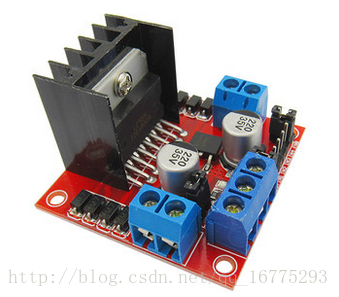
# 材料准备

## 小车套件：



## 驱动模块：

（在此我选用比较经典的驱动模块L298N，该驱动可以满足我们的需求，也可以把电池的高电压转换为我们需要的5V,3.3V电压）



电池：在此我选用的电池型号是18650

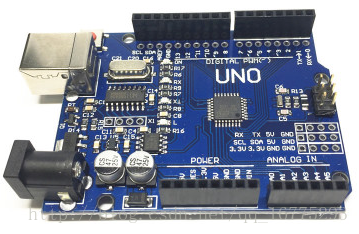


电池座：电池座一定要选取与电池型号相匹配



## Arduino板子

在此我使用的是Arduino UNO，其他版本的Arduino也可以参照该教程



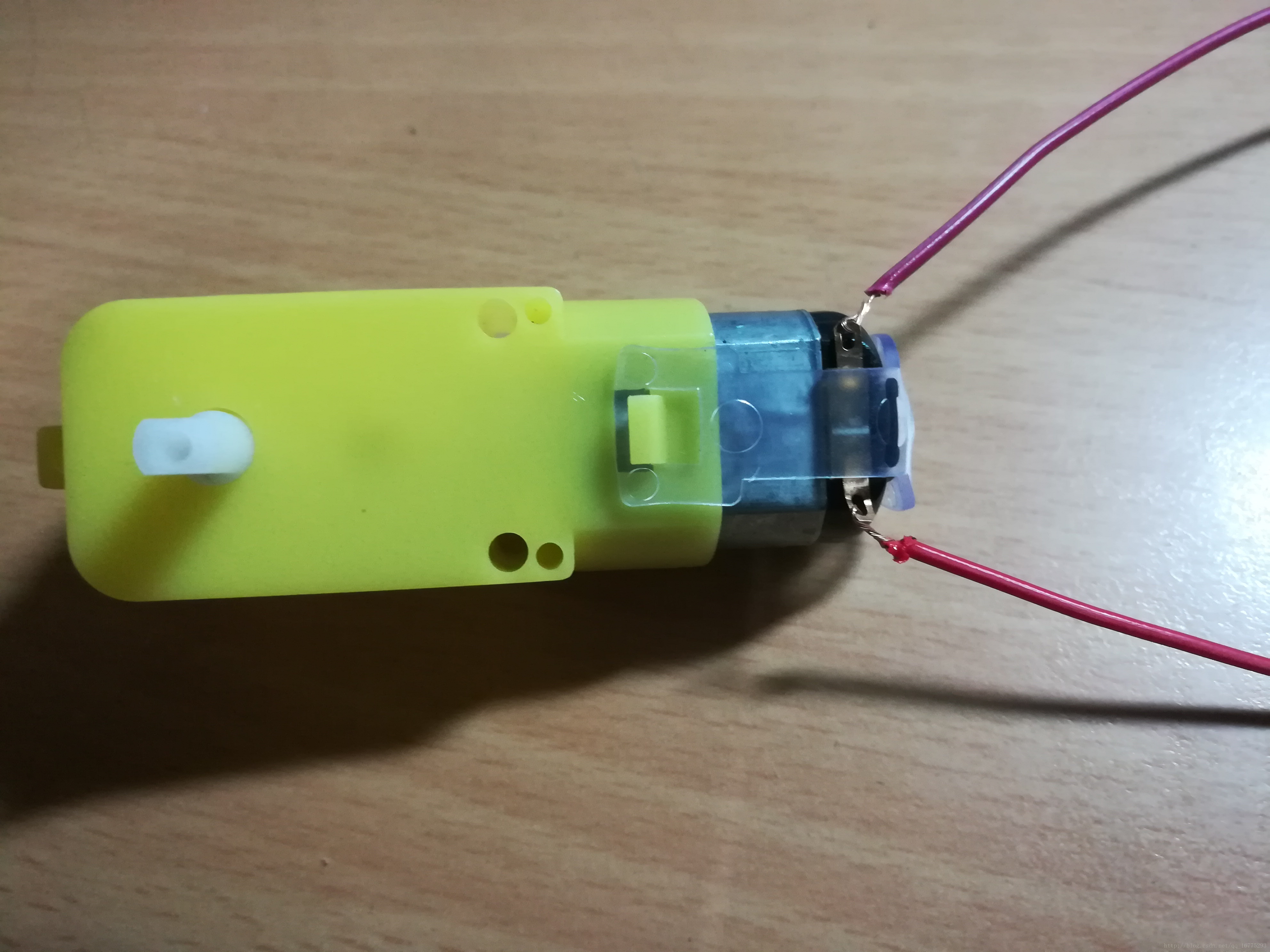
做项目开发肯定少不了电烙铁，焊锡，电源线，杜邦线，排针，排插这些基本工具。

至此，所需的基本材料已经准备完毕，接下来就可以进行小车的拼装了。

# 拼装篇

## 电机线的焊接

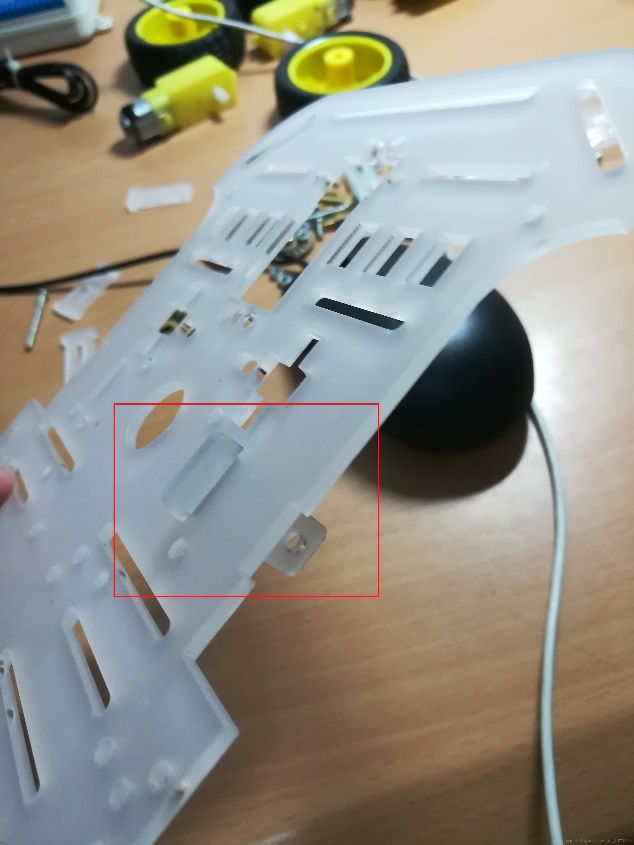
套件中的电机一般都是没有连线的，我们需要自己动手把准备好的**电源线**固定在电机上，一般不建议直接把电源线绑在电机上，这样会经常导致小车的电机接触不良，因此我建议大家把电源线直接焊在电机上。



