

# FIRAHandler Official Documentation :: <Gmotor.h> & <Grobot.h>

Author VisualDust , 201907092259LastUpdate

## 《FIRAHandlerG库从入门到放弃全教程》

- 以下是入门部分

### <Gmotor.h>的参数说明

```
int cnt1,cnt2; //链接L298N驱动芯片的两个pwm控制端口
int speed; //电机的记忆速度
int solid = 0; //电机在停转时是否有功率
double prop = 1.0; //电机速度比例矫正, 当出现转速过慢或两个电机转速不一时进行调整
int direc = 0; //用来记忆电机目前的正反转情况
```

### <Gmotor.h>方法说明

```
Gmotor(int cnt1, int cnt2); //初始化方法, 传入参数为链接L298N驱动芯片的两个pwm控制端口的端口号
void stop(int mode); //即刻生效的电机停止方法, 传入参数停止模式, 0为立即停止, 非零为渐停
void setSolid(int solid); //用以更改solid参数, 传入一个新值作为solid值
void setSpeed(int speed); //设置电机速度并立刻生效
int getSpeed(); //返回一个整数值, 为电机当前功率
void setProp(double prop); //设置电机矫正值
void exchangePin(); //一个内部方法, 用以交换两个控制端口。由于调试可见性为public
void gradientTo(int speed, int ms); //使电机转速从当前值耗时传入参数ms毫秒渐变到传入参数speed
void walk(int speed, int ms); //使该电机以传入参数speed的速度旋转传入参数ms毫秒
```

### <Gmotor.h>的使用样例

```
Gmotor* myMotor = new Gmotor(3, 4); //使用3、4号口建立一个电机控制。注意, 传入参数顺序3,4和4,3不一样, 两者电机旋转方向不同, 即当你发现写3,4转的不对, 可以改成4,3
myMotor->exchangePin(); //如果开机后我发现这个电机旋转方向不是我想要的, 调用这个方法更改方向
myMotor->setSpeed(120); //将电机速度设置为120, 立即生效, 电机开始疯狂旋转
myMotor->setProp(0.8); //经历一波调试后我发现电机总体转速太高了, 将prop参数设置为0.8, 这时如果再setSpeed(120)实际速度会是120*0.8=96, 记住, setProp方法不能传入大于1的数, 如果传入大于1的数会按照1处理
myMotor->walk(120,1000); //我的这个电机以120的速度走了1秒
```

```
myMotor->gradientTo(50, 1000); //我的这个电机在1秒内从转速120渐变成了转速50
myMotor->walk(120, 1000); //我的这个电机以50的速度走了1秒
```

什么？你说你的车车上有两个电机，一下控制俩太麻烦？好的没问题。接下来我们看一下<Grobot.h>这个新工具。当然，很明显，从字面意思上理解，Gmotor.h一看就是用来控制电机的，那么Grobot.h当然就是用来控制机器人的。

## <Grobot.h>的参数说明

```
//这部分参数可见性为全局可见
extern uint8_t SmallFont[]; //声明小字体矩阵，用于TFT显示屏
extern uint8_t BigFont[]; //声明大字体矩阵，用于TFT显示屏
extern uint8_t SevenSegNumFont[]; //声明特大数字字体矩阵，用于TFT显示屏
byte          vibrate = 0; //用于控制遥控手柄震动

//下面的参数可见性为private
Gmotor *mtl, *mtr; //这个机器人包含两个电机
uc      s1, s2, s3, s4, s5; //五个链接光电传感器的analog端口
UTFT * GLCD      = new UTFT(QD_TFT180A, 51, 52, 32, 34, 33); //默认初始化一个TFT显示屏
PS2X * controller = new PS2X(); //默认初始化一个遥控手柄接收器
int    threshold[6]; //用来记忆几个光电传感器的阈值
int    lineCrossTime = 70; //大致穿线一次的时间，单位ms
int    negativeSpeed = 100, negativeRuntime = 50; //急停反转功率和反转时间
```

## <Gmotor.h>的方法说明

```
Grobot(); //默认初始化方法，其实里面什么也没有
void configMotor(Gmotor *leftMotor, Gmotor *rightMotor); //为你的机器人连接两个电机
void configSensor(uc s1, uc s2, uc s3, uc s4, uc s5); //为你的机器人设置5个传感器
void configTFT(byte pos); //为你的机器人定义一个显示屏，使用默认端口初始化并规定显示方向pos
void configTFT(byte model, int RS, int WR, int CS, int RST, byte pos, int SER = 0); //为你的机器人定义一个显示屏，使用自定义端口初始化并规定显示方向pos
byte configController(uc clk, uc cmd, uc att, uc dat); //为你的机器人初始化一个遥控器接收器，该方法返回一个byte，0表示已连接，1表示未连接
void configController(uc clk, uc cmd, uc att, uc dat, bool pressures, bool rumble); //为你的机器人初始化一个遥控器接收器，并设置是否允许模拟压感和震动
void configServo(int armPin, int handPin); //用这个方法为你的机器人添加两个舵机以控制机械手，armPin为链接机械臂控制舵机的端口号，handPin为链接机械手控制舵机的端口号。
void testSensors(); //以引导的形式测量场地传感器值并自行计算阈值。该方法将使用TFT显示屏辅助矫正，所以在这之前必须已经完成了显示器配置
void setThreshold(int s1, int s2, int s3, int s4, int s5); //这个方法为5个传感器设置阈值
```

```

void setThreshold(int sensorNum, int val); //这个方法为端口号为sensorNum的
传感器设置阈值val
int getThreshold(int sensorNum); //这个方法用来获得记忆中端口号为sensorNum
的传感器阈值，返回一个整数值。
void setLineCrossTime(int); //设置大致穿线时间
void setHunterSensor(int, int, int); //设置巡线用到哪三个传感器

int getSensorVal(uc pin); //用来获得链接于端口pin的传感器当前值
Gmotor *getMotor(uc motor); //用来得到一个电机指针，传入参数'l'或'r'获得左边
或右边电机的返回指针

void walkTime(int speed, int ms); //让你的机器人以speed的电机转速往前走ms毫
秒，注意，这与Gmotor中的walk不同，Grobot中的walk会使两个电机同时转动
void walkBlock(int speed, int lineCnt); //以speed的速度在场地上直行lineCnt
个格
void turnLeft(int speed, int mode); //机器人以speed的电机转速左转一格
void turnRight(int speed, int mode); //机器人以speed的电机转速右转一格
void huntLine(int baseSpeed, int subSpeed, int lineCnt); //机器人以speed
的电机转速左转一格
void stop(int mode); //以mode模式停止，0为自然停止，1为瞬间停止

void waitForButtonPress(ui BUTTON); //等待遥控手柄上的BUTTON键被按下
void waitForButtonRelease(ui BUTTON); //等待遥控手柄上的BUTTON键被松开

void initialRobot(); //用这个方法初始化机器人，否则机器人无法正常运行
void pair(); //用这个方法使遥控器配对
void enterManualMode(); //调用这个方法进入手动遥控模式

```

---

看完入门部分有什么感想？是不是完全没看懂？那就对了。我自己也看不懂。接下来我们进入放弃部分（没错这才是有用的部分上面是用来撑字数的）

---

## FBI Warning：前方核能

---

- 以下是放弃部分

### Getting Start 入门

来试试看在脑子里想象一个机器人首先我们需要一个便于我们想象一个机器人的环境。所以....

```
#include "Grobot.h"
```

好了，包含这个头文件能够构造时空断裂（我瞎扯的）以便于我们的脑海中生成一台机器人，机器人长什么样子来着.....那么我们新建一个机器人...

```
Grobot *robot = new Grobot();
```

