交流伺服电机驱动器

YZ-ACSD608_v6.7

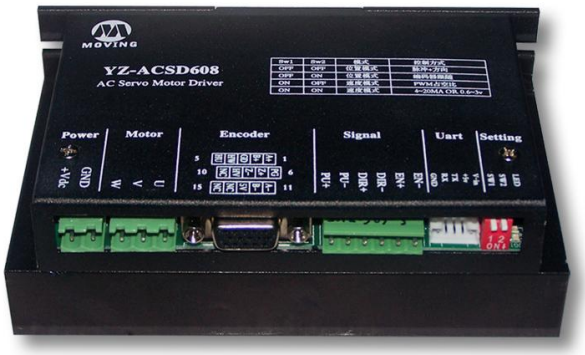
一．产品特性

高转速，高精度，高稳定性

低噪音，低发热，低成本

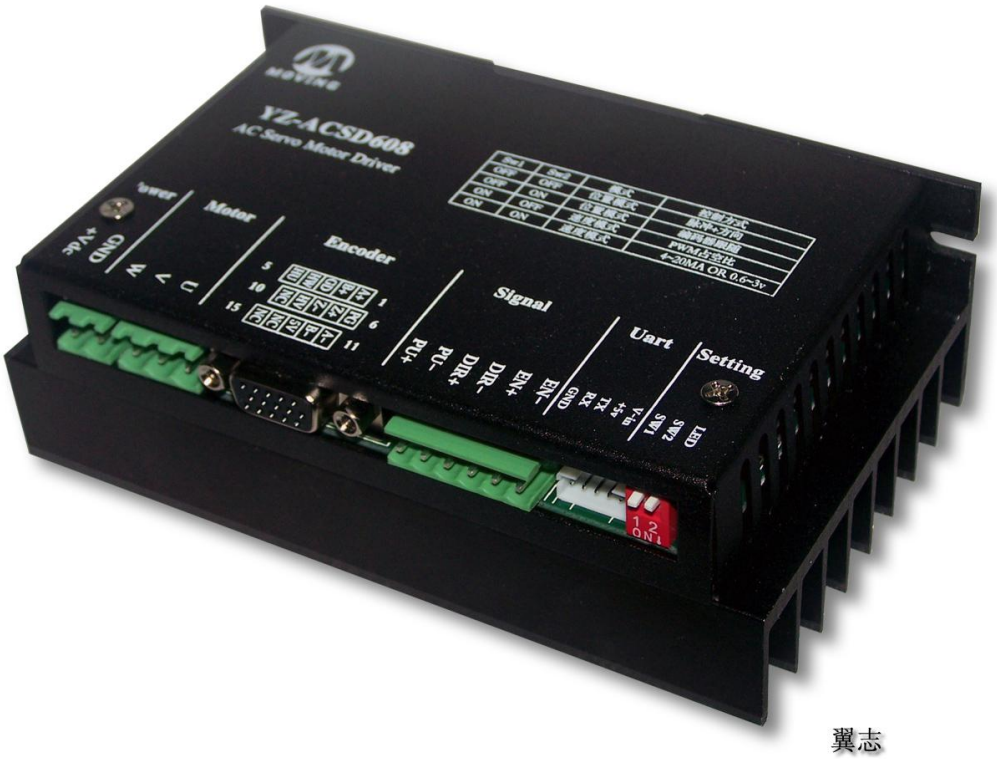
全数字交流伺服电机驱动器

1. 采用最新32位dsp处理器
2. FOC磁场定向矢量控制
3. S型加减速曲线
4. 16位电子齿轮
5. 20~50VDC输入，支持50~500W交流伺服电机
6. 2500/1250线编码器，转速1RPM~3000RPM



翼志

www.wingzstudio.net



翼志

www.wingzstudio.net

1. 采用 32 位高速 DSP 芯片。
2. FOC 场定向矢量控制，支持位置/速度闭环。
3. 位置模式支持指令脉冲+方向 或 正交脉冲信号。
4. 速度模式支持 PWM 占空比信号 或 4~20ma 电流 或 0.6~3V 电压信号控制。
5. 16 位电子齿轮功能，1~65535 / 1~65535 。
6. 供电电压+20V~50V。支持 50~500W 交流伺服电机。
7. 内部隔离 485（modbus 协议 RTU 模式）（19200，8，N,1）控制方式。可以设定驱动器地址，简化控制系统。也可以直接通过 PC 机控制，并提供 PC 机测试软件。
8. 具有欠压，过压，堵转，过热保护。
9. 提供隔离输出的 到位信号、报警输出信号、编码器零点信号。

