无感工业风机调试手册

三相电机控制 MCU
FU6812L2

峰岹科技(深圳)股份有限公司

目录

1 概述

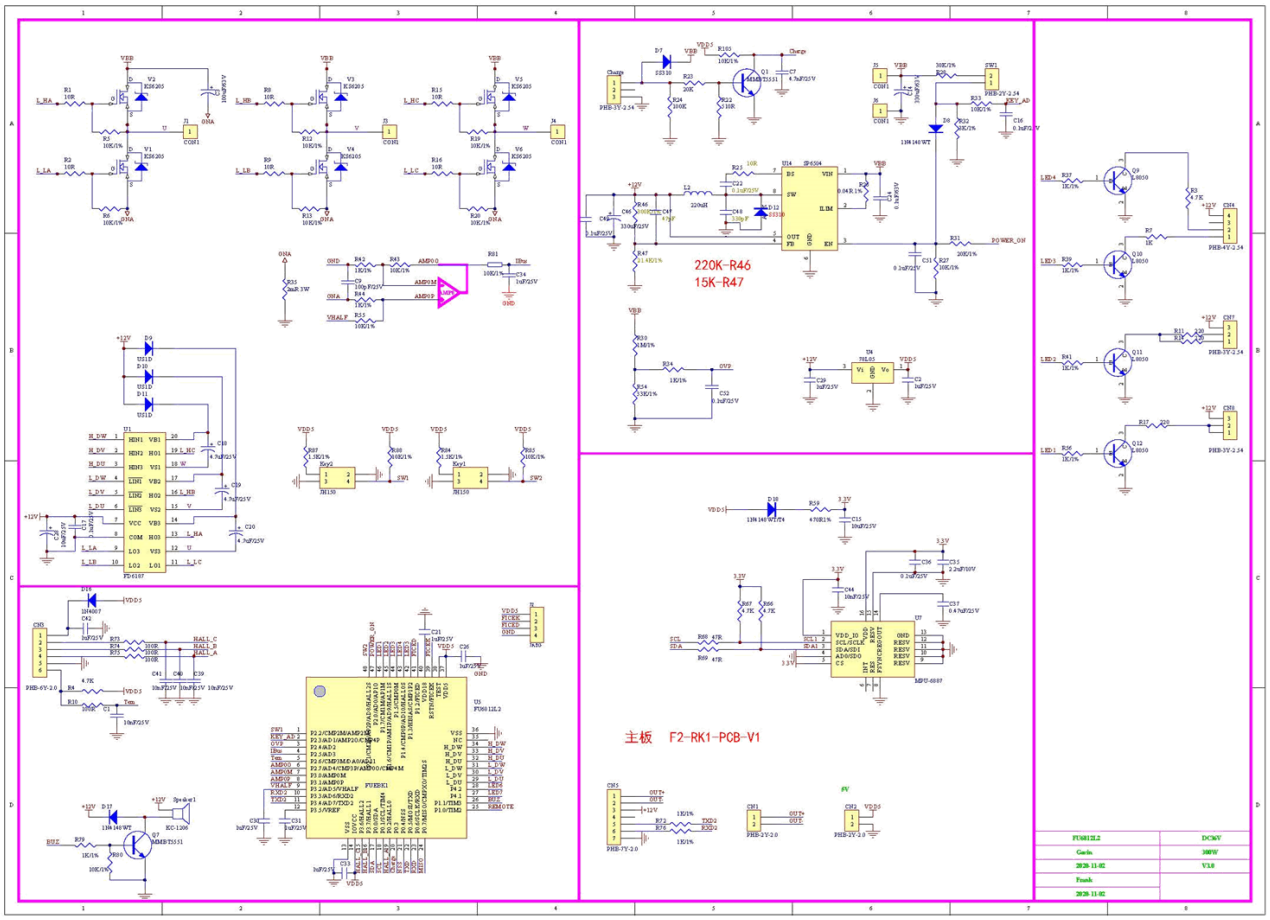
本调试手册详细介绍了如何使用峰岹科技的 FU6812L2 芯片，对低压直流无刷工业风机电机实现无霍尔的 FOC 驱动控制。阅读手册时，第二章节硬件原理跟第三章节软件原理可以大致先浏览一遍，重点放在第四章调试步骤。

涉及的软/硬件

软/硬件和模块	名称	章节	备注
软件	FU6812L2 工业风机应用方案标准程序	全部	调试需在该工程软件上进行
硬件		全部	调试需在该硬件上进行

2 硬件原理与参数配置

2.1 硬件原理



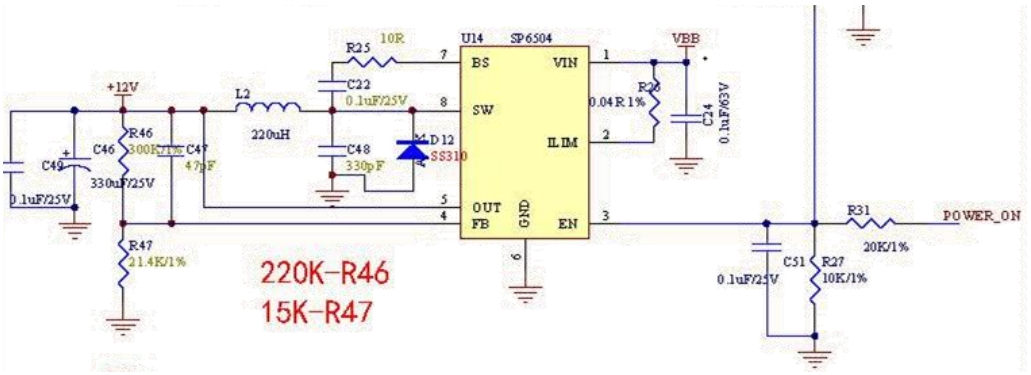
使用方式:

该板子作为工业风机的驱动板，直接上电即可使用。

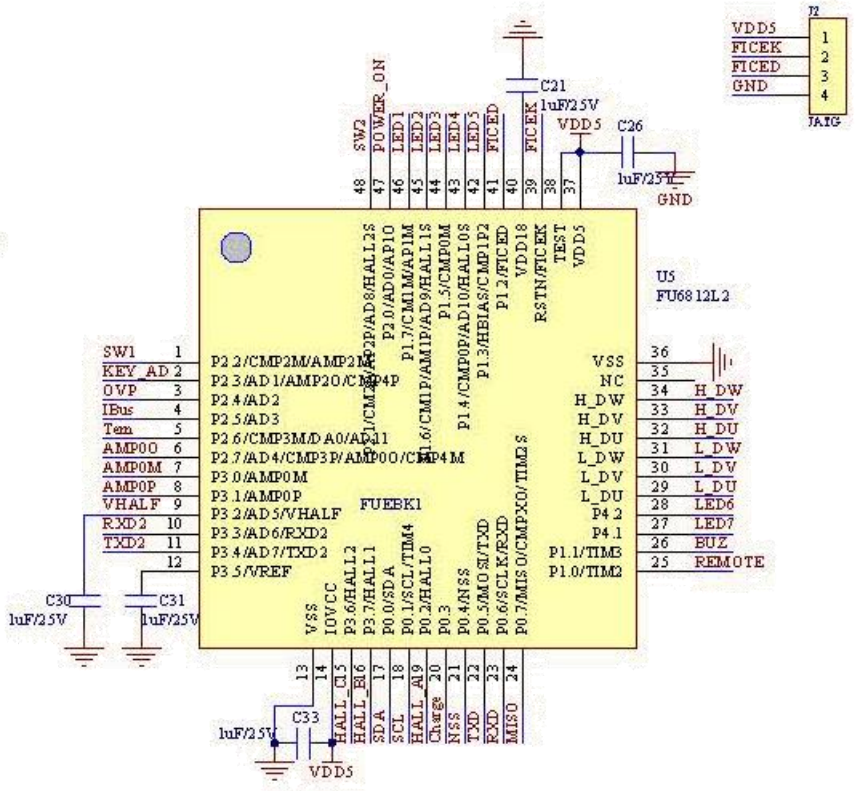
注意事项:

根据具体电机电压和电流大小，合理配置母线电压比，运放放大倍数，采样电阻，反电动势检测电路分压比。

2.1.1 电源部分



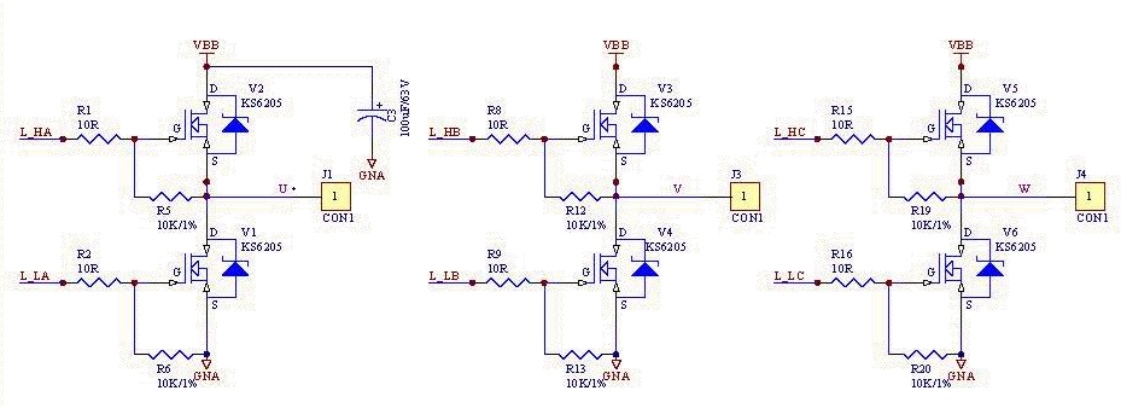
2.1.2 芯片主体



使用方式:

FU6812L2 应用于中高压 6-NMOSFET 驱动应用。

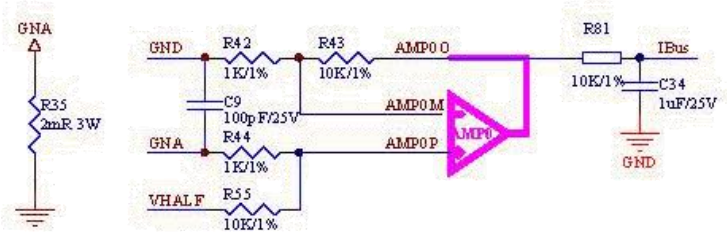
2.1.3 功率驱动部分



注意事项:

最大电流情况下，采样电阻功率不能超过额定功率的 80%。

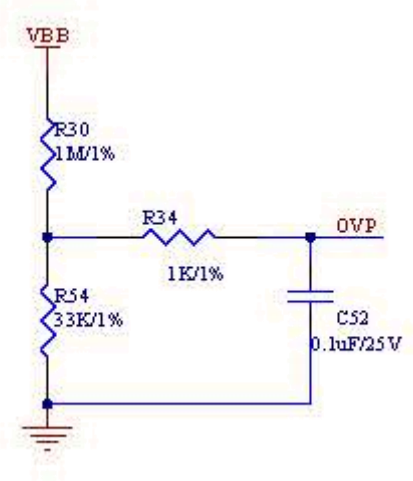
2.1.4 运放配置电路



注意事项:

1. C9 参数不可调整，精度要求 10%；
2. R42、R43、R44、R55 需要用 1%精度电阻；
3. 放大倍数 = R43/R42 = R55/R44；
4. 最大采样电流 = (VREF – 2.5)/放大倍数/采样电阻值。

2.1.5 母线电压采样



注意事项:

1. R27、C23 参数不可调整；
2. R25、R30 需要用 1%精度电阻；
3. 最大采样电压 = (R25 + R30)/(R30)*VREF；
4. 最大采样电压一般选择为 2 倍的最大应用电压，OVP 此处的电压需要低于 0.8*VREF。

3 软件原理

3.1 电机状态机流程图

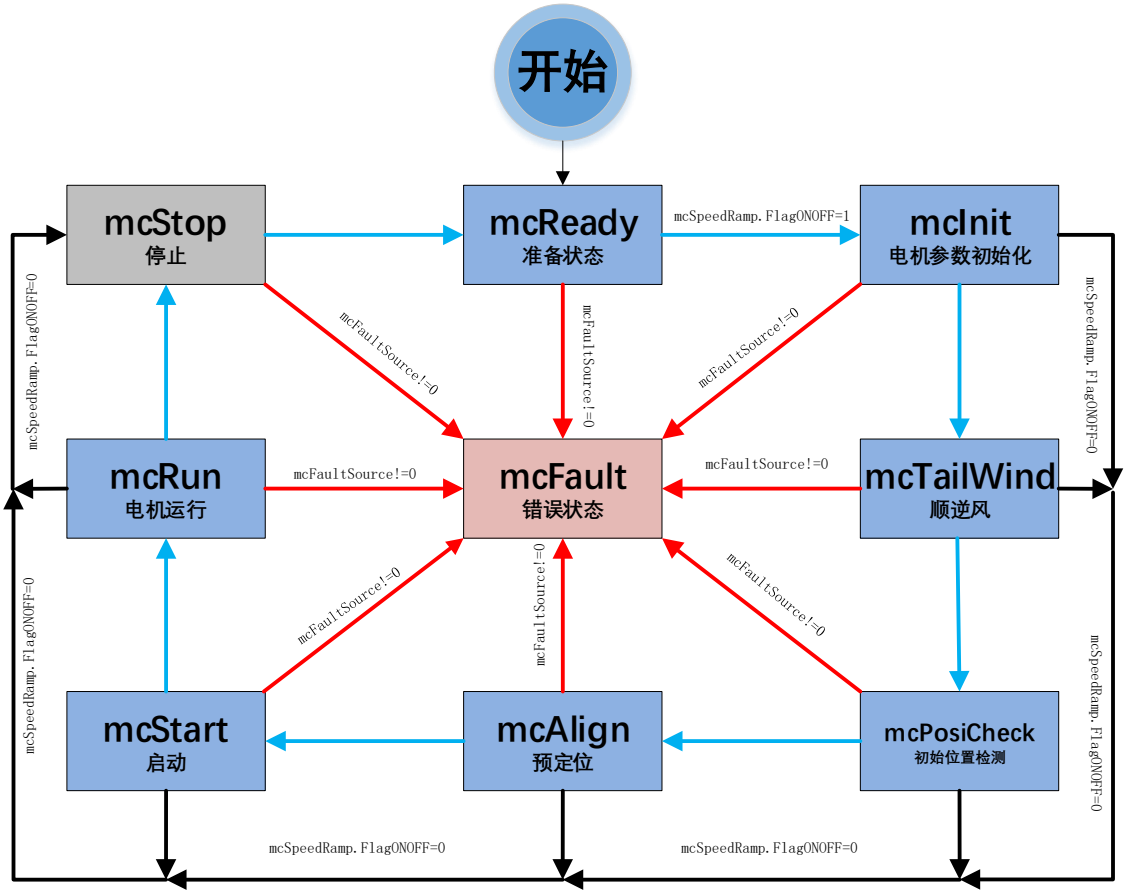


图 3-1 电机状态机流程图

如图所示，电机状态机分为三条路径:

1. 运行: mcReady -> mcInit -> mcTailWind -> mcPosiCheck -> mcAlign -> mcStart -> mcRun；
2. 停机: mcInit、mcTailWind、mcPosiCheck 、mcAlign、mcStart、mcRun 状态下如果检测到关机信号则会切入到 mcStop 状态进行降速关机；
3. 故障: 所有状态下发生故障均会跳转至 mcFault 状态，在 mcFault 状态将不再进行故障检测，因此不支持多故障并发的同时上报。

说明:

1. mcReady: 准备状态，等待开机命令，如果开机使能则跳转到 mcInit 状态；
2. mcInit: 相关变量和 PI 初始化，关闭电流，母线采样的外部 ADC 触发，然后跳转到下一状态；
3. mcTailWind: 顺逆风检测状态，检测到顺风时，直接切到 mcRun 状态运行；检测到逆风时，先刹车再往下执行(工业风机没有逆风的情况)；检测到静止时，往下执行；

4. mcPosiCheck: 初始位置检测状态，检测电机的初始位置，再正常启动；
5. mcAlign: 预定位状态，该状态下控制器输出恒定的电流将电机强行拖动到固定的角度上。定位结束则跳入下一个状态 mcStart；
6. mcStart: 启动状态，该状态主要用于电机的启动代码配置，对相关寄存器代码与变量进行配置之后则转入下一个状态 mcRun。电机启动过程由 ME 内核实现；
7. mcRun: 运行状态，该状态包含: 电机启动阶段,电机运行阶段，电机速度的控制在该状态进行；
8. mcStop: 停机状态，该状态用于停机操作，当速度降低到比较低的转速之后关闭输出，切入到 mcReady 状态，等待新的开机命令；
9. mcFault: 错误状态，当发生保护时，程序会记录错误源并且状态机会跳转到错误状态关机保护，当错误源被清掉时，会切入到 mcReady 状态，等待新的开机命令。

注意事项:

1. 电机状态机一共分为 9 个状态，状态之间只允许固定的状态跳转 例如: mcReady 状态只能向 mcInit 和 mcFault 状态跳转；
2. 特别的，mcTailWind，mcAlign 两个状态都有使能位，当没使能时，直接跳转到下一个状态。例如: mcAlign 没使能时，mcPosiCheck 直接跳转到 mcStart 状态。

