AK系列动力模组使用说明书

目录

☆使用前请仔细阅读说明书

☆请妥善保存 以便查阅

1. 安全注意事项与免责声明

a. 请在本文件规定的温度范围（-20℃-50℃）和湿度90%以下的工作环境中使用。

b. 避免异物进入电机内部，导致转子运行异常。

c. 使用前确保接线正常、稳固。

d. 用户请勿私自拆卸电机，否则会影响电机的控制精度，甚至导致电机运行异常。

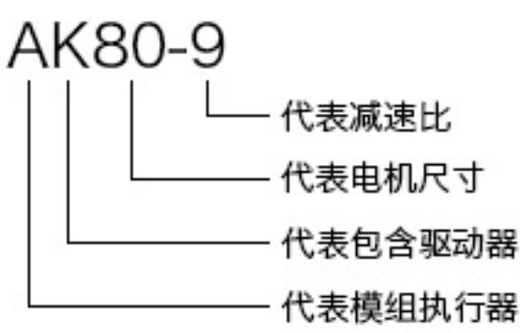
2. 产品概述

2.1 产品简介

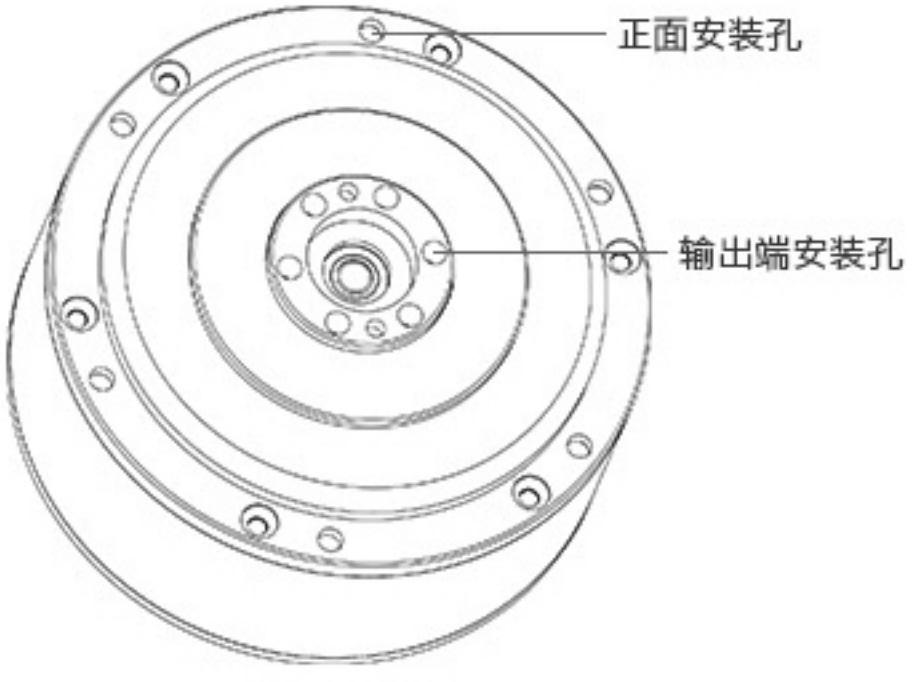
本公司模组电机是一款集成行星减速器、驱动器、编码器、高可靠性的无刷直流电机。可广泛应用于足式机器人、机械臂、自动化设备、科研教育等领域。驱动器采用磁场定向控制（FOC）算法，配合高精度的角度传感器，能实现精准的位置、力矩控制。多槽极数的设计、稀土材料磁铁和高精度的减速器为电机能够输出更稳定、更大的扭矩。本公司模组电机支持多种通信协议，通过与PC通信，提供人机交互界面，使用户更快、更精准的控制电机。