二. 驱动器接口

1. 控制信号接口（驱动器上标识：Singal）

名称	功能
PU+(+5V) PU-(PU)	脉冲控制信号：脉冲上升沿有效；PU-高电平时 4～5V，低电平时 0～0.5V。为了可靠响应脉冲信号，脉冲宽度应大于 1.2μs。如采用+12V 或+24V 时需串电阻。
DIR+(+5V) DIR-(DIR)	方向信号：高/低电平信号，为保证电机可靠换向，方向信号应先于脉冲信号至少 5μs 建立。DIR-高电平时 4～5V，低电平时 0～0.5V。
EN+(+5V) EN-(EN)	使能信号：此输入信号用于使能或禁止。ENA+ 接+5V，ENA-接低电平（或内部光耦导通）时，驱动器将切断电机各相的电流使电机处于自由状态，此时脉冲不被响应。当不需用此功能时，使能信号端悬空即可。另外，当驱动器报警时，可以禁止再使能，可以清除报警。

ACSD608 驱动器采用差分式接口电路可适用差分信号，单端共阴及共阳等接口，内置高速光电耦合器，允许接收长线驱动器，集电极开路和 PNP 输出电路的信号。在环境恶劣的场合，我们推荐用长线驱动器电路，抗干扰能力强。

2. 强电接口（驱动器上标识：Power Motor）

名称	功能
+V	直流电源正极，+24V~+48V，电压过低会引起驱动器报警停机。
GND	直流电源地。
U	电机 U 相线圈。
V	电机 V 相线圈。
W	电机 W 相线圈。

3. 电机信号接口（驱动器上标识：Encoder）

端子号	符号	功能
1	A+	编码器 A 相正
2	B+	编码器 B 相正
3	GND	+5v 电源地
4	HALL_W	磁极信号 W 相
5	HALL_U	磁极信号 U 相
6	PG	外壳
7	Z+	编码器 Z 相正
8	Z-	编码器 Z 相负
9	HALL_V	磁极信号 V 相
10	NC	无连接
11	A-	编码器 A 相负
12	B-	编码器 B 相负
13	+5V	+5v 电源，给传感器供电
14	NC	无连接
15	NC	无连接

4. 通信与输出信号接口（驱动器上标识：Uart）

可以通过专用 485 电缆连接 PC 机。通过提供的上位机软件可以设置驱动器参数，和测试驱动器，并提供一些诊断信息，来排除驱动器故障。

注：1 脚为下排左边第一个引脚，6 脚为上排左边第一个引脚

端子号	符号	功能
1	GND	信号地
2	485A	485 通信 A 端（D+端）
3	485B	驱动器串口接收端（TTL 电平）
4	+5V	外供 5V 最大 100mA
5	V_in	调速电压或电流信号输入
6	COM	输出信号公共端
7	WR	报警信号输出。报警时为低电平
8	PF	到位信号输出。到位时为低电平
9	ZO	编码器零点信号输出。到零点时为低电平
10	5V_485	内部 485 供电接口（需要外部供电）。485 供电的地为 COM 引脚

5. DIP 开关

功能选择：

SW1	SW2	模式	控制方式
OFF	OFF	位置模式	脉冲+方向
OFF	ON	位置模式	编码器跟随
ON	OFF	速度模式	PWM 占空比
ON	ON	速度模式	4~20MA 或 0.6~3V