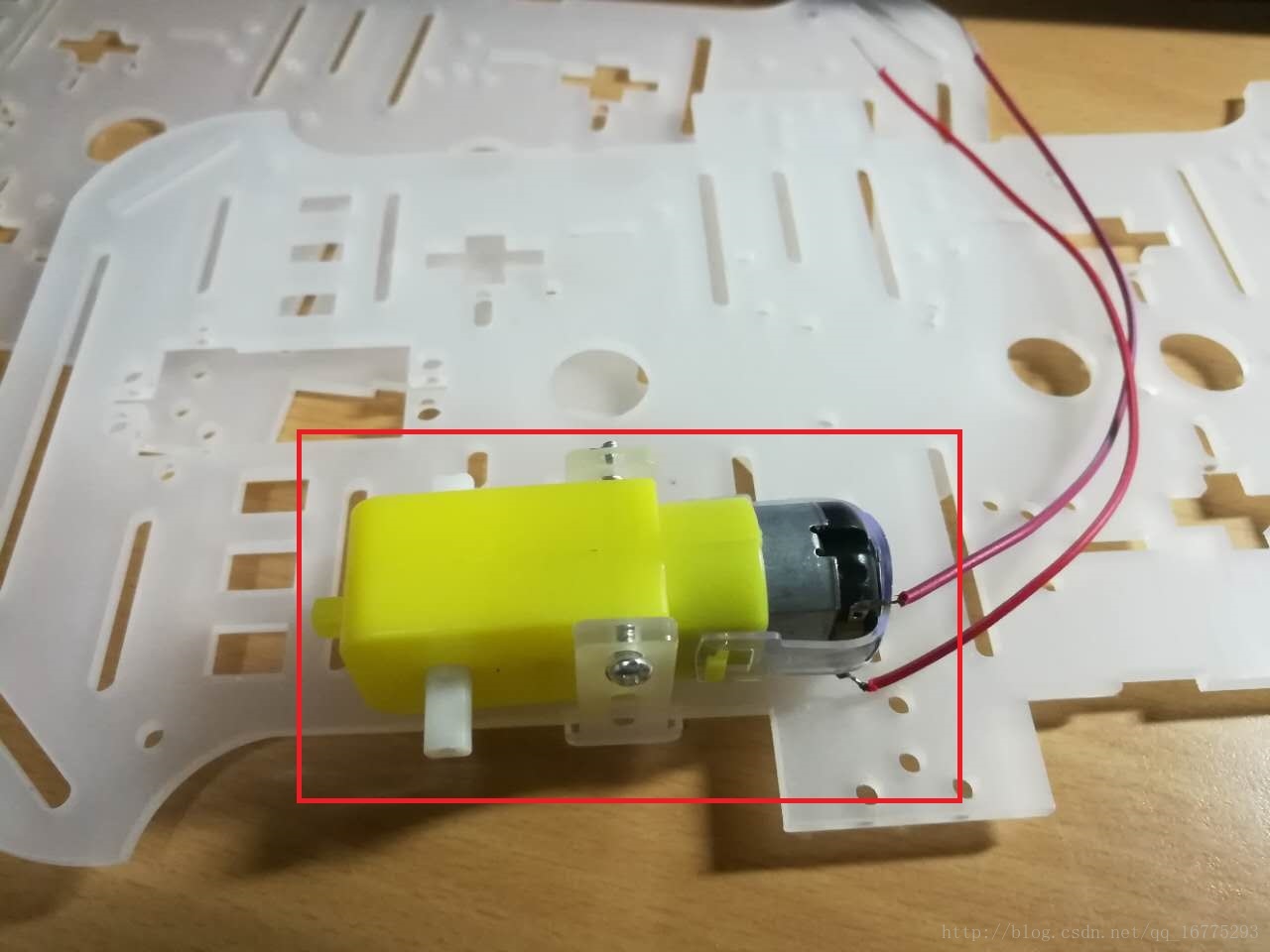
## 电机固定



