ASURO 快速入门手册

关于 ASURO 的性能和可扩充性，学习性这里我就不介绍了！
关键就是 ASURO 很多客人刚拿到手的时候，一头雾水，不知道强大的功能如何下手，要学习什么？
在这里我简要的介绍一下：

一， ASURO_kit 焊接时应该注意以下问题

ASURO_kit（ASURO_KIT 散装套件的意思），那么组装的时候要注册以下问题：

ASURO 的散件，就是让学生认识电子元件的同时，学习组装，这里就体现了个人的细心程序和动手能力了。如：

1> IC 方向【IC 缺口标记方向对应线路板上的缺口标识方向】;

2> 二极管方向【普通 LED 发光二极管和红外传感器一定要注意极性，包括三脚的 LED 灯都是有极性的】;

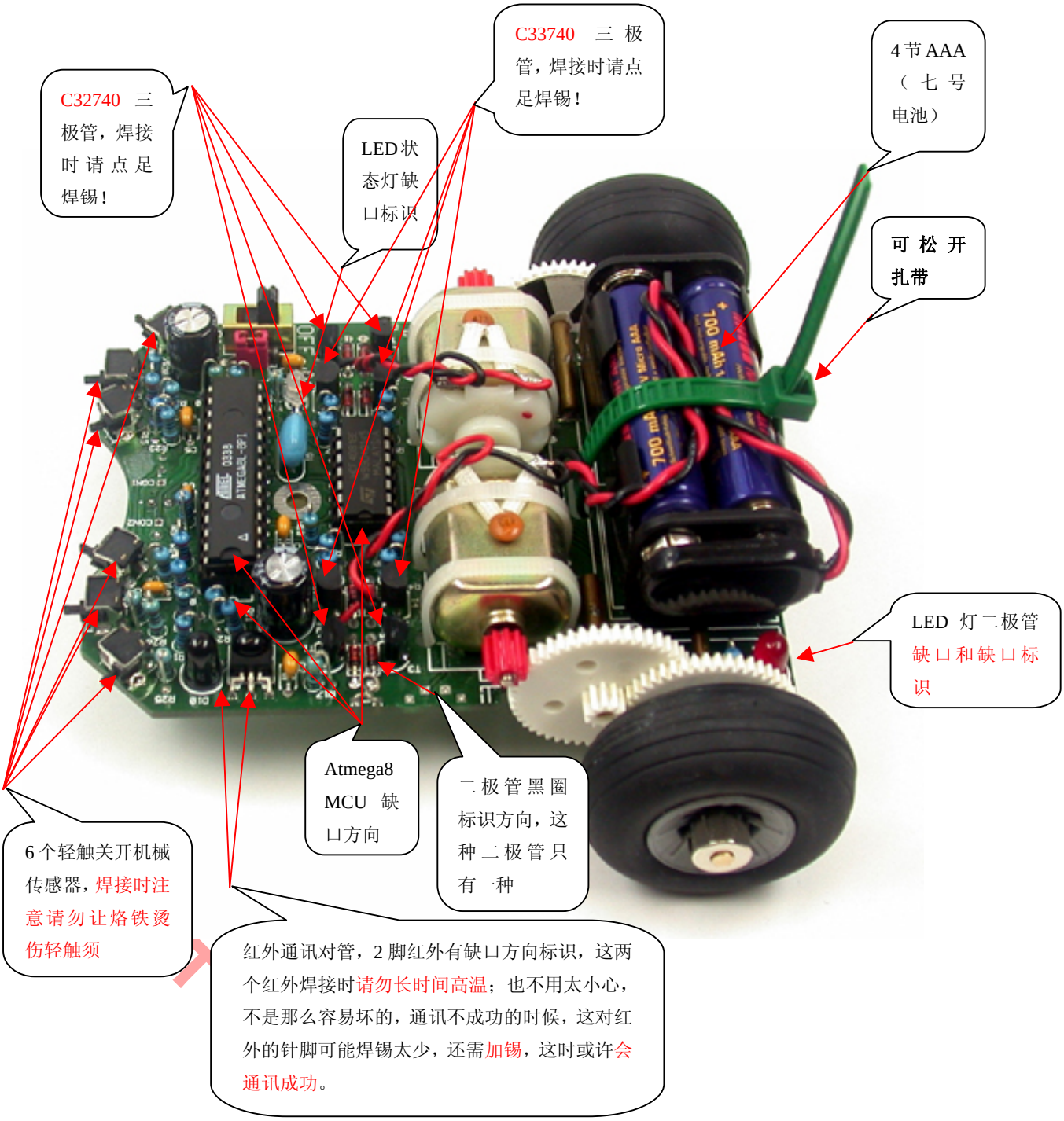
3> 三极管方向【8 个三极管搞错任一个或几个搞错，会颠倒马达的转动顺序或不动，马达不动可以检查三极管问题，出厂的成品机除外】;

4> 马达和电源导线极性【马达极性弄错也会造成传动方向混乱，电源方向弄错，整机开不了机或出现其它异常】;

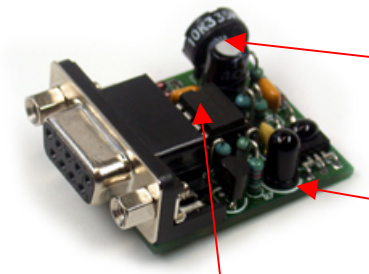
5> 长、短铜标装配问题【长、短铜杆装配一定按照装配说明书上的步骤来操作，一定先刮要焊接位置的铜杆，或者很难粘得到焊锡，并且焊锡总是像水珠一样在铜柱上打转，就是焊得上也焊得很难看。并且，所有操作之前，最难焊也是最先焊的就是铜杆，因为铜杆焊接时温度稍高，如焊好红外再焊铜杆或许对已经焊好的红外有影响。千万别小看这个工作，焊得不好，会关系到整个外观效果和齿轮的转动是否顺畅。】

小技巧：把刮好焊接位置的铜杆用烙铁按在 PCB 线路板上的焊接位置上，在烙铁和铜杆接触的位置加少许焊锡后稍等十稍钟（这个步骤是为了预热铜杆），当铜杆有一定的热量后，在铜杆和 PCB 线路板接触的地方用手拿锡线加锡，这时是很容易操作的，然后加足焊锡后，用金属小工具按住铜杆固定位置，移走烙铁，冷却后，该铜杆的焊接工作就大工告成了！