6. 状态指示与报警

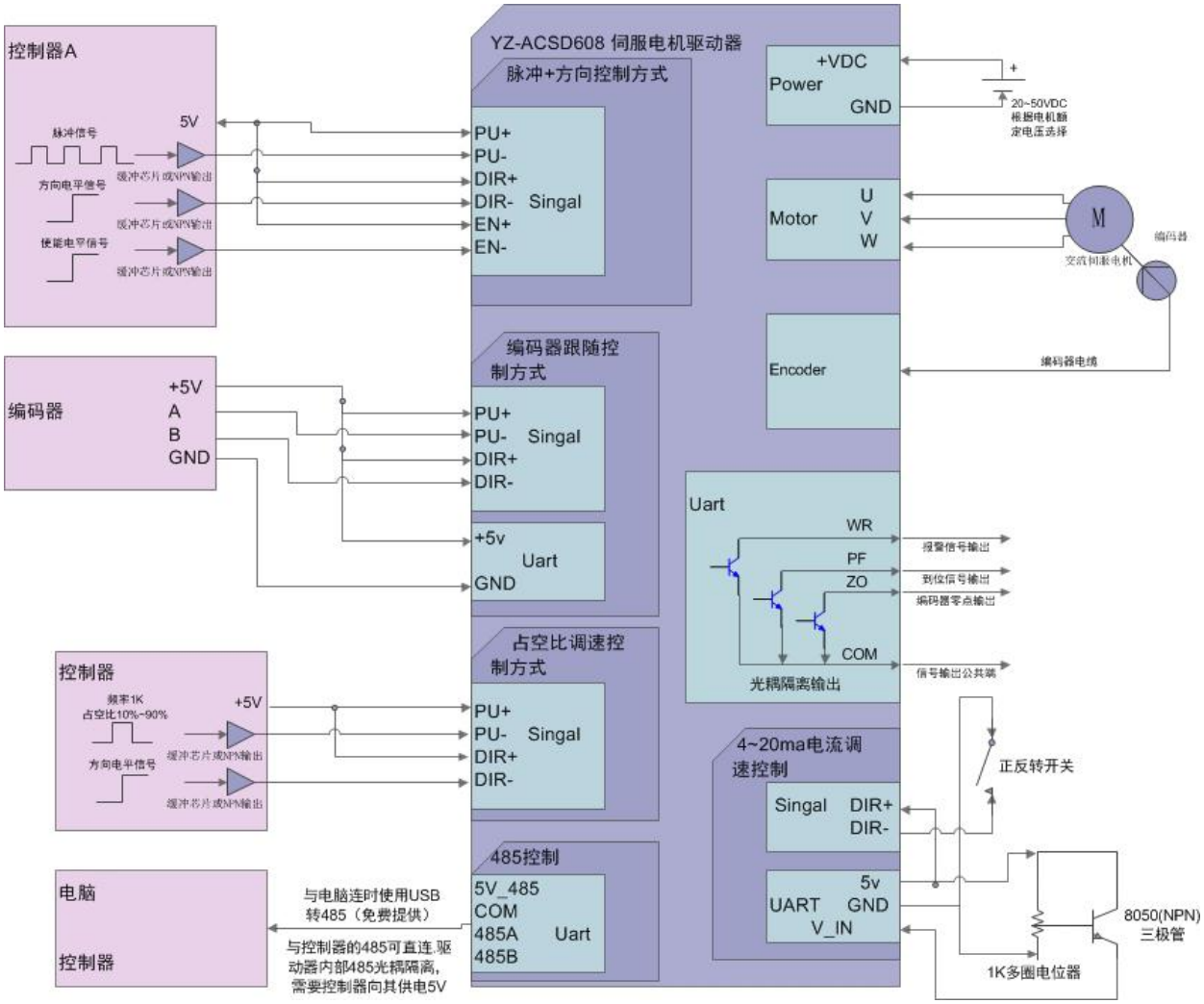
开机后红灯绿灯都亮一次，用于检验 LED 是否工作正常。而后绿灯亮，红灯灭为正常状态。如果遇到报警状态，可以通过红色闪烁来判断原因，也可以通过 modbus 读取报警代码。

报警代码	红灯闪烁	报警原因	报警处理
0x10	一长闪	系统高温报警 >70℃	继续运行
0x20	二长闪	写 flash 失败	继续运行
0x11	一长闪一短闪	系统过热报警 >90℃	停机 温度降至 70℃以下继续运行
0x12	一长闪二短闪	系统堵转报警	停机
0x13	一长闪三短闪	系统欠压报警 <20V	停机 电压超过 20V 继续运行
0x14	一长闪四短闪	失速报警，负载过重	停机

三．驱动器接线图与控制方式

1. 驱动器典型接线图

如下图所示，驱动器工作需要接上 20~50VDC 的电源，电压根据电机的额定电压来选择，电机的 UVW 和编码器已做好插头，直接插在驱动器上，如果需要加长线可做转接线加长。此版本提供光耦输出的 报警信号、到位信号、编码器零点信号，可以根据需要连接。下图有 5 个小框，分别是可以选择的 5 种控制方式，上电后只能选择其中一种控制方式，接线如下图所示。