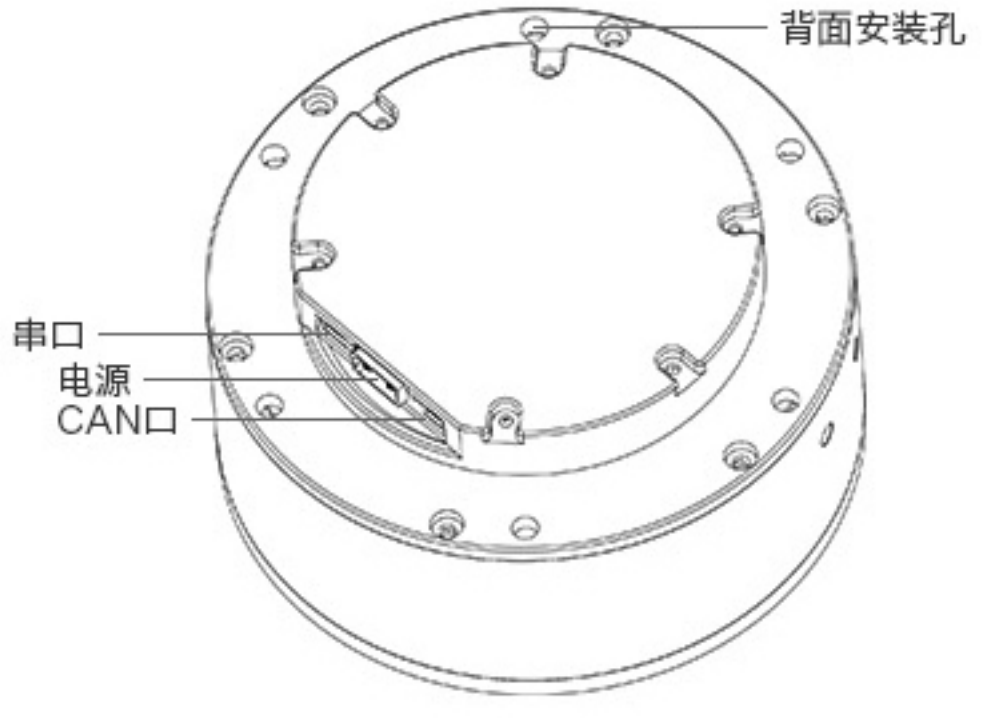
2.2 模组电机型号定义

例如:



2.3 电机外观，安装结构与连接口定义





2.4 串口信号端口

通过UART连接PC端，设置AK电机的零点位置，校准编码器，设置驱动器ID号，最大电流等参数。

线序：从上到下分别是A(GND)、B(RX)、C(TX)。线长200mm。

2.5 电源接口

通过XT30接头连接电源（额定电压24V或48V）为电机供电。

线序：从上到下分别是A(电源负极--黑色)、B(电源正极--红色)。

2.6 CAN信号接口

外部设备可以通过CAN信号线发送控制指令，反馈电机状态信息。

CAN总线比特率为1Mbps，主机ID号默认为0x00。

线序：从上到下分别是A（CAN_H）、B(CAN_L)。

线长：200mm。

3. 使用说明

3.1 CAN 通信协议

CAN速率：1MHZ，电机接收报文格式。用于向电机发送控制指令，控制电机的位置、转速、电流。

3.2 特殊CAN代码

进入电机控制模式 {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,0XFC },

退出电机控制模式 {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0XFD }

设置电机当前位置为0点 {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0XFE }

注意：使用CAN 通信控制电机时必须先进入电机控制模式!

标识符：设置的电机ID号（默认为1）　　帧类型：标准帧　　帧格式：DATA　　DLC：8字节

数据域	DATA[0]	DATA[1]	DATA[2]	DATA[3]		DATA[4]	DATA[5]
数据位	7-0	7-0	7-0	7-4	3-0	7-0	7-0
数据内容	电机位置高8位	电机位置低8位	电机速度高8位	电机速度低4位	KP值高4位	KP值低8位	KD值高8位