3.2 程序流程图

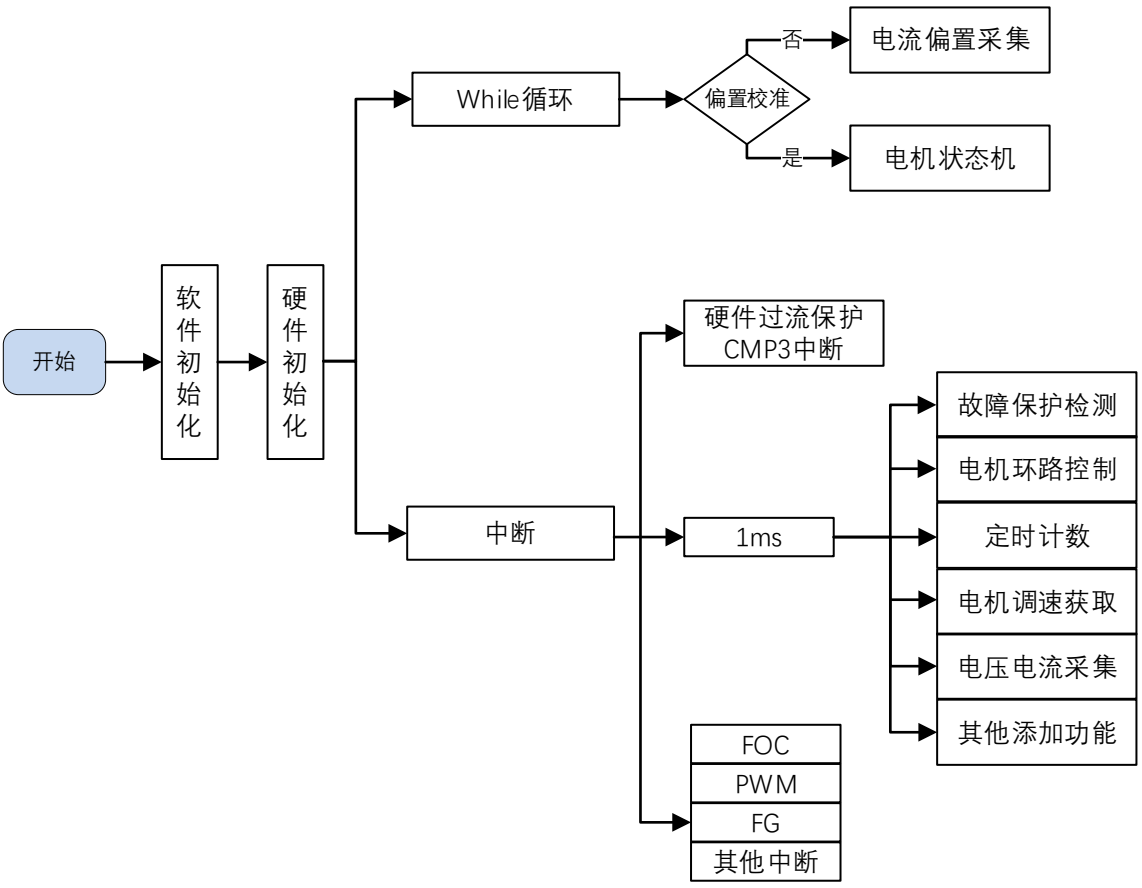


图 3-2 程序执行流程图

3.3 程序说明

3.3.1 Main 函数:

程序初始化 -> 偏置电压检测 GetCurrentOffset() + 电机运行控制 MC_Control() + 1ms 定时处理 TickCycle_1ms()等；

3.3.2 1ms 定时中断

产生 1ms 标志位，通过该标志位在主函数中调用 TickCycle_1ms()函数，TickCycle_1ms()函数主要包含了调速、故障保护检测、母线电流、母线电压采集等相关处理，调用子函数如下:

```
Speed_response();              // 环路控制函数
Fault_Detection();             // 故障检测
LED_Display ();                // LED 故障警报提示
```

```
TargetRef_Process ();          //调速接口
ATORamp ();                    // 电机启动 ATO 爬坡控制
FGOutput ();                   // FG 输出
CloseLoop_Parameter ();  //闭环运行后相关参数更新处理
```

3.3.3 FOC 中断

FOC 中断，即载波中断，主要处理一些时序比较快的程序，如调用除法器、过流处理等。

3.3.4 CMP3 中断

比较器 3 中断主要是处理硬件过流保护，具体原理可参考章节 5.2.1。

3.3.5 Timer3 中断

Timer3 中断主要是 PWM 占空比的获取，通过该中断获取到 PWM 的高电平 TIM3__DR 跟 PWM 的周期值 TIM3__ARR，之后再通过计算算出 PWM 的占空比大小。

4 调试步骤

4.1 配置电机参数

4.1.1 电机参数

1. 电机极对数 Pole_Pairs；
2. 电机的相电阻 RS、相电感 LD、LQ，以及反电动势常数 Ke；
3. 电机速度基准，速度基准 MOTOR_SPEED_BASE = 2*电机额定转速。

4.1.2 电机参数测量方法

1. 极对数 Pole_Pairs: 电机设计时需给出的参数；
2. 相电阻 Rs: 万用表或者电桥测量电机两相线电阻 RL，相电阻 Rs = RL/2；
3. 相电感 Ls：电桥测 1KHz 频率下的两相线电感 LL，相电感 Ls = LL/2；LD = LQ = Ls；
4. 反电动势常数 Ke: 示波器的探头接电机的一相，地接电机另外两相中的某一相，转动负载，测出反电动势波形。取中间的一个正弦波，测量其峰峰值 KeVpp 和周期 KeT。计算公式如下：

$$Ke = Pole\_Pairs * \frac{KeVpp * KeT}{207.846}$$

示例，测量反电动势波形如下：

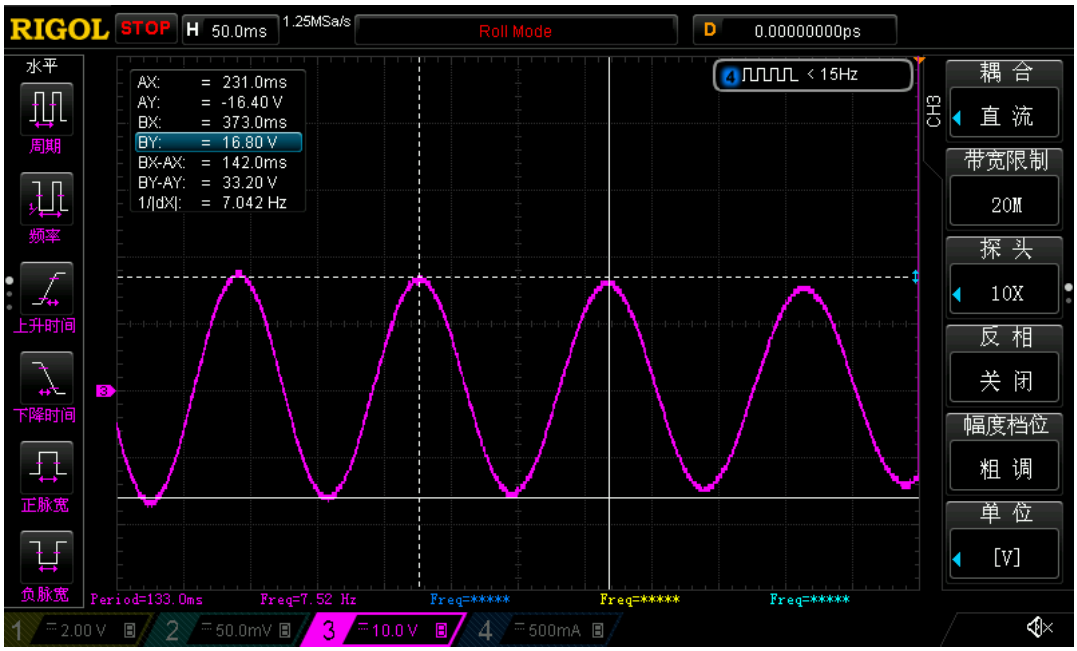


图 4-1 反电动势波形

测量峰峰值 KeVpp 为 33.2V，周期 KeT 为 142ms，极对数 Pole_Pairs 为 4，则：

反电动势$\mathrm{Ke} = 4 * \frac{33.2*142}{207.846} = 90.73$

5. 速度基准 MOTOR_SPEED_BASE: 速度基准一般设置为电机最大转速的 2 倍左右，该值会影响启动等性能，一般需要提前定好之后，后面不要轻易改动。

4.1.3 对应程序

Customer.h

```
/* ------------------------------------------------------------------------------------------------------
                                        电机参数值配置
------------------------------------------------------------------------------------------------------ */
#define R                               (1.0)                           //相电感对应系数值
#define Pole_Pairs                      (4.0)                           // 极对数
#define RS                              (1.55)                          // 相电阻,测量两根相线之间电阻/2,单位:Ω
#define LD                              (0.00279*R)                     // D轴相电感,测量两根相线之间电感/2,单位:H
#define LQ                              (0.00279*R)                     // Q轴相电感,测量两根相线之间电感/2,单位:H
// 若选择AO自适应观测器 则无需填写Ke
#define KeVpp                           (6.7)                           // 反电动势测量的峰峰值,单位:V
#define KeT                             (34.2)                          // 反电动势测量的周期,单位:ms
#define Ke                              (Pole_Pairs * KeVpp * KeT / 207.846)    // 反电动势常数,单位:V/KRPM

#define MOTOR_SPEED_BASE                (15000.0)                       // (RPM) 速度基准,建议电机空载最大转速的2倍
```

4.2 确认芯片内部相关数据配置

Customer.h

```
/* ------------------------------------------------------------------------------------------------------
                                        芯片内部相关参数配置
------------------------------------------------------------------------------------------------------ */
/*PWM Parameter*/
#define PWM_FREQUENCY                   (20.0)                          // (kHz) 载波频率

/*deadtime Parameter*/
#define PWM_DEADTIME                    (1.0)                           // (us) 死区时间

/*single resistor sample Parameter*/
#define MIN_WIND_TIME                   (PWM_DEADTIME + 1.0)            // (us) 单电阻最小采样窗口，建议值死区时间+0.9us，不能>载波周期/16!!!

/*正反转设置   CW:正转 CCW:反转*/
#define FRMODE                          (CCW)
```

注意事项:

1. 载波频率一般需要设置为最大电周期 10 倍左右，载波频率会影响启动，MOS 温升等等，调试之前需要选择好合适的载波频率。工业风机一般默认 20K 即可；
2. 死区大小根据实际的 MOS 开关速度设置，保证没有直通风险；
3. 最小采样窗口设置，最小窗口最小需要大于 2 倍的死区，小于载波周期的 1/16，即 1000/16/PWM_FREQUENCY > MIN_WIND_TIME > 2*PWM_DEADTIME；
4. 正反转设置，根据实际接线设置，如果电机反转了，则直接修改 FRMODE 配置即可。