2. 指令脉冲+方向位置控制模式

脉冲指令形式	CCW	CW	参数设定值
脉冲列符号	PUL DIR		指令脉冲+方向

一圈的脉冲数 = 编码器线数*4 / 电子齿轮

例如： 2500 线的编码器，电子齿轮为 4:1 ，

一圈的脉冲数 = 2500 * 4 / (4/1) = 2500

指令脉冲频率 = (需要电机运行的转速/60) * 一圈的脉冲数

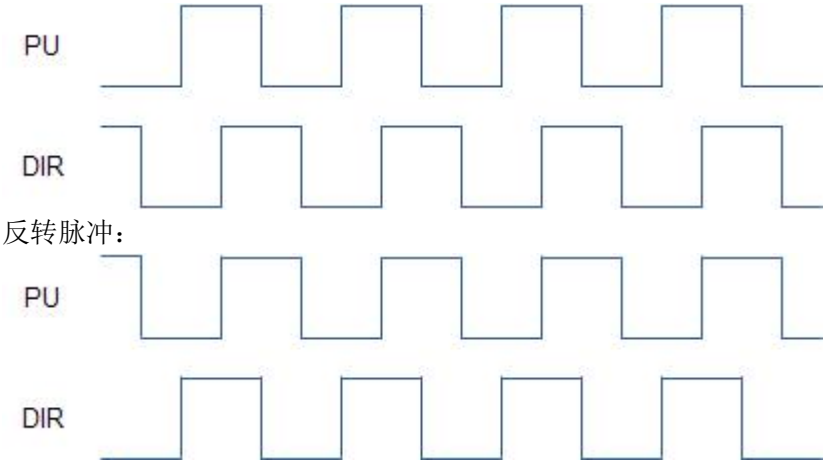
例如： 需要点击 3000RPM 一圈脉冲数为 2500

脉冲频率　= 3000/60 * 2500 = 125000 =125k

3. 正交指令脉冲位置控制模式

这种模式可以用于编码器跟随，如一个轴接了编码器，将编码器输出接到驱动器(接线方式如 驱动器典型接线图)，驱动器就能控制伺服电机，按输入编码器的信号，随动于控制的编码器。可以通过调节电子齿轮，来设置控制编码器和电机转动角度的比例。

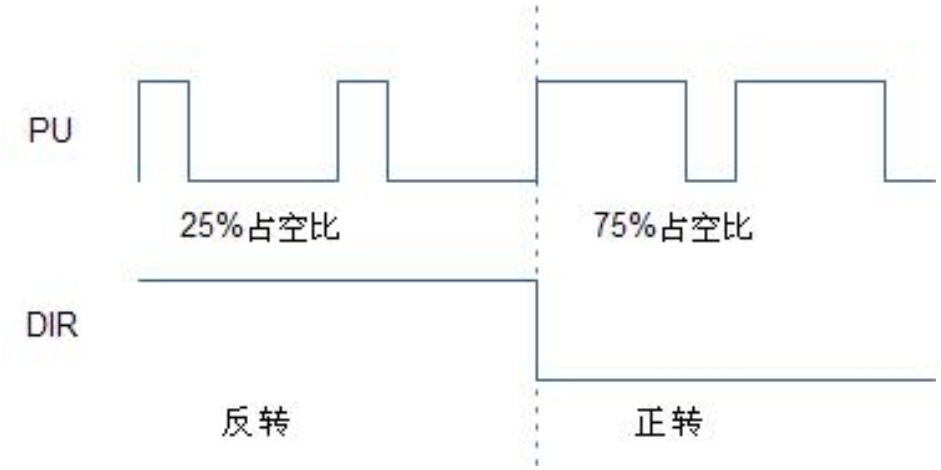
正转脉冲：



反转脉冲：

电机转动的方向：PU 上升沿超前 DIR 上升沿 为正转。PU 上升沿滞后 DIR 上升沿 为反转。

4. PWM 占空比速度控制模式



通过给 PU 的脉冲的咱空比来控制转速， 占空比转速范围 10%~90%代表 0~Max_Speed（Max_Speed 为位置模式保存的目标转速，通过设置这个参数，可以更精确的控制需要的转速，也不用担心出现超过所设定的速度）。给 PU 的频率为 1K~10K。

PU 占空比 = （目标转速/3000）* 80% + 10%

例如：需要转速 2000

PU 占空比 = (2000/3000) * 0.8 + 0.1 = 63.3%

5. 电压或电流信号速度控制模式

通过给 V-IN 和 GND 之间电压或电流信号以控制转速。4mA~20mA（或 0.6~3V）信号对应 0~Max_Speed（Max_Speed 为位置模式保存的目标转速，通过设置这个参数，可以更精确的控制需要的转速，也不用担心出现超过所设定的速度）。 如果需要电位器控制转速，按 驱动器典型接线图 中的接法即可。

四. 参数调试

根据电机所接负载不同，参数需要调整才能达到最佳效果。

1. 内部加减速曲线

根据控制器输出信号的不同来选择是否使用内部加减速曲线。

使用内部加速曲线：

当电机加速度小于 60000 时，驱动器会使能内部加减速曲线，具体加速度的大小就和设置的值相同。

使用场合：使用内部加速曲线，会产生滞后脉冲的现象，一些不需要实时跟随的场合，可以使用内部加速曲线。有些控制器，脉冲直接给到对应速度的频率，没有加减速的情况，就使用内部加减速曲线，可以降低控制器编程难度。