数据域	DATA[6]		DATA[7]
数据位	7-4	3-0	0-7
数据内容	KD值低4位	电流值高4位	电流值低8位

电机发送报文格式

标识符：0X00+驱动器ID　　帧类型：标准帧　　帧格式：DATA　　DLC：6字节

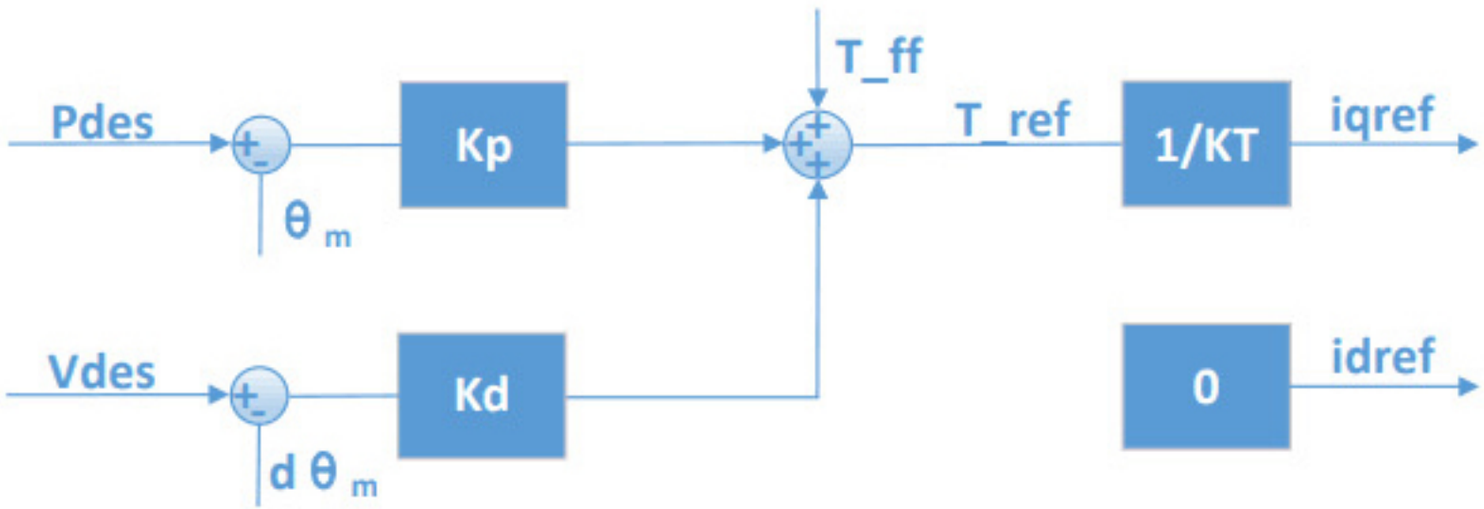
数据域	DATA[0]	DATA[1]	DATA[2]	DATA[3]	DATA[4]	DATA[4]	DATA[5]
数据位	7-0	7-0	7-0	7-0	7-4	3-0	7-0
数据内容	驱动器ID号	电机位置高8位	电机位置低8位	电机速度高8位	电机速度低4位	电流值高4位	电流值低8位

3.3 参数范围

模组电机	AK60-6	AK80-6	AK80-9	AK10-9		AK70-10		AK80-80	
电机位置(rad)	-12.5f~12.5f								
电机速度rad/S	-41.87~41.87	-38.2~38.2	-25.64~25.64	24V	48V	24V	48V	24V	48V
				-23.24~23.24	-46.57~46.57	-20.94~20.94	-41.87~41.87	-2.93~2.93	-5.86~5.86
电机扭矩Nm	-9~9	-12~12	-18~18	-54~54		-24.8~24.8		-144~144	
KP的范围	0~500								
KD的范围	0~5								

3.4 内部控制器 pid 示意图

若想实现纯位置、纯速度、纯转矩控制，仅需给对应的变量赋值其余量赋 0 即可。比如若想实现位置控制，则在发包的时候将电机位置赋值，转矩和速度则发 0。Kp控制的是位置环的参数，Kd控制的是速度环的参数，所以理论来说在纯位置模式下Kd应该赋 0。纯速度模式下Kp应该赋 0。通过CAN输入的控制命令包含了期望位置Pdes、期望速度Vdes、前馈转矩ttf、控制参数Kp、Kd。由代码可知，这些控制命令按如下控制框图的方式组合成闭环。其中,θm表示机械角度、dθm表示机械角速度、tef表示参考转矩、KT是转矩常数、 iqrefl 和idref分别表示q轴和d轴的参考电流,作为FOC控制的输入。可以看出，这样的控制方式很灵活，既可以是纯位置控制、纯速度控制、纯转矩控制，也可以位置加转矩前馈、速度加转矩前馈等方式控制电机。



