


# 高 品 变 频 驱 动 器

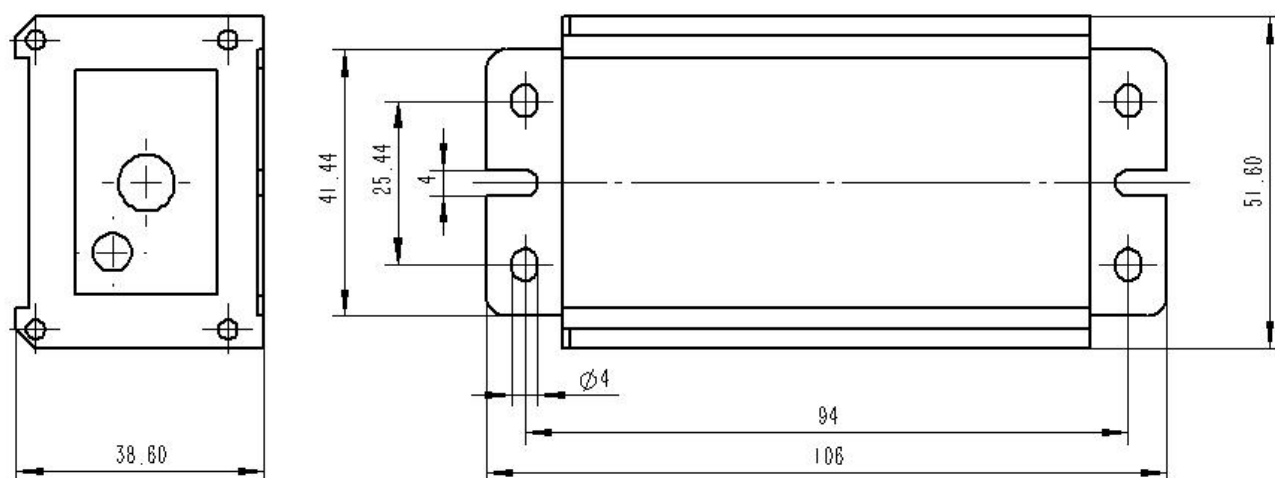
12-48VDC 使用说明书

 使用本公司驱动器前，请先阅读该使用说明书，以免出现意外，如有任何疑问，请与我司联系。

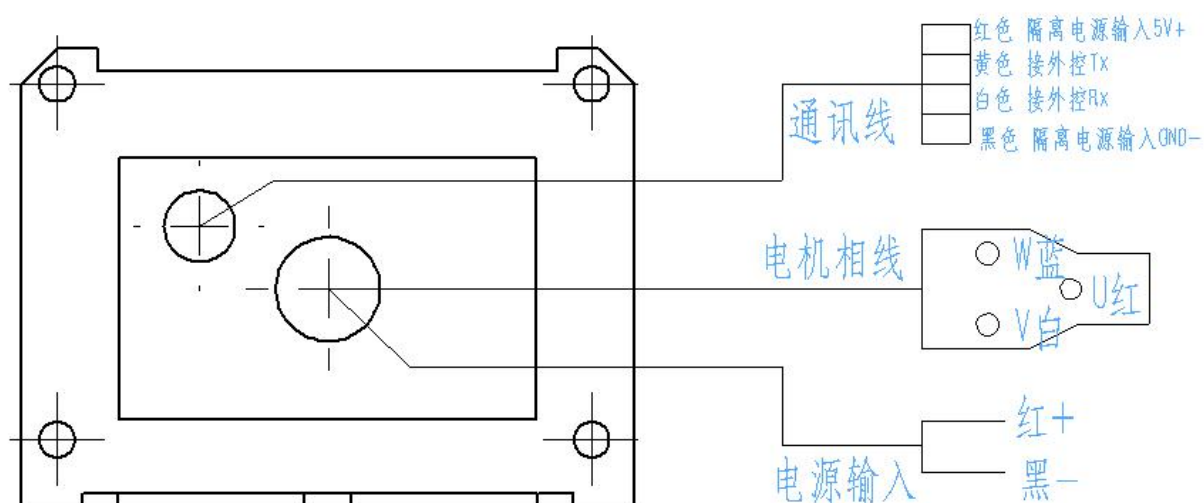
## 一、 主要电气参数

项目	GP4804C	GP2408C	GP1212C
适用对象	高品 48V 压缩机	高品 24V 压缩机	高品 12V 压缩机
输入最大电流	6.5A	13A	17A
最大输入功率	280W	250W	180W
最小转速	1800rpm	1800rpm	1800rpm
最大转速	6000rpm	6000rpm	4800rpm
压缩机转速精度	60rpm	60rpm	60rpm
输入电压(额定)	48VDC	24VDC	12VDC
输入电压范围	DC 32V-60V	DC 18V-30V	DC 8V-16V
变频器效率	>90%		
冷却方式	铝壳自然冷却 保持通风		
工作环境	温度：-20℃ ~ 55℃ 湿度：30% ~ 90%		
保护功能	过、欠压保护		
	电流保护		
	过电流保护		
	散热器过热保护		
	缺相保护		
保护动作	过压保护功能		
	欠压保护功能		
	电流保护功能		
	散热器过热保护		
	当发生故障时，系统在停机后 3 分钟后重启		
驱动方式	正旋波矢量驱动		
接口方式	隔离 UART 通讯		
电控盒尺寸	106mmx51.6mmx38.6mm		