禁止内部加速曲线：

当电机加速度大于等于 60000 时，驱动器根据外部脉冲的加减速允许，内部加速度无效。

使用场合：例如雕刻机，控制器输出的脉冲就是有加减速的，就不需要驱动器内部的加速曲线，如果这个时候使用，会滞后于实际的脉冲。

2. 丝杆负载

首先介绍下扭矩，先用 400W 电机，1.3NM。负载是 5mm 螺距的丝杆，就是电机轴转一圈负载移动 5mm，这样的话，

负载等效力臂 = 5mm / 3.14 = 1.592 mm

那电机能提供的推力就是

经过丝杆传动的推力 = 1.3NM / （1.592mm*0.001）= 816 N

那能推动负载的重量就大约是 80KG，这个是垂直的，平推可以稍微大些。

由于丝杆负载电机转动一圈移动的距离较短，所以驱动器的参数 （加速度可以较大，如 20000，位置环 KP 可以较大，如 3000）。伺服电机最适合此种负载。

3. 皮带轮负载

伺服电机其实不是很适合接这种负载。因为皮带轮一般直径比较大，例如直径 30mm。那电机转一圈，负载移动的距离就是 30mm* π = 94.2，比上面说的丝杆 5mm 大了很多倍。那电机能提供的推力就是

经过皮带传动的推力 = 1.3NM /（30mm*0.001）= 43.3 N

那能推动负载的重量大约是 4.3KG. 所以伺服电机其实不适合接同步轮，因为同步轮转动一圈负载移动的距离太长，力臂长。如果这种场合要用伺服电机，可以选择直接尽量小的同步轮或通过电机轴接小同步轮，负载端接大同步轮，这样减速几倍，可以达到较好的效果。这种场合驱动器参数（加速度设置较小，如 5000， 位置环 KP 设置较小，如 1000），这样设置参数的目的是减小加速度和减速度，因为负载等效惯性大。

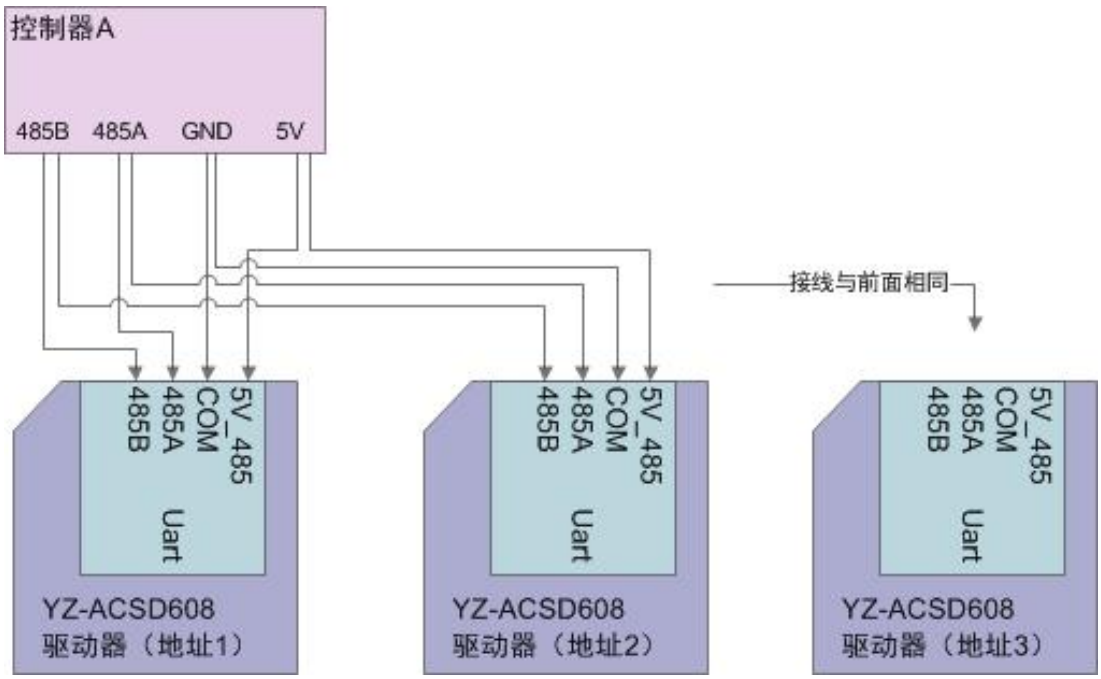
4. 圆盘负载

这种负载伺服无法直接带动，一般都需要接减速器。例如直径 200mm 重量 10KG 的圆盘。半径就是 100mm，重量等效半径就是 50mm。力臂很大。如果伺服要接此类负载，比较接减速器再接负载。