3.5 CAN 通信收发代码例程

注*以下函数内定义的浮点型需根据电机的型号设置，该参数仅供参考，
请根据上文“3.3参数范围”的表格进行设置。

发送例程代码

```
void pack_cmd(CANMessage * msg, float p_des, float v_des, float kp, float kd, float t_ff){
/// limit data to be within bounds ///
float P_MIN =-95.5;
float P_MAX =95.5;
float V_MIN =-30;
float V_MAX =30;
float T_MIN =-18;
float T_MAX =18;
float Kp_MIN =0;
float Kp_MAX =500;
float Kd_MIN =0;
float Kd_MAX =5;
float Test_Pos=0.0;
p_des = fminf(fmaxf(P_MIN, p_des), P_MAX);
v_des = fminf(fmaxf(V_MIN, v_des), V_MAX);
kp = fminf(fmaxf(KP_MIN, kp), KP_MAX);
kd = fminf(fmaxf(KD_MIN, kd), KD_MAX);
t_ff = fminf(fmaxf(T_MIN, t_ff), T_MAX);
/// convert floats to unsigned ints ///
int p_int = float_to_uint(p_des, P_MIN, P_MAX, 16);
int v_int = float_to_uint(v_des, V_MIN, V_MAX, 12);
int kp_int = float_to_uint(kp, KP_MIN, KP_MAX, 12);
int kd_int = float_to_uint(kd, KD_MIN, KD_MAX, 12);
int t_int = float_to_uint(t_ff, T_MIN, T_MAX, 12);
/// pack ints into the can buffer ///
    msg->data[0] = p_int>>8;                    //位置高8
    msg->data[1] = p_int&0xFF;                  //位置低8
msg->data[2] = v_int>>4;                    //速度高8位
    msg->data[3] = ((v_int&0xF)<<4)|(kp_int>>8);  //速度低4位   KP高4位
    msg->data[4] = kp_int&0xFF;                 //KP低8位
    msg->data[5] = kd_int>>4;                   //Kd高8位
    msg->data[6] = ((kd_int&0xF)<<4)|(t_int>>8);  //KP低4位扭矩高4位
    msg->data[7] = t_int&0xff;                  //扭矩低8位
    }
```

例程的程序截图:

```
void pack_cmd(uint8_t *msg, float p_des, float v_des, float kp, float kd, float t_ff)
{
    /// limit data to be within bounds ///
    float P_MIN =-95.5;
    float P_MAX =95.5;
    float V_MIN =-30;
    float V_MAX =30;
    float T_MIN =-18;
    float T_MAX =18;
    float Kp_MIN =0;
    float Kp_MAX =500;
    float Kd_MIN =0;
    float Kd_MAX =5;
    float Test_Pos=0.0;


    p_des = fminf(fmaxf(P_MIN, p_des), P_MAX);
    v_des = fminf(fmaxf(V_MIN, v_des), V_MAX);
    kp = fminf(fmaxf(Kp_MIN, kp), Kp_MAX);
    kd = fminf(fmaxf(Kd_MIN, kd), Kd_MAX);
    t_ff = fminf(fmaxf(T_MIN, t_ff), T_MAX);
    /// convert floats to unsigned ints ///

    int p_int = float_to_uint(p_des, P_MIN, P_MAX, 16);
    int v_int = float_to_uint(v_des, V_MIN, V_MAX, 12);
    int kp_int = float_to_uint(kp, Kp_MIN, Kp_MAX, 12);
    int kd_int = float_to_uint(kd, Kd_MIN, Kd_MAX, 12);
    int t_int = float_to_uint(t_ff, T_MIN, T_MAX, 12);

    /// pack ints into the can buffer ///

    msg[0] = p_int>>8; //位置高 8                                    (High position
    /// pack ints into the can buffer ///

    msg[0] = p_int>>8; //位置高 8                                    (High position
    msg[1] = p_int&0xFF; //位置低 8                                  (Low position
    msg[2] = v_int>>4; //速度高 8 位                                 (High speed 8)
    msg[3] = ((v_int&0xF)<<4)|(kp_int>>8); //速度低 4 位 KP 高 4 位    (The speed is
    msg[4] = kp_int&0xFF; //KP 低 8 位                               (KP Low 4 )
    msg[5] = kd_int>>4; //Kd 高 8 位                                 (KP Higt 4 )
    msg[6] = ((kd_int&0xF)<<4)|(t_int>>8); //KP 低 4 位扭矩高 4 位    (KP 4 lower to
    msg[7] = t_int&0xff;  //扭矩低 8 位                              (Torque is 8 b


}

int float_to_uint(float x, float x_min, float x_max, unsigned int bits)
```

发包时所有的数都要经以下函数转化成整型数之后再发给电机。

```
int float_to_uint(float x, float x_min, float x_max, unsigned int bits)
{
/// Converts a float to an unsigned int, given range and number of bits ///
float span = x_max - x_min;
if(x < x_min) x = x_min;
else if(x > x_max) x = x_max;
return (int) ((x- x_min)*((float)((1<<bits)/span)));
}
```