以下是作者的“蜘蛛网图解”，请大家细心品味！



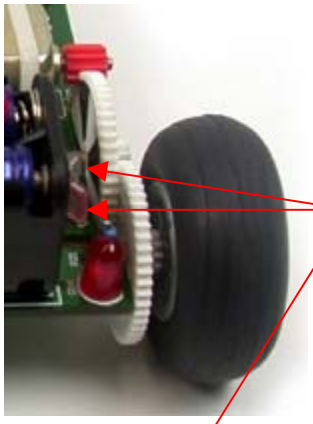
RS232 红外通讯板



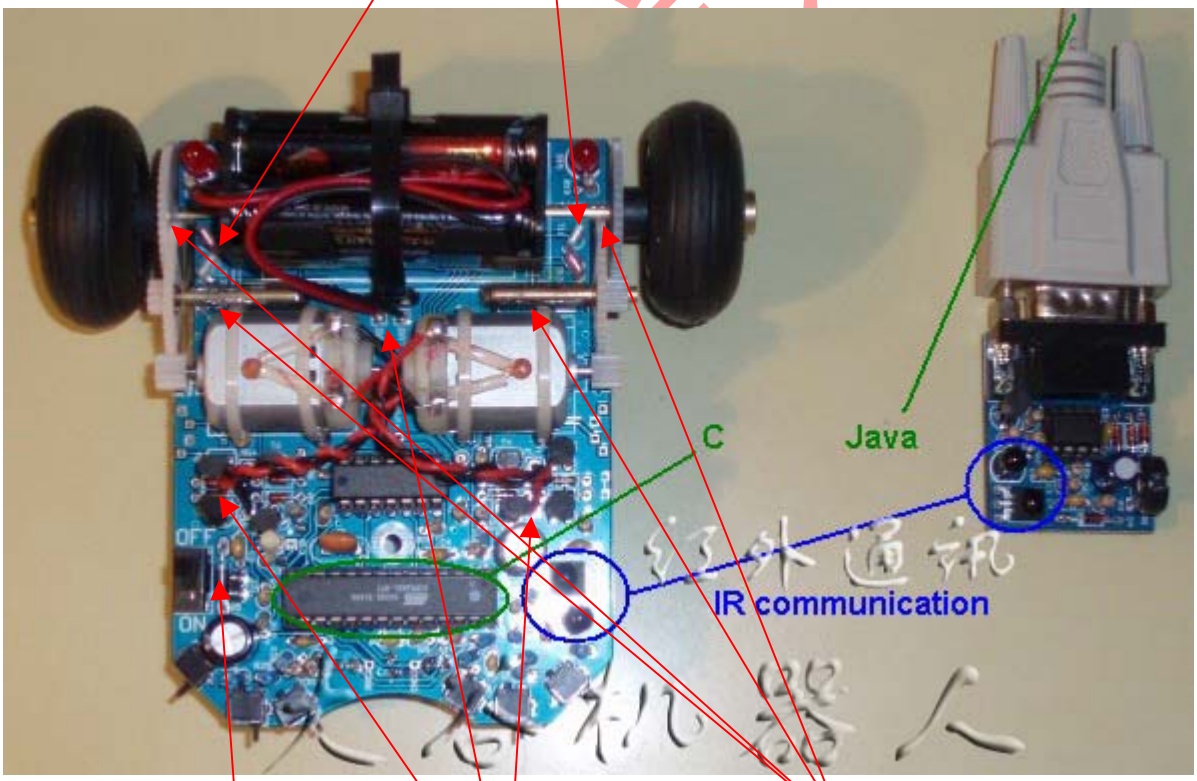
IC 缺口和缺口标识

红外通讯对管，2 脚红外有缺口方向标识，这两个红外焊接时请勿长时间高温；也不用太小心，不是那么容易坏的，通讯不成功的时候，这对红外的针脚可能焊锡太少，还需加锡，这时或许会通讯成功。

可调电位器，和 ASURO 通讯前，应该把 RS232 红外通讯板超级终端里面进行调，当在超级终端里面不可以输入字符时，微调此电位器，直到出现你输入的字符和最少乱码为止。注：用 falsh 软件和 ASURO 通讯（通常就是我们说的传程序、切换程序）时，要关掉调试用的超级终端，因为占用了同一个端口。



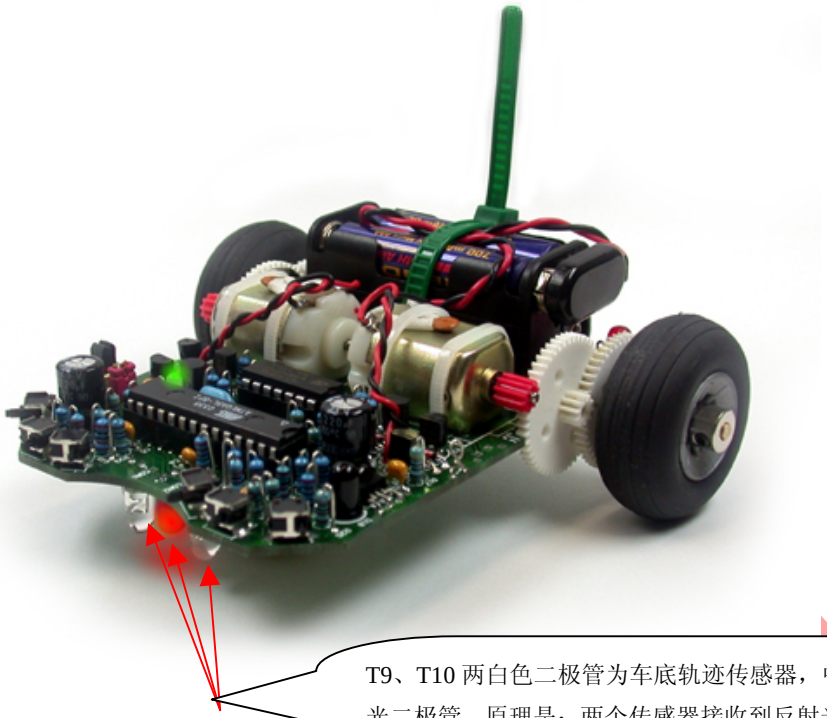
齿轮转速编码器，两个红外位置极为重要，一收一发。D13、D14 为红色，T11、T12 为白色。放大凸头均朝外面的齿轮上贴的编码方向，方向一定不可以弄错，两对红外编码器在 PCB 板上的位置并不是对称的，请大家注意。另外，在测试步骤里分别转动这两个轮子状态灯没有闪烁时，有可能是这四个红外没有和 PCB 线路板垂直，这种情况用摄子扳正时就迎刃而解。



二极管白圈方向标识

注意:马达，电池盒，红色导线外为“+”极，黑色导线为“-”极，PCB 线路板上均有标识“+”、“-”极标识，请勿弄错。

注意长、短铜杆的前后位置和上下位置，所有焊接装配工作之前，最先做的就是焊铜杆，并且焊接前一定要刮铜杆，这样更容易粘锡。千万别小看这个工作，焊得不好，会关系到整个外观效果和齿轮的转动是否顺畅。



T9、T10 两白色二极管为车底轨迹传感器，中间为红色 LED（D11）发光二极管，原理是：两个传感器接收到反射光的参数大小，来判断黑色轨迹的位置。LED 二极管和两个白色的红外传感器有缺口标记和 PCB 线路板上有对应的缺口标识，焊接时一定要注意极性。

二， 成品测试步骤

成品测试程序，是 ASURO 出厂时 IC 自带的程序，此程序有一个测试时钟，就是在指定的每个指定的时间段测试或展示不同的功能。当你心爱的 ASURO 费了一番心思组装完成后，这还没有完，完成这个测试程序测试 OK 后，算是大功告成了。

