹记忆而设定的。虽然这种对小车的定位方法很简陋，但是使用在小车走迷宫时，效果还是蛮明显的，具体的行走环境如下图所示：



类似于上面，为了了解小车内部的具体检测信息，我们需要构建一个信息显示界面（如下图所示）。此画面主要是用来了解小车内部轨迹的记忆情况的。画面中的每一位计数值代表着实际行走中经过每一个单元格的次数。

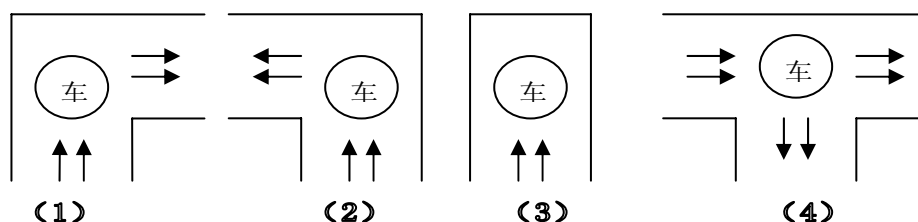


传感器需求：在完成此功能时同样启用了四个红外对管，一个用于检测定位感应线，一个用于弥补 SHARP 测距传感器的盲区（0~10CM），两个用于小车距左右墙壁的距离，SHARP 传感器（10CM~80CM），及光敏电阻的使用和脉冲计数了，分布位置如下图所示：



不过，需要注意的就是，左右的两个红外管在左右距离检测时，虽然在程序上和硬件上进行了背景光消除的处理，但还是受到环境光的影响的。所以在进行操作时还是尽量保持迷宫内部的光线强度尽量一致。下面再来看看控制方法吧！

在完成小车走迷宫时，小车首先就是在行走中不能碰到墙壁了，这种控制类似于小车跑直线的纠偏控制。当小车偏向左侧时，左轮速度调整为大于右轮速度；当小车偏向右侧时，右轮速度调整为大于左轮速度；当左右距离检测值相等时，左右轮设定速度相等。这样控制就能完成基本要求了。接下来就是小车在路口的行驶策略了，有些决策可以靠小车的检测信号决定，而有些决策需要程序参与选择。具体出现的情况如下图所示：



在图（1）中，当检测到左侧距离和前方距离小于 8CM，右侧距离大于 8CM 时，小车就执行右转程序。在图（2）中，当检测到右侧距离和前方距离小于 8CM，左侧距离大于 8CM 时，小车就执行左转程序。在图（3）中，当检测到左右和前方距离都小于 8CM 时，小车就执行掉头程序。而在图（4）中，此时小车有一侧检测距离小于 8CM，而另一侧和前方检测距离都大于 8CM，小车在进行决策时就有两种选择，这时就需要程序加入进行判断选择了。但是，在程序中进行转向选择时，是不能进行随机选择的，随机选择的结果会让小车走回头路，也可能照成在一个岔道上寻坏好几次才能进入新岔道继续寻找。因此，为了能让小车提高工作效率，这就必须使其具有轨迹的记忆功能。

具体实现是这样的：观测迷宫结构，我们可以把其分为 4*7 格，同样我们也在小车内建一个 4*7 的数组，数组中每一个元素都对应着实际迷宫中的一个单元格。小车在行驶到相应的单元格上时，就让相对应的数组单元进行计数加一操作。好了，到此时对于上面

图（4）中的问题就有解决的办法了。小车在决策时可以先比较其前方单元计数值和距离大于 8CM 的一侧单元计数值的大小，小车比较后向计数值小的方向行驶就可以了。是不是很简单？

在实际行走中，为了增加小车检测信号的可信度，我们还需要插入一段对检查信号的校验程序。就是当小车检测信号有阶跃变化时，就让小车前进三个脉冲的距离，然后再进行组合信号的检测，如果两次检测的信号一致，则信号可靠，否则维持小车的原行驶状态。

最后就是软件的结构了，受 PLC 控制中状态转移图的控制思想启发，我们在小车的控制中可以使用此种方法。可以把小车的每一个状态编制成一个对应的子程序，在控制中进行相应的状态切换就可以了。在 C 语言中 Switch 语句非常与其形式相似，所以可以 Switch 语句进行实现。

到此时，三个功能都已经介绍完了，如有错误或不妥之处还望大家见谅。而双“核”圆梦车的功能也远不止这些，比如说，你可以把凌阳单片机的语音识别功能加到圆梦车上，让我们的圆梦车也能听懂你的讲话等（还有一些潜在的功能等着大家去挖掘）。

在以上三个功能实现中，我们都是放在 51 机中实现的，而上位机只是给出一个与子程序对应的编号让 51 机 PC 指针转向相应的子程序而已。我们不妨可以试试把走迷宫等功能放到上位机中实现，很有意义。这样就要求双机之间的对话要有比较高的快速性及准确性了，有兴趣者可以试试^_^。

在这里，只是把我所做的和大家一起分享，同时也希望“圆梦小车”能成为大家学习和交流的平台。

秦小勇

2009 年 4 月 6 日

参考资料：

1： SPCE061A 单片机数据手册

2： STC12LE5410AD 单片机数据手册

3： TS12864 字符液晶使用手册（包括字库表）