4.3 确认硬件参数

1. 通过电机的电压范围和功率范围确认母线分压比、采样电阻值、放大倍数。
2. 电阻阻值跟放大倍数选取规则:
   1) 母线分压电阻:
      - 分压比不宜太小: 一般建议最大采集电压为 0.8*VREF，如某电机的最大电压为 30V，ADC 基准 VREF 为 4.5V，此时分压比建议不低于: 30/0.8/4.5 = 8.33；如果分压比太小，如分压比为 5，则 30V 时，经过分压后到 AD 口的电压为 6V，此时溢出了。

- 分压比不宜太大: 分压比太大的话会导致 AD 采集电压精度不够，如最大电压为 30V，当分压比为 40 时，经过 AD 口的电压为 30V/40V = 0.75V，28V 时为 0.7V，此时精度比较低，而且 AD 还有 4.5 - 0.75 = 3.75V 的余量。

2) 采样电阻与放大倍数:

最大采集电流 = VREF/HW_RSHUNT/HW_AMPGAIN；这里要注意的是，最大采集电流不是电源上显示的电流(电源上显示的是滤波后的)，而是流经采样电阻的电流。

- 采样电阻不宜太大: 太大的话容易导致采样溢出，或者本身的功率超过范围；2512 封装的采样电阻常见功率为 1W 或者 2W，1206 封装电阻的功率常见位 1/4W，选择时，要注意流经采样电阻的功率 I^2R 不要超过该功率。
- 采样电阻不宜太小，太小的话精度不够
- 放大倍数结合采样电阻调整，先确定了采样电阻，再去调整放大倍数

其中，HW_RSHUNT 为采样电阻，HW_AMPGAIN 为放大倍数

3. 母线分压比、采样电阻值、放大倍数对应填写到程序中(在 Customer.h 文件)

Customer.h

```
/* ----------------------------------------------------------------------------------------------------
                                    硬件相关参数值配置
---------------------------------------------------------------------------------------------------- */

/*PWM high or low level Mode*/
/* ----------------------------------------------------------------------------------------------------
    PWM_Level_Mode :驱动有效电平配置
    High_Level     :上下桥驱动电平均高有效
    Low_Level      :上下桥驱动电平均低有效
    UP_H_DOWN_L    :上桥臂高电平有效，下桥臂低电平有效
    UP_L_DOWN_H    :上桥臂低电平有效，下桥臂高电平有效
---------------------------------------------------------------------------------------------------- */
#define PWM_Level_Mode                (UP_H_DOWN_L)

/*hardware current sample Parameter*/
/* ----------------------------------------------------------------------------------------------------
    AMP_MODE     :运放模式配置
    AMP_PGA_DUAL :运放PGA模式，放大倍数由内部配置
    AMP_NOMAL    :运放非PGA模式，放大倍数由外部配置
---------------------------------------------------------------------------------------------------- */
#define AMP_MODE                      (AMP_NOMAL)

/* ----------------------------------------------------------------------------------------------------
    HW_AMPGAIN_Choose     :运放放大倍数选择，运放PGA模式时可选 AMP2x/AMP4x/AMP8x/AMP16x，否则按实际放大倍数填写
    AMP2x                 :运放PGA模式X2倍
    AMP4x                 :运放PGA模式X4倍
    AMP8x                 :运放PGA模式X8倍
    AMP16x                :运放PGA模式X16倍
    其他数字              :运放非PGA模式，按实际放大倍数填写
---------------------------------------------------------------------------------------------------- */
#define HW_AMPGAIN_Choose             (5.0)                           // 运放放大倍数

/*运放采样电阻及放大倍数*/
#define HW_RSHUNT0                    (0.1)                           // (Ω)  AMP0端采样电阻
#define AMP0_VHALF                    (1)                             // AMP0是否接偏置电压,0,无偏置电压;1,有偏置电压

#define HW_RSHUNT                     (0.1)                           // (Ω)  AMP1、AMP2端采样电阻
```

Customer.h

```
/*VREF电压选择*/
/* ----------------------------------------------------------------------------------------------------
    VREF3_0     :3V
    VREF4_0     :4V
    VREF4_5     :4.5V
    VREF5_0     :5V
---------------------------------------------------------------------------------------------------- */
#define HW_ADC_VREF                   (VREF5_0)                       // (V)  ADC参考电压

/* ----------------------------------------------------------------------------------------------------
    VREF_OUT_EN :对于芯片P3.5引脚有引出来的，需强制配置为基准电压VREF对外输出使能；否则Disable！！！
---------------------------------------------------------------------------------------------------- */
#define VREF_OUT_EN                   (Disable)                       // 基准电压VREF对外输出使能

#define VHALF_OUT_EN                  (Disable)                       // VHALF输出使能

/*hardware voltage sample Parameter*/
/*母线电压采样分压电路参数*/
#define RV1                           (47.0)                          // (kΩ) 母线电压上分压电阻1
#define RV2                           (0.0)                           // (kΩ) 母线电压上分压电阻2
#define RV3                           (3.0)                           // (kΩ) 母线电压下分压电阻3
```

4.4 保护参数设置

1. 电流保护设置:
   - 硬件过流: 根据功率器件的最大电流值，设置硬件过流保护值，一般硬件过流保护值 OverHardcurrentValue 设置大于母线最大电流值，小于功率器件最大电流值。
   - 软件过流: OverSoftCurrentValue 一般设置比硬件过流小一点即可，需设置小于电机的退磁电流，软件过流为软件触发，保护时间不及硬件过流。
2. 设置过欠压保护以及保护恢复参数，详细设置参考章节 5.2.2；
3. 关闭上述保护的其他保护，防止启动的时候误触发，后面添加需要的保护再确认，其中过流保护是一定要开的，因此没有使能位；
4. 将参数对应填写到程序中(在 Protect.h 文件)。

Customer.h | Protect.h

```
#define HardwareCurrent_Protect        (Hardware_CMP_Protect)          // 硬件过流保护方式选择
#define OverHardcurrentValue           (4.5)                           // (A) DAC模式下的硬件过流值，不能>最大采样电流!!!

/*软件过流保护*/
#define OverSoftCurrentValue           I_Value(4.0)                    // (A) 软件过流值，不能>最大采样电流!!!

/*过流恢复*/
#define CurrentRecoverEnable           (1)                             // 过流保护恢复使能位，0，不使能；1，使能
#define OverCurrentRecoverTime         (3000)                          // (ms) 过流保护恢复时间
#define OVCurrentTimesRestartTimes     (5)                             // 软硬件过流保护重启次数，最大255，单位：次

/*偏置电压保护恢复*/
#define GetCurrentOffsetValue          _Q14(0.20)                      // (单位:100%)偏置电压保护误差范围，超过该范围保护
#define IbusOffsetRecoverEnable        (1)                             // 偏置电压保护恢复使能位，0，不使能；1，使能
#define IbusOffsetRecoverTime          (100)                           // (ms) 偏置电压保护恢复时间
#define IbusOffsetRestartTimes         (5)                             // 偏置电压保护重启次数，最大255，单位：次

/*过欠压保护*/
#define VoltageProtectEnable           (1)                             // 电压保护，0,不使能；1，使能
#define Over_Protect_Voltage           (46)                            // (V) 直流电压过压保护值
#define Over_Recover_Vlotage           (44)                            // (V) 直流电压过压保护恢复值
#define Under_Protect_Voltage          (28)                            // (V) 直流电压欠压保护值
#define Under_Recover_Vlotage          (30)                            // (V) 直流电压欠压保护恢复值

/*堵转保护*/
#define StallProtectEnable             (1)                             // 堵转保护，0,不使能；1，使能
#define StartRecoverTime               (20)                            // (ms) 启动保护延时重启时间
#define StallRecoverTime               (1000)                          // (ms) 堵转保护延时重启时间
#define StallProtectRestartTimes       (5)                             // 堵转保护重启次数，最大255，单位：次

#define EsThresholdValue0              (15.0)                          // (RPM) 方法1,ES<EsThresholdValue0 触发

#define EsThresholdValue1              (100.0)                         // (RPM) 方法1,估算转速>EsThresholdSpeed1且ES<EsThresholdValue1 触发
#define EsThresholdSpeed1              S_Value(3000)                   // (RPM) 方法1,估算转速>EsThresholdSpeed1且ES<EsThresholdValue1 触发

#define STALL_MAX_SPEED                S_Value(9000.0)                 // (RPM) 方法2,估算转速>MOTOR_SPEED_STAL_MAX_RPM 触发
#define STALL_MIN_SPEED                S_Value(200.0)                  // (RPM) 方法2,估算转速<MOTOR_SPEED_STAL_MIN_RPM 触发

/*缺相保护*/
```

4.5 启动参数配置

启动参数都先采用自带的**默认参数**，等启动有问题或者启动不顺的时候再做调整。启动常见的问题即参数调整可以参考章节 5.1。

Customer.h | Protect.h

```
/***启动电流****/
#define ID_Start_CURRENT               I_Value(0.0)                    // (A) D轴启动电流
#define IQ_Start_CURRENT               I_Value(1.0)                    // (A) Q轴第一拍启动电流
#define IQ_Start_CURRENT_END           I_Value(2.0)                    // (A) 切外环前Q轴最大限制电流

/***运行电流****/
#define ID_RUN_CURRENT                 I_Value(0.0)                    // (A) D轴运行电流
#define IQ_RUN_CURRENT                 I_Value(0.5)                    // (A) Q轴运行电流
```

