

STACKFORCE

栈擎

轻量机器人开发平台

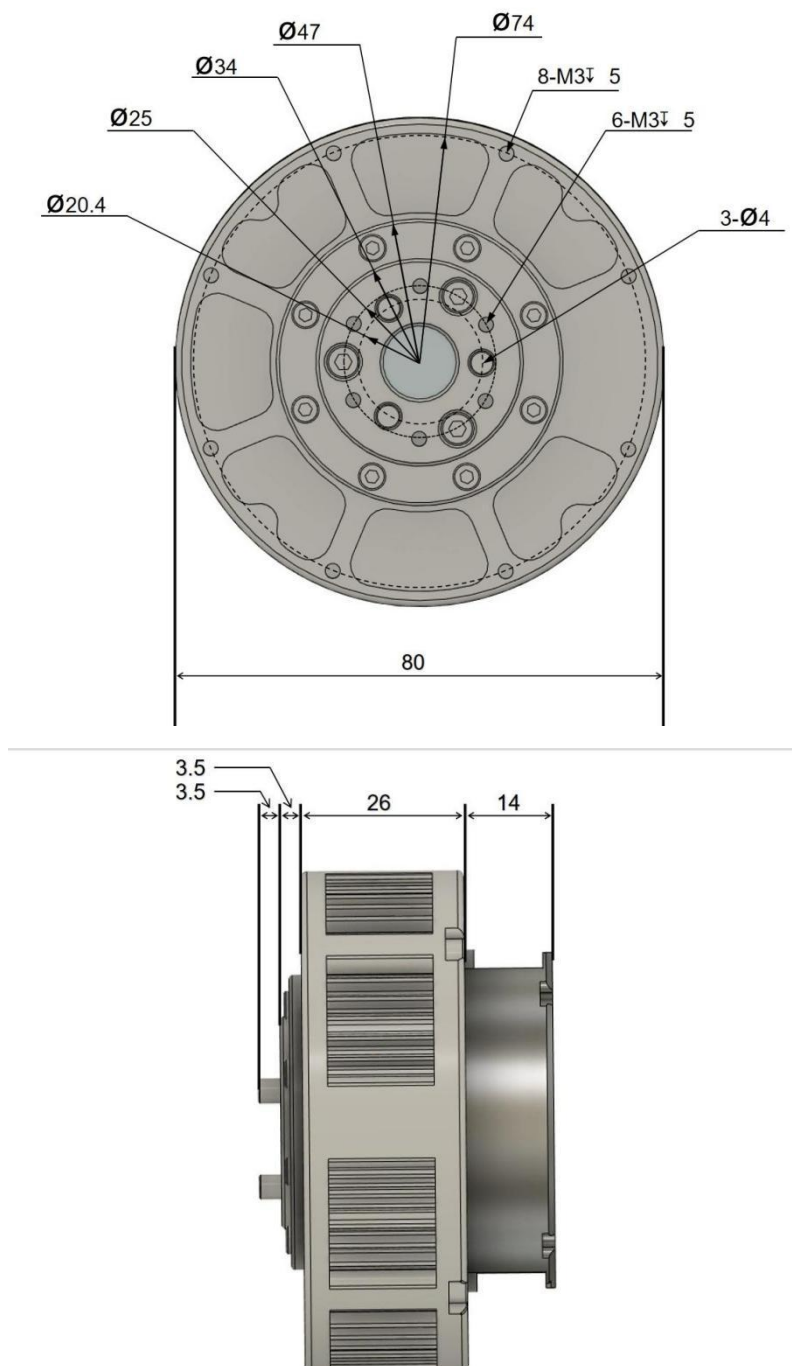
6010 一体化关节电机 使用说明

目录

第一章 电机规格参数	- 1 -
1.1 电机尺寸	- 1 -
1.2 电气特性	- 2 -
第二章 电机接口	- 3 -
2.1 电源供电接口与 CAN 接口	- 3 -
2.2 RGB 指示灯	- 3 -
2.3 CAN 总线终端电阻开关	- 3 -
2.4 串口通信端口	- 4 -
第三章 CAN 通信协议及说明	- 5 -
3.1 电机反馈帧(心跳包)	- 5 -
3.2 MIT 模式控制帧	- 6 -
3.2.1 MIT 模式介绍	- 6 -
3.2.2 MIT 模式命令信号	- 7 -
3.3 重启电机	- 9 -
3.4 使能电机	- 9 -
3.5 失能电机	- 9 -
3.6 位置零点	- 9 -
第四章 串口通信	- 10 -
4.1 串口工具	- 10 -
4.2 修改 CANID	- 10 -
4.3 修改 CAN 模式	- 11 -