好了我们创见了一个机器人。如你所见，robot的意思就是机器人，怎么样是不是很简单？就像是你新建一个int变量时写的[ int i = 0 ]一样。

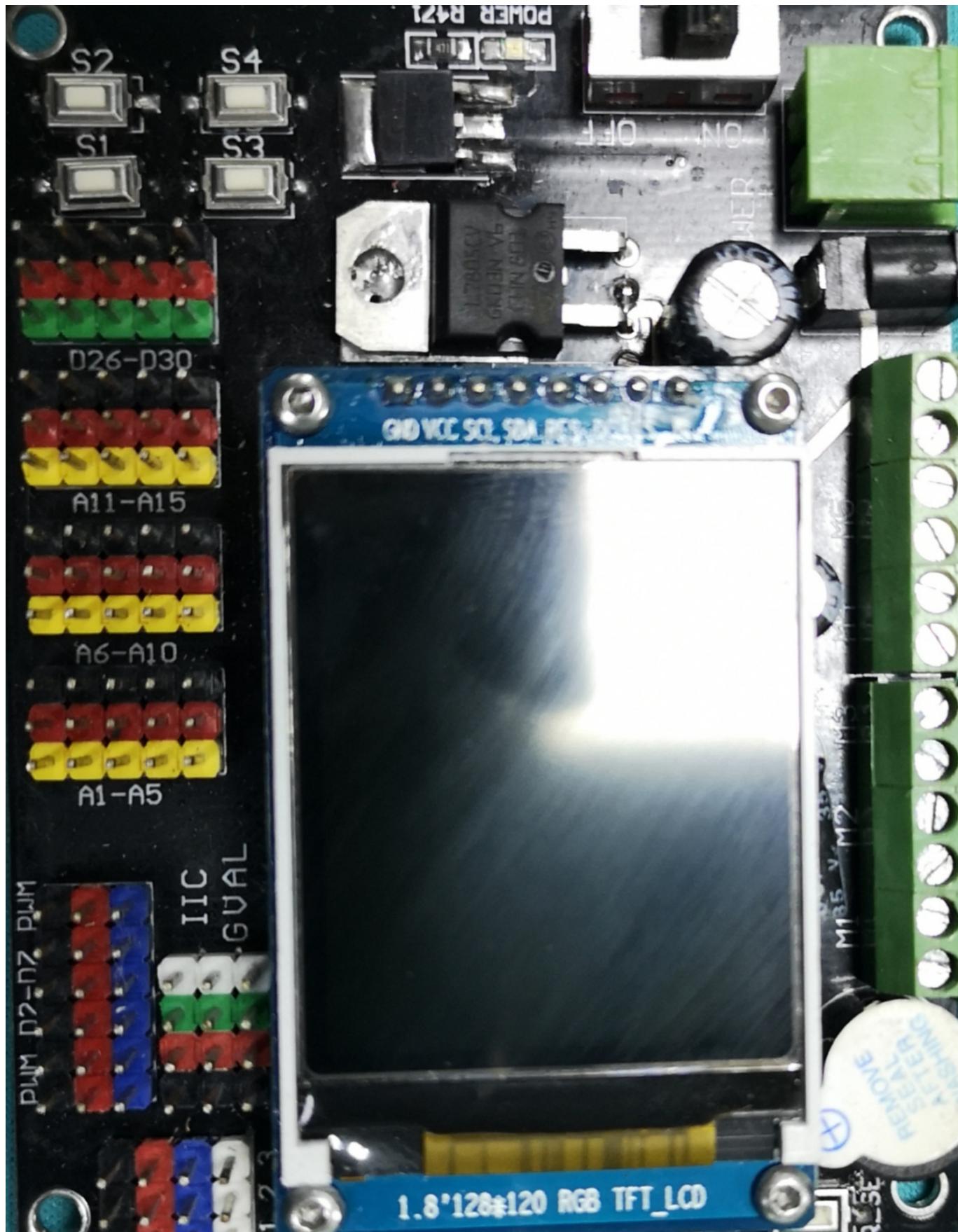
emmmmmm我想想好像有两个轮子来着，所以就有控制这两个轮子的电机。所以我们再想象两个电机...

```
Gmotor *lmt = new Gmotor(3, 2), *rmt = new Gmotor(5, 4);
```