五． Modbus 控制方式

1. 硬件连接

驱动器内部 485 都通过光耦隔离，解决了一台主机连接多台从机容易被干扰和损坏的问题。



2. 寄存器说明

驱动器可以通过 modbus（RTU 模式）来控制驱动器。主机可以通过 modbus 的读写寄存器功能来设置驱动器参数和控制运行。驱动器支持的功能码为 0x3（读寄存器）、0x6（写寄存器）、0x78（写目标位置）、0x7a（修改设备地址）。

寄存器列表如下：

地址	参数名称	只读/读写	参数范围	参数说明
0	Modbus 使能	读写	0~1	0：modbus 禁止 1：modbus 使能
1	驱动器输出使能	读写	0~1	0：驱动器输出禁止 1：驱动器输出使能
2	电机目标速度	读写	0~3000 r/min	速度模式时，目标速度 位置模式时，最大速度
3	电机加速度	读写	0~65535 (r/min)/s	参数小于 60000 时，驱动器内部产生加减速曲线，参数大于 60000 时，驱动器内部不产生加减速脉冲

4	电机起始速度	读写	0~500 r/min	位置模式时，电机的最小转速(建议设置 0~10)
5	速度环比例系数	读写	0~10000	代表 0.0~10.0
6	速度环积分时间	读写	2~2000 ms	积分时间 2~2000ms
7	位置环比例系数	读写	60~5000	位置模式下速度下降的速率
8	速度前馈	读写	0~8.0V/KRPM	速度前馈每 1KRPM 对应的电压幅值
9	DIR 极性	读写	0~1	0：外部 DIR 不导通顺时针旋转 1：外部 DIR 导通顺时针旋转
10	电子齿轮分子	读写	0~65535	16 位电子齿轮分子
11	电子齿轮分母	读写	1~65535	16 位电子齿轮分母
12	目标位置低 16 位	只读		需要走步数的高 16 位
13	目标位置高 16 位	只读		需要走步数的低 16 位
14	报警代码	只读		
15	系统电流	只读	0~32767	实际电流为 x/2000(A)
16	电机当前速度	只读	-30000~30000 r/min	实际电机转速=电机当前速度/10
17	系统电压	只读	0~32767	实际电压为 x/327（V）
18	系统温度	只读	0~100	摄氏度
19	系统输出的 PWM	只读	-32768~32767	代表-100%~100%
20	参数保存标志	读写	0~1	0：参数未保存 1：保存参数中 2：保存完毕
21	设备地址	只读	0~255	设备地址
22	绝对位置低 16 位	读写		走过步数的高 16 位
23	绝对位置高 16 位	读写		走过步数的低 16 位
24	速度滤波频率	读写	1~2000	默认 100
25	位置前馈	读写	0~100	如果控制器输出脉冲已经包含加减速曲线(如雕刻机系统)，这个参数设置为 100,。 如果使用驱动器内部加减速，则设置成 0。

3. Modbus 通信格式

a. modbus 主机读取数据及从机应答格式 （功能码 03)

主机读取数据 格式							
设备地址	功能码	第一个寄存器的高位地址	第一个寄存器的低位地址	寄存器个数高位	寄存器个数低位	CRC 高位	CRC 低位
0x01	0x03	0x00	0x00	0x00	0x01	0x84	0x0a

从机应答						
设备地址	功能码	数据长度	第一个数据高字节	第一个数据低字节	CRC 高位	CRC 低位
0x01	0x03	0x02	0x00	0x01	0x79	0x84

串口接收到的数据都是无符号数，如果寄存器是有符号数，发送的则是二进制补码的格式，转换成有符号数的算法如下（VB 代码）：

```
If modbus.data(11) > 32767 Then
    disp_modbus_data.PU = (modbus.data(11) - 32768) * 65536 + modbus.data(10)
    disp_modbus_data.PU = -((&H7FFFFFFF - disp_modbus_data.PU) + 1)
Else
    disp_modbus_data.PU = dmodbus.data(11) * 65536 + modbus.data(10)
End If
```

注: modbus.data(11)为目标位置高 16 位　modbus.data(10)为目标位置低 16 位

b. modbus 主机写数据及从机应答格式 (功能码 06)

主机写数据　格式							
设备地址	功能码	第一个寄存器的高位地址	第一个寄存器的低位地址	数据高位	数据低位	CRC 高位	CRC 低位
0x01	0x06	0x00	0x00	0x00	0x01	0x48	0x0a