例程的程序截图：

```
int float_to_uint(float x, float x_min, float x_max, unsigned int bits)
{
  /// Converts a float to an unsigned int, given range and number of bits ///
  float span = x_max - x_min; //计算差值
  /*****   判断是否小于最小值   ****/
  if(x < x_min) //小于最小值，取最小
    x = x_min;
  else if(x > x_max)//大于最大值，取最大
    x = x_max;
  return (int) ((x- x_min)*((float)((1<<bits)-1)/span));
}
```

接收例程代码

```
void unpack_reply(CANMessage msg){
    /// unpack ints from can buffer ///
    int id = msg.data[0];                          //驱动ID号
    int p_int = (msg.data[1]<<8)|msg.data[2];             //电机位置数据
    int v_int = (msg.data[3]<<4)|(msg.data[4]>>4);        //电机速度数据
    int i_int = ((msg.data[4]&0xF)<<8)|msg.data[5];       //电机扭矩数据
    /// convert ints to floats ///
    float p = uint_to_float(p_int, P_MIN, P_MAX, 16);
    float v = uint_to_float(v_int, V_MIN, V_MAX, 12);
    float i = uint_to_float(i_int, -I_MAX, I_MAX, 12);
          if(id == 1){
postion = p;                          //根据ID号读取对应数据
speed = v;
torque = i;
       }
```

例程的程序截图：

```
138 void unpack_reply(uint8_t *data)
139 {
140   /// unpack ints from can buffer ///
141   int id = data[0]; //驱动 ID 号
142   int p_int = (data[1]<<8)|data[2];        //电机位置数据
143   int v_int = (data[3]<<4)|(data[4]>>4);    //电机速度数据
144   int i_int = ((data[4]&0xF)<<8)|data[5];   //电机扭矩数据
145   /// convert ints to floats ///
146   float p = uint_to_float(p_int, P_MIN, P_MAX, 16);
147   float v = uint_to_float(v_int, V_MIN, V_MAX, 12);
148   float i = uint_to_float(i_int, -T_MAX, T_MAX, 12);
149   if(id == 1)
150   {
151     postion = p;      //根据 ID 号读取对应数据
152     speed = v;
153     torque = i;
154   }
155 }
156
```

Position 全局变量，位置

Speed 全局变量，速度

Torque 全局变量，扭矩

收包时所有的数都要经以下函数转化成浮点型。

float uint_to_float(int x_int, float x_min, float x_max, int bits)

{

/// converts unsigned int to float, given range and number of bits ///

float span = x_max - x_min;

float offset = x_min;

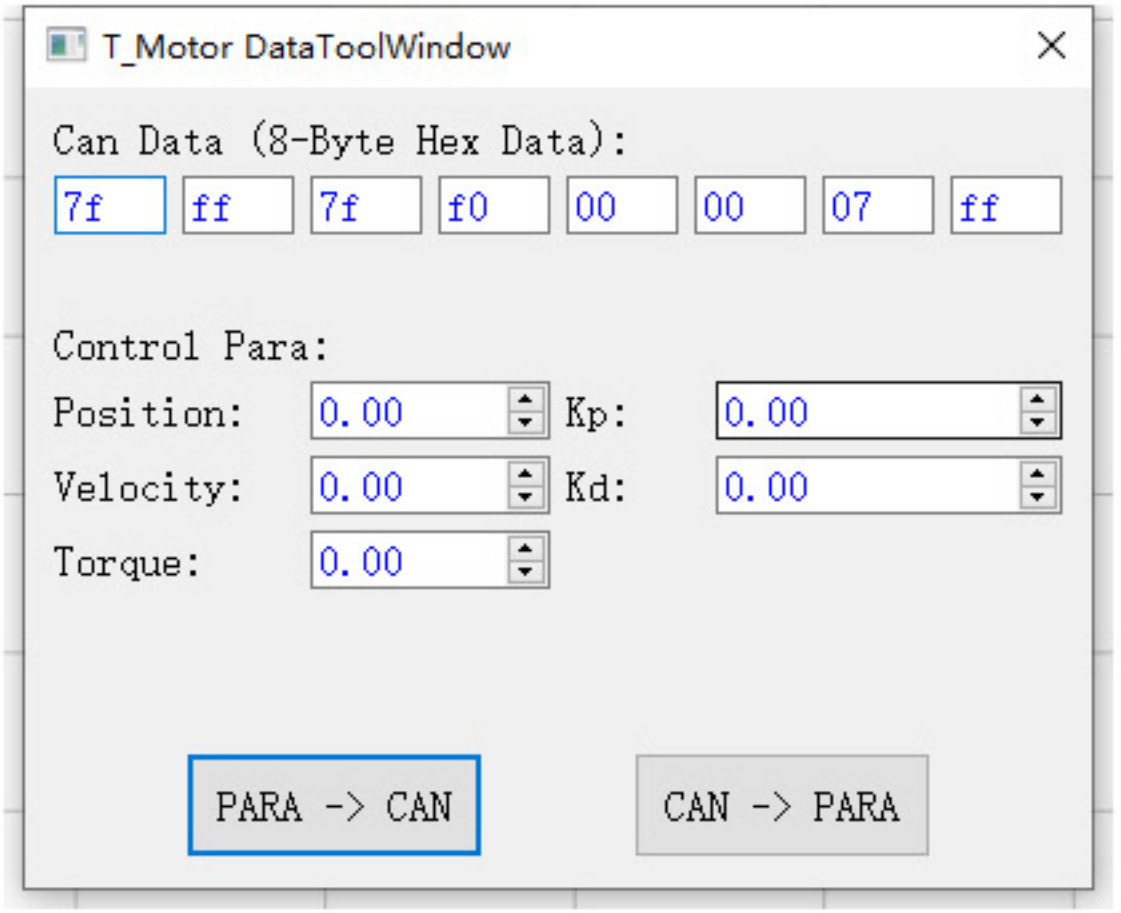
return ((float)x_int)*span/((float)((1<<bits)-1)) + offset;