第一章 电机规格参数

1.1 电机尺寸



单位: mm

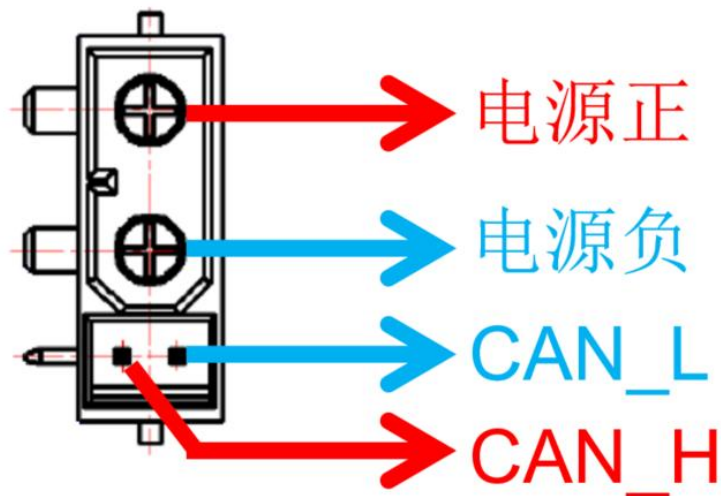
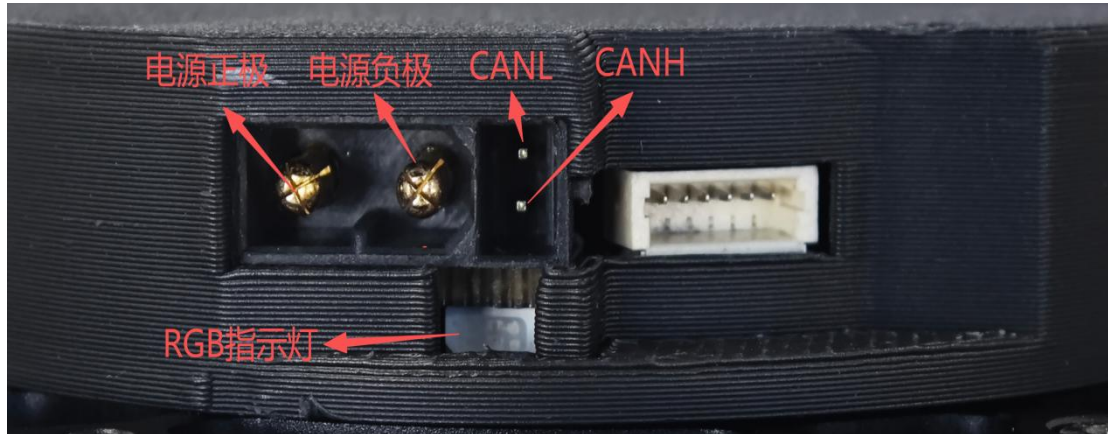
1.2 电气特性

规格	参数
额定电压 (V)	24
额定电流 (A)	10.5
额定功率 (W)	240
额定扭矩 (Nm)	5
堵转扭矩 (Nm)	11
额定转速 (RPM)	120
最大转速 (RPM)	270
减速比	8:1
控制模式	MIT 协议
	位置控制 (rad)
	速度控制 (rad/s)
	力矩控制 (Nm)
外径 (mm)	80
厚度 (mm)	47
编码器	磁编+霍尔 (14 位)
极对数	14
重量 (g)	392±10%
相电阻 (Ω)	0.48±10%
相电感 (uH)	368±10%