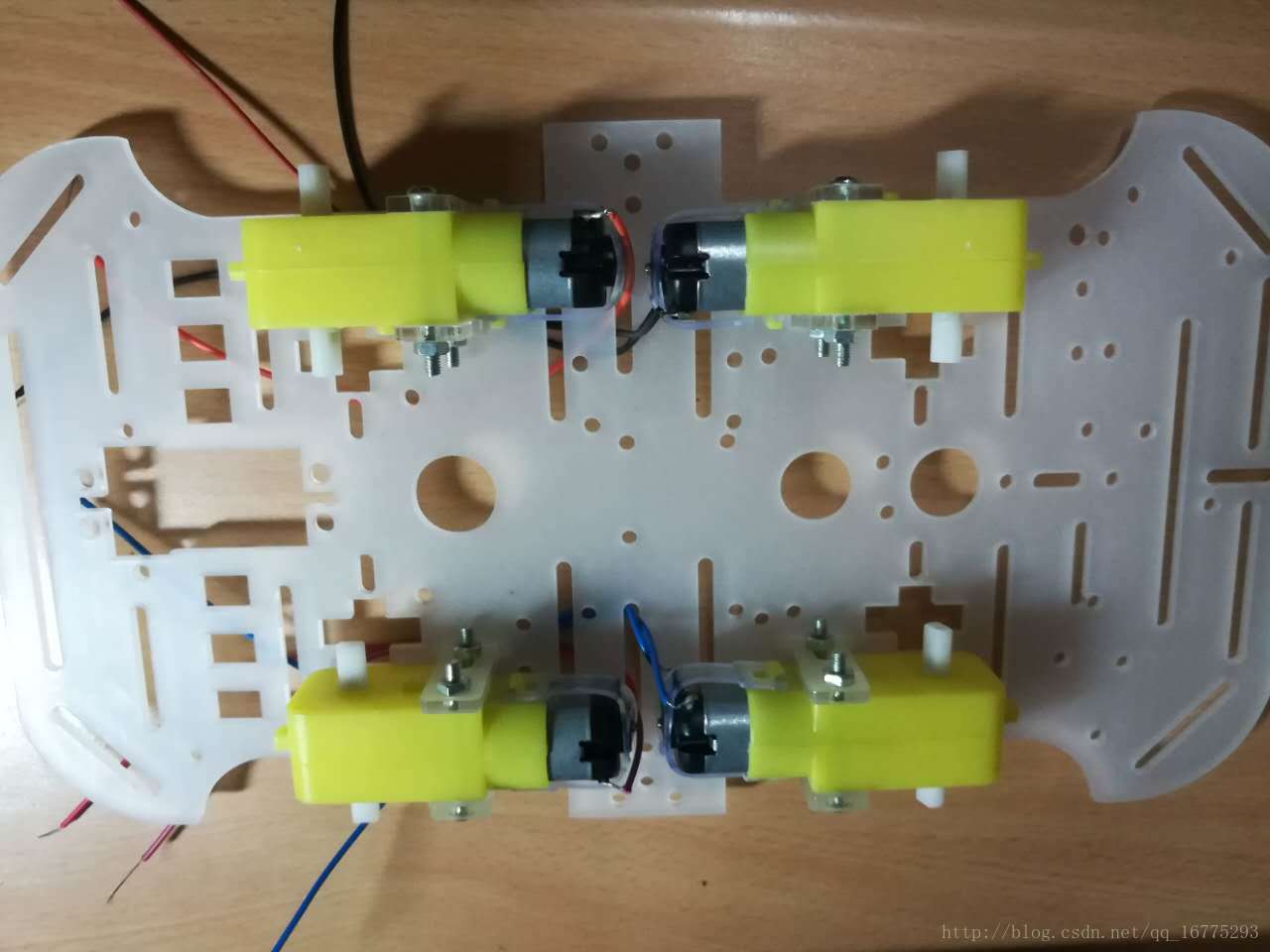
将上图所示的电机固定架插入小车底盘底板中，一般小车有上下两个底板，任意拿一块即可当做底板。