}

例程的程序截图：

```
30 float uint_to_float(int x_int, float x_min, float x_max, int bits)
31 {
32   /// converts unsigned int to float, given range and number of bits ///
33   float span = x_max - x_min;
34   float offset = x_min;
35   return ((float)x_int)*span/((float)((1<<bits)-1)) + offset;
36 }
37
```

上位机的程序截图：



Can发送报文数据截图：

序号	系统时间	时间标识	CAN通道	传输方向	ID号	帧类型	帧格式	长度	数据
01929	17:36:28.598	0xF5AF10	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01930	17:36:28.718	0xF5B2EC	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01931	17:36:28.809	0xF5B6C9	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01932	17:36:28.898	0xF5BAA6	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01933	17:36:29.018	0xF5BE82	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01934	17:36:29.107	0xF5C25F	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01935	17:36:29.198	0xF5C63C	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01936	17:36:29.319	0xF5CA18	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01937	17:36:29.408	0xF5CDF5	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01938	17:36:29.498	0xF5D1D1	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01939	17:36:29.588	0xF5D5AE	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01940	17:36:29.708	0xF5D98B	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01941	17:36:29.799	0xF5DD67	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01942	17:36:29.888	0xF5E144	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF
01943	17:36:30.009	0xF5E520	ch1	接收	0x0200	数据帧	标准帧	0x08	x| 7F FF 7F F0 00 00 07 FF

注*如果上位机的设置的报文与接收到报文对不上，请设置修改上位机的一些基础参数。

3.6 调参软件

使用 USB 转串口工具，将电机连接至计算机，对电机进行参数设置。

1、使用配套信号线，连接电机和USB转串口工具，2p接口为CAN，3p接口为串口。然后将USB转串口工具接入计算机。

2、接通电源为电机供电，设置完成前请勿切断电源或连接。

3、运行调参软件。软件界面选择串口后，单击CONNECT即可连接。选择电机后，单击ENTER_MA_MODE进入电机模式，进行调试。

特征参数

模组电机	AK60-6	AK80-6	AK80-9	AK10-9		AK70-10		AK80-80	
最大空载转速rpm	400	365	485	24V	48V	24V	48V	24V	48V
				222	445	200	400	28	57
空载电流 A	1								
额定扭矩 Nm	3	6	9	18		8.3		48	
定位精度 bit	12								
额定电流 A	7.4	12	12	22		8.8		13	
最大效率	80%								
堵转扭矩 14Nm	9	12	18	54		24.8		144	
堵转电流 24A	22	24	24	68		26.1		40	
电机极对数 21	14	21							

注意：由于编程环境不同，以上例程例程如有错误，可以对比协议，如有差异一切以协议为主。

以上例程环境是keil5 5.33版本，运用库是hal库的STM32Cube FW_F4 V1.25.2的版本。

4. 包装清单