第二章 电机接口

2.1 电源供电接口与 CAN 接口

本关节电机的电源接口与 CAN 通信的接口集成在了同一个端子上，这个端子的型号为 XT30(2+2)，下图所示。



2.2 RGB 指示灯

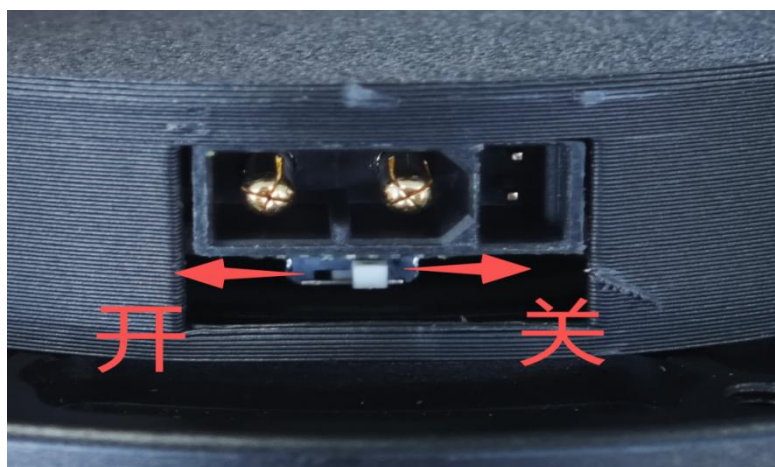
RGB 亮红灯，电机处于失能状态。

RGB 亮绿灯，电机处于使能状态。

2.3 CAN 总线终端电阻开关

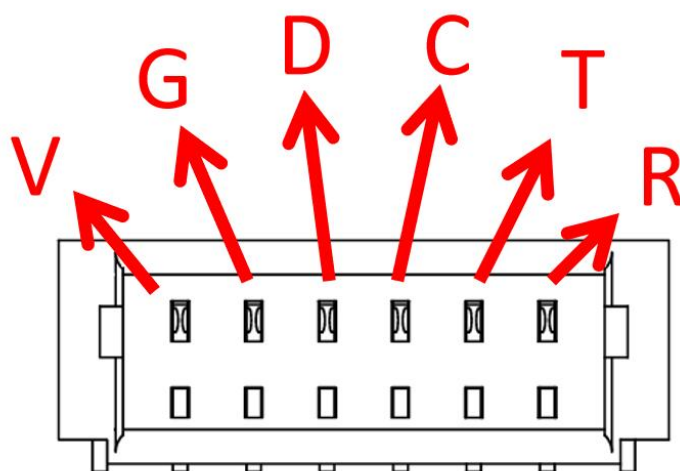
本关节电机在驱动板中集成了 $120\ \Omega$ 的 CAN 通信终端电阻，该终端电阻的选择波动

开关位于电源端子的背面，如下图所示。



2.4 串口通信端口

本关节电机具备串口通信端口，同时该端口可用于升级固件与配置，端子型号为 SH1.0 6p 端子，如下图所示。



第三章 CAN 通信协议及说明

电机出厂已校准好，发送控制帧即可使用，电机默认使用 CAN 标准帧格式，固定波特率为 1Mbps，可通过串口切换 FDCAN 通信，FDCAN 固定波特率为 5MBPS。下面按功能分为控制帧和反馈帧，控制帧及主设备发送给电机的控制命令，反馈帧为电机向主设备发送电机的状态数据。

PS：电机出厂默认电机 ID 号为 0X01。

3.1 电机反馈帧(心跳包)

电机反馈帧 ID 由命令 ID 0x780 + 电机 ID 组成（假如电机 ID 为 0x01，则反馈帧 ID = 0x781），电机需要使能后才会回传反馈帧，反馈内容包括电机输出轴的位置、速度、扭矩和温度信息，帧格式定义为：

帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4		Byte5	Byte6	Byte7
ID	待定	POS[15:8]	POS[7:0]	VEL[11:4]	VEL[3:0]	T[11:8]	T[7:0]	TMos	TMotor

其中：

- POS 表示电机的位置信息（单位 rad）
- VEL 表示电机的速度信息（单位 rad/s）
- T 表示电机的扭矩信息（单位 Nm）
- TMos 表示驱动上的 MOS 温度（单位℃）
- TMotor 表示电机内部线圈的温度（单位℃）