## 二、 驱动器几何尺寸



## 三、 驱动器接口定义:



## 四、 通信控制功能

### 4.1 驱动器控制

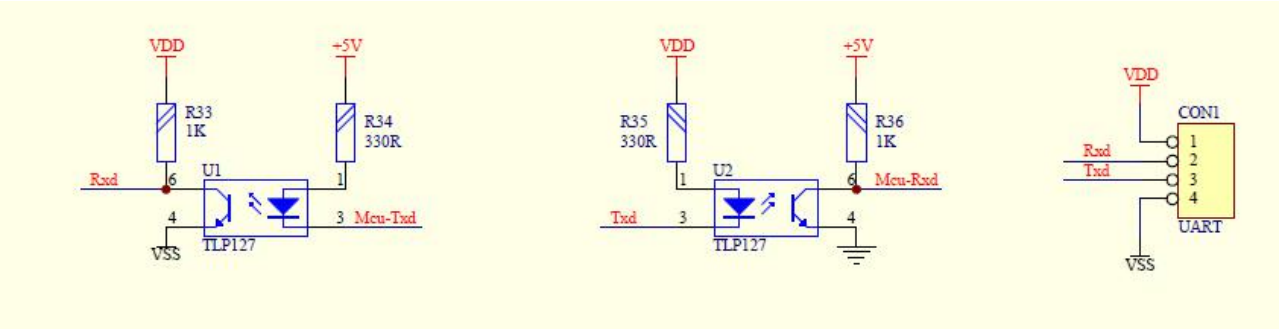
1, 发送启动指令到驱动器, 无论设定速度多少, 驱动器初始化启动的速度为 3000rpm, 稳定在 3000rpm 速度 1 分钟后, 速度再逐步自动闭环到设定速度值。

2, 发送停机指令, 如果当前速度大于 3000rpm, 则驱动器先减速, 低于 3000rpm 后压缩机停机工作, 如果当前速度小于 3000rpm, 则驱动器直接停机。

3, 再次启动, 驱动器的运行间隔是 2.5 分钟, 如果没有间隔小于该时间, 则驱动器处于等待状态, 到达该间隔时间后再进行启动判断。

4, 驱动器发生故障后, 除电流故障外都会自动重启 5 次, 如果大于 5 次则停机等待电源复位。重新发送启动指令可以清楚除电流故障外的其他故障。

4.2 隔离通信电路：



4.3 通信协议

驱动器接受 TTL 电平的 RS232（UART）通讯方式。

波特率：9600bps

数据格式：1 位起始位，8 位数据，1 位停止位

驱动器每 1s 钟自动发送一帧数据给上位机，具体的格式如下表。

驱动器在运行状态中会检测是否有指令发送过来，超过 2 分钟没有收到正确的数据则自动停机，因此在控制驱动器运转时候，最长时间指令间隔不能少于 2 分钟。

上位机控制发出内容：

0	0xAA	起始码
1	0X00	地址
2	指令	0：关机 1：开机
3	设定转速	低字节
4	设定转速	高字节
5	0x00	
6	0x00	
7	0x00	
8	0x00	
9	0x00	
10	0x00	
11	0x00	
12	0x00	
13	0x00	
14	校验和	
15	0x55	结束码

驱动器发送出内容：

0	0xAA	起始码
1	0X01	从机地址
2	压缩机转速	低字节
3	压缩机转速	高字节
4	压缩机电流	低字节，精度为 0.1A
5	压缩机电流	高字节

6	母线电压	低字节，精度为 0.1v
7	母线电压	高字节
8	驱动器温度	20，35-85/每 10 度 1 级
9	故障代码	
10	工作状态	1：工作中；0：待机中
11	0x00	
12	0x00	
13	0x00	
14	校验和	
15	0x55	结束码

其中校验和为：chkck=sum（字节 1，....，字节 13）；取低 2 位值，最大 255

#### 4.4 接受程序分析代码示例：

//本程序是 Arduino 的示例程序，其他 MCU 请移植。

```
void Receive_command(){
    unsigned char inByte,temp;
    if(Serial.available() > 0) {
        inByte = Serial.read();
        if(inByte==0xAA){Receive_item=0;}
        Receive_data_temp[Receive_item]=inByte;
        Receive_item++;
        if(Receive_item>15){
            Receive_item=0;
        }

        if(inByte==0x55){
            temp=0;
            //对接受的数据进行累计校验
            for(int i=0;i<14;i++){
                temp += Receive_data_temp[i];
            }

            if(temp==Receive_data_temp[14]){
                for(int j=0;j<16;j++){Receive_data[j]=Receive_data_temp[j];}
                //如果接受到的数据正确则 copy 到 Receive_data 数组中
            }
            Uart.Receive_flag=1;
        }
    }
}
```

#### 4.5 驱动器故障代码：

故障号	故障说明	备注
1	软件检测电流故障	电流过大
2	硬件检测电流故障	电流过大
3	输入电压低于设定规范	
4	输入电压高压设定规范	
5	缺相错误	
6	电机堵转	
7	启动失败	
17	通信故障	