lmt啥玩意啊你逗我呢？！

别别别急啊Left(左)MoTor(电机)缩写不就变成lmt了吗，谁没有个手懒的时候用个缩写咋了你有意见啊？同理RightMotor(右电机)的缩写也就是rmt。由于左电机是由3, 2号PWM口控制的，右电机是由5, 4号PWM口控制的，你至少告诉它自己是那个口控制的吧？奥对忘告诉你了...

我们先看一下FIRA官方提供的开发板长什么样子...



对对对！没错就是这个！刚才我们提到控制电机的pwm口...emmm...你看左下方隐约写着PWM D2-D7 PWM字样的那一排，蓝色的就是PWM接口了。因为过一会我们还要往机器人身上安装传感器，而传感器用的是analog口，也就是左侧好几排黄色的接口---所以在这里我们把它们加以区分：

蓝色的这个pwm口可以输出pwm信号，所以能向电机驱动模块传递转速信号。  
黄色的analog接口可以读取pwm信号，所以能够读取传感器传来的信息。

总之，蓝色的能输出，黄色的能读入。嘿嘿，想想看如果把这两个接在一起.....啊呸我在想些什么！

回归正题，我们说到....奥对，我们新建了两个电机！光新建了不行啊我们还得把它们安装到你的机器人上！void setup你应该知道吧，那么试试看在setup里写下这句话。什么你不知道什么是setup？

---

啊....好吧，当你新建一个arduino空程序时，会自动生成这两个方法(method,也叫函数,function)：

```
void setup(){}  
void loop(){}  
  
这两个方法从字面意思理解,setup就是"建立"的意思。当你的arduino开机时，"会首先运行初始化代码"，也就是会首先运行setup里面的内容。  
而loop字面上是循环的意思。当arduino执行完setup之后，会进入loop，并循环运行里面的代码。唯一的跳出条件是return。所以，我们进行电机配置、显示屏配置等工作都需要在"初始化"也就是setup的时候做。
```

---

好了我们回到正题。刚才说怎么把电机链接到你的机器人上着：

```
robot->configMotor(lmt, rmt);  
  
config意思是配置，motor意思是电机。(可别写错了大小写昂，Motor的M要大写。以后我们写什么都要注意大小写别弄错了...)连起来就是配置电机的意思。也就是，我们把刚才创建的左电机lmt和右电机rmt配置到了机器人上。至于中间那个箭头(->)....那个是指针...呸，别想指针，你可以理解成机器人接下来要做一件事的时候用一个箭头为它指明方向！比如说"机器人->吃屎..." 呔我什么也没说。
```

好的...我再看看...你看这个板子上是不是有个显示屏啊，我们得把这个也安装上，才显得我们足够狂飙炫酷吊炸天。

```
robot->configTFT(QD_TFT180A, 51, 52, 32, 34, 0, 33);  
  
还记得config是什么意思吧，对是配置的意思。至于这个TFT，是显示器的一种。configTFT就是给机器人搞一个TFT显示屏。这个显示屏的具体型号是QD_TFT180A，我给你填好了你记住就行。
```

还记得config是什么意思吧，对是配置的意思。至于这个TFT，是显示器的一种。configTFT就是给机器人搞一个TFT显示屏。这个显示屏的具体型号是QD\_TFT180A，我给你填好了你记住就行。

那么接下来是遥控器的接收器。我们知道，这个比赛项目有自动和手动两个部分。手动部分当然是要遥控的，所以需要一个遥控器和一个遥控器的接收器。我们试着在setup里为你的机器人配置一个接收器：

```
robot->configController(A14, A7, A13, A6, true, true);
```