打开 ASURO 开关，出现以下测试时间段：

1>显示灯测试

将小车开关打开，接着状态灯（D12）呈现橙色，后面的两个发光二极管（D15、D16）也会很淡的闪亮一下，接着应该是有以下状态：

- 状态灯（D12）呈现绿色，接着呈现红色
- 车底灯（D11）亮
- 后面左边的灯（D15）亮
- 后面右边的灯（D16）亮
- 所有的灯一起亮

2>数据传感器测试（T9,T10——车底两个无色的 LED）

如果状态灯（D12）呈现~~红色~~（说明书和程序都显示是绿色，实际操作是红色），那么预示着这个过程开始了，这个过程要进行大约 10s，随着传感器（T9,T10）对应的光线亮，相应的后面的灯（D15，D16）也会一起亮，如果传感器对应的光线暗，相应的后面的灯也会熄掉。将小车车低对着日光灯，此时后面的两个灯都会亮，而用手挡住左边传感器T9,相应的左边的灯D15 会熄灭，再用手挡住右边传感器T10,相应的左边的灯D16 会熄灭（即，如果左边的T9 光线强，那么后面左边的D15 也会跟着亮；如果左边的T9 光线弱，那么后面左边的D15 也会跟着灭）

3>接触开关检测

此时，所有的灯都熄灭，车子也不会动，这个过程将进行 15s。

用手轻轻敲击那些接触开关出现以下情况：

- K1：状态灯（D12）呈现~~红色~~（说明书和程序都显示是绿色，实际操作是红色）
- K2：状态灯（D12）呈现~~绿色~~（说明书和程序都显示是红色，实际操作是绿 色）
- K3：下面的灯亮
- K4：后面左边的灯亮
- K5：后面右边的灯亮
- K6：左边的轮子开始转动

4>速度感应器测试

数据传送指示灯（D11）开始亮这预示着下面一个检测的开始，这一个过程大概需要 15s

此时如果拿一张白纸放在传感器的前面，这时候状态灯会亮。首先把白纸放在左边传感器（T11）的前面，状态灯（D12）呈现绿色，然后把白纸放在右边的传感器（T12）的前面，状态灯（D12）呈现红色。白纸拿走后，状态灯（D12）熄灭。

5>引擎测试

此时后面的两个灯（D15，D16）同时亮，这个测试将要 15s。

- 左边的轮子开始从 0 开始加速到最大值 255，然后又减速到 0
- 左边的轮子开始反方向转，也是从 0 开始加速到最大值 255，然后又减速到 0
- 右边的轮子也会重复以上操作。
- 最后两个轮子一起按照上面方式转动。

6>IR 接口测试

如果这时候状态灯（D12）间歇式呈现绿色，那么最后的这个测试就开始了，这个测试将持续 15s 左右。

这个时候 IR 接口将会接收和传送数据。要获取这些数据，就要把我们前面测试好的接收器（RS232 或者 USB 接口）连接在我们的 pc 机上，然后启动我们上面设置好的超级终端。将接收器对着小车，注意距离不能超过 50cm。此时如果按键盘上的任何键，在超级终端的界面上应该出现你敲击的那个字符和这个字符的下一个字符。如，你如果按 q 则应该出现 qr，如果按 3 则应该出现 34，如果按 s 则应该出现 st，如果在一定时间内没有按键，那么在超级终端上会出现字母“T”

7>测试结束

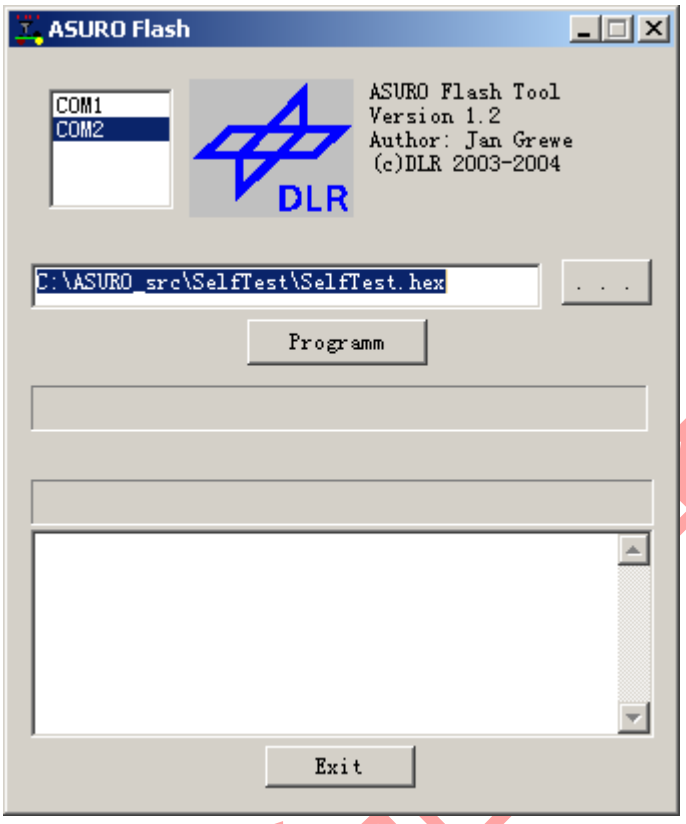
一次测试结束后，小车会循环重复以上所有过程，如果您遗漏了某些过程，可以重新测试。

三， 程序传送软件（flash）的使用介绍

1>打开工具

2>选择好你连接 IR 的 com 端口（RS232 是选 com1/com2 口，USB 发射/遥控板选 com1/com2 以外的 com 口调

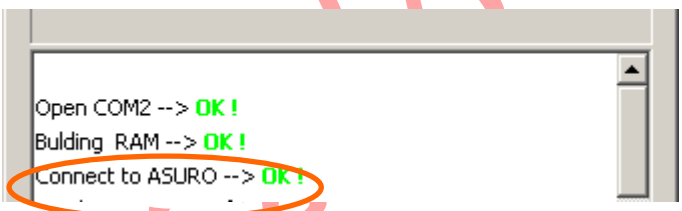
试）



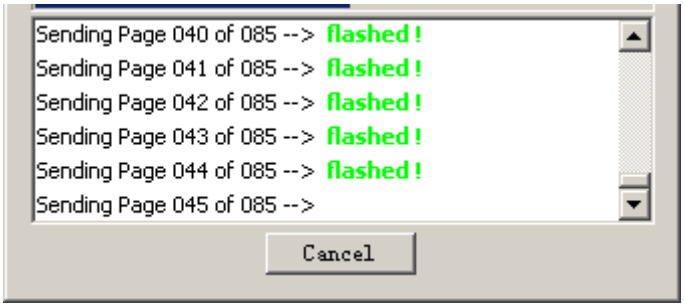
找到编译好的机器代码文件（.hex）

3>程序传送

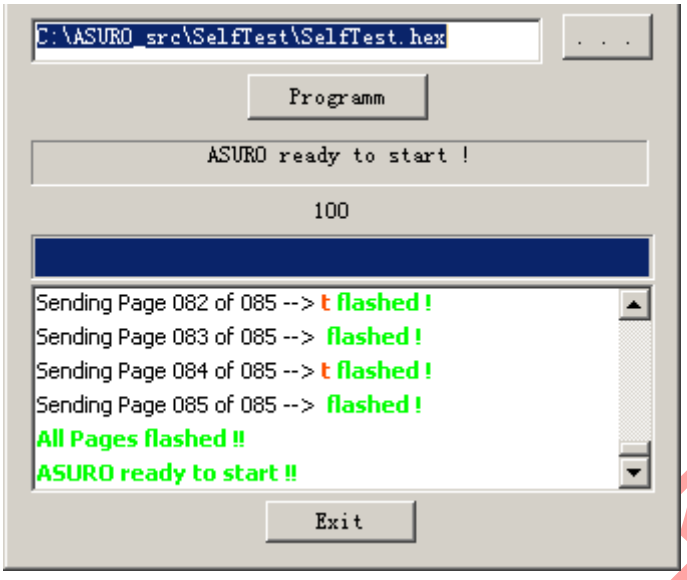
将 IR 接口放到距离小车 50cm 处，点击 Programm 按钮，在状态指示条到达最右边之前，将小车的开关打到 ON 的位置。如果连接上小车，则有以下信息出现