从机应答　格式							
设备地址	功能码	第一个寄存器的高位地址	第一个寄存器的低位地址	数据高位	数据低位	CRC 高位	CRC 低位
0x01	0x06	0x00	0x00	0x00	0x01	0x48	0x0a

c. modbus 主机写脉冲数 （功能码 0x10）

主机写双字节数据 （写 PU 脉冲数）						
设备地址	功能码	第一个寄存器的高位地址	第一个寄存器的低位地址	寄存器个数高位	寄存器个数低位	数据长度
0x01	0x10	0x00	0x0c	0x00	0x02	0x04
PU:8~15 位	PU:0~7 位	PU:24~31 位	PU:16~23 位	CRC 高位	CRC 低位	
0x27	0x10	0x00	0x00	0xf8	0x8b	

脉冲数是有符号数，一个负数(假设此数为 X)转换成 32 位 16 进制数的算法如下(vb 代码)：

```
If X < 0 Then
   X = &H7FFFFFFF + (X + 1)
   PU24_31 = Fix(X / (256 * 65536)) + &H80
Else
   PU24_31 = Fix(X / (256 * 65536))
End If
   PU16_23 = Fix(X / 65536) mod 256
```

PU8_15 = Fix(X / 256) mod 256

PU0_7 = X mod 256

注：fix() 为取整函数

从机应答 格式							
设备地址	功能码	第一个寄存器的高位地址	第一个寄存器的低位地址	寄存器个数高位	寄存器个数低位	CRC 高位	CRC 低位
0x01	0x10	0x00	0x0c	0x00	0x02	0x81	0xcb

d. modbus 主机写脉冲数 （特殊功能码 0x78）

主机特殊功能码 0x78 格式 （写 PU 脉冲数）							
设备地址	功能码	PU:24~31 位	PU:16~23 位	PU:8~15 位	PU:0~7 位	CRC 高位	CRC 低位
0x01	0x78	0x00	0x00	0x27	0x10	0xbb	0xfc

从机应答 格式							
设备地址	功能码	PU:8~15 位	PU:0~7 位	PU:24~31 位	PU:16~23 位	CRC 高位	CRC 低位
0x01	0x78	0x27	0x0e	0x00	0x00	0xca	0xb7

4. CRC 校验示例代码

```
unsigned short CRC16(puchMsg, usDataLen)
unsigned char *puchMsg ; /* 要进行 CRC 校验的消息 */
unsigned short usDataLen ; /* 消息中字节数 */
{
unsigned char uchCRCHi = 0xFF ; /* 高 CRC 字节初始化 */
unsigned char uchCRCLo = 0xFF ; /* 低 CRC 字节初始化 */
unsigned uIndex ; /* CRC 循环中的索引 */
while (usDataLen--) /* 传输消息缓冲区 */
{
uIndex = uchCRCHi ^ *puchMsgg++ ; /* 计算 CRC */
uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex} ;
uchCRCLo = auchCRCLo[uIndex] ;
}
return (uchCRCHi << 8 | uchCRCLo) ;
}
/* CRC 高位字节值表 */
static unsigned char auchCRCHi[] = {
```

0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40
} ;
/* CRC 低位字节值表*/
static char auchCRCLo[] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9,
0x09, 0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F,
0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13,
0xD3, 0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6,
0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA,
0x3A, 0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA,
0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6,
0x26, 0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3,
0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF,
0x6F, 0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79,
0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74,
0x75, 0xB5, 0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0, 0x50,
0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54,
0x9C, 0x5C, 0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58,
0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D,
0x4C, 0x8C, 0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41,
0x81, 0x80, 0x40 }