1. 启动电流: 一般 ID_Start_CURRENT 固定设置为 0，IQ_Start_CURRENT 根据实际电机设置确认；
   注意事项:
   IQ_Start_CURRENT，为启动时的第一拍电流，不能过小否则启动力矩太小导致启动失败；也不能过大否则启动过冲还会引入启动噪声。

IQ_Start_CURRENT_END，为启动给第一拍电流后递增的最大电流，即切外环前 Q 轴最大限制电流，需要比 IQ_Start_CURRENT 大一些以克服启动阻力。

2. 运行电流: IQ_RUN_CURRENT 只决定顺风启动一瞬间的电流。通过实际观测相电流波形，可以适当调整 IQ_RUN_CURRENT 解决切换电流不平滑；
3. 启动 ATO: 由于在较低转速下估算器输出存在误差，此时需要设置 ATO_BW(速度带宽滤波值)，以限制 FOC 估算器的最大转速输出；

Customer.h | Protect.h

```
/********Omega启动的参数**********/
#define ATO_BW                  (0.0)                   // 观测器带宽的滤波值，经典值为1.0-200.0
#define ATO_BW_RUN              (10.0)
#define ATO_BW_RUN1             (25.0)
#define ATO_BW_RUN2             (35.0)
#define ATO_BW_RUN3             (45.0)
#define ATO_BW_RUN4             (55.0)
#define ATO_BW_RUN5             (80.0)
```

注意事项:

对于工业风机而言，启动的前 3 个 ATO 影响比较明显，需要根据实际情况调整。一般第一个 ATO_BW 直接给 0。

4. SMO 运行最小转速 MOTOR_SPEED_SMOMIN_RPM；

Customer.h | Protect.h

```

#define MOTOR_SPEED_SMOMIN_RPM      (150.0)                 // (RPM) SMO运行最小转速
```

注意事项:

MOTOR_SPEED_SMOMIN_RPM 需设置为小于最小运行转速值，对启动性能影响较大

5. Omega 启动参数设置，影响启动的电流频率，即电机的启动加速度。

Customer.h | Protect.h

```
/* motor start speed value */
//OMEGA启动参数
#define Motor_Omega_Ramp_ACC        (1)                     // omega启动的增量,需要给int型数据
#define MOTOR_OMEGA_ACC_MIN         (10.0)                  // (RPM) omega启动的最小切换转速
#define MOTOR_OMEGA_ACC_END         (100.0)                 // (RPM) omega启动的限制转速

/* motor loop control speed value */
#define MOTOR_LOOP_RPM              (150.0)                 // (RPM) 即闭环切换转速
```

注意事项:

1) Motor_Omega_Ramp_ACC 参考值范围 1 ~ 20
2) MOTOR_OMEGA_ACC_MIN 参考值范围 0 ~ 100
3) MOTOR_OMEGA_ACC_END 参考值范围 100 ~ 500
4) MOTOR_LOOP_RPM 需要大于 MOTOR_OMEGA_ACC_END，参考值范围 100 ~ 500

6. 电流环 PI: 电流环 PI 分启动的电流环 PI 跟运行时的电流环 PI；

Customer.h | Protect.h

```
/*电流环参数设置值-------------------------------------------------------------*/
#define DQKPStart                   _Q12(1.0)               // 启动时DQ轴电流环KP
#define DQKIStart                   _Q15(0.01)              // 启动时DQ轴电流环KI
```

Customer.h | Protect.h

```
#define DQKP                        _Q12(1.0)               // 运行时DQ轴KP
#define DQKI                        _Q15(0.01)              // 运行时DQ轴KI
```

注意事项:

1) 启动的电流环 PI，影响电机的启动；
2) 运行的电流环 PI，影响电流的稳定性，也影响效率；
3) DQKP 建议范围 0.1 ~ 3.0；
4) DQKI 建议范围 0.001 ~ 0.05。

7. DQ 轴最大输出限幅: D 轴影响电机的磁通，Q 轴影响电机的转矩；

Customer.h Protect.h

```
/* D轴参数设置 */
#define DOUTMAX                 _Q15(0.6)                   // D轴最大限幅值，单位：输出占空比
#define DOUTMIN                 _Q15(-0.6)                  // D轴最小限幅值，单位：输出占空比

/* Q轴参数设置，默认0.99即可 */
#define QOUTMAX                 _Q15(0.99)                  // Q轴最大限幅值，单位：输出占空比
#define QOUTMIN                 _Q15(-0.99)                 // Q轴最小限幅值，单位：输出占空比
```

注意事项:

1) FOC__UQ 反馈电机是否已经输出饱和；
2) FOC__UD 正得越多表示角度越超前，可以通过增加补偿角(FOC_THECOMP)让电机角度超前，此时能提升最大转速，FOC__UD 是一个正值；
3) 过多的超前角度，会导致关机时候电流过冲，可以通过低压预警关机处理，也可以通过快速欠压保护处理；
4) 过多的超前角度，会导致效率变差，相同功率下，相电流幅值更大，需要合理设置补偿角度。

4.6 硬件驱动电路检测

Customer.h Protect.h

```
/* ----------------------------------------------------------------------------------------------------
    IPMState    :MCU输出模式配置
    IPMtest     :IPM测试或者MOS测试，MCU输出固定占空比，主要验证板子输出是否正常
    NormalRun   :正常按电机状态机运行
---------------------------------------------------------------------------------------------------- */
#define IPMState                        (NormalRun)
```

Customer.h Protect.h

```
/* ----------------------------------------------------------------------------------------------------
    SPEED_MODE      :调速模式选择
    NONEMODE        :直接给定转速，不调速
    PWMMODE         :Duty调速，需初始化对应定时器
    SREFMODE        :模拟调速，需使能对应AD口
    UARTMODE        :UART调速
---------------------------------------------------------------------------------------------------- */
#define SPEED_MODE                      (SREFMODE)
```

将IPMState设置为IPMtest，SPEED_MODE选择NONEMODE开机，不接电机，若UVW三相有固定的PWM波形输出，则硬件驱动电路正常，否则需要查找硬件问题。

4.7 调试电流环

1. 将闭环方式选择为 IQ_LOOP_CONTROL；

```
Customer.h  Protect.h
/* ------------------------------------------------------------------------------------------------------
                                    调速相关参数值配置
------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------
    Motor_Speed_Control_Mode    :闭环方式选择
    IQ_LOOP_CONTROL             :恒相电流
    POWER_LOOP_CONTROL          :功率环
    SPEED_LOOP_CONTROL          :速度环
    UQ_LOOP_CONTROL             :电压环
    CURRENT_LOOP_CONTROL        :恒母线电流
------------------------------------------------------------------------------------------------------ */
#define Motor_Speed_Control_Mode        (IQ_LOOP_CONTROL)
```

2. 调速方式先选为直接给定值，调整给定值 Motor_Max_IQ 的大小，以此来控制电流环的电流大小(注意给的是相电流的值，而且因为选的调速方式是直接给定的，程序只认 Motor_Max_IQ，此时 Motor_Min_IQ 是无效的)；

```
Customer.h  Protect.h
/* ------------------------------------------------------------------------------------------------------
    SPEED_MODE      :调速模式选择
    NONEMODE        :直接给定转速，不调速
    PWMMODE         :Duty调速，需初始化对应定时器
    SREFMODE        :模拟调速，需使能对应AD口
    UARTMODE        :UART调速
------------------------------------------------------------------------------------------------------ */
#define SPEED_MODE                      (NONEMODE)
```

```
Customer.h  Protect.h
#define Motor_Max_IQ                I_Value(3.0)                //Q轴电流最大值,恒相电流控制时配置
#define Motor_Min_IQ                I_Value(0.15)               //Q轴电流最小值,恒相电流控制时配置
```

3. 烧录程序，上电启动电机，当电机启动不起来时(目前一般都能起来)，通过调整以下启动参数:
   - 启动电流: IQ_Start_CURRENT，电流不够时电机起不来，可以慢慢增加，也不要一次性给太大
   - 影响启动频率的 ATO、Omega 和 MOTOR_SPEED_SMOMIN_RPM 的参数等等
4. 当上电，电机能跑后，加大电流环给定值，达到客户目标功率。
5. 确认电流环情况下最大功率、转速
6. 记录最大功率下的 Q 轴输出 duty 值 mcFocCtrl.UqFlt(该值即是电压环时的给定最大参考值)，以及此时设置的相电流 Motor_Max_IQ 大小(该值可作为外环 SOUTMAX 的参考值)。

注: 母线电流采集的 AD 口要对应上，要根据实际硬件电路去修改。具体位置如下图

```
Customer.h  Protect.h  AddFunction.c
#if ((Current_LIMIT_ENABLE) || (Motor_Speed_Control_Mode == CURRENT_LOOP_CONTROL))
{
    /*****RC母线电流采样，限流或恒母线电流用******/
    AdcSampleValue.ADCIbus  = ADC5_DR << 3;
    if(AdcSampleValue.ADCIbus > mcCurOffset.Iw_busOffset)
    {
        AdcSampleValue.ADCIbus   = AdcSampleValue.ADCIbus - mcCurOffset.Iw_busOffset;
    }
    else
    {
        AdcSampleValue.ADCIbus   = 0;
    }
    mcFocCtrl.mcIbusFlt  = LPFFunction(AdcSampleValue.ADCIbus,mcFocCtrl.mcIbusFlt,50);
}
#endif
```