再复习一次config什么意思来着？什么你忘了？来这位同学出去站着去。

咳咳剩下的同学我们继续。不想出去陪他站着你们就好好背单词。是的Controller是控制器的意思。词根是control(控制)，加个er表示控制器。所以configController连一起就是"配置控制器"的意思。至于里面填了啥，你也不用管，都给你填好了复制粘贴就行。想了解参数含义的可以进PS2XLIB库里面暗中观察一下。

搞完了控制器我们来看看各个传感器。你的机器人上应该有5个传感器用来巡线以及进行各种判定，我们习惯上直接插在analog口的，从上往下看，五个传感器从左到右依次是S1(也就是Sensor1),S2,S3,S4,S5,分别插在A1,A2,A3,A4,A5上。那么代码就是：

```
robot->configSensor(A1, A2, A3, A4, A5);
```

会有老师教你们接线的，不用记住。这些config有关的你们都可以复制粘贴。

那么，这个机器人差不多成型了，接下来我们要为它注入灵魂，欸嘿嘿...

```
robot->initialRobot();
```

initial是初始化的意思，Robot我们再复习一次是机器人的意思。在setup里吟唱这句咒语会为你的机器人注入灵魂。一定不要忘了加上这句话！否则你的机器人将会成为一个没有灵魂的灭世者的存在

这么强大的存在，当然要据为己有！现在我们来构筑灵魂链接！！！

```
robot->pair();
```

呃...好中二啊我...其实就是跟遥控器配个对...pair的意思就是配对啦...这句话一写上你的机器人就会等待你的遥控器配对，把遥控器打开按下上面的SELECT按钮，你的机器人显示屏上就会显示配对提示。注意，这个方法会在内部用到显示屏和遥控器接收器，所以在配对之前必须已经为机器人配置了显示屏和控制器，并且你的机器人已经注入了灵魂(滑稽)。

- 这时候你的所有代码应该已经长成这个样子了：

```
#include "Grobot.h"

Grobot *robot = new Grobot();
Gmotor *lmt = new Gmotor(3, 2), *rmt = new Gmotor(5, 4);

void setup()
{
    robot->configMotor(lmt, rmt);
    robot->configTFT(QD_TFT180A, 51, 52, 32, 34, 0, 33);
    robot->configController(A14, A7, A13, A6, true, true);
    robot->configSensor(A1, A2, A3, A4, A5);
    robot->initialRobot();
    robot->pair();
}

void loop()
```

- 如果不一样的话，请立即检查自己哪里理解错了并立马报警

嗯？这位同学把手放下，你想问什么？奥，奥好的你是说为什么没有在setup里看到熟悉的pinMode之类的对吧？好的关于这个....你可以理解成其实你的机器人帮你做了这件事情。在你配置各个部分的时候你的机器人就会自行处理这些，用不着您大驾光临自己写了。是不是很方便？

---

现在摆在你们面前的，有一个坏消息和一个更坏的消息。坏消息是接下来我们才开始真正的学习。更坏的消息是上面这一堆看上去复杂的东西你们都自学了，因为它们都是固定不变的，到时候直接复制粘贴就行的。

怎么样是不是很想打我？欸嘿嘿嘿.....

---

好的前排这位已经快吐了的同学，请你立马打起精神来，我们现在开始真正的学习！也就是到时候你们要写的，`void loop(){}` 里面的内容！

---

这个标题是用来充字数的并表示我们即将开始正式课程

咳咳....(喝水的声音)好的欢迎回来。在刚才的课程中我们已经学习了没有用的内容...(低头躲过飞过来的香蕉皮和拖鞋)，那么现在...啊！(被西红柿爆头)

保安请把这位同学带走出去站着去

好的我们继续。1

Contributors



VisualDust