4>传送过程中，状态灯呈现橙色



5>传送结束，状态灯呈现红色或者绿色



这样就成功的将一个程序的十六进制代码传送到了小车。如果中间传送失败，则从第 3 步开始重做，直到所有的页面都传送到小车。

6>程序运行

此时将小车重新启动，即将将小车的开关打到 OFF 的位置，再打到 ON 的位置，小车就会运行我们刚刚传送的程序。

```
#include "asuro.h"

int main(void)
{
    Init();

    while(1);
    return 0;
}
```

此程序由于不包含任何动作，所以下车初始化完毕也不会有任何动作。

四，电脑无法跟车子通信啊,怎么办

无法通信的话：

情况一，用 RS232 与 ASURO 进行通信：

1，检察 RS232 和 ASURO 的红外发射对管，有没有焊接错极性！

2，com 口选择一般是，1 或 2，可以分别试一下，台式电脑如果只有一个 com 口的话，就选 com1.

3，RS232 上的可变电阻调到合适的位置，可以用超级终端进行测试，调到输入的字符或数字没有乱码为止。

4，RS232 和 ASURO 红外正对，距离 5-15CM 为宜。

5，用 FLASH 上传程序的时候，一定要先点击 programme,再打开 ASURO 上的开关！

情况二，用 USB 传输板与 ASURO 进行通信(需安装驱动)：

1，检察 ASURO 的红外发射对管，有没有焊接错极性！

2，com 口选择一般是显示的 flash 中最大的一个。

3，RS232 和 ASURO 红外正对，距离 5-15CM 为宜。

4，用 FLASH 上传程序的时候，一定要先点击 programme,再打开 ASURO 上的开关！

因 USB 传输板出货时都是测试过的，不会有品质问题，一般不能通信出在焊接问题或传程序时的先后顺序！

再无法解决，请加QQ群：53302059 旺旺：tyler_yyh TEL:13528107292/0760-88811951-806

五，软件调试

需要安装的软件(这里是光碟自带的旧的 AVR 使用方法，也可以下载新的 AVR 版本，但操作方法是不同的)

Flash 工具：将你自己编写的程序传送到小车 asuro

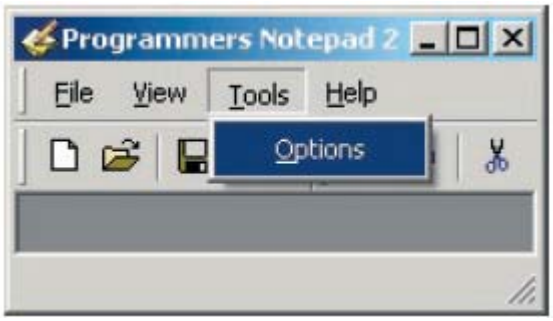
程序编写(Programmers Notepad 2, PN2)和编译（WinAVR）工具

我们提供的光盘上的例程（ASURO_src），放在您自己的硬盘上（如 c:\ ASURO_src）

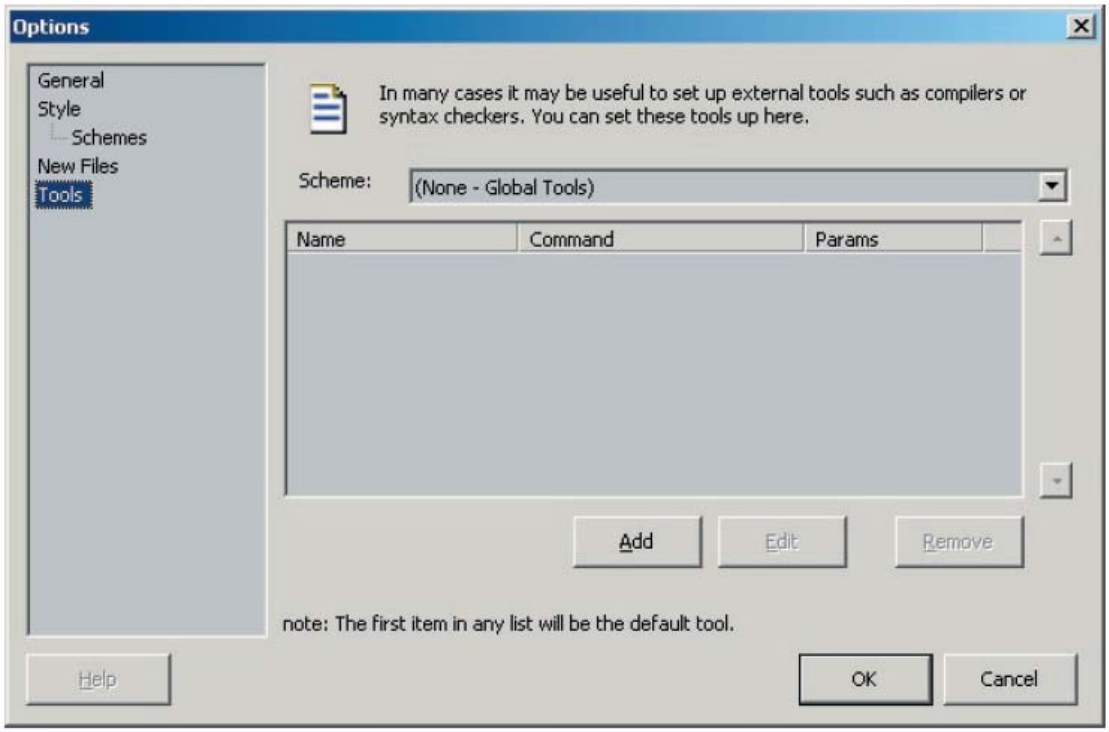
程序编写软件（PN2）的使用介绍

打开您安装好的 programmers notepad2

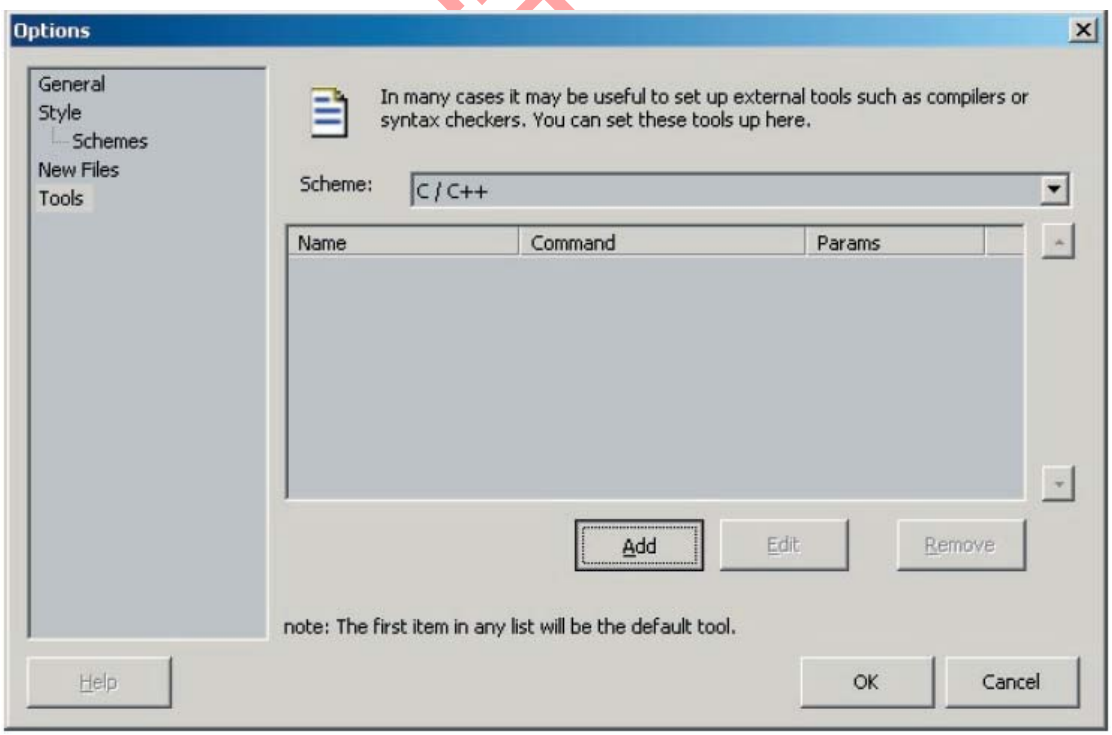
选择 Tools 下面的 option 菜单



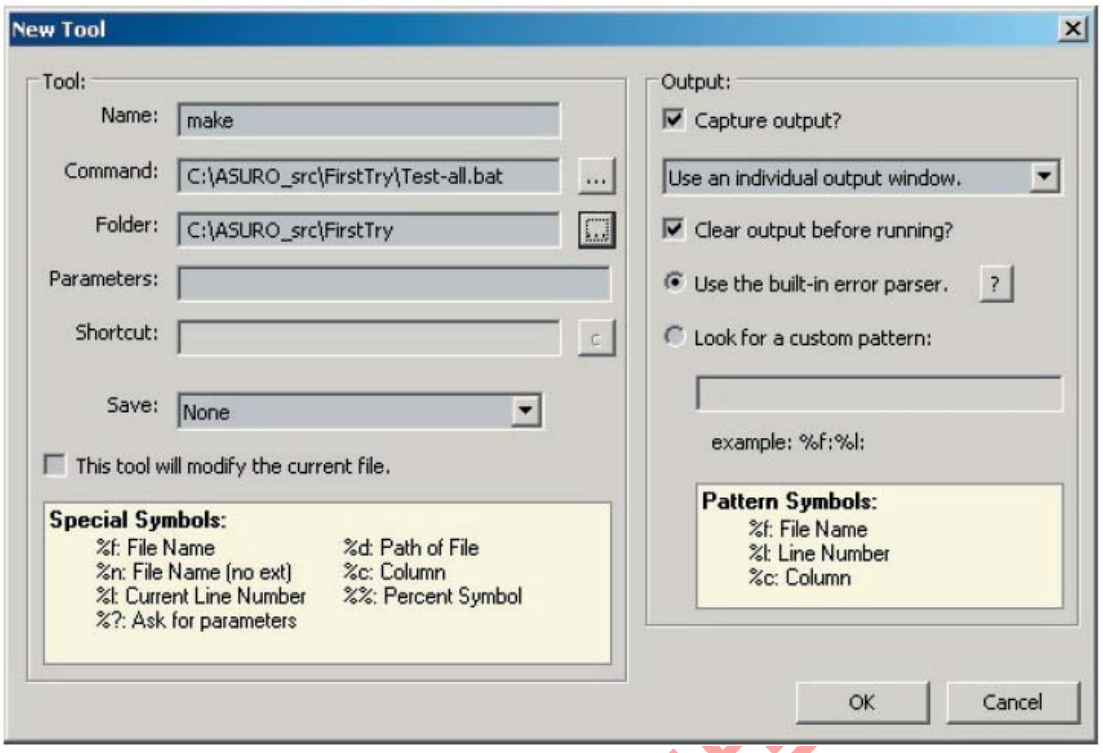
选择 Tools



在右边的下拉列表中选择 c/c++，再点击 Add 按钮



填入以下数据后点击 ok 按钮

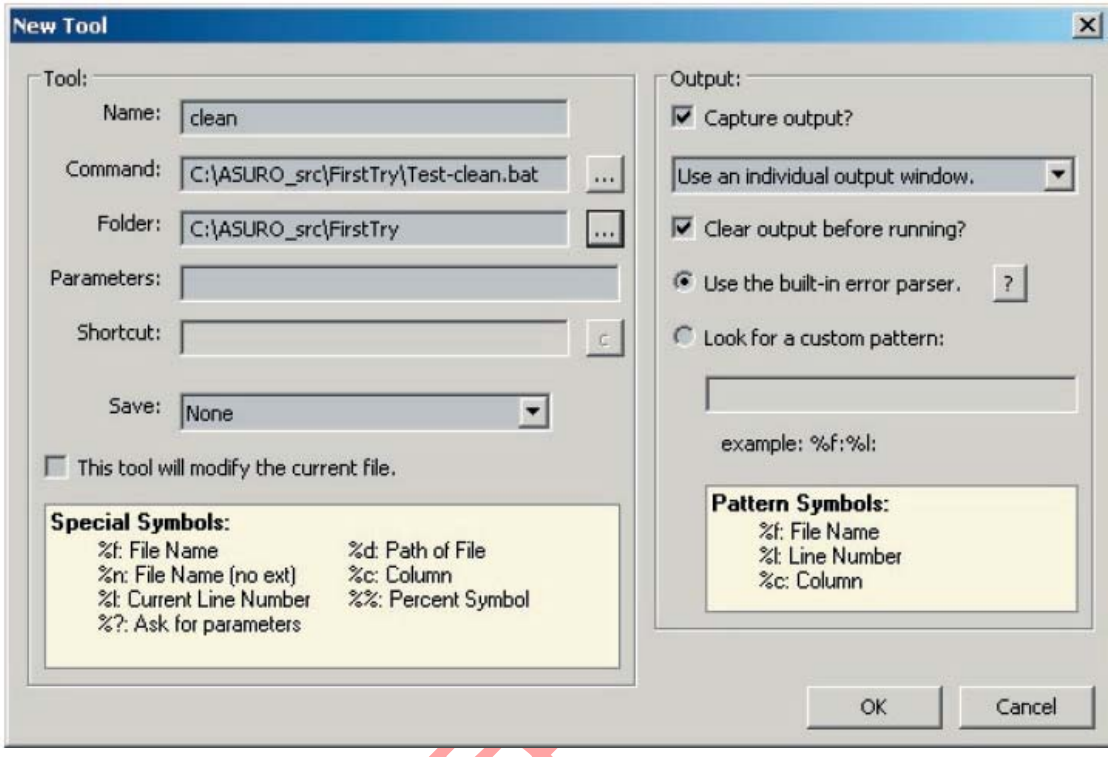


此时在软件的 Tools 菜单下面会出现一个 make 菜单

如果点击 make,那么程序会执行 c:\ ASURO_src\FirstTry 下的 test-all.bat 命令，用来编译源代码，生成的文件也会放在 c:\ ASURO_src\FirstTry 文件夹内。

同样的方法添加 clean 菜单

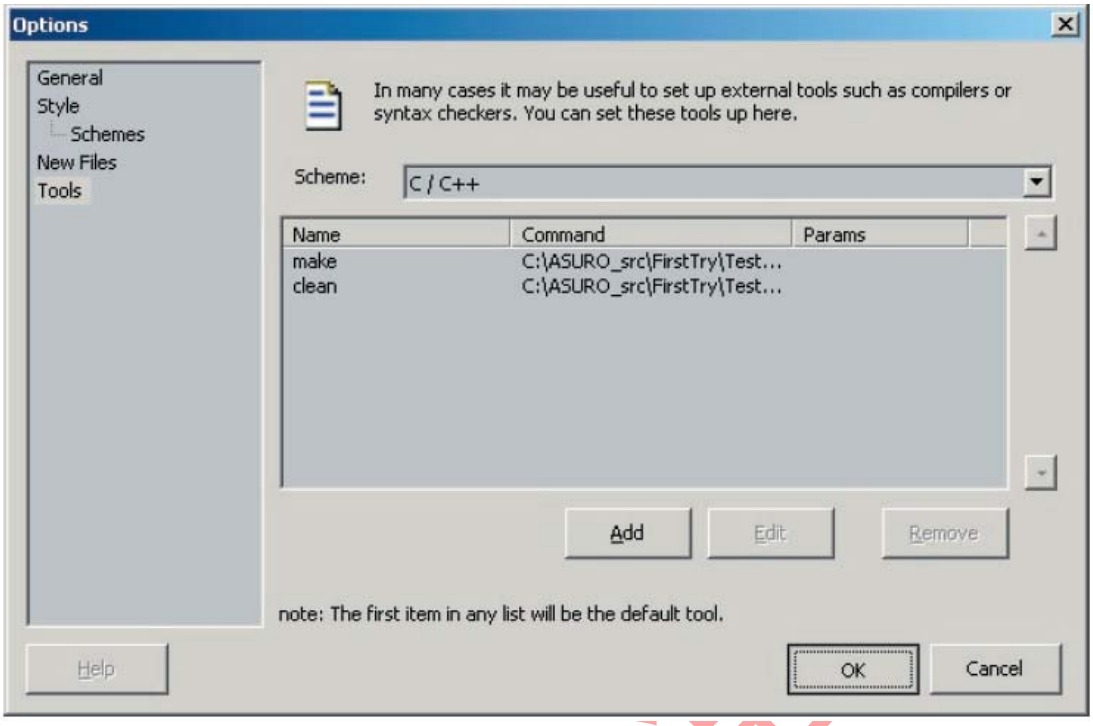
再点击 Add 按钮，填入以下数据，然后点击 OK 按钮



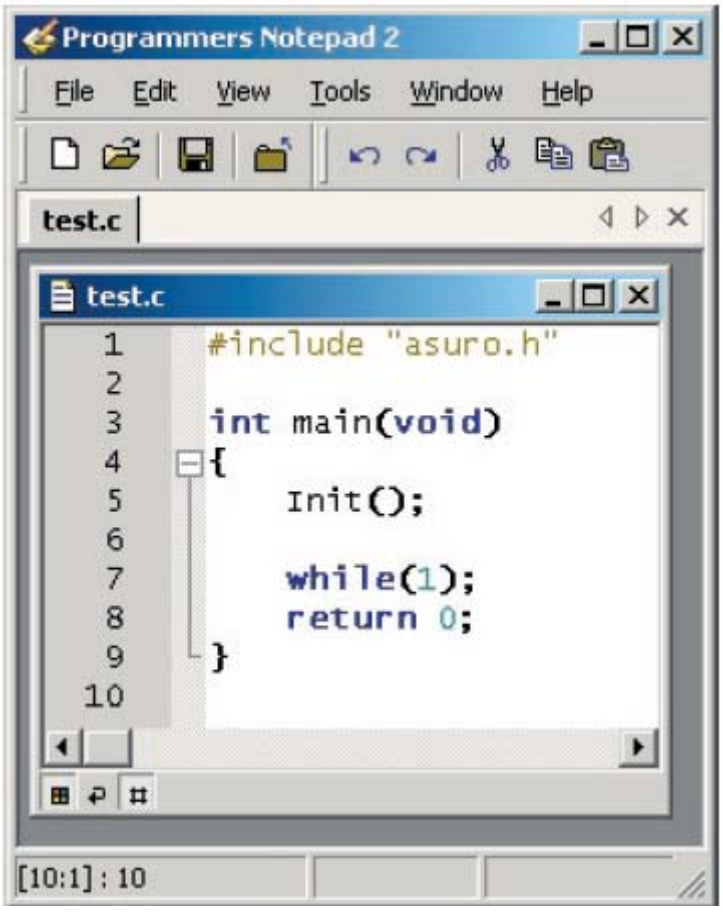
此时在软件的 Tools 菜单下面会出现一个 clean 菜单

如果点击 clean,那么程序会执行 c:\ ASURO_src\FirstTry 下的 test-clean.bat 命令，用来清除 c:\ ASURO_src\FirstTry 文件夹内的编译生成的文件。

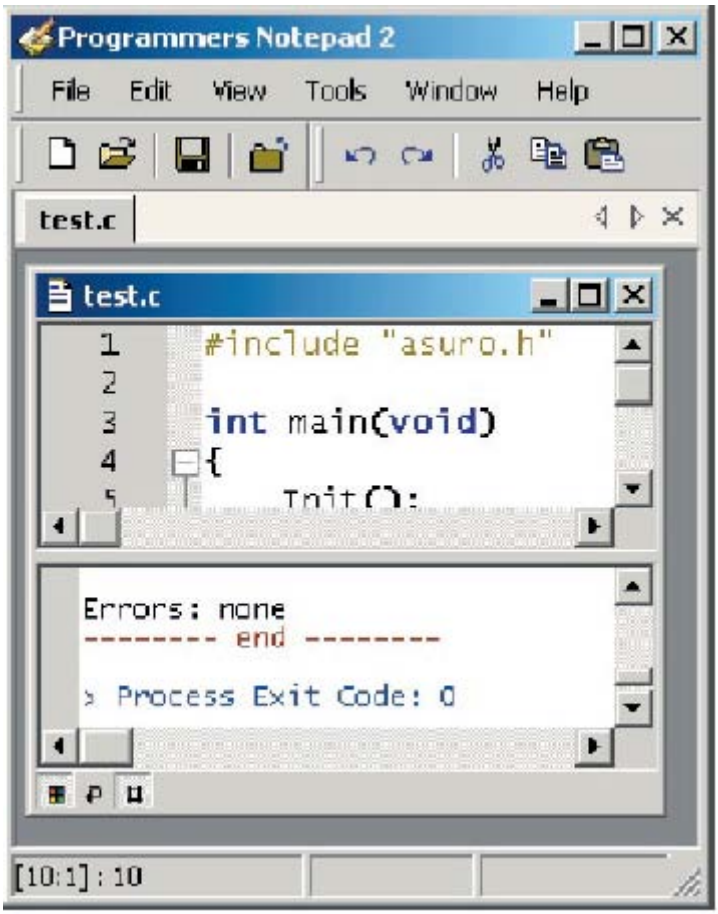
点击 ok 按钮，结束工具添加



打开 c:\ ASURO_src\FirstTry 文件夹内的 test.c 文件



选择工具 make



如果没有出现错误，那么程序编译结束。

编译的结果

- 从 test.c 和 asuro.c 生成了一个新的机器代码文件 test.hex
- 我们等会要把这个文件通过 flash 工具传送到小车的存储器。

编译原理

- 当我们点击 make 菜单时，test-all.bat 命令被调用，这个批处理文件里面只有一个 make all 命令，make 命令会被执行，而这个命令会在当前文件夹内生成一个 makefile 文件
- makefile 文件是一个没有扩展名的文本文件，里面定义了如何编译一个或者多个程序。
- 如果你的程序只有一个文件，那么你只需要将 test 程序中的 makefile 文件中的 TARGET = test，改成 TARGET = 『文件名』。如果以后您编写的程序包含有很多文件，则这些文件之间必须要以一个正确的方式链接起来。那么 makefile 文件也会复杂很多。
- 在例程中的这个 makefile 文件是为我们的 test.c 所写的。
- 注意：如果您不会写 makefile 文件，仅仅将它拷贝在你自己的文件夹内，那么你的文件只能命名为 test.c

- 如果想了解更多的makefiel文件请参照http://www.gnu.org/directory/make.html

清除编译文件

点击 clean 菜单即可清除掉我们编译好的.hex 文件。调试程序时，改好的程序要现存盘，才能够重新编译，建议重新编译之前，先用 Clear 清除掉原来生成的文件。

函数介绍

void Init(void)

用来将小车的微处理器复位到初始状态，它是程序真正开始前必须执行的一个函数，如果没有这个函数，那么处理器就不知道如何工作。一个最简单的程序至少要有以下代码：

```
#include “asuro.h”
int main(void) { // 定义变量
Init(); // 插入自己编写的程序
while(1); // 无限循环
return 0; //这一步不会被执行到
}
```

为什么我们要在 main 函数的末尾加上一个无限循环呢？这是因为正常情况下，main 函数会通过返回值 0 结束——return 标志着程序的结束，但是在 asuro 中，我们以前传入进去的程序可能仍然在微处理器的存储器中，也可能会被执行，这可能会导致一些不可预料的结果。为避免这些老的程序片断被执行，我们在程序当中加上一个无限循环，使这些程序片断永远没有机会被执行到，这样程序就可以按照我们预定的方式结束。

void StatusLED (unsigned char color)

用来控制状态灯（D12）的开与关，合法参数是：OFF, GREEN, RED 和 YELLOW。例如：StatusLED (RED);会将状态灯变成红色。

```
#include „asuro.h“
int main(void) {
Init();
StatusLED (YELLOW);
while(1);
return 0;
}
```

void FrontLED (unsigned char status)

用来控制前面的灯（D11）的开与关，合法参数是：OFF 和 ON。例如：FrontLED(ON);

void BackLED (unsigned char left, unsigned char right)

用来控制后面两个灯（D15 和 D16）的开与关，其中第一个参数描述后面左边的灯 D15 的状态，而第二个参数描述的后面右边的灯 D15 的状态，合法参数是：OFF 和 ON。例如：BackLED(OFF,ON);左边的灯灭，而右边的灯亮。

void Sleep (unsigned char time72kHz)

这个函数可以使处理器等待一段时间。处理器等待的时间由一个参数决定，而这个参数的最大值只能是 255，并且从一个 72kHz 的定时器计算周期。

例如：这个计算器会等待 3ms

$$==> \quad \frac{0{,}003s}{\frac{1}{72KHz}} = 216.$$

所以，sleep(216)会迫使处理器等待 3ms

void MotorDir (unsigned char left_dir, unsigned char right_dir)

这个函数控制小车两个轮子的旋转方向，一般在速度控制函数之前使用，合法参数 FWD (向前), RWD (向后), BREAK (停止)和 FREE (惯性滑行)。例如：MotorDir(FWD,BREAK);将会使左边的轮子向前，而右边的轮子停止。

void MotorSpeed (unsigned char left_speed, unsigned char right_speed)

这个函数用来控制两个轮子的速度，最大速度是 255。根据机械条件，发动家将会以大约 60 的值旋转。这个参数的值实际上控制的是发动机的功率，而实际的转速还受很多其他因素的限制，不如说摩擦力和坡面的倾斜度。

MotorSpeed (255,0);左边的轮子以最大速度转动，而右边的轮子不动，当然在这个函数调用之前，要有 MotorDir()来设置轮子转动的方向。

void SerWrite (unsigned char *data, unsigned char length)

这个函数通过串行接口 IR，以 2400Bit/s，8 位数据位，无奇偶校验，停止位 1，无数据流的方式从 ASURO 接收数据。参数描述呢要传送的字符，和传送的长度。例如：SerWrite (“Hello how are you?”,18); 如果启动超级中断，这时小车会将 Hello how are you?传送到超级终端

void SerRead(unsigned char *data, unsigned char length, unsigned int timeout)

与上面的函数相反，这个函数的作用是用于 ASURO 接收我们发出去的字符串，第一个参数用来保存字符串指针，第二个参数描述希望小车接收到多少个字符串，第三个参数描述的是延时时间。Timeout 参数用来防止当小车没有接收到足够多的字符的时候，无限等待。如果在延时时间之内没有足够多的字符接收到，那么这个函数将会被忽略，而接收到的字符将会被‘T’ (=Timeout)代替。如果你把 Timeout 参数定义为 0，那么这个函数将会一直等待，直到接收到所有期望的字符。

```
#include “asuro.h”
int main(void) {
char data; //用于存放接收到的字符串
Init();
SerRead (data,1,0); //读取数据
If(data==’G’){//如果接收到的字符是 G 那么小车就向前走。
   MotorDir(FWD,FWD);
   MotorSpeed(120,120);
}
while(1); // Eternal loop
return 0;
}
```

void LineData(unsigned int *data)

这个函数用来读取在小车车底两侧光敏晶体管的强度。你首先必须要定义一个能够存放两个 int 型数据的指针。这个函数将两个光敏晶体管的模拟量转换成数字量，而数组里面的第一个值存放的就是左边晶体管的数字量，第二个存放的是右边的。最大强度是 1023，而全黑的时候，就是 0。通常情况下，不会有这些极限数据检测到，而实际能够检测到的值就是在 0 到 1023 之间。例如：

```
unsigned int data[2];
LineData(data);
data[0] 描述左边晶体管的强度
data[1] 描述右边晶体管的强度
int main(void) {
unsigned int data[2];//定义数组
Init();
FrontLED(ON); // 前面灯亮
MotorDir(FWD,FWD); // 两个轮子方向都向前
while(1){

LineData(data);//读取晶体管的强度
if (data[0]>data[1]) //左边比右边强
{MotorSpeed(200,150);} // 左边轮子加速
else
{MotorSpeed(150,200);} //右边轮子加速
}
```

```
return 0;
}
```

void OdometrieData(unsigned int *data)

这个函数扫描反射的光敏元件（D13, D14）。当这两个传感器是激活的，那么这个函数就返回光敏传感器(T11, T12)的数模转换值。和 LineData()函数一样，这些数据同样的放在 data 数组中，其中第一个整数描述的是左边的传感器(T11)的数模转换值，而第二个整数描述的是右边(T12)的值。其中，最大的亮度是 0，而全黑的时候就是 1023。通常情况下，不会有这些极限数据检测到，而实际能够检测到的值就是在 0 到 1023 之间。

data[0] 描述左边晶体管的强度

data[1] 描述右边晶体管的强度

注意：这个函数并不是提供的旋转的数据，而实际上描述的是反射的光敏元件的数据。通过辨别明暗，计算明暗过渡的次数，而要计算旋转的次数，这个就留给编程员自己去做了。

unsigned char PollSwitch (void)

这个函数扫描接触开关(K1-K6)，并且返回一个字节，这个字节描述了哪个开关被触碰了。其中 K1 对应的是字节的第 0 位，K2 对应第 1 位，依次类推 K6 对应字节第 5 位。所以有下面结论：

Bit0 (1) -> K6
Bit1 (2) -> K5
Bit2 (4) -> K4
Bit3 (8) -> K3
Bit4 (16) -> K2
Bit5 (32) -> K1

所以如果开关 1、3、5 被触碰，那么返回的值就是 42 (32+8+2 = 42).

注意：如果想得到准确数据，电容 C7 必须要先放电，这可能要花费你一些时间。

操作示例

```
#include "asuro.h"
#include <stdlib.h>
//接收字符定义
#define DIARWD     'b'//减速
#define DIAFWD     'g'//加速
#define DIALEFT    'l'//左传
#define DIARIGHT 'r'//右转
#define DIASTOP    's'//停止

#define OFFSET     255
int STEP=40;//速度变化速度

int speedLeft,speedRight;//左轮速度，右轮速度

void IRFwd(void)//小车加速
{
      speedRight += STEP;
```

```
    speedLeft    += STEP;
    if (speedLeft < 0 && speedLeft >= -OFFSET) speedLeft = 1;
    if (speedRight < 0 && speedRight >= -OFFSET) speedRight = 1;
    FrontLED(ON);
    BackLED(OFF,OFF);
    SerWrite("\tgogo",5);
}

void IRRwd(void)//小车减速
{
    speedRight -= STEP;
    speedLeft    -= STEP;
    if (speedRight > 0 && speedRight <= OFFSET)    speedRight = -1;
    if (speedLeft > 0 && speedLeft <= OFFSET)    speedLeft = -1;
    FrontLED(OFF);
    BackLED(ON,ON);
    SerWrite("\tback",5);
}

void IRLeft (void)//小车左转
{
    speedLeft    -= STEP;
    if (speedLeft > 0 && speedLeft <= OFFSET) speedLeft = -1;
    speedRight += STEP;
    if (speedRight < 0 && speedRight >= -OFFSET) speedRight = 1;
    FrontLED(OFF);
    BackLED(ON,OFF);
    SerWrite("\ttrun left",9);
}

void IRRight (void)//小车右转
{
    speedLeft    += STEP;
    if (speedLeft < 0 && speedLeft >= -OFFSET) speedLeft = 1;
    speedRight -= STEP;
    if (speedRight > 0 && speedRight <= OFFSET) speedRight = -1;
    FrontLED(OFF);
    BackLED(OFF,ON);
    SerWrite("\ttrun right",9);
}


void IRStop(void)//小车停止
{
    speedRight = speedLeft = 0;
    FrontLED(OFF);
    BackLED(OFF,OFF);
    SerWrite("\tstop",5);
}

void Speed(void)
{
unsigned char leftDir = FWD, rightDir = FWD;
if (speedLeft > 0 && speedLeft <    OFFSET) speedLeft += OFFSET;
                if (speedLeft < 0 && speedLeft > -OFFSET) speedLeft -= OFFSET;
                if (speedRight > 0 && speedRight <    OFFSET) speedRight += OFFSET;
                if (speedRight < 0 && speedRight > -OFFSET) speedRight -= OFFSET;

                leftDir = rightDir = FWD;
                if (speedLeft < 0)    leftDir = RWD;
                if (speedRight < 0) rightDir = RWD;

                if (speedLeft >     255) speedLeft    =    255;
                if (speedLeft <    -255) speedLeft    = -255;
```

```
                if (speedRight >    255) speedRight =    255;
                if (speedRight < -255) speedRight = -255;

                MotorDir(leftDir,rightDir);
                MotorSpeed(abs(speedLeft),abs(speedRight));
}

int main(void)
{
    unsigned char cmd;
    unsigned int i;
    unsigned char sw;

    Init();
    SerWrite("\n\rHello I love you really¡\\n",30);

    while(1)
    {
        for (i = 0; i < 0xFE; i++)
            {
            SerRead(&cmd,1,0xFFFE);

            switch (cmd) {
                case DIARWD :     STEP=40;IRRwd(); break;
                case DIAFWD :     STEP=40;IRFwd(); break;
                case DIALEFT:     STEP=40;IRLeft(); break;
                case DIARIGHT:    STEP=40;IRRight(); break;
                case DIASTOP :    STEP=40;IRStop(); break;
            }
            Speed();
        }

        for (i = 0; i < 0xFE; i++)
        {
            sw = PollSwitch();
            if (sw == 0x01) {STEP=20;    IRLeft();Speed();IRFwd();Speed();}
            if (sw == 0x02) {STEP=20;    IRStop();Speed();IRLeft();Speed();IRFwd();Speed();}
            if (sw == 0x04) {STEP=20;    IRStop();Speed();IRLeft();Speed();IRFwd();Speed();}
            if (sw == 0x08) {STEP=20;    IRStop();Speed();IRLeft();Speed();IRFwd();Speed();}
            if (sw == 0x10) {STEP=20;    IRStop();Speed();IRLeft();Speed();IRFwd();Speed();}
            if (sw == 0x20) {STEP=20;    IRRight();Speed();IRFwd();Speed();}

        }
    }
}
```


>>>更多的源码下载地址：

1，ftp://asuro:asuro@218.88.186.17

2，http://www.61mcu.com/bbs/dispbbs.asp?boardid=11&Id=50

3， http://home.planet.nl/~winko001/Asuro/Software/AsuSfwPagFrm.htm