再将焊接好的电机配套另外一个固定架固定在底板上。

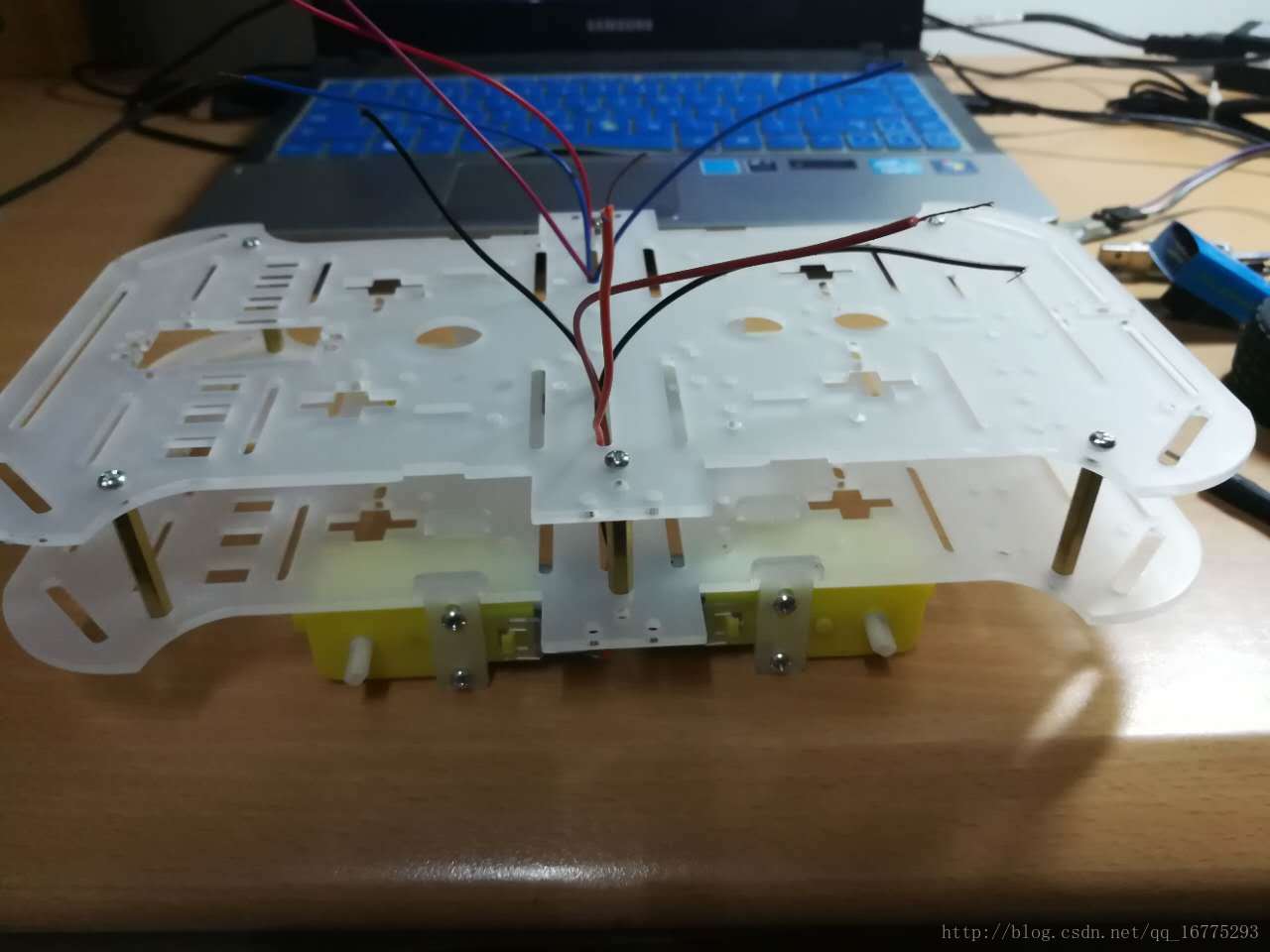


同样固定其他三个电机

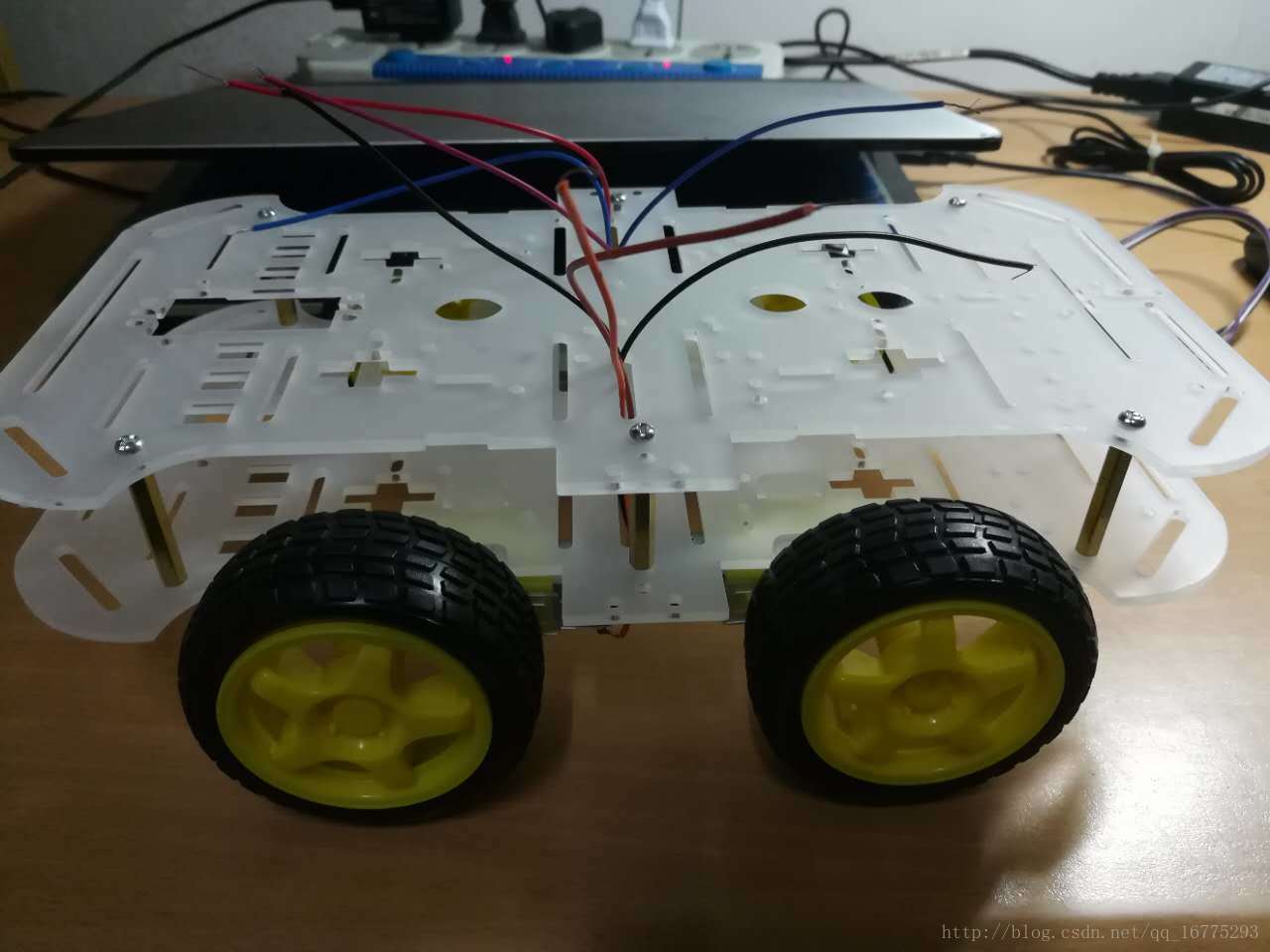


## 安装背板支柱

一般套件都会提供六根铜柱，我们将其固定在小车的周围。



## 安装轮子



# 驱动模块安装

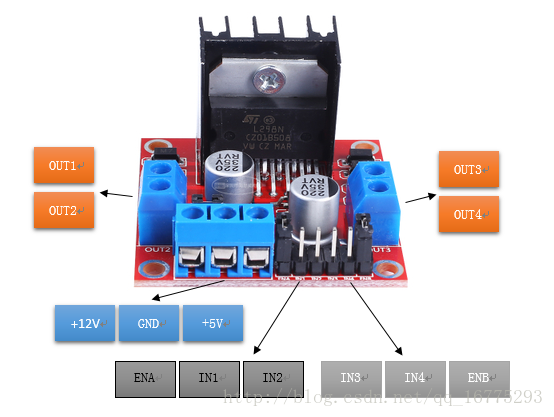
驱动是干嘛的，为什么要驱动，小时候玩四驱车的时候直接装上电池小车就跑了，干嘛还要驱动模块。

答案很简单，四驱车他只能朝着一个方向运动，而且**永远都是以最大速度运行**，我们所做的智**能小车通常要控制小车电机的转速和运行方向**，因此驱动是必不可少的模块。驱动模块的具体工作原理不在这里做详细的介绍，想了解的朋友可以自行查阅资料。

## 驱动模块用法简介

一般拿到一个模块之后都要去对应的官网找到它的资料包，查看其详细用法，在某宝上买的模块一般店家都有整理好的资料包，所以某宝也是一个很好的资料库，大家一定要合理运用哦~