例：

其中位置采用 16 位数据，速度和扭矩均使用 12 位，温度则使用 8 位，下面以电机回传的反馈帧为例说明如何换算，如下：

帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
781	01	7F	A6	7F	F8	16	4B	44
对应十六进制	[]	Pos=0x7FA6		Vel=0x7FF		T=0x816	TMos=0x4B	TMotor=0x44
对应十进制	[]	Pos=32678		Vel=2047		T=2070	TMos=75	TMotor=68

```

-12.56f, // float posMin; rad
12.56f, // float posMax; rad
-45.0f, // float velMin; rad/s
45.0f, // float velMax; rad/s
0.0f, // float kpMin;
500.0f, // float kpMax;
0.0f, // float kdMin;
5.0f, // float kdMax;
-18.0f, // float torMin;
18.0f, // float torMax;
0.0f, // float tempMin;
100.0f, // float tempMax;

```

```

float uint_to_float(uint32_t x_int, float x_min, float x_max, int bits)
{
    float span = x_max - x_min;
    float offset = x_min;
    return ((float)x_int)*span/((float)((1<<bits)-1)) + offset;
}

```

基于“线性映射”的关系，根据参数设定范围，带入 uint_to_float 函数，将反馈报文换算出对应的数据：

```

POS    = (32678 * 12.56 * 2 / 2^16) - 12.56 ≈ -0.034rad
Vel    = (2047 * 45.0 * 2 / 2^12) - 45.0 ≈ 0rad/s
T      = (2070 * 18.0 * 2 / 2^12) - 18.0 ≈ 0.198Nm
TMOS   = 75 * 100 / 2^8 ≈ 29.41℃
TMOTOR = 68 * 100 / 2^8 ≈ 26.67℃

```

3.2 MIT 模式控制帧

3.2.1 MIT 模式介绍

MIT 模式可实现 力矩、位置、速度三者混合控制。位置环与速度环是并联形式，位置环与速度环的输出值与前馈力矩 t_{ff} 相加得到参考力矩 T_{ref} 具体控制式子如下：

$$T_{ref} = k_p * (p_{des} - \theta_m) + k_d * (v_{des} - d\theta_m) + t_{ff}$$

其中：

- t_{ref} : 参考力矩，单位 N/m;
- k_p : 位置增益;
- k_d : 速度增益;
- p_{des} : 电机输出轴的期望位置，单位 rad;
- θ_m : 电机输出轴输出轴的当前位置，单位 rad;
- v_{des} : 电机输出轴的期望速度，单位 rad/s;
- $d\theta_m$: 电机输出轴的当前速度，单位 rad/s;
- t_{ff} : 前馈力矩，单位 N/m;

在 MIT 模式使用上，值得注意以下几点：

1. 当 $k_p=0$, $k_d \neq 0$ 时：匀速转动控制
 - 控制特性：给定目标速度 v_{des} ，可实现电机匀速转动
 - 存在问题：
 - 匀速转动过程中存在静差（实际速度与目标速度的偏差）。
 - 此外 k_d 不宜过大，若 k_d 过大，系统阻尼过强，会引起电机震荡（速度剧烈波动）。
2. 当 $k_p=0$, $k_d=0$ 时：输出恒定力矩
 - 控制特性：给定 t_{ff} 即可实现电机持续输出恒定扭矩。
 - 局限性：
 - 当电机空转或负载较小时，如果给定 t_{ff} 较大，电机将持续加速直到最大速度，但此时无法达到目标力矩 t_{ff} 。
 - 原因：扭矩未被负载消耗，全部转化为加速度，导致速度饱和而力矩不足。
3. 当 $k_p \neq 0$, $k_d=0$ 时：位置控制震荡风险
 - 控制特性：给定目标位置 p_{des} 可电机转动固定位置，有震荡情况。
 - 存在问题：
 - 对位置进行控制时，若 $k_d=0$ ，系统缺少微分阻尼作用，会引发电机 剧烈震荡甚至失控。
 - 原理说明：比例控制 k_p 仅根据位置误差调节力矩，缺乏对于速度变化的抑制能力，导致系统动态响应过冲后无法快速稳定。
4. 当 $k_p \neq 0$, $k_d \neq 0$ 时：复合控制场景
 - 这里下面简单介绍两种情况：
 - (1)、期望位置 p_{des} 为常量，期望速度 v_{des} 为 0 时：定点控制
 - 控制特性：实际位置趋近于 p_{des} ，实际速度 $d\theta_m$ 趋近于 0，实现位置定点稳定。
 - (2)、 p_{des} 是随时间变化的连续可导函数时，同时 v_{des} 是 p_{des} 的导数：位置与速度跟踪控制
 - 控制特性：电机按 v_{des} 跟踪旋转，并且精确跟随 p_{des} 的角度变化，实现位置与速度的双重跟踪。