常见问题及解决办法:

1. 加大电流给定，还是达不到客户要的最大功率值；

   解决: 电流波形正弦的情况下，通过观测 FOC__UQ 是否饱和，如果饱和，且 FOC__UD 值比较大的话，通过调整补偿角 FOC_THECOMP(正负都调整看看)确认是否能达到客户需求。

2. 运行过程中，触发过流保护；

   解决: 看相电流波形是否异常，看是否是设定值比较小正常触发了过流保护。如果没异常的情况下，查看硬件布线等是否有问题。

3. 相电流波形有抖动；

   解决: 调整电流环 PI(即 DQKP，DQKI)的值，电流环 PI 和电流采样对于电流波形的稳定性影响比较大。

4.8 增加外部环路

1. 一般工业风机都是用的电压环，因此将环路选择为电压环；

```
Customer.h | Protect.h | AddFunction.c
297 /* ------------------------------------------------------------------------------------------------------
298     Motor_Speed_Control_Mode    :闭环方式选择
299     IQ_LOOP_CONTROL             :恒相电流
300     POWER_LOOP_CONTROL          :功率环
301     SPEED_LOOP_CONTROL          :速度环
302     UQ_LOOP_CONTROL             :电压环
303     CURRENT_LOOP_CONTROL        :恒母线电流
304 ------------------------------------------------------------------------------------------------------ */
305 #define Motor_Speed_Control_Mode        (UQ_LOOP_CONTROL)
```

2. 设置调速曲线的最大值，跟 SOUTMAX 的值，这两个值在章节 4.7 最后一步已经有记录的参考值，其中 SOUTMAX 需要在记录的参考值的基础上再增加一点，以防电压上升时电流还要进一步上升，要有足够的上升空间。Motor_Min_UQ 为曲线的最小输出 duty，可根据客户实际需要设定；

```
Customer.h | Protect.h | AddFunction.c
266 #define SOUTMAX                 I_Value(3.0)            // (A) 外环PI输出最大限幅值
267 #define SOUTMIN                 I_Value(0.01)           // (A) 外环PI输出最小限幅值

Customer.h | Protect.h | AddFunction.c
313 #define Motor_Min_UQ            _Q15(0.1)               //最小运行占空比,电压环时配置
314 #define Motor_Max_UQ            QOUTMAX                 //最大运行占空比,电压环时配置
```

3. 通过调整电压环 PI(SKP 和 SKI)和电压环爬坡增量保证环路稳定，启动响应快且不过冲；

```
Customer.h | Protect.h | AddFunction.c
263 #define SKP                     _Q12(0.4)               // 外环KP
264 #define SKI                     _Q12(0.01)              // 外环KI

Customer.h | Protect.h | AddFunction.c
269 #define SpeedRampStart          (50.0)                  // 档位切换时加速增量
270 #define SpeedRampEnd            (2.0)                   // 稳定运行时加速增量
```

说明:

1) SpeedRampStart: 加减速过程中环路未稳定时的爬坡增减量
2) SpeedRampEnd: 环路稳定时的爬坡增减量

4.9 增加调速等功能

1. 一般工业风机为 PWM 或 VSP 调速，以 PWM 调速为例:
   1) 将调速模式修改为 PWM 调速；

```
Customer.h | Protect.h | AddFunction.c
/* ------------------------------------------------------------------------------------------------------
    SPEED_MODE      :调速模式选择
    NONEMODE        :直接给定转速，不调速
    PWMMODE         :Duty调速，需初始化对应定时器
    SREFMODE        :模拟调速，需使能对应AD口
    UARTMODE        :UART调速
------------------------------------------------------------------------------------------------------ */
#define SPEED_MODE                      (PWMMODE)
```

2) 根据客户给的 PWM 曲线，调整开关机、最大最小占空比；

```
/*PWM调速模式下电机开关机的设置-------------------------------------------------------------*/
/* motor ON/OFF value */
#define OFFPWMDuty                  _Q15(0.08)          // 关机PWM占空比，小于该占空比关机
#define ONPWMDuty                   _Q15(0.1)           // 开机PWM占空比，大于该占空比时开机
#define MINPWMDuty                  _Q15(0.1)           // 调速曲线上最小PWM占空比
#define MAXPWMDuty                  _Q15(0.98)          // 调速曲线上最大PWM占空比
```

3) 根据客户给的曲线，调整最大最小运行占空比；

```
#define Motor_Min_UQ                _Q15(0.1)           //最小运行占空比,电压环时配置
#define Motor_Max_UQ                QOUTMAX             //最大运行占空比,电压环时配置
```

得到的曲线最低点为(MINPWMDuty，Motor_Min_UQ)，最高点为(MAXPWMDuty，Motor_Max_UQ)

4) 确认客户是正 PWM 调速还是负 PWM 调速：正 PWM 调速：转速随着占空比增大而增大；负 PWM 调速：转速随 PWM 增大而减小；

```
/* ------------------------------------------------------------------------------------------------------
    PWMDUTY_Choose :PWM正负反馈选择
    PosiPWMDUTY    :正PWMduty调速
    NegaPWMDUTY    :负PWMduty调速
------------------------------------------------------------------------------------------------------ */
#define PWMDUTY_Choose                  (PosiPWMDUTY)
```

注意事项:

1. 根据 PWM 频率，在 Timer3 初始化的时候，选择合理的 Timer 分频；
2. 开关机占空比，要留有一定的滞回区间，如 10%开机，8%关机。留 2%的滞回区间。开机跟关机占空比如果一样的话，会导致时开时关；
3. 当 PWM 占空比获取不对时，看进入芯片引脚的 PWM 信号是否已经失真，如果滤波电容太大的话，会导致 PWM 信号失真；
4. PWM 信号有干扰的，尝试打开捕获 TIM 口的滤波功能，或者调整 PWM 硬件滤波电容，尽量靠近芯片引脚。

2. FG 输出方式配置；

```
Customer.h | Protect.h | AddFunction.c
/* ------------------------------------------------------------------------------------------------------
    FG_MODE :FG输出方式配置
    DISABLE_FG_OUTPUT     :不反馈FG
    SOFT_TIMFG_OUTPUT     :通过手动配置定时器输出FG信号,输出引脚需接对应定时器
    HARD_TIMFG_OUTPUT     :通过 TIMER4 输出FG信号,输出引脚需接对应定时器
    THETA_FG_OUTPUT       :通过角度输出FG信号,输出引脚接GPIO即可
------------------------------------------------------------------------------------------------------ */
#define FG_MODE                         (SOFT_TIMFG_OUTPUT)
#define FPin                            (GP11)                  //FG输出引脚
#define FG_K                            (1.0)                   //一个电周期输出脉冲数
```

注意事项:

对于 FG 反馈稳定性要求较高时，需配合硬件用定时器方式输出，可选 SOFT_TIMFG_OUTPUT 或 HARD_TIMFG_OUTPUT 两种方式；否则可选普通的 GPIO 输出方式 THETA_FG_OUTPUT

3. 添加保护功能，根据客户需求使能启动保护、堵转保护、缺相保护、过欠压保护等。所有其他程序中还没添加的保护，则要额外再添加。具体保护介绍参考章节 5.2。

4.10 可靠性测试

4.10.1 功能可靠性

全部功能添加完成后，要再按照客户需求表重新测试确保没异常状态发生。

4.10.2 保护可靠性

保护添加之后，要验证保护都可以正常触发，且在电机运行时不会误触发。例如：如果堵转保护的参数设置不合理，可能会导致电机在正常运行时也会误报堵转保护；或者是电机发生堵转后，不会触发堵转保护。

4.10.3 启动稳定性

在功能都基本调试完成之后，要做启动的可靠性测试，可先手动测试，手动测试没问题后，再进行老化测试。老化测试步骤:

1. 调速模式选择为 ONOFFTEST；

Customer.h

```
/* ------------------------------------------------------------------------------------------------------
    SPEED_MODE      :调速模式选择
    NONEMODE        :直接给定转速，不调速
    PWMMODE         :Duty调速，需初始化对应定时器
    SREFMODE        :模拟调速，需使能对应AD口
    UARTMODE        :UART调速
    ONOFFTEST       :启停测试工具
    OTHERS          :其他调速方式
------------------------------------------------------------------------------------------------------ */
#define SPEED_MODE                  (ONOFFTEST)
```

2. 根据实际情况配置运行时间 StartON_Time 和停止时间 StartOFF_Time；调整 ONOFFTEST_SPEED 的值可以修改启停的转速大小；

Customer.h

```
/******启停测试参数******/
#define StartON_Time                (6500)          // (ms) 启动运行时间
#define StartOFF_Time               (1000)          // (ms) 停止时间
#define ONOFFTEST_SPEED             (300)           //启停测试设置目标转速
```

3. 先用工具堵住电机上电，看是否能正常触发堵转保护，且保护后电机不会重启，即验证了启停时如果触发保护电机不会二次重启；
4. 再次上电进行老化测试即可。最后根据电机是否处于停止状态判断启动是否有异常，启动失败后，电机会一直停机不再重启。一般测试 3000 次以上没问题则认为启动可靠(时间允许的情况下越多越好)。

5 功能介绍

目前拿到初始版本程序，配置好电机参数，硬件参数后，给开机信号时，电机基本都能正常启动。若不能正常启动，则在排除是硬件问题的前提下，再调整启动参数。

5.1 启动调试

5.1.1 Omega 启动

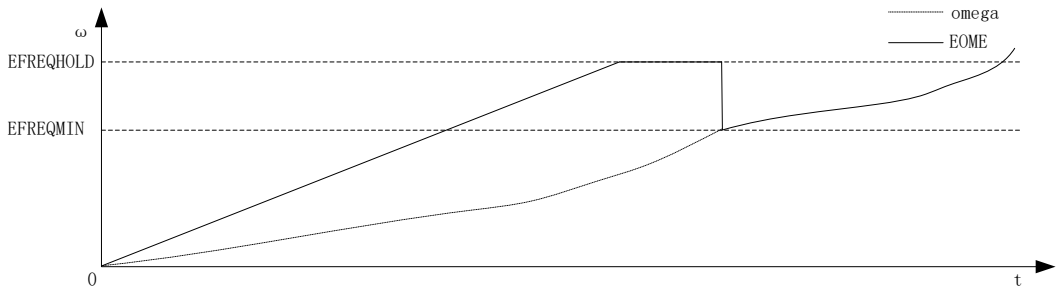
工业风机选择 Omega 启动，程序对应默认即是该启动方式。

```
Customer.h
227 /* ------------------------------------------------------------------------------------------------------
228     Open_Start_Mode     :开环启动模式选择，默认用Omega_Start即可
229     Open_Start          :开环强拖启动
230     Omega_Start         :Omega启动
231     Open_Omega_Start    :先开环启，后Omega启动
232 ------------------------------------------------------------------------------------------------------ */
233 #define Open_Start_Mode                 (Omega_Start)
```

当估算器的估算速度 OMEGA 小于用户设定的最小值 FOC_EFREQMIN(对应 MOTOR_OMEGA_ACC_MIN 参数)，强制速度从 0 开始，每个运算周期与速度增量 FOC_EFREQACC (Motor_Omega_Ramp_ACC)参数相加，同时根据 FOC_EFREQHOLD (MOTOR_OMEGA_ACC_END 参数)进行最大值限幅，输出强制速度作为最终速度 EOME 供角度计算模块算出估算器角度 ETHETA；当估算器的估算速度 OMEGA 大于等于 EFREQMIN 时，输出估算速度 OMEGA 作为最终速度 EOME。

```
Customer.h
215 /* motor start speed value */
216 //OMEGA启动参数
217 #define Motor_Omega_Ramp_ACC        (1)             // omega启动的增量,需要给int型数据
218 #define MOTOR_OMEGA_ACC_MIN         (10.0)          // (RPM) omega启动的最小切换转速
219 #define MOTOR_OMEGA_ACC_END         (100.0)         // (RPM) omega启动的限制转速
```

启动过程如下图所示:



5.1.2 启动常见问题&解决方式

常见问题	解决方法
电机动一下后静止，且一直有电流输出	1.启动电流太小电机输出无法切换到下一拍换相，增大 IQ_Start_CURRENT； 2. 估算器输出速度太小无法换相到下一拍，若排除 A 后此时依次加大 ATO_BW、ATO_BW_RUN、ATO_BW_RUN1、ATO_BW_RUN2； 3.排除 A、B 后，检查硬件电路运放 AMP0 部分是否有问题导致电流采样不准，估算器估算不正常。 4.也有可能是 omega 加速度的频率太高导致，可以减小 Motor_Omega_Ramp_ACC
电机转一下后停下且一直抖动	1.此种情况一般为 ATO_BW 值太大，导致估算器输出转速较高，启动时失步，此时依次减小 ATO_BW、ATO_BW_RUN、ATO_BW_RUN1、ATO_BW_RUN2。 2. Omega 启动参数也会有影响。
启动正序旋转一定角度后卡顿一下再正常旋转	1.此时可估计从启动到出现卡顿的时间，再设置对应时间的 ATO_BW 值，例如：启动 1s 后电机卡顿一下然后继续正常运行，1s 时间对应的值为 ATO_BW_RUN1、ATO_BW_RUN2，此时该 ATO_BW 值较小限制了电机加速，相应加大该值即可。 2.omega 加速度太小时也会造成卡顿的情况，可以加大 Motor_Omega_Ramp_ACC
电机启动反转后正转时持续抖动	1.电机启动反转一下后正转所需要的时间较长，此时 ATO_BW 已加到比较大的值，因此减小相应时间的 ATO_BW 值即可。 2.也有可能是 omega 加速度的频率太高导致，可以修改 Motor_Omega_Ramp_ACC

5.2 保护介绍

每个项目，不同电机，不同板子的保护值都会有所不同，各种保护的保护值都要根据实际项目去匹配。当发现保护，特别是堵转保护或缺相保护触发不了，或者正常运行时，误触发保护时，说明是保护设定值不合理导致的，此时要调整保护的设定值。

5.2.1 过流保护

1. 硬件过流保护

芯片通过比较器 3 做硬件过流保护，检测方法: 母线电流流经采样电阻，在采样电阻上形成一个电压，这个电压经过运算放大器放大送入比较器的正向输入端。比较器的负向输入端会被设置一个参考电压,这个参考电压可选择 DAC 产生或者由外部分压得到(目前都是用的 DAC 产生)。当母线电流增大到一定数值之后，就会导致比较器的正向输入端的电压高于负向输入端电压，这个时候就会触发 MCU 的比较器中断，MCU 发生中断并自动关闭 MOE(可选择自动或者不自动关闭 MOE，目前默认都是自动关闭 MOE)，从而完成过流保护。硬件过流保护只需要修改保护值 OverHardcurrentValue 的大小即可。

Interrupt.c | Customer.h* | AddFunction.c | Parameter.h | Protect.h

```
18
19  #define HardwareCurrent_Protect        (Hardware_CMP_Protect)              // 硬件过流保护方式选择
20  #define OverHardcurrentValue           (4.5)                               // (A) DAC模式下的硬件过流值，不能>最大采样电流!!!
```

2. 软件过流保护

程序通过获取三相电流值，当相电流值超过设定的软件过流保护值 OverSoftCurrentValue 时，则计一次；计数 3 次后触发保护。

Interrupt.c | Customer.h* | AddFunction.c | Parameter.h | Protect.h | MotorProtect.c | FU68xx_2_MCU.h

```
21
22  /*软件过流保护*/
23  #define OverSoftCurrentValue           I_Value(4.0)                        // (A) 软件过流值，不能>最大采样电流!!!
```

5.2.2 电压保护

程序通过AD2口检测电压，当检测到的电压超过设定值时，则报过压保护；此时当电压重新低于过压恢复值时，清除过压保护故障。当电压低于设定的欠压值时，则报欠压保护。此时当电压重新高于欠压恢复值时，清除欠压保护故障。

Customer.h | Protect.h

```
36  /*过欠压保护*/
37  #define VoltageProtectEnable           (1)                                 // 电压保护，0,不使能；1，使能
38  #define Over_Protect_Voltage           (46)                                // (V) 直流电压过压保护值
39  #define Over_Recover_Vlotage           (44)                                // (V) 直流电压过压保护恢复值
40  #define Under_Protect_Voltage          (28)                                // (V) 直流电压欠压保护值
41  #define Under_Recover_Vlotage          (30)                                // (V) 直流电压欠压保护恢复值
```

5.2.3 缺相保护

电机发生缺相时，三相电流是不对称的。因此可以通过在程序中检测一定时间内的三相电流的最大值，判断三相电流的最大值是否有不对称的情况来实现缺相保护。

具体程序实现方法：若检测到其中一相的最大电流大于另一相最大电流的3倍，且该相最大电流大于设定的PhaseLossCurrentValue值，则判定为缺相。

Customer.h | Protect.h

```
57  /*缺相保护*/
58  #define PhaseLossProtectEnable         (1)                                 // 缺相保护，0,不使能；1，使能
59  #define PhaseLossCurrentValue          I_Value(0.05)                       // (A)  缺相判断电流阈值,注意在最低运行电流下是否会误触发!!!
60  #define PhaseLossRecoverTime           (3000)                              // (ms) 缺相保护延时重启时间
61  #define PhaseLossRestartTimes          (5)                                 // 缺相保护重启次数，最大255，单位：次
```

注意事项：

有些方案在缺相时，由于缺的那一相会有毛刺的存在，可能会导致采集的最大电流值跟另外两相差不多，这时候通过上述方法可能检测不出来。解决方法：可以通过积分的方式，在一定时间内通过去比较电流累计值的大小去判断缺相。

5.2.4 堵转保护

堵转保护有4种方法检测:

通过检测估算器计算出来的FOC_ESQU(估算器计算的反电动势的平方)判断，正常情况下，电机转速越高，FOC_ESQU会越大。在电机发生堵转时，电机失步的情况下，估算转速会很高，但是FOC_ESQU会很小，因此可以通过该方式判断。

具体程序实现方法:

1. 方法一，运行1s内，当估算转速 > 设定值STALL_MAX_SPEED或估算转速 > 设定值EsThresholdSpeed1且FOC_ESQU的值 < 设定值EsThresholdValue1时，触发堵转保护，延时StartRecoverTime时间重启；

Customer.h | Protect.h | MotorProtect.c

```
182
183   //方法一，判断转速太大或当转速太大反电动势太小，超过一定次数后则认为堵转
184   if ((mcFocCtrl.SpeedFlt > STALL_MAX_SPEED)
185       || ((mcFocCtrl.SpeedFlt > EsThresholdSpeed1) && (mcFocCtrl.EsValue < EsThresholdValue1)))
186   {
187       mcFaultDect.StallDectESSpeed++;
188       if (mcFaultDect.StallDectESSpeed > 50)
189       {
190           mcFaultDect.StallDectESSpeed = 0;
191           mcProtectTime.StallFlag      =  1;
192           mcFaultSource                = FaultStall;
193       }
194   }
195   else
196   {
197       if (mcFaultDect.StallDectESSpeed)
198       {
199           mcFaultDect.StallDectESSpeed--;
200       }
201   }
202
203   mcFaultDect.StallReTime = StartRecoverTime;                    //启动恢复时间
```

2. 方法二，运行1s后，当FOC_ESQU的值 < 设定值EsThresholdValue0或估算转速 > 设定值EsThresholdSpeed1且FOC_ESQU的值 < 设定值EsThresholdValue1时，触发堵转保护，延时StallRecoverTime时间重启

Customer.h | Protect.h | MotorProtect.c

```
131   if(Time.RunStateCnt > 1000)
132   {
133       //方法二，判断反电动势Es太小或当转速太大反电动势太小，超过一定次数后则认为堵转
134       if ((mcFocCtrl.EsValue < EsThresholdValue0)
135           || ((FOC__EOME > EsThresholdSpeed1) && (mcFocCtrl.EsValue < EsThresholdValue1)))
136       {
137           mcFaultDect.StallDectEs++;
138           if (mcFaultDect.StallDectEs >= 50)
139           {
140               mcFaultDect.StallDectEs  = 0;
141               mcProtectTime.StallFlag  =  2;
142               mcFaultSource            = FaultStall;
143           }
144       }
145       else
146       {
147           if (mcFaultDect.StallDectEs)
148           {
149               mcFaultDect.StallDectEs--;
150           }
151       }
```

3. 方法三，运行1s后，当估算转速 < 设定值STALL_MIN_SPEED或估算转速 > 设定值STALL_MAX_SPEED时，触发堵转保护，延时StallRecoverTime时间重启
4. 方法四，运行1s后，仍然未切闭环，触发堵转保护，延时StallRecoverTime时间重启

```
Customer.h | Protect.h | MotorProtect.c
172     //方法四，长时间没闭环，则认为启动失败
173     if(mcFocCtrl.CtrlMode < 1)
174     {
175         mcProtectTime.StallFlag  =  4;
176         mcFaultSource            = FaultStall;
177     }
178
179     mcFaultDect.StallReTime = StallRecoverTime;                    //堵转恢复时间
```

5.2.5 偏置电压保护

电机开始之前，会先采集偏置电压，有接 VHALF 时，偏置电压采集值理论上为 2048，左移 3 位后为 16383 左右；没接 VHALF 时，理论值为 0；当采集的值 ± 超过理论值的百分比 GetCurrentOffsetValue 时，则认为偏置电压异常。其中，0.20 代表 20%。

```
Customer.h | Protect.h | MotorProtect.c
30  /*偏置电压保护恢复*/
31  #define GetCurrentOffsetValue          _Q14(0.20)          // (单位:100%)偏置电压保护误差范围，超过该范围保护
32  #define IbusOffsetRecoverEnable        (1)                 // 偏置电压保护恢复使能位，0，不使能；1，使能
33  #define IbusOffsetRecoverTime          (100)               // (ms) 偏置电压保护恢复时间
34  #define IbusOffsetRestartTimes         (5)                 // 偏置电压保护重启次数，最大255，单位：次
```

5.2.6 其他保护

根据客户需求自行添加其他保护。

6 其他常见功能调试

6.1 限功率功能

使用电压环控制时，当工业风机全速运行且负载较重时，母线电流可能较大而容易把电源拉复位，故需要使用限功率功能对功率进行限定，

限功率功能目前有 3 种方式:

1. 通过对目标值限制实现，当检测功率超过保护阈值之后，在爬坡函数中对目标值限制从而达到限功率，此方法容易发生震荡故不做详细说明
2. 通过切换不同闭环实现，当全速运行且负载较重的时候，检测到功率超过限制值，那么程序会切入功率闭环从而达到限功率功能，当负载较轻时转速会超过目标值，此时切回电压闭环，从而实现了负载变化时限功率的功能。此方法需要调节电压环 PI 和功率环 PI 以及 PI 响应周期，而且切环过程容易出现震荡，故不做详细说明
3. 双 PI 的方式限制转速，硬件 PI 实现电压闭环，软件 PI 实现功率限制，软件 PI 输出限制 FOC_QMAX, 代码如下:

Customer.h | Protect.h | MotorProtect.c | AddFunction.c

```
522 #if (SPEED_LIMIT_ENABLE)//限速
523 {
524     FOC_QMAX = PIDControl(&SpeedPID, Motor_Limit_Speed, mcFocCtrl.SpeedFlt);
525 }
526 #elif (POWER_LIMIT_ENABLE)//限功率
527 {
528     FOC_QMAX = PIDControl(&SpeedPID, MOTOR_LIMIT_POWER, mcFocCtrl.Powerlpf);
529 }
530 #endif
```

目前程序已经添加了限速限功率功能，可直接使用。

Customer.h | Protect.h | MotorProtect.c

```
272 //限制环路相关参数
273 #define LimitLoopKp                 _Q12(0.7)
274 #define LimitLoopKi                 _Q12(0.01)
275
276 /******限速限功率调节参数******/
277 #define SPEED_LIMIT_ENABLE          (Disable)          //限速使能,SPEED_LIMIT_ENABLE/POWER_LIMIT_ENABLE只能二选一!
278 #define MOTOR_SPEED_LIMIT_RPM       (6800.0)           //(RPM) 限制转速
279
280 #define POWER_LIMIT_ENABLE          (Enable)           //限功率使能,SPEED_LIMIT_ENABLE/POWER_LIMIT_ENABLE只能二选一!
281 #define MOTOR_LIMIT_POWER           (15000)            //功率上限
```

6.2 限流功能

母线限流使能时，需要打开对应的 AD，如下

Customer.h | Protect.h | MotorProtect.c | AddFunction.c

```
1309    #if ((Current_LIMIT_ENABLE) || (Motor_Speed_Control_Mode == CURRENT_LOOP_CONTROL))
1310    {
1311        /*****RC母线电流采样，限流或恒母线电流用******/
1312        AdcSampleValue.ADCIbus  = ADC5_DR << 3;
1313        if(AdcSampleValue.ADCIbus > mcCurOffset.Iw_busOffset)
1314        {
1315            AdcSampleValue.ADCIbus   = AdcSampleValue.ADCIbus - mcCurOffset.Iw_busOffset;
1316        }
1317        else
1318        {
1319            AdcSampleValue.ADCIbus   = 0;
1320        }
1321        mcFocCtrl.mcIbusFlt  = LPFFunction(AdcSampleValue.ADCIbus,mcFocCtrl.mcIbusFlt,50);
1322    }
1323    #endif
```

Customer.h | Protect.h | MotorProtect.c | AddFunction.c

```
283  //母线限流使能
284  #define Current_LIMIT_ENABLE          (Disable)                          //母线限流使能
285  #define LIMITCurrent                  I_Value(2.5)                       //限流值
286  #define LIMIT_CURRENTRecover          (LIMITCurrent- I_Value(0.02))      //最大恢复值
287  #define LimitCurrentDec               (30)                               //限流时PI调节量
```

7 方案调试难点&解决方法

电压环调试	
常见问题	**解决方法**
启动一直有异常	启动一直调试不好，软件问题排除后，可以查看硬件采样布线等是否有问题。
顺风启动有异常，一直检测不准	查看硬件反电动势检测电路部分的地线是否布置合理，一般检测不准大都是地线等不合理有干扰导致。
电机速度响应较慢	1. 调试外环的 SKP,SKP； 2. 调节时间 LOOP_TIME； 3. 如果只是加减速比较慢，就调节加减速的增减量
在堵住吸风口后，快速放开，这时候电流会突然变很大导致硬件过流	一般是由于内环电流环没响应过来导致，可以加大电流环的 PI，即 DQKP，跟 DQKI
转速或者功率达不到客户要求	1. 电流波形正弦的情况下，通过观测 FOC__UQ 是否饱和； 2. FOC_UQ 饱和，且 FOC__UD 值比较大的话，通过调整补偿角 FOC_THECOMP(正负都调整看看)确认是否能达到客户需求； 3. 以上还是达不到要求时，确认是电机问题时，可让客户直接修改电机。
电机运行到高转速后容易出现大电流的情况	1. 调整补偿角 FOC_THECOMP； 2. 挪一下采样点，即修改采样点延时时间 FOC_TRGDLY
电流波形存在正弦度失真	1. 看采样偏置基准是否正常； 2. 修改电流环的 PI，即 DQKP,DQKI； 3. 修改采样点延时时间 FOC_TRGDLY； 4. 修改载波频率(注意修改后会影响启动跟运行)。
注意事项：修改参数后，基本都会影响启动和运行性能，解决好问题后要重新测试确认。	

8 修改记录

版本号	修改详细内容说明	生效日期	修订者
V1.0	初稿	2022/09/27	汤伟
V1.1	修订格式	2024/01/26	付倩雯

版权说明

峰岹科技（深圳）股份有限公司
深圳市南山区科技中二路深圳软件园二期 11 栋 2 楼 203
邮编：518057
电话：0755-26867710
传真：0755-26867715
网址：www.fortiortech.com