在此我们选用的是L298N模块，该模块引脚分配如下：

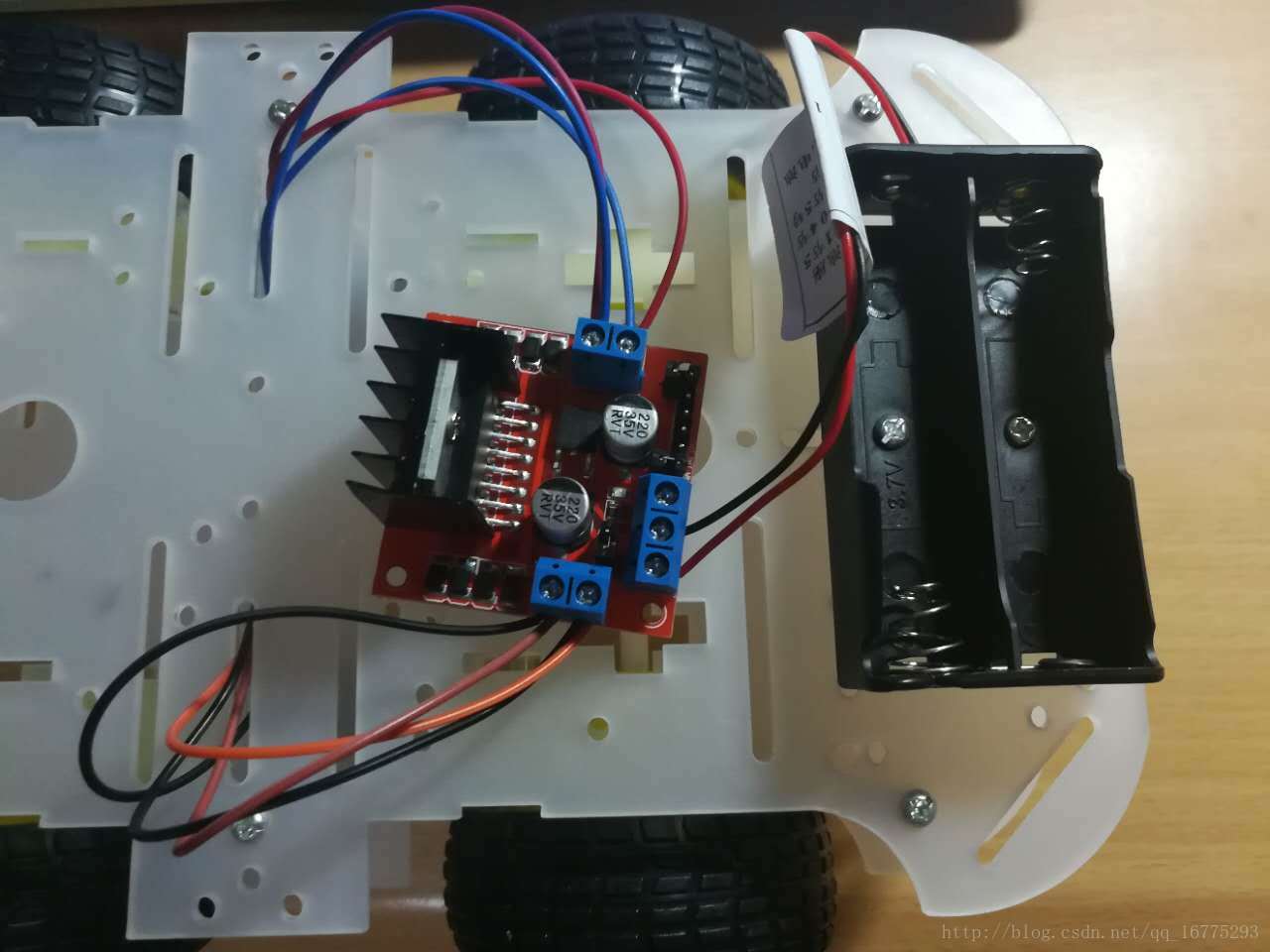


* +12V：该引脚接的电压是驱动模块所能输出给电机的最大电压，一般 直接接电池。12V是由L298N芯片所能接受最大电压而定，一般介入5~12V电压。在此我们接入的电压为两节18650串联的电压，即3.7+3.7=7.4V；
* **GND：** 在该项目中GND即为电源的负极，同时要保证Arduino开发板，驱动模块等所有模块的GND连在一起才可以正常工作。在某些复杂的项目中还需要区分数字地和模拟地，在此不做详细介绍。
* **+5V：** L298N模块（注意不是L298N芯片）内含稳压电路（将高电压转换为低电压的电路），在模块内部将"+12V"引脚输入的电压转化为可供开发板使用的+5V电压，一般将次输出接入到开发板为开发板供电。
* L298N有两路输出，所以可以控制小车前进、后退、转弯，其中：  
  **ENA：** 代表第一路输出的电压大小。驱动模块输出电压越高，电机转速越快。  
  1.当其输入为0V的时候，驱动模块输出对第一路电机输出电压为0V；  
  2.当其输入为3.3V的时候，驱动模块对第一路电机输出电压为"+12V"引脚的输入电压。  
  3.由于ENA输入电压的高低控制驱动对电机的输出电压，因此当我们需要对小车运动速度进行控制的时候，一般通过PWM对"ENA"引脚进行控制。
* IN1/IN2：这两个引脚控制电机正反转方向。例:假如IN1输入高电平3.3V，IN2输入低电平0V，ENA为3.3V，电机正转，此时将IN1输入改为0V，IN2输入改为3.3V，其他条件不变，则电机将会反转。
* OUT1/OUT2：这两个引脚分别接电机的两极。
* ENB，IN3/IN4，OUT3/OUT4**引脚控制第二路输出，与上述**ENB，IN3/IN4，OUT3/OUT4功能相似。

## 驱动安装

L298N电源端在前，从左往右电机的输出端分别为out1、out2、out3、out4.

* 左边第一个轮的下边线接out1上边线接out2;
* 左边第二个轮的下边线接out2，上边线接out1;
* 右边第一个轮的下边线接out3，上边线接out4;
* 右边第二个轮的下边线接out4，上边线接out3;
* 并用螺丝刀将拧蓝色接线柱上方的螺丝拧紧。