3.2.2 MIT 模式命令信号

该帧为电机控制帧，帧 ID 由命令 ID 0x200+电机 ID 组成（假如电机 ID 为 0x01，

则帧 ID=0x201)，如下：

帧 ID	Byte0	Byte1	Byte2	Byte3		Byte4	Byte5	Byte6		Byte7
ID	p_des [15:8]	p_des [7:0]	v_des [11:4]	v_des [3:0]	Kp [11:8]	Kp [7:0]	Kd [11:4]	Kd [3:0]	t_ff [11:8]	t_ff [7:0]

其中：

- p_des：位置给定（单位：rad）
- v_des：速度给定（单位：rad/s）
- Kp：位置比例系数
- Kd：速度比例系数
- t_ff：前馈转矩给定值（单位：N.M）

例：

将五个参数按位组合放在 8 个字节中，位置占 16 位，速度占 12 位，Kp 占 12 位，Kd 占 12 位，前馈力矩占 12 位，下面以 MIT 控制速度为例说明如何换算，如下：

其中，位置为 0，不做控制；速度为 10rad/s；Kp 为 0，位置不做增益；Kd 为 1；前馈力矩为 0。

```
-12.56f, // float posMin; rad
12.56f,  // float posMax; rad
-45.0f,  // float velMin; rad/s
45.0f,   // float velMax; rad/s
0.0f,    // float kpMin;
500.0f,  // float kpMax;
0.0f,    // float kdMin;
5.0f,    // float kdMax;
-18.0f,  // float torMin;
18.0f,   // float torMax;
```

```
uint16_t float_to_uint(float x, float x_min, float x_max, uint8_t bits){
    float span = x_max - x_min;
    float offset = x_min;
    return (uint16_t) ((x-offset)*((float)((1<<bits)-1))/span);
}
```

基于“线性映射”的关系，根据参数设定范围，带入 float_to_uint 函数，将需要发送的 MIT 信息换算一下：

$$\begin{aligned}
 p_des &= (0 - (-12.56)) * ((2^{16} - 1) / 25.12) \approx 32,767 = 0x7FFF \\
 v_des &= (10 - (-45)) * ((2^{12} - 1) / 90) \approx 2,502 = 0x9C6 \\
 Kp &= (0 - 0) * ((2^{12} - 1) / 500) = 0 = 0x000 \\
 Kd &= (1 - 0) * ((2^{12} - 1) / 5) \approx 819 = 0x333 \\
 t_ff &= (0 - (-18)) * ((2^{12} - 1) / 36) \approx 2027 = 0x7FF
 \end{aligned}$$

帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
	p_des=0x7FFF		v_des=0x9AB		Kp=0x000	Kd=0x333		t_ff=0x7FF
201	0x7F	0xFF	0x9C	0x60	0x00	0x33	0x37	0xFF

3.3 重启电机

发送该控制帧电机重新启动。帧 ID 由命令 ID 0x00 + 电机 ID（假如电机 ID 为 0x01，即帧 ID 为 0x01），如下：

帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
ID	FF	FF	FF	FF	FF	FF	FF	FB

3.4 使能电机

电机上电默认失能状态，需要发送“使能”帧才可以控制。帧 ID 由命令 ID 0x00 + 电机 ID（假如电机 ID 为 0x01，即帧 ID 为 0x01），如下：

帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
ID	FF	FF	FF	FF	FF	FF	FF	FC

3.5 失能电机

电机上电默认失能状态。帧 ID 由命令 ID 0x00 + 电机 ID（假如电机 ID 为 0x01，即帧 ID 为 0x01），发送该命令可使电机失能，如下：

帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
ID	FF	FF	FF	FF	FF	FF	FF	FD