5. modbus 方式主机控制过程

a：位置模式

（1）写 设备 1（功能码 0x06　　地址 0　　Modbus 使能）　　1

（2）读 设备 1（功能码 0x03 地址 16 报警代码） （读到 0 时为运行正常）

（3）写 设备 1（功能码 0x06 地址 3 电机目标速度） 2000

（4）写 设备 1（功能码 0x10 地址 22 电机绝对位置） 1000

（5）读 设备 1（功能码 0x10 地址 22 电机绝对位置）

读取 绝对位置以查询是否走到位

（6）重复（4）（5）过程，控制电机位置。

注：

电机的目标速度为电机运行的最大速度，如果目标位置较小，电机可能达不到最大速度即停止。加速过程

绝对位置是允许+-2 的误差，读到绝对位置是给定位置的+-3 范围内就可以写下一个值。

当系统需要重新定绝对位置零点时，需要先将电子齿轮分子设置成 0，然后电机绝对位置写 0，此时绝对位置会变 0，而电机位置不变。

b：速度模式

（1）写 设备 1（功能码 0x06 地址 0 Modbus 使能） 1

（2）读 设备 1（功能码 0x03 地址 14 报警代码） （读到 0 时为运行正常）

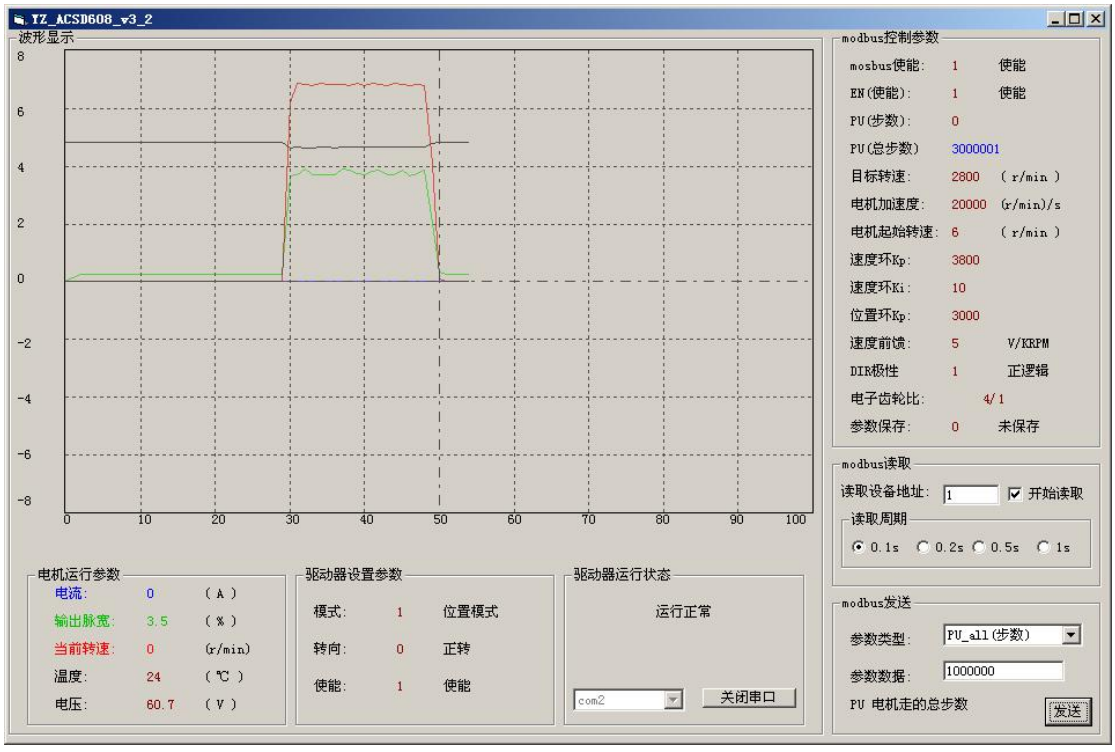
（3）写 设备 1 （功能码 0x06 地址 3 电机加速度） 6000

（4）写 设备 1 （功能码 0x06 地址 2 电机目标速度） 2000

（5）重复（4）过程，控制电机转速。

六．上位机软件使用说明

本驱动器提供一个上位机软件，用于监测和测试驱动器。可以通过软件查看和设置驱动器内部参数。



如上图所示，软件分为波形显示，电机运行参数等几个部分。下面介绍一下各个部分的功能和作用。

波形显示：一共有 4 个通道，分别用 4 种颜色表示。颜色和 电机运行参数 内的字体颜、色相同。即：蓝表示电流，绿表示输出的脉宽，红表示当前转速，黑表示电压。

电机运行参数：表示电机运行的实时数据。

驱动器设置参数：显示驱动器的拨码开关，和方向使能设置。如果是 modbus 模式，此栏无效。

驱动器运行状态：此栏会显示驱动器的报警状态，如果没有报警会显示运行正常。

Modbus 控制参数：此栏内的参数是驱动器内部的参数，如果要修改这些参数，必须先对 modbus 使能写 1。具体的参数含义参考 寄存器说明。

Modbus 读取：此栏可设定驱动器的地址，读取驱动器数据的周期，和是否读取。

Modbus 发送：此栏用于修改驱动器参数，首先选定参数类型，再设定好参数数据，然后点发送即可。

七. 常见问题处理

1. 如果脉冲控制端口 DIR 导通方向和我所需要的方向不同怎么改？

答：可以通过上位机来设置 DIR 的极性，接上上位机后，modbus 使能写 1 ， DIR 极性 写 0. 最后参数保存标志写 1. 重新上电即可。

2. 上电驱动器不报警，给运行脉冲后驱动器报警，怎么回事？

答：检查接插件是否松动，电机 UVW 和编码器线任何一根断或顺序交换都会无法正常运行。

3. 怎么通过上位机控制运行

答：连接好电机，驱动器上电，通过提供的 USB 线连接驱动器和电脑，根据“上位机使用说明”设置好，使得上位机能读取到数据后。 Modbus 使能 发送 1。然后 PU 步数 发送需要走的距离。