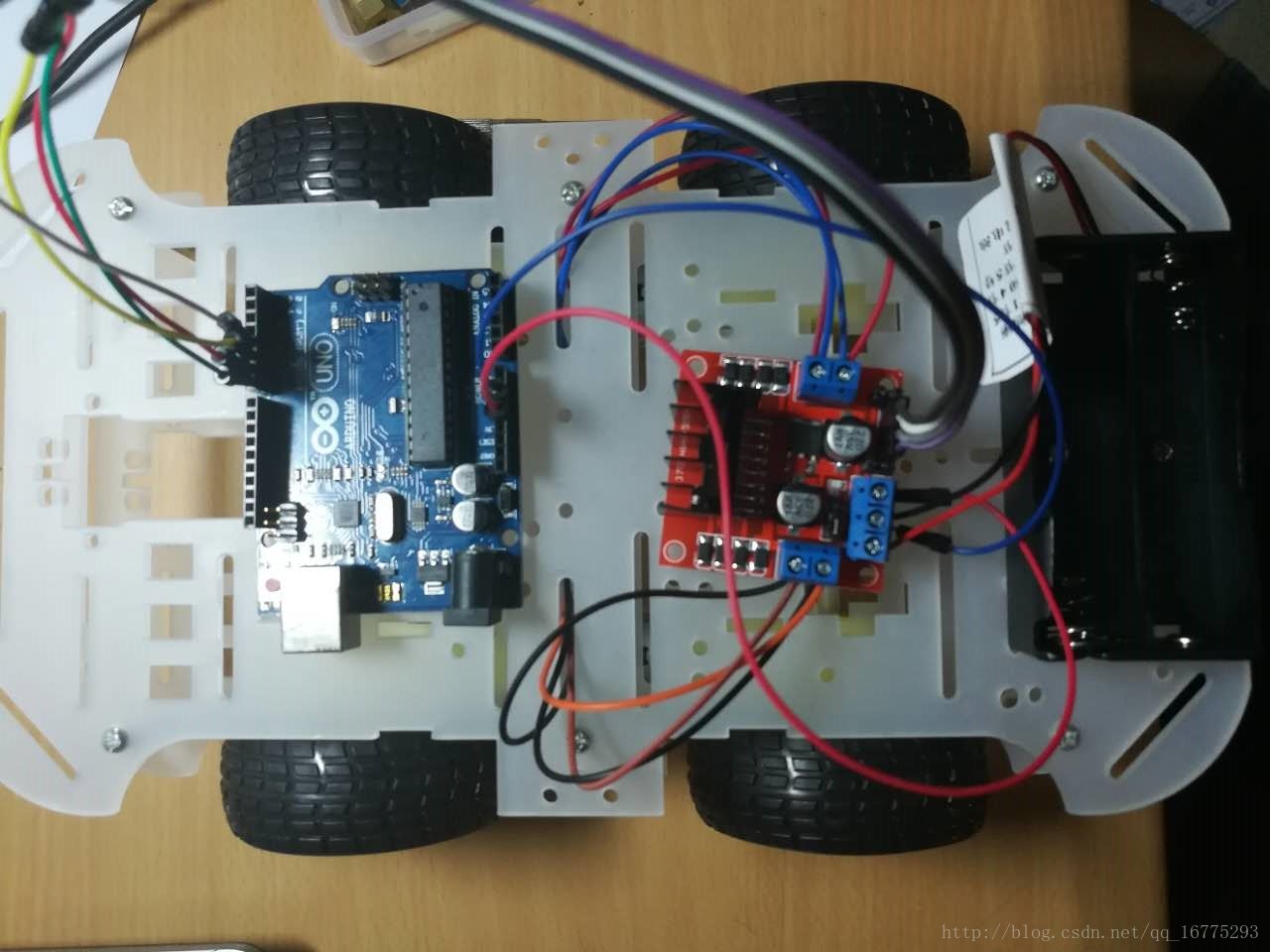
# 电池座固定

将电池固定在小车尾部，将电源的两根线链接到+12V和GND引脚，红色代表正极，接到+12V，黑色代表负极，接到GND（一**般电路中默认红色为正，黑色为负**），并拧紧螺丝固定。

将Arduno板子用螺丝固定在小车中部，由于小车运动中也需要对开发板供电。

1. 此时我们用两根公对公的杜邦线为其供电，红色(也可以为其他颜色)杜邦线一边插入Arduino板的"5V"引脚，一边插到L298N驱动的"+5V"引脚；
2. 黑色(也可以为其他颜色)杜邦线一边插入Arduino板的"GND"引脚，一边插到L298N驱动的"GND"引脚。

为控制电机的正反转，此时我们需要拿四根公对母杜邦线，将L298N驱动的IN1/IN2/IN3/IN3引脚与Arduino板的4/5/6/7号引脚对应相连，最终拼装图如下：



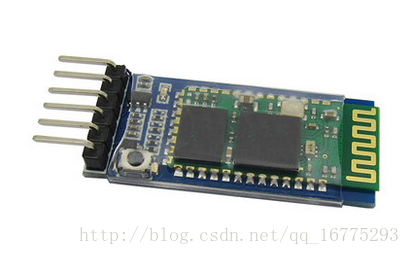
可以用扎带或者皮筋将其捆好固定

至此，我们的小车基本拼装完成，接下来就要开始写程序来控制小车运动咯，有没有很激动~

# 蓝牙模块

蓝牙（ Bluetooth® ）：是一种无线技术标准，可实现固定设备、移动设备和楼宇个人域网之间的短距离数据交换（使用2.4—2.485GHz的ISM波段的UHF无线电波）。

我们在此使用的蓝牙模块(HC-05)已经在内部实现了蓝牙协议，不用我们再去自己开发调试协议。这类模块一般都是借助于串口协议通信，因此我们只需借助串口将我们需要发送的数据发送给蓝牙模块，蓝牙模块会自动将数据通过蓝牙协议发送给配对好的蓝牙设备。





TX: 接Arduino UNO开发板"RX"引脚

RX: 接Arduino UNO开发板"TX"引脚

GND: 接Arduino UNO开发板"GND"引脚

VCC: 接Arduino UNO开发板"5V"或"3.3V"引脚

# Ardunio程序编写

Arduino的程序编写一般使用Arduino IDE，该软件安装比较简单，大家可以自行安装，安装成功后打开IDE，在程序里写入下述代码：

|  |
| --- |
| #include <Servo.h>  //定义五中运动状态  #define STOP 0  #define FORWARD 1  #define BACKWARD 2  #define TURNLEFT 3  #define TURNRIGHT 4  //定义需要用到的引脚  int leftMotor1 = 4;  int leftMotor2 = 5;  int rightMotor1 = 6;  int rightMotor2 = 7;  void setup() {  // put your setup code here, to run once:  //设置控制电机的引脚为输出状态  pinMode(leftMotor1, OUTPUT);  pinMode(leftMotor2, OUTPUT);  pinMode(rightMotor1, OUTPUT);  pinMode(rightMotor2, OUTPUT);  }  void loop() {  // put your main code here, to run repeatedly:  int cmd;  for(cmd=0;cmd<5;cmd++)//依次执行向前、向后、向左、想有、停止四个运动状态  {  motorRun(cmd);  delay(2000);//每个命令执行2s  }  }  //运动控制函数  void motorRun(int cmd)  {  switch(cmd){  case FORWARD:  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, HIGH);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, HIGH);  break;  case BACKWARD:  digitalWrite(leftMotor1, HIGH);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, HIGH);  digitalWrite(rightMotor2, LOW);  break;  case TURNLEFT:  digitalWrite(leftMotor1, HIGH);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, HIGH);  break;  case TURNRIGHT:  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, HIGH);  digitalWrite(rightMotor1, HIGH);  digitalWrite(rightMotor2, LOW);  break;  default:  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, LOW);  }  } |

## 代码详解

为方便代码的编写，提高代码的可读性，在此我们先定义出小车可能的运动状态

|  |
| --- |
| #define STOP 0 //停止  #define FORWARD 1 //前进  #define BACKWARD 2 //后退  #define TURNLEFT 3 //左转弯  #define TURNRIGHT 4 //右转弯 |

电机运动需要经过驱动模块驱动，而驱动模块的输出状态又取决去IN1/IN2/EN，IN3/IN4/ENB这两组引脚的控制。本实验只是简单控制电机的运动，不用控制电机的转速，因此ENA,ENB默认接入高电平（买过来模块的时候，你会发现这两个引脚都通过跳线帽和"+5V"的引脚相连，即输出最大电压），此时我们只需控制IN1/IN2，IN3/IN4两组引脚即可对小车的运动状态进行控制。因此我们在此定义以下四个引脚

|  |
| --- |
| //定义需要用到的引脚  int leftMotor1 = 4;  int leftMotor2 = 5;  int rightMotor1 = 6;  int rightMotor2 = 7; |

当电机的两个输入端加入的电压有电压差，且电压差满足一定条件时电机才会转动，为控制L298N驱动OUT1/OUT2，OUT3/OUT4两路输出，我们需要了解该模块的使用方法。

**下面两个表格为L298N的输入输出对应关系，其中H：高电平，L：低电平，ENA、ENB均为高电平**

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | | 输出 | |
| IN1 | IN2 | OUT1 | OUT1 |
| H | L | H | L |
| L | H | L | H |

|  |  |  |
| --- | --- | --- |
| 输入 | 输出 | |
| IN3 | IN4 | OUT3 | OUT4 |
| H | L | H | L |
| L | H | L | H |

由上述表格可以清晰看出控制小车运动时，只需要将同一边的两个引脚设置成不同的输出电压即可,例如让小车向前运动时左右两边的IN1和IN2可以设置为

|  |
| --- |
| digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, HIGH);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, HIGH); |

为了提高代码的执行效率，我们在此将小车的四种运动状态封装在函数里，方便调用。

|  |
| --- |
| void motorRun(int cmd)  {  switch(cmd){  case FORWARD:  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, HIGH);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, HIGH);  break;  case BACKWARD:  digitalWrite(leftMotor1, HIGH);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, HIGH);  digitalWrite(rightMotor2, LOW);  break;  case TURNLEFT:  digitalWrite(leftMotor1, HIGH);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, HIGH);  break;  case TURNRIGHT:  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, HIGH);  digitalWrite(rightMotor1, HIGH);  digitalWrite(rightMotor2, LOW);  break;  default:  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, LOW);  }  } |

至此，小车应该便可以成功运动起来咯，快为自己庆祝下吧。接下来我们将在此基础上进项进一步的开发，大家再接再厉哦

##调试中可能出现的问题

问题1.

下载完程序，小车向前运动时，有些轮子向前，有些轮子运动方向相反：

将运动方向错误的电机的两根电源线在驱动上的接线位置对换。例如将本来接OUT1的电源线换到OUT2，将接OUT2的电源线换到OUT1。

问题2.

代码没变，但是电机一会儿动一会不动：

这样的情况一般是由于接触不良造成的，检查你的驱动输出到电机的电源线是否有松动或者接触不良的情况。

问题3.

向前运动时他向后，向后运动时他向前，向左运动时他向右：

解决方案1：按照 问题1 中那样的解决方案，一个一个换线

解决方案2：将void motorRun(int cmd)函数中，引脚输出状态反过来，例如解决向前运动时他向后这种错误状态

|  |
| --- |
| case FORWARD: //原来的代码  digitalWrite(leftMotor1, LOW);  digitalWrite(leftMotor2, HIGH);  digitalWrite(rightMotor1, LOW);  digitalWrite(rightMotor2, HIGH);  break; |

|  |
| --- |
| case FORWARD: //修改后的代码  digitalWrite(leftMotor1, HIGH);  digitalWrite(leftMotor2, LOW);  digitalWrite(rightMotor1, HIGH);  digitalWrite(rightMotor2, LOW);  break; |