3.6 位置零点

发送该帧会设置当前位置为零点，且掉电不丢失。帧 ID 由命令 ID 0x00 + 电机

ID（假如电机 ID 为 0x01，即帧 ID 为 0x01），如下：

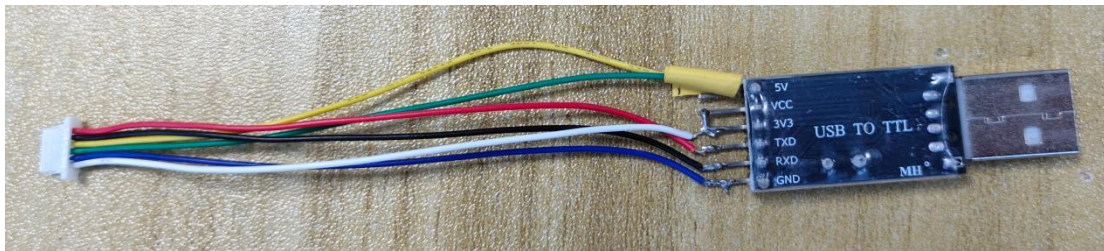
帧 ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
ID	FF	FF	FF	FF	FF	FF	FF	FE

第四章 串口通信

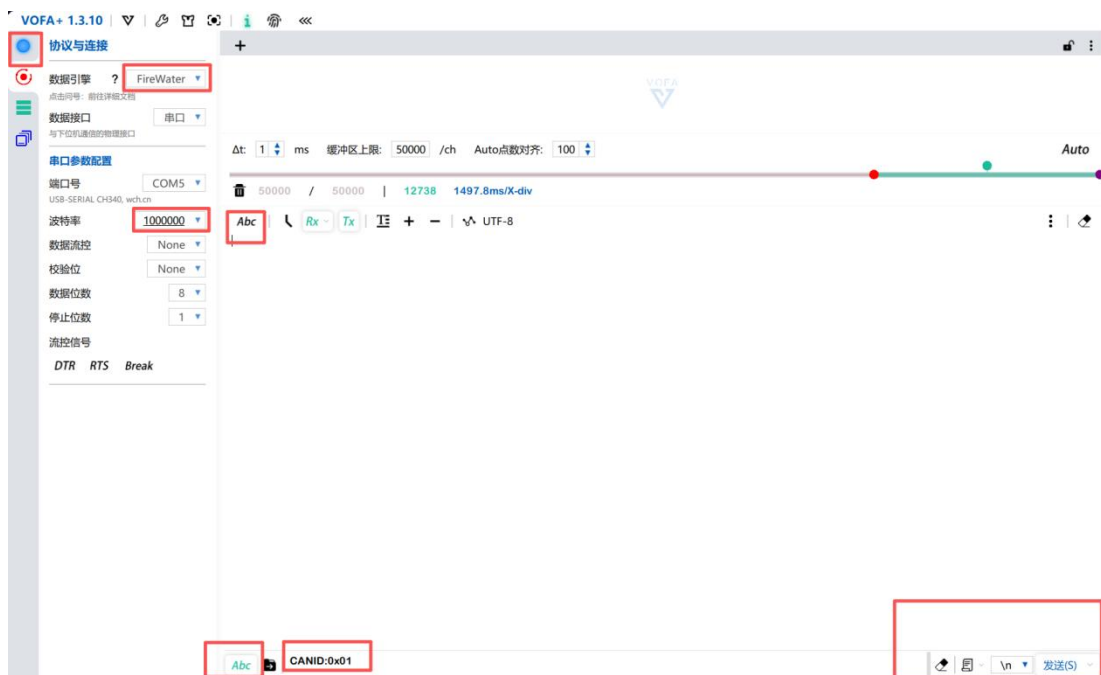
电机的 CANID 与 CANMode 都是通过串口进行修改，电机出厂默认 CANID 为 0x01，CANMode 为 CAN2.0 1Mbps。

4.1 串口工具

由上 2.4 章节所示接口，依次将 V，G，T，R 连接到串口通信模块的 VCC（3.3V），GND，RX，TX（RX 与 TX 要交叉连接）。如下图所示：



4.2 修改 CANID：（串口波特率为 1Mbps）



发送的 CANID 是 0x**, 设置的 ID 就是 0x**, 最大不能超过 0x7F。设置 CANID 成功后会打印如下日志:

```
CANID:0x02
Set Success: CAN ID = 0x02
CAN_ID = 0x02
CAN Mode = CAN
Control Mode: MIT Mode
Encoder:1
```

4.3 修改 CAN 模式:

串口发送 CANMODE:0 或 CANMODE:1

CANMODE:0 为 CAN2.0 模式 (1Mbps), CANMODE:1 为 CANFD 模式 (5Mbps)。

修改 CAN 模式成功如下图所示:

